Personal Health Train: Advancing Distributed Machine Learning in Healthcare with Data Privacy and Security

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
M.Sc. Marius Simon de Arruda Botelho Herr
aus Gräfelfing

Tübingen
2025

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	05.03.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Nico Pfeifer
2. Berichterstatter:	Prof. Dr. Sven Nahnsen

The ones who are crazy enough to think that
they can change the world, are the ones who do.

Apple Inc. attributed to
Steve Jobs

Abstract

Transferring data between different hospitals is often restricted, and federated analysis of clinical data is a viable alternative. Existing federated analytics frameworks are often limited in the type of input data to process or analysis that can be performed. In the Personal Health Train paradigm, the analysis algorithm (wrapped in a 'train') travels between multiple sites (e.g., hospitals - so-called 'train stations'), hosting the data in their protected infrastructure, and only transfers results rather than the data. Within the established infrastructure of the German Medical Informatics initiatives, patients' structured pseudonymized clinical data is stored in FHIR servers at Data Integration Centers based on the HL7/FHIR profiles of the German National Core Data Set.

Implementing trains as secured containers enables complex data analysis workflows to travel between sites, i.e., genomics pipelines or deep-learning algorithms - analytic methods that are generally not easily amenable. We present PHT-meDIC, a productively deployed, interoperable, open-source implementation of the Personal Health Train paradigm. The scope of applications for this platform ranges from machine learning algorithms to sophisticated omics and image analysis with arbitrary input data. Light-weight virtualization permits the automated deployment of complex data analysis pipelines (e.g., genomics, image analysis) across multiple hospitals in a secure and scalable manner. We combine different open-source third-party services with several custom-developed services. A separation into various services allows flexible adaption and extension in a scalable form. We achieve constant monitoring and persistent execution of trains and are providing governance template documents for deployment. In our proposed security protocol, hospitals have pseudo-identifiers within the infrastructure and can only access their repository, so that such inference attacks are less likely. Results are always encrypted at rest. Only participating sites and the submitting user can access them. Manipulation of trains will be detected at any stage.

Furthermore, researchers can use additional privacy mechanisms (e.g., Paillier cryptosystem). The execution is within an encapsulated environment using project-specific FHIR servers or data warehouses. We successfully deployed the implementation for distributed analyses of large-scale data. Our platform has been extended for interoperability in the Leuko-Expert project with other Medical Informatics Initiative partners' architecture.

Zusammenfassung

Die Übertragung von Daten zwischen verschiedenen Krankenhäusern ist oft eingeschränkt und die föderierte Analyse von klinischen Daten ist eine gute Alternative. Bestehende föderierte Analyse-Plattformen sind oft eingeschränkt in Bezug auf die Art der zu verarbeitenden Eingabedaten oder die durchführbaren Analysemethoden. Im Paradigma des Personal Health Trains reist der Analysealgorithmus (in einem 'Zug' verpackt) zwischen mehreren Standorten (z.B. Krankenhäusern - sogenannten 'Bahnhöfen'), die die Daten in ihrer geschützten Infrastruktur vorhalten, und überträgt nur Ergebnisse anstelle der Daten selbst. Innerhalb der etablierten Infrastruktur der deutschen Medizininformatik-Initiative werden strukturierte pseudonymisierte klinische Daten der Patienten in FHIR-Servern an Datenintegrationszentren bereitgestellt, basierend auf den HL7/FHIR-Profilen des deutschen nationalen Kernsatzes.

Die Implementierung von Zügen als gesicherte Container ermöglicht es, komplexe Datenanalyse Arbeitsabläufe zwischen Standorten zu transportieren, z.B. Genomanalysen oder Deep-Learning-Algorithmen; Analysemethoden, die im Allgemeinen nicht leicht anwendbar sind. Wir präsentieren PHT-meDIC, eine produktiv eingesetzte, interoperable, Open-Source-Implementierung des Personal Health Train-Paradigmas. Der Anwendungsbereich für diese Plattform reicht von maschinellen Lernalgorithmen bis hin zur anspruchsvollen Analyse von Genomen und Bildern mit beliebigen Eingabedaten. Virtualisierung ermöglicht die automatisierte Bereitstellung komplexer Datenanalyse-Arbeitsabläufe (z.B. Genom oder Bildanalyse) über mehrere Krankenhäuser hinweg in sicherer und skalierbarer Weise. Wir kombinieren verschiedene Open-Source-Drittanbieterdienste mit mehreren eigens entwickelten Diensten. Eine Aufteilung in verschiedene Dienste ermöglicht eine flexible Anpassung und Erweiterung in skalierbarer Form. Wir haben eine ständige Überwachung und konsistente Ausführung von Zügen erreicht und stellen Betriebs-Vorlagendokumente für die Bereitstellung zur Verfügung. In unserem vorgeschlagenen Sicherheitsprotokoll haben Krankenhäuser Pseudo-Identifikatoren innerhalb der Infrastruktur und können nur auf ihren Projektserver zugreifen, wodurch solche Schlussfolgerungsangriffe weniger wahrscheinlich sind. Ergebnisse sind immer verschlüsselt. Nur teilnehmende Standorte und der aktive Benutzer können darauf zugreifen. Manipulationen an Zügen werden in jeder Phase erkannt.

Darüber hinaus können Forscher zusätzliche Datenschutzmechanismen verwenden (z.B. das Paillier-Kryptosystem). Die Ausführung erfolgt in einer abgeschlossenen Umgebung unter Verwendung von studienspezifischen FHIR-Servern oder Datenservern. Wir haben die Implementierung erfolgreich für die verteilte Analyse von groß angelegten Daten eingesetzt. Unsere Plattform wurde im Rahmen des Leuko-Expert-Projekts für die Interoperabilität mit der Architektur anderer Partner der Medizininformatik-Initiative erweitert.

Acknowledgments

First, I would like to express my gratitude to my advisors, Prof. Nico Pfeifer and Prof. Oliver Kohlbacher, for allowing me to embark on this exhilarating journey of interdisciplinary research, culminating in this thesis. I am profoundly grateful to both, whose instrumental support has been pivotal throughout this work. Their willingness to listen to my concerns, unfailing meeting availability, and trust in my work have been invaluable. Furthermore, I thank Prof. Sven Nahnsen for taking the time to review my dissertation.

From Tübingen the PHT-meDIC Team; Pfeifer lab; Kolhbacher lab; Akguen lab and meDIC Tübingen: namely Peter Placzek, Michael Graf, Tyra Stickel, Felix Bötte, Florian König, David Hieber, Alexander Röhl; Nicolas Kersten, Jacqueline Wistuba-Hamprech, Agnes Molden; Claudia Walter, Azemina Ceman, Lars Neth; Ali Burak Ünal, Cem Ata Baykara; Patrick Knoblich, Andreas Daul, Sascha Rehm. You all supported me and my work in various manners.

I thank Peter Placzek, especially for dedicating countless unpaid overtime to work alongside me. I am fortunate to have found an exceptional developer and employee in him, a treasured friend. I also thank Sascha Welten for collaborating on the German PHT Implementation Network. I would also like to thank my companions from other Data Integration Centers (TUM, LMU, Erlangen, Freiburg, Gießen, Leipzig, Regensburg, and Berlin) within the MII and my 'SySS Buddies' for their insightful discussions on software security, pen-testing, and privacy. I appreciate Christopher Herr and Dr. Bernhart Herr, who generously proofread the dissertation.

Finally, I want to extend my heartfelt gratitude to my parents, Barbara Borkert-Herr and Dr. Bernhart Herr, my siblings, Julia Herr, Marlen Herr, and Jonas Herr, my friend Rainer Klein, as well to my currently still wife, Anna Carolina de Arruda Botelho Herr. Your unwavering support and continuous encouragement throughout my years of study have been invaluable. Your belief in me and your motivation has been the driving force behind my academic journey. I am deeply thankful for your constant support, which has inspired me to persevere and excel in my work.

Thank you!

General Remarks

- I used commercial services and applications such as Adobe Inc. Creative Cloud, IconScout, and Creately to create illustrations and figures for this thesis.
- While preparing this work, I used Grammarly to improve my language and spelling and enhance the research and writing process. After using this service, I reviewed and edited the content as needed and take full responsibility for the content of this dissertation.
- In accordance with the standard scientific protocol, I will use the personal pronoun ***we*** to indicate the reader and the writer, or my scientific collaborators and myself.

Contents

Chapter 1

Introduction

Motivation

Artificial Intelligence (AI) in healthcare is revolutionizing the medical field by enhancing diagnostic accuracy, personalizing treatments, and improving patient care or treatment costs. Foremost, AI's ability to analyze complex medical data such as magnetic resonance images (MRI) scans, X-rays, and pathology slides significantly improves diagnostic accuracy. Research has shown that AI algorithms can identify patterns in medical images with precision, often surpassing human experts, particularly in specialties like radiology, dermatology, and pathology[1]. Personalized medicine is another significant benefit, where AI processes vast amounts of patient data, including genetic information, to tailor treatments to individual needs. This approach is promising in oncology, where AI is used to analyze genetic mutations and predict the most effective treatment for specific cancer types, leading to more effective and less invasive treatments[2]. Moreover, AI is enhancing patient care and engagement. AI-powered chatbots and virtual health assistants provide 24/7 support and personalized advice, improving patient engagement and treatment adherence. These technologies are particularly beneficial for chronic disease management, where continuous monitoring and personalized advice can make a significant difference[3]. In drug discovery and development, AI is speeding up the process and reducing costs by predicting how different drugs will work on various diseases. This leads to faster and more efficient development of new medications[4]. Furthermore, AI is playing a critical role in managing healthcare resources. By predicting patient admission rates and identifying potential health crises, AI enables better resource allocation and preparation, thereby enhancing the overall efficiency of healthcare systems[5]. Lastly, AI's role in training and education for healthcare professionals must be considered. Through simulations and predictive analytics, AI aids medical professionals' training, providing students with realistic scenarios and data-driven insights to enhance their skills[6].

However, integrating AI into healthcare, while promising all the aspects above, brings many complex challenges. With AI's reliance on vast amounts of personal health information, the risk of data breaches and misuse of sensitive patient data is significant, raising major privacy and security issues. This situation is further complicated by the potential for AI to perpetuate biases present in its training data. Such biases could lead to healthcare disparities, where certain demographic groups might receive less accurate diagnoses or treatments[7]. AI's "black box" nature, where decision-making processes are often opaque, is another critical hurdle[8,9]. Incorporating AI into existing healthcare systems requires substantial changes in workflows and processes[10]. The cost and accessibility of AI technologies, especially in under-resourced settings, raises concerns about exacerbating healthcare disparities[11].

In summary, AI's current application in healthcare is markedly improving diagnostic precision, tailoring treatments, elevating patient care quality, expediting drug development, streamlining healthcare resource allocation, and strengthening medical training. These advancements are supported by extensive research and practical implementations, transforming healthcare into a more efficient and patient-focused domain. Nevertheless, challenges such as data privacy and security, technical complexities, methodological concerns, and integration hurdles persist and require attention. Adopting distributed open-source learning platforms offers a viable approach to mitigate these technical and non-ethical issues.

Distributed Machine Learning

Machine learning (ML), particularly distributed ML, is a cornerstone for applying AI and data analytics, especially in the context of evolving data regulations and the European AI Act established in June 2024[12]. This strategy involves dispersing ML algorithms to various sites, often to accommodate privacy concerns and comply with stringent data regulations such as GDPR[13] and HIPAA[14]. These regulations impose strict limitations on data sharing and storage, making it imperative to find innovative ways to harness the power of AI while safeguarding sensitive information.

In response to these challenges, large consortia like GO:FAIR[15] and Medical Informatics Initiative (MII)[16] and organizations[17–20] increasingly embrace distributed learning methods. These consortia pool their resources, computational power, and expertise to develop AI models that can operate without losing control at each site. By distributing algorithms to local data sources, they reduce the risk of data breaches and maintain compliance with data regulations. However, this approach entails a trade-off between functionality and security[21]. While it mitigates privacy risks, it may limit the AI's performance due to the fragmentation of data across multiple sites[22]. To strike a balance between functionality and security, regulations like the AI Act aim to provide a framework for ethical and responsible AI development. Such regulations establish guidelines for AI system transparency, accountability, and compliance

with data protection standards[12]. In addition to regulatory measures, advanced cryptographic and federated learning techniques are emerging as powerful tools to facilitate secure and privacy-preserving distributed ML, enabling organizations to harness the full potential of AI while respecting data privacy and regulatory constraints[23].

In centralized model learning, contributing entities extract data from their systems and forward it to a central repository for analysis. This model simplifies the execution of analyses and is highly convenient for researchers. However, it poses significant control and security issues for hospitals, as they need more oversight over how their data is used once it leaves their sites. Even though they are a technically convenient solution, data protection concerns have been raised against the complete centralization of patient data. Central databases pose a greater risk of re-identifying patient data due to their centralized nature and the possibility of combining the information with other publicly available data sets[24,25].

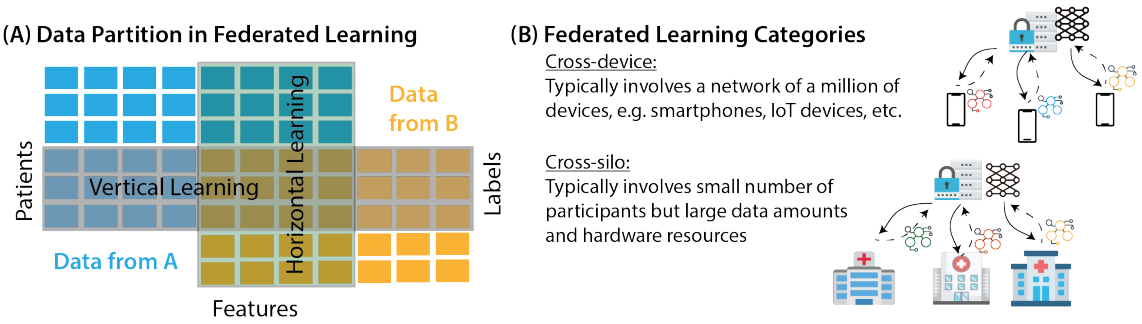


Figure 1.1: (A) Data Alignment in Federated Learning: Horizontal FL involves different sites (A and B) with unique samples but shared features, while Vertical FL involves shared samples but distinct features. **(B) Federated Learning in Practice:** Institutions collaboratively train a model with local computations and secure aggregation, maintaining data privacy. FL is categorized into *cross-device* and *cross-silo* learning.

Federated learning (FL) can be divided into two main types: *horizontal* and *vertical* FL, each designed for different data distribution scenarios across participants (see Figure 1.1 A). This distinction becomes apparent when considering two sites, A and B, with datasets involving patients and features (e.g., lab values).

In *horizontal* FL, the participating sites (A and B) each have datasets that include different groups of samples (patients) but share the same feature space (e.g., the same types of lab values). This setup is typical when multiple hospitals collect the same types of data from distinct patient populations. The goal is to train a model collaboratively across all patients from both sites without transferring actual patient data between them. For example, suppose Site A has data on patients 1–100 and Site B on patients 101–200, collecting the same lab values. In that case, they can use horizontal FL to create a model that benefits from their datasets' combined diversity and size.

In *vertical* FL, the participating sites (A and B) have datasets with the same samples (patients) but different sets of features (e.g., one site records specific lab values, while the other focuses on clinical measurements). This scenario arises when hospitals specialize in different medical fields, such as cardiology and neurology, and collect complementary data types on shared patients. Vertical FL enables these sites to enrich their feature space by combining their unique datasets. The model is trained as if all the data were centralized without sharing sensitive information between sites. This approach enhances predictive accuracy by leveraging the full range of data available for each patient while maintaining data privacy.

FL is divided deeper into *cross-device* and *cross-silo* learning (see Figure 1.1 B). Cross-device learning typically involves leveraging an expansive network of a million or more devices, such as smartphones, wearables, or IoT devices. Each contributes small but valuable datasets. This approach harnesses these devices' collective data and computational power to perform sophisticated learning tasks without centralizing data, prioritizing privacy and efficiency. Cross-silo learning involves collaboration among several institutions or 'silos' - such as hospitals - allowing them to build a collective machine learning model without centralizing data. This approach addresses privacy concerns and regulatory restrictions by keeping sensitive data localized within each silo while still benefiting from the insights gathered across multiple distinct datasets. Usually, each institute operates powerful data centers equipped with thousands of CPUs, hundreds of GPUs, and vast storage capacities. FL in healthcare usually refers to cross-silo learning[26].

FL supports various types of model training (see Figure 1.2 B), including iterative, parallel, and swarm learning methodologies. Iterative models are sequentially updated across sites, parallel models combine local updates into a global aggregated model, and swarm learning allows for autonomous site operations with independent updates and model distributions. In swarm learning, no central orchestration is usually needed, and each site operates independently as a participator or orchestrator.

Distributed ML is a crucial strategy for navigating the complex landscape of data regulations and privacy concerns. It allows organizations to leverage the benefits of AI while protecting sensitive information. However, striking the best trade-off between functionality and security requires a combination of regulatory measures like the AI Act and deploying advanced methods to ensure that the potential of distributed ML applies to heterogeneous and big data. Distributed ML, especially in healthcare settings, has ethical, technical, and organizational limitations to overcome.

In ML, performance metrics are crucial for evaluating and comparing the effectiveness of algorithms. Accuracy, precision, recall, and the F1 score are commonly used metrics for classification problems, providing insights into the model's correctness and ability to manage imbalanced datasets[27]. For regression tasks, metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are used to quantify the difference between predicted and actual values, offering a measure of prediction accuracy[28].

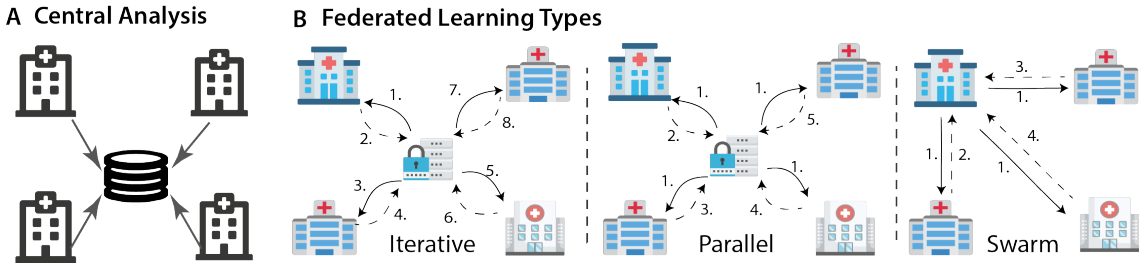


Figure 1.2: Comparison of Centralized Analysis and Federated Learning Types. (A) Centralized analysis involves pooling all data into a single location for model training, which compromises data privacy. **(B) Federated Learning (FL) types**: *(Left)* Iterative FL: Institutions perform local training in parallel and iteratively exchange model updates with a central server, enabling gradual improvement of a global model while preserving data privacy. *(Middle)* Parallel FL: Local models are trained independently, and updates are shared simultaneously with a central server or among peers, allowing faster convergence of the global model. *(Right)* Swarm Learning: A fully decentralized approach where institutions collaborate dynamically through peer-to-peer sharing of model updates, eliminating the need for a central server while supporting adaptive distributed learning.

Area Under the Receiver Operating Characteristic Curve (AUC-ROC) is another significant metric, especially useful for binary classification problems, as it evaluates the model's ability to discriminate between classes[29].

Additionally, newer metrics like Precision-Recall AUC provide alternatives for imbalanced class distributions, focusing on the positive class more critically[30]. These metrics serve as benchmarks for model optimization and selection and to ensure that models can be evaluated.

In distributed settings, computing the AUC-ROC while maintaining data privacy raises challenges, mainly when operating with sensitive labels such as patient data. Ensuring privacy demands techniques such as differential privacy or secure multi-party computation, which can complicate the direct use of labels in AUC computation and potentially reduce the metric's accuracy due to added noise or approximation errors. The need to protect privacy when sharing labels among distributed nodes can limit the granularity of feedback available for model tuning, potentially affecting the reliability of the AUC-ROC as a performance metric in scenarios where precise label information is crucial for accurate evaluation. We present the distributed privacy-preserving exact (DPPE) AUC computation method to solve this, combining several privacy-enhancing technologies.

Thesis outline

This dissertation is located at the intersection of medical informatics —emphasizing privacy-preserving distributed open-source machine learning and data integration—and the domain of computer science, focusing on cybersecurity and software architecture. It successfully translates

novel academic concepts into practical applications, enabling the operation of the PHT-meDIC platform in hospitals with actual patient data.

Chapter 2 provides essential organizational and methodological backgrounds. The related work section highlights the specific need for the PHT-meDIC implementation. Subsequent chapters, 3, detail the platform's methods and concepts, 4 present results, 5 discuss the importance and critical thoughts on the implementation. Concluding with future outlooks in chapter 6, guiding the reader through this research's comprehensive conclusion and outlook.

Part I: Distributed Learning: From vision to productive deployment

This thesis's initial segment examines the distributed learning paradigm, addressing the challenges of deploying self-developed open-source software within hospital environments. The integrated national standards to enhance the research efficiency of the German medical Informatics initiative are presented.

The background section is complemented by a detailed introduction to the Personal Health Train (PHT) implementation network, representing an innovative combined effort for distributed health data analysis. The section extends into the domain of privacy-enhancing technologies, outlining various purposes and procedures designed to strengthen the confidentiality and integrity of sensitive data.

Furthermore, an exhaustive review of related work enriches the background final section, comprehensively comparing existing systems. This review articulates the distinctions between these systems and outlines the necessity for the PHT. The background section lays a solid foundation for the thesis's contribution to the science of distributed learning and privacy preservation in healthcare data analysis.

Part II: Personal Health Train: PHT-meDIC

The second part of the thesis is structured to present the core contributions and findings related to the PHT-meDIC platform. Chapter Methods illustrates the PHT-meDIC's foundational concepts, detailing its application across various use cases and explaining its security framework, the privacy-enhancing AUC computation method, and studies on interoperability. This chapter concludes by focusing on aspects of interoperability with other PHT platforms.

Following this, the results section 4 showcases the architectural framework through use cases, extending to actual analysis of patient data. It evaluates the performance and scalability of the privacy-preserving AUC method, culminating with interoperability findings.

The thesis progresses to a discussion on the implications of PHT-meDIC and the broader application of AI in healthcare, pointing out considerations of the future direction of this research area. The concluding chapters use the PHT-meDIC outcomes, proposing a forward-looking perspective on potential research and technological advancements in medical informatics.

Chapter 2

Background

The content of Section 2.1 constitutes an extended version of the manuscript:

Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train (PHT-meDIC)[31]

The content in Section 2.1.4 constitutes an extended version of the manuscript:

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis[32]

The content in Section 2.2.5 constitutes an extended version of the manuscript:

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC[33]

The content in Section 2.3 constitutes an extended version of the manuscript:

Federated learning and analysis solutions for biomedical data science: A scoping review

This chapter introduces the required background to this thesis. The first subchapter describes the go:FAIR initiative and the German medical informatics initiative (MII) and their context to the PHT. Furthermore, it explains the required administrative steps to deploy software in clinical infrastructure. The MII-aimed data standards and how the data integration center (DIC) infrastructure can support distributed learning are presented. The second subchapter brings the fundamental cybersecurity background of this thesis: privacy and security. The last subchapter describes the related work in distributed learning and the trade-offs each platform had to set.

2.1 Open-Source Software and German University Hospitals

Introduction

As part of the German National Medical Informatics Initiative (MII)[34,35], German university hospital centers have chosen a distributed data management and governance approach. Since 2018, university hospitals have established data integration centers (DICs). Each university hospital DIC integrates its patient data from various distinct primary source systems (e.g., laboratory system, patient management, documentation, or radiology imaging systems) in a standardized way locally. Furthermore, its task is to provide researchers with secure and reliable access to pseudonymized healthcare data while allowing the local university medical centers to control that data. As part of this infrastructure, healthcare data will be analyzed using centralized and distributed analysis tools to achieve the benefits outlined in the previous section. Due to its simple and efficient way, DICs have first chosen centralized analysis where one site receives the data of all participating sites[36]. From the beginning, the MII has established a distributed learning task force[37] to evaluate and discuss the feasibility of open-source distributed learning techniques and software. We from Tübingen were represented and introduced from the beginning our ideas and concepts of the PHT.

The PHT is a paradigm for distributed healthcare data analysis proposed by the GO-FAIR initiative[15,38]. The basic concept of the PHT is that the analysis algorithm (wrapped in a 'train') travels between multiple sites (so-called 'train stations'), which securely hosts the data. Light-weight container-based virtualization is typically used to implement trains, enabling the rapid deployment of complex software pipelines (e.g., for genomics or image analysis) without additional software installation. These concepts have been described in the literature before[39–42], but production-ready implementations have been lacking so far.

The PHT-meDIC was based on organizational and operative desires developed to begin as a cross-silo iterative learning platform. The primary focus is on horizontal FL. The distinction of services allows rapid transformation to other execution types. The design and architectural requirements of distributed learning systems vary significantly between cross-device and cross-silo learning, as well as among the different execution methods. Despite these differences, all forms face comparable cybersecurity challenges, emphasizing the need for robust security measures across all distributed learning models. However, cross-silo federated learning comes with its own set of challenges, including managing the communication overhead between silos, ensuring robustness against data heterogeneity (where data distributions differ significantly across silos), and addressing potential issues of trust and governance among participating entities.

In centralized learning models, contributing entities extract data from their systems and forward it to a central repository for analysis. This model simplifies the execution of analyses and is highly convenient for researchers. However, it poses significant control and security

issues for hospitals, as they need more oversight over how their data is used once it leaves their sites. Moreover, the history of data breaches and cyber-attacks highlights the vulnerabilities inherent in centralized systems.

2.1.1 Deploy Software at University Hospitals

Deploying any and especially self-developed software at university hospitals involves several steps and considerations. Since hospital infrastructure is categorized as system-relevant or critical infrastructure of the government[43], these steps are essential to ensure that the software meets the stringent standards of healthcare environments, particularly regarding patient safety, data security, and regulatory compliance.

On a regulatory level, each German state has independent regulations and laws[44]. These may vary from state to state, but the European GDPR Compliance for data protection and privacy is generally essential. Besides these standards, four documents are essential for deploying software within university hospitals. These clearances are considered to be the approval of the IT security officer, data protection officer, and board of directors.

First, a technical document (Technisches Konzept) is required to fulfill a clinical IT department's security requirements. It describes how the software is integrated into existing systems, what APIs or ports are used, and what software languages and tasks are used for each service.

Second, an operational document (Betriebskonzept) is mandatory for the data protection officer. This document describes what data is transmitted over which interfaces between services and what processes are provided to detect malicious or abnormal behavior. It describes how quality assurance, testing, data management, and what roles are needed to operate the software. Also, financial aspects or where users can find documentation, liability aspects, continuous improvement, and post-deployment monitoring are described in this document. This document also defines downtime, backup, and recovery strategies.

The third document outlines the Technical and Organizational Measures (TOMs), detailing the procedures and safeguards implemented to handle specific events. These measures include processes for revoking user tokens in case of compromised credentials, granting and managing user permissions to ensure appropriate access control, and integrating various system components to maintain operational efficiency. For example, the TOMs might specify how to deactivate a user's access immediately upon detecting unauthorized activity or how permissions are reviewed and updated when a user transitions to a new role within the system. Additionally, it may describe protocols for combining audit logs from different services to ensure traceability and compliance with regulatory standards.

The last required document to get final approval from a German university hospital to deploy software is a risk assessment (Riskobewertung), which describes the impact on the

infrastructure and other systems if something is not behaving as expected in the deployed software.

Patient Data Usage and Governance

Besides the approval to operate the software, the patient consent of MII is the fundamental legal basis[45] for analyzing patient data in Germany. Within the established DIC infrastructure, each patient's consent must be provided upon request and describe in detail how patients agreed on involvement. Patients can decide what kind of data they want to provide for scientific purposes.

Furthermore, the FL platforms need to consider the Research Data Act (FDG) goals from the coalition agreement. From the FDG consultation[46], the following aspects can be derived: Data sovereignty is essential to enable responsible parties to control and monitor the use and further processing of data. Distributed analysis allows data to remain where it is generated or stored, reducing the need for centralized data collection and associated security risks (data security and data protection). An infrastructure for distributed analysis also supports self-determined data management (access management) and adaptation to regional data protection regulations.

Furthermore, distributed analysis frameworks can be an additional system for traceability of data access, leading to increased transparency, as it must always be possible to trace how and for what purposes data is being used. The demand for transparency in data processing and usage directly leads to the need for an open-core platform, as it is the only solution that enables disclosure and verifiability of the fundamental source code.

Involvement in a diverse open-source community provides valuable feedback and new perspectives and promotes knowledge exchange and collaborative innovation. Community contributions enable an agile development cycle, leading to faster error resolution and the rapid implementation of new features[47]. This increases flexibility in responding to demand changes and secures a competitive advantage through continuous development. Unlike proprietary systems, where internal processing mechanisms often remain hidden, open-source software allows anyone to have insight into the exact functioning of data processing. While logs and notifications about system activities can inform, they do not provide insight into the actual program logic and algorithms that control data processing. This is crucial to strengthen user trust in how their data is handled and to ensure that data processing complies with established ethical and legal standards.

The European Union's Cyber Resilience Act (CRA)[48] is a landmark legislative initiative aimed at improving cybersecurity across the EU by introducing unified cybersecurity requirements for digital products and services. The main idea is to classify products into three categories based on their cybersecurity risks: Class I (lower risk), Class II (higher risk), and an unclassified or default category for products without critical cybersecurity vulnerabilities. The

default category covers most connected devices like photo-editing software and video games and requires companies to self-assess their vulnerabilities. Products with higher risk levels, categorized as Class I and Class II, are subject to stricter compliance requirements. Class II products, in particular, must undergo third-party assessments to demonstrate conformity. It is highly likely that even more stringent obligations will apply to sectors like healthcare or critical national infrastructure, such as hospitals, where cybersecurity risks have far-reaching implications.

The CRA mandates that companies adopt a security-by-design approach during the design and development phase of products, addressing vulnerabilities early. It outlines essential security and vulnerability handling requirements, such as ensuring products are delivered without known exploitable vulnerabilities, protecting data confidentiality and integrity, and addressing vulnerabilities through regular updates.

Besides the CRA, the EU AI Act[12] represents a pioneering effort to regulate AI within the European Union, aiming to ensure AI systems are safe, transparent, traceable, non-discriminatory, and environmentally friendly. This legislation, the first of its kind globally, categorizes AI systems based on the risk they pose to society, with specific regulations tailored to each level of risk. The AI Act sets clear obligations for high-risk AI systems, which could negatively impact safety, fundamental rights, or the environment. These include a mandatory fundamental rights impact assessment and specific requirements for AI systems in sensitive sectors like elections, law enforcement, and critical infrastructure.

Criticism of the Cyber Resilience Act (CRA) and the EU AI Act has been raised by open-source organizations and technology experts, particularly regarding the potential impact on open-source software development and the challenges these regulations may impose on solo developers, start-ups, and small businesses.

At the time of writing, both the CRA and the AI Act had entered into force but were not yet fully applicable. The CRA, which became effective on December 10, 2024, will be enforceable starting December 11, 2027, while the AI Act, effective from August 1, 2024, will apply from August 2, 2026. This transitional period allows stakeholders to prepare for compliance with the new requirements.

Additionally, nearly six years after the General Data Protection Regulation (GDPR) came into effect, ambiguities in certain provisions remain, with varying interpretations by legal professionals and courts. This precedent highlights potential challenges in the consistent implementation and enforcement of the CRA and AI Act, raising concerns about their practical implications for the technology sector.

2.1.2 HL7/FHIR and the MII

The MII aims to make clinical patient data from various university hospitals available via DIC for biomedical research. The MII core dataset (see Figure ref2.1) is divided into modules, enabling a structured data presentation. The overall aim is to enable this data to be used across all sites. The Core Data Set (CDS) specification facilitates collaboration and further development in working groups. All MII consortia are actively involved in the design of the core dataset and accept the results as binding for data exchange between the consortia, with decisions being made by consensus. Cooperation between various interdisciplinary experts is crucial for quality and acceptance[49].

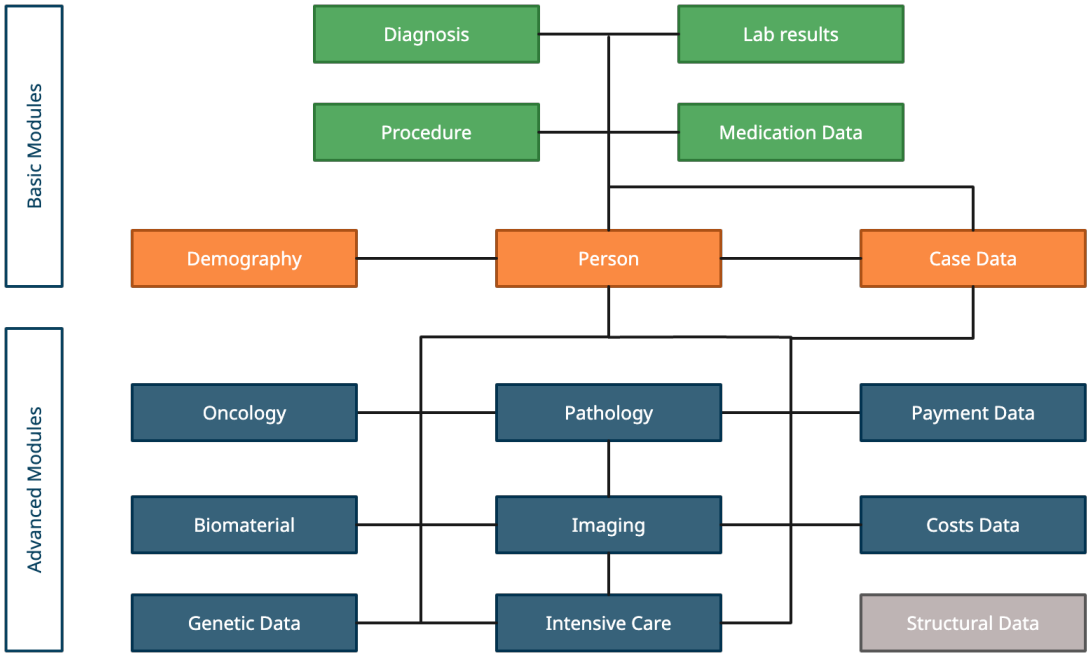


Figure 2.1: FHIR CDS of the MII: The model is structured into Basic Modules (green and orange) and Advanced Modules (blue and gray). Basic Modules encompass foundational data categories such as diagnosis, lab results, and case data, while Advanced Modules include specialized data such as oncology, biomaterials, and genetic data. This hierarchical organization supports interoperability and modularity for clinical and research data integration.

The MII does not primarily strive for standardization but works with national and international standardization organizations such as HL7 Germany. International standards are preferred; in-house developments are integrated into international standardization processes. In-house developments that violate international standards are avoided[49].

However, these standardization efforts are not implementable out of the box. A variety of Fast Healthcare Interoperability Resource (FHIR) servers exist: Hapi[50], Blaze[51], and

Linux4Health[52] are just three prominent examples. Usually, these servers provide structured and interoperable access to patient data, but a search-query translation must be provided if different servers are used within one distributed analysis. Therefore, the PHT uses its custom-developed library *FHIR-kindling*[53] for this purpose. New extensions of MII CDS modules are handled in different versions of CDS. In practice, updating existing resources to new CDS versions is not trivial and is very time-consuming on FHIR servers with many resources. Furthermore, a semantic matching of all CDS versions has to be provided upfront for analysis and data discovery.

Further efforts were made to represent the FHIR CDS as resources in graph databases[54]. Representing FHIR data in graphs would allow more straightforward data exploration and sophisticated filtering mechanisms. However, this is within the proof-of-concept stage during the writing of this thesis.

2.1.3 GO:FAIR, PHT Implementation Networks and the MII

The GO:FAIR initiative emerged in response to the European Commission's 2016 launch of the European Open Science Cloud (EOSC), which aimed to establish a federated environment for sharing and reusing scientific data across Europe. Building on the FAIR Guiding Principles—Findable, Accessible, Interoperable, and Reusable—GO:FAIR seeks to enhance the openness and accessibility of scientific research data, aligning closely with the EOSC's objectives.

In 2018, a coalition of "early mover" EU member states, led by the Netherlands, initiated the GO:FAIR initiative as an independent, community-driven effort. This initiative emphasizes the practical implementation of the FAIR Principles, fostering an open and inclusive ecosystem for scientific data sharing. Central to this effort are Implementation Networks (INs), which are self-governed consortia operating in an open, inclusive, and community-led manner[55].

INs are critical in advancing GO:FAIR's mission by developing specific tools and resources that contribute to the Internet of FAIR Data and Services (IFDS). These networks embody the initiative's core principles of openness, inclusivity, and community leadership while engaging participants from diverse disciplines, countries, and organizations worldwide. Their primary objectives are centered around three key areas: Implementation, Community Building, and Communication.

Aligned with the GO:FAIR initiative's strategic pillars, INs focus on three core activities:

- **GO Build:** Driving technological advancements and infrastructure development.
- **GO Change:** Promoting cultural transformation towards open and FAIR practices.
- **GO Train:** Educating and training stakeholders in FAIR principles and practices.

Through these efforts, INs are transforming data access and sharing, fostering a collaborative and open scientific ecosystem that aligns with the overarching goals of the EOSC[15].

One of these various INs is the Personal Health Train with its German subchapter. The PHT IN Manifesto, signed by several universities, serves as a comprehensive declaration outlining the vision, objectives, and guiding principles of the PHT initiative, particularly within the ambit of the GO:FAIR movement. At its core, the manifesto aims to revolutionize health research and care by advocating for secure, efficient, and ethical access to health data across various institutions, leveraging federated learning and privacy preservation principles[56]. Central to the manifesto is the adherence to the FAIR Principles. This commitment underscores the initiative's dedication to enhancing the utility and accessibility of health data while strictly upholding patient privacy and adhering to the highest ethical standards. The manifesto emphasizes the importance of maintaining individuals' privacy, ensuring that data handling and analysis processes always respect and protect patient confidentiality. The technological underpinnings and infrastructure supporting the PHT are also detailed, highlighting cutting-edge technologies and standardized protocols to facilitate secure data sharing and analysis without needing to centralize sensitive information physically. This approach safeguards privacy and fosters innovation by enabling researchers to seamlessly access and analyze data across borders and institutions. Collaboration and community engagement are pivotal themes within the manifesto, with a concrete call to action for researchers, healthcare providers, patients, and policymakers to come together. This collaborative spirit is essential for driving forward the goals of the PHT, facilitating a community-led approach that harnesses collective expertise and resources for the greater good. The role of Implementation Networks is particularly emphasized, pointing to their crucial function in operationalizing the PHT concept. These networks are tasked with developing tools, standards, and practices that are essential for the practical application of the PHT. Furthermore, the manifesto outlines clear, actionable objectives for the PHT Implementation Network, including creating interoperable health data platforms, establishing educational programs for researchers, and enhancing public awareness and engagement with the initiative. It also addresses governance and sustainability, laying out a framework for decision-making, collaboration, and the long-term viability of the PHT initiative. Since the beginning, the MII funded SMITH and DIFUTURE[35] consortium, namely Aachen, Leipzig, and Tübingen University, participated in the PHT IN and founded a German subchapter. The German chapter focuses on national goals and developments[57].

2.1.4 Interoperability

Interoperability is divided (see Figure 2.2) into two main categories: *vertical* and *horizontal* interoperability. Vertical interoperability focuses on the compatibility of software objects within a stack, encompassing, for example, hardware-level drivers and executable programs. Horizon-

tal interoperability pertains to the compatibility of multiple software stacks within a workflow. In this context, we consider a workflow as the execution of a train within a PHT infrastructure. In terms of the PHT concept usually horizontal interoperability between two distinct platforms is meant by connecting at least two stacks including the data provision in one analysis process.

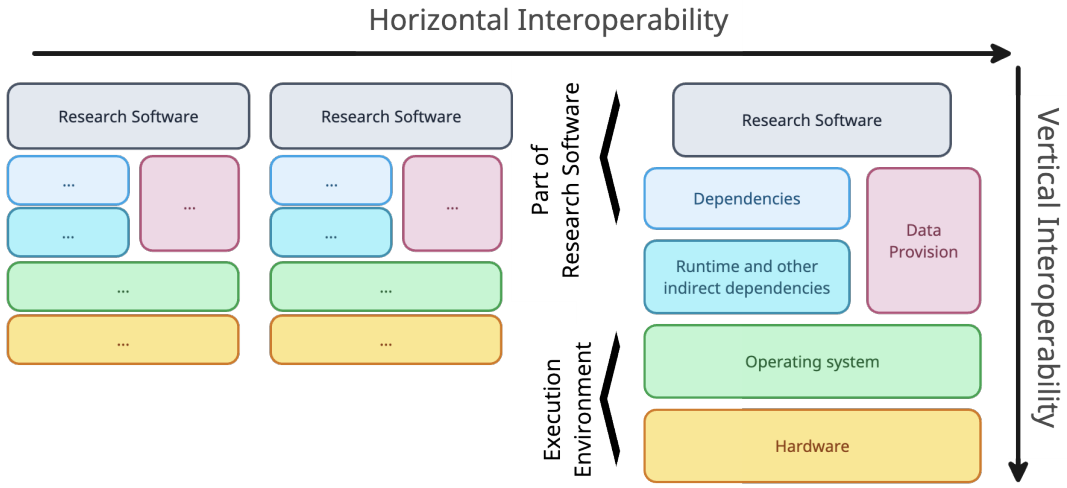


Figure 2.2: Refined definition of interoperability: Interoperability is divided into two main categories: vertical and horizontal interoperability. Vertical interoperability focuses on the compatibility of software objects within a stack, encompassing, for example, hardware-level drivers and executable programs. Horizontal interoperability pertains to the compatibility of multiple software stacks within a workflow. Adapted from Lamprecht et al.[58]

Multi Homing

An essential element of the discussion revolves around the comparison between the benefits of interoperability and what we term 'multi-homing' - *'the situation in which users tend to use several competing platform services in parallel'* as defined by the Directorate-General for Communications Networks of the European Commission[59]. In our specific PHT case, this implies that a single institution operates at least two station software applications from distinct PHT implementations and operates them concurrently. This scenario would make interoperability, as described in this thesis, irrelevant.

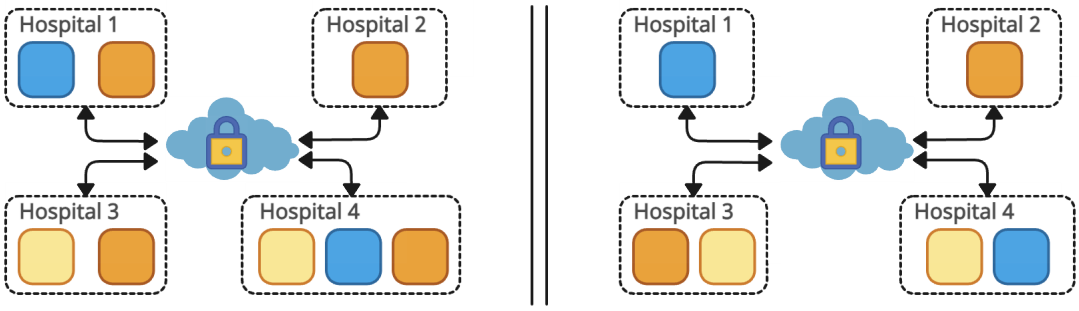


Figure 2.3: Multi-homing Scenarios in PHT Applications. *Left:* Ideal multi-homing scenario where all institutions deploy a common PHT application (depicted in orange). This uniformity simplifies data analysis and enables seamless collaboration across institutions. *Right:* A heterogeneous landscape where institutions utilize diverse PHT applications (represented in blue and orange). Interoperability between these infrastructures facilitates efficient data analysis across varying deployments. Colors (blue, orange, and yellow) are used to highlight different PHT applications or data components, emphasizing the need for compatibility in both homogeneous and heterogeneous deployment scenarios.

Layers of interoperability

Benson et al.[60] defined different layers of interoperability in the context of healthcare interoperability and gave hints as to why large consortia and software projects fail in terms of interoperability. Overall they state four different layers:

1. **Technical interoperability**
   This layer ensures that data can be exchanged between two or more systems and that the receiving system can receive the data. This level does not imply that the receiving system can interpret or process the data.

2. **Semantic interoperability**
   It ensures that the information exchanged between systems is understandable and can be processed by the receiving system. This level requires that data is semantically interoperable using standard terminologies and ontologies so that both parties exchange and interpret it identically. Therefore, it is the highest level of interoperability.

3. **Process interoperability**
   Process interoperability occurs when there is a shared understanding among individuals across a network, enabling business systems to work together and processes to be synchronized effectively. It brings tangible advantages when people can utilize information from external sources in their daily tasks, with safety and privacy being crucial elements. This represents the human dimension of interoperability.

4. **Clinical interoperability**
   This level involves the policies, social issues, and agreements that enable the secure use of data within and between institutions, entities, or regions. It encompasses the governance, standardization, and infrastructure that enable shared goals and purposes.

From these four layers of interoperability, we defined, inspired by Benson et al.[60], our five interoperability layers in the PHT context (see Figure 2.4). These can be summarized in the following:

- **Layer 0 - Data Integration**
  The foundational step in our multi-layered approach involves harmonizing data across different infrastructures. This layer focuses on aligning and integrating the different data formats, structures, and standards from various data sources into a unified format such that it can be seamlessly processed by the analysis train.

- **Layer 1 - Assigning (Globally Unique) Identifiers to Stations**
  In order to transfer trains between infrastructures, it is necessary to establish a method for identifying the station unambiguously across infrastructural borders. This is essential to ensure the correct routing of trains between the infrastructures and stations.

- **Layer 2 - Harmonizing the Security Protocols**
  The PHT infrastructures were developed with different requirements regarding the security protocols and the encryption of the train. Therefore, it is essential to formulate an overarching security protocol that aligns with infrastructure-specific requirements.

- **Layer 3 - Common MetaData Exchange Schema**
  By employing distinctive station identifiers (Layer 1), we establish the initial building block of a shared communication standard. As the security protocol also requires metadata for proper functioning (e.g., exchange of public keys), our third objective is to create a common set of metadata that not only facilitates technical interoperability but also extends to a first foundation for semantic compatibility. This layer primarily merges the metadata items from Layers 1 and 2 into a machine-readable format.

- **Layer 4 - Overarching Business Logic**
  After we have established all the preliminaries mentioned above, we need to develop the actual business logic to transfer trains between the infrastructures from a technical perspective based on the route defined by the identifiers (Layer 1).

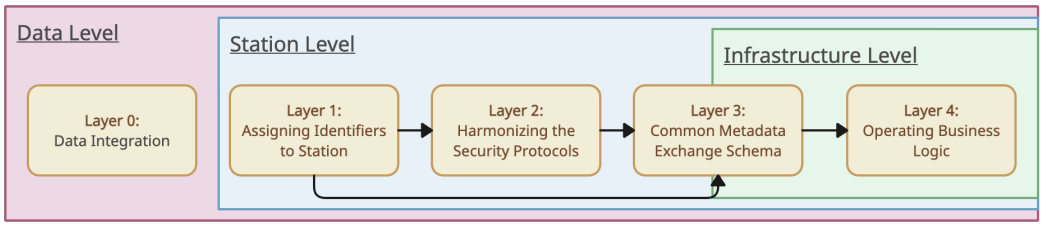


Figure 2.4: Our multi-layered framework for interoperability: From data integration to business logic. In our interoperability framework, Layer 0 is associated with the data level. The harmonization of station types, such as PHT PADME platform or PHT-meDIC, is addressed in Layers 1 and 2, and to some extent in Layer 3. The overarching business logic is encapsulated within Layer 4 at the infrastructure level. The arrows illustrate the inter-dependencies and collaborative interactions across the layers: Layer 4 utilizes the metadata established by Layer 3 to navigate the trains through the infrastructures. Layer 3 consolidates the essential metadata produced by Layers 2 and 1. Layer 2 then uses the unique station identifiers to secure the trains accordingly.

2.2 Cybersecurity

Introduction

The immense sensitivity of medical data requires confidentiality and safeguarding it against misuse. Health data, by its very nature, contains deeply personal details about an individual's physical, genetic, and mental well-being. If improperly handled, this information could lead to significant privacy violations and potential discrimination.

The need for robust data privacy and security measures in the healthcare sector cannot be overstated. With the immense progress of technology and the increasing digitization of health records, the risks associated with data breaches, unauthorized access, and cyber attacks have escalated[61]. These risks not only compromise the privacy of individuals but also erode the trust in healthcare systems.

Effective data privacy policies and security protocols are essential to protect sensitive health information. These protocols include implementing advanced encryption methods, strict access controls, and regular security audits. Healthcare providers and organizations must comply with legal and ethical standards, such as the GDPR in Europe or HIPAA in the United States, which set the groundwork for protecting health information. Protecting health data is not just a technical issue but a fundamental aspect of preserving human dignity and rights. Ensuring the privacy and security of medical data is crucial for maintaining patients' trust.

2.2.1 Data Breaches and AI Challenges in Healthcare

Healthcare data breaches are becoming increasingly frequent and severe, presenting a substantial risk to patient privacy. Data breaches in the healthcare sector expose sensitive patient

information, including medical histories, diagnoses, and insurance details, which, if leaked, could lead to identity theft, blackmail, or discrimination[62]. The healthcare industry is a prime target for cybercriminals due to the high value of medical data on black markets, often priced higher than financial data because of its permanence and detail[63]. A study found that the cost of healthcare data breaches is often the highest among all industries, with each breach costing millions of dollars due to legal consequences, compensation, and lost trust[64].

The rapid integration of Artificial Intelligence (AI) into healthcare has introduced new cybersecurity concerns. AI algorithms are trained on immense amounts of data to operate effectively, and healthcare data has been used extensively to train diagnostic, predictive, and operational models[65]. However, this dependency on data poses additional risks, as AI systems are vulnerable to both direct and indirect forms of attack. For example, adversarial attacks can manipulate input data to cause AI models to misclassify or misinterpret information, potentially leading to harmful consequences in patient care[66]. Additionally, AI models, particularly those trained on personal health data, may unintentionally memorize sensitive information. This issue, known as model inversion, can allow attackers to reconstruct data used in training, thus posing a significant privacy risk[67].

Addressing these cybersecurity concerns requires a multifaceted approach that includes robust data encryption, rigorous authentication protocols, and regular testing against adversarial threats. Healthcare organizations must also ensure that AI models comply with regulatory standards and undergo rigorous validation to mitigate risks associated with model inversion and adversarial attacks. As AI continues to evolve in healthcare, developing and enforcing security standards designed to handle the unique risks associated with AI and machine learning models is essential for protecting patient privacy and maintaining trust in healthcare systems.

2.2.2 Privacy vs. Security

The distinction between privacy and security is crucial in discussions about data management. Privacy deals with how personal data is used and controlled, whereas security protects that data. While it is possible to have security measures without privacy safeguards, privacy cannot exist without security. Both computer security and privacy are crucial for handling personal and sensitive information. Privacy usually relates to personal details and preferences about sharing them, while security focuses on protection against potential threats. Cybersecurity involves protecting data from unauthorized use or access.

Privacy concerns how organizations collect, manage, store, and use data. Traditionally, in information technology, personal privacy has been seen as not available to everyone, unlike security, which is considered a basic necessity. Public security is often valued more than individual privacy, but a lack of personal privacy can lead to public security issues. Although they are different concepts, security and privacy are essential and must be carefully maintained.

2.2.3 Anonymization vs. Pseudonymization

Anonymization and pseudonymization are two distinct methods for protecting personal data, but they differ significantly in their approaches and the level of data protection they offer. Anonymization involves altering personal data so that the individual cannot be identified directly or indirectly, even if the data is combined with other information. Once data is anonymized, it is no longer considered personal data and does not fall within the scope of the GDPR[68]. In contrast, pseudonymization is a process in which identifying information from a data record is replaced by one or more synthetic identifiers or pseudonyms. While this does not make the data completely anonymous, it adds a layer of security as the true identity is not directly linked with the data. However, unlike anonymized data, pseudonymized data can still be attributed to a specific individual if additional information is provided, making it a less robust form of data protection compared to anonymization. In practice, anonymization is more secure but could result in the loss of too much relevant information, which could affect the quality of the analysis. Usually, the data anonymization process is not trivial and requires a deep understanding of the underlying attributes and features[69].

2.2.4 Threat Models

In cybersecurity, threat models can be categorized based on the nature and intent of the potential attacker. One model differentiates between *honest but curious* and *malicious adversaries*. An honest but curious user is assumed to follow protocol specifications but is interested in gaining additional information. At the same time, a malicious user actively seeks to compromise the system or data integrity. These models consider potential attackers' capabilities, objectives, and resources to understand and mitigate risks.

Besides the honest-but-curious and malicious models, other threat models exist. An *insider threat* means a legitimate user with authorized access attempts to misuse their access to negatively impact the institution's critical assets. An *eavesdropping attack* means an attacker attempts to steal information that computers, smartphones, or other devices transmit over a network. In *collusive attacks*, multiple adversaries work together to launch a coordinated attack. Lastly, *physical attacks* need to be mentioned, involving physical damage to infrastructure or hardware. These models help anticipate the types of threats and design appropriate defense mechanisms.

2.2.5 Privacy Enhancing Technologies

Privacy has emerged as a paramount concern in the ever-evolving landscape of software engineering. The increasing interconnectivity of digital systems and the exponential growth of personal data collected have heightened the risk of privacy breaches. This situation under-

scores the critical need for privacy-enhancing technologies (PETs) in software engineering. PETs are a suite of methods and tools designed to protect users' personal information. The overall goal is to minimize the risk of unauthorized access or use of data, ensuring that individual privacy rights are respected and maintained. These technologies are not just beneficial; they are becoming essential in a world where data breaches and unauthorized surveillance are increasingly common. The integration of PETs into software engineering practices addresses several crucial aspects:

- **Regulatory Compliance**: With the introduction of the GDPR in the EU or the California Consumer Privacy Act (CCPA) in the United States, compliance has become a significant driver for the adoption of PETs. These technologies help companies and organizations adhere to legal requirements by ensuring that personal data is handled responsibly and securely.
- **Trust and Reputation**: In an era where consumers are more aware of their digital rights, implementing PETs can enhance a software product's trustworthiness. Companies that prioritize user privacy are more likely to build a positive reputation.
- **Data Minimization and Anonymization**: PETs facilitate data minimization and anonymization techniques. By ensuring that only necessary data is collected and cannot be easily traced back to individuals, these technologies significantly reduce the potential harm from data leaks or misuse.
- **Secure Communication**: Encryption and secure communication protocols are integral to PETs, safeguarding data in transit and at rest. Secure communication is crucial in healthcare, where sensitive information is regularly transmitted.
- **User Empowerment**: PETs often include mechanisms for giving users control over their data. User empowerment can include consent frameworks, transparent data usage policies, and easy-to-use privacy settings, which provide a sense of control over their personal information.
- **Mitigating Insider Threats**: PETs also address security concerns from within an organization, such as unauthorized access or data misuse by administrators or employees. Techniques like role-based access control are essential in this context.

The next subchapters explain the essential concept and background of various PET methods and concrete used protocols.

Hashes and Signatures

Digital signatures and cryptographic hash functions (see Figure 2.5) are fundamental to ensuring the integrity and authenticity of digitally transmitted messages. A cryptographic hash

function takes an input (or *message*) and returns a fixed-size string of bytes, typically a digest that appears to be random. The property of these Hash algorithms, such as SHA-256, is that any change in the input will produce a significantly different output, making it infeasible to generate the original input from the hash value (collision resistance) from any input data. Digital signatures allow the sender of a message to generate a signature by encrypting the message's hash value with their private key. The receiver can then verify the integrity and authenticity of the message by decrypting the signature with the sender's public key to retrieve the hash value and then comparing this with the hash value they compute from the received message. If the two hash values match, it confirms that the message has not been manipulated and that the owner of the corresponding private key indeed sent it.

This combination of hashing and digital signatures provides a robust mechanism for secure message transmission, ensuring that messages are tamper-proof and authentically from the claimed sender. These technologies underpin various digital systems' security, including secure email, software distribution, SSL/TLS for secure web connections, and blockchain technology[70].

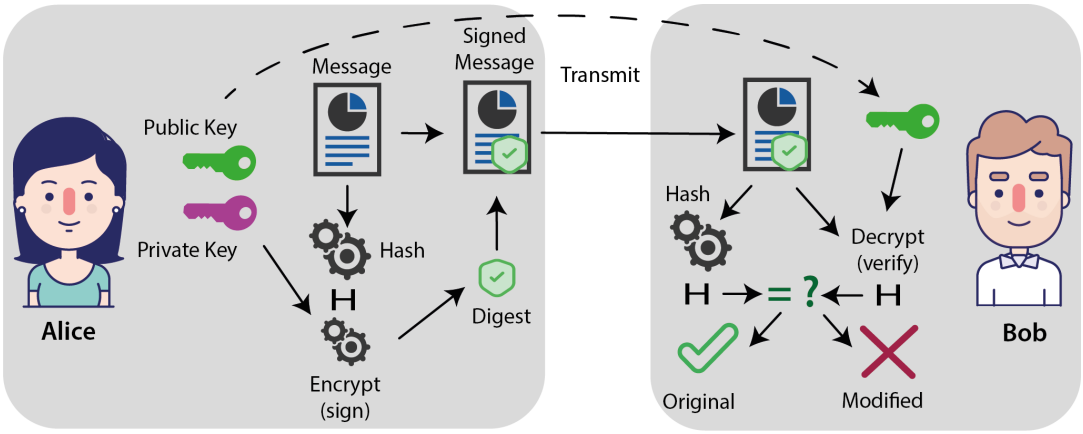


Figure 2.5: Digital Signing and Verification Process: This figure illustrates digitally signing a message and verifying its authenticity. *Left:* Alice creates a hash of the original message, encrypts the hash using her private key (purple), and appends it as a digital signature to the message. *Right:* Bob receives the signed message, computes the hash of the received message, and decrypts the signature using Alice's public key (green). If the computed hash matches the decrypted hash, the message is verified as authentic and unmodified; otherwise, it is marked as tampered.

Asymmetric encryption

Asymmetric encryption, also known as RSA (Rivest–Shamir–Adleman) encryption or public key cryptography, is a commonly used method for data encryption and decryption using a pair of public and private keys. The fundamental principle of asymmetric encryption is that what one key encrypts, only the other can decrypt. As the names indicate, the public key can be shared

with other users, while the private key has to stay confidential. This key pair is mathematically related but not identical, and it is computationally infeasible to derive the private key from the public key with today's computers.

The method is separated into four basic steps: Key creation, key distribution, encryption, and decryption.

Key creation

The procedure to generate RSA algorithm keys is the following:

1. Choose two large prime numbers p and q which are kept secret. The standard method chooses them randomly until two prime founders are found.
2. Compute $n = p \times q$ where n is the modulus of public and private keys.
3. Calculate the Euler totient function $\phi(n) = (p-1) \times (q-1)$, which must stay secret since it is used for public and private key calculation.
4. A public exponent e is a coprime to $\phi(n)$ with 65537 being a common choice for e due to its properties of being a prime and having a short binary ($2^16+1 = 65537$ representation. The requirements for e are that $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$.
5. The private exponent d is calculated as the modular multiplicative inverse of e modulo $\phi(n)$. This means d is the solution to the equation $d \times e \equiv 1 \mod (\phi(n))$. The value of d must be kept secret.
6. The public key is composed of the modulus n and the public exponent e, while the private key consists of the modulus n and the private exponent d. The public key can be shared with anyone, while the owner must keep the private key secure.

The RSA algorithm's security relies on the difficulty of factoring the product of two large prime numbers. The public and private keys are mathematically linked, but with the current state of computing, it is not feasible to efficiently derive the private key from the public key.

Key distribution and encryption

The public key, which consists of the modulus n and the public exponent e, is freely distributed and can be shared with anyone who wants to send an encrypted message. Digital certificates, which are verified by a trusted certificate authority (CA), can be used to distribute public keys securely. These certificates ensure that a public key belongs to the entity claimed to be the owner, mitigating the risk of man-in-the-middle attacks. The integrity and authenticity of the public key are paramount. They ensure that it truly belongs to the claimed owner and has not been tampered with or replaced by a malicious party.

To send a secure message to the key owner, a sender encrypts the message using the recipient's public key. This process transforms the plaintext message into ciphertext using n and e. Since the encryption uses the recipient's public key, only the corresponding private key can decrypt the message, ensuring that only the intended recipient can read the content.

Decryption

The private key, composed of the modulus n and the private exponent d, is a confidential component securely held by its owner and never shared. It is used to decrypt messages that have been encrypted with the corresponding public key. The strength of RSA encryption depends critically on the confidentiality of the private key; any exposure of the private key would render the encrypted communication vulnerable and compromise its security.

Effective key distribution and management are critical to maintaining RSA-encrypted communications' security. Techniques such as Public Key Infrastructure (PKI) are often employed to manage the distribution and verification of public keys. This ensures that users can confidently encrypt messages, knowing that only the intended recipient can decrypt them.

Symmetric encryption

Symmetric encryption protocols ensure the confidentiality and integrity of information exchanged across insecure networks. The principle of symmetric encryption involves using a single, shared secret key for the encryption of plaintext and the decryption of ciphertext. This methodology allows for a secure communication channel between parties, even in the presence of potential eavesdroppers.

Symmetric encryption protocols (see Figure 2.6) start when Alice wishes to send a confidential message to Bob. She generates the symmetric key to encrypt her message, converting the message into ciphertext, which obscures the original content from unauthorized entities. Upon receiving the ciphertext, Bob applies the same symmetric key to decrypt the ciphertext, thereby regaining the original message. This process underscores the critical importance of the symmetric key, which must be securely shared between Alice and Bob before any encrypted communication.

Symmetric encryption is renowned for its computational efficiency, particularly in scenarios requiring the encryption of large data volumes. This efficiency, however, is contingent upon the secure management and exchange of the symmetric keys. A failure in this aspect, such as an adversary's interception of the key, compromises the confidentiality of the encrypted messages. Consequently, the security of symmetric encryption is inherently linked to the mechanisms employed for key distribution and management. Consequently, it is desirable to update keys more frequently.

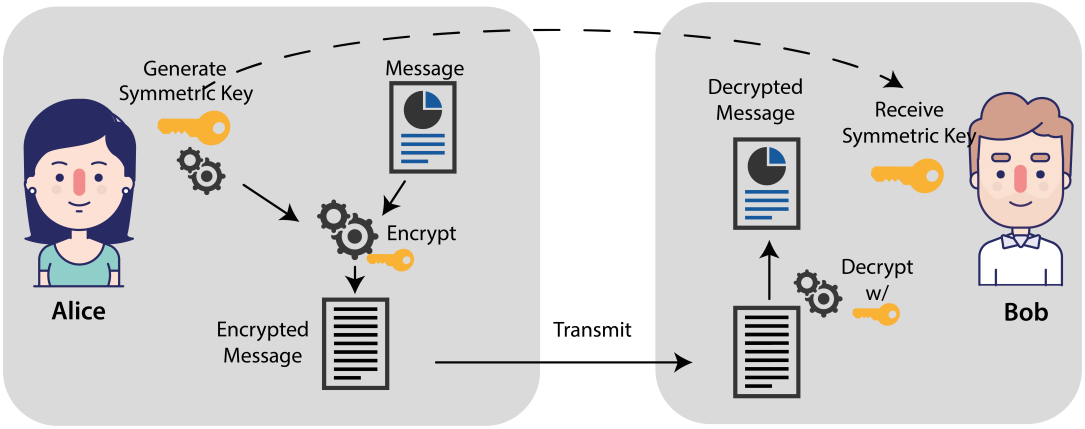


Figure 2.6: Symmetric Encryption and Decryption Process: This figure demonstrates the process of encrypting and decrypting a message using a symmetric key. *Left:* Alice generates a symmetric key (depicted in yellow) and uses it to encrypt the message, producing an encrypted message that is ready for transmission. *Right:* Bob receives both the encrypted message and the symmetric key. Using the same key, he decrypts the message to retrieve the original content. Symmetric encryption relies on the secure exchange of the key, as encryption and decryption require the same shared secret.

Among the vast amount of symmetric encryption algorithms, the Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Triple DES (3DES) are prominent. AES, in particular, is acclaimed for its robust security features and has been adopted globally to secure sensitive information in various sectors, including government and finance. Our PHT-meDIC security protocol also uses AES encryption to encrypt sensitive information, such as models, queries, and algorithms. Key management in symmetric encryption is challenging in environments requiring secure communications among multiple parties. This has prompted the development of sophisticated key management protocols, which are designed to facilitate the secure generation, distribution, storage, and renewal of symmetric keys.

Envelope Encryption

Envelope encryption is a security technique that protects sensitive data using a hierarchical encryption protocol. It is instrumental in cloud computing environments and for protecting data at rest. This method combines the benefits of symmetric and asymmetric encryption to offer a secure, efficient way to manage and protect vast amounts of data.

The following steps describe the protocol (see Figure 2.7):

1. First, the message or plaintext data is encrypted using a symmetric encryption algorithm. Symmetric encryption is preferred for encrypting the data because it is much faster than asymmetric encryption. The key used for this process is known as the data key.

2. Next, the symmetric key is encrypted with a second key, a so-called wrapping key. The symmetric key is protected using the asymmetric encryption public key.

3. The encrypted data key is stored alongside the encrypted data, while the wrapping key is stored separately, often in a dedicated key management service (KMS). This separation of keys adds an extra layer of security, making it more challenging for unauthorized users to access encrypted data.

4. When authorized access to the encrypted data is required, the private part of the wrapping key decrypts the data key. The decrypted data key is used to decrypt the data, allowing it to be accessed.

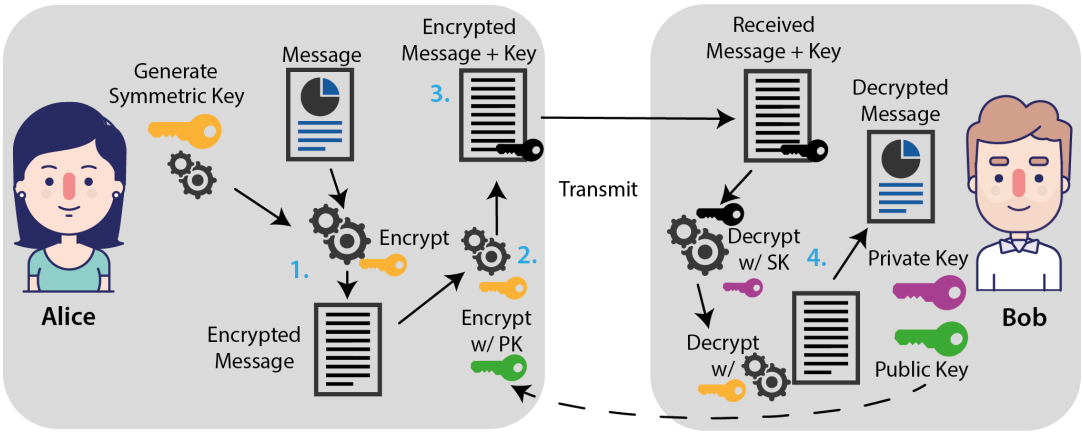


Figure 2.7: Envelope Encryption: Combining Symmetric and Asymmetric Techniques: This figure demonstrates envelope encryption, a method that combines the efficiency of symmetric encryption with the security of asymmetric encryption for key exchange. *Left:* Alice generates a symmetric key (yellow) to encrypt the message, ensuring efficient encryption. The symmetric key is then encrypted using Bob's public key (green) and sent alongside the encrypted message. *Right:* Bob receives the encrypted message and symmetric key. He uses his private key (purple) to decrypt the symmetric key, which is then used to decrypt the message. This approach ensures secure transmission of the symmetric key while leveraging its speed for encrypting large messages.

The following main advantages of envelope encryption exist:

- **Scalability**: It allows for efficient key management, especially in systems where large amounts of data need to be encrypted. Using a single wrapping key to secure multiple data keys simplifies the process of rotating and managing keys.
- **Performance**: By using symmetric encryption for the data, envelope encryption ensures high performance, even with large datasets.
- **Security**: It offers robust security by combining the strengths of both symmetric and asymmetric encryption protocols. Even if the encrypted data is leaked, the data remains secure without access to both the data and wrapping keys.

- **Flexibility**: Envelope encryption can be implemented in various environments and applications, providing flexibility in securing data across different platforms and technologies.

Envelope encryption is widely used in cloud services to protect sensitive information, ensuring that data stored in the cloud remains secure against unauthorized access, even if it is compromised. It is a critical component of a comprehensive data security strategy, especially for organizations that handle sensitive or regulated data on a large scale. The ML models can easily be several gigabytes large, so using envelope encryption is an excellent strategy for our security protocol 3.2.4.

Homomorphic encryption

Homomorphic encryption is a form of encryption that allows computations to be carried out on ciphertexts (encrypted values), generating an encrypted result that, when decrypted, matches the result of operations performed on the plaintext values[71]. This unique property enables secure processing of sensitive information in encrypted form without needing access to the plaintext data, thus preserving privacy. Homomorphic encryption schemes can be classified into partially homomorphic, somewhat homomorphic, and fully homomorphic encryption (FHE), each offering different computational capabilities.

Paillier Encryption

The Paillier cryptosystem[72], invented by Pascal Paillier in 1999, is a partial homomorphic encryption scheme that allows the following two types of computation:

- addition of two encrypted values
- multiplication of an encrypted value with a plaintext number

Similar to the RSA schema, the encryption schema is separated in key generation, encryption and decryption:

Key creation

The procedure to generate RSA algorithm keys is the following:

1. $p, q \in P$ with equal length.
2. $n = p \times q$
3. $g = 1 + n$
4. $\phi(n) = (p-1) \times (q-1)$
5. $\mu = \phi(n)^{-1} mod\ n$ (μ is used as a private key and kept secret by the owner).

Encryption
To encrypt a message m the following steps are required:

1. Plaintext $m < n (m \in Z_n)$
2. Choose random r where $0 < r < n$ and $gcd(r, n) = 1$
3. Ciphertext $c = g^m * r^n \ mod \ n^2$

Decryption
To decrpyt the ciphertext c the message m where $L()$ is the logarithm function

1. Ciphertext $c < n^2$
2. Plaintext $m = L(c^\lambda \ mod \ n^2) * \mu \ mod \ n$

Paillier encryption, a partially homomorphic scheme, supports unlimited additions but not multiplications of encrypted values. This makes it more efficient and practical for specific use cases, such as secure aggregation of encrypted data or secure voting.

Modified Paillier Cryptosystem The modified Paillier system[73,74] supports the addition of two ciphertexts and the multiplication of a ciphertext with a plaintext constant. This allows users to perform computations on ciphertexts, which results in an identical output as that performed on plaintexts. In this system, the public key is $(n, g, h = g^x)$. n is the product of two safe primes: z and y. The secret key is $x \in [1, \frac{n^2}{2}]$ and g that is in order of $\frac{(z-1)(y-1)}{2}$ equals to $-a^{2n}$ where $a \in \mathbb{Z}^*_{n^2}$

- **Encryption:** The message $m \in \mathbb{Z}_n$ is encrypted as follows:
  $c_1 = g^r \mod n^2$ and $c_2 = h^r(1 + mn) \mod n^2$
  where a random $r \in [1, n/4]$.
- **Decryption:** To recover m, the decryption of (c_1, c_2) is performed as follows:
  $m = dec(\frac{(c_2/c_1^x - 1) \mod n^2}{n})$
- **Proxy Re-encryption:** The secret key x is split into two shares such that $x = x_1 + x_2$. In the modified Paillier system the ciphertext (c_1, c_2) can be partially decrypted to (c'_1, c'_2) as follows:
  $c'_1 = c_1$ and $c'_2 = c_2/c_1^{x_1} \mod n^2$
  Then, (c'_1, c'_2) can be decrypted to m using x_2 instead of x with the decryption method described above.

Homomorphic Properties The modified Paillier cryptosystem is homomorphic with respect to both addition and scalar multiplication. This means that certain operations on encrypted data correspond directly to the same operations on the underlying plaintexts without requiring the data to be decrypted first. Such homomorphic properties are fundamental in enabling secure computations on encrypted data.

- **Addition**: Let $(c_{1,1}, c_{1,2})$ be the encryption of a message m_1 and $(c_{2,1}, c_{2,2})$ be the encryption of a message m_2. In the modified Paillier scheme, "adding" these ciphertexts (through the group operation in $\mathbb{Z}_{n^2}$, typically component-wise multiplication) produces a new ciphertext $(c_{3,1}, c_{3,2})$.

  $(c_{3,1}, c_{3,2}) = (c_{1,1} \cdot c_{2,1} \bmod n^2, \quad c_{1,2} \cdot c_{2,2} \bmod n^2)$.

  Decrypting $(c_{3,1}, c_{3,2})$ yields $m_1 + m_2 \mod n$. Hence,

  $\text{dec}(c_{3,1}, c_{3,2}) = m_1 + m_2 \pmod n$. This property is crucial in scenarios where multiple values must be summed securely.

- **Scalar Multiplication**: Additionally to addition, the Paillier cryptosystem also supports the multiplication of an encrypted value by a plaintext constant. Suppose (c_1, c_2) encrypts a message m, and let k be a known integer in $\mathbb{Z}_n$. Then raising the ciphertext to the power k corresponds to multiplying the underlying plaintext by k:

  $(c_1^k \bmod n^2,\ c_2^k \bmod n^2) \quad \longrightarrow \quad m \times k \pmod n$

  Consequently, $\text{dec}(c_1^k, c_2^k) = m \times k \pmod n$

  This ability to apply scalar multiplication homomorphically is useful for computing weighted sums or scaling factors while preserving the confidentiality of the underlying data.

These homomorphic properties allow us to calculate the exact AUC with partial decryption in proxy re-encryption settings — without ever exposing the underlying plaintexts.

Full Homomorphic Encryption (FHE)

FHE is the most powerful, supporting addition and multiplication operations an unlimited number of times, allowing for arbitrary computation on encrypted data. However, FHE is known for its significant computational overhead and scalability challenges, making it impractical for some real-world applications due to long runtimes[75].

Even simple arithmetic operations on encrypted data are substantially more complex than those on plaintext, leading to performance issues for large-scale applications. Despite ongoing research and optimization, including developing more efficient algorithms and hardware acceleration techniques, these challenges still need to be addressed for general adoption.

Randomized Encoding for Multiplication

One of the privacy-enhancing techniques that we use in this study is Randomized Encoding (RE)[76][77]. The idea of RE is to use random values to hide a function's inputs and reveal only its output. It creates components of encoding of the desired function by using random values. Then, the output of this function can only be obtained by combining these components in a certain way. There are REs for several functions in the literature for privacy-preserving machine learning[78][79]. Among those, we benefit from the RE for *multiplication*[80].

Definition 1 (Modified RE for Multiplication[80]) *Let f be a function of x_1 and x_2 defined over a ring* R *such that $f(x_1, x_2) = x_1 \cdot x_2$. f can be perfectly encoded by the function $\hat{f}$ defined as follows:*

$$\hat{f}(x_1, x_2, r_1, r_2, r_3) = (x_1 + r_1, x_2 + r_2, r_2 x_1 + r_1 x_2 + r_1 r_2)$$

where r_1, r_2 and r_3 are uniformly chosen random values. In order to obtain the result of $y = f(x_1, x_2)$, one needs to compute $c_1 \cdot c_2 - c_3$ given the encoding (c_1, c_2, c_3).

2.2.6 Trade-Off between Functionality and Security

In distributed ML platforms, particularly within healthcare scenarios, the balance between usability, security, and performance emerges as a critical area of focus. Implementing such security measures, while crucial, introduces significant communication overheads, additional runtime, and impacts on the system's overall usability, presenting a complex trade-off scenario that requires careful navigation.

In distributed ML, data is processed across multiple nodes to leverage computational resources and preserve data privacy. Introducing encryption for data-in-transit and at-rest, alongside secure multi-party computation (SMPC) protocols for privacy-preserving analytics, significantly increases the communication overhead. Each layer of security necessitates additional data exchanges, encryption/decryption operations, and coordination among nodes, thereby inflating the runtime and potentially diminishing system responsiveness. This is particularly critical in healthcare scenarios where timely data processing is essential for patient care and outcomes.

From a usability perspective, healthcare professionals require intuitive systems that facilitate rather than hinder their workflows. The added complexity of secure data handling procedures—such as accessing data through secure interfaces or navigating additional authentication steps—can reduce system accessibility and efficiency. This reduction in usability can lead to resistance from healthcare providers, potentially compromising the adoption and effective use of distributed ML technologies in clinical settings. Another shortcoming of the FHE or SMPC integration in such platforms is the need for more verification since data managers or administrators cannot inspect the input data or the results, as all information remains encrypted throughout the computation process.

The level of security implemented within distributed ML platforms should be proportional to the sensitivity of the data and the potential impact of a data breach. The need for strong security measures is indisputable in healthcare, where data sensitivity is high. However, not all data require the same level of protection. Differentiating between varying data sensitivity levels can allow for tiered security measures, reducing unnecessary overheads for less sensitive information while maintaining strict controls for highly sensitive data such as genetic information or disease diagnoses[81].

Addressing the tradeoff between usability, security, and performance necessitates the adoption of adaptive security mechanisms[23]. These mechanisms dynamically adjust security protocols based on the context of the data exchange, the type of computation being performed, and the current threat landscape. For instance, lighter encryption schemes or reduced authentication steps might be employed during low-risk operations or when handling less sensitive data, thereby minimizing communication overheads and improving system performance without compromising overall data security.

The design and implementation of distributed ML platforms in healthcare require considering at a fundamental level the interdependencies between usability, security, and performance. By employing adaptive security mechanisms and differentiating security measures based on data sensitivity, it is possible to mitigate communication overheads and runtime impacts, thereby enhancing the usability of these platforms for healthcare professionals. Ensuring the effective balance of these factors is essential for distributed ML technologies' widespread adoption and success in improving patient care and healthcare outcomes. Therefore, the developed components of the PHT-meDIC must be generic enough to include various combined approaches within the same architecture.

2.3 Related Work

Introduction

Before related federated learning frameworks, platforms, and SMPC are introduced, we briefly aim to define and distinguish them. Frameworks are collections of code libraries, tools, and best practices that provide a structured way to develop software applications. It dictates the architecture and design of a project by offering reusable components, predefined classes, and functions, enabling developers to focus on the unique features of their application rather than the underlying infrastructure. Frameworks are designed to be used (or extended) rather than modified directly, meaning developers write code that plugs into the framework according to its rules and conventions.

In a technological context, a platform refers to an environment that allows software to run or a base upon which applications are developed and run. It can include hardware architec-

tures, operating systems, and runtime environments. Platforms provide a broader scope than frameworks, offering libraries, tools, and infrastructure services such as computing resources, identity and access management, storage, and networking capabilities. Platforms can support multiple frameworks and languages and often provide a level of abstraction over hardware and system-level details. In contrast, secure multi-party computation frameworks (SMPC) can combine both. They are essential for ensuring security and privacy in federated analyses, operating under malicious threat models to compute global results from fully encrypted input data. Traditionally, SMPC solutions have been domain-specific and highly specialized, with limitations in analysis functionalities and scalability due to performance overheads.

While frameworks help in the development phase by providing a structured way to build software, platforms encompass a wider range of capabilities, including deployment, execution, and management environments for the developed software. As mentioned before, SMPC solutions can vary widely in their use and application. A few selected are compared.

Furthermore, it must be mentioned that emerging blockchain technologies are not discussed as related work since blockchain approaches make only sense in federated device learning settings with many participants and comparable hardware resources. The 51% attack is a problem in cross-silo learning with only few participants[82] and therefore the arising blockchain approaches are not compared with exisiting federated learning platforms.

2.3.1 Frameworks

A library is at the lowest level of complexity. The term's meaning remains consistent while various software libraries exist for different programming languages, tasks, etc. Libraries are usually defined assets and collections of reusable code. Their primary focus is to provide pre-prepared functionality. Code, which has already been written to fulfill a specific task, can easily be called into the current workflow to provide a specific solution to a given problem[83–85]. It must be noted that the code writer makes use of previously implemented solutions and retains control over the entirety of the workflow. A framework is focused on the inversion of this control. While software frameworks are also a form of reusable functionality, they represent an entire architecture of a solution applicable to a given problem. The over-arching workflow is usually defined and controlled by the framework, the code writer defining the functionality for specific tasks, essentially filling in the gaps in the structure outlined by the framework[86].

Several frameworks have emerged as significant contributors in the federated learning domain. TensorFlow Federated (TFF)[87], initiated by Google (Alphabet Inc.), is an open-source framework engineered for machine learning on decentralized data. It supports the development of models that utilize data distributed across devices, ensuring that this data remains localized. PySyft[88], developed by OpenMined, extends popular libraries such as PyTorch and TensorFlow to support encrypted computations, including federated learning, differential

privacy, and multi-party computation, positioning it as a pivotal tool for privacy-preserving distributed learning. While broader in its application, Apache Flink[89] offers an open-source framework for distributed data processing and real-time analytics relevant to distributed machine learning scenarios. Horovod[90], developed by Uber, facilitates the streamlined training of deep learning models across multiple GPUs, interfacing seamlessly with TensorFlow, Keras[91], PyTorch[92], and Apache MXNet[93] and requiring minimal code modifications for distributed training. Ray presents a simple, universal API for constructing distributed applications, with Ray RLlib[94] specifically designed to scale reinforcement learning experiments across multiple CPUs or GPUs. MPI for Python (mpi4py)[95] affords efficient parallel execution of Python scripts over multi-core and multi-node clusters, thanks to its bindings of the Message Passing Interface (MPI) standard. FedML[96] and FLower[97] further complement the federated learning landscape by enabling the development and deployment of federated learning algorithms and systems tailored to accommodate diverse computing environments and designed with flexibility for research and production purposes. Fed-BioMed[98], another open federated learning framework for empowering biomedical research, has many more medical domain-specific library extensions; however, no FHIR data support. The swarm-learning framework by Hewlett Packard Enterprise (HPE)[99] is also a promising decentralized, privacy-preserving ML framework. However, the license allows only non-commercial and experimental use. Furthermore, the HPE association may raise concerns about the framework's independence, similar to TFF with Alphabet Inc.

Despite the capabilities of these frameworks, a critical consideration emerges for their application in critical healthcare infrastructures, such as university hospitals: handling patient data. The need for more ready-to-use integration for such infrastructures, stemming from insufficient network monitoring, identity and access management, traceability, and data and algorithm monitoring, underscores the need for significant extensions. These enhancements are essential to transform these frameworks into fully functional platforms capable of addressing the stringent requirements of healthcare data management with security and privacy. Furthermore, frameworks are usually limited to a specific programming language.

2.3.2 Platforms

In the ascending order of complexity, a platform is the most complex, both structurally and in terms of execution. Software platforms are defined as the entire system of hardware components and computer code, which facilitates inter-system communication between hardware and provides the foundation for additional applications and functionalities to run[100]. The term 'platform' can also refer to the so-called Platforms-as-a-Service (PaaS). PaaS describes a final, deployed version of the product, where the user only uses the product that is deployed, and its services are maintained by a provider[101][102].

As mentioned in the introduction, the concept of the PHT is not novel, but a productive deployable version still needs to be provided. A recently published federated PHT architecture PADME[103] provides more details regarding the application but needs to provide a detailed security concept. Another federated PHT implementation, vantage6[104], mainly focuses on image analysis. Vantage6 utilizes R and Python clients, which require a technical understanding to use.

DataSHIELD[105], another open-source distributed learning framework, is more generic and extensible. In combination with the management tool Opal, it is a platform. However, DataSHIELD is limited to the R programming language. Only functions installed by each data provider can be executed, making it challenging to deploy the complex pipelines often required for genomics or imaging. Within our work in the task force distributed learning of the MII, Tübingen provided an operational and technical concept template to deploy the DataSHIELD platform at German university hospitals. Nevertheless, many sites still need help with using and maintaining it.

Kaapana[106] is an open-source platform designed to facilitate the deployment and scaling of data-driven applications, with a particular focus on medical imaging analytics. It is developed to support the processing, analysis, and even sharing of medical data, offering tools and services that are tailored to the needs of healthcare professionals, researchers, and IT developers working within the medical radiology field. Similar to the PHT concept implementation, Kaapana integrates a variety of technologies and frameworks to provide a comprehensive ecosystem for building and deploying medical imaging workflows and models. It is built with flexibility and scalability, enabling users to handle large volumes of medical data efficiently. The platform leverages containerization technologies, such as Docker, and orchestration tools, like Kubernetes, to facilitate the deployment of applications in a scalable and manageable manner.

The open-source federated learning framework Substra[107,108], developed initially by Owkin, which is now hosted by the Linux Foundation for AI and Data, is based on a fully decentralized distributed ledger, with productive deployment in hospitals[109] and has a centralized or decentralized orchestration. However, it is essential to note that Substra is mainly designed to work with ML models, and other types of analysis are not currently supported.

The Medical Informatics Platform (MIP) of the Human Brain Project (HBP)[110] is an open-source federated platform with comprehensive deployment and focus for neuroscience applications. The homepage and documentation of the platform were not available at the time of writing this dissertation, and the documentation was last updated three years ago. Therefore, it has to be assumed that the platform could be used in more promising, productive ways.

The Observational Health Data Sciences and Informatics (OHDSI)[111] is a multi-stakeholder, interdisciplinary collaborative to bring out the value of health data through large-scale analytics. OHDSI's mission is centered on improving health by empowering a community to collaboratively generate the evidence that promotes better health decisions and better care.

It involves a global network of researchers, clinicians, developers, and industry professionals working together to create and apply open-source tools and methodologies for observational health research. OHDSI utilizes a standardized data model (Common Data Model (CDM)[112]) to analyze disparate healthcare databases systematically. This model enables the harmonization of data from diverse sources, including electronic health records (EHRs), administrative claims, and registries, making it possible to conduct large-scale and replicable research across multiple databases. The collaborative develops and supports a suite of open-source software tools that leverage the CDM for conducting observational research. These tools include data characterization, population-level estimation, patient-level prediction, and more. Besides the general facing of distributed learning problems like data quality of source data, biases, and generalization of findings, the most significant drawbacks are the enforced CMD models. The focus on medical queries, for e.g., cohort selection and descriptive statistics, and the limitation on the R programming language are general criticisms of OHDSI.

Analytical capabilities

The second part of the evaluation assesses the analytical potential offered by each federated learning solution, as each implementation may prioritize different analytical methods for diverse use cases:

- **Data queries:** Refers to executing fundamental bio-statistical queries on distributed data. This functionality enables efficient, privacy-preserving access to essential statistics across participating hosts, supporting streamlined decision-making without needing raw data centralization.
- **General statistical methods:** Encompasses traditional statistical approaches such as generalized linear models, principal component analysis, and conventional machine learning algorithms. These methods support various analytical tasks within the federated environment, promoting insights without compromising data locality.
- **Deep learning methods:** Designed for more complex analytical demands, deep learning models can extract intricate patterns from distributed data. This capability is precious for solutions focused on high-dimensional, large-scale data analysis, where larger models improve the quality of trained models across federated nodes.

Data-type compatibility

Medical research relies on diverse data types and formats, each bringing unique integration challenges within federated systems. These challenges are not only technical, such as those related to storage, but also pertain to the distinct analytical approaches needed for each data type.

- **Clinical data formats:** Standard clinical workflows collect essential demographic and patient history information, including age, gender, weight, medical history, diagnostic test results, and prescribed medications. This data is commonly stored in tabular formats or structured according to dedicated healthcare data standards like FHIR, which facilitate interoperability in healthcare systems.
- **Omics data formats:** With advancements in Next-Generation Sequencing (NGS) technologies, omics data (e.g., genomics, transcriptomics) is increasingly generated for personalized diagnostics. This data requires extensive preprocessing and is typically available in specialized formats (e.g., FASTQ, VCF). Handling omics data in federated systems necessitates dedicated analytical tools to address its high-dimensional and complex structure.
- **Imaging data formats:** Imaging data is integral to modern diagnostics and spans various applications, each with specific formats and analytical needs. For example, RGB images, like those used in dermatology and ophthalmology for skin and retinal examinations, respectively, differ from radiological images such as CT and MRI scans, which are often in 3D or 4D formats. These formats require substantial storage and specialized algorithms to analyze spatial and temporal features.

Accessibility and interoperability

In federated learning, ensuring that solutions are accessible and interoperable across diverse environments is essential for seamless collaboration and broad usability. The following aspects address key considerations for creating adaptable and user-friendly federated solutions:

- **Technical dependencies:** The specific hardware or software requirements needed for a federated learning solution to operate effectively. High dependency on specialized components can limit accessibility and increase deployment complexity, particularly in multi-institutional settings with varied infrastructures.
- **Graphical user interface (GUI):** A well-designed GUI can enhance usability, allowing non-technical users to interact with the federated system easily. For large organizations and consortia, intuitive GUIs reduce the learning curve and facilitate broader adoption by enabling users without extensive technical backgrounds to manage and operate the system effectively.
- **Compatibility with other platforms:** Given the diversity of federated learning frameworks available, ensuring compatibility between different platforms supports collaborations across institutions and consortia. Interoperable solutions enable data exchange and shared analyses between institutions using distinct federated systems, fostering collaborative research without compromising data privacy.

Support

Effective support mechanisms are crucial for federated learning solutions' sustainable use and development. This section evaluates the types of support available to users, developers, and institutions implementing these solutions:

- **License:** The type of license under which the software is made available, ranging from various open-source licenses (e.g., MIT, GPL) to proprietary licenses. The licensing terms affect the freedom to modify, distribute, and commercialize the software, a critical consideration for institutions aiming for flexibility and compliance with regulatory standards.
- **Active support:** Indicates whether the software is actively maintained, with regular updates and a network of resources (e.g., forums, guides, documentation) to facilitate installation and usage. Active support helps users stay updated with the latest features, security patches, and troubleshooting advice, ensuring a smoother experience.
- **Developer outreach:** Availability of direct communication channels to connect with the developers. This can expedite problem resolution, enable feedback exchange, and foster collaboration, essential for specialized use cases or troubleshooting complex issues.
- **Community:** Examines the presence of an active user community, including the organization of meetings, workshops, or other collaborative events where members can share insights and development plans. A strong community enhances knowledge-sharing and supports the software's evolution through collective contributions and shared best practices.

Our evaluation of federated learning (FL) solutions reveals a diverse range of tools designed to meet different requirements in biomedical data science. We categorized these solutions into four main types: libraries, frameworks, platforms, and Platforms-as-a-Service (PaaS). A comprehensive overview of these solutions, integrating technical characteristics, analytical capabilities, data compatibility, security, and support information, is provided in Table 2.1.

Detailed tables summarizing the technical classification, analytical capabilities E.2, data compatibility E.2, security E.3 accessibility, interoperability E.4, and developer support E.5 of the federated learning solutions are provided in the supplementary material table section.

2.3.3 SMPC Solutions

Secure multi-party computation frameworks (SMPC) are essential for ensuring security and privacy in federated analyses, operating under malicious threat models to compute global results from fully encrypted input data. Traditionally, SMPC solutions have been domain-specific and highly specialized, with limitations in analysis functionalities and scalability due

Table 2.1: Comprehensive Overview of Federated Learning and Analysis Solutions: This table compares FL and analysis solutions, highlighting their technical characteristics, analytical capabilities, data compatibility, security measures, and support information. **Solution Type: L**ibrary, **F**ramework, **P**latform, Paa**S**. **Federation Structure: C**entralized, **D**istributed, **F**lexible. **Analytical Capabilities: D**ata **Q**ueries, **S**tatistical **M**ethods, **D**eep **L**earning. **Data Compatibility: Clin**ical, Omics, and **Imag**ing data. **Security & Privacy: A**ggregation, **D**ifferential Privacy, **H**omomorphic Encryption, **R**ole-Based Access Control, Secure **M**ultiparty Computation, Access **C**ontrol, **Z**ero-Knowledge **P**roofs, **Z**ero-Trust Architecture. **Infrastructure: L**inux, **m**acOS, **W**indows, **D**ocker, **C**loud. **Licenses: A**pache-**2**.0, **G**PL-**3**.0, **P**roprietary, **MIT** License, **Mix**ed, **O**pen **S**ource. This overview provides a detailed resource for selecting appropriate FL solutions tailored to specific research or application needs.

Name	Type	Struct.	DQ	SM	DL	Clin.	Omics	Imag.	Sec.	Infra.	GUI	Lic.
TensorFlow Federated	L	C	✗	✗	✓	✓	✗	✗	AC	LmW	✗	A2
FedML	L	F	✗	✗	✓	✓	✗	✓	AD	LmW	✗	A2
Fed-Biomed	F	C	✗	✗	✓	✓	✗	✓	AHR	L	✗	A2
Flower	F	F	✗	✗	✓	✓	✓	✓	ADR	LmW	✓	A2
Swarm Learning	F	D	✗	✗	✓	✓	✓	✓	ACZ	CL	✓	P
DataSHIELD	P	D	✓	✓	✓	✓	✓	✗	ACD	CLW	✓	G3
PHT-meDIC	P	C	✓	✓	✓	✓	✓	✓	MR	L	✓	MIT
PADME	P	D	✓	✓	✓	✓	✓	✓	MR	L	✓	MIT
Vantage6	P	F	✓	✓	✓	✓	✓	✓	CDH	DL	✓	A2
Substra	P	D	✓	✓	✓	✓	✓	✓	MPR	DL	✓	A2
OHDSI (ATLAS)	P	D	✓	✓	✗	✓	✓	✗	CR	CLmW	✓	A2
EasySMPC	P	D	✓	✗	✗	✓	✗	✗	CM	L	✗	MIT
OpenFL	P	C	✗	✗	✓	✓	✓	✓	ADR	LW	✓	A2
Kaapana	P	C	✗	✓	✓	✓	✗	✓	CR	DL	✓	A2
Med. Inf. Plat.	S/P	C	✓	✓	✗	✓	✓	✓	DHR	CL	✓	Mix
FeatureCloud	S	D	✓	✓	✓	✓	✓	✓	ADR	C	✓	A2
sfkit	S	C	✗	✓	✗	✓	✓	✗	AC	LmW	✓	A2
i2b2 tranSMART	P	C	✓	✓	✗	✓	✗	✗	C	CLW	✓	OS

to performance overheads. Recent advancements have significantly improved these scalability aspects[113,114].

Sharemind MPC[115], though proprietary and primarily focused on R statistics, represents an early approach to SMPC. More recent platforms, such as MedCO[116], leverage a combination of protocols and techniques[114,117] to enable secure analysis on distributed data, albeit with specific storage limitations to tranSMART[118]. When this thesis was written, a note on the MedCo homepage mentions that this version is no longer maintained and is only available for research purposes. The introduction of FAHME[119], utilizing lattigo[120] and offering advancements in fully homomorphic encryption, marks a significant step forward, although it still encounters limitations in analysis functionalities due to fully homomorphic encryption.

Other open-source SMPC solutions are becoming increasingly popular, offering the flexibility and extensibility required for various applications. These frameworks often support a variety of computational models and are designed to be more accessible to researchers and developers. Integrating machine learning algorithms into SMPC frameworks is a promising area of research. This allows for executing complex data analyses and predictive modeling on encrypted data without compromising privacy. Such capabilities are crucial for sensitive domains like healthcare or finance. The beginning of the integration of quantum-resistant algorithms within SMPC aims to future-proof these frameworks against potential quantum computing threats. Incorporating these algorithms ensures that encrypted data remains secure even as computational power advances. These topics underscore the dynamic nature of SMPC research and development, highlighting ongoing efforts to address the challenges of scalability, functionality, and security in the context of federated analyses.

EasySMPC[121] is a tool designed to make SMPC accessible through a user-friendly, no-code interface. This tool facilitates the secure summation of predefined sets of variables among different parties in a process involving two rounds of communication, input sharing, and output reconstruction without the need for traditional cryptographic key exchanges during setup or before computation. From a design perspective, EasySMPC is constructed with three main concepts: Studies, Participants, and Variables, guiding users through a process that includes two rounds of data exchange for secure computation. This process supports semi-manual and automated message exchanges, enhancing the flexibility and usability of the tool for various user preferences. It's designed to be robust against a high degree of potential corruption by participating parties, ensuring high data privacy and security throughout the computation process. EasySMPC represents a step towards democratizing the use of secure multi-party computation by lowering the barrier to entry for non-technical experts and fostering wider adoption in practical applications, especially in fields requiring confidentiality and data privacy. However, no ML training is possible besides secure forms or variable processing.

Regardless, these frameworks often demand a deep understanding of cryptographic principles and the specific security protocols they use. Additionally, tailoring SMPC systems to

specific applications requires intricate customization, making it difficult for non-experts to deploy them effectively. The need for significant computational resources and the potential for performance bottlenecks further complicate their straightforward application, especially in environments with limited IT infrastructure. To conclude, SMPC provides various desired functionalities of federated learning frameworks. However, platforms such as MedCO, which are strictly limited to only SMPC methods, make it less practical to the various medical data science needs.

2.3.4 The Need of a Flexible and Secure Platform

The discourse in preceding sections underlines the need for platforms that harmoniously combine functionality with security. Achieving such functionality necessitates a framework that is both flexible, to accommodate the dynamic requirements of data scientists, and general enough to process an extensive array of data inputs without having its capabilities shortened by security constraints. Nevertheless, this integration often poses a trade-off with security considerations or rather organizational than technical solutions.

In response to these requests, we developed PHT-meDIC, an open-source implementation of the Personal Health Train (PHT) concept, tailored for productive deployment within healthcare. PHT-meDIC distinguishes itself by providing comprehensive operational and technical documentation to simplify its deplyoment in hospital settings. The architecture of PHT-meDIC separates into central and local components, leveraging container technologies to prevent the need for local installation tasks, thus streamlining the distribution of complex analysis workflows.

Our approach contrasts markedly with other PHT solutions by rooting security within its design. We ensure the integrity of code executed at each node of the network through a robust security protocol, preventing tampering during transit or at any point of the process. Moreover, data encryption safeguards the confidentiality of results during transmission, accessible exclusively to the algorithm's author and the participating study sites. PHT-meDIC's design facilitates the iterative execution of analysis workflows and result updates by now. To augment user-friendliness, a user interface for code submission and detailed guides for the platform's setup, administration, and usage are provided, aiming to develop a solution that fully addresses the range of demands.

To further mitigate technical shortcomings, we embraced organizational strategies to bridge the gap. A 'human in the loop' mechanism has been implemented to mitigate the risk of executing malicious code within hospital IT infrastructures. This symbiotic combination of technical safeguards and organizational checks positions PHT-meDIC as a resilient, secure, and scalable platform for data processing and analytical tasks in healthcare settings, offering adaptability and customization that other platforms lack, especially in terms of programming

language or task flexibility. Moreover, while scalability is deemed a key attribute, automatic approval or execution of code from priviliged users can easily be set. Today this feature is only available on development and testing instances. Several existing scalable frameworks, including TensorFlow and Keras, often fall short of meeting the stringent governance and IT security prerequisites vital for clinical environments, a gap PHT-meDIC aims to fill effectively by integrating such frameworks within the platform.

Chapter 3

Methods

The content of Section 3.2.2 and 3.2.4 constitutes an extended version of the manuscript:

Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train (PHT-meDIC)[31]

The content in Section 3.2.5 constitutes an extended version of the manuscript:

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC[33]

The content in Section 3.2.7 constitutes an extended version of the manuscript:

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis[32]

The Method section defines the concept and individual service components of the PHT-meDIC, including an overall analysis workflow. Furthermore, it describes in detail the robust security protocol of the PHT-meDIC and how it provides security and privacy. Later, we describe the required algorithms to calculate the exact Area Under the Curve (AUC) in distributed settings in a privacy-preserving way (DPPE-AUC) and an approximation method (DPPA-AUC). Additionally, it offers insights into the range of input data processed and its adaptability across various presented use cases. The interoperability efforts with other PHT architectures close this section.

3.1 Introduction

The PHT is a paradigm for distributed healthcare data analysis proposed by the GO-FAIR initiative[15,38]. The basic concept of the PHT is that the analysis algorithm (wrapped in a

'train') travels between multiple sites (so-called 'train stations'), which securely host the data. Light-weight container-based virtualization is utilized to implement trains, enabling the rapid deployment of complex software pipelines (e.g., for genomics or image analysis) without additional software installation. These concepts have been described in the literature before[39–42], but production-ready implementations have been lacking so far.

Goals of the PHT-meDIC

In this work, we present a detailed software architecture for a PHT with a focus on security features and an open-source implementation of this architecture. This implementation is productively deployed at three German university hospitals. PHT-meDIC demonstrates how modern cloud techniques can be leveraged for complex, distributed, privacy-preserving medical data analytics. The key contributions of our work are a) an architecture and security framework, b) a semantically integrated data access mechanism based on HL7/FHIR, c) an implementation of data governance processes within this framework, d) two experiments demonstrating distributed machine learning (ML) on big data, e) a concept for data discovery demonstrated on actual patient data at hospitals and f) a freely available open-source implementation of these features.

Exemplifying secure privacy-preserving count queries shows how our PHT-meDIC architecture can be extended with additional data privacy-preserving methods.

To the best of our knowledge, this represents the first distributed decentral PHT implementation that ensures security, data privacy, and all required governance operational documents to deploy the platform productively at German university hospitals.

3.2 Materials and Methods

3.2.1 Concepts

Our proposed architecture is divided into central and local components. Our framework uses container technologies to enable the direct distribution of complex analysis pipelines, minimizing local administration. Compared to other proposed PHT solutions, our solution implements security by design. Our security protocol guarantees that the code executed at each train station cannot be manipulated in transit or during any other process stage. Additionally, the results are encrypted in transit to protect them from interception by anybody but participants conducting a specific study. The decryption of the result is only possible for the creator of the algorithm and participating study sites.

PHT-meDIC enables the secure analysis of distributed biomedical data, including a wide range of complex bioinformatics pipelines, training machine learning models such as deep

neural networks, or secure count queries and summary statistics. An analysis in the context of the PHT-meDIC is defined as arbitrary Python or R code submitted by researchers to be containerized and executed sequentially in distributed train stations, which in turn can execute additional tools (e.g., existing read mapping tools). The sensitive input data remains under controlled access by each participating site (see Figure 3.2) and does not leave the train station.

The main benefit of the PHT architecture compared to other federated analysis systems is its ability to transport rather complex pipelines consisting of many different tools to the sites without local software installation. PHT achieves this through the use of lightweight container-based virtualization. By now, the single point of communication between stations and PHT central services is reached utilizing docker pull and push commands. The security concept of the PHT-meDIC aims to minimize risks while still allowing rich functionality. We are, of course, aware of the inherent security risks implied by transporting arbitrarily complex software - which hence cannot be reviewed in full. PHT-meDIC implements technical security mechanisms at the train stations to mitigate those risks.

3.2.2 Software Architecture

The architecture of the PHT-meDIC consists of central and local services that enable iterative execution of analysis code wrapped in trains, as illustrated in Figure 3.1.

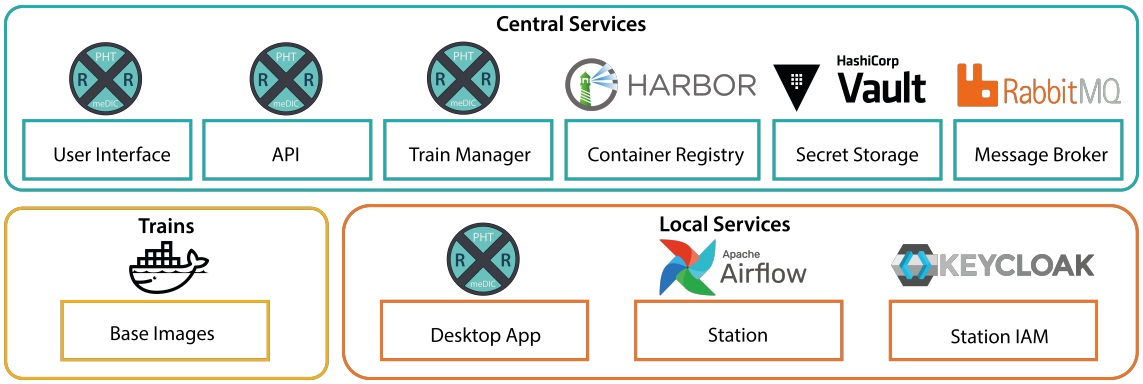


Figure 3.1: Open-Source and Self-Developed Service Components of the PHT-meDIC: This diagram highlights the existing open-source service components of the PHT-meDIC framework, represented by their respective service icons. Self-developed components are marked with the PHT-meDIC icon and are implemented in Node.js. Third-party services, such as Harbor, Vault, RabbitMQ, Docker, Apache Airflow, and Keycloak, are integrated into the framework and listed on the GitHub PHT-meDIC organization page. The architecture showcases the modular and extensible nature of the PHT-meDIC services.

The *User Interface* (UI) is the central entry point into our platform. Users can submit study requests (project proposals), analysis code for proposals, and manage participation in analyses. The *Desktop App* is an application installed on a user's PC that can be used to generate private and public key pairs. These key pairs can then be used to sign the contents of a train or

decrypt results cryptographically. The *Train Manager* (TM) summarizes several tasks: Train Building (TB), Train Routing (TR), and Result Extraction (RE). TB creates valid 'trains' based on the uploaded files and constructs the required configuration file to execute a 'train.' TR is the task of interacting with the Secret Storage and the *Container Registry* (CR) to make the submitted train available for each station. After successful execution, the RE makes the encrypted results available to download within the UI. The *API* is a REST interface to manage central resources and trigger processes between services. The CR stores and distributes 'train' (see 3.2.2) containers. We use the open-source registry Harbor[122] to provide this functionality. For storage of sensitive data such as train routes or submitted public keys of stations and users, we utilize the open-source tool Vault[123]. Internal communication between central services is based on the asynchronous open-source message broker RabbitMQ[124].

Trains are based on base images, which include all study-specific software dependencies and the train library. The train library includes the security protocol and domain logic of trains.

The station is a local service, based on Airflow[125], hosted locally at each study site, providing access to the local data in a secure environment and enabling the execution of trains under the security protocol. A station only requires communication with the CR to participate in the analysis. Encrypted results can be downloaded from the UI and decrypted using the *Desktop App* with the user's private key to obtain results locally.

The diagram (see Figure 3.2) represents a complex workflow likely related to software or data processing, with various components interacting. Here is a step-by-step description following the numbers in the diagram:

1. **Redirect Login**: The user starts by logging in through the Desktop Browser, which redirects them to a login service from his organization.

2. **Auth code**: The IAM service provides an authentication code after the login credentials are entered.

3. **Exchange Auth code for an access token**: The Browser exchanges the authentication code for an access token.

4. **Sign hash**: The Desktop App signs a hash for security protection to verify that the algorithm and patient query can not be manipulated during execution.

5. **Send task**: The User Interface sends a train-building task via a Message Broker to the Train Builder.

6. **Consume queue**: The Train Manager Building service consumes tasks from a Message Broker queue.

7. **Post route**: The Train Manager Building service posts a Train Route to a Secret Storage.

8. **Receive public keys**: The Train Manager Building service receives public keys from Secret Storage.
9. **Build and push train image**: A container image is built and then pushed to an incoming Container registry repository.
10. **Send train events**: Train events are sent as triggers for the next steps.
11. **Publish train events**: These events are published to be consumed by the Train Manager Routing service.
12. **Read route**: The Train Manager Routing service reads the route for the train from the Secret Storage.
13. **Switching repos of train**: The train's repository is switched between stations according to the route.
14. **Pull**: Station 1 pulls the necessary data or code from its repository (Station 1 Repo).
15. **Train execution at Station 1**: The train's entry point is executed in the local environment of Hospital 1, processing read-only station data.
16. **Push**: The updated results from Station 1 are signed, rebased, and pushed to the station's repository (Station 1 Repo) and tagged as executed.

    At this point, the results are moved from the Train Router to the next Repo. The following stations, 2 and 3, execute the train in the same way as in steps 14 to 16
17. **Pull results**: The final results are pulled from the Train Result manager in the Outgoing Repo.
18. **Return results**: The encrypted results are returned over the User Interface to download locally.
19. **Decrypt results**: The Desktop App decrypts the encrypted results for the user to view.

Throughout this workflow, various repositories (Incoming Repo, Station 1 Repo, etc.) serve as storage and management points for code or result data. Hospitals have their stations for data or code execution, each with its own IAM (Identity and Access Management) service, such as KeyCloak, and its data storage.

This workflow is a distributed train execution with the PHT-meDIC, where different "stations" (which are hospitals) handle the result updates. Using "train" terminology is an analogy to data or code flowing through a pipeline like a train on tracks. The system emphasizes security and compartmentalization using public keys, authentication tokens, and encrypted results.

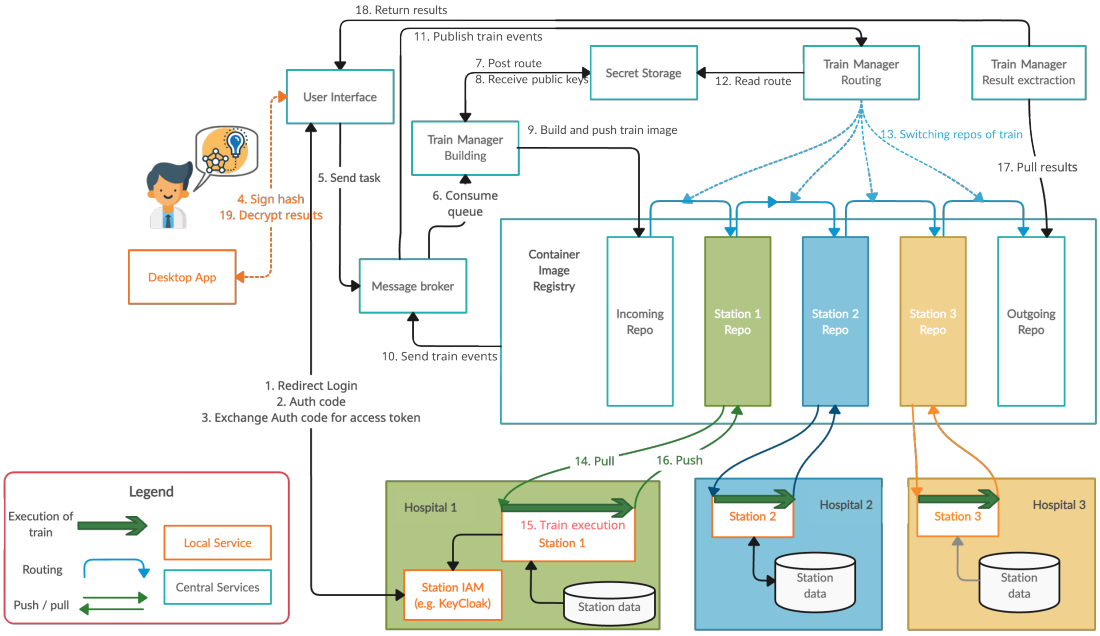


Figure 3.2: Comprehensive Overview of Central Service Interactions for Train Execution Across Three Stations: This diagram describes the interactions between central and local services required to execute a train over three stations (hospitals). *Green Arrows*: Represent the execution of trains locally at each hospital. *Colored Lines*: Show the updates to results within the central and hospital-specific service boxes. *Numbered Steps*: Indicate the sequential order of processes, from user login and task submission to train routing, execution, and result retrieval. The workflow illustrates the coordination between user interfaces, central services (e.g., image registry, message broker, secret storage), and hospital-specific components (e.g., Station IAM, data repositories) to enable secure and efficient distributed data processing.

Train components

A train is based on Docker[126] container technologies and is self-contained with all required software dependencies to run the analysis code. This enables local execution without any additional installation efforts. We provide a set of trusted base images in a publicly accessible repository containing all packages and libraries to execute the submitted algorithm. Suppose the existing base images do not fulfill the user's requirements. In that case, users can propose modifications to existing images or create new ones and request these new images to be approved as a base image. All base images are regularly scanned for software vulnerabilities. The user-submitted algorithm and FHIR search query are stacked on top of the base image. These files are static and do not change during execution. The train's results are encrypted at rest and updated iteratively during execution. The train image includes a configuration file containing the public keys of stations and users, the hash of static files, the result signatures of

previous executions, and the encrypted symmetric key. This file is used to perform the security protocol.

Train submission and execution

After acceptance of a study proposal by each station, the user uploads an algorithm and specifies the participating stations. During the submission process, the user is required to sign the hash of the uploaded algorithm using the Desktop App. After the train configuration is finished a building command is sent to the message broker. The TM consumes this building command, after requesting the algorithm from the API and obtains all required public keys from Vault, the train image is built and the route stored in Vault. The newly built train is pushed to the incoming train repository of the CR, only accessible by the TR. When the user issues the start command in the UI, the train is moved to the initial station repository based on the route stored in vault and is ready for executions. Station repositories can only be accessed by the respective station (and the TR for distribution). If a train is available, the station administrator *pulls* and executes the train using airflow. Required resources to run the analysis are specified in the station UI.

After successful execution, the station *rebases* and *pushes* the train back to the stations private repository in the CR. The TR process moves the updated train to the next station repository. **A train never stores any of the data provided for analysis - only encrypted results are stored and distributed**. Every station on the route follows the same procedure. When the train has reached the end of it's route, it is moved to the outgoing repository. The RE scans the project for successfully executed trains, extracts the encrypted results, and transmits these to the UI. The UI notifies the user, who can download the results and decrypt them locally using the Desktop App.

Train stations

Stations operate at each hospital based on Apache Airflow[125]. This open-source workflow management platform allows persistent and monitored execution of incoming trains as tasks defined as directed acyclic graphs (DAGs). The only required communication channel between stations and central services is the CR. Within the Airflow web interface, a station administrator can specify which train should be executed and provides additional execution configurations. The station administrator can set configurations (e.g., data sources, station identifiers, or keys) globally for all incoming trains or individually for each train. Any clinic's study system (e.g., FHIR, RedCAP, or tranSMART) is independently operated and can be used as a data source. After pulling a train from the CR, the station verifies that no static files were manipulated during transit. If there were previous executions of the train, the previous results and execution order are validated using the hash chain-based digital signature. After successfully verifying the

train, the encrypted previous results are decrypted, and the algorithm is executed using the provided data. The updated results are encrypted and stored in the outgoing train image. To overcome the limitation of 127 layers of a docker image[127], a rebasing is required to maintain a maximum of $n+1$ layers in the resulting image.

3.2.3 Input Data and Analysis Flexibility

As explained in chapter 2.2.6, typically, there is a trade-off between security and analysis capability by design. A significant focus of implementing our security concept was not to technically limit the analysis capability either by the input data that can be processed or the analysis functionality that can be executed. This significant flexibility is provided by using container technologies for trains.

Therefore, we only limit the functionality capacities on an organizational level. We define the programming languages or methods to execute in master images or within individual analysis requests upfront. Technically, anything is feasible as long as code reviewers do not detect any deviations from the ethics board or proposal approval. In the proposal, all sites must agree on what data and in which format will be provided for this project. The overall identical semantics or site-specific data preprocessing is a requirement. We assume a user has partial access to one site's data to develop analysis code. With this limitation, hospitals and individual institutes can agree upfront on the data and methods they want to examine without installing additional software locally. Trains can process tabular data, such as CSV format or structured FHIR data, and object data, like raw genome or MRT data. The operation mode Data Discovery 3.2.3 guarantees this requirement.

Since the MII has defined FHIR as a national standard for structured clinical data provisioning, the PHT-meDIC must support various FHIR server types. Depending on station and project-specific FHIR warehouses, the defined patient query may need to be translated to request data. To do so, we developed a library[53], which is included in the base image to translate the user-defined patient query to different FHIR servers. By doing so, we ensure that the provided data by each station is semantically interoperable.

Submission Modes

Each submission of analysis has to be approved by humans at each site, and execution of the analysis has to be done manually by each site. This human intervention and review is a deployment requirement of many German university hospitals. Since heavy analysis workflows could run several hours or even days, we want to ensure the analysis can be executed at all sites upfront. Therefore, we minimize errors from all sites during execution without large overheads. Consequently, we generally distinguish between two execution modes: *Data Discovery*

and *Data Analysis* (see Figure 3.3. In the following paragraphs, each mode is explained in detail.

Data Discovery

The concept of the *data discovery* is to ensure with minimal effort and execution time that the input data of each site is as expected for an analysis. Data discovery can be seen as an analysis before the analysis and is identical to an actual analysis train in the building and execution process. The minimal requirement of data discovery is that all specified data entries are available, but the complexity can increase. In practice, the first stage is to ensure that all specified data of the analysis can be loaded. By tabular data, this means the column names have the same name, and by image or genome data, the data objects can be read or accessed by the train at each site. The second stage of the discovery is to guarantee that the values of each accessed input data set are of the same type across all sites. This means, e.g., that each site uses the same data types, encoding, or syntax of specific data. The third and last stage is to provide details about the distribution, quality, or reports about the data itself. Patterns and trends between variables are also aimed at being discovered. This step examines the data to understand the characteristics of the data of each contributing site. For instance, class imbalances between input variables impact the prediction quality of the later executed[128,129] analysis. Alternatively, if varying sample sizes are reported, starting the analysis at the hospital makes sense, as it provides most subjects for the specific analysis. The last stage can also be seen as an explorative data analysis. Verifying these three steps upfront ensures that later stations are not blocking the previous station's hardware or human resources. A train will be terminated if it cannot run through all sites. This means station administrators would unnecessarily need to review the same train multiple times; thus, data discovery helps to save their time.

Data Analysis

After the user knows the data can be accessed equally at each site, the semantics of features match, and the algorithm considers possible data biases, the actual analysis or data science can start. Some model training, like deep neural networks for image classification tasks, can take several days of training while also requiring immense hardware resources, so previous data discovery is an important step. During the training of the models, varying hardware resource limitations may occur. For instance, one analysis could require large memory specifications while others require GPUs within the same proposal. Varying hardware resource availability and sample sizes will result in different execution times across all sites. After successful training, evaluating the performance of a model is crucial to ensure its accuracy and generalizability. Different performance evaluation strategies are beneficial based on the user's trained models. Sometimes, an independent holdout or cross-validation in different fold sizes is more helpful. Other models require error metrics such as accuracy and precision. As discussed in chapter 1 some metrics cannot be applied trivially in distributed settings.

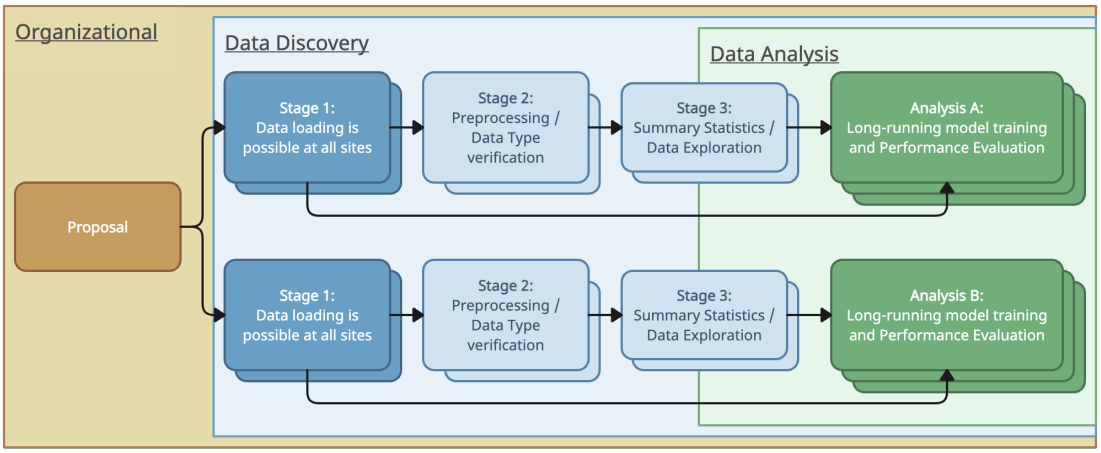


Figure 3.3: Data Discovery and Analysis Workflow: This diagram outlines the sequential process of data discovery (blue) and analysis (green), emphasizing the requirement of a proposal as the minimal prerequisite for initiating any workflow. Data Discovery: Stage 1 ensures data loading is possible across all sites and serves as a mandatory step before analysis. Stages 2 and 3, involving preprocessing, data type verification, and summary statistics or data exploration, can increase in complexity but are optional. Stage 3 is particularly sensitive as it generates results that may reveal individual patient data, positioning it between discovery and analysis. Data Analysis: Following discovery, analysis focuses on long-running model training and performance evaluation. Multiple boxes indicate different runs of analysis, while one proposal can encompass various types of analyses. This workflow illustrates the structured and iterative data discovery and analysis approach, ensuring compliance and comprehensive evaluation.

Overall Execution Workflow

The platform ensures data remains within its environment, following stringent regulatory and governance frameworks while allowing researchers to conduct complex analyses in distributed manners. This section outlines the PHT-meDIC platform's workflow and component interactions for the execution of the train (see Figure 3.4).

The PHT-meDIC platform is designed to start with an analyst's project request. An analyst initiates the process through the application's user interface, aiming to submit an analysis project proposal describing the aimed project goals, requirements, and associated studies. This request is a required starting point, which triggers a series of interactions across the platform's multi-faceted infrastructure. Authentication and authorization are managed sites independently on OpenID 2.0 supported IAM e.g., KeyCloak, a dedicated identity and access management service. This service confirms the analyst's credentials and privileges, providing a secure gateway to the subsequent stages of the workflow.

Upon submission, the request is directed toward a Central User Interface backend hosted within the deNBI computing environment. The proposal must be reviewed, commented on, approved, or rejected at each site. Following approval, a request for data provision is placed to

the Clinical Infrastructure, represented in figure 3.4 by the Tübingen meDIC data warehouse. This infrastructure is responsible for the stewardship and dissemination of clinical research data. The data manager, a crucial actor in the process, prepares the data for the analysis request. It exports data from a Master Warehouse and Station Data, loading it into a project-specific data warehouse and connecting the data source to the station.

The following steps describe technically how trains, independent of discovery or analysis mode, are built and distributed. The user uploads his analysis scripts, where the user interface calculates a unique hash of the analysis. This hashing process is pivotal, as it ensures the integrity and non-repudiation of the request throughout its lifecycle within PHT-meDIC. The analyst signs the hash with its private key in the desktop app on his computer locally. A Message Broker queues messages for processing and is an intermediary between the services and components. A train-building task is transmitted to the train-building service.

The Vault component is fundamental to the PHT-meDIC's security architecture. It manages cryptographic public keys, essential for securing data in transit and at rest, pseudo identifiers of projects and sites, and train routes. The Train Routing system is a metaphorical representation of the data's travels through the platform, ensuring trains reach their destination for processing.

The station data is linked to a Station Virtual Machine (VM), orchestrated by an Airflow managing system. Airflow provides the automation necessary for executing predefined tasks in an orderly and reliable manner. The data is processed within this VM, preparing it for analysis within the PHT-meDIC environment. Each station admin independently reviews and approves, or rejects the analysis requests.

Harbor manages and serves Docker images, encapsulating the analysis tools and environments. Interaction with a Public Registry offers an exchange with broader community resources for fetching community-endorsed tools or method examination for wider dissemination. The Private Registry is a passive inbox, storing station-specific trains and executed results, ensuring that only authorized sites can decrypt and execute the data.

The Station Admin is responsible for executing the trains and granting them credentials based on the local FHIR standards, a cornerstone for interoperability within the health data ecosystem. The admin pulls and executes the docker image with locally provided station read-only data. The Exec Train component is the execution environment where a decryption process is carried out before the data analysis process. Post-execution, the results are encrypted and extracted along with the train configuration, safeguarding the analysis's outcome. The updated result layer is pushed to the private registry. The train router moves labeled trains along their route and the following stations execute trains analogously. After extracting the encrypted results from the outgoing repository and downloading by the analyst, the workflow concludes with the analyst decrypting and evaluating the results, thus closing the loop of the analysis request. The encrypted status and outcome of the analysis are made available to the analyst, maintaining the confidentiality and integrity of the data throughout the process.

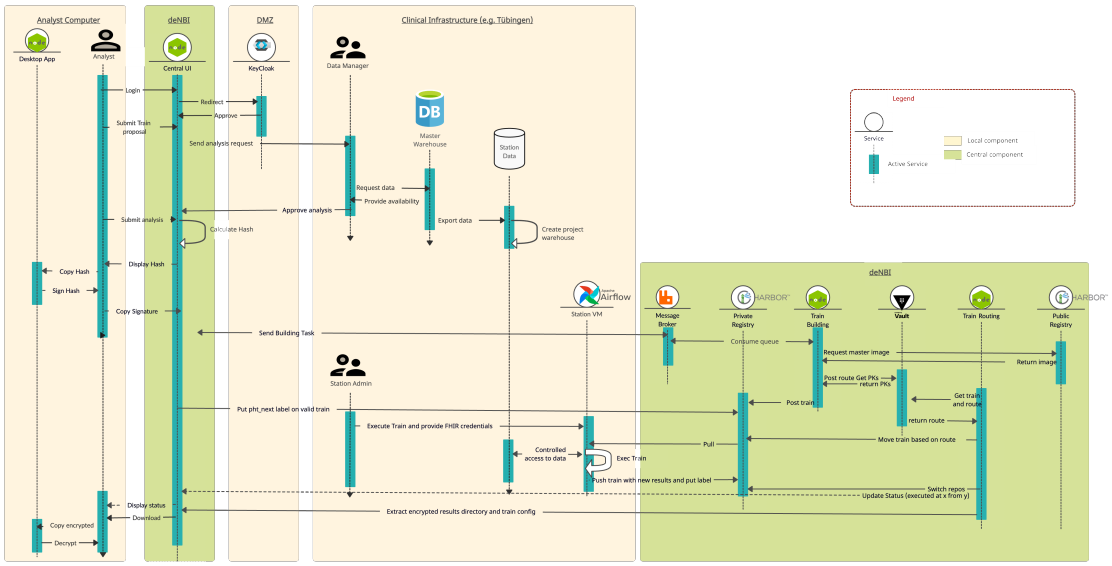


Figure 3.4: Flowchart of Analysis and Interactions Between PHT-meDIC Service Components: This diagram illustrates the end-to-end workflow for building and executing a train at one station within the PHT-meDIC framework. *Green Boxes*: Represent centrally hosted components, such as the central UI, TM, and registry services. *Pastel Orange Boxes*: Indicate components within the user's or clinic's domain, including the desktop application and clinical data infrastructure. The workflow begins with the user submitting and signing a train proposal, which is processed and built centrally. The station administrators manually execute the train at the clinical station, ensuring controlled access to data. Once execution is complete, encrypted results are pushed back to the central registry, and users decrypt the results locally for analysis. This flow emphasizes the secure and modular interaction between central and local service components.

3.2.4 Security concept

The highly sensitive nature of medical data, i.e., mental health or genetic data, demands the highest effort in keeping this information private. The security protocol of the PHT-meDIC aims to provide security and privacy without restricting functionality. We consider the following threat models for our services: *Container registry* and the *user* can delete, modify, or change parts of the train to obtain sensitive information. Therefore, they are considered malicious. The *stations* are considered to be honest-but-curious. They can attempt to infer sensitive information from results stored in the train. The *secret storage* Vault and the *Train Manager* (train building, train routing, and result extraction) are assumed to be trusted parties. TM can alter user-defined parameters, yet it does not have direct access to station data. Accessing results necessitates knowledge of the station's private keys, but it only knows their public keys. To gain access to previous stations' results, TM can create a fake station that knows both the public and private keys. However, this manipulation is thwarted by the fact that each station verifies the public key of other stations through public key certificates. Creating a fraudulent

public key certificate for the fake station would be unfeasible without access to the CA private key. As a result, TM's attempt to create a fake station and access the results is thwarted, as stations can detect such an attack by verifying public key certificates. Several security features ensure that only registered users have access and that only reviewed and approved trains can be executed. Furthermore, our security protocol guarantees that the submitted code from a user remains unchanged during the whole execution and verify all previous executions of the train. A combination of symmetric and asymmetric encryption, known as envelope encryption, allows for accelerated encryption and decryption of results. As an additional security measure, trains do not have network access and never directly access data from local sources. Read-only access to the input data is provided to the train by the station.

An honest-but-curious station can access the results of previously visited stations and the route. A station can perform model inversion attacks[130] or membership inference attacks[131]. However, these attacks rarely succeed on models with a high-dimensional input space[132]. Calculating simple count queries over patient data may provide an attacker enough information to re-identify patients or hospitals[133,134]. To solve this, the train library supports paillier homomorphic encryption[135]. Other, more sophisticated privacy-preserving methods, such as Cheon-Kim-Kim-Song (CKKS)[136], can be integrated into the train library. Stations can also compute statistics based on the configuration file, such as the number of times a specific station was involved in the previous analysis. However, since only pseudo identifiers are present in the train, they are unlikely to be matched to actual identifiers. The main reason stations are also honest-but-curious is that they have to use actual patient input data during train execution. Apart from this requirement, stations can be assumed to be malicious. They can modify the train by changing the algorithm for malicious purposes. However, such modifications will be detected by the next station, preventing any modified trains from being executed. Without having access to the user's private key, no station can modify the train content or signature without being detected by the pre-run protocol (1).

A malicious container registry (CR) can access all station repositories and images, and CR can move an image from one repository to another. Stations detect this change by verifying the route using the hash chain-based digital signature created by each previous station. Stations detect any modification on the algorithm, the route, and the query because it is impossible to generate a valid user signature for malicious parties. Any changes to the encrypted partial results done by CR are also detected because stations digitally sign their intermediate results using their private key. Furthermore, it is not possible to change the image for CR, such as adding malicious software. The base image is taken from the list of trusted images which is prepared by governance. Thus it is simple to detect any changes in the image using docker technology.

Even in the case of collusion between the user, CR, and stations, it is not possible to change the algorithm, query, route, or docker image because the algorithm, query, and route are signed

by the trusted TM, and the list of trusted images is stored in TM. If sites collaborate and do not provide patient data, they can infer individual sites' actual data.

Security Protocol

Preliminaries of the security protocol

For the security concept the following definitions of asymmetric encryption function, symmetric encryption function and hash function are used:

- ENCP() is an asymmetric function that encrypts a message M using a RSA public key P.
- DECP() is an asymmetric function that decrypts a ciphertext C encrypted with a PK using corresponding RSA private key SK.
- ENCS() is a function that encrypts a message M under the given symmetric key K.
- DECS() is a function that decrypts a ciphertext C under the given symmetric key K.
- HASH() is a cryptographic hash function which takes a message of an arbitrary length and produces a fixed length output. For a given output y_i, it is computationally infeasible to find an input x_i satisfying $\mathsf{HASH}(x_i) = y_i$.
- SIGN() is a function that cryptographically signs the message M using a private key SK of a sender, the output can not be replicated without the private key and the content of M verified
- VRFY() is a function that takes three inputs: a message M, a ciphertext C which is the encryption of $HASH(M)$ with SK, and a public key PK of a sender. It returns $\mathsf{HASH}(M) \stackrel{?}{=} \mathsf{DECP}(C, PK)$.

Security protocol assumptions

- The train registry can not change the predefined order of the stations participating in the analysis. It can not access the unencrypted FHIR search query. It can not access the intermediary results and the final result of the analysis.
- Data stations have to manually review and accept the algorithm and query. We assume that they are malicious parties. An adversary compromising only one data station can access the intermediary results and the final result. The adversary can not access the data of other stations.
- The user is malicious. If she or he collaborates with the private registry, the registry can access the query or intermediate results.

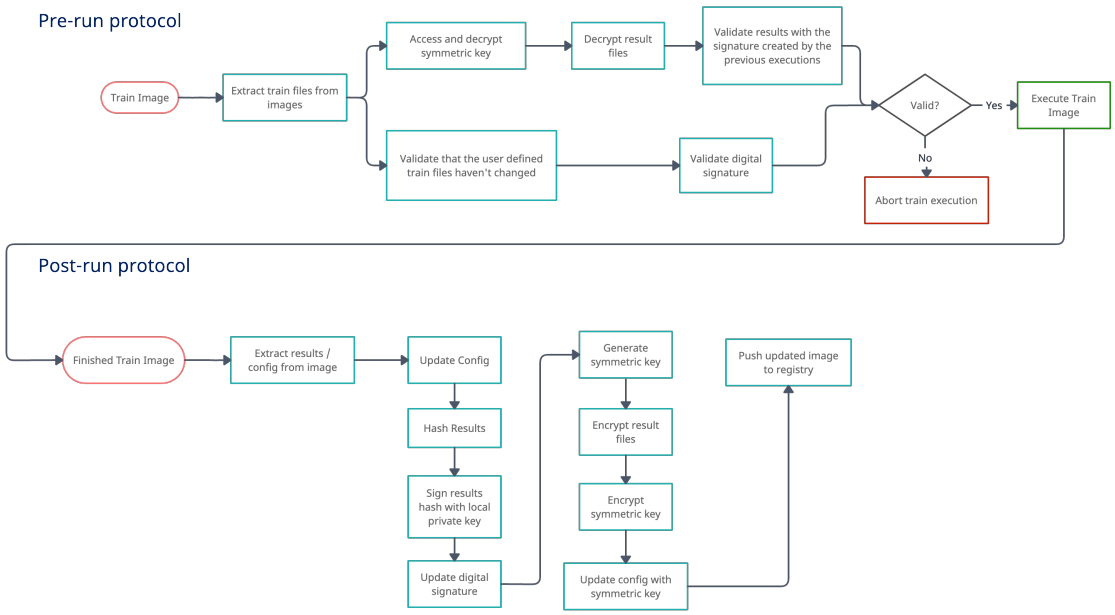


Figure 3.5: Security Protocol of the PHT-meDIC: Pre-Run and Post-Run Phases The security protocol consists of two distinct phases: *Pre-Run Phase*: Train files are extracted, decrypted, and validated to ensure no manipulation has occurred. Digital signatures and previous execution results are verified. If any manipulation is detected, the train execution aborts. *Post-Run Phase*: After successful train execution, results are re-encrypted, new hashes are generated, and the digital signature is updated. The encrypted results and updated configuration are then pushed to the registry, ensuring integrity and confidentiality for future processes.

Table A in the appendix defines the notations used throughout the thesis and security concept.

Train proposal and submission

The following steps are required to submit secured trains:

1. A user U uses his private credentials from his local governed identity management system to log into the central UI to create a train. U locally creates initially a private SK_U and public PK_U key pair using the Desktop App. He uploads his PK_U to the central UI. TB has a private SK_{TB} and public PK_{TB} key pair to sign users signatures.

2. The central service will upload the public key to vault.

3. U asks all stations to accept or reject a train proposal. All stations have unique pseudo identifiers (PIDs) within the PHT-meDIC, only the UI knowing the matching real identifier.

4. Based on the train proposal approval, U specifies the included stations for his train.

5. U provides algorithm A, query Q and included stations S_i in the UI. UI matches the station's ID to the corresponding $PIDs$ and sends the TB the $PIDs$, provided A and the public key of the user PK_U.

6. TB pulls one of the trusted base images I of the train from the public read-only accessible section of the container registry CR.

7. TB generates the route $R = (PID_{S_1}, PID_{S_2}, ..., PID_{S_j})$ and posts R to vault and receives the public keys of the stations PK_S.

8. TB picks a random number $N \in \{0,1\}^l$ that is the session id of the analysis.

9. TB picks a random number $K_U \in \{0,1\}^l$ that is the session key of the analysis.

10. TB calculates the hash (Equation 3.1) of $(ID_U||A||Q||R||N)$ where A is the algorithm, Q is the query and ID_U is the identifier of U.

$$H_U = \mathsf{HASH}(ID_U||A||Q||R||N) \tag{3.1}$$

11. TB provides U with H_U

12. U signs H_U (Equation 3.2) with its private key SK_U using the Desktop App and returns the signature $\mathcal{SIG}_{H_U}$ to the U, which will forward it to TB.

$$\mathcal{SIG}_{H_U} = \mathsf{SIGN}(H_U, SK_U) \tag{3.2}$$

13. TB encrypts (Equation 3.3 and 3.4) Q and A with the symmetric session key K_U.

$$\mathcal{E}_Q = \mathsf{ENCS}(Q, K_U) \tag{3.3}$$

$$\mathcal{E}_A = \mathsf{ENCS}(A, K_U) \tag{3.4}$$

14. TB signs $\mathcal{SIG}_{H_U}$ (equation 3.5) with its private key SK_{TB}.

$$\mathcal{SIG}_{TB} = \mathsf{SIGN}(\mathcal{SIG}_{H_U}, SK_{TB}) \tag{3.5}$$

15. TB encrypts (Equation 3.6) K_U with the public key of the all stations in R.

$$\mathcal{E}_{K_U} = \{\mathsf{ENCP}(K_U, PK_{S_1}), \mathsf{ENCP}(K_U, PK_{S_2}), ..., \mathsf{ENCP}(K_U, PK_{S_j})\} \tag{3.6}$$

16. TB builds an image I_0 and writes $(ID_U, \mathcal{E}_A, \mathcal{E}_Q, R, N, \mathcal{E}_K, \mathcal{SIG}_H, \mathcal{SIG}_{TB})$ within. I_0 is named by a universally unique identifier (UUID) generated by the UI logic.

17. TB pushes I_0 to the incoming repository of the CR.

Train execution

The previous valid built train can be executed by stations with these steps:

1. The train routing TR process reads the route from vault and moves the image I_0 to the corresponding station repository.
2. The data station S_i pulls I_i from it's own station repository.
3. S_i checks whether I_i was built using one of the trusted base images.
4. S_i starts the pre_run protocol (algorithm 1) by running I_i

Algorithm 1 *pre_ run()* protocol

procedure PRE_RUN(ID_{S_i}, SK_{S_i})
  $Q = \mathsf{DECS}(\mathscr{E}_Q, K_U)$
  $A = \mathsf{DECS}(\mathscr{E}_A, K_U)$
  valid $= \mathsf{VRFY}(ID_U||A||Q||N, \mathscr{SIG}_{H_U}, PK_U, PK_{TB})$
  if not *valid* **then** abort train
  if *first station* **then**
    run train
  else
    for $j = i-1, i-2, \ldots, 1$ **do**
      valid $= \mathsf{VRFY}(DS_j[0], DS_j[1], PK_j)$
      if not *valid* **then** abort train
    $K_{i-1} = \mathsf{DECP}(\mathscr{E}_{K_{i-1}}[i], SK_{S_i})$
    $D_{i-1} = \mathsf{DECS}(\mathscr{E}_{D_{i-1}}, K_{i-1})$
    valid $= \mathsf{VRFY}(\mathscr{D}_{i-1}||N, \mathscr{SIG}_{H_{D_{i-1}}}, PK_{S_{i-1}})$
    if not *valid* **then** abort train
    run train

5. S_I rebases decrypted results D_{i-1} and executes train container C_{I_1}
6. S_I obtains results D_i and runs the $post\_run$ protocol (algorithm 2)

Algorithm 2 *post_ run()* protocol

procedure POST_RUN(D_i, ID_{Si}, SK_{S_i})
  $H_{D_i} = \mathsf{HASH}(\mathscr{D}_i||N)$
  $\mathscr{SIG}_{H_{D_i}} = \mathsf{SIGN}(H_{D_i}, SK_{S_i})$
  $K_i = \{0,1\}^l$
  $\mathscr{E}_{\mathscr{D}_i} = \mathsf{ENCS}(\mathscr{D}_i, K_i)$
  $\mathscr{E}_{K_i} = \{\}$
  for $j = 1, 2, \ldots, n$ **do**
    $\mathscr{E}_{K_i}[j] = \mathsf{ENCP}(K_i, PK_{S_j})$
  $\mathscr{E}_{K_i}[n+1] = \mathsf{ENCP}(K_i, PK_U)$
  if *first_station* **then**
    $DS_i = \mathsf{SIGN}(\mathsf{HASH}(\mathrm{N}), PK_{S_i})$
  else
    $DS_i = DS_{i-1}||\mathsf{SIGN}(\mathsf{HASH}(DS_{i-1}), PK_{S_i})$
  return $\mathscr{E}_{\mathscr{D}_i}, \mathscr{E}_{K_i}, DS_i, \mathscr{SIG}_{H_{D_i}}$

7. S_i writes $\mathscr{E}_{\mathscr{D}_i}, \mathscr{E}_{K_i}, DS_i$ to C_{I_0}. S_i commits C_{I_0} and creates an image I_i. I_i is pushed to the stations own repository.

8. TR moves the the successfully executed train of S_i to the next stations repository according to the route.

The PHT-meDIC security concept is developed to ensure the integrity and confidentiality of data within the platform. It employs encryption to secure result data when not in use - even centrally - and utilizes mechanisms to detect manipulation with algorithms, patient queries, or results. Envelope encryption enhances the performance of encryption and decryption processes, making them swift and efficient. Importantly, this security framework is implemented not to restrict any analytical functionalities of the platform, maintaining full operational capability while ensuring robust security measures. Privacy is added by using pseudo identifiers for users and stations and allowing additional methods, such as Paillier encryption within the train-container-library.

3.2.5 PP-AUC Calculation

Preliminaries of the protocol

Assume we have n participating stations (S_1, S_2, ..., S_n) and a initialization and proxy station S_0 and S_P in the network. The initialization station S_0 and the proxy station S_P do not participate in the analysis but act as an intermediary for key creation and secure computations. Each station holds its local data, and the machine learning model has already produced prediction values and randomly assigned flag subjects for privacy preservation.

Let M be the number of samples, and let the masked prediction values at each station be denoted as $\{pre\}$, where the predictions are masked as $\{pre\} = (r_1 \cdot pre) + r_2$, with r_1 and r_2 being randomly generated values to obscure the real prediction data.

Our protocols utilize the following functions: asymmetric encryption, symmetric encryption, symmetric key generation, Paillier encryption, and random number generation. These functions are defined as follows:

- RANDS(): Generates random integers.
- SYMM(): Generates a symmetric Fernet key for a station S_i.
- ENC(M, K): Encrypts a message M using a key K, which can be an RSA, Paillier, or symmetric key.
- DEC(C, K): Decrypts a ciphertext C using a key K, which can be an RSA, Paillier, or symmetric key.
- ADD(C_1, C_2): A homomorphic function that performs the addition of ciphertexts C_1 and C_2, encrypted with the same public key PK.
- ADD(C, M): A homomorphic function that performs the addition of a ciphertext C and a plaintext M. The plaintext M is first encrypted using a public key PK.
- MUL(C, M): A Paillier function that multiplies a plaintext M with a ciphertext C under a public key PK.

DPPE-AUC

In this section, we introduce our distributed privacy-preserving method for exact AUC computation, referred to as DPPE-AUC, which has been seamlessly integrated into the PHT-meDIC platform. **Initialization**
Each station is equipped with an RSA public and private key pair (PK_{S_i}, SK_{S_i}). The RSA public key of every station is accessible to all other stations within the network. Station S_0 generates a public and private key pair (PK_P, SK_S) for a modified Paillier cryptosystem. The private key SK_S is randomly divided into two parts, SK_{S_0} and SK_{S_1}, such that $SK_S = SK_{S_0} + SK_{S_1}$.

The public key PK_P is distributed to all stations. The split private key SK_{S_0} is securely transmitted to the proxy station S_P, while SK_{S_1} is securely sent to each station S_i, where $i \in \{1, 2, \ldots, n\}$. The distribution of these keys is performed securely by encrypting them with the RSA public keys of the respective stations.

This initialization process is carried out using a train (image) updated by station S_0, which sequentially visits all stations along the route (S_1, S_2, ..., S_n, S_P).

Client Computation

During the execution of Algorithm 3, each station generates a symmetric key k_i. The first station generates a random value r_1, used to obscure the prediction values by multiplication. At each station, an additional random component r_2 is calculated as $r_2 = (pre \mod r_1)$, further masking the prediction values. These masked predictions are encrypted using the symmetric key k_i, which is encrypted with the proxy station's public key PK_{S_p}.

The random value r_1 is encrypted by the RSA public keys of stations other than S_0 and S_P. Additionally, binary labels and flag values $\in \{0, 1\}$ are encrypted using Paillier public key PK_P. The encrypted results are stored within the train image and passed to the next station in the route. The subsequent stations decrypt $\langle r_1 \rangle$ using their RSA private key. The process of obfuscating prediction values and encrypting labels and flags continues analogously across all stations.

Algorithm 3 *dppe-auc* Station Protocol - Step I

procedure DPPE-AUC_STATION(ID_{S_i}, SK_{S_i})
  n is the number of stations
  m is the number of predictions confidences
  Step 1: Symmetric Key Generation
  $K_i \leftarrow \mathsf{SYMM}(ID_{S_i})$
  Step 2: Generation of Random Values
  if S_i is the first station **then**
    $r_1 \leftarrow \mathsf{RANDS}()$
    for $j = 2$ to n **do**
      $\langle r_1 \rangle_j \leftarrow \mathsf{ENC}(\langle r_1 \rangle, PK_{S_j})$
  else
    $r_1 \leftarrow \mathsf{DEC}(\langle r_1 \rangle_i, SK_{S_i})$
  Step 3: Encrypt Predictions, Labels and Flags
  for $j = 1$ to m **do**
    $tmp \leftarrow r_1 \cdot pre[j] + (pre[j] \mod r_1)$
    $\langle pre[j] \rangle \leftarrow \mathsf{ENC}(tmp, K_i)$
    $\langle label[j] \rangle \leftarrow \mathsf{ENC}(label[j], PK_P)$
    $\langle flag[j] \rangle \leftarrow \mathsf{ENC}(flag[j], PK_P)$
  Step 4: Encrypt Symmetric Key for Proxy
  $\langle K_i \rangle \leftarrow \mathsf{ENC}(K_i, PK_{S_p})$

Proxy computation

The proxy station retrieves the train image and executes Algorithm 4. First, it decrypts the symmetric keys ($\langle K_1 \rangle$, $\langle K_2 \rangle$, …, $\langle K_n \rangle$) using its private asymmetric key SK_{S_p}. The masked prediction values from all stations are decrypted and concatenated into a single table along with the corresponding encrypted labels and flags. S_P then sorts this table according to the

masked prediction confidences while keeping the actual prediction values hidden, ensuring that it does not access the real input data.

Based on the Paillier-encrypted labels and flag values, the proxy computes the encrypted true positive $\langle TP \rangle$ and false positive $\langle FP \rangle$ values. For each masked prediction value $pre[i]$, the proxy sums all labels greater than $pre[i]$ to calculate $\langle TP[i] \rangle$. The corresponding $\langle FP[i] \rangle$ value is determined by subtracting $\langle TP[i] \rangle$ from the sum of the flag values greater than $pre[i]$.

Due to the limitations of Paillier encryption, the proxy cannot directly perform multiplications on encrypted values. Instead, it delegates these computations to the stations. However, to prevent stations from inferring private information about other participants, we apply randomized encoding techniques to mask both the numerator (N) and denominator (D) values.

For the denominator D, the proxy computes the sum of all labels as $\langle TP_A \rangle$ and the sum of all flag values minus $\langle TP_A \rangle$ (Equation 3.7) as $\langle FP_A \rangle$. These values are multiplied by randomly generated constants a and b, respectively, using Paillier multiplication. The results are then used to compute the randomized encoding components of the denominator: $\langle D_1 \rangle$, $\langle D_2 \rangle$, and $\langle D_3 \rangle$, with random integers r_A^1 and r_A^2:

$$\begin{aligned} \langle D_1 \rangle &= \mathsf{ADD}(\langle TP_A \rangle, r_A^1) \\ \langle D_2 \rangle &= \mathsf{ADD}(\langle FP_A \rangle, r_A^2) \\ \langle D_3 \rangle &= \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle TP_A \rangle, r_A^2), \mathsf{MUL}(\langle FP_A \rangle, r_A^1)), r_A^1 \cdot r_A^2) \end{aligned} \tag{3.7}$$

For the numerator N, the proxy computes the difference in successive false positive values as $\langle dFP[i] \rangle = \langle FP[i] \rangle - \langle FP[i-1] \rangle$ and the sum of successive true positive values $\langle sTP[i] \rangle = \langle TP[i] \rangle + \langle TP[i-1] \rangle$ across all threshold values (Equation 3.8). All $\langle sTP[i] \rangle$ and $\langle dFP[i] \rangle$ values are multiplied by a and b, respectively. To handle ties in prediction values, we use the method from[137] to compute unique indices for the masked predictions. The proxy generates three random values r_i^1, r_i^2 and z_i for each index i. All z_i values are summed to zero. These random values are used to compute the randomized encoding components for the numerator:

$$\begin{aligned} \langle N[i]_1 \rangle &= \mathsf{ADD}(\langle sTP[i] \rangle, r_i^1) \\ \langle N[i]_2 \rangle &= \mathsf{ADD}(\langle dFP[i] \rangle, r_i^2) \\ \langle N[i]_3 \rangle &= \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle sTP[i] \rangle, r_i^2), \mathsf{MUL}(\langle dFP[i] \rangle, r_i^1)), r_i^1 \cdot r_i^2) \end{aligned} \tag{3.8}$$

To improve runtime efficiency, the sum of all $\langle N[i]_3 \rangle$ values is computed as $\langle N_3 \rangle$. The proxy station then partially decrypts all random components $\langle D_1 \rangle$, $\langle D_2 \rangle$, $\langle D_3 \rangle$, $\langle N[i]_1 \rangle$, $\langle N[i]_2 \rangle$, and $\langle N_3 \rangle$, where $i \in \{1, 2, \ldots, m\}$, using its partial private key SK_p^1. These components are written to the train results, and the proxy pushes the updated image back to the stations.

Final Computation

Each station S_i pulls the updated image and extracts the partially decrypted random compo-

Algorithm 4 *dppe-auc* Aggregation Protocol

procedure DPPE-AUC_PROXY(ID_{S_i}, SK_{S_p})
  Step 1: Decrypt masked prediction confidences
  Step 2: Sort Paillier-encrypted labels and flags according to masked prediction confidences
  Step 3: Compute True Positives (TPs) and False Positives (FPs) for a unique set of masked prediction values
  Step 4: Compute Denominator Components
  $a, b, r_A^1, r_A^2 \leftarrow \mathsf{RANDS}()$
  $\langle TP_A \rangle \leftarrow \mathsf{MUL}(\langle TP[last] \rangle, a)$
  $\langle FP_A \rangle \leftarrow \mathsf{MUL}(\langle FP[last] \rangle, b)$
  $\langle D_1 \rangle \leftarrow \mathsf{ADD}(\langle TP_A \rangle, r_A^1)$
  $\langle D_2 \rangle \leftarrow \mathsf{ADD}(\langle FP_A \rangle, r_A^2)$
  $\langle D_3 \rangle \leftarrow \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle TP_A \rangle, r_A^2) + \mathsf{MUL}(\langle FP_A \rangle, r_A^1)), r_A^1 \cdot r_A^2)$
  Step 5: Compute Numerator Components
  $\langle N_3 \rangle \leftarrow 0$
  for $i = 0$ to m **do**
    $r_i^1, r_i^2 \leftarrow \mathsf{RANDS}()$
    $\langle sTP[i] \rangle \leftarrow \mathsf{MUL}(\mathsf{ADD}(\langle TP[i] \rangle, \langle TP[i-1] \rangle), a)$
    $\langle dFP[i] \rangle \leftarrow \mathsf{MUL}(\mathsf{SUB}(\langle FP[i] \rangle, \langle FP[i-1] \rangle), b)$
    $\langle N[i]_1 \rangle \leftarrow \mathsf{ADD}(\langle sTP[i] \rangle, r_i^1)$
    $\langle N[i]_2 \rangle \leftarrow \mathsf{ADD}(\langle dFP[i] \rangle, r_i^2)$
    $\langle N[i]_3 \rangle \leftarrow \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle sTP[i] \rangle, r_i^2) + \mathsf{MUL}(\langle dFP[i] \rangle, r_i^1)), r_i^1 \cdot r_i^2 + z_i)$
    $\langle N_3 \rangle \leftarrow \mathsf{ADD}(\langle N_3 \rangle, \langle N[i]_3 \rangle)$
  Step 6: Partially Decrypt Components
  $(\langle N[i]_1 \rangle, \langle N[i]_2 \rangle) \leftarrow \mathsf{DEC}((\langle N[i]_1 \rangle, \langle N[i]_2 \rangle), SK_{P_1})\ i \in \{1, 2, \ldots, m\}$
  $(\langle D_1 \rangle, \langle D_2 \rangle, \langle D_3 \rangle, \langle N_3 \rangle) \leftarrow \mathsf{DEC}((\langle D_1 \rangle, \langle D_2 \rangle, \langle D_3 \rangle, \langle N_3 \rangle), SK_{P_1})$

nents and fully decrypts them using its private key SK_1. Station S_i can then locally compute the DPPE-AUC using the following equation:

$$\text{DPPE-AUC} = \frac{\sum_{i=1}^{M} \left(\langle N[i]_1 \rangle \cdot \langle N[i]_2 \rangle \right) - N_3}{2 \times (D_1 \cdot D_2 - D_3)} \tag{3.9}$$

It is impossible to eliminate the randomness present in the fully decrypted values for the stations. The only method to completely remove all randomness from these values is to calculate the AUC using Equation above.

DPPA-AUC

In this section, we provide a detailed explanation of our approximate AUC computation method, referred to as DPPA-AUC.

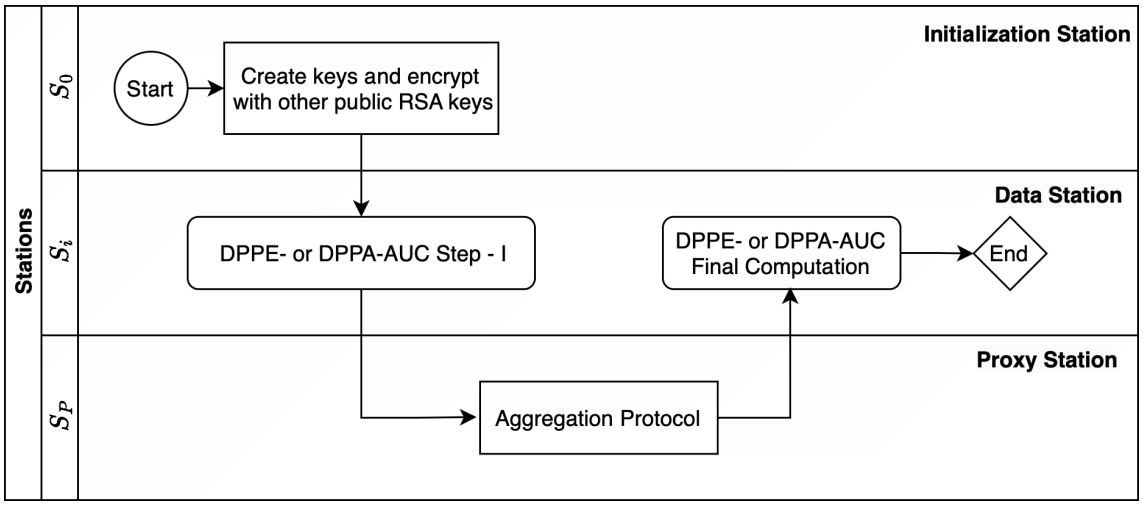


Figure 3.6: Overview of the DPPE- and DPPA-AUC Process: The workflow involves three types of stations: the initialization Station (S_0), Data Stations (S_i), and Proxy Station (S_P). The process starts at the Initialization Station with Paillier key generation and encryption using RSA public keys. Data Stations perform DPPE- or DPPA-AUC Step I, followed by an Aggregation Protocol at the Proxy Station. The results are then processed in final computation at the Data Stations, completing the protocol.

The use of approximate AUC for privacy-preserving model performance evaluation in distributed settings was first proposed by Sun et al.[138]. Sun et al. employed differential privacy to compute the AUC score of a global model in a federated environment. Although DPPA-AUC adopts the same underlying approximation technique, rather than differential privacy, it uses Paillier cryptosystem and randomized encoding to compute the approximate AUC value within the PHT-meDIC platform.

Initialization

The initialization phase of DPPA-AUC is identical to the DPPE-AUC. Each station is equipped with an RSA public and private key pair (PK_{S_i}, SK_{S_i}), and the RSA public key of every station is accessible to all others. Station S_0 generates a modified Paillier cryptosystem key pair (PK_P, SK_S) and splits SK_S into SK_{S_0} and SK_{S_1}, distributing these securely via RSA encryption. The public key PK_P is shared across the network, while the private key parts are distributed along a predefined route using a train (image) updated by S_0.

Client Computation

During the execution of Algorithm 5, each station computes their TP and FP for a set of predefined decision points D. The number of prediction confidences is denoted as m.

For each decision point $D[i]$, the algorithm accumulates the counts of *ones* (representing true labels equal to 1) and *zeros* (representing true labels equal to 0). These counts are updated while traversing the prediction values until the confidence values fall below the current decision

point $D[i]$. If the last prediction value is reached and has not yet been visited, the algorithm ensures it is processed to finalize the counts.

The computed true positives $TP[i]$ and false positives $FP[i]$ for each decision point are then encrypted using the Paillier public key PK_P. The encrypted values $\langle TP[i] \rangle$ and $\langle FP[i] \rangle$ are passed to the next station or the proxy for further processing.

Algorithm 5 *dppa-auc* Station Protocol

procedure DPPA-AUC_STATION(ID_{S_i}, SK_{S_i})
  Step 1: Compute TP and FP for Decision Points
  m is the number of predictions confidences
  D is the set of decision points
  $ones, zeros, p, last\_one\_visited \leftarrow 0, 0, 0, \text{False}$
  for d **in** D **do**
    while $p < m$ **and** $pre[p] > d$ **do**
      $ones \leftarrow ones + label[p]$
      $zeros \leftarrow zeros + (1 - label[p])$
      $p \leftarrow p + 1$
    if p **equals** m **and** $\neg last\_one\_visited$ **then**
      $last\_one\_visited \leftarrow \text{True}$
      $ones \leftarrow ones + label[m]$
      $zeros \leftarrow zeros + (1 - label[m])$
    $TP[i], FP[i] \leftarrow ones, zeros$
  Step 2: Encrypt Data
  for $i = 0$ to $len(D)$ **do**
    $\langle TP[i] \rangle, \langle FP[i] \rangle \leftarrow \text{ENC}(TP[i], PK_P), \text{ENC}(FP[i], PK_P)$

Proxy Computation

During the execution of Algorithm 6, the proxy station aggregates the TP and FP values across all stations for a set of decision points D. For each decision point $D[j]$, the proxy computes the global sums $\langle TP_{global}[j] \rangle$ and $\langle FP_{global}[j] \rangle$ by adding the encrypted contributions $\langle TP_{S_i}[j] \rangle$ and $\langle FP_{S_i}[j] \rangle$ from all stations using the Paillier encryption scheme.

Next, the proxy calculates the final TP and FP values. For each decision point j (from the second point onward), the proxy adds the cumulative values from the previous decision point to the current point for $\langle TP_{final}[j] \rangle$. For $\langle FP_{final}[j] \rangle$, the proxy computes the difference between the current and previous FP values. These calculations are performed securely using Paillier operations to preserve data privacy.

From this point on, Algorithm 6 follows the same steps as Algorithm 4 for computing the numerator and denominator components, as well as partially decrypting these components. Specifically, the proxy computes randomized encoding components for the numerator and denominator using securely scaled true positive and false positive values. The proxy then

partially decrypts these randomized components using its partial private key SK_{S_p} and shares them back with the stations.

Algorithm 6 *dppa-auc* Aggregation Protocol

procedure DPPA-AUC_PROXY(ID_{S_i}, SK_{S_p})
  D is the set of decision points
  Step 1: Aggregate TP and FP Across Stations
  for $j = 0$ to $len(D)$ **do**
    for each station i **do**
      $\langle TP_{global}[j]\rangle \leftarrow \mathsf{ADD}(\langle TP_{global}[j]\rangle, \langle TP_{S_i}[j]\rangle, PK_p)$
      $\langle FP_{global}[j]\rangle \leftarrow \mathsf{ADD}(\langle FP_{global}[j]\rangle, \langle FP_{S_i}[j]\rangle, PK_p)$
  Step 2: Compute Denominator Components
  $a, b, r_A^1, r_A^2 \leftarrow \mathsf{RANDS}()$
  $\langle TP_A\rangle \leftarrow \mathsf{MUL}(\langle TP_{global}[last]\rangle, a)$
  $\langle FP_A\rangle \leftarrow \mathsf{MUL}(\langle FP_{global}[last]\rangle, b)$
  $\langle D_1\rangle \leftarrow \mathsf{ADD}(\langle TP_A\rangle, r_A^1)$
  $\langle D_2\rangle \leftarrow \mathsf{ADD}(\langle FP_A\rangle, r_A^2)$
  $\langle D_3\rangle \leftarrow \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle TP_A\rangle, r_A^2) + \mathsf{MUL}(\langle FP_A\rangle, r_A^1)), r_A^1 \cdot r_A^2)$
  Step 3: Compute Numerator Components
  for $j = 1$ to $len(D)$ **do**
    $\langle TP_{final}[j]\rangle \leftarrow \mathsf{ADD}(\langle TP_{global}[j]\rangle, \langle TP_{global}[j-1]\rangle, PK_p))$
    $\langle FP_{final}[j]\rangle \leftarrow \mathsf{ADD}(\langle FP_{global}[j]\rangle, \mathsf{p\_mul\_const}(\langle FP_{global}[j-1]\rangle, -1), PK_p)$
  $\langle N_3\rangle \leftarrow 0$
  for $i = 0$ to $len(D)$ **do**
    $r_i^1, r_i^2 \leftarrow \mathsf{RANDS}()$
    $\langle sTP[i]\rangle \leftarrow \mathsf{MUL}(\mathsf{ADD}(\langle TP_{final}[i]\rangle, \langle TP_{final}[i-1]\rangle), a)$
    $\langle dFP[i]\rangle \leftarrow \mathsf{MUL}(\mathsf{SUB}(\langle FP_{final}[i]\rangle, \langle FP_{final}[i-1]\rangle), b)$
    $\langle N[i]_1\rangle \leftarrow \mathsf{ADD}(\langle sTP[i]\rangle, r_i^1)$
    $\langle N[i]_2\rangle \leftarrow \mathsf{ADD}(\langle dFP[i]\rangle, r_i^2)$
    $\langle N[i]_3\rangle \leftarrow \mathsf{ADD}(\mathsf{ADD}(\mathsf{MUL}(\langle sTP[i]\rangle, r_i^2) + \mathsf{MUL}(\langle dFP[i]\rangle, r_i^1)), r_i^1 \cdot r_i^2 + z_i)$
    $\langle N_3\rangle \leftarrow \mathsf{ADD}(\langle N_3\rangle, \langle N[i]_3\rangle)$
  Step 4: Partially Decrypt Components
  $(\langle N[i]_1\rangle, \langle N[i]_2\rangle) \leftarrow \mathsf{DEC}((\langle N[i]_1\rangle, \langle N[i]_2\rangle), SK_{P_1})\ i \in \{1, 2, \ldots, len(D)\}$
  $(\langle D_1\rangle, \langle D_2\rangle, \langle D_3\rangle, \langle N_3\rangle) \leftarrow \mathsf{DEC}((\langle D_1\rangle, \langle D_2\rangle, \langle D_3\rangle, \langle N_3\rangle), SK_{P_1})$

Final Computation

DPPA-AUC computation utilizes the same Equation 3.9 as the DPPE-AUC to compute the final AUC value locally at each station.

PP-AUC Security Analysis

Assume there are N stations in the PHT framework. An adversary controlling $(N-M)$ stations cannot infer any information about the honest M stations' predictions, labels, or flags. Labels

and flags are encrypted using the modified Paillier cryptosystem, with encryption performed using the proxy station's public key. Since stations only hold a share of the private key of the proxy, an adversary controlling $N - M$ stations cannot access the other part of the private key. Predictions are encrypted with AES-GCM, and the symmetric key is encrypted using the proxy station's RSA public key. Consequently, without the proxy's private RSA key, the adversary cannot access the predictions of honest stations.

An adversary controlling only the proxy station cannot access private data, as labels and flags are encrypted with the proxy's public key. The proxy station cannot decrypt this information since it only knows a portion of its private key. Although predictions are masked with two random values, the adversary can observe the masked values but cannot infer meaningful differences between them due to the presence of dummy values. These dummy values significantly reduce the likelihood of identifying the real samples, preventing the adversary from confidently estimating the relative differences between actual prediction values.

Additionally, the adversary cannot determine the number of predictions at each station due to inserting dummy values. However, if an adversary controls both the proxy station and at least one participating station, they could decrypt all predictions, labels, and flags by combining the private keys. For this reason, our security model assumes no collusion between the proxy station and other stations.

The security of our scheme depends on the security properties of the modified Paillier system and randomized encoding. The security of the Paillier cryptosystem was analyzed by[73], which showed two important guarantees: one-wayness, meaning that the ciphertext and public key cannot be used to deduce the plaintext, and semantic security, ensuring that an adversary cannot match plaintext-ciphertext pairs even when given two plaintexts. The definitions and security proofs for the randomized encoding of addition and multiplication used in this scheme are provided by[80].

This security analysis applies to both DPPE-AUC and DPPA-AUC, since the only methodological difference (exact vs. approximate thresholds) does not affect the underlying cryptographic primitives or trust assumptions.

3.2.6 Use Cases

This chapter presents the diverse use cases with which the PHT-meDIC development received funding. The use cases presented in this chapter exemplify the PHT's versatility and usefulness across various domains within the healthcare sector. These examples demonstrate the practical applications of PHT and illuminate its potential to revolutionize data-driven decision-making in healthcare. The use cases showcase how PHT is instrumental in addressing complex challenges in the healthcare industry, from improving patient outcomes to advancing medical research.

DIFUTURE

DIFUTURE is one of four consortia funded since 2018 by the German Ministry of Education and Research as part of the MII. Its goal is to provide detailed and extensive data to physicians and medical researchers to speed up innovation, improve healthcare processes, and support decision-making, benefiting patients. A primary focus is on data integration and sharing, emphasizing high data security, privacy, and patient trust with innovative approaches such as distributed learning. In Use Cases of Parkinson's and Multiple Sclerosis diseases, the established infrastructure was evaluated. Formed by several German universities and medical centers, including the Technical University of Munich, Ludwigs-Maximilians-University Munich, Eberhard Karls University of Tübingen, and others, DIFUTURE also expanded its partnership in 2019 to include Ulm University. It collaborates with networking partners and balances industrial and open-source solutions like i2b2, tranSMART, and openBIS for interoperability. Internationally, DIFUTURE collaborates with the Dutch Techcenter for Life Sciences and has a renowned Scientific Advisory Board with members from global institutions like Harvard, Vanderbilt University, Oxford University, and ETH Zurich. DIFUTURE commits to international collaboration and the GO FAIR Initiative[35].

In January 2023, MIRACUM - another German consortium - and DIFUTURE officially announced their collaboration as one consortium. The collaboration not only aligns with their goals but also helps to streamline future development and reduce redundancy[139].

CORD

Under the German MII, the CORD-MI "Collaboration on Rare Diseases" initiative[140] is a significant effort focused on improving research and care for rare diseases. It collaborates with 24 university hospitals from all four consortia and several partner institutes. Key goals include increasing the visibility of rare diseases, deepening the understanding of care realities, enhancing the quality of patient care, and encouraging research in this specialized field. By 2022, the initiative aimed to develop and implement innovative concepts and solutions for better medical documentation, effective data sharing, and robust data protection research. These efforts facilitate comprehensive nationwide data collection and analysis, contributing significantly to rare disease research and care. Furthermore, the established infrastructure should be used beyond the project time. Within this context, distributed analysis based on DataSHIELD, PADME, and PHT-meDIC platforms was performed.

POLAR

Similar to CORD, the POLAR-MI "POLypharmazie, Arzneimittelwechselwirkungen und Risiken" project[141], also part of the German MII, focuses on addressing health risks associated with polypharmacy, particularly in older patients with multimorbidity. By leveraging data from

various medical institutions, the project aimed to develop methods for detecting potentially inappropriate medications and high-risk prescriptions. The overall goal is to improve medication safety. The initiative involves collaboration among experts from 21 institutions, including 13 university hospitals, to utilize prospective and retrospective patient data on prescribed medications and pharmacy dispensations. Special projects within POLAR-MI also explore data linkage with patient outcomes and the development of a corpus for processing natural language related to adverse drug reactions. Furthermore, distributed analysis platforms were evaluated.

Leuko-Expert

A study within the Leuko-Expert project funded by the German Ministry of Health[142]. The objective of the project is to develop an expert system to aid in the diagnosis of the rare disease (RD) Leukodystrophy, a genetic disorder that affects the brain and causes movement and sensory perception disturbances[143]. The project involves clinical and genetic data that has been collected and generated by the clinics in Aachen, Leipzig, and Tübingen (see Figure 3.7) in Germany. In detail, the patient data is collected from two reference centers specialized in both childhood and adult variants of Leukodystrophies, located at the University Hospitals of Tübingen and Leipzig. Furthermore, data is also collected from patients who received differential diagnoses at the University Hospital Aachen. Regarding data provision, the project relies on the data integration centers (DICs) at the corresponding university hospitals that are part of the MII and facilitate external access to the data[144]. Our designed analysis 'Data Discovery' encompasses the execution of a train exchanged between the infrastructures PADME and PHT-meDIC. In the Leuko-Expert scenario, the interoperability between these infrastructures to analyze more data becomes desirable, since two DICs have deployed PADME and one has deployed PHT-meDIC (see Figure 3.7).

3.2.7 PHT Interoperability

In this section, we briefly review the key components of each infrastructure and further indicate if and how the derived commonalities or differences support or complicate our plan for technical interoperability. For an overview, Figure 3.8 provides a top-level perspective of the components. For a detailed description of the infrastructures, we refer to the corresponding publications[31,145].

PoC of different layers

The five different layers are described in Chapter 2.1.4 and are method wise described in the following pages.

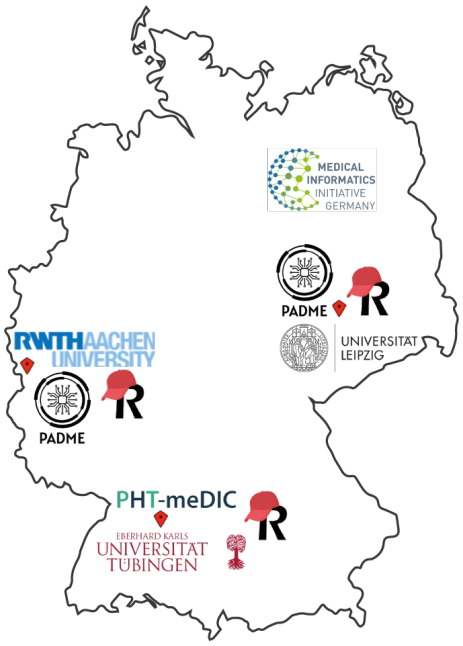


Figure 3.7: Deployment of stations within the Leuko-Expert project Leipzig and Aachen use a PADME station, whereas Tübingen utilizes a PHT-meDIC station. In Layer 0, a Research Electronic Data Capture (REDCap) system is implemented at each institution to facilitate data provision. Since two PHT implementations are present, their interoperability might be desirable to enlarge the global dataset.

Layer 0 - Data Integration

Conceptualization

In our scenario, the challenge lies in integrating three decentralized datasets from different institutions. Each institution houses its raw data in various clinical application systems. For example, as mentioned above, the data provided by Tübingen and Leipzig is highly specialized with respect to both childhood and adult variants of leukodystrophies, while Aachen only provides data on differential diagnoses. This data is unstructured and varies significantly in volume, format, and content, posing a significant challenge for standardization and integration. Therefore, a central data repository is needed in each institution that consolidates this diverse data in a more structured and accessible format. A key aspect of this integration process is the definition and implementation of a mutual data schema, such as the German MII CDS. This schema provides a standardized structure for the data, ensuring consistency and facilitating interoperability among the different institutions at the data level. Be aware that the design of such an ETL (Extract, Transform, Load) pipeline can vary widely, influenced by factors such as the nature of the data, the chosen data schema, and the specific technologies for data provision. In the subsequent section, we will briefly outline our specific strategy to harmonize the distributed data on leukodystrophy.

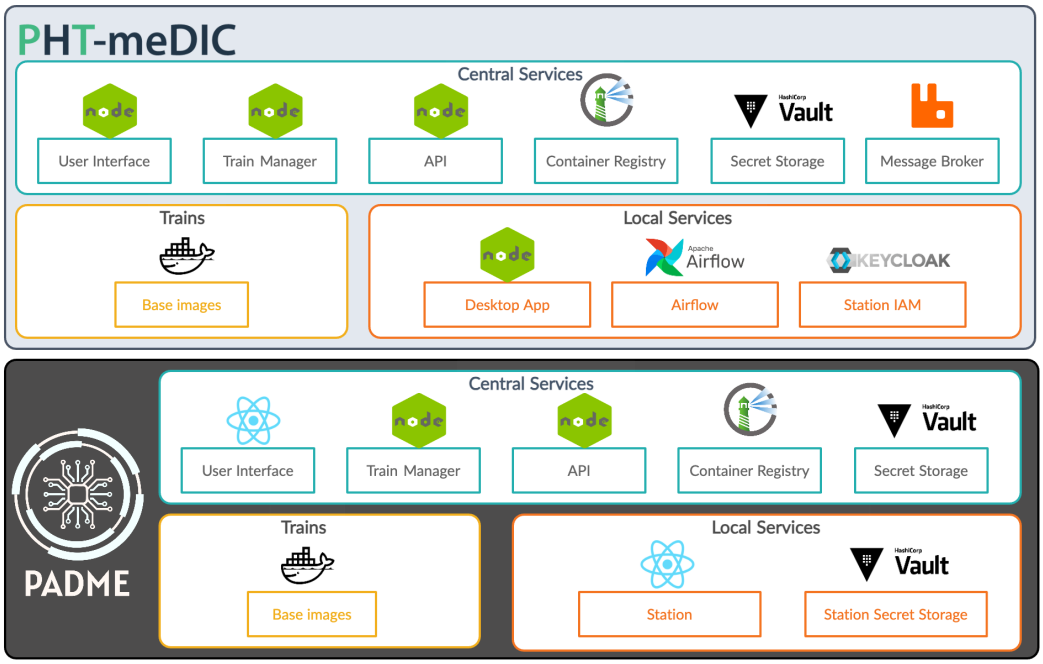


Figure 3.8: Overview of PHT infrastructures: PADME and PHT-meDIC. PHT-meDIC and PADME follow a container-based security-by-design approach, employing tools like Harbor and Vault for secure operations. Both consist of two primary components: the Central Service (CS) and the station software, also referred to as local service (LS). The CS manages train repositories, including the central train repository using Harbor. The station software operates as a control interface for the container engine. Encryption in PADME uses envelope encryption with symmetric and asymmetric keys. The security concept of PHT-meDIC also focuses on the detection of encryption and manipulation detection using digital signatures.

PoC Implementation

In order to harmonize the data and make it analysis-ready, we store the data in a Research Electronic Data Capture (REDCap - https://www.project-redcap.org/) database system. REDCap is essentially designed for data collection and management in research studies and clinical trials and, therefore, suitable for our purpose[146,147]. At this point, we acknowledge that other technologies like FHIR could also facilitate data provision. However, in our scenario, REDCap was pre-installed at all participating sites and the medical teams were already familiar with this tool. For these reasons, we opted to use REDCap for our PoC. Each participating institution received a pre-compiled REDCap questionnaire, establishing a data schema to collect, structure, and standardize patient data across all institutions. The process of filling out the questionnaires for each patient has been performed manually or by semi-automatic means by the clinical teams in each hospital. The results of the REDCap questionnaire, which we refer to as the dataset later, is divided into three distinct sections:

- **Baseline:** Master data that contains patient details such as sex, age, and diagnosis.
- **Examination:** Data that provides information regarding the patient's health status. This includes, among others, attributes such as abnormalities in higher brain functions or loss of libido that are defined in the Human Phenotype Ontology (HPO)[148]. The answers to the questions within this section encompass a wide spectrum of response types, spanning from yes/no/unknown options to open text fields.
- **Genetics:** Data that refers to genetics and serves as a documentation of the results obtained from genetic testing. This section includes details such as the observation year, the specific affected gene, and other attributes related to genes, such as the classification of the American College of Medical Genetics (ACMG) or the underlying mutation in the nomenclature of the Human Genome Variation Society (HGVS)[149][150].

While the baseline section is mandatory for each patient, the others are optional and can consist of multiple instances (i.e., multiple examinations). In the final step, after all patient data instances have been entered into the REDCap system at each DIC, the REDCap database has been made accessible to the station software located at the respective DIC (see Figure 3.7).

After the establishment of the data provision for the stations Aachen, Leipzig, and Tübingen, we need to move a layer up to ensure that the stations are accessible and can pull the trains. This includes assigning unique identifiers to the stations to provide a clear and unambiguous destination for the analysis train.

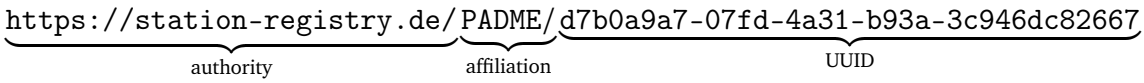
Layer 1 - Assigning (Globally Unique) Identifiers to Stations

Conceptualization

In this layer, we address the disambiguation of the stations within a PHT ecosystem, especially during the route selection process. Up until now, the infrastructures have been using their own custom station identifiers, creating a potential risk of ambiguities due to the absence of a shared agreement on these identifiers between different infrastructures. Furthermore, the lack of global identifiers prevents one from choosing stations that belong to a different ecosystem for the train route. Therefore, we employ an additional authority on top of the infrastructures that acts as an indexing or directory service. In general, this approach is inspired by the well-established Domain Name System (DNS), which manages the namespace of the Internet[151][152]. Establishing such a namespace for the stations is our main objective, such that we can select stations for the train route and support the routing through our business logic (see Layer 4). In essence, we require that each station's identifier must be a Uniform Resource Locator (URL). This URL represents the destination where the train is to be directed, enabling its execution at that particular station.

PoC Implementation

Taking inspiration from DNS servers, we use the Station Registry (SR) as a component that serves as a central database for stations[153]. Note that the SR has already been partially introduced in the work by Welten et al.[154]. Station administrators must register their stations at the SR before or after the installation and specify the affiliation of the station, e.g., in our case, either as a PADME or a PHT-meDIC station. Similarly to DNS, we have chosen a basic hierarchical structure for our station identifiers:



Using these URLs, the business logic (as described in Layer 4) can effectively resolve the URLs and orchestrate the routing of trains between stations. This process is analogous to how requests are navigated on the Internet, starting from a higher level (the PHT ecosystem) and moving down to a more granular level (the individual stations). This hierarchical structure allows us to precisely locate each station, determine its association with a particular ecosystem, and dispatch the trains accordingly.

After establishing a method to identify each station through its URL, our next step involves synchronizing the internal workflows of each station type, whether it is a PADME or PHT-meDIC station. In our study, this specifically entails the alignment of the security protocols used by each station, which is part of the next section.

Layer 2 - Harmonizing the Security Protocols

Conceptualization

The next aspect that we need to address is the conceptualization of an overarching security concept, coupled with the alignment of the PADME and PHT-meDIC security protocols. One crucial challenge is that each infrastructure adheres to its own protocols and workflows with respect to encryption or the signing of digital assets. To overcome this, we propose adopting modular software containers, a methodology suggested by Hasselbring et al.[155], to enhance software and workflow portability. Consequently, our strategy involves the containerization of all security-related processes of each ecosystem's workflows to enable interoperability[155]. With this strategy, we achieve the flexibility required to implement these workflows across various ecosystems, while also ensuring the seamless distribution of protocol updates without substantial modifications in other ecosystems. The extracted containerized workflow can serve as a supplemental service, accessible for other PHT ecosystems, allowing different infrastructures to integrate this service into their own workflows. Alternatively, a comparable and probably

less resource-intensive approach to distributing security protocols in a container could involve structuring them in a library-like structure that is installed in each involved infrastructure. All in all, the overarching protocol requires the containerized process steps to be arranged in sequence (see Figure 3.9 as an example). The train undergoes decryption (pre-run) or encryption (post-run) incrementally, with each containerized module being invoked in succession to perform the necessary operations. The practice of nesting various workflows has the additional advantage that it can be applied multiple times, especially when more than two infrastructures are involved in the process. Having this concept in mind, we argue that the concept of this overarching security protocol, potentially comprising several sub-protocols, adheres to the established security policies of each PHT infrastructure involved, due to the sequential execution of each sub-protocol. In the following, we provide one exemplary integration of a security protocol in the context of our PoC.

PoC Implementation

In our PoC, the security and encryption protocols of the PHT-meDIC ecosystem have been modularized through containerization. Please find the container code in the supplementary materials. This module is then seamlessly integrated into the execution environment of the PADME station(s). It is essential to note that this modular approach is reciprocal; the PADME protocol can also be similarly containerized and integrated within the PHT-meDIC station. Conceptually and metaphorically speaking, we consider this module image as a supplementary train that can be pulled and placed next to the *real* train that arrives at a station. During train execution, both security protocols are executed in sequence while preserving the integrity of the original system's processes. The operation of this series of security protocols, as exemplified by our two protocols, is shown in Figure 3.9 and works as follows:

- **Train Arrival (Beginning of Process):** Upon arrival at a PADME station, the train was secured with dual encryption: The external layer conforms to the PADME protocol, whereas the internal layer is encrypted following the PHT-meDIC protocol.
- **Pre-run Protocol of PADME (Decryption - Outer Layer):** The initial step involves the PADME pre-run protocol decrypting the external encryption layer.
- **Pre-run Protocol of PHT-meDIC (Decryption - Inner Layer):** After the decryption of the outer layer, the PHT-meDIC pre-run protocol decrypts the inner encryption layer, preparing the train for its execution.
- **Train Execution:** Upon successful validation, the train is executed by the station environment.

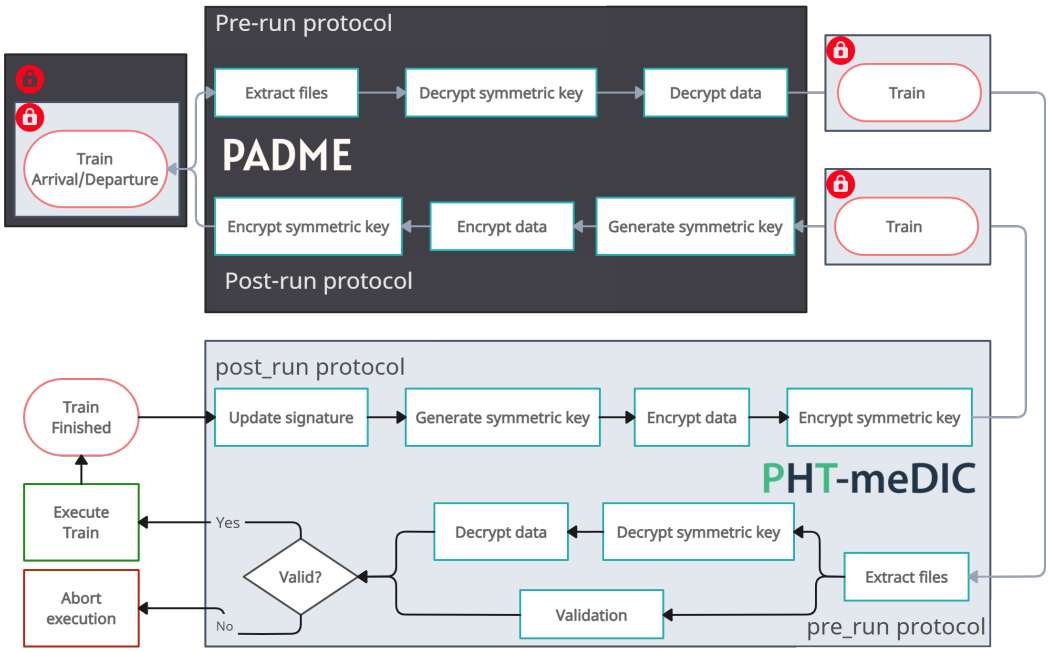


Figure 3.9: Interaction of the security concepts of the PADME and PHT-meDIC platforms The overarching security concept for each infrastructure represents a combination of both. In our scenario, the PADME security protocol is applied on top of the PHT-meDIC protocol, when the train is at a PADME station. At a PHT-meDIC station, the order of the protocols would be flipped: first the PHT-meDIC, then the PADME protocol.

- **Post-run Protocol of PHT-meDIC (Re-encryption - Inner Layer):** Post-execution, the train is re-encrypted, starting with the PHT-meDIC post-run protocol addressing the inner layer.
- **Post-run Protocol of PADME (Re-encryption - Outer Layer):** Following the inner layer's re-encryption, the PADME post-run protocol re-encrypts the external layer.
- **Train Departure (End of Process):** This marks the completion of the entire train lifecycle at a station. After this step, the train is dispatched to the next station on the route.

In Layers 1 and 2 described above, we incorporated essential metadata elements for each dispatched train. This includes details such as the station's URL for precise train routing and security-related items such as the public keys of each station for encryption, which are especially crucial in Layer 2. To integrate these pieces of information into a single entity, we propose the creation of a unified data exchange schema. This schema will be instrumental in the decryption and encryption processes of the stations, as well as in directing the train on a given route. The formulation of this schema will be discussed below.

Layer 3 - Common MetaData Exchange Schema

Conceptualization

The implementation of a modular security protocol (Layer 2) also requires the presence of metadata (e.g., public keys, hashes, or signatures), which plays a crucial role in ensuring proper functionality. Consequently, based on the initial two layers outlined, we identify an additional requirement: the establishment of a standardized metadata schema. As already pointed out by Lamprecht et al., software metadata is a necessity for (semantic) interoperability[58]. This corresponds to the definition of Benson et al., which characterize semantic interoperability as the capability *'for computers to share, understand, interpret and use data without ambiguity'*[60]. Hence, our aim is to develop a standardized set of metadata elements that can be used and exchanged in diverse workflows, ensuring that *'both the sender and the recipient have data that means exactly the same thing'*[60]. We aim for a solution that can be effortlessly expanded to incorporate security-related and infrastructure-specific requirements. We divide the metadata set into two categories:

- **Business Logic Metadata:** This information is used by the business logic workflow, enabling tasks such as the routing of trains between stations. Within Layer 1, we have initiated the disambiguation of the stations by globally identifying them and incorporating them into the business logic metadata.
- **Security Protocol Metadata:** This category contains essential elements of modern security protocols and (asymmetric) encryption systems, such as public keys, hashes, and signatures that enable secure communication and data transmission by ensuring confidentiality, integrity, and authenticity[156]. The information provided can be utilized by the infrastructures to either apply external security protocols (as described in Layer 2) or their own protocols.

At this point, the question of the location of metadata storage for subsequent processing arises. Given the various possible approaches, such as centralized or decentralized storage, we require that the set of metadata is attached to the train and that each infrastructure handles it for internal business logic. This aligns with the train definitions proposed by Bonino et al., where trains are decomposed into metadata and the payload that contains the analysis[157]. Attaching metadata to the train offers the benefit of eliminating the need to query this information from external services. This approach is also autonomous and does not rely on any central authority, which ensures that the information is readily available wherever it is required.

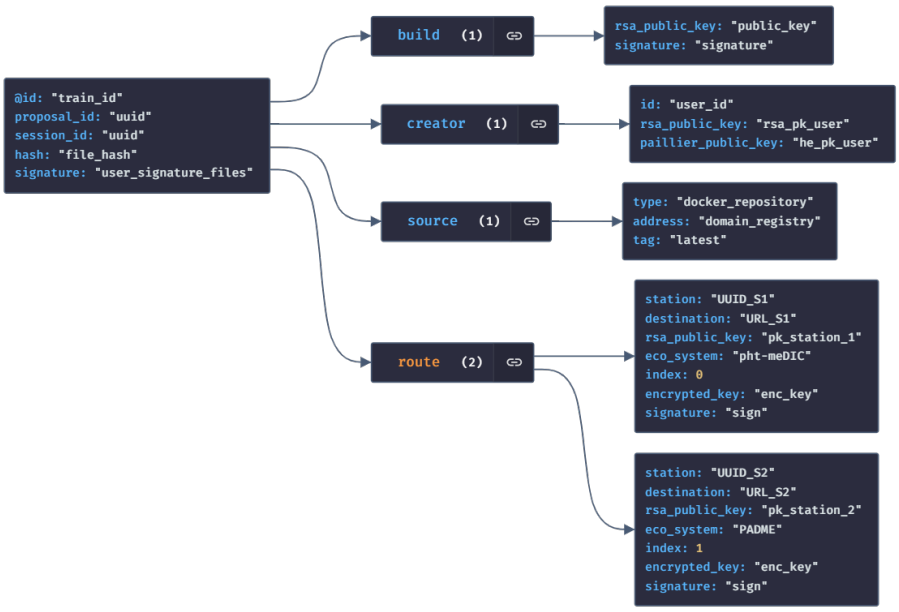


Figure 3.10: Metadata schema to enable interoperability. The schema contains information of two categories: Business Logic and Security Protocol metadata. Business Logic metadata is required for the train orchestration, while the security-related metadata is used by the security protocols.

PoC Implementation

According to our conceptualization, we use a small selection of metadata items as depicted in Figure 3.10. This schema covers business-related information concerning the train's origin (e.g., the source repository), its creator, and the identifiers of the stations to be visited. Note that these (basic) items share similarities with the definitions outlined by Bonino et al.[157]. Specifically, the `route` array captures essential station-related information and the station identifiers that have been introduced above in Layer 1. As each route item has a specific index the route can be systematically executed step by step. Additionally, the metadata includes the public keys of all participating entities, guaranteeing that assets can be encrypted in a manner that only the designated recipient can decrypt them. This is achieved in collaboration with the security protocols outlined in Layer 2, which process these keys. In terms of ensuring the integrity of the train assets, we also utilize signatures and hashes. It is important to mention that, in our particular case, each station's public key is obtained from the SR. The installation and registration process of a station in the SR necessitates the provision of its public key, which is then retrieved from the SR during the route selection. Before the train is dispatched, the metadata is attached to the train.

Once we have established an identifier mechanism (see Layer 1), aligned the security protocols involved (see Layer 2), and laid the groundwork for a common metadata exchange schema (see Layer 3), the next step is to facilitate actual technical interoperability. This implies that we leverage the foundational work established in the previous layers to facilitate the exchange of actual trains between the two ecosystems, PADME and PHT-meDIC. In the following section, we will introduce a routine that outlines the business logic required for this process.

Layer 4 - Overarching Business Logic

Conceptualization

The objective of this layer is to perform the transfer of the analysis from one ecosystem to another. As the metadata attached to the train entails the concrete route information, the business logic is able to transfer the digital assets representing the train to a given destination. The destination endpoint can be a container repository, identified by a URL, to which the train can be sent for further processing. There might be several approaches and techniques to transfer and transform a train from one ecosystem to another. In the following, we briefly present our overarching business logic used to enable technical interoperability between PHT-meDIC and PADME.

PoC Implementation

We refer to Figure 3.4 and Figure 3.11 as a graphical representation of our overarching business logic to transfer (container) trains between ecosystems. From a top-level perspective (see Figure 3.4), our business logic is modeled after the real-world practice of reloading cargo from one train to another. The encrypted contents (the cargo) of the train, such as code and files, are unloaded from the arriving, external train and loaded into a new train that is compatible with the ecosystem.

From a technical perspective and to streamline the transfer of trains between ecosystems, we established a dedicated input repository (the 'transfer station') within each central service component. This repository is the designated location to which external trains should be pushed. A webhook monitors this repository, identifies incoming trains, and initiates the infrastructure-specific business logic to direct the train to its intended station after the train contents have been reloaded. This portal-inspired design decision has the advantage of keeping the actual stations private within the infrastructure, with only one repository accessible by external services or infrastructures. To send a train on its track, the entity requesting the train can select a route that utilizes information from the SR, potentially including stations within the same infrastructure, as well as those that connect different infrastructures. If the next station is assigned to the current infrastructure, the standard business logic proceeds to dispatch the train. In the other case, when the next station is assigned to another infrastructure,

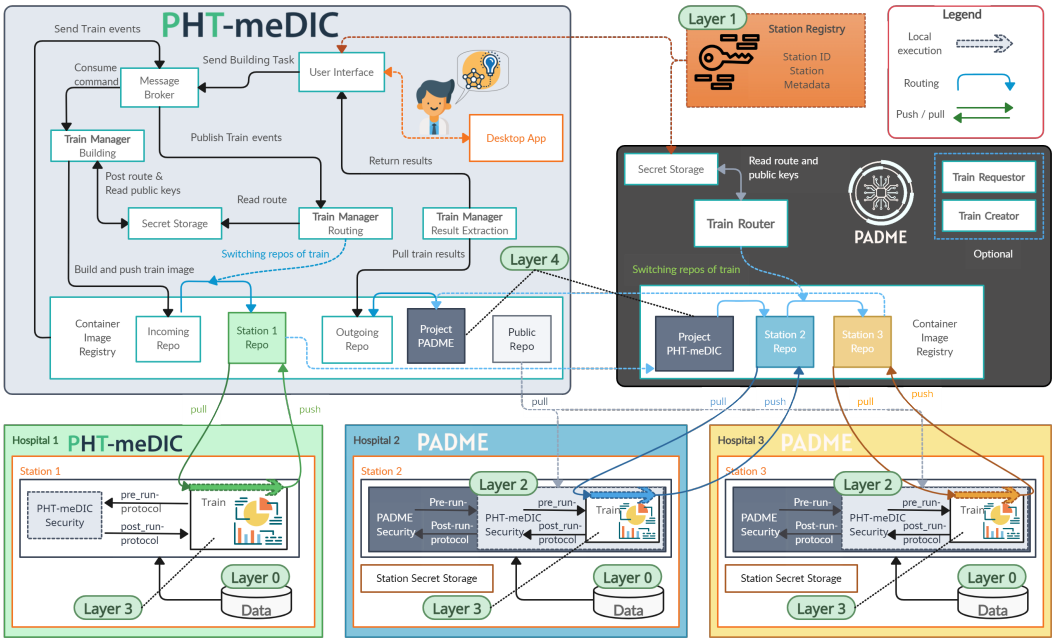


Figure 3.11: Proof-of-Concept implementation of our interoperability concept involving the platforms PADME and PHT-meDIC. This figure shows the routing of the train from PHT-meDIC to PADME. The reverse route, from PADME to PHT-meDIC, follows a similar pattern.

the business logic needs to proceed differently: As shown in Figure 3.4 and Figure 3.11, the business logic of the sending infrastructure detects the external station based on the metadata and transfers the train to the dedicated repository of the receiving infrastructure using a 'push' command (when using container trains). Subsequently, the receiving infrastructure's business logic handles the train and prepares/reloads it for the next station along the route. The station receives the train and decrypts the envelope encryption using the modularized security protocol that has been introduced in Layer 2. After the execution of the train, it is re-encrypted using the envelope encryption and pushed back to the central service. Depending on the route, the train either remains in the infrastructure and is made available for the next station or is forwarded to the other infrastructure, following the same principle as described above. This cycle continues until all stations along the route have been processed. Up to now, we have enabled the interoperability of the two involved ecosystems.

Chapter 4

Results

The content of Section 4.2 and 4.3 constitutes an extended version of the manuscript:

Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train (PHT-meDIC)[31]

The content in Section 4.4 constitutes an extended version of the manuscript:

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC[33]

The content in Section 4.5 constitutes an extended version of the manuscript:

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis[32]

4.1 Introduction

In the first two sections part of this chapter, we present the analysis results through a series of showcases. The first demonstration illustrates a bioinformatic task implemented with homomorphic encryption. The second case compares image analysis in distributed and central settings. Furthermore, they demonstrate how two unrelated tasks relating to different high-volume medical input data can be solved using the PHT-meDIC platform. The image analysis compares centralized with decentralized results. The outcomes are further processed within the bioinformatics pipeline with additional privacy-preserving methods to prevent stations from examining intermediate results.

We demonstrate the performance and scalability of the distributed privacy-preserving AUC computation method. Lastly, we delve into experiments related to the Leuko-Expert project,

which involved performing descriptive statistics on two interoperable PHT platforms, incorporating task combinations with additional privacy-enhancing mechanisms.

Last, we highlight the interoperability results with the data discovery and descriptive statistics from Leuko-Expert on patient data.

The following two experiments demonstrate the PoC of the functionality of our platform.

4.2 Bioinformatics pipeline

4.2.1 Problem and input

The Human Leukocyte Antigen (HLA) system is a gene complex that encodes the major histocompatibility complex (MHC) proteins in humans. MHC genes are highly polymorphic and have many variations across humans[158]. It is pleiotropic, and its primary function is to support T cells in recognizing foreign proteins (antigens), an essential process in adaptive immunity. Therefore, it is a significant driver for initiating and coordinating immune responses[159]. Determining HLA types of subjects is important in several areas: therapeutics can be affected by HLA type, potential transplants require a match of donor and recipient HLAs, and certain infectious diseases can progress more rapidly[159].

Traditionally, HLA typing is done by medical centers. Results are usually returned within medical reports and are, by default, not accessible for further analysis. Approaches like the German Human Genotyping and Phenotyping Archive (GHGA[160]) aim to solve this problem. If clinical sites have genome sequencing data of patients available, it is usually not trivial to analyze the raw data ad hoc. By developing standardized and reproducible bioinformatics analysis pipelines, the nf-core[161] initiative aims to overcome this bottleneck. Usually, these pipelines are run by medical centers and not the hospital itself. Using the PHT-meDIC, it is now possible to run these complex analyses and combine the results across sites.

4.2.2 Analysis

We used 216 next-generation sequencing (NGS) samples (see Table E) consisting of Illumina HiSeq 2000 and Genome Analyzer II exome sequencing runs from the 1000 Genomes Project[162] to demonstrate the use of such pipelines. We executed an nf-core HLA typing pipeline[163] over three stations with the PHT-meDIC. Subject data is mapped to FHIR profiles, which contain references (URIs) to locally stored NGS data, which the train can process. The base image of the train contains *nf-core/hlatyping*[163] software dependencies and the Python algorithm requirements. The algorithm and query are stacked on top of the base image following PHT principles. The algorithm has two tasks:

1. Compute HLA types of subjects and securely determine the number of patients with class I HLA type B*35:01 (fast progression of HIV/Aids[159]).

2. Plot the most frequent 15 HLA types of all provided samples.

4.2.3 Results

Task 1: Locally decrypting the results reveals the homomorphically encrypted result. Decrypting this number with the Paillier private key reveals 24 occurrences of Class I allele **B*35:01** over all stations.

Task 2: The resulting figure of the second task (plotting the most frequent HLA types) is displayed in Figure 4.1. The plot contains the 15 most common HLA types across the stations included in the analysis.

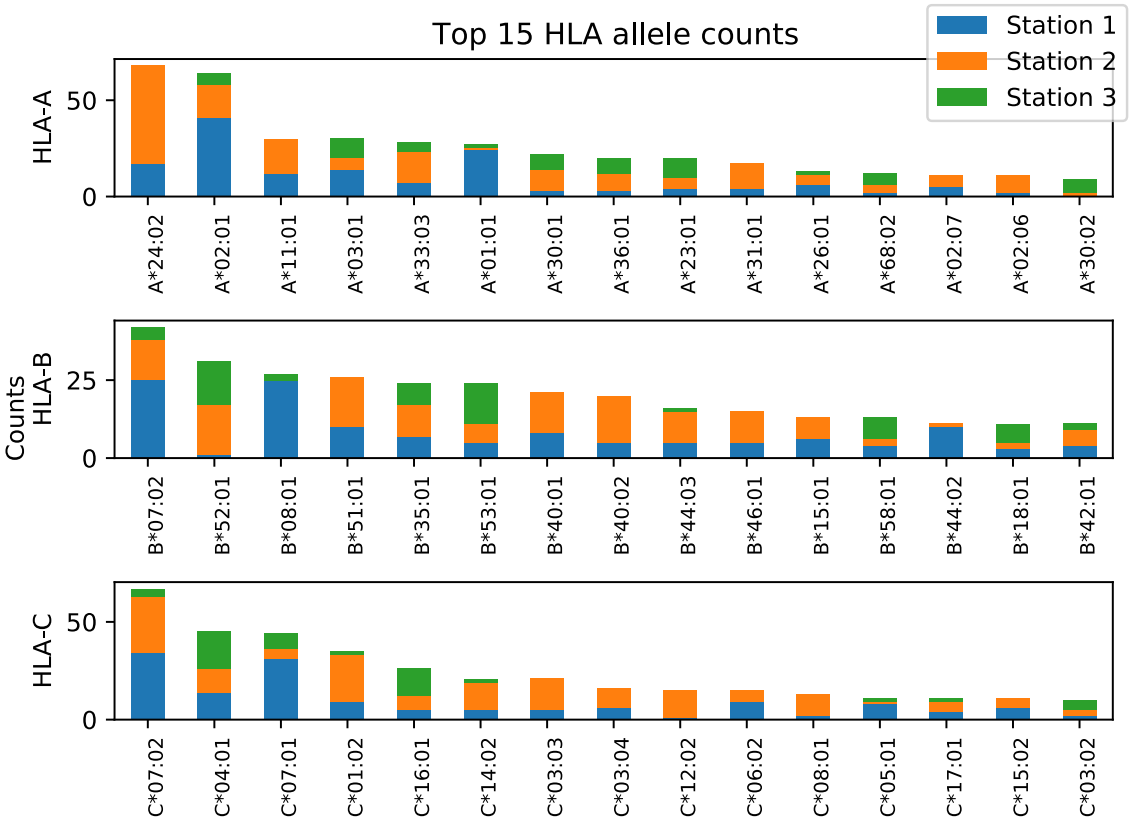


Figure 4.1: Results of the second task of the bioinformatics pipeline train This plot shows three different HLA classes and their allele counts ordered by number of occurrences. Colors indicate data originating from different stations.

The total execution time of the HLA typing demonstration train was 44h and 49 min on a Ubuntu 20.04.1 LTS cloud instance (table 4.1) with 28 Intel(R) Xeon(R) Gold 6140 CPU @ 2.30 GHz cores and 256 GB memory. The final results are 22 KB, and below 1s were taken at each station to decrypt and encrypt using the PHT-meDIC security protocol.

Station	# of samples	Execution time
1	84	13h 46min
2	94	24h 26min
3	38	6h 37min
Total	216	44h 49min

Table 4.1: Station-wise and total number of samples and execution time of the nf-core HLA typing train

4.3 Image analysis

4.3.1 Problem and input

The International Skin Imaging Collaboration (ISIC) 2019 is a dermoscopic image analysis benchmark challenge containing more than 25 thousand images of eight different skin cancer types[164–166]. The goal of the challenge is to support research and development of algorithms for automated melanoma diagnosis by providing data with an annotated ground truth by medical experts. For demonstration purposes, we use the publicly available algorithm of the challenge champions[167]. In this showcase we demonstrate the feasibility of performing image anaylsis with the PHT-meDIC. Additionally, we compare the performance of the model trained in a distributed setting with the traditionally trained model.

4.3.2 Analysis

The task of the analysis is to classify eight different types of skin cancer. We use all publicly available training images and train an efficientnet-b6 Deep Neural Network (DNN). In the centrallized analysis setting, we use the provided 5-Fold Cross-Validation (CV) splits from the challenge's first place submission[167]. In the distributed setting, we perform a 5-Fold CV at each station locally using the identical fold splitting as within the central analysis setup. The distribution of input data over each station is shown in table 4.2. The algorithm takes class imbalance into account by assigning weights to each class according to its availability. Input data is mapped to FHIR and processed by the train as in the first experiment.

Site	MEL	NV	BCC	AK	BKL	DF	VASC	SCC	Total
Station 1	1507	4291	1107	289	874	79	84	209	**8440**
Station 2	1507	4291	1107	289	874	79	84	209	**8440**
Station 3	1508	4293	1109	289	876	81	85	210	**8451**
Central	**4522**	**12875**	**3323**	**867**	**2624**	**239**	**253**	**628**	**25331**

Table 4.2: Distribution of melanoma imaging data over three stations and total number of samples of each different class of skin lesions. The cancer class abbreviations can be found in the supplementary material (see Table A).

4.3.3 Imaging train results

The efficientnet-b6 model, trained with all data at one station has a final averaged accuracy of 0.682 over 5-folds. Accuracy is continuously calculated class-wise and averaged with all classes. The distributed model, trained with only 20 epochs at each station for comparison, has a final averaged accuracy of 0.681. Even with equal distribution of each class at each station, there is a drop in performance between the second and the third stations. This drop might have been caused by crucial forgetting (or *Catastrophic Forgetting*)[168–170], which can depend on data quality differences or data availability.

The execution time of the second example train on a cloud Ubuntu 20.04.1 LTS instance with 2 *Nvidia TeslaV100* GPUs, 16 Intel(R) Xeon(R) Gold 6140 vCPU @2.30 GHz cores and 170GB memory is 37 h 9m and 8s in total. The encryption and decryption of results over three stations took in total 5m and 31s. Station wise execution time and size can be seen in Table 4.3.

Station	pre_run	Execution time	post_run	Acc	Sens
1	0 / 0MB / 10s	11h 26m 29s	8 / 625 MB / 1min 2s	0.737	0.737
2	8 / 625 MB / 59s	12h 47m 24s	13 / 625 MB / 1m 5s	0.829	0.784
3	13 / 625 MB / 57s	12h 55m 22s	3 / 654.64MB / 1m 1s	0.681	0.69
Total	time 2m 6s	1d 13h 9m 8s	time 3m 25s	0.68	0.69
Central	0 / 0MB / 8.817s	3d 23h 46m 25s	3 / 654.64MB / 1m 1s	0.684	0.625

Table 4.3: Execution time and performance of ISIC showcase model at different stations. *pre_run* and *post_run* protocols are security protocol steps. Number of files / file size / execution time is reported at each station in the protocol columns. Weighted accuracy (Acc) and weighted sensitivity (Sens) is averaged over all classes and reported from the last epoch at each station.

Performance comparison of centralized and distributed trains

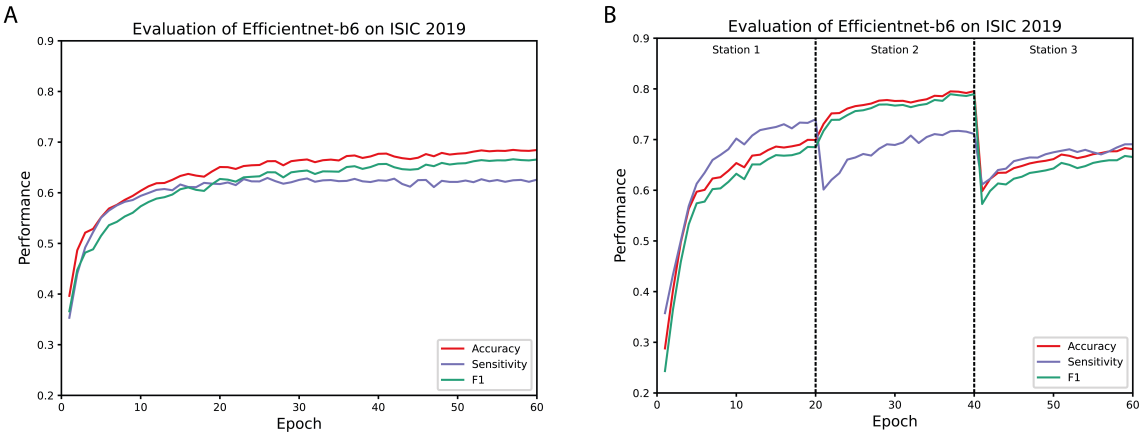


Figure 4.2: Performance comparions imaging showcase
(A) Averaged performance over 5 folds of a centrally trained model with all data available at one station trained for 60 epochs. Red lines indicate accuracy, green lines mean of the precision and recall, and blue lines sensitivity. Exponential moving average is implemented in tensorboard[87] for the smoothed lines.
(B) Averaged performance over 5 folds of a decentrally trained model with all classes **equally distributed** over three stations. Training was performed for **20 epochs** at each station.

4.4 PP-AUC Showcase and Experiments

4.4.1 Showcase

To evaluate the efficacy and privacy-preserving capabilities of our DPPE- and DPPA-AUC methods, we conducted experiments using both real-world HIV-1 V3-loop sequence data and synthetic datasets, benchmarking against ground truth (GT) AUC using the *roc_auc_score* function from the scikit-learn metrics library[171].

Showcase: HIV-1 Coreceptor Binding Prediction

For the primary showcase, we utilized 10462 HIV-1 V3-loop sequences annotated with coreceptor binding data[172]. The binary classification task is to predict binding preferences for coreceptors $CCR5$ and $CXCR4$, with data distributed across three stations for decentralized analysis. Each station trained a local model using its partitioned dataset, while a 10% holdout set was maintained for independent evaluation. This setup allowed us to test our method's effectiveness in preserving data privacy across distributed sites.

For performance reasons, we employed a flag patient generation approach, injecting synthetic samples only within value range of actual patient data to not increase the number of

thresholds. We performed all experiments and showcases on a local computer (Apple® Mac mini, M4Pro, 48GB memory).

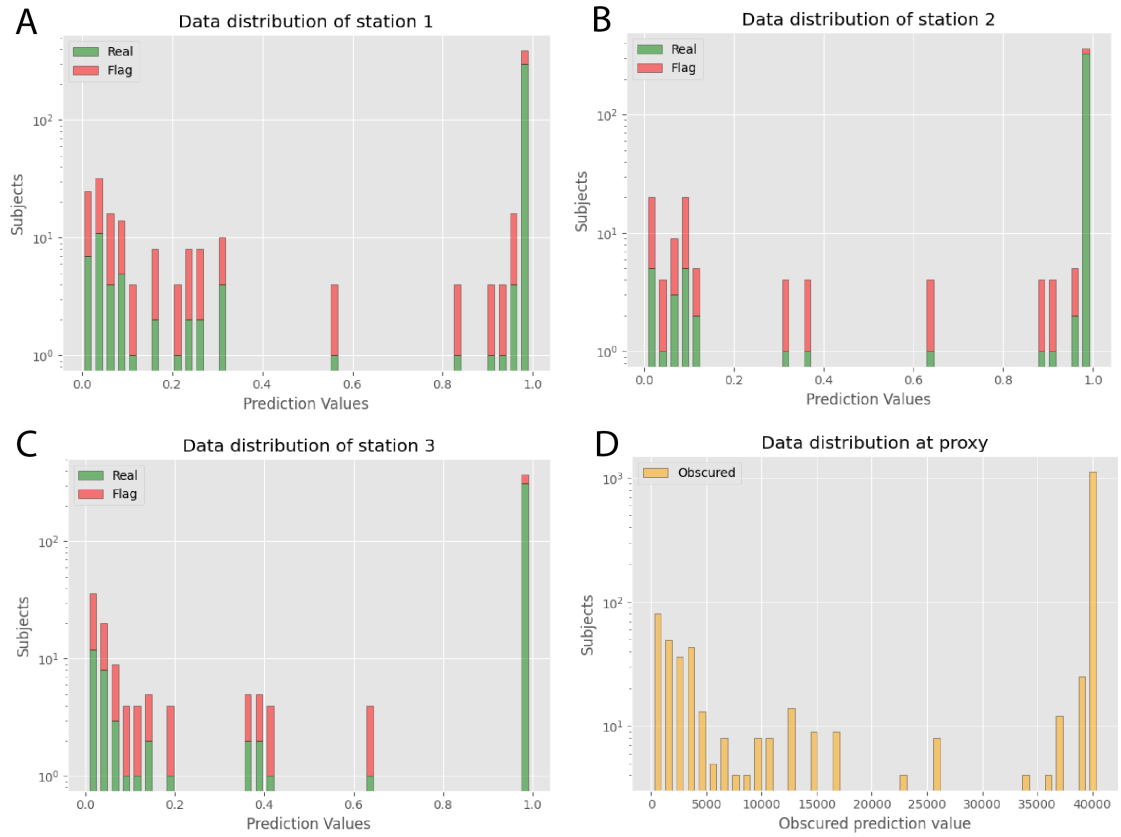


Figure 4.3: (A-C) Input data site 1-3 Prediction value distribution at Station 1 to 3, highlighting the balance between 'Real' and 'Flag' categories. Peaks at both ends of the prediction range indicate high variability in the dataset.
(D) Input data at Proxy: Obscured prediction value distribution at the proxy station, showing transformed prediction data aggregated from all sites. The distribution indicates effective obfuscation of individual prediction values.

We computed the corresponding DPPE- and DPPA-AUC components for our method and derived the difference to the GT. Our experiments included the DPPE- and DPPA-AUC method compared in terms of time and deviations from GT AUC score. The results in Table 4.4 highlight a trade-off between accuracy and computational efficiency: DPPE-AUC achieves near-perfect accuracy with a minimal difference from the GT but incurs a higher computational cost, while DPPA-AUC demonstrates faster runtime with a slightly larger but acceptable deviation from the GT AUC.

For each run, Across ten runs, the average difference between the GT and DPPE-AUC values was 10^{-16}, likely attributable to floating-point arithmetic precision limits in Python.

Method	Time (s)	Deviation to GT	Thresholds
DPPE-AUC	38.71	1.11E-16	145
DPPA-AUC	26.499	0.0037	100

Table 4.4: Comparison of DPPE-AUC and DPPA-AUC methods based on runtime, the difference from the ground truth AUC, and the number of thresholds used.

4.4.2 Experiments

In the showcase described above, we observed a high AUC score, which resulted in numerous repetitive prediction values. This led to data bias, making a performance evaluation based on increasing sample size and station count impractical. To address this, we generated synthetic data to examine the impact of these values more accurately. At each station, subjects were randomly assigned prediction values between 0 and 1 and labels from the set $0, 1$. Random flag patients were created identically to those in the showcase. We compared the GT AUC with the DPPE- and DPPA-AUC, excluding flag samples in the GT calculation to validate our method. Across experiments, differences between GT and DPPE-AUC values remained consistently at 10^{-18}.

Experiment 1

In this experiment, we analyzed the runtime performance of both methods compared to GT AUC. The total dataset size was fixed at 1500 subjects, combining both actual and dummy (flag) patients. The experiment aimed to simulate a realistic PHT-meDIC environment, where an increasing number of participating stations (or input parties) operate decentralized. As the number of sites increased, the sample size per site was proportionally reduced per site.

This configuration allowed us to analyze how computational efficiency scales with the number of sites while maintaining a constant total sample size. Each site independently computed its local AUC contribution iteratively, with the overall execution time determined by the combined decentralized encryption procedures. The synthetic data generation in this experiment mimicked a real-world scenario by introducing biases around prediction values near 0 and 1. Flag data was generated only for existing subjects to obscure the data distribution, ensuring no increase in the number of threshold values. This approach effectively minimized computational overhead while preserving data realism.

As expected, the runtime increased with the number of participating sites in Fig 4.4 due to decentralized computations overhead. Both methods demonstrated robust performance, though DPPE-AUC required slightly more time than DPPA-AUC due to its complete processing of input data, while DPPA-AUC only uses 100 approximation points.

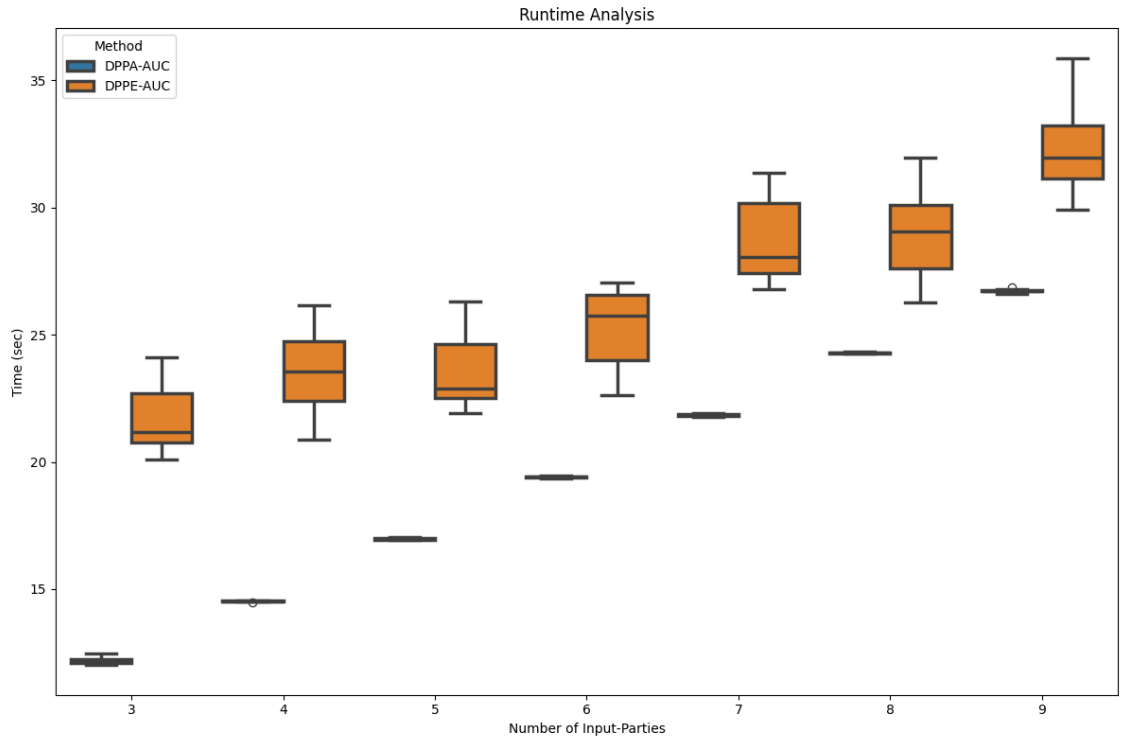


Figure 4.4: Increasing Number of Input Parties Runtime performance over 10 runs of DPPE-AUC and DPPA-AUC methods with a fixed total dataset size of 1500 subjects. The number of participating sites (input parties) varies between 3 and 9, illustrating the scaling behavior of both methods. While execution time increased with more stations, both methods maintained robust performance.

Experiment 2

In this experiment, we examined how runtime performance scales with increasing sample sizes while keeping the number of stations fixed at three. This setup evaluated the computational scalability of the DPPE-AUC method compared to the DPPA-AUC method as the amount of data per station increases.

The evaluation considered up to 23,000 subjects, reflecting a practical sample size in typical PHT-meDIC studies. The results in Fig 4.5 demonstrated a linear increase in execution time for the DPPE-AUC method as the sample size grew. This behavior aligns with the computational demands of DPPE-AUC, where encryption and processing steps scale directly with the input data to process. In contrast, DPPA-AUC's constant runtime offers a clear advantage in large-scale applications where efficiency and fixed computational cost are critical. The data generation followed a uniform random distribution, representing a worst-case scenario in terms of computational overhead.

These findings highlight distinct scalability characteristics for the two methods. While DPPE-AUC is well-suited for moderate-scale datasets, DPPA-AUC's fixed runtime offers a compelling

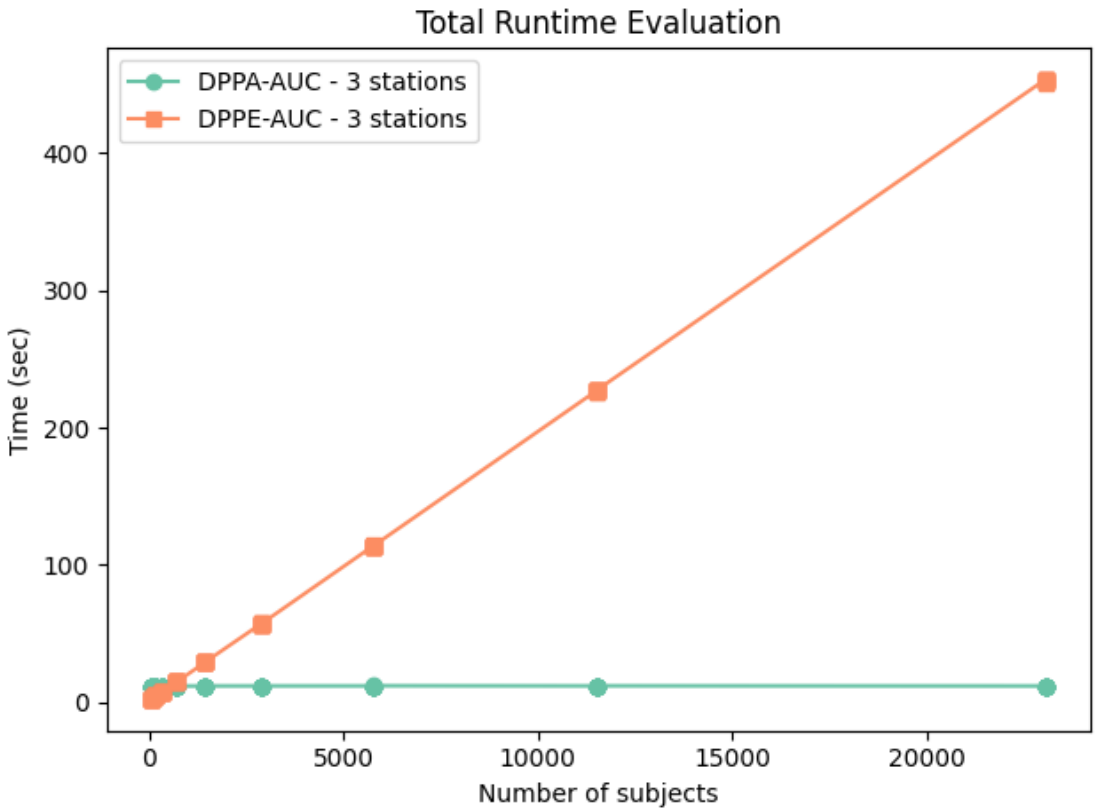


Figure 4.5: Increasing Number of Input Samples Runtime evaluation of DPPE-AUC and DPPA-AUC methods with a fixed number of stations (3) and increasing dataset sizes (up to 23,000 subjects). DPPE-AUC exhibits linear scaling with increasing sample sizes, while DPPA-AUC maintains a constant runtime due to its reliance on a fixed number of decision points. The results emphasize DPPA-AUC's suitability for large-scale datasets.

advantage for large-scale applications where efficiency and predictable computational costs are critical.

4.5 Conducting Cross-Infrastructural Analysis

Upon completion of our PoC development, we validate our approach within a real-world scenario and real patient data. For our interoperability study, we have opted to conduct a basic statistical analysis of the leukodystrophy data at hand. Since our concept is based on container trains, we argue that other trains with various analytical functions will also operate successfully within our PoC. The process of our evaluation is shown in Figure 4.6.

After we have deployed our described components in the previous section and in Figure 3.11 (Step 1), we set up a station at each of the three sites - Aachen, Tübingen, and Leipzig - and ensure that the REDCap databases at these locations are accessible via these stations (Step 2).

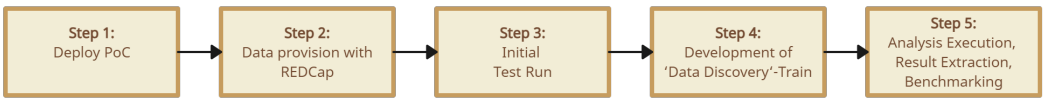


Figure 4.6: Validation process for the PoC. Step 1 encompasses the deployment of the PoC (Layer 1-4). In Step 2, we set up the data provision using REDCap. Step 3 involves an initial test run to ensure operational functionality. Step 4 is dedicated to the development of the 'Data Discovery' train that performs the analysis. The final Step 5 includes executing the analysis, extracting results, and benchmarking our PoC to acquire quantitative performance metrics for our interoperability concept.

After passing the functional tests (Step 3), our infrastructure is now operational. With access to the data, we can proceed with data analysis. The train design is part of the next section (Step 4).

4.5.1 Designing the 'Data Discovery'-Train

The purpose of this study 'Data Discovery' is to initially inspect the data provided by the DICs, which, for example, can 1) support the identification of analogous studies that can be used as a reference point, 2) provide insights into the data quality, or 3) can be used as starting point for the design of a more sophisticated follow-up data analysis study. Since our work focuses on container trains, we create the train for our data analysis using Python (Version 3.10 - https://peps.python.org/pep-0619/) and Docker (https://www.docker.com) as containerization technology. The train consists of two steps. The initial step involves loading the REDCap data into the train, while the subsequent step involves the core data processing unit responsible for generating statistics and producing a PDF report for researchers. To streamline the data querying process, we have developed a custom data importer routine. This routine retrieves the data via direct access to the REDCap database, using the necessary credentials entered by the station administrator before initiating the train execution. For the creation of the report representing the analysis results, we gather several statistics. First, we determine the total number of male and female patients at each station (Table 4.5). Furthermore, we determine the age distribution at each station according to a k-anonymity level of 5 (Figure 4.7)[173]. Lastly, we create plots showing the counts of the Baseline, Examination, and Genetic questionnaires (Figure 4.8).

4.5.2 Execution and results of the analysis

After creating the container train, we executed the train (Step 5) according to our workflow depicted in Figure 3.11. In summary, Figures 4.7 and 4.8 provide an overview of the data underlying each station, providing information, such as the age distributions or the number of the Baseline, Examination, and Genetics sections in the questionnaire. The investigation of

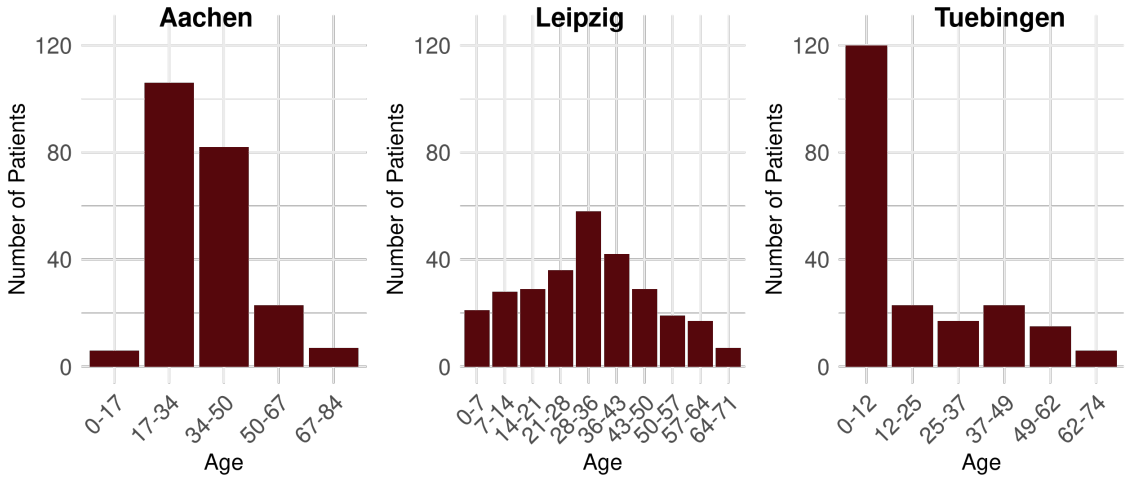


Figure 4.7: The age distribution from the Aachen, Leipzig and Tübingen. We organized the number of patients into bins that are tailored to ensure compliance with a k-anonymity threshold of k = 5.

the age distribution, as shown in Figure 4.7, indicates that in Tübingen, there is a balanced representation of male and female patients, most under the age of 20 years. This demographic trend is probably a result of the focus of the medical center on the treatment of leukodystrophies in children. On the contrary, Leipzig has a higher proportion of male patients and the majority of them are adults. Lastly, Aachen and Leipzig contribute more male patients than female (Table 4.5). Importantly, the dataset from Aachen is missing genetic data, as shown in Figure 4.8, in contrast to the complete sets of questionnaires available from Leipzig and Tübingen. Measuring the impact of our interoperability concept in terms of the size of the dataset (number of patients), we come to the following conclusion. In cases where only one infrastructure is used, we either have 485 patients (Aachen + Leipzig) or 216 (Tübingen). However, through the interoperability of both ecosystems, we have achieved a total patient count of 701 (Aachen + Leipzig + Tübingen). These results highlight the success of our approach, enabling us to expand the total dataset and enhance our data-sharing capabilities.

Sex	Count
Male	140
Female	95

(a) Aachen

Sex	Count
Male	179
Female	71

(b) Leipzig

Sex	Count
Male	116
Female	100

(c) Tübingen

Table 4.5: Number of males and females from the stations in Aachen, Leipzig, and Tübingen.

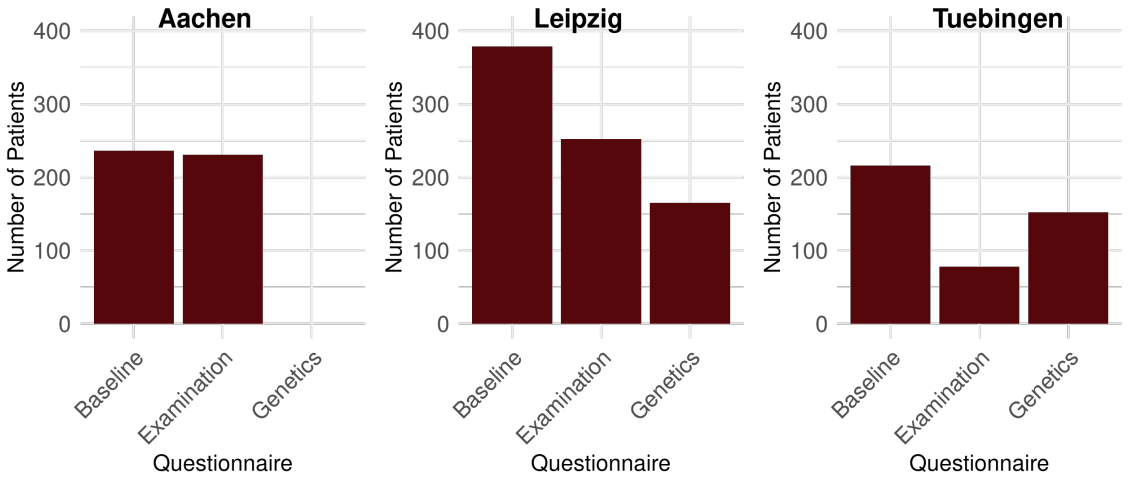


Figure 4.8: The count of patients across the three sections: Baseline, Examination and Genetics.

Chapter 5

Discussion

The content of Section 5.2 constitutes an extended version of the manuscript:

Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train (PHT-meDIC)[31]

The content in Section 5.3 constitutes an extended version of the manuscript:

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC[33]

The content in Section 5.4 constitutes an extended version of the manuscript:

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis[32]

5.1 Introduction

The ensuing section examines the outcomes of PHT-meDIC experiments and its architectural framework and highlights certain constraints observed in the present implementation. It will address both generic limitations inherent to distributed machine learning and the specific burdens confronting all current platforms in medical informatics. The final subsection will explore the results related to the interoperability of PHT, encompassing a broader discussion on the balance between interoperability and multi-homing. This analysis highlights the complex interplay between ensuring seamless system interoperability and supporting multiple services in a 'multi-homing' scenario.

5.2 PHT-meDIC Limitations

Our container-based architecture is designed for flexibility, allowing researchers to run complex analytics on hospital infrastructure without additional installations. Unlike platforms like DataSHIELD, which require separate local package updates and approvals, PHT-meDIC operates with a general authorization model but mandates explicit approval for each analysis. This process retains a necessary "human in the loop" to ensure the code follows protocol.

A current assumption is that all participating sites contribute original patient data. While synthetic data generation enables rapid prototyping, it complicates validating data plausibility. Future efforts will incorporate methods to verify data authenticity and quality.

The platform currently processes algorithms sequentially, which is adequate for many bioinformatics and machine learning tasks but can create challenges when data are unevenly distributed among sites. Sequential execution may affect model quality and reproducibility, as varying routes can yield different prediction outcomes. Moreover, if participating sites have comparable sample sizes and resources, execution time increases linearly with the number of sites. Parallel processing could address these execution time concerns.

Another optimization opportunity lies in reducing communication overhead. Presently, the system updates algorithm results by returning the entire container image; returning only the results would be more efficient. Key generation, algorithm signing, and result decryption also require a separate desktop application, which can complicate the user experience.

Performing distributed analysis on real patient data in German university hospitals must adhere to data protection laws, ethics board approvals, and IT security assessments. Within the German MII framework, these processes involve Use and Access Committees (UAC) and inter-site agreements. The new German National Portal for Medical Research Data (FDPG) streamlines these procedures, paving the way for broader interoperability, single sign-on, and more efficient proposal and contract management.

Users' partial access to data simplifies software development, allowing them to verify schemas and run distributed analyses. However, a more advanced data discovery process with integrated privacy checks and security clearances could automatically execute analyses and foster true federated learning. We plan to address these enhancements in future development, ultimately providing a secure, reliable platform that respects governance requirements and meets researchers' needs.

5.3 PP-AUC Limitations

The primary challenge identified in the current implementation for both methods is the increase in runtime as the number of participants grows, largely due to the iterative execution within the PHT-meDIC framework. To mitigate this, transitioning to a federated execution of the

station-side algorithms and final computation steps would enable parallel processing, thereby significantly reducing overall total runtime, as evidenced by our experimental results.

Furthermore, the runtime performance of DPPE-AUC strongly depends on how prediction values are distributed. If every prediction value is unique, the sorting and processing steps reach their highest overhead. To both obscure the real distribution and limit uniqueness, our current approach inserts "dummy" prediction values drawn from the existing set, thereby reducing the overall number of distinct values. While this technique conceals the original distribution effectively, it remains a straightforward implementation. Consequently, there is still significant room for optimization, particularly at the proxy station during final sorting and processing, to further enhance execution speed.

The potential of achieving marginal AUC improvements, such as the ≈ 0.0037 difference provided by DPPE-AUC compared to the DPPA-AUC method, is mainly relevant in critical healthcare domains. In rare disease diagnoses, small gains in AUC can significantly improve early detection rates and reduce misdiagnoses[174]. Similarly, such improvements enhance early tumor detection in radiology and lower false negatives[175]. In ICU sepsis prediction, even minor performance increases can directly impact timely interventions and patient outcomes[176]. Furthermore, in personalized medicine and adverse drug reaction prediction, higher precision minimizes unnecessary treatments and associated risks[177][178]. These examples highlight the importance of fine-grained performance gains in machine learning models for healthcare applications, emphasizing the value of DPPE-AUC.

5.4 PHT Interoperability

PHT Interoperability vs. 'Multi-Homing'

An essential element of the discussion revolves around the comparison between the benefits of interoperability and what we term multi-homing - *'the situation in which users tend to use several competing platform services in parallel'* as defined by the Directorate-General for Communications Networks of the European Commission[59]. In our specific PHT case, this implies that a single institution deploys at least two station software applications from at least two distinct PHT implementations and operates them concurrently. This scenario would make interoperability, as described in this work, redundant. Throughout our data analysis and execution of the train, we identified various factors that posed challenges to the multi-homing of both platforms. In the following, we discuss several disadvantages of multi-homing associated with hosting multiple platforms at a single location, in contrast to achieving interoperability:

- **Increased Complexity:** Multi-homing introduces an increased level of complexity. Institutional personnel must undergo training to proficiently manage and utilize two distinct systems, which requires significant investment in education and adaptation. In our

specific situation, for instance, we would need to provide training to the personnel in Tübingen for operating the PADME station.

- **Governance Challenges:** Maintaining two infrastructures can entail governance hurdles. This involves obtaining approvals and ensuring compliance for both systems, which can be cumbersome and time-consuming. Further, the introduction of a potential redundancy might raise the question of whether a second platform is even necessary. This would prevent the institutional authorities from multi-homing, which would lead to a *quasi* data silo as the data is only accessible through a single platform. This approach would limit the access of data to a certain initiative, project, or geographic area. In our case, for instance, PADME is set up in Aachen and Leipzig, while PHT-meDIC is based in Tübingen (see Figure 3.7). However, data analysis should not only be limited to accessible locations but should also encourage cooperation between scientists and data providers on an interregional or international scale. Thus, achieving interoperability among these platforms allows for data access and sharing across different regions or initiatives. In the particular scenario we encountered in this work, the interoperability framework we developed allowed for the integration of data from three distinct institutions into our study that have been affiliated with different initiatives (e.g., SMITH or DIFUTURE).

- **Security Risks:** Especially in a sensitive environment such as healthcare, multi-homing opens the door to additional security concerns (related to the governance aspect). With two ecosystem endpoints active, institutions face additional attack vectors that could potentially make them more vulnerable to security breaches and vulnerabilities. Trivially, achieving interoperability with other platforms necessitates just one endpoint.

- **Resource Consumption:** Running several PHT stations simultaneously requires more resources. This results in increased operational costs and may strain the institution's IT infrastructure (related to the increased complexity aspect). Depending on where the software is deployed (e.g., in a cloud environment), an increased consumption of the resources might also impact the financial resources of an institution.

- **Data Integration:** Depending on the specifics of the infrastructures, data integration becomes a challenge. Data must potentially be integrated multiple times and made accessible through the PHT station application. Since data integration is inherently a time-intensive process, this stage would involve duplicating and storing data multiple times to ensure compatibility with multiple infrastructures (related to the aspect of resource consumption). Additionally, each 'multi-home' may establish its own data schema and involve a unique data storage technology. When a clear schema mapping exists, this necessitates an extra step of transforming data from one schema format to another. If there is no defined schema mapping, this process becomes even more complex. In this

work, we addressed this challenge through our Layer 0 defining a global data schema for the provided data to enable interoperability on the data level for both infrastructures.

- **Analysis Development Overhead:** When adopting a multi-homing approach, analysis development becomes a duplicate effort. In a heterogeneous landscape, where certain institutions use multiple PHT applications, while others do not, analyses may need to be developed separately for each system. Finally, merging results, either ad hoc or manually, adds an additional layer of complexity and time-consuming work for analysis developers. The debugging overhead we mentioned earlier, which we have encountered during our feasibility study, could potentially also have a negative impact on this aspect, as it can occur twice. Especially, this is a challenge that can be addressed through interoperability. With interoperability, the analysis code only needs to be developed once and can be effortlessly distributed across the PHT ecosystems.

Limitations

Despite successfully achieving technical interoperability between two distinct PHT infrastructures, our approach has certain limitations. First, it addresses only one aspect of the broader interoperability challenge: clinical, governance, and regulatory considerations[60]. While our metadata schema (Layer 3) includes core elements for two security protocols and business logic, additional work is needed to ensure wider semantic interoperability. Similarly, the data schema (Layer 0) was designed for a specific feasibility study and may not directly apply to other use cases.

Second, our study focuses on a limited technology stack in two PHT infrastructures; its suitability for larger-scale integration or other technologies has yet to be determined. Although our approach relies on container trains, other train types that do not use containerization may require different strategies[157].

Third, our hierarchical identifier system introduces a top-level authority, which could serve as a single point of failure. While caching station identifiers reduces this risk, alternative solutions, such as decentralized identifiers, may be more robust[179].

Addressing these points can advance future research to refine and scale the proposed interoperability concept. Broad community alignment on global station identifiers or a standardized security protocol can further streamline interoperable PHT platforms. Our five-layer model provides an initial framework for these efforts. As part of the German PrivateAim project and its Federated Learning and Analysis in Medicine (FLAME) initiative[180], this work establishes a conceptual and technical foundation for a collaborative platform for distributed analytics. In the future, we aim to strengthen collaboration between the PADME and PHT-meDIC infrastructures to enhance interoperability.

5.5 AI in Healthcare and Ethics

Integrating AI into healthcare offers transformative potential yet also raises important ethical considerations involving data privacy, informed consent, transparency, bias, accountability, autonomy, and equitable access. Data privacy and consent remain critical because AI-based healthcare models rely on large volumes of patient data and must comply with regulations such as HIPAA or GDPR. Distributed learning can help mitigate data exposure risks but introduces new governance challenges. Modern AI systems, often seen as black boxes, require careful development to ensure transparency and explainability so clinicians and patients can understand and trust AI-driven recommendations. Bias and equity concerns persist because AI models trained on non-representative datasets can perpetuate unequal treatment of particular demographic groups. These biases highlight the need for ongoing monitoring and model training on diverse datasets to achieve equitable healthcare outcomes. Accountability remains complex when AI informs clinical decisions, as responsibilities span developers, healthcare providers, and others; frameworks such as the CRA and the AI Act are emerging to define roles and liabilities in the event of errors. Patient autonomy calls for AI tools that support rather than replace healthcare professionals. It emphasizes the importance of communicating how AI is integrated into patient care and preserving individuals' control over their personal health choices. Fair access to AI's benefits in healthcare should also be ensured, as well-designed and ethically guided AI solutions have the potential to reduce rather than worsen existing disparities. As AI continues to reshape healthcare, ethical considerations must remain at the forefront. To realize AI's potential responsible, innovation must be balanced with patient well-being, fairness, and trust.

Chapter 6

Conclusion and Outlook

The content in Section 6.1.1 constitutes an extended version of the manuscript:

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis[32]

The content in Section 6.1.2 constitutes an extended version of the manuscript:

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC[33]

This work shows how distributed learning is developed from a theoretical concept of the Personal Health Train to an actual platform PHT-meDIC that is deployable at university hospitals. The last year's work was only a starting point upon which several extensions and modifications can be built. The following conclusion 6.1 is related to PHT-meDIC architecture, and the subsection 6.1.1 is related to PHT interoperability. Section 6.2 provides details on how future projects and research will benefit from the work of the PHT-meDIC.

6.1 PHT-meDIC - a proof-of-concept

Our innovative architecture for the PHT facilitates complex analyses of patient data securely across multiple hospitals. At all times, the control of patient data remains within the hospital's. Our security framework ensures that only the analysis participants and the user can decrypt the results. Additionally, our security concept guarantees that any tampering with the queries or algorithms, whether during execution or in transit, is immediately identifiable.

Our software is currently in productive use and is accessible under a permissive open-source license.

The data discovery experiment involving real patient data across two hospitals showcases the versatility of our PHT-meDIC system. The bioinformatics and image analyses are complete, distinct analyses of large-scale input data, demonstrating the framework's flexibility. Furthermore, it highlights our solution's ability to adapt to different analyses without extra configurations or installations at hospitals, a significant challenge for other available systems, and the possibility of combining several tasks in one analysis, like PETs with highly extensive working pipelines, such as HLA typing. Our architecture supports a wide range of online analyses, from basic statistical evaluations to sophisticated privacy-preserving or SMPC techniques, all within the same framework or analysis.

6.1.1 The Costs of Our Interoperability Concept

Based on aspects in the discussion 5.4 and the results of our feasibility study, we conclude that the multi-homing of similar infrastructures presents various drawbacks concerning efficiency, security, and governance, making interoperability desirable. Particularly regarding governance, achieving interoperability emerges as a necessary, and possibly the sole, solution in scenarios where the landscape is heterogeneous and multi-homing is not practiced—see our scenario of the feasibility study. Additionally, at this level of scale, it cannot be assumed that there will be only a single data analysis infrastructure covering a nationwide or international scope, which would make interoperability unnecessary. Therefore, considering these factors, interoperability could be an appropriate strategy to address the aforementioned challenges. Ultimately and importantly, multi-homing can still be a viable option, especially beneficial when multiple infrastructures that offer complementary features, such as diverse functionalities, are hosted together.

This brings us to the question of the expenses or overhead associated with our concept of interoperability. A crucial aspect to consider is the incorporation of the containerization of the security protocol(s). This addition involves an extra layer in the local protocol and introduces possible time and resource demands, as the additional container must be pulled before the train execution. In our PoC, the security container has a size of 460.23 MB, which can be pulled once for future reuse. The security container is also considerably smaller than the train used in the feasibility study: 1.22 GB. In contrast with the multi-homing scenario, we would have to install either a station of approximately 2.9 GB (PADME) or a size of 1.7 GB (PHT-meDIC). Hence, our concept is more efficient in terms of resources. Since our interoperability concept encompasses the integration of multiple security protocols (in the form of an envelope), it results in an extended overall duration of the security protocol at the stations. In our specific situation and with our involved hardware, both decryption processes require 3 seconds each, resulting in a total decryption time of 6 seconds. Conversely, the encryption procedures take 8 seconds for the first step and an additional 15 seconds for the second step. The retrieval

of the security container takes 6 seconds and only needs to be done once. Consequently, the added time required for our nested encryption approach is relatively small compared to the manual effort to manage two concurrent infrastructures during analysis. Regarding our business logic, in layer 4, the reloading/copying of the contents from one train to another takes less than 10 seconds. However, this reloading depends on the size of the contents and whether the images are already in the cache. If they are not, they need to be downloaded once. Furthermore, the introduction of an additional authority at the top level, in turn, requires potential administrative efforts regarding the creation and maintenance of standardized global identifiers. In our PoC, the SR is considered the station's central registry. As we have shown in our previous work, SR can be seamlessly integrated into the station installation workflow[154]. Therefore, administrative efforts are manageable.

6.1.2 Using PP-AUC with PHT-meDIC

In this work, we introduced DPPE-AUC and DPPA-AUC, two cryptographic protocols that enable the computation of the global AUC in distributed and privacy-sensitive settings. DPPE-AUC calculates the AUC exactly, making it particularly useful when high-precision model assessment is important; it maintains linear scaling with the dataset size. DPPA-AUC instead relies on a sampling strategy with a fixed set of thresholds, achieving constant runtimes suitable for larger or more diverse datasets where computational efficiency is critical. Both methods are fully integrated into the PHT-meDIC framework, underscoring their practical feasibility.

By combining modified Paillier encryption, symmetric and asymmetric cryptography, and randomized encoding techniques, these protocols protect sensitive model predictions and labels while still allowing global performance metrics to be derived. This leads to a robust and flexible system that meets the privacy demands of multi-institutional collaborations in healthcare and beyond. In future work, we plan to optimize the protocols further by employing parallelization strategies, which could substantially reduce runtime in large-scale distributed networks. Through such optimizations, we aim to make secure, privacy-preserving performance evaluation an even more powerful tool for real-world medical and scientific applications.

6.2 From Train to FLAME

In early 2023, the four consortia of the MII joined forces, leveraging the strengths of over 17 university hospital partners, to develop an advanced distributed learning platform named FLAME (Federated Learning and Analysis MEthodology) as part of the PrivateAim[181] project. This work aims to integrate and enhance the functionalities of existing MII-developed platforms—namely PADME, Kaapana, and PHT-meDIC—into an integrated platform. Under the technical leadership of Tübingen University Hospital, our objective is to construct, implement,

and standardize a platform that facilitates more sophisticated and efficient access to patient data and distribution of analysis tasks. This collaborative effort is structured around a series of work packages. The outlook can be categorized into technical, organizational, and ethical components.

6.2.1 Technical

From a technical standpoint, the current execution strategy operated by PHT-meDIC, represented by its iterative execution and result updating process, stands out for its simplicity but is also noted for being time-intensive. A significant advancement offered for FLAME is the parallel execution of analyses, including global result updates through an enhanced aggregation service, marking a key enhancement in processing efficiency.

The approval and execution of the code are only done with a human in the loop - even if this person always has to stay there - there are various possible technical contributions to assist the approval and execution of algorithms on each site. For instance, within FLAME, incorporating a difference checker for multiple analysis submissi«ons could streamline the approval process by providing a comprehensive summary when several submissions require vetting. Adopting large language models to summarize submitted code could simplify the approval process for individuals with limited technical expertise. Additionally, integrating previously approved code into a reusable library could minimize the need for reapproval, except in instances of data or parameter modifications. Moreover, introducing secure execution environments, such as sandboxes or secured enclaves, could offer a robust mechanism to technically ensure the integrity of executed code, safeguarding against malicious attempts.

Another notable innovation in FLAME is the optimization of result updates through a monitored 'result API,' which replaces the time and storage-intensive process of returning entire docker images. This streamlined approach is expected to significantly reduce network and storage demands, thereby accelerating the computation of results.

Moreover, the architectural shift in key creation and result decryption processes — from a standalone application to integration within the user's browser storage — promises to alleviate installation and maintenance burdens on users' devices, enhancing overall usability.

Despite the technical feasibility of offering users in the future FLAME architecture the autonomy to select site-independent update strategies, the varying implications of these choices necessitate a balanced approach. While such flexibility is desirable from a research perspective, the practical need for users to understand the outcomes of different strategies, coupled with hospitals' preferences to maintain control over result updates, underscores the importance of addressing these considerations not just technically but also organizationally.

6.2.2 Organizational

In the context of organizational benefits, the FLAME platform introduces an essential advancement by establishing a unified deployment and release framework for federated learning platforms across German university hospitals. This harmonization facilitates a streamlined process for feature prioritization and development. Moreover, the diverse regulations of data protection officers across various states will advise creating legally binding agreements, enabling a more coherent approach to compliance. Implementing projects like FLAME simplifies identifying and resolving discrepancies across the system. Additionally, beyond offering technical support in code review processes at different nodes, introducing a potential "trust or privilege model" promises to reduce the time for analysis approval significantly. This model would enable particular project and user-specific functionalities to avoid the need for separate approvals at each site, contingent on predefined conditions in standardized inter-site agreements. These conditions might include assessments of potential impacts or risks associated with the data of individual projects, acknowledging that the sensitivity of data varies (e.g., genomic data poses a greater privacy risk than blood pressure information). From an organizational perspective, an alternative to technical interventions could be adopting a majority voting system for code approvals. Under this system, if most sites consent to executing an analysis, the remaining sites would automatically grant their approval. While technically straightforward, this approach requires mutual trust and cooperation and the development of robust guidelines. These guidelines will ensure fairness and efficacy in decision-making processes, all under the guarantee of security and privacy.

Within this framework, advanced researchers would benefit from the ability to employ dynamic execution patterns and more sophisticated aggregation strategies in algorithm execution. This flexibility will equip users with the essential resources to create robust and reproducible models.

Standardizing national federated learning platforms presents a strategic pathway toward achieving greater interoperability with other European and international emerging standards. Such standardization efforts should inherently prioritize interactive improvement processes and embrace open-source principles, such as PrivateAim's FLAME.

Establishing a standardized framework for federated learning platforms can facilitate seamless integration and algorithm interactions across borders, enhancing collaborative research and development efforts. This accelerates innovation and ensures that advancements in healthcare and technology are universally accessible and benefit a broader population. Interactive improvement, a continuous cycle of feedback and enhancement, ensures that the platforms evolve in response to new challenges, technological advancements, and user needs. This dynamic approach allows for much faster adaptation to changing regulatory landscapes and emerging ethical considerations.

Incorporating open-source principles is pivotal for fostering a collaborative environment where developers, researchers, and stakeholders from various sectors can contribute to and benefit from the collective knowledge base. Open-source software promotes transparency, security, and community-driven development, which is essential for building trust and ensuring the widespread adoption of federated learning platforms.

Moreover, the open-source approach aligns with the values of scientific research, where accessibility to data and methodologies underpins reproducibility and validation of results. It also enables smaller institutions and countries to participate in and contribute to global research efforts without the barriers imposed by proprietary systems.

6.2.3 Ethical

As we look to the future of federated learning platforms and their integration within healthcare settings, particularly within the context of future platforms, a pivotal shift towards patient-centric data control is desirable. Today, the legal basis is the patients ' consent. This consent can change over time, and patients would like to be informed of improved medications and treatments based on their contributed data. For dynamic consenting and traceability, blockchain approaches are promising.

This evolution of future platforms aligns with the core principles of the EU General Data Protection Regulation (GDPR), the Cybersecurity Act, the Cyber Resilience Act (CRA), and the Artificial Intelligence Act (AI Act), setting a new standard for data privacy and security in the era of advanced analytics and artificial intelligence.

The GDPR has laid a strong foundation for data protection, emphasizing the right to privacy and control over personal data for individuals within the European Union. In the realm of federated learning in healthcare, this translates to an increased emphasis on transparency and patient consent. Patients are not merely subjects of data but active participants who vote on how their data is used, shared, and protected. Looking forward, mechanisms that enable patients to easily manage their consent preferences, understand the purpose of data processing, and exercise their right to data portability will become integral to healthcare platforms.

Integrating the CRA into standardizing national federated learning platforms is a crucial step toward enhancing their interoperability. This strategic alignment emphasizes the importance of cybersecurity and resilience within the framework of digital healthcare innovation. By adopting the principles of the CRA, the standardization process will inherently prioritize not only interoperability but also the security and robustness of federated learning environments.

The AI Act aims to set comprehensive rules for developing, deploying, and using artificial intelligence within the EU, focusing on high-risk applications, including those in healthcare. This regulatory framework will likely necessitate significant adjustments in how federated learning platforms operate, particularly ensuring that AI algorithms handling patient data are

transparent, equitable, and underpinned by robust ethical standards. A critical outlook is establishing systems that allow patients to understand and control the AI-driven use of their data, ensuring that AI acts in their best interests.

Incorporating all of these future regulatory laws and guidances, future federated learning platforms will need to integrate sophisticated data control mechanisms that are directly accessible to patients. These include digital consent management tools, data usage dashboards, and AI explainability interfaces. Such tools will empower patients to make informed decisions about their data, understand how it contributes to research and clinical care, and retain control over its usage.

Achieving this vision requires collaboration among technologists, legal experts, healthcare providers, and patients. Developing standards and practices that align with GDPR, CRA, and the AI Act while meeting the operational needs of healthcare research will be challenging. However, it represents a crucial step towards realizing a healthcare ecosystem that respects patient autonomy, enhances data security, and harnesses the power of AI for the greater good.

In conclusion, the outlook for federated learning platforms under the GDPR, CRA, and the AI Act has increased patient empowerment and data control, strengthening the ethical basis of such developments. As we navigate this evolving landscape, the focus will undoubtedly shift towards creating a more transparent, secure, and patient-centered approach to data usage in healthcare.

Bibliography

[1] Xiaoxuan Liu, Livia Faes, Aditya U. Kale, Siegfried K. Wagner, Dun Jack Fu, Alice Bruynseels, Thushika Mahendiran, Gabriella Moraes, Mohith Shamdas, Christoph Kern, Joseph R. Ledsam, Martin K. Schmid, Konstantinos Balaskas, Eric J. Topol, Lucas M. Bachmann, Pearse A. Keane, and Alastair K. Denniston. A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis. *The Lancet Digital Health*, 1(6):e271–e297, Oct 2019. 1

[2] Claudio Luchini, Antonio Pea, and Aldo Scarpa. Artificial intelligence in oncology: current applications and future perspectives. *British Journal of Cancer*, 126(1):4–9, Jan 2022. 1

[3] Abhishek Aggarwal, Cheuk Chi Tam, Dezhi Wu, Xiaoming Li, and Shan Qiao. Artificial intelligence–based chatbots for promoting health behavioral changes: Systematic review. *J Med Internet Res*, 25:e40789, Feb 2023. 1

[4] Gregor Urban, Kevin Bache, Duc T T Phan, Agua Sobrino, Alexander K Shmakov, Stephanie J Hachey, Christopher C W Hughes, and Pierre Baldi. Deep learning for drug discovery and cancer research: Automated analysis of vascularization images. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 16(3):1029–1035, May 2019. 1

[5] IBM Education. The benefits of ai in healthcare. `https://www.ibm.com/blog/the-benefits-of-ai-in-healthcare/`, 2023. Accessed: 2025-01-16. 1

[6] MOHAMMAD MUZAFFAR MIR, GULZAR MUZAFFAR MIR, NADEEM TUFAIL RAINA, SABA MUZAFFAR MIR, SADAF MUZAFFAR MIR, ELHADI MISKEEN, MUFFARAH HAMID ALHARTHI, and MOHANNAD MOHAMMAD S. ALAMRI. Application of artificial intelligence in medical education: Current scenario and future perspectives. *Journal of Advances in Medical Education & Professionalism*, 11(3):133–140, 2023. 1

[7] Mohammad Muzaffar Mir, Gulzar Muzaffar Mir, Nadeem Tufail Raina, Saba Muzaffar Mir, Sadaf Muzaffar Mir, Elhadi Miskeen, Muffarah Hamid Alharthi, and Mohannad Mohammad S Alamri. Application of artificial intelligence in medical education: Current scenario and future perspectives. *J. Adv. Med. Educ. Prof.*, 11(3):133–140, July 2023. 2

[8] Cynthia Rudin and Joanna Radin. Why Are We Using Black Box Models in AI When We Don't Need To? A Lesson From an Explainable AI Competition. *Harvard Data Science Review*, 1(2), nov 22 2019. https://hdsr.mitpress.mit.edu/pub/f9kuryi8. 2

[9] Ahmad Chaddad, Jihao Peng, Jian Xu, and Ahmed Bouridane. Survey of explainable AI techniques in healthcare. *Sensors (Basel)*, 23(2):634, January 2023. 2

[10] Junaid Bajwa, Usman Munir, Aditya Nori, and Bryan Williams. Artificial intelligence in healthcare: transforming the practice of medicine. *Future Healthc J*, 8(2):e188–e194, July 2021. 2

[11] Marzyeh Ghassemi, Tristan Naumann, Peter Schulam, Andrew L Beam, Irene Y Chen, and Rajesh Ranganath. Practical guidance on artificial intelligence for health-care data. *Lancet Digit. Health*, 1(4):e157–e159, August 2019. 2

[12] Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts, 2021. 2, 3, 11

[13] European Union. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation) (text with eea relevance). *OJ*, L 119:1–88, 2016-04-05. 2

[14] Centers for Medicare & Medicaid Services. The Health Insurance Portability and Accountability Act of 1996 (HIPAA). Online at http://www.cms.hhs.gov/hipaa/, 1996. 2

[15] GO Fair. GO FAIR initiative. `https://go-fair.org`, 2018. Accessed: 2025-01-13. 2, 8, 14, 43

[16] Medizininformatik-Initiative. Vernetzen. forschen. heilen. forschung stärken, versorgung verbessern. medizininformatik. `https://www.medizininformatik-initiative.de/de/start`, 2018-10-10. Accessed: 2025-01-13. 2

[17] Z. Qiong, H. Bill, F. Gng, Olivia C., Newton P, and Skrinak K. Reinventing a cloud-native federated learning architecture on aws. `https://aws.amazon.com/blogs/machine-learning/reinventing-a-cloud-native-federated-learning-architecture-on-aws/`, 2023-10-10. Accessed: 2025-01-16. 2

[18] Google AI. Federated learning. `https://federated.withgoogle.com/`, 2024. Accessed: 2025-01-16.

[19] Karan Chadha, Junye Chen, John Duchi, Vitaly Feldman, Hanieh Hashemi, Omid Javidbakht, Audra McMillan, , and Kunal Talwar. Differentially private heavy hitter detection using federated analytics, 2023.

[20] Mingbin Xu*, Congzheng Song*, Ye Tian, Neha Agrawal, Filip Granqvist, Rogier van Dalen, Xiao Zhang, Arturo Argueta, Shiyi Han, Yaqiao Deng, Leo Liu, Anmol Walia, and Alex Jin. Training large-vocabulary neural language model by private federated learning for resource-constrained devices. In *ICASSP*, 2023. 2

[21] F. N. Wirth, T. Meurers, M. Johns, and F. Prasser. Privacy-preserving data sharing infrastructures for medical research: systematization and comparison. *BMC Med Inform Decis Mak*, 21(1):242, Aug 2021. 2

[22] Alessandro Falcetta and Manuel Roveri. Privacy-preserving deep learning with homomorphic encryption: An introduction. *IEEE Computational Intelligence Magazine*, 17(3):14–25, 2022. 2

[23] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, May 2020. 3, 31

[24] Naveen Goud. Cloud security breach leads to a leak of 957,000 patient records. https://www.cybersecurity-insiders.com/cloud-security-breach-leads-to-a-leak-of-957000-patient-records, 2019. Accessed: 2025-01-16. 3

[25] Pam Baker. Epic hipaa fail: Over 1 billion medical records exposed online. https://www.channelfutures.com/mssp-insider/epic-hipaa-fail-over-1-billion-medical-records-exposed-online, 2020. Accessed: 2024-02-08. 3

[26] Madhura Joshi, Ankit Pal, and Malaikannan Sankarasubbu. Federated learning for healthcare domain - pipeline, applications and challenges. *ACM Trans. Comput. Healthcare*, 3(4), nov 2022. 4

[27] David M. W. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation, 2020. 4

[28] T. Chai and R. R. Draxler. Root mean square error (rmse) or mean absolute error (mae)? – arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 7(3):1247–1250, 2014. 4

[29] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ROC Analysis in Pattern Recognition. 5

[30] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 233–240, New York, NY, USA, 2006. Association for Computing Machinery. 5

[31] Marius de Arruda Botelho Herr, Michael Graf, Peter Placzek, Florian König, Felix Bötte, Tyra Stickel, David Hieber, Lukas Zimmermann, Christopher Mohr, Stephanie Biergans, Mete Akgün, Nico Pfeifer, and Oliver Kohlbacher. Bringing the algorithms to the data – secure distributed medical analytics using the personal health train (pht-medic), 2022. 7, 43, 70, 81, 95

[32] Sascha Welten, Marius de Arruda Botelho Herr, Lars Hempel, David Hieber, Peter Placzek, Michael Graf, Sven Weber, Laurenz Neumann, Maximilian Jugl, Liam Tirpitz, Karl Kindermann, Sandra Geisler, Luiz Olavo Bonino da Silva Santos, Stefan Decker, Nico Pfeifer, Oliver Kohlbacher, and Toralf Kirsten. A study on interoperability between two personal health train infrastructures in leukodystrophy data analysis. *Scientific Data*, 11(1):663, Jun 2024. 7, 43, 81, 95, 101

[33] Marius de Arruda Botelho Herr, Cem Ata Baykara, Ali Burak Ünal, Nico Pfeifer, and Mete Akgün. Privacy-preserving auc computation in distributed machine learning with pht-medic. *medRxiv*, 2025. 7, 43, 81, 95, 101

[34] Sebastian C. Semler, Frank Wissing, and Ralf Heyder. German medical informatics initiative. *Methods of information in medicine*, 57:e50–e56, 07 2018. 8

[35] Fabian Prasser, Oliver Kohlbacher, Ulrich Mansmann, Bernhard Bauer, and Klaus A. Kuhn. Data integration for future medicine (difuture). *Methods Inf Med*, 57(S 01):e57–e65, 2018. e57. 8, 14, 69

[36] Bundesministerium für Bildung und Forschung. Mii_num - medizininformatik-struktur. https://www.gesundheitsforschung-bmbf.de/de/mii-num-medizininformatik-struktur-16912.php, 2023. Accessed: 2025-01-16. 8

[37] Task Force Data Sharing. Arbeitsgruppe data sharing. https://www.medizininformatik-initiative.de/de/zusammenarbeit/arbeitsgruppe-data-sharing, 2018. Accessed: 2025-01-16. 8

[38] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J. G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A. C. 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3:160018 EP –, Mar 2016. Comment. 8, 43

[39] Ananya Choudhury, Johan van Soest, and Andre Dekker. Personal health train on fhir: A privacy preserving federated approach for analyzing fair data in healthcare. *Communications in Computer and Information Science*, 1240, 2020. 8, 44

[40] Zhenwei Shi, Ivan Zhovannik, Alberto Traverso, Frank J. W. M. Dankers, Timo M. Deist, Petros Kalendralis, René Monshouwer, Johan Bussink, Rianne Fijten, Hugo J. W. L. Aerts, Andre Dekker, and Leonard Wee. Distributed radiomics as a signature validation study using the personal health train infrastructure. *Scientific Data*, 6(1):218, 2019.

[41] Timo M. Deist, Frank J.W.M. Dankers, Priyanka Ojha, M. Scott Marshall, Tomas Janssen, Corinne Faivre-Finn, Carlotta Masciocchi, Vincenzo Valentini, Jiazhou Wang, Jiayan Chen, Zhen Zhang, Emiliano Spezi, Mick Button, Joost Jan Nuyttens, René Vernhout, Johan van Soest, Arthur Jochems, René Monshouwer, Johan Bussink, Gareth Price, Philippe Lambin, and Andre Dekker. Distributed learning on 20 000+ lung cancer patients – the personal health train. *Radiotherapy and Oncology*, 144:189 – 200, 2020.

[42] Oya Beyan, Ananya Choudhury, Johan van Soest, Oliver Kohlbacher, Lukas Zimmermann, Holger Stenzhorn, Md. Rezaul Karim, Michel Dumontier, Stefan Decker, Luiz Olavo Bonino da Silva Santos, and Andre Dekker. Distributed analytics on sensitive medical data: The personal health train. *Data Intelligence*, 2(1-2):96–107, 2020. 8, 44

[43] Federal Office for Information Security. ehealth - cyber security in healthcare. https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/E-Health/e-health.html, 2022. Accessed: 2025-01-16. 9

[44] Christian Runte and Dr. Michael Biendl. Data protection and cybersecurity laws in germany. https://cms.law/en/int/expert-guides/cms-expert-guide-to-data-protection-and-cyber-security-laws/germany?format=pdf&v=5, 2021. Accessed: 2025-01-16. 9

[45] Task Force Consent. Arbeitsgruppe consent. https://www.medizininformatik-initiative.de/de/zusammenarbeit/arbeitsgruppe-consent, 2018. Accessed: 2025-01-16. 10

[46] Federal Office for Research and Education. Konsultation zum forschungsdatengesetz gestartet. https://www.bmbf.de/bmbf/shareddocs/kurzmeldungen/de/2023/03/230306-forschungsdatengesetz.html, 2023. Accessed: 2024-01-27. 10

[47] Georg von Krogh, Sebastian Spaeth, and Karim R Lakhani. Community, joining, and specialization in open source software innovation: a case study. *Research Policy*, 32(7):1217–1241, 2003. Open Source Software Development. 10

[48] Proposal for a regulation of the european parliament and of the council on horizontal cybersecurity requirements for products with digital elements and amending regulation (eu) 2019/1020, 2022. 10

[49] Matthias Löbe, Danny Ammon, Andreas Bietenbeck, Martin Boeker, Karoline Buckow, Naomi Deppenwiese, Thomas Ganslandt, Birger Haarbrandt, Ulrich Sax, Josef Schepers, Holger Stenzhorn, Sylvia Thun, and Alexander Zautke. Geschäftsordnung für die weiterentwicklung des mii-kerndatensatzes. Accessed: 2025-01-16. 12

[50] HAPI FHIR. Hapi fhir. https://github.com/hapifhir/hapi-fhir, 2024. Accessed: 2025-01-16. 12

[51] Arpita Patra and Ajith Suresh. Blaze: Blazing fast privacy-preserving machine learning, 01 2020. 12

[52] LinuxForHealth. Linuxforhealth fhir server. https://github.com/LinuxForHealth/FHIR, 2018. Accessed: 2025-01-16. 13

[53] Michael Graf. Fhir-kindling. https://github.com/migraf/fhir-kindling. Accessed: 2025-01-16. 13, 50

[54] Felix Menzel, Dagmar Waltemath, and Ron Henkel. Exploring new possibilities for research data exploration using the example of the german core data set. *Stud Health Technol Inform*, 302:749–750, May 2023. 13

[55] Dutch Techcentre for Life Sciences. Personal health train. https://www.dtls.nl/fair-data/personal-health-train/, 2018. Accessed: 2025-01-16. 13

[56] Andre Dekker, Oliver Kohlbacher, Oya Beyan, Peter-Bram 't Hoen, Stefan Decker, Kevin Sayers, Thierry Sengstag, Martin Ingvar, Ronald Cornet, and Inga Tharun'. Personal health train manifesto. https://www.go-fair.org/wp-content/uploads/2019/05/Personal-Health-Train-Implementation-Network-Manifesto.pdf, 2018. Accessed: 2025-01-16. 14

[57] Oya Beyan. Personal health train in german chapter workshop report. https://www.go-fair.org/personal-health-train-in-german-chapter-workshop-report/, 2018. Accessed: 2025-01-16. 14

[58] Anna-Lena Lamprecht, Leyla Garcia, Mateusz Kuzak, Carlos Martinez, Ricardo Arcila, Eva Martin del Pico, Victoria Dominguez Del Angel, Stephanie van de Sandt, Jon C. Ison, Paula Andrea Martínez, Peter McQuilton, Alfonso Valencia, Jennifer L. Harrow, Fotis Psomopoulos, Josep Ll Gelpi, Neil Philippe Chue Hong, Carole A. Goble, and Salvador Capella-Gutiérrez. Towards fair principles for research software. *Data Sci.*, 3:37–59, 2020. 15, 77

[59] European Commission, Content Directorate-General for Communications Networks, Technology, E Barcevičius, D Caturianas, A Leming, and G Skardžiūtė. *Multi-homing – Obstacles, opportunities, facilitating factors – Analytical paper 7*. Publications Office, 2021. 15, 97

[60] Tim Benson and Grahame Grieve. *Why Interoperability Is Hard*, pages 21–40. Springer International Publishing, Cham, 2021. 16, 17, 77, 99

[61] Frank Cremer, Barry Sheehan, Michael Fortmann, Arash N Kia, Martin Mullins, Finbarr Murphy, and Stefan Materne. Cyber risk and cybersecurity: a systematic review of data availability. *Geneva Pap. Risk Insur. Issues Pract.*, 47(3):698–736, February 2022. 18

[62] Ponemon Institute. Cost of a data breach report 2024. *Ponemon Institute*, 2021. Available at https://www.ibm.com/security/data-breach Accessed: 2025-01-16. 19

[63] Josephine Wolff. Why are ransomware attacks targeting health care providers? Technical report, TuftsNow, 2024. Available at https://now.tufts.edu/2024/04/09/why-are-ransomware-attacks-targeting-health-care-providers. 19

[64] IBM Security. Cost of a data breach report 2023. *IBM Security*, 2023. Available at https://www.ibm.com/security/data-breach Accessed: 2025-01-16. 19

[65] Sri Sunarti, Ferry Fadzlul Rahman, Muhammad Naufal, Muhammad Risky, Kresna Febriyanto, and Rusni Masnina. Artificial intelligence in healthcare: opportunities and risk for future. *Gaceta Sanitaria*, 35:S67–S70, 2021. The 1st International Conference on Safety and Public Health. 19

[66] Samuel G Finlayson, John D Bowers, Joe Ito, Jonathan L Zittrain, Andrew L Beam, and Isaac S Kohane. Adversarial attacks on medical machine learning. In *Science*, volume 363, pages 1287–1289. American Association for the Advancement of Science, 2019. 19

[67] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015. 19

[68] AEPD. Anonymisation - european data protection supervisor. `https://www.edps.europa.eu/system/files/2021-04/21-04-27_aepd-edps_anonymisation_en_5.pdf`, 2021. Accessed: 2025-01-16. 20

[69] Ana Ferreira, Francisco Bischoff, Rute Almeida, Luís Nogueira-Silva, Ricardo Cruz-Correia, and Joana Muchagata. How anonymous are your anonymized data? the anymapp case study. In Constantine Stephanidis, Margherita Antona, Stavroula Ntoa, and Gavriel Salvendy, editors, *HCI International 2023 – Late Breaking Posters*, pages 456–463, Cham, 2024. Springer Nature Switzerland. 20

[70] William Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall Press, USA, 6th edition, 2013. 22

[71] Yi-Fan Tseng, Chun-I Fan, Ting-Chuan Kung, Jheng-Jia Huang, and Hsin-Nan Kuo. Homomorphic encryption supporting logical operations. In *Proceedings of the 2017 International Conference on Telecommunications and Communication Engineering*, ICTCE '17, page 66–69, New York, NY, USA, 2017. Association for Computing Machinery. 27

[72] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 223–238, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. 27

[73] Emmanuel Bresson, Dario Catalano, and David Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 37–54. Springer, 2003. 28, 68

[74] Matthew Pirretti, Patrick Traynor, Patrick McDaniel, and Brent Waters. Secure attribute-based systems. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 99–112, 2006. 28

[75] Liam Morris. Analysis of partially and fully homomorphic encryption. 2013. 29

[76] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006. 30

[77] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc^0. *SIAM Journal on Computing*, 36(4):845–888, 2006. 30

[78] Ali Burak Ünal, Mete Akgün, and Nico Pfeifer. A framework with randomized encoding for a fast privacy preserving calculation of non-linear kernels for machine learning applications in precision medicine. In *Cryptology and Network Security: 18th International Conference, CANS 2019, Fuzhou, China, October 25–27, 2019, Proceedings*, page 493–511, Berlin, Heidelberg, 2019. Springer-Verlag. 30

[79] Ali Burak Ünal, Mete Akgün, and Nico Pfeifer. Escaped: Efficient secure and private dot product framework for kernel-based machine learning algorithms with applications in healthcare, 2020. 30

[80] Benny Applebaum. Garbled circuits as randomized encodings of functions: a primer. In *Tutorials on the Foundations of Cryptography*, pages 1–44. Springer, 2017. 30, 68

[81] Felix Nikolaus Wirth, Thierry Meurers, Marco Johns, and Fabian Prasser. Privacy-preserving data sharing infrastructures for medical research: systematization and comparison. *BMC Med. Inform. Decis. Mak.*, 21(1):242, August 2021. 31

[82] Xiaoqi Li, Peng Jiang, Ting Chen, Xiapu Luo, and Qiaoyan Wen. A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107:841–853, 2020. 32

[83] Yongbeom Kim and Edward A Stohr. Software reuse: Survey and research directions. *Journal of Management Information Systems*, 14(4):113–147, 1998. 32

[84] Todd L Veldhuizen. Software libraries and their reuse: Entropy, kolmogorov complexity, and zipf's law. *arXiv preprint cs/0508023*, 2005.

[85] Lars Heinemann. *Effective and efficient reuse with software libraries*. PhD thesis, Technische Universität München, 2012. 32

[86] Alessandro Pasetti. *Software frameworks and embedded control systems*. Springer, 2002. 32

[87] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 32, 86

[88] Théo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. A generic framework for privacy preserving deep learning. *CoRR*, abs/1811.04017, 2018. 32

[89] Ellen Friedman and Kostas Tzoumas. *Introduction to Apache Flink: Stream Processing for Real Time and Beyond*. O'Reilly Media, Inc., 1st edition, 2016. 33

[90] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow, 2018. 33

[91] François Chollet et al. Keras. https://keras.io, 2015. Accessed: 2025-01-16. 33

[92] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019. 33

[93] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems, 2015. 33

[94] Eric Liang, Richard Liaw, Philipp Moritz, Robert Nishihara, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning, 2018. 33

[95] Lisandro Dalcin and Yao-Lung L. Fang. mpi4py: Status update after 12 years of development. *Computing in Science & Engineering*, 23(4):47–54, 2021. 33

[96] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Xinghua Zhu, Jianzong Wang, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning, 2020. 33

[97] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A friendly federated learning research framework, 2022. 33

[98] Francesco Cremonesi, Marc Vesin, Sergen Cansiz, Yannick Bouillard, Irene Balelli, Lucia Innocenti, Santiago Silva, Samy-Safwan Ayed, Riccardo Taiello, Laetita Kameni, Richard Vidal, Fanny Orlhac, Christophe Nioche, Nathan Lapel, Bastien Houis, Romain Modzelewski, Olivier Humbert, Melek Önen, and Marco Lorenzi. Fed-biomed: Open, transparent and trusted federated learning for real-world healthcare applications, 2023. 33

[99] Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N. Ahmad Aziz, Sofia Ktena, Florian Tran, Michael Bitzer, Stephan Ossowski, Nicolas Casadei, Christian Herr, Daniel Petersheim, Uta Behrends, Fabian Kern, Tobias Fehlmann, Philipp Schommers, Clara Lehmann, Max Augustin, Jan Rybniker, Janine Altmüller, Neha Mishra, Joana P. Bernardes, Benjamin Krämer, Lorenzo Bonaguro, Jonas Schulte-Schrepping, Elena De Domenico, Christian Siever, Michael Kraut, Milind Desai, Bruno Monnet, Maria Saridaki, Charles Martin Siegel, Anna Drews, Melanie Nuesch-Germano, Heidi Theis, Jan Heyckendorf, Stefan Schreiber, Sarah Kim-Hellmuth, Paul Balfanz, Thomas Eggermann, Peter Boor, Ralf Hausmann, Hannah Kuhn, Susanne Isfort, Julia Carolin Stingl, Günther Schmalzing, Christiane K. Kuhl, Rainer Röhrig,

Gernot Marx, Stefan Uhlig, Edgar Dahl, Dirk Müller-Wieland, Michael Dreher, Nikolaus Marx, Jacob Nattermann, Dirk Skowasch, Ingo Kurth, Andreas Keller, Robert Bals, Peter Nürnberg, Olaf Rieß, Philip Rosenstiel, Mihai G. Netea, Fabian Theis, Sach Mukherjee, Michael Backes, Anna C. Aschenbrenner, Thomas Ulas, Angel Angelov, Alexander Bartholomäus, Anke Becker, Daniela Bezdan, Conny Blumert, Ezio Bonifacio, Peer Bork, Bunk Boyke, Helmut Blum, Thomas Clavel, Maria Colome-Tatche, Markus Cornberg, Inti Alberto De La Rosa Velázquez, Andreas Diefenbach, Alexander Dilthey, Nicole Fischer, Konrad Förstner, Sören Franzenburg, Julia-Stefanie Frick, Gisela Gabernet, Julien Gagneur, Tina Ganzenmueller, Marie Gauder, Janina Geißert, Alexander Goesmann, Siri Göpel, Adam Grundhoff, Hajo Grundmann, Torsten Hain, Frank Hanses, Ute Hehr, André Heimbach, Marius Hoeper, Friedemann Horn, Daniel Hübschmann, Michael Hummel, Thomas Iftner, Angelika Iftner, Thomas Illig, Stefan Janssen, Jörn Kalinowski, René Kallies, Birte Kehr, Oliver T. Keppler, Christoph Klein, Michael Knop, Oliver Kohlbacher, Karl Köhrer, Jan Korbel, Peter G. Kremsner, Denise Kühnert, Markus Landthaler, Yang Li, Kerstin U. Ludwig, Oliwia Makarewicz, Manja Marz, Alice C. McHardy, Christian Mertes, Maximilian Münchhoff, Sven Nahnsen, Markus Nöthen, Francine Ntoumi, Jörg Overmann, Silke Peter, Klaus Pfeffer, Isabell Pink, Anna R. Poetsch, Ulrike Protzer, Alfred Pühler, Nikolaus Rajewsky, Markus Ralser, Kristin Reiche, Stephan Ripke, Ulisses Nunes da Rocha, Antoine-Emmanuel Saliba, Leif Erik Sander, Birgit Sawitzki, Simone Scheithauer, Philipp Schiffer, Jonathan Schmid-Burgk, Wulf Schneider, Eva-Christina Schulte, Alexander Sczyrba, Mariam L. Sharaf, Yogesh Singh, Michael Sonnabend, Oliver Stegle, Jens Stoye, Janne Vehreschild, Thirumalaisamy P. Velavan, Jörg Vogel, Sonja Volland, Max von Kleist, Andreas Walker, Jörn Walter, Dagmar Wieczorek, Sylke Winkler, John Ziebuhr, Monique M. B. Breteler, Evangelos J. Giamarellos-Bourboulis, Matthijs Kox, Matthias Becker, Sorin Cheran, Michael S. Woodacre, Eng Lim Goh, Joachim L. Schultze, COVID-19 Aachen Study (COVAS), and Deutsche COVID-19 Omics Initiative (DeCOI). Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862):265–270, Jun 2021. 33

[100] David S Evans, Andrei Hagiu, and Richard Schmalensee. *Invisible engines: How software platforms drive innovation and transform industries*. The MIT Press, 2008. 33

[101] Stefan Kolb. *On the Portability of Applications in Platform as a Service*, volume 34. University of Bamberg Press, 2019. 33

[102] Peter Mell and Timothy Grance. The nist definition of cloud computing. Technical Report 800-145, National Institute of Standards and Technology (NIST), Gaithersburg, MD, September 2011. 33

[103] Sascha Welten, Yongli Mou, Laurenz Neumann, Mehrshad Jaberansary, Yeliz Yediel Ucer, Toralf Kirsten, Stefan Decker, and Oya Beyan. A privacy-preserving distributed analytics platform for health care data. *Methods Inf Med*, 61(S 01):e1–e11, Jan 2022. 34

[104] Arturo Moncada-Torres, Frank Martin, Melle Sieswerda, Johan van Soest, and Gijs Geleijnse. Vantage6: an open source privacy preserving federated learning infrastructure for secure insight exchange. In *AMIA Annual Symposium Proceedings*, pages 870–877, 2020. 34

[105] Rebecca C. Wilson, Oliver W. Butters, Demetris Avraam, James Baker, Jonathan A. Tedds, Andrew Turner, Madeleine Murtagh, and Paul R. Burton. Datashield - new directions and dimensions. *Data Science Journal*, 16, April 2017. 34

[106] Klaus Kades, Jonas Scherer, Maximilian Zenk, Marius Kempf, and Klaus Maier-Hein. Towards real-world federated learning in medical image analysis using kaapana. In Shadi Albarqouni, Spyridon Bakas, Sophia Bano, M. Jorge Cardoso, Bishesh Khanal, Bennett Landman, Xiaoxiao Li, Chen Qin, Islem Rekik, Nicola Rieke, Holger Roth, Debdoot Sheet, and Daguang Xu, editors, *Distributed, Collaborative, and Federated Learning, and Affordable AI and Healthcare for Resource Diverse Global Health*, pages 130–140, Cham, 2022. Springer Nature Switzerland. 34

[107] Mathieu N. Galtier and Camille Marini. Substra: a framework for privacy-preserving, traceable and collaborative machine learning. *CoRR*, abs/1910.11567, 2019. 34

[108] Jean Ogier du Terrail, Armand Leopold, Clément Joly, Constance Béguier, Mathieu Andreux, Charles Maussion, Benoît Schmauch, Eric W. Tramel, Etienne Bendjebbar, Mikhail Zaslavskiy, Gilles Wainrib, Maud Milder, Julie Gervasoni, Julien Guerin, Thierry Durand, Alain Livartowski, Kelvin Moutet, Clément Gautier, Inal Djafar, Anne-Laure Moisson, Camille Marini, Mathieu Galtier, Félix Balazard, Rémy Dubois, Jeverson Moreira, Antoine Simon, Damien Drubay, Magali Lacroix-Triki, Camille Franchet, Guillaume Bataillon, and Pierre-Etienne Heudel. Federated learning for predicting histological response to neoadjuvant chemotherapy in triple-negative breast cancer. *Nature Medicine*, 29(1):135–146, Jan 2023. 34

[109] Melloddy consortium. Melloddy. `https://www.melloddy.eu/`, 2020. Accessed: 2023-10-18. 34

[110] Alberto Redolfi, Silvia De Francesco, Fulvia Palesi, Samantha Galluzzi, Cristina Muscio, Gloria Castellazzi, Pietro Tiraboschi, Giovanni Savini, Anna Nigri, Gabriella Bottini, Maria Grazia Bruzzone, Matteo Cotta Ramusino, Stefania Ferraro, Claudia A. M. Gandini Wheeler-Kingshott, Fabrizio Tagliavini, Giovanni B. Frisoni, Philippe Ryvlin, Jean-François Demonet, Ferath Kherif, Stefano F. Cappa, and Egidio D'Angelo. Medical informatics platform (mip): A pilot study across clinical italian cohorts. *Frontiers in Neurology*, 11, 2020. 34

[111] George Hripcsak, Jon D. Duke, Nigam Haresh Shah, Christian G. Reich, Vojtech Huser, Martijn J. Schuemie, Martijn J. Schuemie, Marc A. Suchard, Rae Woong Park, Ian Chi Kei Wong, Peter R. Rijnbeek, Johan van der Lei, Nicole L. Pratt, G. Niklas Norén, Yu-Chuan (Jack) Li, Paul E. Stang, David Madigan, and Patrick B. Ryan. Observational health data sciences and informatics (ohdsi): Opportunities for observational researchers. *Studies in health technology and informatics*, 216:574–8, 2015. 34

[112] OHDSI. *The Book of OHDSI: Observational Health Data Sciences and Informatics*. OHDSI, 2019. 35

[113] Mete Akgün, Ali Burak Ünal, Bekir Ergüner, Nico Pfeifer, and Oliver Kohlbacher. Identifying disease-causing mutations with privacy protection. *Bioinformatics*, 07 2020. btaa641. 39

[114] David Froelicher, Juan Ramón Troncoso-Pastoriza, João Sá Sousa, and Jean-Pierre Hubaux. Drynx: Decentralized, secure, verifiable system for statistical queries and machine learning on distributed datasets. *CoRR*, abs/1902.03785, 2019. 39

[115] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy-preserving computations. In Sushil Jajodia and Javier Lopez, editors, *Computer Security - ESORICS 2008*, pages 192–206, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 39

[116] Jean Louis Raisaro, Juan Ramón Troncoso-Pastoriza, Mickaël Misbach, João Sá Sousa, Sylvain Pradervand, Edoardo Missiaglia, Olivier Michielin, Bryan Ford, and Jean-Pierre Hubaux. Medco: Enabling secure and privacy-preserving exploration of distributed clinical and genomic data. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 16(4):1328–1341, 2019. 39

[117] David Froelicher, Patricia Egger, João Sá Sousa, Jean Louis Raisaro, Zhicong Huang, Christian Mouchet, Bryan Ford, and Jean-Pierre Hubaux. Unlynx: A decentralized system for privacy-conscious data sharing. *PoPETs*, 2017(4):232–250, 2017. 39

[118] E. Scheufele, D. Aronzon, R. Coopersmith, M. T. McDuffie, M. Kapoor, C. A. Uhrich, J. E. Avitabile, J. Liu, D. Housman, and M. B. Palchuk. transmart: An open source knowledge management and high content data analytics platform. *AMIA Jt Summits Transl Sci Proc*, 2014:96–101, 2014. 39

[119] David Froelicher, Juan R. Troncoso-Pastoriza, Jean Louis Raisaro, Michel A. Cuendet, Joao Sa Sousa, Hyunghoon Cho, Bonnie Berger, Jacques Fellay, and Jean-Pierre Hubaux. Truly privacy-preserving federated analytics for precision medicine with multiparty homomorphic encryption. *Nature Communications*, 12(1):5910, 2021. 39

[120] Lattigo v6. Online: https://github.com/tuneinsight/lattigo, August 2024. EPFL-LDS, Tune Insight SA. 39

[121] Felix Nikolaus Wirth, Tobias Kussel, Armin Müller, Kay Hamacher, and Fabian Prasser. Easysmpc: a simple but powerful no-code tool for practical secure multiparty computation. *BMC Bioinformatics*, 23(1):531, Dec 2022. 39

[122] The Linux Foundation. Harbor. https://goharbor.io. Accessed: 2025-01-16. 46

[123] HashiCorp Inc. Vault. https://www.vaultproject.io. Accessed: 2025-01-16. 46

[124] Broadcom Inc. Rabbitmq. https://www.rabbitmq.com. Accessed: 2025-01-16. 46

[125] The Apache Software Foundation. Airflow. https://airflow.apache.org. Accessed: 2025-01-16. 46, 49

[126] Docker. https://www.docker.com. Accessed: 2023-12-28. 48

[127] ROSHIT RAJAN. Slim docker images. https://labs.sogeti.com/slim-docker-images/, 2020. Accessed: 2025-01-16. 50

[128] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 51

[129] Annie Abay, Yi Zhou, Nathalie Baracaldo, Shashank Rajamoni, Ebube Chuba, and Heiko Ludwig. Mitigating bias in federated learning, 2020. 51

[130] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 1322–1333. ACM, 2015. 55

[131] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 3–18. IEEE Computer Society, 2017. 55

[132] Si Chen, Ruoxi Jia, and Guo-Jun Qi. Improved techniques for model inversion attacks. *CoRR*, abs/2010.04092, 2020. 55

[133] J. Vaidya, B. Shafiq, X. Jiang, and L. Ohno-Machado. Identifying inference attacks against healthcare data repositories. *AMIA Jt Summits Transl Sci Proc*, 2013:262–6, 2013. 55

[134] S. A. Vinterbo, A. D. Sarwate, and A. A. Boxwala. Protecting count queries in study design. *J Am Med Inform Assoc*, 19(5):750–7, 2012. 55

[135] PHT Team Tübingen. Secure addition in train library of pht tübingen. https://gitlab.com/PersonalHealthTrain/implementations/germanmii/difuture/library/train-container-library/-/blob/demo/train_lib/security/HomomorphicAddition.py, 2021. Accessed: 2025-01-16. 55

[136] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 409–437, Cham, 2017. Springer International Publishing. 55

[137] Ali Burak Ünal, Nico Pfeifer, and Mete Akgün. ppaurora: Privacy preserving area under receiver operating characteristic and precision-recall curves with secure 3-party computation, 2021. 63

[138] Jiankai Sun, Xin Yang, Yuanshun Yao, Junyuan Xie, Di Wu, and Chong Wang. Dpauc: Differentially private auc computation in federated learning, 2022. 65

[139] MII. Konsortien difuture und miracum vereinbaren umfangreiche kooperation, 2023. Accessed: 2025-01-16. 69

[140] Medical Informatics Initiativee. Use case cord-mi. https://www.medizininformatik-initiative.de/de/CORD, 2021. Accessed: 2025-01-13. 69

[141] Medical Informatics Initiativee. Use case polar_mi. https://www.medizininformatik-initiative.de/de/POLAR, 2021. Accessed: 2025-01-13. 69

[142] Leuko-expert homepage. https://leukoexpert.hs-mittweida.de. Accessed: 2023-12-28. 70

[143] Genetic and rare diseases (gard) information center. https://rarediseases.info.nih.gov/diseases/6895/leukodystrophy. Accessed: 2023-12-28. 70

[144] Medical Informatics Initiative. Mii consortia data integration centers. https://www.medizininformatik-initiative.de/en/consortia/data-integration-centres. Accessed: 2023-12-28. 70

[145] Sascha Welten, Yongli Mou, Laurenz Neumann, Mehrshad Jaberansary, Yeliz Yediel Ucer, Toralf Kirsten, Stefan Decker, and Oya Beyan. A Privacy-Preserving Distributed Analytics Platform for Health Care Data. *Methods of Information in Medicine*, January 2022. 70

[146] Paul A. Harris, Robert Taylor, Robert Thielke, Jonathon Payne, Nathaniel Gonzalez, and Jose G. Conde. Research electronic data capture (redcap)—a metadata-driven methodology and workflow process for providing translational research informatics support. *Journal of Biomedical Informatics*, 42(2):377–381, 2009. 72

[147] Paul A. Harris, Robert Taylor, Brenda L. Minor, Veida Elliott, Michelle Fernandez, Lindsay O'Neal, Laura McLeod, Giovanni Delacqua, Francesco Delacqua, Jacqueline Kirby, and Stephany N. Duda. The redcap consortium: Building an international community of software platform partners. *Journal of Biomedical Informatics*, 95:103208, 2019. 72

[148] Sebastian Köhler, Michael Gargano, Nicolas Matentzoglu, Leigh C Carmody, David Lewis-Smith, Nicole A Vasilevsky, Daniel Danis, Ganna Balagura, Gareth Baynam, Amy M Brower, Tiffany J Callahan, Christopher G Chute, Johanna L Est, Peter D Galer, Shiva Ganesan, Matthias Griese, Matthias Haimel, Julia Pazmandi, Marc Hanauer, Nomi L Harris, Michael J Hartnett, Maximilian Hastreiter, Fabian Hauck, Yongqun He, Tim Jeske, Hugh Kearney, Gerhard Kindle, Christoph Klein, Katrin Knoflach, Roland Krause, David Lagorce, Julie A McMurry, Jillian A Miller, Monica C Munoz-Torres, Rebecca L Peters, Christina K Rapp, Ana M Rath, Shahmir A Rind, Avi Z Rosenberg, Michael M Segal, Markus G Seidel, Damian Smedley, Tomer Talmy, Yarlalu Thomas, Samuel A Wiafe, Julie Xian, Zafer Yüksel, Ingo Helbig, Christopher J Mungall, Melissa A Haendel, and Peter N Robinson. The Human Phenotype Ontology in 2021. *Nucleic Acids Research*, 49(D1):D1207–D1217, 12 2020. 73

[149] C. Sue Richards, Sherri Bale, Daniel B. Bellissimo, Soma Das, Wayne W. Grody, Madhuri R. Hegde, Elaine Lyon, Brian E. Ward, and Molecular Subcommittee of the ACMG Laboratory Quality Assurance Committee. ACMG recommendations for standards for interpretation and reporting of sequence variations: Revisions 2007. *Genetics in Medicine: Official Journal of the American College of Medical Genetics*, 10(4):294–300, April 2008. 73

[150] Johan T. den Dunnen, Raymond Dalgleish, Donna R. Maglott, Reece K. Hart, Marc S. Greenblatt, Jean McGowan-Jordan, Anne-Francoise Roux, Timothy Smith, Stylianos E. Antonarakis, and Peter E.M. Taschner. HGVS Recommendations for the Description of Sequence Variants: 2016 Update. *Human Mutation*, 37(6):564–569, 2016. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/humu.22981. 73

[151] Domain names - concepts and facilities. RFC 1034, November 1987. 73

[152] Domain names - implementation and specification. RFC 1035, November 1987. 73

[153] Station registry. https://station-registry.de/. Accessed: 2023-12-28. 74

[154] Sascha Welten, Lars Hempel, Masoud Abedi, Yongli Mou, Mehrshad Jaberansary, Laurenz Neumann, Sven Weber, Kais Tahar, Yeliz Ucer Yediel, and Matthias Löbe. Multi-Institutional Breast Cancer Detection Using a Secure On-Boarding Service for Distributed Analytics. *Applied Sciences*, 12(9):4336, 2022. 74, 103

[155] Wilhelm Hasselbring, Leslie Carr, Simon Hettrick, Heather Packer, and Thanassis Tiropanis. From fair research data toward fair and open research software. *it - Information Technology*, 62(1):39–47, 2020. 74

[156] Christian Stohrer and Thomas Lugrin. *Asymmetric Encryption*, pages 11–14. Springer Nature Switzerland, Cham, 2023. 77

[157] Luiz Olavo Bonino da Silva Santos, Luís Ferreira Pires, Virginia Graciano Martinez, João Luiz Rebelo Moreira, and Renata Silva Souza Guizzardi. Personal Health Train Architecture with Dynamic Cloud Staging. *SN computer science*, 4(1):14, 2023. 77, 78, 99

[158] Erik Thorsby. The human major histocompatibility system. *Immunological Reviews*, 18(1):51–129, 1974. 82

[159] Jenefer M. Blackwell, Sarra E. Jamieson, and David Burgner. Hla and infectious diseases. *Clinical microbiology reviews*, 22(2):370–385, 2009. 82

[160] The German Human Genome-Phenome Archive. The german human genome-phenome archive. https://ghga.dkfz.de/, 2021. Accessed: 2025-01-16. 82

[161] Philip A. Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm, Maxime Ulysse Garcia, Paolo Di Tommaso, and Sven Nahnsen. The nf-core framework for community-curated bioinformatics pipelines. *Nature Biotechnology*, 2020. 82

[162] G. R. Abecasis, A. Auton, L. D. Brooks, M. A. DePristo, R. M. Durbin, R. E. Handsaker, H. M. Kang, G. T. Marth, and G. A. McVean. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491(7422):56–65, 2012. 82

[163] Sven Fillinger Christopher Mohr, Alexander Peltzer. nf-core/hlatyping. https://github.com/nf-core/hlatyping, 2021. Accessed: 2025-01-16. 82

[164] Tschandl P., Rosendahl C., and Kittler H. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci. Data*, 5:180161, 2018. 84

[165] Noel C.F. Codella, M.Emre Celebi David Gutman, Brian Helba, Michael A. Marchetti, Stephen W. Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection. In *A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)"*, 2017.

[166] Marc Combalia, Noel C.F. Codella, Veronica Rotemberg, Brian Helba, Veronica Vilaplana, Ofer Reiter, Allan C. Halpern, Susana Puig, and Josep Malvehy. Bcn20000: Dermoscopic lesions in the wild". 84

[167] Nils Gessert, Maximilian Nielsen, Mohsin Shaikh, René Werner, and Alexander Schlaefer. Skin lesion classification using ensembles of multi-resolution efficientnets with meta data. *MethodsX*, 7:100864, 2020. 84

[168] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013. 85

[169] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[170] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 85

[171] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 86

[172] Los Alamos National Laboratory. Los alamos HIV database. `http://www.hiv.lanl.gov/`, 2023. Accessed: 2025-01-16. 86

[173] LATANYA SWEENEY. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002. 91

[174] K. M. Boycott, M. R. Vanstone, D. E. Bulman, and A. E. MacKenzie. Rare-disease genetics in the era of next-generation sequencing: Discovery to translation. *Nature Reviews Genetics*, 14(10):681–691, 2013. 97

[175] M. P. McBee, O. Awan, A. T. Colucci, P. Bindal, J. Leitch, K. Koehler, and A. P. Kansagra. Deep learning in radiology. *Academic Radiology*, 25(11):1472–1480, 2018. 97

[176] T. Desautels, J. Calvert, J. Hoffman, M. Jay, Y. Kerem, L. Shieh, D. Shimabukuro, U. Chettipally, M. D. Feldman, C. Barton, D. J. Wales, and R. Das. Prediction of sepsis in the intensive care unit with minimal electronic health record data: A machine learning approach. *JMIR Medical Informatics*, 4(3):e28, 2016. 97

[177] F. S. Collins and H. Varmus. A new initiative on precision medicine. *New England Journal of Medicine*, 372(9):793–795, 2015. 97

[178] N. P. Tatonetti, P. P. Ye, R. Daneshjou, and R. B. Altman. Data-driven prediction of drug effects and interactions. *Science Translational Medicine*, 4(125):125ra31, 2012. 97

[179] Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. Decentralized Identifiers (DIDs) v1.0. *W3C Recommendation*, 2022. 99

[180] Privateaim. https://www.medizininformatik-initiative.de/de/privateaim-sichere-verteilte-auswertung-medizinischer-daten. Accessed: 2023-12-28. 99

[181] Privateaim - sichere verteilte auswertung medizinischer daten. https://www.medizininformatik-initiative.de/de/privateaim-sichere-verteilte-auswertung-medizinischer-daten. Accessed: 2024-02-08. 103

Appendix A

Abbreviations

Notations and abbreviations used in this dissertation are listed alphabetically below. Abbreviations in cursive are notations from security concepts.

Numbers	
3D	Three-dimensional

A	
A	Algorithm files defined by the User
AI	Artificial intelligence
AK	Actinic keratosis
AUC	Area Under the Curve

B	
BCC	Basal cell carcinoma
BKL	Benign keratosis (solar lentigo)

C	
C_I	Container created of image I
CDS	Core Data Set
CR	Container Registry
CRA	Cyber Resilience Act

D	
D_i	Data (A,Q, and Results) of party i as cargo of the train
deNBI	German Network for Bioinformatics Infrastructure
DIC	Data Integration Center
DF	Dermatofibroma

DL	Distributed Learning
DNN	Deep Neural Network
DS_i	Digital Signature of party i

E

$\mathscr{E}_D$	Encrypted value of data D

F

FAIR	Findable Accessible Interoperable Reusable
FDG	Research Data Act
FHIR	Fast Healthcare Interoperability Resources

G

GB	Gigabytes
GDPR	General Data Protection Regulation
GPU	Graphical Processing Unit

H

h	hours
HLA	Human Leukocyte Antigen

I

I	Base image
IN	Implementation Network
ID_i	ID of party i
ID_U	Identifier of user U

K

K	Random number of length l as session key of the analysis
KB	Kilobytes

M

m	minutes
MEL	Melanoma
MHC	Major Histocompatibility Complex
MII	Medical Informatics Initiative
ML	Machine Learning
MRT	Magnetic Resonance Imaging

N

N	Random number of length l as session ID of the analysis
NV	Melanocytic nevus

P

PC	Personal Computer
PDR	Private Docker Registry
PHT	Personal Health Train
PK_i	Public key of the party i
PoC	Proof of Concept

Q

Q	Query operated on database defined by the User

R

R	Defined Route of the train defined by the User
RE	Result extraction process

S

S	Station
SCC	Squamous cell carcinoma
SK_i	Private key of the party i
SMPC	Secure Multi-Party Computation

T

TB	Train building process

U

U	User
UC	Use Case
UI	central User Interface do manage trains and submit algorithms
URI	Uniform Resource Identifier

V

VASC	Vascular lesion
vCPU	virtual Central Processing Unit

Appendix B

Contributions and Licensing

All ideas, approaches and results presented in this work were developed and discussed with my supervisors and co-workers. Co-authors also contributed to the different projects:

- Dr. Mete Akgün (M.A.)
- Cem Ata Baykara (C.A.)
- Atanas Aleksandrov (A.A.)
- Dr. Stephanie Biergans (S.B.)
- Prof. Dr. Harald Binder (H.B.)
- Markus Bujotzek (M.B.)
- Dr. Ali Burak Ünal (A.B.)
- Felix Bötte (F.B.)
- Prof. Dr. Stefan Decker (S.D.)
- Prof. Dr. Christoph Dietrich (C.D.)
- Prof. Dr. Ralf Floca (R.F.)
- Michael Graf (M.G.)
- Dr. Sandra Geilser (S.G.)
- David Hieber (D.H.)
- Lars Hempel (L.H.)
- Marius de Arruda Botelho Herr (M.H.)
- Maximilian Jugl (M.J.)
- Florian König (F.K.)
- Karl Kindermann (K.K.)
- Prof. Dr. Oliver Kohlbacher (O.K.)
- Prof. Dr. Toralf Kirsten (T.K.)
- Dr. Christopher Mohr (C.M.)
- Laurenz Neumann (L.N.)
- Peter Placzek (P.P.)
- Prof. Dr. Fabian Prasser (F.P)
- Prof. Dr. Nico Pfeifer (N.P.)
- Dr. Luiz Olavo Bonino da Silva Santos (L.O.B.S.S)
- Patric Tippmann (P.T.)
- Liam Tirpitz (L.T.)
- Tyra Stickel (T.S.)
- Sven Weber (S.W.)
- Sascha Welten (S.W.)
- Lukas Zimmermann (L.Z.)

Federated learning and analysis solutions for biomedical data science: A scoping review

The manuscript in preparation had the following contribution. P.T., M.H., A.A., M.B. Acquised and analyzed or interpreted the data and wrote the initial draft of the paper. T.K., R.F., C.D., O.K., N.P., F.P., and H.B. supervised the study. All authors discussed and commented on the manuscript, which will be available under the CC BY 4.0 license.

Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC

The published manuscript had the following contribution. M.H. contributed to software development, the study, and wrote the paper. C.A. and A.B. revised the manuscript. N.P. and M.A supervised the study. All authors discussed and commented on the manuscript. The manuscript is available under the CC BY-NC-ND 4.0 license.

A Study on Interoperability between Two Personal Health Train Infrastructures in Leukodystrophy Data Analysis

The published manuscript had the following contribution. S.W. and M.H. contributed to software development, the study, and wrote the paper. L.H. and D.H. carried out experiments. P.P., M.G., D.H., S.W., L.N., and M.J. contributed to software development. S.D., N.P., O.K and T.K. designed the study. L.T., K.K., S.G., and L.O.B.S.S reviewed the manuscript. All authors discussed and commented on the manuscript. The manuscript is available under the CC BY 4.0 license.

Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train (PHT-meDIC)

The published manuscript had the following contribution. Conceptualization, M.H, N.P., M.A., and O.K.; Software, M.H., M.G., P.P., F.K., T.S., F.B., L.Z., and D.H.; Investigation: M.H., D.H., and C.M., Writing - Original Draft, M.H.; Writing - Review and Editing, M.H., D.H., M.A., N.P, S.B., O.K. Supervision, S.B., M.A., N.P. and O.K. All authors discussed and commented on the manuscript. The manuscript is available under the CC BY 4.0 license.

Appendix C

Presentations and Posters

2024

In January **Marius de Arruda Botelho Herr** presented '**From PHT analysis trains towards FLAME** ' at the IMI Forschungsseminar in Heidelberg. The presentation is available on Google presentation for more details.

2023

Marius de Arruda Botelho Herr delivered a PHT-meDIC presentation titled '**Secure analysis of large-scale, distributed data using the PHT-meDIC**' in March at the AMIA 2023 Informatics Summit in Seattle, USA. The presentation is available on Google presentation for more details.

In May, **Marius de Arruda Botelho Herr** received the *Best Presentation Award* for his presentation '**Secure Distributed Medical Analytics for German Healthcare Institutions using the Personal Health Train (PHT-meDIC)**' at the 1st Spring Symposium for Medical Informatics in Heidelberg. The presentation is available on Google presentation for more details.

2022

In June, **Marius de Arruda Botelho Herr** presented a PHT-meDIC presentation followed by an interactive workshop with David Hieber at the MIRACUM & DIFUTURE Summer School 2022. The presentation slides are available in German and can be found on Google presentation."

In August, **Marius de Arruda Botelho Herr** delivered a presentation titled '**PHT-meDIC Overview**' (in German) at the MIRACUM DIFUTURE Kolloquium 2022. You can view the presentation on YouTube.

2021

In July, **Marius de Arruda Botelho Herr** presented a PHT-meDIC presentation and poster titled '**Bringing the Algorithms to the Data**' during the TransMed track at ISMB/ECCB 2021. This presentation received the *Best Presentation Award*, and the materials are available on YouTube.

In November, **Marius de Arruda Botelho Herr** delivered a presentation titled '**PHT-meDIC im Kontext von CORD: Eine Demonstration der Analyse Funktionalität** ' on the PHT-meDIC CORD demo at the 16th CORD-MII-WebWorkshop. The presentation was conducted in German, and you can access the slides on Google presentation.

2020

In February, represented by **Marius de Arruda Botelho Herr** , showcased a poster at the PEMED conference held in Munich. The poster's title was '**Bringing the Algorithms to the Data - Secure Distributed Medical Analytics using the Personal Health Train**'.

In October, **Marius de Arruda Botelho Herr** , Sascha Welten, and Oya Beyan presented a collaborative status update with Aachen, featuring the PHT-meDIC architecture from Tübingen. The presentation was titled '**The Personal Health Train Network Germany**' included a demo from Aachen and discussions about the next steps. This event took place at the MII Workshop for Distributed Analysis in Germany and details are available on Google presentation.

2019

In May, **Marius de Arruda Botelho Herr** delivered a poster presentation at the PhD Day in Tübingen, focusing on the Personal Health Train (PHT). The presentation was titled '**Bringing the Algorithms to the Data - Distributed Medical Analytics using the Personal Health Train**'.

In July, represented by **Marius de Arruda Botelho Herr** , showcased a poster at the ISMB conference held in Basel, Switzerland. The poster's title was '**Bringing the Algorithms to the Data - Distributed Medical Analytics using the Personal Health Train Paradigm**'.

In October, **Marius de Arruda Botelho Herr** delivered a presentation providing a security concept release and a status update on recent progress in Tübingen. This presentation took place at the DIFUTURE Symposium 2019 in Tübingen.

Appendix D

Publications

Manuscripts in preparation

Tippmann P.*, **de Arruda Botelho Herr M.***, Aleksandrov A.*, Bujotzek M., Kirsten T., Floca R., Dietrich C., Kohlbacher O., Pfeifer N., Prasser F., Binder H. Federated learning and analysis solutions for biomedical data science: A scoping review (* contributed equally)

Submitted manuscripts

de Arruda Botelho Herr M., Ata Baykara C., Burak Ünal A., Pfeifer N., Akgün M. Privacy-preserving AUC Computation in Distributed Machine Learning with PHT-meDIC *PLOS Digital Medicine*

2024

Welten S.*, **de Arruda Botelho Herr M.***, Hempel L., Hieber D., Placzek P., Graf M., Weber S., Neumann L., Jugl L. and Liam Tirpitz L. and Kindermann K., Geisler S., Olavo Bonino da Silva Santos L., Decker S., Pfeifer N., Kohlbacher O., Kirsten T. A study on interoperability between two Personal Health Train infrastructures in leukodystrophy data analysis *Scintific Data* **11**, 663 (* contributed equally)

2022

Häger L., Wendland P., Biergans S., Lederer S., **de Arruda Botelho Herr M.**, Erhardt C., Schmauder K., Kschischo M., Malek N.P., Bunk S., et al. External Validation of COVID-19 Risk Scores during Three Waves of Pandemic in a German Cohort—A Retrospective Study *Journal of Personalized Medicine* **12**, (11):1775

2018

Röder B.*, Kersten N.*, **de Arruda Botelho Herr M.***, Speicher N.K., Pfeifer N. "web-rMKL: a web server for dimensionality reduction and sample clustering of multi-view data based on unsupervised multiple kernel learning" *Nucleic Acids Research* **47**, W605–W609. (* contributed equally)

Appendix E

Supporting Tables

Detailed tables summarizing the technical classification, analytical capabilities, data compatibility, security, accessibility, interoperability, and developer support of the federated learning solutions are provided here in the following tables.

Table E.1: Analytical capabilities of federated solutions.

Name	Data Queries	Statistical Methods	Deep Learning	Customizability
Federated Libraries				
TensorFlow Federated	✗	✓	✓	✓
FedML	✗	✗	✓	✓
Federated Frameworks				
Fed-Biomed	✗	✗	✓	✓
Flower	✗	✗	✓	✓
Swarm Learning	✗	✗	✓	✓
Federated Platforms				
DataSHIELD	✓	✓	✓	✓
PHT-meDIC	✓	✓	✓	✓
PADME	✓	✓	✓	✓
Vantage6	✓	✓	✓	✓
Substra	✓	✓	✓	✓
OHDSI (ATLAS)	✓	✓	✗	✓
EasySMPC	✓	✗	✗	✗
OpenFL	✗	✗	✓	✓
Platform as a Service (PaaS)				
Medical Informatics Platform	✓	✓	✗	✗
FeatureCloud	✓	✓	✓	✓
sfkit	✗	✓	✗	✗

Legend

- **Data Queries**: Ability to perform data queries across federated datasets.
- **Statistical Methods**: Support for statistical analysis methods.
- **Deep Learning**: Capability to perform deep learning tasks.
- **Customizability**: Ability to customize or extend the solution.
- **Symbols**: ✓ indicates support/presence; ✗ indicates lack of support/absence.

Table E.2: Data type compatibility of federated solutions.

Name	Clinical Data Formats	Omics Data Formats	Imaging Data Formats
Federated Libraries			
TensorFlow Federated	✓	✓	✓
FedML	✓	✗	✓
Federated Frameworks			
Fed-Biomed	✓	✗	✓
Flower	✓	✓	✓
Swarm Learning	✓	✓	✓
Federated Platforms			
DataSHIELD	✓	✓	✗
PHT-meDIC	✓	✓	✓
PADME	✓	✓	✓
Vantage6	✓	✓	✓
Substra	✓	✓	✓
OHDSI	✓	✓	✗
EasySMPC	✓	✗	✗
OpenFL	✓	✓	✓
Platform as a Service (PaaS)			
Medical Informatics Platform	✓	✓	✓
FeatureCloud	✓	✓	✓
sfkit	✓	✓	✗

Legend

- **Clinical Data Formats**: Compatibility with standard formats used for clinical data, such as Electronic Health Records (EHRs).
- **Omics Data Formats**: Support for data formats associated with genomics, proteomics, and other omics fields.
- **Imaging Data Formats**: Ability to handle formats commonly used in medical imaging, such as DICOM.
- **Symbols**: ✓ indicates support; ✗ indicates lack of support.

Table E.3: Security features of federated learning solutions.

Name	Threat Model	Permissions	Id. Manag.	Proj. Manag.
Federated Libraries				
TensorFlow Federated	Secure Agg.	Basic RBAC	✗	✗
FedML	Secure Agg., DP	Basic RBAC	✗	✗
Federated Frameworks				
Fed-Biomed	Secure Agg., HE	RBAC	✓	✓
Flower	Secure Agg., DP	RBAC	✓	✗
Swarm Learning	Zero Trust Arch., Secure Agg.	Adv. Access Control	Fed. Identity Mgmt.	✓
Federated Platforms				
DataSHIELD	DP, Secure Agg.	Fine-Grained Access Control	✓	✓
PHT-meDIC	SMPC	RBAC	✓	✓
PADME	SMPC	RBAC	✓	✓
Vantage6	HE, DP	Fine-Grained Access Control	✓	✓
Substra	ZK Proofs, SMPC	Adv. RBAC	✓	✓
OHDSI	Secure Data Access Control	RBAC	✓	✓
EasySMPC	SMPC	RBAC	✗	✗
OpenFL	Secure Agg., DP	RBAC	✓	✓
Medical Informatics Platform	HE, DP	RBAC	✓	✓
Platform as a Service (PaaS)				
FeatureCloud	DP, Secure Agg.	RBAC	✓	✓
sfkit	Secure Agg.	Basic Access Control	✗	✗

Legend

Threat Model abbreviations:
Secure Agg. (Secure Aggregation), DP (Differential Privacy), HE (Homomorphic Encryption), SMPC (Secure Multiparty Computation), ZK Proofs (Zero-Knowledge Proofs), Zero Trust Arch. (Zero Trust Architecture)

Permissions abbreviations:
RBAC (Role-Based Access Control), Adv. Access Control (Advanced Access Control)

Id. Manag.: Identity Management; Fed. Identity Mgmt. (Federated Identity Management)

Proj. Manag.: Project Management

Symbols: ✓ indicates support; ✗ indicates lack of support

Table E.4: Accessibility and interoperability of federated learning solutions.

Name	Infra. Dep.	Intra-plat. Compat.	GUI
Federated Libraries			
TensorFlow Federated	Linux (L), macOS (m), Windows (W)	✗	✗
FedML	L, m, W	✓	✗
Federated Frameworks			
Fed-Biomed	L	✓	✗
Flower	L, m, W	✓	✓
Swarm Learning	L, Cloud (C)	✓	✓
Federated Platforms			
DataSHIELD	L, W, C	✓	✓
PHT-meDIC	L, Docker (D)	✓	✓
PADME	L, D	✓	✓
Vantage6	L, D	✓	✓
Substra	L, D	✓	✓
OHDSI	L, W, m, C	✓	✓
EasySMPC	L, W, m	✓	✗
OpenFL	L, W	✓	✓
Medical Informatics Platform	C, L	✓	✓
Platform as a Service (PaaS)			
FeatureCloud	C	✓	✓
sfkit	L, m, W	✓	✓

Legend

Infra. Dep.: Infrastructure Dependency; requirements for specific hardware or operating systems.
Abbreviations:
L (Linux), m (macOS), W (Windows), D (Docker), C (Cloud)

Intra-plat. Compat.: Intra-platform Compatibility; ability to operate across different components of the platform.

GUI: Graphical User Interface; availability of a GUI for ease of use.

Symbols: ✓ indicates support/presence; ✗ indicates lack of support/absence.

Table E.5: Licenses and support for federated learning solutions.

Name	License	Support	Dev. Outreach	Community
Federated Libraries				
TensorFlow Federated	A2	Active	GHI, CF	Large OSC
FedML	A2	Active	GH, CF	Active OSC
Federated Frameworks				
Fed-Biomed	A2	Active	GHI, M	Discord Community
Flower	A2	Active	GHI, M	Slack Community
Swarm Learning	Proprietary	Limited	Enterprise Support	Enterprise Customers
Federated Platforms				
DataSHIELD	GPL-3.0	Active	M, GHI	CF
PHT-meDIC	MIT	Active	GHI	Support Group
PADME	MIT	Active	GHI, Research Net.	Research Community
Vantage6	A2	Active	GH, M	CF
Substra	A2	Active	GHI, Slack	OSC
OHDSI	A2	Active	GHI, Forums	Large Global Comm.
EasySMPC	MIT	Active	GH, CF	OSC
OpenFL	A2	Active	GHI, Slack	OSC
Medical Informatics Platform	Mixed	Limited	GH, Contact Supp.	Research Net.
Platform as a Service (PaaS)				
FeatureCloud	A2	Active	GH, M	OSC
sfkit	A2	Active	GH, CF	OSC

Legend

License abbreviations:
A2 (Apache License 2.0), MIT (MIT License), GPL-3.0 (GNU General Public License v3.0)

Dev. Outreach abbreviations:
GH (GitHub), GHI (GitHub Issues), M (Mailing List), CF (Community Forum)

Support:
Active (A), Limited (L)

Community abbreviations:
OSC (Open Source Community), CF (Community Forum), Comm. (Community), Supp. (Support), Net. (Network)

Table E related to used subjects and probes in the bioinformatics experiment.

Subject	Probe	Station
NA06985	SRR709972	1
NA06994	SRR070528	1
NA06994	SRR070819	1
NA07000	SRR766039	1
NA07048	SRR099452	1
NA07056	SRR764718	1
NA07357	SRR764689	1
NA07357	SRR764690	1
NA10847	SRR070531	1
NA10847	SRR070823	1
NA10851	SRR766044	1
NA11829	SRR710128	1
NA11831	SRR709975	1
NA11832	SRR766003	1
NA11840	SRR070532	1
NA11840	SRR070809	1
NA11881	SRR766021	1
NA11992	SRR701474	1
NA11994	SRR701475	1
NA11995	SRR766010	1
NA12003	SRR766061	1
NA12004	SRR766059	1
NA12005	SRR718067	1
NA12006	SRR716422	1
NA12043	SRR716423	1
NA12043	SRR716424	1
NA12044	SRR766060	1
NA12144	SRR766058	1
NA12154	SRR702067	1
NA12155	SRR702068	1
NA12156	SRR764691	1
NA12234	SRR716435	1
NA12249	SRR070798	1
NA12716	SRR081269	1
NA12717	SRR071172	1

NA12750	SRR081238	1
NA12750	SRR794547	1
NA12750	SRR794550	1
NA12751	SRR071136	1
NA12751	SRR071139	1
NA12760	SRR081223	1
NA12760	SRR081251	1
NA12761	SRR077753	1
NA12761	SRR081267	1
NA12762	SRR718076	1
NA12763	SRR077752	1
NA12763	SRR081230	1
NA12812	SRR715913	1
NA12813	SRR718077	1
NA12813	SRR718078	1
NA12814	SRR715914	1
NA12872	SRR716647	1
NA12874	SRR764692	1
NA12878	SRR098401	1
NA12891	SRR098359	1
NA12892	ERR034529	1
NA18501	SRR100022	1
NA18502	SRR764722	1
NA18502	SRR764723	1
NA18504	SRR100028	1
NA18505	SRR716648	1
NA18505	SRR716649	1
NA18507	SRR764745	1
NA18507	SRR764746	1
NA18508	SRR716637	1
NA18508	SRR716638	1
NA18516	SRR100026	1
NA18517	ERR034551	1
NA18522	SRR107025	1
NA18523	ERR034552	1
NA18526	ERR031854	1
NA18532	ERR031956	1
NA18537	ERR032033	1

NA18537	ERR032034	1
NA18542	ERR031855	1
NA18545	ERR031856	1
NA18547	ERR031957	1
NA18550	ERR031958	1
NA18552	ERR031959	1
NA18558	ERR031960	1
NA18561	ERR031858	1
NA18562	ERR031859	1
NA18563	ERR031860	1
NA18566	ERR031862	1
NA18571	ERR031868	1
NA18572	ERR031869	2
NA18573	ERR031870	2
NA18576	ERR031871	2
NA18577	ERR032035	2
NA18577	ERR032036	2
NA18579	ERR032037	2
NA18579	ERR032038	2
NA18582	ERR031961	2
NA18592	ERR031962	2
NA18593	ERR034531	2
NA18603	ERR031872	2
NA18608	ERR031874	2
NA18609	ERR031875	2
NA18620	ERR031877	2
NA18621	ERR034595	2
NA18622	ERR032027	2
NA18622	ERR032028	2
NA18623	ERR032008	2
NA18624	ERR031928	2
NA18632	ERR031929	2
NA18633	ERR031878	2
NA18635	ERR031879	2
NA18636	ERR031930	2
NA18853	SRR100011	2
NA18858	ERR034553	2
NA18861	ERR034554	2

NA18870	SRR100031	2
NA18871	SRR100029	2
NA18912	SRR111960	2
NA18940	ERR034596	2
NA18942	ERR034597	2
NA18943	ERR034598	2
NA18944	ERR034599	2
NA18947	ERR034601	2
NA18948	ERR034602	2
NA18949	ERR034603	2
NA18951	ERR034604	2
NA18952	ERR034605	2
NA18953	SRR099546	2
NA18959	SRR099545	2
NA18960	SRR099533	2
NA18961	SRR099544	2
NA18966	SRR071175	2
NA18967	SRR071192	2
NA18967	SRR071196	2
NA18968	SRR077480	2
NA18968	SRR081231	2
NA18969	SRR081266	2
NA18969	SRR081273	2
NA18970	SRR071116	2
NA18970	SRR071127	2
NA18971	SRR077447	2
NA18972	SRR077490	2
NA18972	SRR081255	2
NA18973	SRR077861	2
NA18973	SRR078846	2
NA18974	SRR077456	2
NA18974	SRR081248	2
NA18975	SRR078849	2
NA18976	SRR077451	2
NA18976	SRR077757	2
NA18978	SRR716650	2
NA18981	SRR077477	2
NA18981	SRR077751	2

NA18987	SRR077491	2
NA18990	SRR077454	2
NA18990	SRR077486	2
NA18991	SRR077450	2
NA18991	SRR077855	2
NA18992	SRR716428	2
NA18994	SRR716431	2
NA18995	SRR764775	2
NA18997	SRR702078	2
NA18998	SRR766013	2
NA18999	SRR112297	2
NA19000	SRR099528	2
NA19003	SRR099532	2
NA19005	SRR715906	2
NA19007	SRR099549	2
NA19012	SRR112294	2
NA19092	SRR100012	2
NA19093	SRR100033	2
NA19098	SRR077460	2
NA19099	SRR748771	2
NA19099	SRR748772	2
NA19102	SRR100034	2
NA19116	SRR100021	2
NA19119	SRR077471	2
NA19119	SRR081271	2
NA19130	SRR107026	2
NA19131	SRR070494	2
NA19131	SRR070783	2
NA19137	SRR081226	2
NA19137	SRR081237	2
NA19137	SRR792542	2
NA19137	SRR792560	3
NA19138	SRR070472	3
NA19138	SRR070776	3
NA19141	SRR077433	3
NA19141	SRR077464	3
NA19143	SRR077445	3
NA19143	SRR081272	3

NA19144	SRR077392	3
NA19152	SRR071135	3
NA19152	SRR071167	3
NA19153	SRR070660	3
NA19153	SRR070846	3
NA19159	SRR070478	3
NA19159	SRR070786	3
NA19171	SRR077492	3
NA19171	SRR077493	3
NA19172	SRR111962	3
NA19200	SRR077432	3
NA19200	SRR078847	3
NA19201	SRR077439	3
NA19201	SRR077462	3
NA19204	SRR077857	3
NA19204	SRR081263	3
NA19206	SRR070491	3
NA19206	SRR070781	3
NA19207	SRR081254	3
NA19207	SRR081256	3
NA19209	SRR077489	3
NA19209	SRR077859	3
NA19210	SRR078845	3
NA19210	SRR081222	3
NA19238	SRR071173	3
NA19238	SRR071195	3
NA19238	SRR792121	3
NA19238	SRR792165	3
NA19239	SRR792097	3
NA19239	SRR792159	3
NA19240	SRR792091	3
NA19240	SRR792767	3

Table E.6: Included subjects, probes and associated station for demonstration of nf-core pipeline with PHT-meDIC experiment bioinformatics.