International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246.IJETCSIT-V6I4P104
Eureka Vision Publication | Volume 6, Issue 4, 24-29, 2025

Original Article

Cloud Microservices for Secure Health Data Access

Harshith Kumar Pedarla[1], Devika Jagarlamudi[2]
[1] Software Developer, Amazon, Seattle, USA.
[2] Product Manager, CurerTech, Chicago, USA.

Received On: 24/08/2025 ***Revised On: 29/09/2025*** ***Accepted On: 05/10/2025*** ***Published On: 11/10/2025***

Abstract *- Wellness systems are rapidly adopting cloud-native designs to increase the proportion of scalable, standardized, and high-performance applications. This enables them to quickly deploy applications that allow patients and employees to integrate multiple services or data pipes through well-defined functions easily. Computed using the ideas and principles that emerged as a result of the First World War, life-prolonging electronic medical interventions have been on the rise. However, this might also pose a significant danger to the patient's private data. For that reason, this paper investigates the several patterns and mechanisms that should be installed in microservices to guarantee that the data is safe from outside manipulations. The regulatory requirements (HIPAA), the most appropriate API software used throughout the industry of cloud management systems (HCR), the service level standards (FHIR /SMART), how the services keep their messages secret (service mesh, mutual TLS), and finally, the security practice quality ideas (in transit and at rest encryption, trust certificate placement) have all been reviewed and compiled in this report. A step-by-step procedure to start implementing this method to secure a health-based cloud service has been included, along with the relevant proposed assessment. Furthermore, regarding the practicality of adhering to the prescribed guidelines, it's essential to consider additional details alongside container management.*

Keywords *- Cloud Microservices, Health Data Security, Electronic Health Records (EHR), Zero-Trust Architecture, HIPAA Compliance.*

1. Introduction

It is unavoidable that architects supporting the digital transformation of the health care industry are tasked with building modular, scalable, and interoperable architectures. These requirements align with the principles supporting microservices, which are designed as small, function-focused facilities with publicized APIs, typically used as RESTful services. This aligns with the operational capacity of Electronic Health Records (EHR) and telemedicine. Microservices are also applied in clinical decision support as they encapsulate the general purpose of clinical algorithms, decision rules, and other support applications that might be subject to frequent change and regulatory requirements [1].

However, dealing with electronic Personal Health Information (ePHI), as echoed by HIPAA, healthcare architects should employ microservices along with strong technical and administrative safeguards. Cloud providers offer managed services such as KMS, IAM, or audit logging. While these can help secure the architectures, if not configured correctly or there is a lack of controls, then one could expose their data to mistakes and wrongly provide access to their data. Data exposure, unauthorized access, noncompliance, and data breaches are among the risks that can be misused or abused by the systems. This dissertation explores the design of secure, auditable access to health data in microservices-native clouds, focusing on the delivery of microservices' value. Primary legal and technical points will guide the recommendations across the board [1].

2. Background and Literature Review

2.1. Microservices in Healthcare

By refactoring monolithic applications into domain-specific units that are connected on well-defined APIs, microservices make these units in the health industry for good; some of these units include (patient-profile, scheduling, billing, imaging). Healthcare services are becoming more independent, with upgrades that are easy to scale and adoptable by other companies. For instance, it specializes in managing patients' health records, making compatibility between the two a prerequisite. Monitoring calls between doctors and patients should ideally be conducted in a professional video setting for enhanced examination [2].

2.2. Regulatory and Standards Context

According to Boda & Immaneni, (2021). The imposition of administrative/technical/physical safeguards will allow to logically extend all the required security measures for a cloud deployment of ePHI by implementing the necessary BAAs and to guarantee, that security measures and safeguards are up to the mark; cooperation between covered entities and cloud service providers have been outlined by our guidance on cloud computing issued by the U.S. HHS [2].

Standardization of fast Healthcare Interoperability Resources (FHIR) and smart Ontology for Health Record (SMART) on FHIR, offers a group of control for the use of application programming interfaces (APIs), as well as the management of access allowing and authentication; Modern health applications necessitate the usage of APIs and a access-control management process, as they offer practical features to ensure security standards. The standards and

regulatory frameworks discussed in the RCSA annotation serve the core objective of ensuring that all microservices created in conjunction with health data are laudably secure [3].

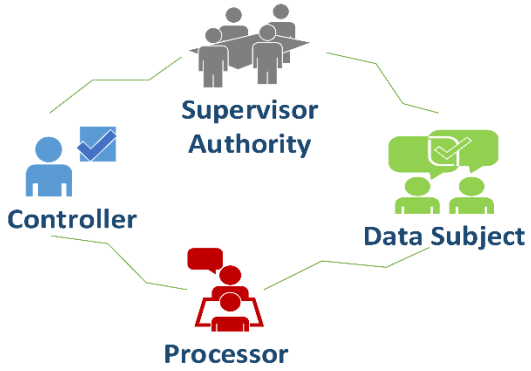

Figure 1. Compatibility of Security Policy for a Cloud Based Healthcare System with the EU General Data

2.3. Platform-Level Security Technologies

Microservice communication and policy operationalizing, such as Istio, primarily utilize service mesh technologies (e.g., Istio) and container orchestration (Kubernetes). We can enforce compliance policies for mutual (mTLS), identity, and observability using these services meshes technologies. Service meshes can enforce encryption, authentication, authorization, and telemetry without requiring changes to application code, addressing a common need in large, intrinsically distributed, and complex healthcare systems. Best practices for safely leveraging microservice communication and Istio deployments can be found by referencing Kubernetes hardening and zero trust guidance from the NIST and CIS Authorities [3].

3. Threat Models and Security Objectives

3.1. Threat Model

The primary concerns of microservices for managing ePHI in the cloud are:

- Unauthorized intake, including vulnerable identifiers and off-chevron traffic, has been exposed to threats by cloud microservices managing ePHI.
- Mis-controlled accumulation of information: This is a significant problem posed by cloud microservices responsible for managing ePHI, as many systems put their data at risk and compromise the security and confidentiality that their clients must gain from a system using ePHI [4].
- Man-in-the-middle / An attacker who disrupts the conversation between two people by using a "Man in the Middle" (MITM) approach for pleasure or benefit.
- Internal problem: An innocent reverse user has breached the access and benefited from the resources.
- Third-party libraries, including compromised container platforms, are critical because they enable unsupervised malware distribution. LOD platform development involves a vast number of containers and software libraries, which facilitates execution [4].

3.2. Security Objectives

The following are some high-level objectives derived from the threat model and regulatory requirements:

- Confidentiality: We have to have encryption to protect all ePHI in transit and at rest.
- Integrity: We must have measures in place to prevent unauthorized alteration and detect if unauthorized alteration has occurred.
- Availability: We must monitor and put into place measures to ensure all critical services are available and systems are resilient to DDoS attacks.
- Accountability / Auditability: All systems must be able to have logs and audit trails that are capable of being unaltered, so that a civil/criminal investigation can be done [5].
- Least privilege & segmentation: All servers and workstations must not be able to communicate with others unless it is necessary for the business. We should also limit the scope of each compromise by using as many mechanisms as prudent.
- Compliance alignment: Implement the controls necessary to meet the HIPAA Security Rule and other standards applicable to their environment.

4. Reference Architecture-Components and Controls

Here is an architecture for cloud microservices that safely provides access to health data.

4.1. Architectural Components (high level)

- API Gateway: Unify intake of external requests to the other layers in the architecture.
- Authentication & Authorization Service: Ensure secure, authenticated, and privacy-preserving access to data only by individuals and applications that have consent.
- Microservices Layer: Enable the modular development of services that interact with each other while enforcing accurate data and contract decoupling [5].
- Service Mesh: Streamline the implementation and detachment of security, circuit breaker, and billing policies into and out of the microservices architecture.
- Data Store(s): Cache persistently data that is transiently needed and store indefinitely data that cannot be recreated.
- Key Management Service (KMS): Protect and rotate unique keys by handing off key management to the cloud.
- Audit & Monitoring: Ensure accountability and detect a security breach through robust event and security logs.
- CI/CD Pipeline and Image Registry: Implement continuous development without compromise, in a secure environment.

- Compliance & Governance Layer: Secure the environment through a secure configuration, compliance, and the presence of business-related and security services [6].

The API Gateway sits as the edge, with the IdP to its side, the service mesh within the cluster, and the KMS and storage part of the secure cloud perimeter.

4.2. Strong Authentication and Authorization

Client and user authentication: Implement OAuth 2.0 and OpenID Connect. Integrations and context flows for patients within EHR: Use the SMART on FHIR service. For short-lived access tokens and refresh tokens, use secure storage. The services will need to grant the user interface permissions for the pet store and the order service, and RBAC and ABAC must be implemented at the service resource levels to prevent unauthorized access. Centralized Auth: Reduces duplicated logic across services. With centralized authorization, people need to implement authorization logic in one place. The redundant, one-off checks can be removed [6].

4.3. Mutual TLS and Service Identity

Use a service mesh to impose mTLS between the services to make sure that only the consumers and producers of the data can access it. Hide the certificate issuance and rotation for mTLS via a certificate authority in the service mesh without relying on a public CA. Service-to-service communication initiated by the client can create a security risk, as attackers may intercept the data.

4.4. Encryption and Key Management

Using a strong key such as AES-256, carry out the encryption of ePHI at rest as well as in transit. Always use the Transport Layer Security of a minimum version of 1.2, whereas it is even preferable to use TLS 1.3. A key management software or a Hardware Security Module (HSM) should be used for the key management. The further result of properly employing this encryption is the logging and recording of the key rotation process and ephemerally encrypting the data key in cloud storage (i.e., envelope encryption). The keys should be kept separate from the rest of the data, and their usage should also be documented and possibly audited. Although encryption is an addressable requirement in HIPAA, it is vital for any modern PAAS services that store or disclose PHI [7].

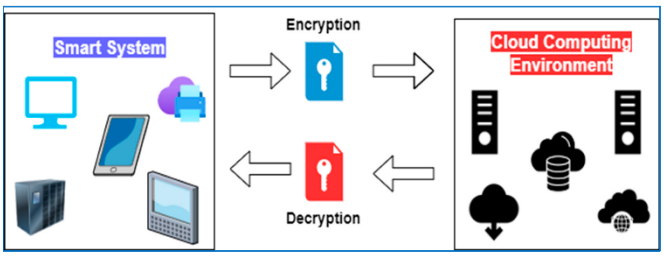

Figure 2. Encryption Techniques for Smart Systems Data Security Offload to the Cloud

4.5. Least Privilege, Segmentation, and Zero Trust

Zero Trust Network access means that every access request must be verified, only the appropriate data must be maintained, and its health must be continuously observed. Network segmentation (namespaces/network policies in Kubernetes) and service bounding policies decrease the blast range. Policies as code (OPA and Rego) can establish very fine-grained policies that are run on the very boundaries of the mesh or the API gateway. The National Institute of Standards and Technology (NIST) Special Publication 800-207 gives a basic design for a zero-trust architecture [7].

4.6. Observability and Auditing

Logs and telemetry across all the layers of the application and infrastructure must be centrally collected, normalized, and stored. By logging various types of access to electronic protected health information (ePHI), we provide a comprehensive audit trail to support forensic investigations. After performing this process a few times, it turned out mostly perfect.

4.7. Secure Development and Supply-Chain Protections

Roll SCA/vulnerability scanning, image signing, and runtime EDR into your CI/CD. Prohibit developers from altering production images or promoting them. Implement, for instance, with OPA, a policy that checks who can run as admin and cascades so that the more secure options are covered. Keep third-party dependencies up to date, keep the world's only parchment SBOM, and swagger the lot as keys to the kingdom [8].

5. Implementation Considerations

5.1. Platform Choices: Kubernetes Istio

Kubernetes offers namespaces, RBAC, and pliability. Istio or analogous service mesh software provides mTLS, policy enforcement, and observability, all without modifying the application code. CIS Benchmarks for Kubernetes should be used, with a security baseline determined by the platform, to configure the cluster's security and establish trust in this security. People are advised to build private clusters (no public endpoint) with strict User and Role Based Access Control (RBAC), and for controlling and limiting pod-to-pod traffic, the use of network policies is highly recommended.

5.2. Identity Provider and Token Management

One IdP that supports OAuth2/OIDC will be enterprise-grade Brite Harnish, PKCE for open clients, and delicate consent flows (SMART on FHIR). The usage of token introspects or JWT confirmation at the gateway and mesh to avoid making own check. Implement multi-layer security (MFA) for the human beings that access to the backside with demanding access [8].

5.3. Data Partitioning and Minimization

When possible, separate PHI into separate data realms and use data minimization—meaning only what microservices actually need, not giving them everything about a patient. Investigate if pseudonymization or tokenization can be used for analytics or nonclinical services.

5.4. BAAs and Legal Compliance

One needs to create legal arrangements between the cloud supplier companies and any third party that accesses ePHI. The BAAs must list acceptable uses, safeguards, and obligations to specify this. Unfurnished establishment for administrative, physical, and technical safeguards for audit, controls that are HIPAA-HITECH specified. Health and Human Services (HHS) has guidelines for using the cloud, its responsibilities for the covered entities, and for the CSPs [9].

5.5. Operational Resilience: Backups, DR, and Patch Management

Backups should be encrypted, while access controls should be incorporated. Carrying out regular tests of the disaster recovery plans would increase reliability. Failing at either of these would lead to a disaster. Patch the container image and node, and if possible, automate security updates. Leading with immutable infrastructure and enabling canary deployment reduces risk when rolling out code.

6. Example Use Cases: Secure FHIR Gateway for Third Party Apps

6.1. Scenario

A hospital can share patient data with third-party clinical applications via the FHIR API, thanks to authorization. The third-party app, which is supposed to obtain approval, should have obtained the patient's consent to work with their data. SMART on FHIR authentications are required for third-party apps, which are intended to receive scoped tokens. These tokens are designed for use by specialists with a single area of expertise [9].

6.2. Secure Flow (high-level)

- Developers register their apps with the hospital's developer portal. Once completed, their apps will obtain their client credentials.
- The end-user will authenticate through the hospital IdP (using MFA). They will then be presented with consent options, which indicate which scopes are being requested.
- The IdP will return access details; this will be in the form of an access token and a refresh token. This will be issued over time and will include patient context, including the SMART context [10].
- The app will make a call to an API gateway. This gateway will be the validation point where existing tokens are verified. Validating this token can be done in numerous ways, including token introspection or JWT verification, which are the two most commonly used ways.
- This specific gateway will then run several requirements based on the token it receives. Depending on the token's proof of validation, the request will be forwarded to the appropriate microservice.
- Microservices will only interact with services with which they have direct access. This is especially the case with connecting to a data service and equally securely returning low-level patient-specific data. The data will be returned to the original user [10].
- All actions will be logged (with a request ID), including user/app identity, so that at any point we can reconstruct the journey of a request for auditing.

The information supplied enforces least privilege, provides patient consent and scope, utilizes token-based authentication as recommended by SMART/FHIR, and benefits from platform-level mTLS and encryption.

7. Evaluation: Risk Assessment and Metrics

7.1. Risk Mitigations Mapped to Controls

- Unwanted access: Things like OpenID Connect or OAuth are the standard go-to things to be handled.
- Data escaping: - It will start on the rest of the requirements for this domain.
- MITM/Tampering: As distributed systems such as Kubernetes are designed to operate in these types of setups, this demands higher-than-usual security for the network [11].
- Insider misuse: This is a challenging domain, with the compensating controls in place requiring strong authentication and authorization controls, and the usage of segmentation and containment in addition to the Detection mechanisms to work effectively.
- Compromise madness: Prepare to always work in a context where trust decreases at every point of internal or external interaction.

7.2. Operational Metrics

They will use to know how proficient their data security measures are if they evaluate digits, namely:

- Alter encrypted try Inter acknowledgment service traffic certificates with the percentage of.
- The time to revoke a compromised token or key.
- Security incidents mean time to detect (MTTD) and security incidents mean time to respond (MTTR).
- Failed authorization attempts per day, the number of (anomaly indicator).
- Patch Compliance Percentage for Runtime Images and Nodes [11].

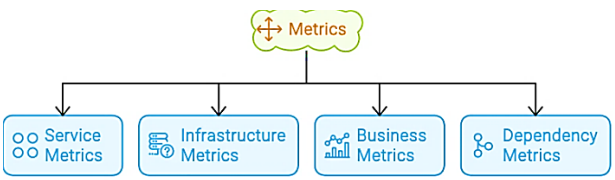

Figure 3. Essential Operational Metrics

7.3. Auditability & Compliance Evidence

Ensuring compliance with relevant standards and frameworks - including PCI, HIPAA, FedRAMP, and SOC2 - is critical across various industries. Including: access logs, BAA documents, KMS logs, patch records, vulnerability scan results achieved, implemented control points, and configured security settings. Automated evidence collection reduces audit time and improves compliance posture for any organization.

8. Conclusion, Limitations, and Recommendations

8.1. Limitations

- Service Mesh, KMS, and fine-grained RBAC require a high degree of operational complexity.
- Latency overheads associated with the use of encryption, token introspection, granular mesh layers, etc
- Security requirements of the mesh are another source of latency and can add to the cost [12].
- Strict security can pose a challenge to developer productivity as well as third-party integration, depending on the implementation of security controls.
- Different architectures might be needed depending on the location of data; e.g., international healthcare projects have to be HIPAA compliant and also accommodate GDPR and national policies on data residency.
- To stay up to date with the evolving standards of security, one would have to constantly review guidance (e.g., NIST updates, FHIR releases)

8.2. Summary of the Findings:

The secure use of Cloud Microservices has been examined for designing and implementing solutions to handle sensitive Healthcare data, particularly electronic protected health information (ePHI). Research into this matter has shown that microservice architectures have an overwhelming level of scalability, flexibility, and interoperability that other architectures cannot match in developing healthcare applications [12]. However, the benefit in question is /are directly proportional in the sense that they are protected with regulatory frameworks that have been put in place to make sure that the information provided to the healthcare applications is protected in accordance with the Health Insurance Portability and Accountability Act [13].

8.3. Importance of Security in Microservices Architectures

There is much more security in cloud-based microservices. It becomes more effective when we transform classic healthcare systems into cloud-based microservices. An attack on every individual microservice may be possible. Thus, central management and continuous follow-up become essential. Security attributes such as mutual TLS, RBAC, and constant vulnerability scanning are also crucial in setting up a secure environment. By contrast, the use of service mesh architecture through Istio enhances security. Removal, encryption, and certification of inter-service communication are largely obsolete. The cleverest thing is that no significant code changes are needed.

8.4. Compliance and Governance Considerations

Cloud services enabled by the IoT can provide economic and creative benefits, but the security and privacy challenges must be considered for its growth to be sustainable. If security measures aren't effectively thought out beforehand, it's nearly inevitable that a data breach or other similar security problem will occur. Cloud providers must consider both regulatory compliance and the impact of shared servers on end customers' compliance requirements, as third-party entities are also required to meet the same healthcare regulatory criteria [14].

8.5. The Role of Zero-Trust and Least Privilege Principles

One of the insights especially relevant to microservice architecture security is referred to as zero trust. The core concept of zero trust is that no entity can be trusted, regardless of its location within or outside the network's perimeter. In other words, this access control model is based on continuously authenticating, authorizing, and continually acting in response to the demands of a least-privilege model. Unlike traditional perimeter-based network defenses, which trust users based on their IP subnet range, the zero-trust model relies on authentication protocols and encryption techniques to protect against a confined and rigid network environment. The access requests must be verified continuously based on identity, context, and device health. The principle of least privilege, or the principle of minimal privilege, means that in a particular abstraction layer of a computing environment, every module (such as a process, a user, or a program, depending on the subject) must be able to access only the information and resources that are necessary to its legitimate purpose [15].

8.6. Technological and Operational Challenges

Aside from the overt benefits, the protection of cloud microservices is advanced and cost-consuming. Service meshes configuration, container orchestration (like Kubernetes), and CI/CD pipelines necessitate expert doctrine. Pervasive HCOs may face limitations in adopting those approaches due to their insufficient ability or budget. Moreover, interoperability of different systems while still preserving protection and compliance with the regulations requires sustained joint ventures of software developers, information assurance and compliance experts, and supervisory authorities.

8.7. Final Reflection

In keeping with the Zero Trust principle, Encryption, Least Privileged Access, and our compliance with the legal Mandate Compliance providers can take full advantage of the benefits of the cloud, which means a patient's safety or the confidentiality of his or her records would not be compromised because of setting up a cloud native architecture. It is not only the improvement of technology that cloud computing, micro-services, and open-source good cybersecurity frameworks mean, but also a robust, strong, adaptable and well vested healthcare system [15].

References

[1] Akerele, J. I., Uzoka, A., Ojukwu, P. U., & Olamijuwon, O. J. (2024). Improving healthcare application scalability through microservices architecture in the cloud. International Journal of Scientific Research Updates, 8(02), 100-109.

[2] Boda, V. V. R., & Immaneni, J. (2021). Healthcare in the Fast Lane: How Kubernetes and Microservices Are Making It Happen. International Journal of Emerging Research in Engineering and Technology, 2(3), 33-42.

[3] Bugshan, N., Khalil, I., Moustafa, N., & Rahman, M. S. (2021). Privacy-preserving microservices in industrial internet-of-things-driven smart applications. IEEE Internet of Things Journal, 10(4), 2821-2831.

[4] Calderón-Gómez, H., Mendoza-Pitti, L., Vargas-Lombardo, M., Gómez-Pulido, J. M., Rodríguez-Puyol, D., Sencion, G., & Polo-Luque, M. L. (2021). Evaluating service-oriented and microservice architecture patterns to deploy ehealth applications in cloud computing environment. Applied Sciences, 11(10), 4350.

[5] Contasel, C., Rughiniș, R. V., Trancă, D. C., & Tsurcanu, D. (2025). Enhancing e-Health cybersecurity and resilience: shifting from monolithic to microservices architecture. UPB Scientific Bulletin, Series C: Electrical Engineering and Computer Science, 21-34.

[6] de Alencar, A. V., Bezerra, M. M., Valadares, D. C., Santos, D. F., & Perkusich, A. (2023, March). An interoperable microservices architecture for healthcare data exchange. In International Conference on Advanced Information Networking and Applications (pp. 193-205). Cham: Springer International Publishing.

[7] Gómez-Pulido, J. M., & Vargas-Lombardo¹, M. (2024, October). Microservices Architecture to Improve. In Technologies and Innovation: 10th International Conference, CITI 2024, Guayaquil, Ecuador, November 11–14, 2024, Proceedings (Vol. 2276, p. 137). Springer Nature.

[8] Martin, J. (2025). Encryption in Transit in Healthcare SaaS: The Role of Mutual TLS. Available at SSRN 5131928.

[9] Mavrogiorgou, A., Kleftakis, S., Mavrogiorgos, K., Zafeiropoulos, N., Menychtas, A., Kiourtis, A., ... & Kyriazis, D. (2021, June). beHEALTHIER: A microservices platform for analyzing and exploiting healthcare data. In 2021 IEEE 34th international symposium on computer-based medical systems (CBMS) (pp. 283-288). IEEE.

[10] Nam, T. B., Khiem, H. G., Triet, M. N., Hong, K. V., Khoa, T. D., Bao, Q. T., ... & Luong, H. H. (2023, September). SPaMeR: securing patient medical records in the cloud-a microservice and brokerless architecture approach. In International Conference on Web Services (pp. 32-46). Cham: Springer Nature Switzerland.

[11] Jamil, F., Qayyum, F., Alhelaly, S., Javed, F., & Muthanna, A. (2021). Intelligent Microservice Based on Blockchain for Healthcare Applications. Computers, Materials & Continua, 69(2).

[12] Pfleger, S. (2024). Data Security in Microservice-Systems (Doctoral dissertation, University of Applied Sciences).

[13] Prakash, C. (2024). Zero-Trust Architecture Approach to Secure Microservices for the Healthcare Insurance Industry. University of the Cumberlands.

[14] Ramalingam, B. C. (2025). Revolutionizing Pharmaceutical Data Transformation with Microservices Architecture. IJSAT-International Journal on Science and Technology, 16(1).

[15] Sheffield, N. C., Bonazzi, V. R., Bourne, P. E., Burdett, T., Clark, T., Grossman, R. L., ... & Yates, A. D. (2022). From biomedical cloud platforms to microservices: next steps in FAIR data and analysis. Scientific Data, 9(1), 553.