In case of emergency, do not break the glass!

Secure cross-organisational data sharing in acute care

Tuler de Oliveira, M.

Publication date
2023

Document Version
Final published version

Marcela Tuler de Oliveira

IN CASE OF EMERGENCY
DO NOT BREAK THE GLASS!

Secure cross-organisational data sharing in acute care

Marcela Tuler de Oliveira

IN CASE OF EMERGENCY DO NOT BREAK THE GLASS!

Secure cross-organisational data sharing in acute care

Marcela Tuler de Oliveira

IN CASE OF EMERGENCY

Secure cross-organisational data sharing in acute care

IN CASE OF EMERGENCY, DO NOT BREAK THE GLASS! SECURE CROSS-ORGANISATIONAL DATA SHARING IN ACUTE CARE

Marcela Tuler de Oliveira

In case of emergency, do not break the glass! Secure cross-organisational data sharing in acute care.

The research described in this Thesis was performed at the e-Science Research Group from the Department of Epidemiology & Data Science and the Department of Biomedical Engineering and Physics, Amsterdam Univerisity Medical Centers, University of Amsterdam, Amsterdam, The Netherlands.

This work was partly funded by the European Union's Horizon 2020 research and innovation program under grant agreement No. 826093 (ASCLEPIOS project), the AMC Medical Research BV, Amsterdam UMC, location AMC, under project No. 22604.

Financial support by the Dutch Heart Foundation for the publication of this Thesis is gratefully acknowledged.

Cover Design from Oleksandra Den
Thesis template: classicthesis by André Miede and Ivo Pletikosić.
Printed and bound by Proefschriften.nl
ISBN: 978-94-6473-024-1

In case of emergency, do not break the glass!

Secure cross-organisational data sharing in acute care

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op donderdag 23 februari 2023, te 16.00 uur

door Marcela Tuler de Oliveira
geboren te Brasil

Promotiecommissie

Promotores:	prof. dr. A.H. Zwinderman	AMC-UvA
	prof. dr. H.A. Marquering	AMC-UvA
Copromotores:	dr. S.D. Olabarriaga	AMC-UvA
Overige leden:	prof. dr. ir. P.H.A.J.M. van Gelder	TU Delft
	prof. dr. M.W.M. Jaspers	AMC-UvA
	prof. dr. ir. R.L. Lagendijk	TU Delft
	prof. dr. ing. A.H.C. van Kampen	AMC-UvA
	dr. F. Regazzoni	Universiteit van Amsterdam
	dr. L.T. van Binsbergen	Universiteit van Amsterdam
	dr. M.E.S. Sprengers	AMC-UvA

Faculteit der Geneeskunde

CONTENTS

1 INTRODUCTION

Electronic Medical Records (EMR) systems allow the electronic entry, storage, and maintenance of digital medical data. Ideally, the EMR systems contain the patients' demographics, test results, medical history, and history of prescribed medications. One of the benefits of using an EMR system is to have comprehensive patient-history records available during treatment. Another benefit is that patient data are shareable, substantially reducing unnecessary tests and optimising communication among the professionals involved in the treatment. All these benefits improve decision-making, and the quality of care [1, 2].

EMR systems are also a top target in healthcare data breaches [3, 4]. EMR are valuable for illegitimate businesses because they often contain a person's identifiable information. According to a Trustwave 2020 report [5], healthcare data may be valued at up to $250 per record in the illicit market compared to $5.40 for a payment card record, which is the second highest value paid for data. According to IBM's 2022 report [6], healthcare data breaches are increasing exponentially yearly and, for the 12^{th} year in a row. In 2022, healthcare had the highest average data breach cost of any industry, with an average total cost of $10M per data breach. Because of the potential gain of accessing EMR for malicious purposes, healthcare professionals should not underestimate this security threat and take steps to safeguard health data. It is essential for healthcare organisations to protect the EMR systems, whether by protecting them against external threats posed by hackers and cyber criminals or by securing internal threats from access abuse by internal users.

Since the Health Insurance Portability and Accountability Act of 1996 (HIPAA) in the US [7], EMR systems have been designed to secure and protect the confidentiality of protected health information. In 2018, the General Data Protection Regulation (GDPR) [8] reinforced the need for personal data protection in Europe, defining data sharing and processing conditions across multiple domains. In summary, healthcare professionals lawfully access a patient's EMR based on three states: First, when the patient has given consent, which is usually bound by the treatment relationship of the healthcare professional with the patient. Second, when it is necessary to protect the vital interest of the patient or another person. Third, when it is needed to perform a task in the public interest [9]. Therefore, any access must present a clear purpose for data processing, and the

EMR system must apply security mechanisms to guarantee that the sensitive data is only processed for a legit purpose.

Data availability is the most crucial characteristic of an EMR system. Although healthcare professionals often see security protocols as a barrier to data access, the same security protocols can be powerful tools to increase data availability. Once privacy and safety regulations are fulfilled, healthcare organisations can build a reputation and trustworthiness, therefore paving the way for cross-organisational data sharing.

1.1 BREAK-GLASS PROCEDURE

In clinical care, a delay in accessing a patient's electronic medical records (EMR) is likely to disrupt patient care [1]. For this reason, an EMR system requires mechanisms assuring that patient care is not impaired by problems caused by user identification, authentication or authorisation during access control[10]. In this thesis, we focus on the authorisation problem. 'Break the glass' refers to a procedure that enables healthcare professionals without privileges to bypass conventional authorisation and access a patient's EMR in emergencies[7]. A break-glass procedure refers to the act of breaking the glass to activate some physical emergency alarm. In EMR systems, the break-glass procedure typically involves alert and control mechanisms, such as a pop-up warning on the screen mentioning that the data being accessed is sensitive and restricted. Sometimes, the user only needs to click on the alert window to proceed, acknowledging that it is a break-glass situation. Every break-glass procedure raises a red flag in the access logs; however, the use of the procedure is widespread, which makes it very difficult to monitor and validate each of the circumstances individually.

Establishing the proper emergency access control protocols is a tricky balancing act. If it is too difficult to access the EMR, users may be tempted to employ the break-glass procedure whenever it takes too much time or effort to prove that the access is legitimate. Any healthcare professional would be, in principle, able to "break the glass" and access confidential data, even when there is no medical relationship between the professional and the patient. The widespread use of the break-glass procedure poses a serious problem, and it happens more often than generally assumed [11]. Typically, a special logs audit trail is created to monitor the usage of break-glass procedures. However, the generated logs often do not have enough information to classify the data processing as legitimate or not, and a large number of access logs break-glass access logs makes case-by-case monitoring even harder. Therefore, the EMR system should establish adequate emergency data-sharing protocols to authorise legit access and minimise the break-glass conditions.

1.2 DATA SHARING DURING ACUTE STROKE CARE CHALLENGES

A stroke is a medical condition that occurs when suddenly the blood supply is blocked for part of the brain, classified as ischemic stroke, or when a blood vessel in the brain bursts, classified as hemorrhagic stroke [12]. Researchers have shown that the sooner the treatment is given, the better the outcomes for the patient are [13]. Moreover, patient transportation at the highest priority and hospital notification before arrival have been associated with faster initiation of stroke care and better outcomes [2].

Acute stroke care usually requires the collaboration of professionals from emergency call centres, ambulance services, hospitals, and general practitioners' clinics. These professionals must evaluate the patient's condition, identify the type of stroke and severity, decide upon the treatment, transport the patient to the adequate care centre, and perform the required intervention. In the Netherlands, healthcare organisations involved in acute stroke care are independent and have different policies and systems for medical records. Thus, there is no unified EMR system that all professionals involved in acute stroke care can access during patient treatment. Therefore, there is a need for an EMR system that enables acute care professionals to share patient data throughout the emergency treatment process, despite the organisation where they work.

Cloud storage services match the needs of multiple healthcare organisations for remote and ubiquitous access to medical data. However, security and privacy concerns still hamper the wide adoption of cloud services. The main reason is that once the EMR is stored in a cloud service provider, the security of the data relies on the cloud provider [14]. This means that the cloud service provider is usually responsible for the cryptography and access control of the data. Researchers suggested protecting the EMR before storing it in the cloud, where the data is outsourced, but the healthcare organisations keep the access control to the data [15, 16]. Assuming that the data are encrypted before being stored in the cloud, access to the data and encryption key needs to be granted and revoked to all users that need to access the EMR dynamically throughout the treatment. This problem leads us to our first research question:

RQ1: How to enable secure data sharing of confidential patient data stored on untrustworthy cloud-based EMR systems during acute care?

Access to a patient's EMR during acute stroke care is legit because it is in the vital interest of the patient [9]. However, for security and privacy sake, the access permission of the professionals must be revoked immediately after they

finish their tasks in emergency treatment. In addition, revoking a user's access rights must not affect the access rights of other physicians treating the patient. Therefore, the EMR system needs an access control mechanism on top of the encrypted data and dynamically allows access to a patient's EMR. This challenge leads us to our second research question:

RQ2: How to model a dynamic and fine-grained access control mechanism to secure patient data during acute care?

1.3 CROSS-ORGANISATION DATA SHARING CHALLENGES

According to the GDPR, as joint controllers [17], healthcare organisations must agree on data sharing and sign a legally-binding document coined the data processing agreement (DPA) [18]. Unfortunately, not all parties fully understand the compelling practical and ethical justifications usually defined in the DPA about the 'how and 'why' for lawful data processing. Moreover, the auditing process to check compliance with the DPA relies on centralised log records, which usually have limited information about the data processing and fail to inform the other parties transparently [19].

To become joint controllers, healthcare organisations must define and enforce common cross-organisation access control policies to the shared EMR. A successful access control system must be dynamic and granular to support the complex nature of cross-organisational data sharing in healthcare. Access logs should be available for auditing and monitoring regulatory compliance. This leads to our third research question:

RQ3: How to facilitate data sharing across multiple organisations by providing means to define joint access control policies and enforcement mechanisms in a transparent and auditable manner?

THESIS OUTLINE

This thesis proposes new mechanisms and explores existing ones for cross-organisation data sharing during acute stroke care. We have explored cryptographic protocols and access control mechanisms to enable secure data sharing among healthcare professionals from multiple organisations during acute

stroke care. In all cases, we aim to increase data availability for healthcare professionals with legitimate reasons to access patient data. The thesis is organised into seven chapters, the introduction, five research chapters and a discussion.

Chapter 2 describes our first attempt to answer RQ1. It presents a security protocol based on Ciphertext-Policy Attribute-Based Encryption. The proposed protocol, coined Red Alert, enables access control to encrypted medical data during emergencies. It dynamically grants and revokes access to electronic medical records using attribute-based encryption.

Chapter 3 presents another solution addressing the scalability limitations of the proposal in Chapter 2. It describes a security protocol based on a combination of Dynamic index-based Symmetric Searchable Encryption and Ciphertext-Policy Attribute-Based Encryption. The proposed protocol, coined Acute Care Access Control (AC-AC), guarantees the confidentiality of the EMR by means of scalable encryption and uses attribute-based encryption to dynamically share the EMR among multiple professionals during acute stroke care.

Chapter 4 describes three contributions to answer RQ2. First, it presents a step-by-step methodology for modelling a context-aware attribute-based access control for the EMR system. Second, it introduces the Acute Care Attribute-Based Access Control (AC-ABAC) model, which resulted from the application of the proposed methodology in the context of acute stroke care. Third, it presents the AC-ABAC implementation, using contextual attributes that legitimate data access for dynamically sharing patient data with the appropriate healthcare professionals for the duration and necessity of acute care.

Chapter 5 describes a qualitative study to collect feedback about the proposed technical solutions from the prominent roles in acute care (emergency call centre, ambulance professionals, hospital professionals and general practitioners). First, we presented the prototype of a cloud-based EMR system for acute care that combines the approaches from Chapters 3 and 4. We used in-depth interviews to capture the medical professionals' perspectives on functions, the implemented design, and the usage of the prototype in a simulated acute care event.

Chapter 6 describes our solution for RQ3. It presents an Attribute-Based Access Control model based on Blockchain and Smart Contracts for cross-organisation medical records sharing. The proposed solution, coined SmartAccess, offers joint agreement among healthcare organisations over access policies to protect sharable data. SmartAccess enables distributed access control enforcement without relying on a trustworthy central service. Moreover, SmartAccess offers transparency and auditability regarding data processing.

These five research chapters contain our core contributions, and the thesis culminates with a discussion in **chapter 7**, where we reflect on the research questions and present some ideas for future work.

2

A BREAK-GLASS PROTOCOL BASED ON CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION TO ACCESS MEDICAL RECORDS IN THE CLOUD

In emergency care, fast and efficient treatment is vital. Availability of Electronic Medical Records (EMR) allows healthcare professionals to access a patient's data promptly, which facilitates the decision-making process and saves time by not repeating medical procedures. Unfortunately, the complete EMR of a patient is often not available during an emergency situation to all treatment teams. Cloud services emerge as a promising solution to this problem by allowing ubiquitous access to information. However, EMR storage and sharing through clouds raise several concerns about security and privacy. To this end, we propose a protocol through which all treatment teams involved in emergency care can securely decrypt relevant data from the patient's EMR and add new information about the patient's status. Furthermore, our protocol ensures that treatment teams will only access the patient's EMR for the period during which the patient is under their care. Finally, we present a formal security analysis of our protocol and some initial experimental results.

This Chapter is based on:

Marcela Tuler de Oliveira*, Alexandros Bakas*, Eugene Frimpong, Adrien ED Groot, Henk A Marquering, Antonis Michalas, and Sílvia D Olabarriaga. "A break-glass protocol based on ciphertext-policy attribute-based encryption to access medical records in the cloud." In: Annals of Telecommunications (2020), pp. 1–17. Springer. [20]

2.1 INTRODUCTION

Time is critical in emergency situations. In a short time frame, health professionals need to evaluate the patient's condition, decide upon the treatment, transport the patient to the adequate care centre, and perform the required intervention. The triage and diagnosis demand and generate a large amount of data, which needs to be shared between treatment teams along the whole process. The use of a single interoperable Electronic Medical Record (EMR) improves the overall quality of care [1], leading to a substantial reduction of unnecessary investigations and to an optimized communication among the healthcare professionals involved in the treatment.

The use of a cloud storage service allows practical and dynamic management of EMRs since a cloud infrastructure enables remote and ubiquitous access to data. However, one of the biggest concerns users have about cloud storage is data security. No one wants their sensitive data jeopardized. Recently, studies propose to send the EMR to a cloud service provider, where it is encrypted and stored. In this scenario, the key used for data encryption is known by the cloud provider, which does not protect the EMR against internal attacks [14]. Researchers suggest to encrypt the EMR with a secret key before storing it in the cloud [15, 16]. This means that the secret key needs to be pre-shared with all users that wish or need to access the EMR at any time throughout the treatment. Nevertheless, if a user needs to be revoked from the process of treatment, the EMR must be re-encrypted with a fresh key, and the new key must be distributed to the other legitimate users. Therefore, revocation in this scenario is not efficient.

In the case of acute stroke care, the phrase *'Time is brain'* conveys the idea that minutes can make the difference between life and death [13]. The availability of patient data is of paramount importance for the triage, diagnosis and treating centre selection. Therefore, it is necessary to provide access to patient data - even if the patient cannot consent explicitly, which is often the case in patients with acute stroke. The so-called 'break-glass' access mechanism provides emergency access to the patient's EMR in such situations. Although some studies approach the break-glass access to encrypted EMR [21–23], its revocation after an emergency is still a problem. For security and privacy sake, immediately after the emergency situation ends, the access needs to be revoked. In addition, revoking a user's access must not affect the access of the rest of the users. Therefore, our goal is to provide a solution that allows break-glass access to a patient's EMR only during an emergency situation for only authorized treatment teams.

OUR CONTRIBUTION We describe a protocol to provide access to a patient's encrypted EMR during acute stroke treatment with an additional security mechanism, which ensures authorisation only for the period when access is necessary. The protocol securely enables sharing of EMR among multiple treatment teams through a cloud platform. The proposed solution adopts the concept of Attribute-Based Encryption (ABE) associated with policies defined for emergency situations. Additionally, it adopts token authentication to grant and revoke access during the timeline of acute stroke treatment. We prove the security of our scheme by constructing a simulator that is computationally indistinguishable from the real protocol. Moreover, we also prove the resilience of our scheme against a set of attacks defined in the threat model. Finally, we prove the effectiveness and robustness of our scheme in real-world situations by implementing the core functions of the proposed protocol.

ORGANIZATION Section 2.2 discusses related works and Section 2.3 summarizes the flow of patient information during stroke emergency. Section 2.4 defines the cryptographic primitives used throughout the paper. In Section 2.5 we present the main entities that participate in our system model and in Section 2.6 we define both the problem statement and the considered threat model. In Section 2.7 we describe our protocol and in Section 2.8 we analyze its security against malicious behaviour. Section 2.9 presents the results of our experiments on the execution times of the proposed protocol's core functions. Section 2.10 discusses the results and limitations. Section 2.11 presents preliminary conclusions.

2.2 RELATED WORK

Break-glass is a term used to refer to security solutions that provide access to information in emergency situations.

In [24] the author proposed an encryption scheme for cloud storage that can be broken by anyone exactly once, in a detectable way. The motivation for break-glass is the case when the legitimate user wants to decrypt the data previously uploaded to the cloud, but she lost all her secret keys. Our work, however, focuses on healthcare emergency situations where the break-glass condition is valid to provide EMR availability to support triage, diagnosis and treatment. Very few research works have considered this requirement.

One of the earliest arguments for a break-glass concept for the healthcare case was formulated by Povey [25]. He stated that the basic approach of an optimistic security system is to assume that any emergency situation requesting data access is legitimate and should be granted. Petrisch and Bruker presented a generic break-glass model in [26] where the data subjects are allowed

to override specific access control permissions. In [27] Zhang *et al*. proposed a concrete break-glass solution based on two-factor encryption: password-based encryption and master secret key-based encryption. In [28] the authors presented 'Rampole', a model that implements access permissions in a fine-grained manner using a declarative query language to explicitly specify a break-glass decision procedure. None of the approaches described above support attribute-based access control.

In [21], the authors use attribute-based encryption (ABE) techniques to control access to patient data. This study approaches break-glass access under emergency scenarios using a unique authority to authenticate the medical staff to access the data. To revoke access, the data needs to be re-encrypted with a new key. Brucker *et al*. [22] presented an integration of fine-grained break-glass concepts into a system based on ABE. The authors present multi-levelled break-glass access control; however, the solution does not enable revoking access after it is granted. Yang *et al.* [23] proposed a solution for ABE access control in which the patient pre-shares her password with the emergency contact person. When the patient reaches an emergency the situation, the contact person utilizes the password to derive the break-glass key and to decrypt the patient's medical files. Even though [21–23] present interesting solutions for the break-glass situation, they do not provide a concrete and efficient solution for access revocation.

Back in 1999, in [29, 30], the authors approached the problem of key revocation in a dynamic group by proposing protocols for key management for multicasting. Similarly to our work, the authors were motivated about the case where a large number of people joining/leaving the authorization groups might affect the efficiency of the cryptographic scheme. Rafaeli and Hutchison presented a survey of key management for secure group communication [31]. Although the works in [29–31] present techniques to minimize the number of message transmissions required, their schemes still need to rekey the multicast authorized group after a revocation. Our approach overcomes the rekeying problem by using a Ciphertext-Policy ABE (CP-ABE) scheme and an access control token scheme to grant and revoke access dynamically without the need to re-encrypt the patient EMR. In addition, our protocol supports the involvement of multiple treatment teams, even from different institutions, which brings the solution closer to a real emergency scenario.

2.3 PATIENT DATA SHARING DURING ACUTE STROKE EMERGENCY

Acute stroke care is a complex collaboration of various parties: professionals at the emergency call centre, ambulance nurses and drivers, and medical doctors and nurses at the hospital. Currently, treatment in the acute phase of ischemic stroke consists of intravenous thrombolysis (IVT, through recombinant tissue-

type plasminogen reactivator) and/or endovascular treatment (EVT). The challenging part is that IVT is provided in almost all hospitals (primary stroke centres), but that EVT is a highly specialized treatment only provided in a few hospitals (comprehensive stroke centres). All of these parties need to share information in the acute setting while treating the patient. Furthermore, earlier research has shown that the earlier the treatment has been given the better functional outcomes are for the patient [13]. Therefore, a break-glass access mechanism to improve data availability has potential benefits.

When a patient suffers a stroke, the patient itself, a family member or the general practitioner is the first to contact the emergency call centre. During the telephone call, trained healthcare workers follow a triage system where a suspected stroke may be concluded. When an ambulance is sent to the patient, the goal is to arrive within 45 minutes. When an ambulance goes to the patient, information already collected by the emergency call centre is sent by messages and displayed in a fixed device inside the ambulance (e.g. age, gender). Once the ambulance arrives, the ambulance team examines the patient and collects more data (e.g. blood pressure, pulse, oxygen saturation, glucose). When the ambulance team suspects a stroke and decides to take the patient to the closest hospital, it contacts this hospital by phone to inform the estimated arrival time. When the ambulance team arrives at the hospital, all information they collected will be presented to the hospital team orally. After delivering the patient, they fill the collected data into an electronic form on their tablet for recording purposes, but this will be too late to turn available for the hospital team.

After the phone call from the ambulance nurse, the concerned hospital get prepared for the patient. The neurologist or resident on call, the neurology nurse, the emergency doctor and nurse, the radiologist and the radiology technician will clear the room for image exams and wait for the patient. If the patient already has a medical record in the hospital, it is evaluated. If not, a new patient identification number will be created to store the new data. Furthermore, the doctor will try to call other hospitals or the patient's general practitioner to obtain more information about the patient. If a patient is eligible for EVT, and she needs to be transferred to a comprehensive stroke centre, all collected information is shared both orally and by e-mail between the sending and receiving hospitals. In this case, the patient is transferred by a second ambulance, which also needs the available information. At last, when the patient receives the EVT, a team of medical doctors await, including the neuro-interventional radiologist, radiology technician, and anesthesiologist, that also need to know all information. After transportation, all collected data is presented to the doctors one more time, orally. Imaging data have been sent through an imaging-exchange system for the neuro-interventionist and radiologist.

Note that three or more teams are involved in the treatment, requiring access to the patient's EMR and generating new content for it. Between all those moments of consultation, data can be missed or forgotten to mention. Therefore, to improve accessibility to medical records and protect patient's privacy, it is necessary to dynamically grant and revoke access to the patient's EMR.

2.4 CRYPTOGRAPHIC PRIMITIVES

Here we define the basic cryptographic primitives used throughout the paper and define a CP-ABE scheme as following [32, 33].

The set of all binary strings of length n is denoted by $\{0,1\}^n$, and the set of all finite binary strings as $\{0,1\}^*$. Given a set V, we refer to the i^{th} element as v_i. Additionally, we use the following notations for cryptographic operations throughout the paper:

- For an arbitrary message $m \in \{0,1\}^*$, $c = \mathsf{Enc}(K, m)$ denotes a symmetric encryption of m using the secret key $K \in \{0,1\}^*$, and $m = \mathsf{Dec}(K, c) = \mathsf{Dec}(K, \mathsf{Enc}(K, m))$ is the corresponding symmetric decryption operation.
- We denote by pk/sk a public/private key pair for an IND-CCA2 secure public key encryption scheme PKE. An encryption of message m under the public key pk is denoted by $c = \mathsf{Enc}_{pk}(m)$ and the corresponding decryption operation by $m = \mathsf{Dec}_{sk}(c) = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m))$.
- $\sigma = \mathsf{Sign}_{sk}(m)$ denotes a EUF-CMA secure digital signature over a message m. The corresponding verification operation for a digital signature is denoted by $b = \mathsf{Verify}_{pk(m,\sigma)}$, where $b = 1$ if the signature is valid, and $b = 0$ otherwise.
- A one-way hash function (H) over a message m is denoted by $H_m = H(m)$.
- We denote by $r = \mathsf{RAND}(n)$ a random binary sequence of length n, where $\mathsf{RAND}(n)$ represents a random function that takes a binary length argument n as input and gives a random binary sequence of this length in return[1].

A CP-ABE scheme is a tuple of the following four algorithms:

1. CPABE.Setup is a probabilistic algorithm that takes as input a security parameter λ and outputs a master public key MPK and a master secret key MSK. We denote this by $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$.

1 We assume that a true random function is replaced by a pseudo-random function, the input-output behaviour of which being "computationally indistinguishable" from that of a true random function.

2. CPABE.Gen is a probabilistic algorithm that takes as input a master secret key, a set of attributes $\mathcal{A} \in \Omega$ and the unique identifier of a user, and it outputs a secret key that is bound both to the corresponding list of attributes and the user. We denote this by $(\mathsf{sk}_{\mathcal{A},i}) \leftarrow \mathsf{Gen}(\mathsf{MSK}, \mathcal{A}, u_i)$.

3. CPABE.Enc is a probabilistic algorithm that takes as input a master public key, a message m and a policy $P \in \mathcal{P}$. After a proper run, the algorithm outputs a ciphertext c_P which is associated to the policy P. We denote this by $c_P \leftarrow \mathsf{Enc}(\mathsf{MPK}, m, P)$.

4. CPABE.Dec is a deterministic algorithm that takes as input a user's secret key and a ciphertext and outputs the original message m *iff* the set of attributes $\mathcal{A}$ that are associated with the underlying secret key satisfies the policy P that is associated with c_P. We denote this by $\mathsf{Dec}(\mathsf{sk}_{\mathcal{A},i}, c_P) \rightarrow m$.

2.5 SYSTEM MODEL

The system model presented here is based on the model introduced in [34]. Below we present an overview of the main entities of the system and the most relevant communication between them.

CLOUD SERVICE PROVIDER (CSP) The cloud computing environment is based on a trusted Infrastructure-as-a-Service (IaaS) provider. The IaaS platform consists of cloud hosts that operate virtual machine (VM) guests and communicate through a network. In our model, we require that the IaaS runs a protocol similar to the one described in [35], where the integrity of the underlying CSP is verified. In principle, such integrity verification can be added to any IaaS. A CSP stores patients' EMR encrypted under a CP-ABE scheme. Additionally, the CSP is responsible for controlling access to the encrypted EMR.

REGISTRATION AUTHORITY (RA) The RA is responsible for the registration of all healthcare entities and users. The RA generates user attributes that will be used for the proper authorization (e.g. membership to a particular treatment team). The RA can run as a separate third party, but can be also implemented as part of the CSP. The registration process is out the scope of this work.

MASTER AUTHORITY (MA) The MA has a master secret key MSK and a public key MPK. The master key is kept private, while the public key is known to everyone. Additionally, the MA uses MSK to generate CP-ABE secret keys for users based on her attributes to authorize access to an encrypted EMR. The MA is also responsible for granting and revoking tokens used for dynamic access control.

USER We consider three different types of users: patients, healthcare professionals and healthcare entities. The set of all patients registered at RA is denoted by $\mathcal{U} = \{u_1, \ldots, u_{N_u}\}$ and the set of all registered healthcare professionals is denoted as $\mathcal{S} = \{s_1, \ldots, s_{N_s}\}$. A healthcare entity is a special type of user represented by an attested smart device. This device serves to confirm the following treatment team locations: Emergency Call Centre (e), Ambulance (a) and Hospital (h). A treatment team is a group of professionals co-located at one of the entities that attest each other's involvement in the emergency situation. Each user from $\mathcal{U}, \mathcal{S}$ and the healthcare entities has a unique public/private key pair (pk/sk) used to communicate securely through an IND-CCA2 secure public key encryption scheme PKE and an EUF-CMA secure signature scheme sign.

2.6 PROBLEM STATEMENT AND THREAT MODEL

2.6.1 *Problem Statement*

Let u_i be a patient from the set $\mathcal{U}$ and $s_j \in \mathcal{S}$ be a member of one of the stroke treatment teams. Let's assume that u_i has a set of N different files stored in the CSP. We denote this set of files as $D_i = \{d_1^i, ..., d_N^i\}$. The problem is to find a way to achieve the following:

1. Enable access to the content of each $d_l^i \in D_i$ to s_j involved in the treatment of u_i;
2. User s_j has access to D_i if and only if she has a legitimate role in the treatment team of u_i at the time, as given by a valid policy;
3. Access control to D_i should be granted and revoked dynamically as requested for the patient's treatment. This should not require to decrypt and re-encrypt the file with a fresh key, and it should not affect the access by the rest of the legitimate users.

2.6.2 *Threat Model*

Our threat model is similar to the one described in [35], which is based on the Dolev-Yao adversarial model [36]. We further assume that privileged access rights can be used by a remote adversary ADV to leak confidential information. ADV, e.g. a corrupted system administrator, can obtain remote access to any host maintained by the CSP, but cannot access the volatile memory of guest VMs residing on the compute hosts of the CSP. Moreover, we extend the above threat model by defining a set of attacks available to ADV.

Attack 1 (Token Alteration Attack)**.** *Let* ADV *be a corrupted user that has been legitimately part of a treatment team in the past.* ADV *successfully launches a Token Alteration Attack if she can modify a token* τ *she received in the past in such a way that it will be considered as valid by the CSP.*

Attack 2 (Token Substitution Attack)**.** *Let* ADV *be a corrupted user who overhears all communication and captures a token issued for another legitimate healthcare professional.* ADV *successfully launches a Token Substitution Attack if she can use this token to add false ciphertexts to the patient's EMR.*

Attack 3 (Revocation of Legitimate Users Attack)**.** *Let* ADV *be a corrupted user. Moreover, let* x *be a team who currently has access to patient's* u_i *EMR.* ADV *successfully launches a Revocation of Legitimate Users Attack if she can manage to revoke team* x *from accessing the patients' encrypted data.*

2.7 RED ALERT PROTOCOL

We propose 'Red Alert Protocol' (RAP) for the problem presented in Section 2.6. More precisely, our approach follows the protocols proposed in [34] and [33], with additions to meet the specific needs of the acute stroke care case described in Section 2.3. RAP was initially presented in [37], but here we extend that work by revising the protocol to address a broader threat model and presenting the protocol in a more formal construction.

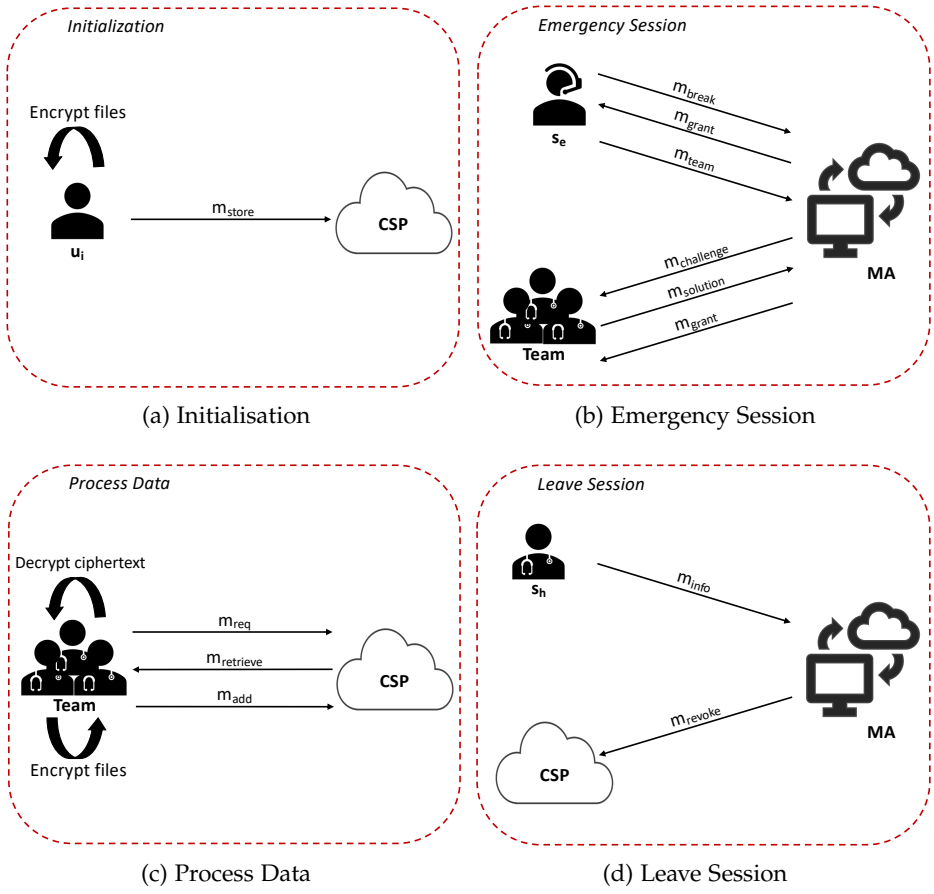
Below we first present an overview of the protocol followed by its definition.

2.7.1 *Protocol Overview*

We assume that each user (from $\mathcal{U}$ or $\mathcal{S}$) is registered through a central RA. However, we consider registration as out of the scope of this paper and assume that all users have been previously registered. Each user receives a unique identifier i, and a set of attributes $\mathcal{A}$ is created based on the user's personal data. For patients, identifying attributes such as name and surname could be used. For healthcare professionals, attributes include identification and function in the organization, and in particular the membership and role in an emergency treatment team. Also, we assume that the EMRs are in a standardized and interoperable format before being encrypted and stored in the CSP.

RAP is divided into *Setup* and four main phases: *Initialisation*, *Emergency Session*, *Process Data* and *Leave Session*. Figure 2.1 shows the messages exchanged between the entities in each phase.

During the *Initialisation* phase, a patient u_i stores her EMR on the CSP as a ciphertext c_p^i. In this paper, we explicitly focus on the problem of how only authorized users can access a patient's EMR during an emergency session. To



(a) Initialisation (b) Emergency Session

(c) Process Data (d) Leave Session

Figure 2.1: Overview of the Red Alert Protocol: entities and their communication during the four phases.

this end, the policy P needs to always contain a condition that will allow a user s_j to successfully decrypt $d_l^i \in D_i, \forall l \in [1, |D_i|]$. Among other conditions in P, the following should be added for u_i: "...OR (Emergency=TRUE AND TreatmentTeamMember=TRUE AND UserInEmergency=i)". A professional s_j will be then granted access to the EMR of u_i only when her attributes satisfy this policy.

In the *Emergency Session* phase, the MA associates the patient with all the treatment teams involved in her emergency session, which ends after complete treatment and patient discharge. The session starts when a patient, or someone on her behalf, contacts the call centre team by phone. Figure 2.2 shows the patient timeline during emergency care. The call centre professional $s_e \in \mathcal{S}$ requests MA to initiate the emergency session; s_e also involves the ambulance team in the session, which ultimately will also involve the hospital treatment

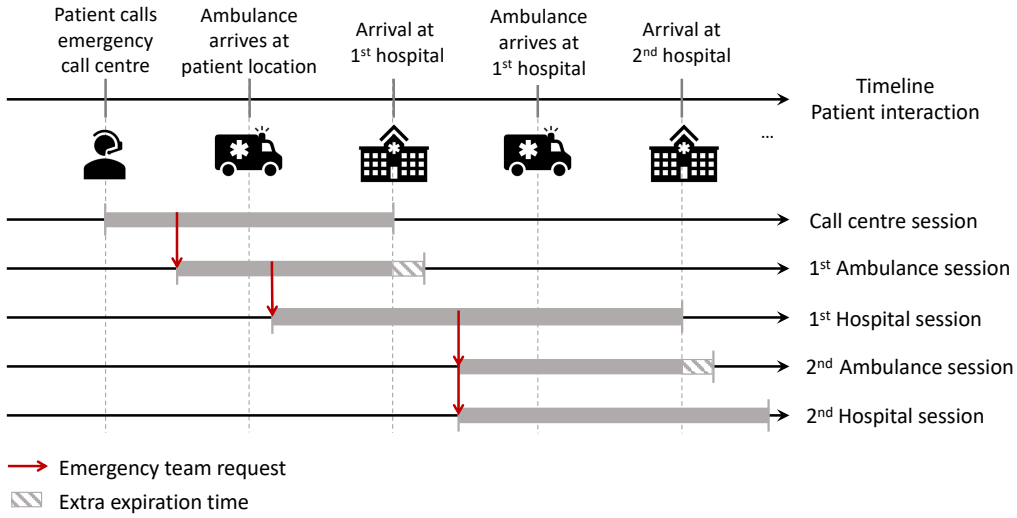


Figure 2.2: Stroke care and data access session timelines. The top line represents events of interaction between healthcare provider entities and the patient. The others show the period of time that each team has access to the patient data.

team. In this proposal, we trust that s_e will contact the MA only if she receives a legitimate phone call from the patient or someone on behalf of the patient. The phone call authentication is very important, but it is considered outside the scope of this paper. However, each involved treatment team needs to prove to the MA that their service is requested. This is done when the treatment team jointly solves a challenge: the healthcare entity x and at least two[2] users respond to the challenge, proving that they are co-located and working together. After the challenge is solved and the users' attributes are validated, the MA generates a CPABE emergency key. As attributes, among others, MA inserts in $\mathcal{A}_e$ the following: "[Emergency, TreatmentTeamMember, i]". This guarantees that the generated key will satisfy the policy bound to u_i's ciphertexts. However, direct sharing of the CPABE emergency key is not secure enough, because getting access to that key would allow anyone to access u_i's ciphertexts at any future moment. Therefore, the MA also generates an access control token τ_x to the team. This token has a default expiration time and also contains the identity of the professionals from the treatment team. The MA subsequently sends the key and token to the team.

In the *Process Data* phase, one of the professionals in the team sends the access token to the CSP to retrieve the patient data. If the token is valid, the CSP grants access to retrieve the ciphertext containing the EMR of the patient under emergency treatment. Through a secure read-only application, the EMR is de-

2 Here we assume that at least two professionals are part of the team, but more could be included in similar way.

crypted by the professional using the CPABE emergency key. Token validation also takes place when s_j adds new data to the patient's EMR by uploading a new ciphertext to the CSP.

The *Leave Session* phase takes place as soon as the patient is no longer under the care of a treatment team. To do so, the MA needs to be informed about the current location of the patient by either check-in or a check-out message. Both messages are sent by the attested smart devices of each treatment team and include a timestamp. These messages can be implemented according to the application. With this information, the MA can revoke the token of the previous team that is no longer involved in the treatment. With this information, the MA can revoke the token of the previous team that is no longer involved in the treatment. In acute stroke care, the moments when the patient arrives and leaves the hospital emergency care unit define the end of the involvement of treatment teams. When the patient arrives at the first hospital, the call centre and ambulance teams leave the emergency session. For the call centre, revocation of τ_e should be immediate. The ambulance team, however, is granted extra time after arrival at the hospital to add their reports into the medical record (see figure 2.2). In principle, τ_h needs to be revoked when the patient leaves the hospital emergency care. However, if the patient needs to be transferred for treatment, the token for the first hospital will be revoked as soon as the patient arrives at the second hospital. As soon as the MA knows the moment when the patient arrived or left the hospital emergency care, it sends a revocation message for the corresponding access token to the CSP. Thus, even if a token is still valid according to the default expiration time, the CSP will not allow any type of access to the data after the revocation time.

The emergency session ends when all tokens associated with it have expired or explicitly revoked. After this, no new team is allowed to join the session anymore.

2.7.2 *Protocol definition*

The 'Red Alert Protocol' (RAP) defines the exchange of messages to grant and revoke access, as well as to rightfully encrypt and decrypt the patient's EMR during an emergency session. In all cases, the entity receiving the message verifies the freshness and the integrity of the message, and it can also authenticate the sender through a signature.

During the phases, all the entities and users interact by running the following algorithms: RAP.Setup, RAP.StoreData, RAP.GrantAccess, RAP.BreakGlass, RAP.JoinTeam, RAP.RetrieveData, RAP.AddData and RAP.RevokeAccess. The phases and algorithms are detailed below as follows: the algorithms are described in-

side frames in each phase where they are used. An overview of all messages exchanged during the execution of the algorithms is presented in Table 2.1.

Setup: before start, each system model entity denoted as ID (for MA, RA, CSP, and users) runs the algorithm RAP.Setup (Algorithm 1). The entities obtain a public/private key pair (pk, sk) for a IND-CCA2 secure public cryptosystem PKE and publish their public key while keeping their private key secret. Furthermore, MA runs CPABE.Setup to acquire a master public/private key pair (MPK, MSK) and publishes the public master key.

Algorithm 1: RAP.Setup

1 **Input**: $(\text{"initialise"}, 1^\lambda)$.
2 **Output**: (pk_{ID}, sk_{ID}), (MPK, MSK).
1: ID runs $(pk, sk) \leftarrow \text{PKE.KeyGen}(1^\lambda)$.
2: ID publishes pk_{ID}.
3: MA runs $(MPK, MSK) \leftarrow \text{CPABE.Setup}(1^\lambda)$.
4: MA publishes MPK.

Initialisation Phase: In this phase, the patient runs RAP.StoreData (Algorithm 2) to encrypt her EMR using CPABE and an emergency policy. After u_i successfully encrypts her data, she sends her c_P^i to the CSP.

Algorithm 2: RAP.StoreData

1 **Input**: A collection of files d_l^i, MPK, emergency policies P.
2 **Output**: A collection of ciphertexts c_P^i stored on the CSP.
1: u_i runs $c_P^i \leftarrow \text{CPABE.Enc}(MPK, d_l^i, P)$
2: u_i generates a random number r_1
3: u_i sends $m_{store} = \langle r_1, c_P^i, \sigma_i(H(r_1 \| c_P^i)) \rangle$ to CSP
4: CSP verifies m_{store}, if the verification fails, output $\perp$.
5: CSP stores c_P^i

Emergency Session: In this phase health professionals involved in an emergency session obtain access to patient data through three algorithms: RAP.BreakGlass, RAP.JoinTeam and RAP.GrantAccess.

RAP.GrantAccess (Algorithm 3): The MA multiple times generates an access token and a CPABE emergency key for each treatment team s_x, where $x \in \{e, a, h\}$.

Algorithm 3: RAP.GrantAccess

1 **Input**: u_i and the team involved in the emergency session (s_x); MSK and emergency attributes $\mathcal{A}_e$.
2 **Output**: Access Token (τ_x) and CPABE emergency key ($sk_{\mathcal{A}_e,i}$).
1: MA calculates the default expiration time: t_{exp}
2: MA generates the timestamp: t_{gen}
3: MA generates a random number r_2
4: MA generates $\tau_x = (t_{gen}, t_{exp}, Enc_{pk_{CSP}}(r_2, s_{x1}, s_{x2}, u_i), \sigma_{MA}(H_\tau))$
5: MA runs $sk_{\mathcal{A}_e,i} \leftarrow CPABE.Gen(MSK, \mathcal{A}_e, u_i)$
6: MA generates a random number r_3
7: MA sends $m_{grant} = \langle r_3, \tau_x, Enc_{pk_{s_x}}(sk_{\mathcal{A}_e,i}), \sigma_{MA}(H(r_3||\tau_x||sk_{\mathcal{A}_e,i}))\rangle$ to the team s_x
8: s_x verifies m_{grant}, if the verification fails, output $\perp$.
9: s_x recovers τ_x and $sk_{\mathcal{A}_e,i}$.

RAP.BreakGlass(Algorithm 4): Through this process the MA acknowledges the emergency event for a patient u_i and begins the emergency session. After identifying patient u_i, s_e contacts MA to notify the emergency event and requests to become part of the emergency session. Upon reception, MA confirms that s_e is indeed part of the call centre team and runs RAP.GrantAccess.

Algorithm 4: RAP.BreakGlass

1 **Input**: u_i, emergency call to s_e.
2 **Output**: MA runs $RAP.GrantAccess(u_i, s_e, MSK, \mathcal{A})$.
1: s_e records the time of the call: t
2: s_e generates a random number r_4
3: s_e sends $m_{break}\langle r_4, Enc_{pk_{MA}}(u_i, t, s_e), \sigma_{s_e}(H(r_4||u_i||t||s_e)\rangle$ to MA
4: MA verifies m_{break}, if the verification fails, output $\perp$.
5: MA checks if $s_e \in \mathcal{S}$; if not, output $\perp$.
6: MA start an emergency session for u_i
7: MA includes s_e in the emergency session

RAP.JoinTeam (Algorithm 5): The MA associates users in a treatment team to an existing emergency session. After the team authentication by solving the challenge, MA includes all the team members (x, s_{x1}, s_{x2}) into the emergency session and runs RAP.GrantAccess.

Algorithm 5: RAP.JoinTeam

1 **Input**: The identities of the new team members x, s_{x1}, s_{x2}, where $x \in \{a, h\}$, and u_i.

2 **Output**: MA runs RAP.GrantAccess$(u_i, s_{x1}, s_{x2}, MSK, \mathcal{A}_e)$.

1: s_j generates a random number r_5
2: s_j sends $m_{team} = \langle r_5, Enc_{pk_{MA}}(x, s_{x_1}, s_{x_2}), \sigma_{s_j}(H(r_5||x||s_{x1}||s_{x2}))\rangle$ to MA
3: MA check if $s_j \in \mathcal{S}$
4: MA verifies m_{team}; if the verification fails, output $\perp$.
5: MA checks if $(x, s_{x1}, s_{x2}) \in \mathcal{S}$; if not, output $\perp$.
6: MA generates a random number v
7: MA splits in three random shares: $v = v_0 + v_1 + v_2$
8: MA generates challenge $= \langle Enc_{pk_x}(v_0), Enc_{pk_{s_{x1}}}(v_1), Enc_{pk_{s_{x2}}}(v_2)\rangle$
9: MA generates a random number r_6
10: MA sends $m_{challenge} = \langle r_6, \text{challenge}, \sigma_{MA}(H(\text{challenge}||r_6))\rangle$ to x, s_{x1}, s_{x2}
11: x, s_{x1} and s_{x2} recover their own share, respectively v'_0, v'_1 and v'_2
12: x, s_{x1} and s_{x2} record their own location, respectively l_x, l_{x1} and l_{x2}
13: x generates a random number r_7
14: x sends $m_{solution} = \langle r_7, Enc_{pk_{MA}}(v'_0, l_x), \sigma_x(H(r_7||v'_0||s_{x1}||s_{x2}||l_x))\rangle$ to MA
15: s_{x1} and s_{x2} send analogous $m_{solution}$ messages to MA
16: MA verifies all $m_{solution}$; if any verification fails, output $\perp$.
17: MA calculates $v' = v'_0 + v'_1 + v'_2$
18: MA verifies if $v = v'$, if not output $\perp$.
19: MA verifies if $l_x = l_{x1} = l_{x2}$, if not output $\perp$.
20: MA includes x, s_{x1}, s_{x2} in the emergency session.

Process Data Phase: After having received the CPABE emergency key and an access token, s_j from team $x \in \{e, a, h\}$ is ready to process the patient's data through either RAP.RetrieveData or RAP.AddData.

RAP.RetrieveData (Algorithm 6): First s_j requests the CSP to retrieve all ciphertexts in the EMR for the patient under emergency treatment. After successful message and token verification, the CSP sends the elegible u_i's ciphertexts to s_j. Finally, s_j uses the CPABE emergency key $sk_{\mathcal{A}_e,i}$ to recover the data from c_P^i.

Algorithm 6: RAP.RetrieveData

1 **Input**: τ_x
2 **Output**: A collection of files d_l^i from patient's EMR.
1: s_j generates a random number r_8
2: s_j sends $m_{req} = \langle r_8, \tau_x, \sigma_{s_j}(H(r_8 \| \tau_x)) \rangle$ to MA
3: CSP verifies m_{req}, if the verification fails, output $\perp$.
4: CSP verifies if τ_x is valid, if not output $\perp$.
5: CSP generates a random number r_9.
6: CSP sends $m_{retrieve} = \langle r_9, c_P^i, \sigma_{CSP}(H(r_9 \| c_P^i)) \rangle$ to s_j
7: s_j runs $d_l^i \leftarrow \mathsf{CPABE.Dec}(\mathsf{sk}_{\mathcal{A}_e,i}, c_P^i)$.

RAP.AddData (Algorithm 7): During and after patient's treatment, all teams may upload new files d_l^i to the patient EMR. The same policy P needs to be used as in the already existing encrypted EMR.

Algorithm 7: RAP.AddData

1 **Input**: MPK, d_l^i, P and τ_x
2 **Output**: A new ciphertext c_P^i stored on the CSP.
1: s_j runs $c_P^i \leftarrow \mathsf{CPABE.Enc}(\mathsf{MPK}, d_l^i, P)$
2: s_j generates a random number r_{10}.
3: s_j sends $m_{add} = \langle r_{10}, \tau_x, c_P^i, \sigma_{s_j}(H(r_{10} \| \tau_x \| c_P^i)) \rangle$ to CSP
4: CSP verifies m_{store}, if the verification fails, output $\perp$.
5: CSP verifies if τ_x is valid, if not output $\perp$.
6: CSP stores c_P^i.

Leave Session:

As soon as the patient arrives or leaves the hospital, s_h initiates RAP.Revoke-Access (Algorithm 8). Subsequently, MA calculates the time to revoke the tokens from the teams which are no longer needed for treatment (see figure 2.2), the MA sends the respective τ_x to be revoked to CSP.

Algorithm 8: RAP.RevokeAccess

1 **Input**: Patient current location.
2 **Output**: τ_x revoked RAP.GrantAccess$(u_i, s_e, \mathsf{MSK}, \mathcal{A})$.
1: s_h records the time of arriving/leaving the hospital: t
2: s_h generates a random number r_{11}
3: s_h sends $m_{info} = \langle r_{11}, \mathsf{E}_{\mathsf{pk}_{MA}}(t), \sigma_{s_h}(H(r_{11}||t))\rangle$ to MA
4: MA verifies m_{info}, if the verification fails, output $\perp$
5: MA checks if $s_h \in \mathcal{S}$, if not output $\perp$
6: MA calculates the time to revoke the tokens
7: MA generates a random number r_{12}
8: MA sends $m_{revoke} = \langle r_{12}, \tau_x, \sigma_{MA}(H(r_{12}||\tau_x))\rangle$ to CSP
9: CSP verifies m_{revoke}, if the verification fails, output $\perp$, and revokes τ_x

Table 2.1: Protocol Messages

Index	Message
m_{store}	$\langle r_1, c_P^i, \sigma_i(H(r_1\|\|c_P^i))\rangle$
m_{grant}	$\langle r_3, \tau_x, \mathsf{Enc}_{\mathsf{pk}_{s_x}}(\mathsf{sk}_{\mathcal{A}_e,i}), \sigma_{MA}(H(r_3\|\|\tau_x\|\|\mathsf{sk}_{\mathcal{A}_e,i}))\rangle$
m_{break}	$\langle r_4, \mathsf{Enc}_{\mathsf{pk}_{MA}}(u_i, t, s_e), \sigma_{s_e}(H(r_4\|\|u_i\|\|t\|\|s_e)\rangle$
m_{team}	$\langle r_5, \mathsf{Enc}_{\mathsf{pk}_{MA}}(x, s_{x_1}, s_{x_2}), \sigma_{s_j}(H(r_5\|\|x\|\|s_{x1}\|\|s_{x2}))\rangle$
$m_{challenge}$	$\langle r_6, \mathsf{challenge}, \sigma_{MA}(H(r_6\|\|\mathsf{challenge}))\rangle$
$m_{solution}$	$\langle r_7, \mathsf{Enc}_{\mathsf{pk}_{MA}}(v_0, l_x), \sigma_{MA}(H(r_7\|\|v_0\|\|s_{x1}\|\|s_{x2}\|\|l_x))\rangle$
m_{req}	$\langle r_8, \tau_x, \sigma_{s_j}(H(r_8\|\|\tau_x))\rangle$
$m_{retrieve}$	$\langle r_9, c_P^i, \sigma_{CSP}(H(r_9\|\|c_P^i))\rangle$
m_{add}	$\langle r_{10}, \tau_x, c_P^i, \sigma_{s_j}(H(r_{10}\|\|\tau_x\|\|c_P^i))\rangle$
m_{info}	$\langle r_{11}, \mathsf{E}_{\mathsf{pk}_{MA}}(t), \sigma_{s_h}(H(r_{11} \parallel t))\rangle$
m_{revoke}	$\langle r_{12}, \tau_x, \sigma_{MA}(H(r_{12}\|\|\tau_x))\rangle$.

2.8 SECURITY ANALYSIS

2.8.1 *Simulation-Based Security*

To prove the security of our protocol, we assume the existence of a simulator coined S . It will simulate the algorithms from the real protocol in a way that any polynomial time adversary ADV will not be able to distinguish between the real algorithms and the simulated ones.

Definition 1 (Simulation-based secure)**.** *We consider the following experiments. In the real experiment, all algorithms run as specified in our protocol. In the ideal experiment,* S *intercepts* ADV*'s queries and replies with simulated responses.*

Real Experiment

1. $\mathbf{EXP}^{real}_{RAP}(1^\lambda)$
2. $(MPK, MSK) \leftarrow RAP.Setup(1^\lambda)$
3. $sk_{\mathcal{A}_e,u_i} \leftarrow ADV^{RAP.GrantAccess(MSK,\cdot\mathcal{A}_e)}$
4. $ct \leftarrow CPABE.Enc(MPK, d^i_l)$
5. *Output* b

Ideal Experiment

1. $\mathbf{EXP}^{ideal}_{RAP}(1^\lambda)$
2. $(MPK) \leftarrow S(1^\lambda)$
3. $sk_{\mathcal{A}_e,u_i} \leftarrow ADV^{S(1^\lambda)}$
4. $ct \leftarrow S(1^\lambda, 1^{|d^i_l|})$
5. *Output* b'

We say that RAP *is sim-secure if for all Probabilistic Polynomial Time (PPT) adversaries* ADV *:*

$$\mathbf{EXP}^{real}_{RAP}(1^\lambda) \approx \mathbf{EXP}^{ideal}_{RAP}(1^\lambda)$$

Theorem 1. *Assuming that* PKE *is an IND-CCA2 secure public key cryptosystem and* Sign *is an EUF-CMA secure signature scheme, then* RAP *is a sim-secure protocol according to Definition 1.*

Proof. We start by defining the algorithms used by the simulator (identified with *). Then, we will replace the real algorithms with the simulated ones. Finally, with the help of a Hybrid Argument, we will prove that the resulted distributions are indistinguishable.

1. RAP.Setup*: Will only generate MPK that will be given to ADV.
2. RAP.StoreData*: Will simulate a ciphertext that has the same length as the output of the real algorithm.
3. RAP.GrantAccess*: Will generate a random string to be sent to ADV. The random string has the same length as the output of the real algorithm. Moreover, S simulates a token τ^* that has the same length as the real token τ.
4. RAP.RetrieveData*: Will return the specified file, without running the decryption protocol.

Then we use a hybrid argument to prove that ADV cannot distinguish between the real and the ideal experiments. The RAP.BreakGlass* and RAP.JoinTeam* oracles are included in the RAP.GrantAccess* one. The reason for this is that, during the execution of both of these algorithms, ADV queries RAP.GenKey and

RAP.GenToken. Moreover, since RAP.Revoke does not produce any output, we can exclude it from our proof. Finally, RAP.AddData can be seen as a special case of RAP.StoreData. As a result, by proving that StoreData is secure, we also prove that RAP.AddData is secure. In a pre-processing phase, S creates a list $\mathcal{L}$ in which it will store the files used by ADV in RAP.StoreData.

Hybrid 0 RAP runs normally.

Hybrid 1 Everything runs like Hybrid 0, but we replace RAP.Setup with RAP.Setup*.

These algorithms are identical from the ADV's perspective and as a result the Hybrids are indistinguishable.

Hybrid 2 Everything runs like Hybrid 1, but we replace RAP.StoreData with RAP.StoreData*.

At this point, S will simulate a ciphertext that will be sent to ADV. Moreover, S will store in $\mathcal{L}$ the tuple $(d_l^i, {c_p^i}^*)$ where d_l^i is a file that was given as input by ADV and ${c_p^i}^*$ is the simulated ciphertext that corresponds to d_l^i. The Hybrids are indistinguishable from ADV's point of view, since she receives what she believes to be a valid ciphertext.

Hybrid 3 Everything runs like Hybrid 1, but we replace RAP.BreakGlass with RAP.BreakGlass*.

Again the algorithms are identical from ADV's point of view and thus, the Hybrids are indistinguishable.

Hybrid 4 Everything runs like Hybrid 2, but we replace RAP.RetrieveData with RAP.RetrieveData*.

At this point, S retrieves $\mathcal{L}$, finds the d_l^i that corresponds to the c_p^i that was given as input by ADV, and returns it. Clearly, since ADV receives the file she was waiting for, the Hybrids are indistinguishable.

With this Hybrid, our proof is complete. We managed to replace the expected outputs with simulated responses in a way that ADV cannot distinguish between the real and the ideal experiment.

□

2.8.2 *Protocol Security*

In this section, we prove the resilience of our protocol against the set of attacks defined in section 2.6.2. We assume that the random numbers r_i generated throughout the protocol are stored locally on each entity. To ensure that r_i is used only once, it is time-variant, including a suitably fine-grained timestamp in its value. In this way we can guarantee the freshness of the exchanged messages.

Proposition 1. *(Token Alteration Attack Soundness) Let* ADV *be a corrupted user and* u_i *be a patient whose EMR is stored in the CSP. Moreover, we assume that* ADV*'s access to* u_i*'s EMR has been revoked. Then,* ADV *cannot successfully perform a Token Alteration Attack.*

Proof. Since ADV's access is revoked, it is implied that at some point in the past ADV received a valid token $\tau_x = (t_{gen}, t_{exp}, \mathsf{Enc}_{pk_{CSP}}(r, s_{x_1}, s_{x_2}, u_i), \sigma_{MA}(H(\tau_x))$ to access the medical records of a user u_i. As a result, all ADV needs to do in order to launch a Token Alteration Attack is to modify the timestamps contained in τ_x. However, since the timestamps are also contained in the hash of the signature of MA, altering the timestamps is equivalent to forging MA's signature, which, given the EUF-CMA security of the signature scheme, can only happen with negligible probability. Therefore, the attack fails. □

Proposition 2. *(Token Substitution Attack Soundness): Let* ADV *be a corrupted user who overhears all communication and captures a token* τ *issued to another legitimate healthcare professional* s_l*. Then* ADV *cannot successfully launch a Token Substitution Attack.*

Proof. ADV can capture the token τ by overhearing the messages m_{grant}, m_{req} and m_{add}. However, the intercepted τ contains the identity of s_l, which cannot be changed because ADV cannot generate a valid signature as we already proved; therefore, the protocol is secure against a Token Alteration Attack. The only alternative for ADV is to use τ directly. To this end, ADV runs $c_P^i \leftarrow \mathsf{CPABE.Enc}(MPK, d_l^i, P)$ for a fake d_l^i, in an attempt to create a valid m_{add} message. However, for ADV to successfully create a m_{add}, she also needs to forge s_l's signature, which can only happen with negligible probability since we assume that the signature scheme is EUF-CMA secure. As a result, the attack will fail. □

Proposition 3. *(Revocation of Legitimate Users Attack Soundeness): Let* ADV *be a corrupted user.* ADV *cannot successfully launch a Revocation of Legitimate Users Attack.*

Proof. ADV commences the attack by trying to construct a valid $m_{info} = \langle r_{11}, \mathsf{E}_{pk_{MA}}(t), \sigma_{s_h} H((r_{11} \| t)) \rangle$ message for MA. However, this message needs to be

signed by a s_h who is already a legitimate member of the team. As a result, creating a valid m_{info} message is equivalent to forging s_h's signature, which can only happen with negligible probability, since the signature scheme is EUF-CMA secure. The only other option for ADV is to bypass MA and try to communicate directly with the CSP. To this end, ADV tries to construct a valid $m_{revoke} = \langle r_{12}, \tau_x, \sigma_{MA}(H(r_{12}||t_x))\rangle$. However, once again the EUF-CMA security will prevent ADV from forging MA's signature, and as a result the attack will fail. □

2.9 EXPERIMENTAL RESULTS

In this section, we present the implementation of the core functions of our protocol. We prove the effectiveness of the proposed protocol by evaluating the processing time of the core functions on a standalone Linux machine. Our experiments mainly focused on the key generation phases in RAP.Setup and RAP.GrantAccess, encryption, decryption, signing, and verification functions. For the encryption/decryption and signing/verifying we used the RSA cryptosystem, and for the Attribute-Based Encryption scheme we used the CPABE library provided in [32]. Finally, SHA256 used as the main cryptographic hash function.

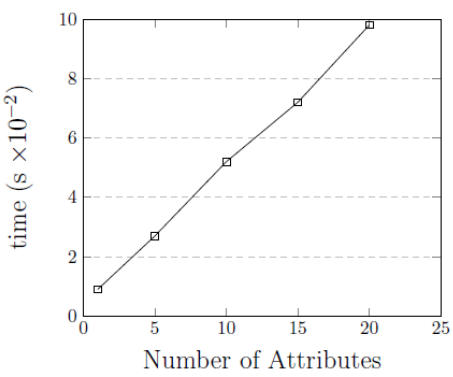
The experiments were carried out on an Intel Core i7-4790 CPU @ 3.60 GHzx8 Ubuntu 18.04.2 Desktop with 16 GB of RAM. The implementation was done in the C language. Furthermore, to provide a well-rounded evaluation of the protocol's performance we simulated a plethora of scenarios using different parameters. To acquire accurate measurements, we ran each experiment 50 times and calculated the average time needed to successfully complete the underlying process. Experiments to measure the execution times of functions related to the CSP and the communication channels utilized by the proposed protocol were considered to be out the scope of this experimentation section. The experiments were carried out in phases according to the RAP protocol steps.

Setup: This phase was dedicated to generating the keys that are used for all cryptographic functions. This phase corresponds to the RAP.Setup in our proposed protocol. We measured the time needed to generate the master public/private key pair using CPABE.Setup as well as the RSA public/private key pairs for each entity. In our protocol a single MA exists – hence, a single (MPK, MSK) key pair is generated during the setup phase. The average execution time measured to generate the (MPK, MSK) key pair was 0.014 sec in 50 iterations. Furthermore, the time required to generate each user's unique (pk, sk) key pair was measured at an average of 0.086 sec per user in 50 iterations.

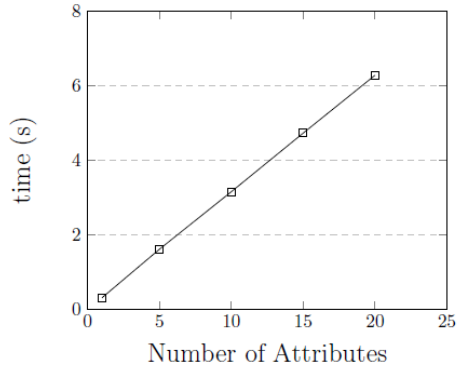
User Key Generation: In these experiments we focused on the generation of an emergency CPABE key for each treatment team – a functionality that is part of protocol's RAP.GrantAccess phase. More precisely, we measured the processing time of the CPABE.Gen function that takes as input an arbitrary number of attributes and outputs a unique secret key for each entity. Two types of attributes were used for these experiments. The first type is of the form ATTRIBUTE_i (i.e a list of attributes), while the other is of the form ATTRIBUTE = i (i.e. we assigned values to attributes). The reason for using different types of attributes is that, while running our experiments, we identified significant differences in the processing time. More precisely, generating keys with attributes to which we have assigned values (e.g. ATTRIBUTE = i) required significantly more time for the generation of a key. The experiments involved an arbitrary number of attributes from 1 to 20. The results of the experiments varied greatly depending on the type of attributes used. For attributes of the type ATTRIBUTE_i, execution times measured for generating a CPABE key with 5 attributes was about 0.026 sec, while with 20 attributes it was about 0.102 sec in 50 iterations (figure 2.3 (a)). For attributes of the type ATTRIBUTE = i, the execution times measured for generating a CPABE key with 5 attributes it was about 1.642 sec and with 20 attributes was about 6.306 sec in 50 iterations (figure 2.3 (b)). Apart from that, generating CPABE keys increases linearly with the number of attributes for both types of attributes. Finally, by comparing the results shown in Figures 2.3 (a) and 2.3 (b) we see that generating keys with simple attributes (i.e. no assigned values) will result in a more efficient implementation of our protocol.

EMR file encryption and decryption: A core function of our protocol is the encryption and decryption of a patient's EMR file using CPABE. To measure this process, we ran experiments where we encrypted files with various sizes associated with an emergency policy and a set of attributes. The combination of an arbitrary number of attributes and file sizes allowed us to simulate more realistic cases. The first part of this experiment involved files of different sizes with a fixed number of attributes (i.e., static policy size). We encrypted files of sizes ranging from 1 MB to 20 MB with a policy requiring five attributes of type ATTRIBUTE_i. The file of size 1 MB was encrypted in about 0.034 sec and decrypted in about 0.019 sec. The file with size 20 MB was encrypted in about 0.1567 sec and decrypted in about 0.1784 sec. Figures 2.3 (c) and 2.3 (d) show that the processing time increases linearly as the file size is increased.

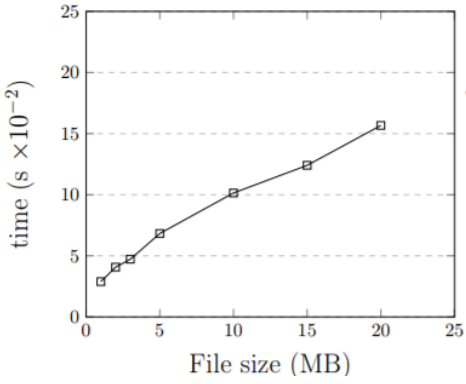
The next experiment involved a file of fixed size and a policy with a variable number of attributes. We encrypted a file of 5MB with a policy with 5 to 20 attributes of the same type as in the previous phase. Encryption and decryption times increased as the policy size increased. A file of size 5MB with a policy of size 20 was encrypted in about 0.1337 sec and successfully decrypted



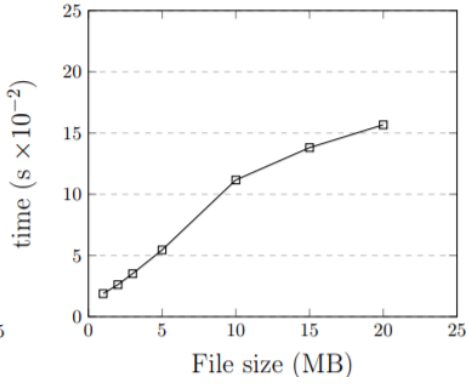
(a) Attribute type ATTRIBUTE_1



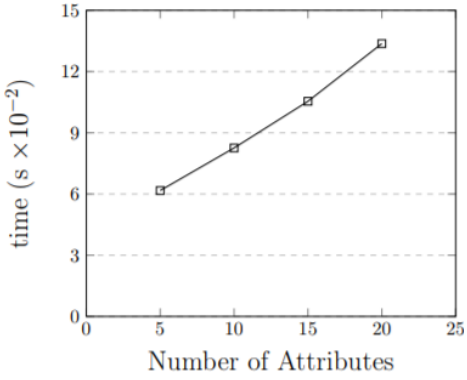
(b) Attribute type ATTRIBUTE = 1



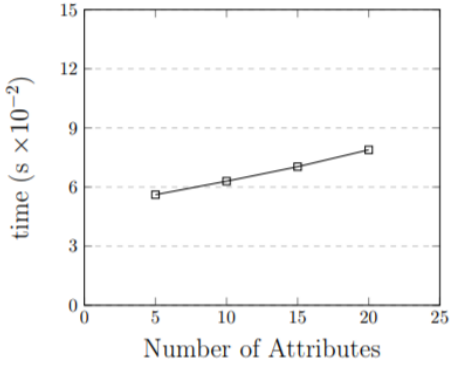
(c) Enc with variable file Size



(d) Dec with variable file Size



(e) Enc with variable policy Size



(f) Dec with variable policy Size

Figure 2.3: Overview of the experimental results

in about 0.0789 sec. Figures 2.3 (e) and 2.3 (f) illustrate the results of our experiments with a policy of variable size.

Token Generation, signing and verification: Our protocol depends heavily on the token generated by the MA in RAP.GrantAccess. In our experiments, we measured the time taken to generate the token, that is, to generate a message comprising of t_{gen}, t_{exp}, $Enc_{pk_{CSP}}(r_2, s_{x1}, s_{x2}, u_i)$ and $\sigma_{MA}(H_\tau)$. Our results indicate a total execution time of about 1.671×10^{-3} sec.

2.10 DISCUSSION

From the results of the presented experiments, we confirm that the performance of the encryption and decryption functions depends on the size of the policy of the ciphertext, the attributes attached to a user's secret key and the size of the EMR file. The overall performance can be improved by optimizing the way we generate the attributes. Attributes of the type ATTRIBUTE_i should be utilised as the execution times for these are more efficient. Furthermore, it is evident from the experimental results that the time needed for the execution of the protocol renders our construction feasible, even when we increase the number of attributes. As a next step, we plan to experiment with different ABE schemes in order to find the one that best suites our construction and evaluate the performance with larger size files, which would be more realistic for images and signals data.

In the protocol, we assume that all users are registered with the RA. However, we understand that there are cases when the patient is not registered or cannot be identified. Thus, for those cases, one possible option is to create a temporary 'John Doe' user to receive the record of the current stroke acute care patient. Using this approach, the professionals would still be able to use the system to share information about patient treatment. However, RAP needs to be able to merge the information as soon as the patient is identified. We plan to support those cases in the next version of the protocol.

In addition, the protocol relies on the token revocation list in the CSP to do the access control. The only way to bypass this is through an internal attack on the CSP. To strengthen the CSP, we could assume the existence of a trusted execution environment, such as Intel's SGX [38], that will further secure the token-based access control. We believe that SGX is a good candidate for our construction since it offers isolation, sealing and attestation functionalities. More information can be found at [39].

Moreover, it is important to emphasise that access to the EMR must be implemented through a secure read-only application, where the EMR is decrypted by the professional using the CPABE emergency key. The application must not allow downloading the files. Thus, the EMR is just available during the emergency session.

2.11 CONCLUSION

In this paper, we proposed Red Alert, a protocol based on Ciphertext-Policy Attribute-Based Encryption that allows access control to encrypted medical data during emergency situations. The proposed scheme enables healthcare professionals to decrypt a patient's encrypted data by making use of time-based tokens that are issued during an emergency situation. After the expiration of the tokens, the users are revoked and can no longer access the patient's data. The security of our scheme is proven using both simulation-based security as well as direct attacks on the protocol. Finally, we proved that the time for the RAP core functions execution is feasible in an emergency situation since the approximate sum of execution times of the primary functions is below 0.5 seconds, and the message exchange between the entities would happen before the patient be actually under the team treatment. Therefore, RAP enables the patient's EMR availability for the teams before the treatment begins, which can potentially improve patient care without compromising security and patient privacy.

3

REVOCABLE ACCESS CONTROL FOR ACUTE CARE TEAMS TO ACCESS MEDICAL RECORDS

Acute care demands the collaboration of multiple healthcare professionals and various organisations. During an emergency, the availability of Electronic Medical Records (EMR) allows acute care teams to access a patient's data promptly, which facilitates the decision-making process. Cloud solutions offer an environment to store and share patients' EMR. However, security and privacy issues arise, which affect the availability of the patient's EMR. Inspired by a hybrid encryption scheme combining Dynamic index-based Symmetric Searchable Encryption (DSSE) and Attribute-Based Encryption (ABE), we proposed the AC-AC protocol. AC-AC is a dynamic revocable access control protocol that enables break-glass access for an authorised member of an acute care team that is treating the patient. The proposed protocol allows a team to grant and revoke access for other teams to the patient's EMR dynamically according to the demands for the treatment. We present a formal security analysis proving that AC-AC protocol is resilient to multiple attacks. Finally, we analysed the overhead in time complexity for the protocol execution and experimented each algorithm. The experimental expected execution time for the AC-AC algorithms was below 170 ms, therefore feasible for an acute care timeline.

This Chapter is based on:

Marcela Tuler de Oliveira, Hai-Van Dang, Lúcio Henrik Amorim Reis, Henk A. Marquering, and Sílvia Delgado Olabarriaga. "AC-AC: Dynamic revocable access control for acute care teams to access medical records." In: Smart Health 20 (2021), p. 100190. Elsevier [40]

3.1 INTRODUCTION

Having the right information at the right time is very important for acute care. During an emergency, healthcare professionals need to evaluate the patient's condition and decide upon the treatment in a short time frame. Only 2–5% of patients who have a stroke receive adequate treatment, mainly due to delays for diagnosis and for reaching the comprehensive hospital with sufficient stroke care conditions [41]. The availability of a unified Electronic Medical Records (EMR) would improve the overall quality of care [1], leading to a substantial reduction of unnecessary investigations and optimised communication among the healthcare professionals involved in the treatment. Researchers have shown that the sooner the treatment is given, the better the functional outcomes for the patient [13]. Moreover, patient transportation at the highest priority and hospital notification before patient arrival were associated with faster stroke care [2].

Emergency treatments usually request a complex collaboration of various healthcare organisations. The use of cloud storage services provides an environment matching the needs for remote and ubiquitous access to EMR. However, security and privacy challenges are impeding the wide adoption of cloud services. Patients and healthcare organisations are afraid of losing control over the EMR when storing it on untrusted third-party clouds [14]. Researchers suggested encrypting the EMR with a secret key before storing it in the cloud [15, 16, 42]. This means that a secret key needs to be shared beforehand with every healthcare professional who wishes or needs to access the EMR at any time throughout the patient's triage and treatment. Other researchers try to address the problem by combining modern cryptographic techniques such as Dynamic index-based Symmetric Searchable Encryption (DSSE) and Attribute-Based Encryption (ABE) to protect data and solve access revocation problems [39, 43]. These modern techniques allow dynamic management of key access, therefore enabling more flexible access control.

Break-glass access embodies the idea that under certain conditions it is possible for a user to break-the-glass and explicitly override a denied access request [28]. Although some studies approach break-glass access to an encrypted EMR [21–23], access revocation after an emergency is still a problem. The General Data Protection Regulation (GDPR) [8] attests that healthcare professionals and healthcare organisations do not have an obligation to ask systematically for patients' consent before they can use the data contained in the EMR. However, they are bound by all the principles described in Article 5, which ensures the exemption from consent is proportionate and limited to what is necessary for the patients' health care. Therefore, the professionals involved in acute care must lose access to the patient's EMR immediately after completing their tasks in the emergency triage and treatment. In [20], the proposed protocol allows

break-glass access to an encrypted patient's EMR only during an emergency and only for authorised treatment teams. The protocol is based on the ABE scheme to encrypt the EMR. However, it has a scalability problem once the size of the data and access control complexity increase. Our goal is to overcome this problem using DSSE to encrypt the patients' EMR and using the ABE scheme to encrypt the symmetric keys, as a second layer of encryption. Moreover, we propose to use a dynamic access control protocol that grants and revokes access to the symmetric keys and the ciphertexts based on scopes of the access rights.

OUR CONTRIBUTION We describe a protocol for dynamic revocable Access Control for Acute Care (AC-AC) to provide access to a patient's encrypted EMR during acute care. The AC-AC protocol is meant to secure EMR's stored on ubiquitous access cloud platforms. The proposed solution extends the concept presented in MicroSCOPE [43], which is based on a hybrid encryption scheme that utilises the advantages of both DSSE and ABE. AC-AC introduces an additional security mechanism that enables break-glass access to the EMR with dynamic revocation to the multiple acute care teams involved during the treatment timeline. We prove the security of the proposed scheme by constructing a simulator that is computationally indistinguishable from the real implemented protocol. Finally, we show the overhead added by the protocol through time complexity analysis and by performance measurements on an implementation of the AC-AC protocol core algorithms.

The paper consists of the following sections. Section 3.2 presents an overview of the related works. After that, section 3.3 describes the scenario of stroke acute care, based on which we identify the problems to be solved in this study. Section 3.4 defines the cryptographic primitives, the system model of the protocol, the threat model, and summarises the MicroSCOPE protocol [43], on which we rely to construct our proposal. Section 3.5 presents our proposal, the Access Control for Acute Care (AC-AC) protocol. Section 3.6 contains a security analysis and section 3.7 presents a performance evaluation. Finally, in section 3.8 we discuss the proposed protocol, and in section 3.9 we conclude the paper contributions.

3.2 RELATED WORK

Security and privacy for cloud computing is an extensive research area. Researchers proposed various access control protocols for known and legitimate users. Our work, however, focuses on the specific case of emergencies, where the users who need access are unknown to the data owner, but legitimate.

Povey [25] was one of the first to formulate the necessity of a security system that allows EMR availability during an emergency. Break-glass access al-

lows overruling the access control enforcement mechanism, which would otherwise deny access. In [26], the authors investigated a break-glass access approach where the data subjects are allowed to override specific access control permissions. Marinovic et al. [28] proposed a model that implements access permissions in a fine-grained manner using a declarative query language that explicitly specifies a break-glass decision procedure. However, [25, 26, 28] only present the architecture models rather than a concrete cryptographic scheme.

The following researches propose access control protocols for cloud solutions using end-to-end encryption, where the cloud is not responsible for the encryption or decryption of the data. Table 3.1 summarises the related works with their main features.

Table 3.1: Characterisation of related work that use end-to-end encryption with respect to their leveraging of novel features

Related work	Break-glass	Dynamic revocation	Fine-grained access control	Hybrid encryption
[27]	✓			
[21]	✓		✓	
[22]	✓		✓	
[23]	✓		✓	
[44]	✓		✓	
[42]	✓		✓	
[45]	✓		✓	
[46]	✓		✓	
[20]	✓	✓		
[47]			✓	✓
[48]			✓	✓
[49]	✓		✓	✓
[43]		✓	✓	✓
AC-AC	✓	✓	✓	✓

Zhang et al. [27] proposed a break-glass solution based on two-factor encryption: password-based encryption and master secret key-based encryption. Other researchers use attribute-based encryption (ABE) techniques to offer a fine-grained control access to patient data, where each data item is given its own access control policy. These studies approach break-glass access under emergency scenarios using a unique authority to authorise the medical staff to access the data [21–23]. Other approaches investigate multi-authority ABE schemes to reduce the trust issues of a single authority [42, 44]. By combining ABE and Role-Based Access Control (RBAC), [45, 46] proposed break-glass ac-

cess control models that enable the professionals who have a legitimate role to access the patient's EMR. Various solutions for break-glass access control have been proposed in [21–23, 27, 42, 44–46]. However, none of these provides a concrete and dynamic solution for access revocation. In a previous work, Oliveira et al. [20] proposed a protocol using ciphertext attribute-based encryption (CP-ABE) to encrypt the patient's EMR under emergency policies. Additionally, it used a short-lived access token to enable revocation. However, this approach faced scalability issues regarding the policy size, the number of attributes and the EMR size. Similarly, the solutions proposed in [21–23, 42, 44–46], which rely on ABE encryption, face the same limitations.

Other researchers proposed hybrid encryption schemes, to overcome the performance issue of ABE by using symmetric encryption combined with ABE. Sun et al. [47] proposed an attribute-based keyword searching scheme with a revocation mechanism. However, that study does not describe how it would work for break-glass access. Guo et al. [48] also proposed a scheme based on symmetric encryption and ABE, in which the data owner encrypts their files using a symmetric key and encrypts the index of the symmetric keys under ABE. This scheme shows limitations to share the keys and does not provide a revocation mechanism. Padhya et al. [49] proposed a protocol that allows break-the-class procedure using key aggregate searchable encryption with a revocation mechanism. However, this revocation mechanism relies on the re-encryption of the documents. To do so, the data owner needs to actively provide a list of documents and a list of user identifiers to be revoked. This protocol does not offer the dynamics needed to revoke access to multiples acute care team involved right after they finish participation in the emergency timeline.

Our proposal is an extension of MicroSCOPE [43], which used DSSE to encrypt the data, and ABE and Software Guard Extension (SGX) for DSSE key's encryption and management. The hybrid encryption scheme solves the problem of dynamic granting and revoking fine-grain access rights defined as scopes. However, MicroSCOPE [43] originally does not define break-glass access because the data owner (e.g. the patient) needs to identify who and at which scopes to share data with beforehand. In contrast, our proposal is designed to support break-glass access, and tailored to emergency scenarios. It focuses on the aspect of dynamic access control to all acute care teams involved in the patient during the treatment.

3.3 STROKE ACUTE CARE

Acute stroke care is a complex collaboration of various parties involving professionals at the emergency call centre, at the emergency ambulance service and at hospitals. Currently, treatment in the acute phase of ischemic stroke consists

of intravenous thrombolysis (IVT) and endovascular treatment (EVT). While almost all hospitals (primary stroke centres) provide IVT treatments, EVT is a highly specialised treatment only provided in a few hospitals (comprehensive stroke centres). A significant challenge is to determine the adequate treatment and place the patient at the treatment centre with adequate conditions for intervention [20]. This decision-making is based on information about the patient's condition at that moment but is also supported by the patient's previous records, which could lead to a substantial reduction of time on unnecessary investigations. The phrase *'Time is brain'* conveys the idea that minutes can make the difference between life and death [13]. Because of that, the availability of the patient's EMR is essential to speed up the treatment and also to optimise the communication among the healthcare professionals involved.

The diagram in Fig 3.1 illustrates a simplified scenario of acute stroke care in which different acute care teams treat a patient. First, the patient is taken by ambulance to a primary care hospital, and then she/he needs to be transferred to a comprehensive stroke centre for EVT. Acute care starts when the patient, a family member or the general practitioner contacts the emergency call centre. During the phone call, a trained healthcare worker follows a triage protocol to determine whether there is suspicion of stroke. This worker then requests an ambulance to pick up the patient. The ambulance service centre assigns an ambulance team to pick up the patient. Once the ambulance arrives and takes charge of the patient, the call centre professional hangs up the call and leaves the treatment. When the ambulance team examines the patient and confirms the stroke suspicion, a request is sent to the closest primary stroke care centre to receive the patient. The ambulance team contacts the hospital by phone to inform the estimated arrival time and relevant patient conditions. The primary hospital administration assigns a team to treat the patient. When the patient arrives at the hospital, the primary hospital team starts the treatment, and the ambulance team leaves the treatment. If the patient is eligible for EVT, the primary hospital team requests a comprehensive stroke centre to receive the patient and also requests an ambulance to transfer the patient. The ambulance central and the comprehensive stroke centre assign teams to transfer and to treat the patient, respectively. After the second ambulance transportation, the ambulance team leaves the treatment, and the patient receives EVT at the comprehensive stroke centre. Finally, when the patient is no longer in risk, the patient is discharged from the emergency department.

From the first call to the emergency call centre, all the teams need to exchange information while treating the patient. Currently, this information is exchanged orally or by phone, as there is no unified EMR that can be shared by all professionals during treatment. Such conventional information sharing method consumes time and effort, being also fault-prone. Therefore, in our proposal, the

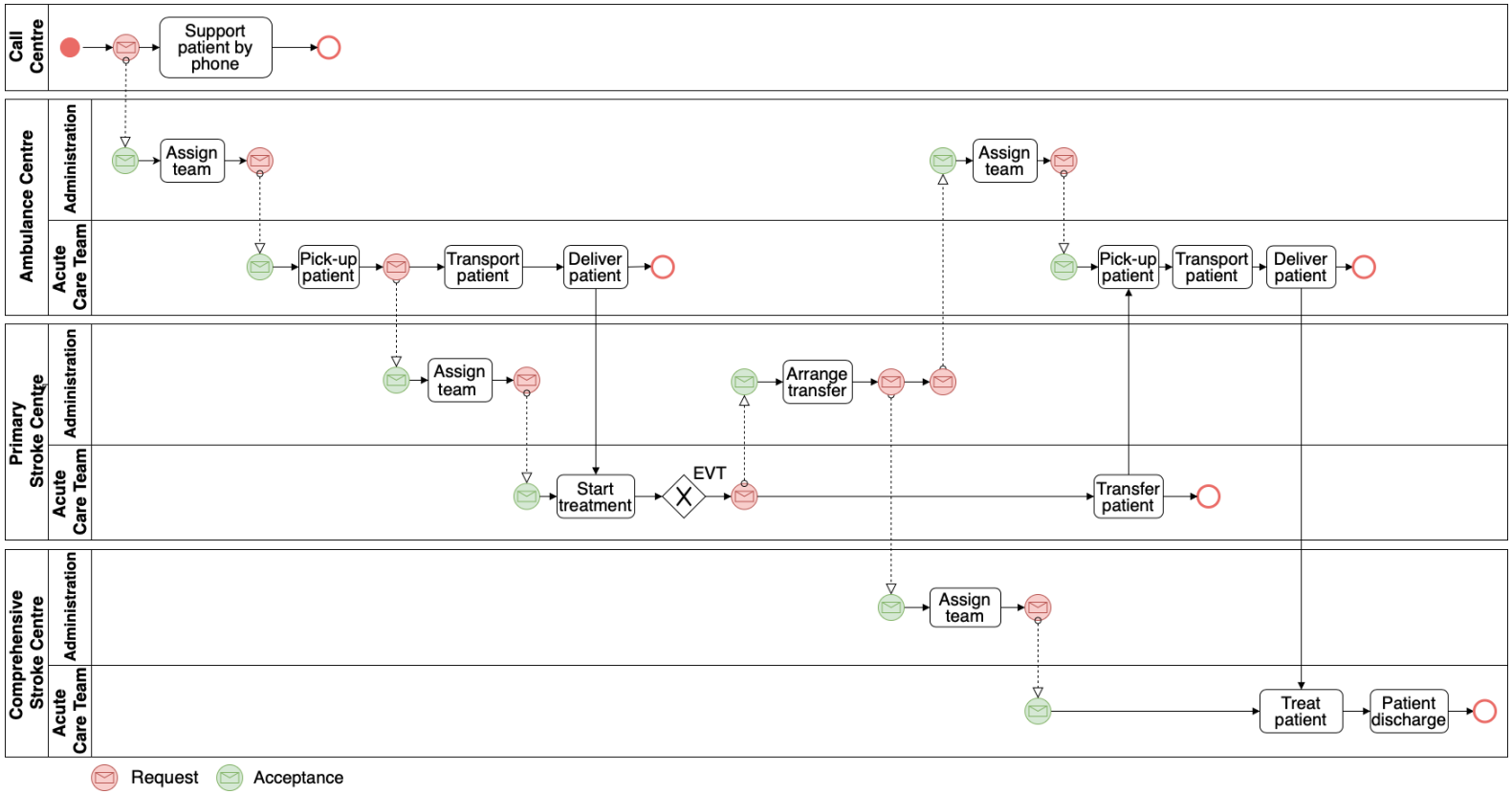


Figure 3.1: Example of interaction among institutions during stroke acute care in Business Process Modeling Notation (BPMN 2.0) [50].

patient's EMR will be stored in a cloud system to improve accessibility and collection of medical records during an emergency. Furthermore, to avoid compromising the patient's privacy, the patient's EMR will be stored as ciphertext in the cloud, and the encryption keys will be shared with the involved teams. Additionally, the proposed protocol will enable dynamic granting and revoking access to the patient's EMR for the healthcare teams, from the emergency phone call until patient discharge. Our goals are further elaborated below.

PROBLEM STATEMENT We assume that a registered patient has his/her EMR encrypted and stored in the cloud. The patient requests an emergency treatment, where various acute care teams may be part of the treatment. Our protocol aims at providing the following features:

1. Enable access to the patient's EMR to the members of all acute care teams involved in the treatment of the patient;
2. Enable access to the patient's EMR if and only if the team has a legitimate invitation to collaborate in the patient's treatment;
3. Access granting and revocation should be dynamic as demanded for patient treatment.
4. Revocation of a team should not require decrypting and re-encrypting the EMR with a new key, and it should not affect the access of other legitimate teams.

3.4 DEFINITIONS

This section introduces the definitions, protocols and models used to build the proposed solution: cryptographic primitives, system model, threat model and MicroSCOPE overview.

3.4.1 *Cryptographic Primitives*

Here we present the notation for cryptographic operations used throughout the paper. Moreover, we present the formal definitions used in [43] for Ciphertext-Policy Attribute-Based Encryption (CP-ABE), Dynamic index-based Symmetric Searchable Encryption (DSSE) and Software Security Guard Extensions (SGX) concepts.

The set of all binary strings of length n is denoted by $\{0,1\}^n$, and the set of all finite binary strings as $\{0,1\}^*$. Given a set V, we refer to the i^{th} element as v_i. When W is a subset of V, we refer to the i^{th} element of W as v_i^w.

- For an arbitrary message $m \in \{0,1\}^*$, $c = \mathsf{Enc}(K, m)$ denotes a symmetric encryption of m using the secret key $K \in \{0,1\}^*$, and $m = \mathsf{Dec}(K, c) = \mathsf{Dec}(K, \mathsf{Enc}(K, m))$ is the corresponding symmetric decryption operation.
- We denote by pk/sk a public/private key pair for an IND-CCA2 secure public key encryption scheme PKE. An encryption of message m under the public key pk is denoted by $c = \mathsf{Enc}_{pk}(m)$ and the corresponding decryption operation by $m = \mathsf{Dec}_{sk}(c)$=$\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m))$.
- $\sigma = \mathsf{Sign}_{sk}(m)$ denotes a EUF-CMA secure digital signature over a message m. The corresponding verification operation for a digital signature is denoted by b =$\mathsf{Verify}_{pk}(m, \sigma)$, where b = 1 if the signature is valid, and b=0 otherwise.
- $H = H(m)$ denoted a one-way hash function (H) over a message m is denoted by $H = H(m)$.
- $r = \mathsf{RAND}(n)$ denotes a random binary sequence of length n, where $\mathsf{RAND}(n)$ represents a random function that takes a binary sequence as input and returns a random binary sequence of the same length[1].

CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION (CP-ABE) The CP-ABE scheme follows the definition presented in [32, 33] as a tuple of the following four algorithms:

1. CPABE.Setup is a probabilistic algorithm that takes as input a security parameter λ and outputs a master public key MPK and a master secret key MSK. We denote this by $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$.
2. CPABE.Gen is a probabilistic algorithm that takes as input a master secret key, a set of attributes $\mathcal{A} \in \Omega$ and the unique identifier of a user u_i, and it outputs a secret key that is bound both to the corresponding list of attributes and the user. We denote this by $(\mathsf{sk}_{\mathcal{A},u_i}) \leftarrow \mathsf{Gen}(\mathsf{MSK}, \mathcal{A}, u_i)$.
3. CPABE.Enc is a probabilistic algorithm that takes as input a master public key, a message m and a policy $P \in \mathcal{P}$. After a proper run, the algorithm outputs a ciphertext c_P that is associated with the policy P. We denote this by $c_P \leftarrow \mathsf{Enc}(\mathsf{MPK}, m, P)$.
4. CPABE.Dec is a deterministic algorithm that takes as input a user's secret key and ciphertext and outputs the original message m *iff* the set of at-

1 We assume that a true random function is replaced by a pseudo-random function, the input-output behaviour of which being "computationally indistinguishable" from that of a true random function.

tributes $\mathcal{A}$ that are associated with the underlying secret key satisfies the policy P that is associated with c_P. We denote this by $\mathsf{Dec}(\mathsf{sk}_{\mathcal{A},i}, c_P) \to m$.

DYNAMIC INDEX-BASED SYMMETRIC SEARCHABLE ENCRYPTION (DSSE) The DSSE scheme follows the definition presented in [39], which consists of the following algorithms:

1. DSSE.KeyGen is a probabilistic key generation algorithm used by the user. It takes as input a security parameter λ and outputs a secret key K.
2. DSSE.InGen is a probabilistic algorithm that takes as input a secret key K and a collection of files f and outputs an encrypted index γ and a sequence of ciphertexts c_f corresponding to her files.
3. DSSE.AddFile is a probabilistic algorithm that takes as input a K and a file f and outputs an add token $\tau_a(f)$ and a ciphertext c_f. The token and the ciphertext are then sent to the storage server, where c_f will be added to the collection of ciphertexts and the index γ will be updated accordingly.
4. DSSE.Search is a deterministic algorithm that takes as input a secret key K and a keyword w and outputs a search token $\tau_s(w)$. The token is then sent to a storage server which will output a sequence of file identifiers $I_w \in c_f$.
5. DSSE.Delete is a deterministic algorithm that takes as input a secret key K and a files identifier $\mathtt{id}(f)$ and outputs a delete token $\tau_d(f)$ for f. The token will be sent to the storage server, which will delete c_f and update the index γ accordingly.

SOFTWARE SECURITY GUARD EXTENSIONS (SGX) SGX is a set of instructions for security enhancement in the CPU. It allows defining private and secure regions of the memory called enclaves, in which every code is protected with cryptography – see more detailed description in [51]. SGX main functionalities are:

1. Isolation, where enclaves are located in a hardware-guarded area of the memory;
2. Attestation, where enclaves can verify other enclaves;
3. Sealing, where the SGX processor with an already defined Root Seal Key encrypts the data that is stored in untrusted memories.

3.4.2 *System Model*

The system model presented here is based on the model introduced in [34]. Figure 3.2 presents the main entities and their direct communication flow. Below we present an overview of the leading entities of the system and the most relevant functions of each component.

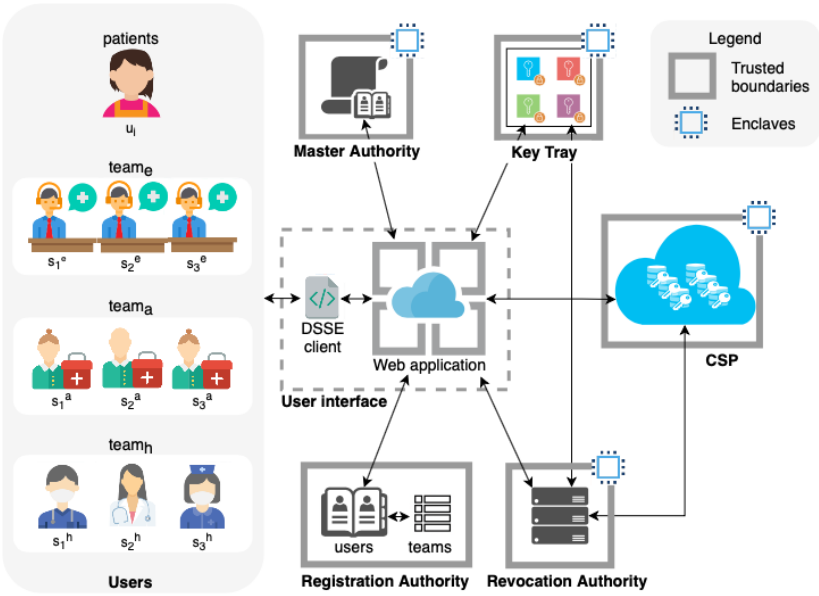


Figure 3.2: System Model showing main components: Cloud Service Provider (CSP) Registration Authority (RA), Master Authority (MS), Key Tray (KT), Revocation Authority (REV) and Users.

USER INTERFACE The users interact with the system through a user interface composed by a secure web application and a DSSE client used to encrypt and decrypt the data locally.

CLOUD SERVICE PROVIDER (CSP) We consider a cloud computing environment similar to the one described in [35]. The CSP is responsible for storing EMR's encrypted under the DSSE scheme. It must be SGX-enabled or analogous security infrastructures, since core entities need to run in a trusted execution environment.

REGISTRATION AUTHORITY (RA) The RA is responsible for the registration, authentication and authorisation of all users and acute care teams. It can run as an independent third party, but it can also be part of the CSP. The RA generates user identifiers and attributes used for the proper identification and authori-

sation. The registration process happens according to the roles carried out in the healthcare organisation where each professional works. Moreover, the RA dynamically updates the table of acute care teams based on each healthcare organisation. The organisation team admin is responsible for assigning healthcare professionals as members to each team according to their shift. Although team management is important and complex, we consider this topic as out of scope in this work.

USER We consider two different types of users: patients and healthcare professionals. All users have a unique identifier. The set of all patients registered at the RA is denoted by $\mathcal{U} = \{u_1, \ldots, u_{N_u}\}$. The set of all registered healthcare professionals is denoted as $\mathcal{S} = \{s_1, \ldots, s_{N_s}\}$. Professionals work in teams that may receive special access rights. Healthcare professionals are assigned to the acute care team dynamically on every work shift. The acute care team is composed of healthcare professionals and are identified as a team using a second identifier. The professionals receive the same team identifier while they belong to the team x, where x represents an Emergency Call Centre team (e), an Ambulance team (a) or a Hospital team (h). Each user from $\mathcal{U}, \mathcal{S}$ has a unique public/private key pair (pk/sk) used to communicate securely through an IND-CCA2 secure public-key encryption scheme PKE and an EUF-CMA secure signature scheme sign.

MASTER AUTHORITY (MS) The MS has a master secret key MSK and a public key MPK. The MPK is known to everyone and used on CPABE.Gen to encrypt user symmetric keys. Additionally, the MS uses MSK to run CPABE.Gen and generate CP-ABE secret keys for users based on their attributes, which grants them access to the ciphertexts of the symmetric keys. The MS runs in an enclave called the Master Enclave.

KEY TRAY (KT) The KT stores encrypted keys, that is, the ciphertexts of the symmetric keys generated by various users. KT stores the patient identifier next to a unique index of the patient's keys and the ciphertexts of the keys. Every authorised user can contact the KT and request access to the stored ciphertexts of the keys. KT runs in an enclave called the KT Enclave.

REVOCATION AUTHORITY (REV) REV is responsible for controlling access rights, named scopes (see section 4.4). It maintains a table that relates the patient identifier, the index of the patient's keys used to encrypt his/her data collections, the identifiers of users with access rights for each data collection and their detailed permissions. REV updates its database whenever the access

rights are granted and revoked for a user. REV is also SGX-enabled and runs in the REV Enclave.

3.4.3 *Threat Model*

Our protocol is based on the following threat model. The adversary (ADV) can interfere with the network, which means that he/she can conduct sniffing attacks [52] and man-in-the-middle attacks [53]. Furthermore, machines hosted in the CSP can be accessed by ADV with obtained privileged access rights, i.e. a corrupted administrator. We enhance adversaries' strength by assuming that ADV can load oracles into the enclaves and record the outputs. However, she/he still cannot access the protected program and data running inside the enclaves. Apart from that, we assume that an internal ADV can access the stored data but has no aim to modify them. This type of adversary refers to honest-but-curious internal entities such as the CSP administrators.

We elaborate the above threat models into the following attacks.

Attack 1 (Break-glass Attack): Let ADV be a corrupted user and u_i be a patient who has an EMR stored in the CSP. ADV has no access to u_i's EMR or the access has been revoked, and ADV has no break-glass attribute. ADV successfully launches a Break-glass Attack if she can access u_i's EMR as a break-glass case.

Attack 2 (Team Impersonation Attack): Let ADV be a corrupted user and x be an acute care team who can access a patient u_i's EMR. Assume also that ADV does not belong to x. ADV successfully launches a Team Impersonation Attack if she can impersonate a member of team x to access u_i's EMR.

Attack 3 (Team Misuse Attack): Let ADV be a corrupted user who is in an acute care team x and has access to a patient u_i's EMR, and let u_w be another user who does not belong to any acute care team. ADV successfully launches a Team Misuse Attack if she can enable u_w to access the patient's EMR.

Attack 4 (Patient Crossing Attack): Let ADV be a member of an acute care team x who can access u_i's EMR. ADV successfully launches a Patient Crossing Attack if she can access another patient u_k's EMR.

Attack 5 (Team Crossing Attack): Let ADV be a member of an acute care team x who can access a patient u_i's EMR, and let y be another acute care team that can access another patient u_k's EMR. ADV successfully launches a Team Crossing Attack if she can revoke a member of team y from accessing u_k's EMR.

Note that here we do not consider denial-of-service (DoS) attacks which can be mitigated by essential practices such as monitoring and scaling. Apart from that, side-channels attacks [54] to SGX can be prevented with the system model design. Because there is no decryption key and sensitive information stored in the enclaves, the programs running in the enclaves are data-obvious, which mitigates the side-channel attack [43]. Finally, we assume proper security measures

are adopted on the used cryptographic primitives such as Ciphertext-Policy Attribute-Based Encryption and Symmetric Searchable Encryption.

3.4.4 *MicroSCOPE Overview*

Here we summarise the protocol proposed as MicroSCOPE [43], which is based on a combination of DSSE and CP-ABE schemes to protect data and manage access control to the shared data. The data are encrypted using the symmetric key from the DSSE scheme before being sent to the CSP. The management and sharing of this key are done under the CP-ABE scheme, where the symmetric key is encrypted under policies and decrypted with the secret CP-ABE key generated if the user's attributes match the policies. Moreover, MicroSCOPE introduces an additional mechanism for defining access fine-grained rights coined 'scopes', which are controlled by the Revocation Authority REV (see section 4.2).

Scope sc_j^i is a one-dimensional array of four bits that represent the access rights (i.e., view, add, delete, revoke access) assigned to user s_j for each data collection encrypted under the symmetric key K_i. For example, if $sc_j^i = [1100]$, s_j may view and add data encrypted under K_i, but not delete or revoke access rights to them. The data owner may use multiple K_i, which can be associated with different scopes through their unique index idx_{K_i}. A data owner can define different scopes for each key, enabling fine-grained access control for each file encrypted under each key and each user. REV maintains a table of indexes idx_{K_i} and a list of users with the respectively assigned scopes. The KT and the CSP interact with REV to check the user's scopes before authorising access to keys and any processing on the encrypted data, respectively.

The MicroSCOPE protocol is composed of seven algorithms described in detail in [43]. MSCOPE.Setup initialises all the system entities, and each one generates a key pair. Moreover, MS generates a master public/private key pair for CP-ABE scheme. MSCOPE.ABEUserKey is executed by the user to receive a secret CP-ABE key. MSCOPE.Store is executed by the user to add data: first he/she generates a K_i and encrypts her/his files, and then sends them to the CSP along with the key index idx_{K_i}. MSCOPE.KeyTrayStore is executed by the user to encrypt K_i using the CP-ABE scheme, send it to the KT and to define scopes to grant other users access to the data. MSCOPE.KeyShare is executed by a user to access the shared key K_i of another user with her/his CP-ABE key. MSCOPE.Search/Update/Delete are algorithms executed by users that have the respective valid scope to process another user's encrypted files. MSCOPE.Revoke is executed by a user to revoke another user's access rights to data.

3.5 ACCESS CONTROL FOR ACUTE CARE PROTOCOL

We propose the Access Control for Acute Care Protocol (AC-AC) for the problem presented in Section 3.3. Our proposal builds upon the concepts of scopes defined by MicroSCOPE, which was briefly presented in Section 3.4.4, to meet the data access dynamics of acute stroke care.

MicroSCOPE is extended to construct a protocol for patient data sharing among acute care teams in emergency cases, leveraging data protection at the CSP by cryptographic encryption and dynamic access revocation. The application of MicroSCOPE here is not trivial because of the following practical issues that the protocol itself does not consider. Firstly, MicroSCOPE assumes that a user u_i knows in advance with whom she/he would like to share data. On the contrary, u_i does not know in advance professionals who will be involved in emergency treatment. Secondly, in MicroSCOPE u_i shares data with other individuals, which is different from emergency care practice. A patient's EMR needs to be shared with acute care teams, with members assigned dynamically by the healthcare organisation. Thirdly, in MicroSCOPE, a user u_j with valid scope values may access u_i's data until the access is revoked explicitly by another user who has valid *revoke* scope value. Differently, during acute care, the revocation must be dynamic since teams need to lose access as soon as they cease to treat the patient.

AC-AC addresses the gaps mentioned above between MicroSCOPE and our practical use case. The main focus is placed on the patient treatment timeline the changing access rights as scope values. Once being granted valid scopes, the user can access the encrypted key and then search over encrypted data, add further data, and re-share data with other users. An acute care team that already finished its tasks has the access revoked by the remaining involved teams in the patient's treatment.

Below we present an overview of the AC-AC protocol and then detail the main algorithms for the two phases: pre-emergency and emergency session.

3.5.1 *Protocol Overview*

AC-AC enables granting and revoking access to a patient's EMR to authenticated members of an acute care team involved in the treatment during the emergency session. The main difference between AC-AC and the original MicroSCOPE is the mechanism for handling scopes. In MicroSCOPE, the user must know upfront all the other users with whom to share the EMR and configure scopes for them. Instead, in AC-AC, we propose to configure scopes on the fly, as the emergency session progresses and as the teams are added or removed from the patient's treatment. This is achieved by associating scopes to

team identifiers instead of only to individual users. In this way, any healthcare professional who has been authenticated and authorised as a member of a team can process the patient EMR according to valid scope values.

During an emergency session, the scopes *view* (search/retrieve), *add* and *revoke* are granted and revoked. We do not allow *delete* for the sake of data integrity. Moreover, we assume that scopes have a default expiration time, after which the access is completely revoked.

All protocol steps adopt general security measures. The users and entities interact through messages exchange and generate a nonce (r) for every message issued, to avoid replay attacks. Upon message receipt, every user and entity verifies the freshness and integrity of the message, and they can also authenticate the sender based on the signature. If any verification fails, the receiver does not process the next step.

The protocol contains two phases: pre-emergency and emergency session.

The *pre-emergency* phase concerns actions taken before an emergency case happens to a patient. Patients and healthcare professionals register with the RA and generate their keys for communication. Patients also generate their DSSE keys to encrypt their EMR locally, before sending it to the CSP. Moreover, patients encrypt their keys using the CP-ABE scheme, which includes emergency policies with break-glass attributes and stores their encrypted keys in the KT.

An *emergency session* begins when a patient, or someone on her/his behalf, calls the emergency call centre or when the patient is taken directly to the first aid section of a hospital. Thus, call centres and hospitals acute care teams have the break-glass attribute and can start an emergency session. In such sessions, the members of the acute care teams, who are currently treating the patient, need to have access to the patient's EMR and may request another team to join the emergency session. Finally, after a team completes its task (e.g. the ambulance delivers the patient at the hospital), the next team revokes the access of a previous team. The emergency session naturally extends to include any number of teams in the process before the patient is discharged, which means the end of the session.

During the professional shifts, the healthcare organisation needs to add and remove professionals from the teams. We consider the management of acute care teams as the responsibility of the healthcare organisations [55–57], so we leave it out of the scope of this paper. Here we assume that the RA maintains updated information about the teams' composition, attributes and the members' attributes.

3.5.2 *Pre-emergency*

The pre-emergency phase combines the necessary protocol algorithms to have the patient's EMR encrypted, to retrieve the CP-ABE key, to store the EMR in the CSP and the encrypted symmetric key in KT: AC-AC.Setup, AC-AC.ABEUserKey, AC-AC.StoreData and AC-AC.StoreKey.

AC-AC.Setup We assume that each user (from $\mathcal{U}$ or $\mathcal{S}$) registers through a RA. Each patient receives a unique identifier i, and a set of attributes $\mathcal{A}_i$ is created. In the same way, each healthcare professional receives a unique identifier j and a set of attributes $\mathcal{A}_j$. The professional also receives a team identifier s_j^x, where $x \in \{e, a, h\}$ represents the acute care team of the organisations.

AC-AC.StoreData A patient u_i who wishes to make her/his EMR available for emergency treatment needs to store it in the CSP by first running MSCOPE.Store. After authentication, u_i generates her symmetric key K_i and encrypts her/his EMR under the DSSE scheme. For this end, u_i sends the resulting ciphertext c_{K_i}, the indexes that support search on ciphertext γ_i, and a unique index of the patient's symmetric keys idx_{K_i} to be stored in CSP.

AC-AC.StoreKey Now u_i needs to share her/his symmetric key with the rightful healthcare professionals. Thus, u_i encrypts K_i under the policies and stores the resulting ciphertext $c_P^{K_i}$ in the KT. These policies must match the attributes of healthcare professionals from any emergency acute care team. The AC-AC.StoreKey (*Algorithm 9*) is an adapted version of MSCOPE.KeyTrayStore. The simplification consists of omitting the communication step between u_i and REV. Because of the identification of professionals and team, scopes definition must happen during the emergency session.

Algorithm 9: AC-AC.StoreKey

1 **Input**: Patient's u_i, MPK, policy P, K_i, idx_{K_i}
2 **Output**: KT stores $c_P^{K_i}$.
1: u_i runs $c_P^{K_i} \leftarrow \mathsf{CPABE.Enc}(MPK, K_i, P)$
2: u_i sends $m_1 = \langle r_1, \mathsf{Enc}_{pk_{KT}}(r_1, u_i, idx_{K_i}), c_P^{K_i}, \sigma_i(H(r_1 \| u_i \| c_P^{K_i} \| idx_{K_i})) \rangle$ to KT
3: KT stores : $\{u_i, c_P^{K_i}, idx_{K_i}\}$

AC-AC.ABEUserKey A healthcare professional s_j^x needs her/his secret CP-ABE key to be able to decrypt the ciphertext of the symmetric key $c_P^{K_i}$ stored in the KT. To do so, s_j^x runs MSCOPE.ABEUserKey, transmitting a request along

with her/his attributes as emergency healthcare professional. After s_j^x retrieves the secret CP-ABE key, she/he can decrypt the $c_P^{K_i}$ and retrieve K_i from any patient that encrypts the key under policies that include emergency healthcare professionals attributes.

3.5.3 *Emergency Session*

AC-AC defines the exchange of messages to grant and revoke access, as well as to rightfully encrypt and decrypt the patient's EMR during an emergency session.

Figure 3.3 depicts the integration of the protocol steps into the seamless transfer process of a patient, from the phone call to a call centre team to the patient's discharge presented on Figure 3.1. The main protocol steps are explained below: AC-AC.BreakGlass, AC-AC.JoinTeam, AC-AC.RevokeTeam, AC-AC.EndSession and AC-AC.ProcessData.

AC-AC.BreakGlass (ALGORITHM 10) This algorithm starts an emergency session. An authorised healthcare professional s_j^x sends to the RA a BreakGlass request, including patient information, such as name, last name, national identifier, etc., and the time of the request. Upon reception, the RA verifies if team x has the break-glass attribute. If so, RA searches for the patient information and retrieves her/his identifier u_i. The RA then generates a message confirming the break-glass authorisation, which is encrypted under REV's public key. This message is sent to the user, who forwards it to REV. REV retrieves the index of the keys for patient u_i from KT and sets the scope values for team x to search, add, and revoke. The scope has a default expiration time that can be configured based on a parameter δ. Thus, any legitimate member of team x has access to the data of patient u_i as long as the scope values are valid.

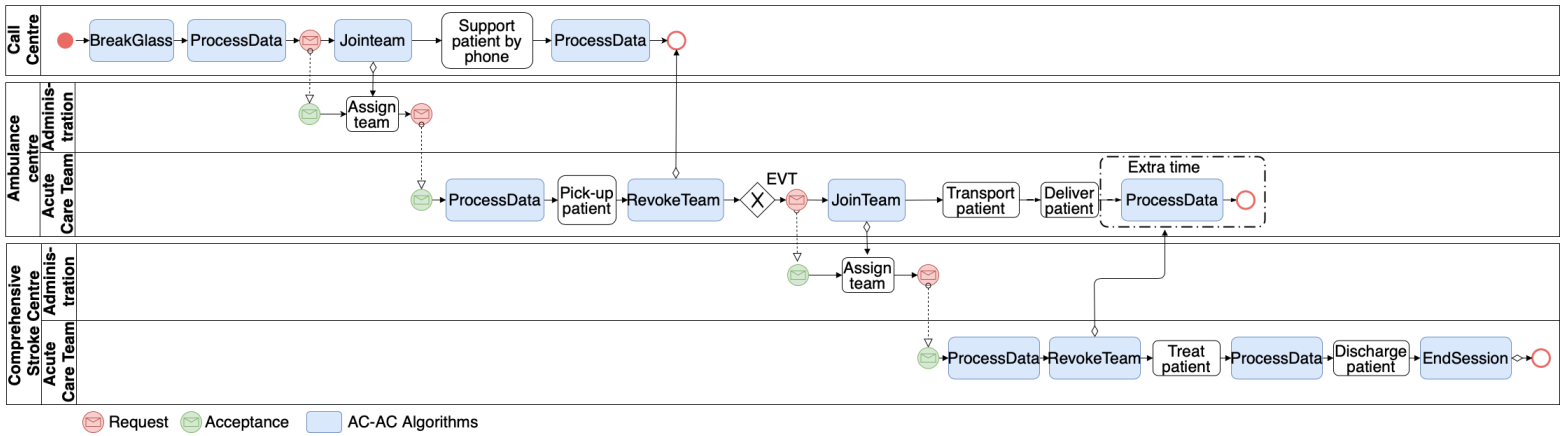


Figure 3.3: An example of interaction among the acute care teams using the AC-AC protocol during an emergency session (see figure 3.1). The steps in blue represent the proposed algorithms.

Algorithm 10: AC-AC.BreakGlass

1 **Input**: $Auth_j^x$ and patient's information pat_{info}.

2 **Output**: team x has valid scopes ($search$, add and $revoke$) to process u_i's EMR.

1: s_j^x sends m_2 to RA,
$m_2 = \langle r_2, Enc_{pk_{RA}}(Auth_j^x, pat_{info}, t_{start^x}), \sigma_j(H(Auth_j^x || pat_{info} || t_{start}))\rangle$;
2: RA checks if team x is authorized to break the glass;
3: RA retrieves patient identifier u_i corresponding to pat_{info};
4: RA sends m_3 to s_j^x,
$m_3 = \langle r_3, Auth_j^x, Enc_{pk_{REV}}(x, s_j^x, u_i, t_{start}), \sigma_{RA}(H(r_3 || x || s_j^x || u_i || t_{start^x}))\rangle$;
5: s_j^x sends m_4 to REV, $m_4 = \langle r_4, m_3, \sigma_j(H(r_4 || m_3))\rangle$;
6: REV decrypts m_3 and sends m_5 to KT,
$m_5 = \langle r_5, Enc_{pk_{KT}}(u_i), \sigma_{REV}(H(r_5 || u_i))\rangle$;
7: KT retrieves idx_{K_i} of u_i keys;
8: KT sends m_6 to REV, $m_6 = \langle r_6, Enc_{pk_{REV}}(u_i || idx_{K_i}), \sigma_{KT}(H(r_6 || u_i || idx_{K_i}))\rangle$;
9: REV sets scope $sc_x^i = [1, 1, 0, 1]$ to the idx_{K_i};
10: REV calculates the expiration time $t_{exp^x} = t_{start^x} + \delta$ to sc_x^i.

AC-AC.JoinTeam (ALGORITHM 11) This algorithm enables adding a new team y to the emergency session for patient u_i. A professional s_j^x in the acute care team x sends a JoinTeam request, including the team identifier y, the patient identifier u_i and the timestamp of the request t_{join} to REV. REV retrieves the unique key index idx_{K_i} for u_i from KT, and verifies if team x has valid scope $revoke$ to idx_{K_i} and if the timestamp to join the new team is newer then x scopes start timestamp. If true, REV also sets a valid scope value for team y, granting to any of its members the rights to search, add, and revoke access rights to u_i's EMR.

Algorithm 11: AC-AC.JoinTeam

1 **Input**: $Auth_j^x$, identifier of invited team y and patient identifier u_i.

2 **Output**: team y has valid scopes (*search*, *add* and *revoke*) to process u_i's EMR.

1: s_j^x sends m_7 to REV,
$m_7 = \langle r_7, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{REV}}}(Auth_j^x, y, u_i, t_{join^y}), \sigma_j(H(r_7 \| Auth_j^x \| y \| u_i \| t_{join^y}))\rangle$;
2: REV sends m_8 to KT, $m_8 = \langle r_8, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{KT}}}(u_i), \sigma_{REV}(H(r_8 \| u_i))\rangle$;
3: KT retrieves idx_{K_i} of u_i keys
4: KT sends m_9 to REV, $m_9 = \langle r_9, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{REV}}}(u_i \| \mathsf{idx}_{K_i}), \sigma_{KT}(H(r_9 \| u_i \| \mathsf{idx}_{K_i}))\rangle$;
5: REV checks if $sc_x^i[3] = 1$, $t_{start^x} < t_{join^y}$ and $t_{now} < t_{exp^x}$. If true, REV sets scope $sc_y^i = [1, 1, 0, 1]$ to the idx_{K_i};
6: REV calculates the expiration time $t_{exp^y} = t_{join^y} + \delta$ to the scope values granted to y.

AC-AC.RevokeTeam (ALGORITHM 12) This algorithm enables removing a team from the emergency session, therefore revoking access to patient data for its members. A member s_j^x of a valid acute care team x sends a RevokeTeam request, indicating the team's identifier y to be revoked and the patient u_i. REV then retrieves the keys' index from KT and verifies if team x still has a valid scope *revoke* to idx_{K_i}. If true, the scope for team y is set to zero, disabling any further access to the patient's EMR to its members.

Algorithm 12: AC-AC.RevokeTeam

1 **Input**: $Auth_j^x$, identifier of team to be revoked y and patient identifier u_i.

2 **Output**: team y has no longer valid scope values to process u_i's EMR.

1: s_j^x sends m_{10} to REV,
$m_{10} = \langle r_{10}, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{REV}}}(Auth_j^x, y, u_i, t_{rev}), \sigma_j(H(r_{10} \| Auth_j^x \| y \| u_i \| t_{rev}))\rangle$;
2: REV sends m_{11} to KT, $m_{11} = \langle r_{11}, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{KT}}}(u_i)), \sigma_{REV}(H(r_{11} \| u_i))\rangle$;
3: KT sends m_{12} to REV,
$m_{12} = \langle r_{12}, \mathsf{Enc}_{\mathsf{pk}_{\mathsf{REV}}}(u_i, \mathsf{idx}_{K_i}), \sigma_{KT}(H(r_{12} \| u_i \| \mathsf{idx}_{K_i}))\rangle$;
4: REV checks if $sc_x^i[3] = 1$, $t_{join^y} < t_{join^x}$ and $t_{now} < t_{exp^x}$;
5: REV sets scope $sc_y^i = [0, 0, 0, 0]$ to the idx_{K_i};

AC-AC.EndSession (ALGORITHM 13) This algorithm is used to revoke access to all remaining teams that treated the patient during the emergency session. A member s_j^x of a valid acute care team x sends a EndSession request, indicating her team's identifier x to be revoked and the patient u_i. REV then retrieves the keys' index from KT and verifies the team x or any other team still

having valid scope values to idx_{K_i}. If x has the newest scope timestamp, REV set to zero all the scope values for x and to other remaining teams.

Algorithm 13: AC-AC.EndSession

1 **Input**: $Auth_j^x$, and patient identifier u_i.
2 **Output**: all remaining teams have no longer valid scopes to process u_i's EMR.
1: s_j^x sends m_{13} to REV,
$m_{13} = \langle r_{13}, Enc_{pk_{REV}}(Auth_j^x, u_i), \sigma_j(H(r_{13}||Auth_j^h||u_i))\rangle$;
2: REV sends m_{14} to KT, $m_{14} = \langle r_{14}, Enc_{pk_{KT}}(u_i), \sigma_{REV}(H(r_{14}||u_i))\rangle$;
3: KT sends m_{15} to REV,
$m_{15} = \langle r_{15}, Enc_{pk_{REV}}(u_i, idx_{K_i}), \sigma_{KT}(H(r_{15}||u_i||idx_{K_i}))\rangle$;
4: REV sets to all remain teams scope $sc^i = [0,0,0,0]$ to the idx_{K_i};

AC-AC.ProcessData (ALGORITHM 14) This algorithm combines three algorithm of MicroSCOPE: MSCOPE.KeyShare, MSCOPE.Search and MSCOPE. Update. Assuming that an acute care team, x, has valid scope values, s_j^x can search/retrieve and add data on EMR. More specifically, s_j^x retrieves from KT the encrypted key under emergency policies $c_P^{K_i}$, a message encrypted to CSP that included the x scope values and a timestamp t. Then, s_j^x generates a DSSE token according to the type of process: DSSE.Search or DSSE.AddFile. s_j^x send to CSP a messages that include the message from KT and the respective DSSE token. If token is $\tau_s(w)$, CSP retrieves a sequence of ciphertext c_f related to searched keywords w. If token is $\tau_a(f)$, CSP and add the new encrypted files under K_i and updates the index γ to u_i's EMR.

Algorithm 14: AC-AC.ProcessData

1 **Input**: $Auth_j^x$, team identifier x and patient identifier u_i.
2 **Output**: s_j^x searches or adds new data to u_i' EMR.
1: s_j^x sends m_{16} to KT;
$m_{16} = \langle r_{16}, Auth_j^x, Enc_{pk_{KT}}(x, u_i), \sigma_j(H(r_{16}||Auth_j^x||x||u_i))\rangle$;
2: KT sends m_{17} to s_j^x; $m_{17} = \langle r_{17}, Enc_{pk_{REV}}(x, idx_{K_i}), \sigma_j(H(r_{17}||x||idx_{K_i}))\rangle$;
3: s_j^x forwards m_{17} to REV;
4: REV sends m_{18} to KT; $m_{18} = \langle r_{18}, Enc_{pk_{KT}}(sc_x^i), \sigma_j(H(r_{18}||sc_x^i))\rangle$;
5: KT sends m_{19} to s_j^x; $m_{19} = \langle r_{19}, Enc_{pk_{CSP}}(x, t, sc_x^i, idx_{K_i}), c_P^{K_i}, \sigma_j(H(r_{19}$
$||x||t||sc_x^i||idx_{K_i}||c_P^{K_i}))\rangle$;
6: s_j^x generates DSSE token $= \tau_s(w)/\tau_a$;
7: s_j^x sends m_{20} to CSP; $m_{20} = \langle m_{19}, token\rangle$;
8: CSP opens m_{19} and verifies if $sc_x^i[n] = 1$, to $n = (1,2)$;
9: If token is $\tau_s(w)$, CSP sends m_{21} to s_j^x; $m_{21} = \langle r_{21}, c_f, \sigma_{CSP}(r_{21}||c_f)\rangle$;
10: If token is $\tau_a(f)$, CSP adds new c_f and updates the index γ of u_i's EMR.

3.6 SECURITY ANALYSIS

We prove the security of our protocol both through simulation-based security analysis and proof on how the protocol is resistant against the attacks defined in Section 3.4.3.

3.6.1 *Simulation-based Security*

Assume that there is a simulator S that can simulate the real protocol's algorithms. S intercepts the communication between an adversary ADV and the real protocol and then replies with the simulated outputs. We strengthen ADV by letting him/her record not only the output but also the intermediary messages created and sent by the protocol components such as m_3 (created by RA) and m_5 (created by REV). We prove that any polynomial-time adversary cannot distinguish between the real protocol and S; hence, the protocol is simulation-based secure.

Definition 2. *(Simulation-based secure) Given the protocol, we construct two experiments: a real experiment in which the protocol runs as defined, and an ideal experiment in which the simulator* S *intercepts and replies the adversary* ADV *with the simulated responses.*

Real Experiment

1. $\mathbf{EXP}^{real}(1^\lambda)$:
2. $(m_3, m_5, m_6) \leftarrow \mathsf{ADV}^{\mathsf{BreakGlass}(\mathsf{Auth}^x, \mathsf{pat}_{\mathsf{info}})}$
3. $(m_8, m_9) \leftarrow \mathsf{ADV}^{\mathsf{JoinTeam}(\mathsf{Auth}^x, y, u_i)}$
4. $(m_{11}, m_{12}) \leftarrow \mathsf{ADV}^{\mathsf{RevokeTeam}(\mathsf{Auth}^x, y, u_i)}$
5. $(m_{14}, m_{15}) \leftarrow \mathsf{ADV}^{\mathsf{EndSession}(\mathsf{Auth}^x, u_i)}$
6. *Output* b

Ideal Experiment

1. $\mathbf{EXP}^{ideal}(1^\lambda)$:
2. $(m_3, m_5, m_6) \leftarrow \mathsf{ADV}^{\mathsf{S}(\mathsf{Auth}^x, \mathsf{pat}_{\mathsf{info}})}$
3. $(m_8, m_9) \leftarrow \mathsf{ADV}^{\mathsf{S}(\mathsf{Auth}^x, y, u_i)}$
4. $(m_{11}, m_{12}) \leftarrow \mathsf{ADV}^{\mathsf{S}(\mathsf{Auth}^x, y, u_i)}$
5. $(m_{14}, m_{15}) \leftarrow \mathsf{ADV}^{\mathsf{S}(\mathsf{Auth}^x, u_i)}$
6. *Output* b'

The protocol is simulation-based secure if all polynomial time adversaries ADV *cannot distinguish the real experiment and the ideal experiment:*

$$\mathbf{EXP}^{real}(1^\lambda) \sim \mathbf{EXP}^{ideal}(1^\lambda)$$

In order to prove the protocol security, we define the algorithms used by the simulator S to replace with the real algorithms.

- $\mathsf{BreakGlass}^*(\mathsf{Auth}^x, \mathsf{pat}_{\mathsf{info}})$: When ADV makes a request, S will simulate the created messages (m_3, m_5, m_6) by random strings with the same structure

and same length as the real messages. However, in contrast to the real experiment, the ideal one will not allow any team to break-glass access to the patient's EMR.

- JoinTeam*(Auth^x, y, u_i): When ADV makes a request, S will simulate the created messages (m_8, m_9) by a random string with the same structure and same length as the real messages. However, in contrast to the real experiment, the ideal one will not join any team.

- RevokeTeam*(Auth^x, y, u_i): When ADV makes a request, S will simulate the created messages (m_{11}, m_{12}) by random strings with the same structure and same length as the real messages. However, in contrast to the real experiment, the ideal one will not revoke any team.

- EndSession*(Auth^x, u_i): When ADV makes a request, S will simulate the created messages (m_{14}, m_{15}) by random strings with the same structure and same length as the real messages. However, in contrast to the real experiment, the ideal one will not end any session.

Then we use a hybrid argument to prove that ADV cannot distinguish between the real and the ideal experiments.

Hybrid 0 The protocol runs normally.

Hybrid 1 The protocol runs as same as Hybrid 0 with only one change. We replace AC-AC.BreakGlass with AC-AC.BreakGlass*.

Hybrid 2 The protocol runs as same as Hybrid 1 with only one change. We replace AC-AC.JoinTeam with AC-AC.JoinTeam*.

Hybrid 3 The protocol runs as same as Hybrid 2 with only one change. We replace AC-AC.RevokeTeam with AC-AC.RevokeTeam*.

Hybrid 4 The protocol runs as same as Hybrid 3 with only one change. We replace AC-AC.EndSession with AC-AC.EndSession*.

The following lemmas prove the indistinguishability among the defined hybrids.

Lemma 1. *Hybrid 1 is indistinguishable from Hybrid 0.*

Proof. Given the security of the signature scheme, there is a negligible probability that ADV can forge signatures in the messages m_3, m_5, m_6. Furthermore, under the assumption of the public key cryptosystem, ADV cannot decrypt the encrypted part of the messages, then verify the signatures. Therefore, without

signature forging and verification, ADV will not be able to distinguish the real from the simulated messages, in which random strings replace random values r_3, r_5, r_6 and the signatures by the simulator S. In other words, ADV can only distinguish Hybrid 1 and Hybrid 0 with negligible probability. □

Lemma 2. *Hybrid 2 is indistinguishable from Hybrid 1.*

Proof. In the same manner as proof of Lemma 1, given the security of the signature scheme and public-key cryptosystem, ADV will not be able to distinguish the real messages m_8, m_9 and the simulated messages, in which random values r_8, r_9 and the signatures are replaced by random strings by the simulator S. Therefore, ADV can only distinguish Hybrid 2 and Hybrid 1 with negligible probability. □

Lemma 3. *Hybrid 3 is indistinguishable from Hybrid 2.*

Proof. In the same manner as proof of Lemma 1, given the security of the signature scheme and public-key cryptosystem, ADV will not be able to distinguish the real messages m_{11}, m_{12} and the simulated messages, in which random values r_{11}, r_{12} and the signatures are replaced by random strings by the simulator S. Therefore, ADV can only distinguish Hybrid 3 and Hybrid 2 with negligible probability. □

Lemma 4. *Hybrid 4 is indistinguishable from Hybrid 3.*

Proof. In the same manner as proof of Lemma 1, given the security of the signature scheme and public-key cryptosystem, ADV will not be able to distinguish the real messages m_{14}, m_{15} and the simulated messages, in which random values r_{14}, r_{15} and the signatures are replaced by random strings by the simulator S. Therefore, ADV can only distinguish Hybrid 4 and Hybrid 3 with negligible probability. □

As presented in the above lemmas, the simulator S can replace all the expected output and/or intermediary with simulated responses in such a way that ADV cannot distinguish between them. We can deduce the indistinguishably between the ideal experiment and the real experiment, which proves that our protocol is simulation-based secure. Therefore, we can summarise the protocol security in the following proposition.

Proposition 4. *Assuming that the signature scheme σ is EUF-CMA secure, and the public key cryptosystem is IND-CCA2 secure than the protocol is a simulation-based secure protocol.*

3.6.2 *Protocol security against threats*

Proposition 5. *(Break-glass Attack Soundness): Let* ADV *be a corrupted user and* u_i *be a patient whose EMR is stored in the CSP. We assume that* ADV *has no access to the EMR of* u_i *or the access has been revoked and that* ADV *has no break-glass attribute. Then* ADV *cannot successfully perform a Break-glass Attack.*

Proof. ADV launches a Break-glass Attack by sending $m_4 = \langle r_4, m_3, \sigma_j(H(r_4||m_3)\rangle$ where $m_3 = \langle r_3, Auth_j^x, Enc_{pk_{REV}}(x, s_j^x, u_i, t_{start}), \sigma_{RA}(H(r_3||x||s_j^x||u_i||t_{start^x}))\rangle$ to REV. Upon reception, REV verifies the integrity of the message. REV can recognize that message m_3 has not been created by the RA because only the RA can create its signature. Therefore, under the assumption of EUF-CMA secure digital signature scheme, ADV fails to forge a valid message and signature of RA and therefore fails to launch the Break-glass Attack. □

Proposition 6. *(Team Impersonation Attack Soundness): Let* ADV *be a corrupted user, and* x *be a team who can access the EMR of a patient* u_i*. Assuming that* ADV *does not belong to* x*, then* ADV *cannot successfully perform a Team Impersonation Attack.*

Proof. ADV launches a Team Impersonation Soundness by joining team x. However, only the administrator of the corresponding healthcare organisation is responsible for the creation and member assignment of the team x. Assume that, without the administrator, ADV cannot join team x. Therefore, ADV fails to join the team x.

Let y be the team to which ADV belongs. ADV launches a Team Impersonation Soundness by sending to REV $m_7 = \langle r_7, Enc_{pk_{REV}}(Auth_{ADV}, y, u_i, t_{join}), \sigma_j(H(r_7||Auth_{ADV}||y||u_i||t_{join^y}))\rangle$. Because anyone can request REV to JoinTeam, REV processes the request as usual. However, based on the token $Auth_{ADV}$ and the message m_9 from KT, REV can recognize that ADV does not belong to team x and cannot grant access to u_i's EMR. Therefore, REV will drop the request and ADV fails to access to patient u_i's EMR. □

Proposition 7. *(Team Misuse Attack Soundness): Let* ADV *be a member of a team* x *who can access the EMR of patient* u_i*. Let* u_w *be a user who does not belong to any team. Assuming that* ADV *belongs to* x*, then* ADV *cannot successfully perform a Team Misuse Soundness.*

Proof. ADV launches an attack by attempting to assign u_w to his/her team x. However, only the administrator of the corresponding health organisation can assign members to the team x. Assume that, without the administrator, ADV cannot assign u_w to the team x. Therefore, ADV fails to conduct the attack. □

Proposition 8. *(Patient Crossing Attack Soundness): Let* ADV *be a member of a team* x *who can access the EMR of patient* u_i*. Let* u_k *be another patient whose EMR team* x *cannot access. Then* ADV *cannot successfully perform a Patient Crossing Attack.*

Proof. ADV launches a Patient Crossing Attack by sending $m_7 = \langle r_7, Enc_{pk_{REV}}(Auth^x_{ADV}, x, u_k, t_{join}), \sigma_{ADV}(H(r_7||Auth^x_{ADV}||x||u_k||t_{join^x}))\rangle$ to REV. However, based on the token $Auth^x_{ADV}$ and the message m_9 from KT, REV can recognise that ADV does not belong to any team who can access u_k's EMR. REV will then drop the request, and ADV cannot successfully perform a Patient Crossing Attack. □

Proposition 9. *(Team Crossing Attack Soundness): Let* ADV *be a member of the team* x *who can access the EMR of patient* u_i*. Let* y *be another team who can access the EMR of another patient* u_k*. Then* ADV *cannot successfully perform a Team Crossing Attack.*

Proof. ADV launches an attack by sending $m_{10} = \langle r_{10}, Enc_{pk_{REV}}(Auth^x_{ADV}, y, u_k, t_{rev}), \sigma_j(H(r_{10}||Auth^x_{ADV}||y||u_k||t_{rev}))\rangle$ to REV. However, based on the token $Auth^x_{ADV}$ and the message m_{12} from KT, REV can recognise that ADV does not belong to any team who can access u_k's EMR. REV will drop the request, and ADV cannot successfully perform the Team Crossing Attack. □

3.7 PERFORMANCE EVALUATION

In this section, we present results of a complexity analysis and performance assessment of the AC-AC protocol. Performance depends on how the various system components are implemented, in particular regarding RA, REV and KT. RA maintains a Registration table, which contains N users registered, and a TeamMember table, which contains all the healthcare professionals related to M teams. REV maintains a Scopes table with R entries, and KT maintains a KeyIndex schedule holding K entries. Access and search operations on these tables depend on data structures, access algorithms and table sizes, which potentially affect the performance of all the algorithms. In the complexity analysis and experiments, we consider a worst-case scenario of sequential search and access to all the tables. The search steps of the algorithms are divides into user authentication, team validation, team attribute validation, patient identification, keys' index retrieval and scopes retrieval. Furthermore, we performed the messages between the healthcare professional and the two entities RA and REV.

3.7.1 *Complexity Analysis*

Table 3.2 shows the definitions, inputs and outputs of each search step underlying the AC-AC algorithms. It also contains the complexity time expressed in Big O notation [58], that describes the limiting behaviour of a function when the argument tends towards a particular value or infinity. Note that linear time complexity is observed for all steps limiting to a constant, except for testing

whether a team has the break glass attribute, which translates into a simple comparison which is O(1).

Table 3.2: Search steps and complexity.

Search steps	Complexity	Description
(A) User authentication	O(N)	Search for $Cred_j$ in a Registration table
(B) Team validation	O(M)	Search for s_j^x in a TeamMember table.
(C) Team attribute validation	O(1)	Verifies if team x has break-glass = TRUE.
(D) Patient identification	O(N)	Search pat_{info} in a Registration table.
(E) Keys' index retrieval	O(K)	Search for u_i in a KeyIndex table.
(F) Scopes retrieval	O(R)	Search for team x in a Scopes table.

Table 3.3 shows a mapping of the search steps defined in Table 3.2 and the overall time complexity to perform the respective search steps the four algorithms. We analysed that even in the worst-case scenario, the time complexity refers to linear searches in multiple tables.

Table 3.3: Mapping of the overall time complexity for the searches steps to the algorithms.

Algorithms	Search steps						Time complexity overall
	A	B	C	D	E	F	
AC-AC.BreakGlass	x	x	x	x	x	x	O(2N+M+K+R)
AC-AC.JoinTeam	x	x			x	x	O(N+M+K+R)
AC-AC.RevokeTeam	x	x			x	x	O(N+M+K+R)
AC-AC.EndSession	x	x			x	x	O(N+M+K+R)

For completeness, in the AC-AC algorithms, the token generation encryption, decryption and signature of messages are based on the 4096-bit key RSA scheme. The time complexity of the RSA core functions depends on the implementation parameters. Therefore, we consider out of the scope of the analysis all the different implementations of the RSA functions. Thus, the time complexity for the messages are not expressed as Big-Oh complexity and just as execution time.

3.7.2 *Experiments*

The motivation for experiments is to assess the overhead that AC-AC new algorithms would add to the MicroSCOPE protocol.

The results reported for MicroSCOPE [43] assess the time for executing functions running in each enclave. In [43], the experiments assume that the users

already have an authorisation token and valid scope values are previously set to search, update and revoke. Those experiments, therefore, do not consider the time for user authentication and authorisation. Nevertheless, these are essential steps that need to happen before the user interacts with the system.

In the AC-AC protocol, the scopes values are granted and revoked dynamically, and the user must be authorised as member of a team. In our experiments, we measured the time overhead of the team validation step added to the authorisation token generation. While in MicroSCOPE [43], the authorisation token only requires the user authentication step. Furthermore, setting scope values dynamically come with a performance cost. Thus, we measured the overhead added from the AC-AC.BreakGlass and AC-AC.JoinTeam that allows the user to run AC-AC.ProcessData.

In the experiment, we focus on the RA searches to process the steps, in Table 3.2: user authentication, team validation, team attribute validation, patient identification and messages. The team validation represents the execution time for the team validation and the $Auth_j^x$ token generation. The team attribute validation represents the execution time for RA to decrypt m_2, verify the signature and retrieve the team identifier x from the $Auth_j^x$ token, then, RA searched for the team and confirmed if the team has the break-glass attribute. Finally, Messages m_2, m_3, m_4, m_7, m_{10} and m_{13} represents the messages from the algorithms, generated encrypted, and signed.

We analysed that the keys' index retrieval, scopes retrieval steps and the messages exchanges between REV and KT were the same as the algorithm MSCOPE. Revoke. Therefore, we take the performance results measured for MicroSCOPE [43] for MSCOPE.Revoke that was on average $22,6$ ms, as is in our analysis, adding to all of the measured times of our experiments.

We use the Django Framework [59] to implement the RA and to measure the time overhead of the search steps. The cryptography functions were implemented based on PyJWT Library [60] for authorisation token generation and PyCrypto Library [61] with RSA 4096 bits pair key for messages encryption, decryption and signing. The source code of the simulator and experiments can be found on the GitHub AC-AC Protocol repository [62].

We populated the Registration table with 165.000 users (150.000 patients and 15.000 healthcare professionals), and the MemberTeam table with 15.000 entries of relations between healthcare professionals and the acute care teams. We measured the time for each message creation and each search step of the AC-AC.BreakGlass, AC-AC.JoinTeam, AC-AC.RevokeTeam and AC-AC.EndSession algorithms until the sending of the messages to REV. We ran the simulations 100.000 times to get the average execution time and confidence interval. The experiments were performed on a Mac Os 10.15.3 notebook with a 1.6 GHz Intel Core i5 8TH (quad-core) processor and 8 GB of RAM.

Figure 3.4 shows the execution time for each step of each algorithm of the AC-AC protocol. From the results, user authentication was the most demanding step, with average execution time of 87.39 ms with 0.02 ms of standard deviation, because this authentication step includes Django-specific procedures. Although, only the user authentication search step took, on average, 25.26 ms, the team validation step, including the $Auth_j^x$ token generation, took on average 1.81 ms with 0.04 ms of standard deviation. The team attribute validation step, on average 10.61 ms with 0.01 ms of standard deviation, and patient identification step took, on average 20.00 ms with 0.04 ms of standard deviation.

From generation through encryption and signing, m_2, m_3 m_7, m_{10} and m_{13} followed the same approach and had similar execution time results. The average time for all these messages was 9.37 ms. Finally, the execution time of m_4 was smaller (7.65 ms on average) because it just needs to be generated and signed.

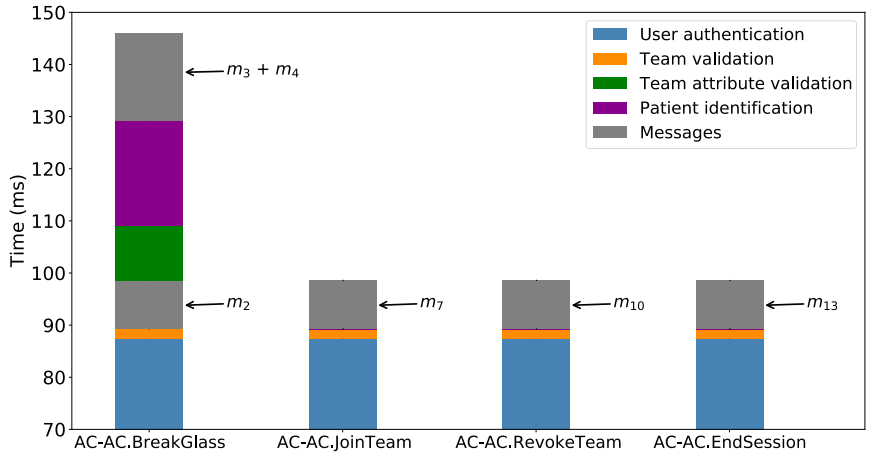


Figure 3.4: Execution time for each step, representing the RA and healthcare professional interaction within the AC-AC algorithms.

From all four algorithms, after REV receives a message m_4, m_7, m_{10} and m_{13} from the user with the patient identifier u_i, REV runs Key's index step and KT suns Scope retrieval step. These steps plus the messages between them experimental time from MSCOPE.Revoke [43] was in average 22.6 ms. Thus, we can estimate the overall overhead in time of the algorithms adding 22.6 ms to the results. Thus, the estimated overall time execution for AC-AC.BreakGlass is on average 168.7 ms and for AC-AC.JoinTeam, AC-AC.RevokeTeam and AC-AC.EndSession is on average 121.2 ms.

3.8 DISCUSSION

In the AC-AC protocol, we assumed that all patients are registered at the RA. Also, each user generates a symmetric key to encrypt and store her medical record and then shares this key to enable future access to the ciphertexts. Nevertheless, this assumption does not hold when the patient identity is unknown or unconfirmed (e.g. patient is unconscious or has no registered identifier and key). In such cases, we propose to adopt user accounts for anonymous patients (e.g. 'John Doe' or 'Joanna Doe'). These accounts are associated with unique symmetric keys that are stored on KT under the emergency policies. During the AC-AC.BreakGlass step, if the patient information is 'unknown' or does not match any of the registered patients, the RA sends back the identifier u_i of one of the 'Jonh Doe' accounts. After that, the algorithm runs in the same way. Finally, when the hospital discharges the patient, the patient can complete the registration and obtain his/her own CP-ABE key. The user then runs the AC-AC.StoreKey algorithm to re-encrypt the symmetric key used during the emergency session, now including also policies that match the patient's attributes. We plan to support these cases and investigate potential security issues in the next version of the protocol.

Furthermore, we analysed that the AC-AC protocol uses the communications that already exist between the healthcare organisations involved in the stroke acute care. The current treating team requests the next organisation to assign another acute care team to participate in the patient's emergency session. After receiving the next team acceptance and identifier, the current treating team can then run the AC-AC.JoinTeam algorithm. The dependency on an different communication protocol can compromise the actual use and implementation. As future work, we will investigate ways to automate this communication step and include it in the next version of AC-AC protocol.

From the performance evaluation, we concluded that the overhead of the team validation step during authorisation is not significant and on average, takes 1.8 ms. Moreover, we analysed the time complexity and the execution time with the messages of the algorithm's steps. Even though the AC-AC.BreakGlass algorithm demanded the most processing time, the execution time for all the algorithms was in the order of milliseconds, which should not affect user experience to process the EMR.

3.9 CONCLUSION

In this paper, we proposed the AC-AC protocol, a dynamic revocable access control protocol to enable acute care teams to access patients' EMR during an emergency session. AC-AC is a hybrid encryption protocol based on Dynamic

index-based Symmetric Searchable Encryption and Ciphertext-Policy Attribute-Based Encryption. Our protocol extends the MicroSCOPE protocol [43] by using scope values to enable granting and revoking access rights to the healthcare professionals who are members of an acute care team. AC-AC algorithms allow the team that is treating the patient to add a new team to the emergency session according to the acute care timeline. AC-AC also allows a team to revoke the access right of another team that has already completed its task. Finally, we proved that the AC-AC protocol is resilient to multiple attacks. Moreover, we analysed that the time complexity of the search steps of the algorithms is linear to the database size. Furthermore, the expected execution time of the AC-AC algorithms used during an emergency session is feasible in an acute care timeline. AC-AC.BreakGlass execution times is the most extended algorithm of the protocol, and it is below 170 ms. Moreover, it only needs to happen once per emergency session. The other algorithms are even faster and should not affect access to EMR or delay revocation. Therefore, the AC-AC protocol enables the patient's EMR availability for the acute care teams before the treatment begins, which can potentially improve patient care without compromising security and patient privacy.

4

AC-ABAC: ATTRIBUTE-BASED ACCESS CONTROL FOR ELECTRONIC MEDICAL RECORDS DURING ACUTE CARE

Acute care demands fast response and procedures from the healthcare professionals involved in the emergency. The availability of electronic medical records (EMR) enables acute care teams to access patient data promptly, which is critical for proper treatment. The EMR contains sensitive data, so proper access control is a necessity. However, acute care situations entail the introduction of dynamic authorisation techniques that are able to swiftly grant access to the acute care teams during the treatment and that at the same time can revoke it as soon as the treatment is over. In this work, our contributions are threefold. First, we propose a step-by-step methodology that defines dynamic and fine-grained access control in acute care incidents. Then, we applied this methodology with the Amsterdam University Medical Center acute stroke care teams, resulting in a new model coined 'Acute Care Attribute-Based Access Control (AC-ABAC)'. AC-ABAC implements access control policies that take into account contextual attributes for dynamically sharing patient data with the appropriate healthcare professionals during the life cycle of acute care. Finally, we evaluate the performance and show the feasibility and correctness of AC-ABAC through a prototype implementation of the model and simulation of patient data requests in various scenarios. The results show that the most complex policy evaluation takes on average 194.89 ms, which is considered worthwhile when compared to the added value to the system's security and the acute care process.

This Chapter is based on:

Marcela Tuler de Oliveira*, Yiannis Verginadis*, Lúcio H. A. Reis, Evgenia Psarra, Ioannis Patiniotakis and Sílvia D. Olabarriaga *"AC-ABAC: Attribute-Based Access Control for Electronic Medical Records during Acute Care"*, accepted in *Expert Systems With Applications*, Elsevier [63].

4.1 INTRODUCTION

The protection of privacy of a patient's Electronic Medical Record (EMR) is imperative, even in emergencies. In critical access control systems, static control rules are typically applied, which usually involve the role (e.g., doctor) or even an explicit enumeration of individuals that should be allowed to access a patient's EMR. As a result, in emergencies, we witness the so-called 'break-glass' procedure [20], during which medical personnel may bypass rigid access control rules and acquire access to a patient's medical history. However, we advocate that access control under emergency conditions should be supported with the required dynamism instead of adopting a break-glass procedure. Furthermore, in many cases, parts of the patient's EMR remain unreachable even in emergencies because they are located in information systems outside the treating hospital's boundaries. Therefore, the availability of EMR across organisational boundaries has been proposed to enhance the quality of information available for decision-making during acute care [1].

Cloud storage services match the needs of remote and ubiquitous access to medical data for multiple healthcare organisations. However, security and privacy challenges still hamper the wide adoption of cloud services. Moreover, patients and healthcare organisations are afraid of losing control over the EMR when storing it on untrusted third-party clouds [14]. In May 2018, the General Data Protection Regulation (GDPR) [8] came to reinforce the need for personal data protection, defining conditions for data sharing and processing across multiple domains. Under the GDPR, the healthcare organisations, the 'data controllers', have accountability for fulfilling these regulatory requirements. The accountability relies on their ability to demonstrate that appropriate procedural security measures are being applied and, most importantly, compliant with the GDPR. When a single cloud-based EMR system is used, the GDPR classifies healthcare organisations as joint data controllers. These jointly determine 'why' and 'how' personal data should be processed to comply with the GDPR rules designed specifically for healthcare data processing [17]. Therefore, a cloud-based EMR system's access control mechanisms should be designed to support multi-organisation collaboration and offer accountability and auditability at individual, team and organisation levels.

The main goal of an EMR system is patient data availability; therefore, the access control must not block any rightful request for the sake of the patient's vital interest. Because of that, the access control models usually are more permissive than needed for patient treatment. This may pose threats to patient privacy [64] because the users might abuse the permissions and use the data for other purposes than treating a particular patient, for example, for curiosity's sake. Researchers have proposed using the Attribute-Based Access Control

(ABAC) model to achieve a more fine-grained access control; however, its adoption in real healthcare applications remains challenging. One reason is that the information workflow during acute care involves cross-organisation data sharing, which is complex and difficult to understand and model adequately. Consequently, the existing access control models using ABAC usually cover well only the conventional access situations (e.g. doctor appointments), leaving the acute care case less protected.

We consider that understanding the information workflow during treatment is fundamental for adequate access control modelling to protect patient privacy without compromising legitimate data availability. Therefore, one of the innovations in this paper is the methodology proposed to apply the Context-Aware ABAC model in the acute care case, which leverages fine-grained access control modelling for EMR systems. This step-by-step methodology guides understanding who the subjects are, their actions, and the resources to be protected. Then, based on the context, the developer can propose rules combining contextual attributes belonging to the subject, resource or environment. The developer must analyse all the rules to guarantee that the information needed to evaluate the rules is available. Finally, the rules can be combined as access control policies and enforced into an EMR system.

The second innovation concerns the application case, which covers a difficult access control situation in acute care. Our major contribution is to propose an access control model that will keep the patient's EMR confidential and private without compromising the data availability for the legit acute care teams. Our proposal applied the step-by-step methodology with the Amsterdam UMC acute care teams. By means of a thorough analysis guided by the methodology, we understood which rules and contextual attributes should and should not be used in the modelling. The resulting model, coined Acute Care Attribute-Based Access Control (AC-ABAC), presents the policies and contextual attributes used by acute care teams to legitimate the emergency session for a patient. The emergency session dynamically includes the teams and professionals involved in the patient treatment according to the demand, granting and revoking access to the patient's data along the treatment timeline and workflow. Moreover, we developed the AC-ABAC prototype to demonstrate the feasibility of our approach as well as to evaluate the correctness and performance of the model. All the prototype code is publicly available.

In summary, our work addresses the challenges of integrating ABAC modelling with an EMR system with the following contributions:

- We propose a step-by-step methodology to define dynamic and fine-grained access control to patient data using Context-Aware Attribute-Based Access Control.

- We present the Acute Care Attribute-Based Access Control model to keep patient data confidential and private without compromising data availability for the legit acute care teams.
- We present a prototype including a Context-Aware Attribute-Based Access Control system and the AC-ABAC model implementation. The code and the simulation results obtained with the prototype are available at [65], which can be used to reproduce, verify and expand our research.

4.2 RELATED WORK

An essential milestone in the access control field is the usage of roles (Role-based Access Control, RBAC). RBAC is an approach that separates users and their permissions regarding resources and places the data requester role at the centre [66, 67]. Various EMR systems consider that the decision to uncover patient data is affected by various factors that comprise the patient's situation, such as a regular medical appointment or an emergency treatment [68–70]. Indeed, one of these factors is the healthcare professional role of who requests the data, but it is not the only one.

Nazerian et al. [69] proposed Emergency RBAC (E-RBAC), which defines emergency roles based on the user trust level and gives access permission to patient data for who has an emergency role. Arora and Gosain [70] proposed a detector to analyse the misuse of E-RBAC through analyses of audit logs to classify users' trust level. Our proposal also considers that trust is given to all professionals in every healthcare organisation.

In 2008, Peleg et al. [71] highlighted the problems with the RBAC model used in the existing EMR systems and proposed a situation-based access control approach based on scenarios of data requests. In addition, the authors proposed a generic method of interviewing healthcare stakeholders and understanding their data needs. Our work focused on the acute care situation and proposed a structured methodology to identify the data needs and legitimate access during the emergency.

Lunardelli et al. [72] proposed an analytic hierarchical process model for solving policy conflict issues in EMR systems. They created a prototype and analysed the system performance which was using the eXtensible Access Control Markup Language (XACML). Calvillo-Arbizu et al. [73] proposed an access control mechanism based on XACML and ABAC, which conforms to ISO 13606. Furthermore, the proposed system applies an ontology for automatic reasoning to the authorisation process. In our previous work [74], we worked on an XACML-based access control system for authorising access to sensitive data persisted in cloud resources. That work was extended in this paper to en-

able further the dynamism of policies for usage in an acute healthcare situation. Moreover, in [75], we have introduced a model of concepts and properties called ASCLEPIOS Context-Aware Security Model (CASM). This model serves as the background ontology required for creating access control policies for EMRs in acute care conditions.

Abomhara et al. [76] proposed a work-based access control model that modifies the user-role assignment model through the concept of team role assigned to a treatment. Seol et al. [77] propose a cloud-based EMR model that performs attribute-based access control using XACML. The authors mentioned the possibility of emergencies and assumed that the system would decide based on this information. However, [76, 77] do not specify how to legitimate teams and professionals during emergency treatment.

Inspired by the related work, we propose a methodology to identify the contextual attributes that legitimate the emergency. These attributes can be used to dynamically grant and revoke access to patient EMR for the acute care teams involved in the emergency session. Furthermore, we develop a model to aggregate these attributes to the ABAC engine and use them to evaluate rules and policies. In Table 4.1, we position the contributions of our work in comparison to the related works mentioned in this section. Specifically, the ABAC or non-ABAC capabilities of these works are listed in the first columns. Notice that most of these related works that lack the ABAC models implement the RBAC paradigm. The second column depicts the reference implementation of the XACML standard, while the third column lists the related works that introduce or adopt a methodology that may lead to appropriate policies' modelling. The fourth column refers to whereas the integration to an EMR system is realised. Finally, the last column depicts the consideration of acute care situations in the enforced access controls. The axes of this comparison are mainly justified by the significant advantages that ABAC approaches exhibit in contrast to other traditional access control paradigms, as it has been argued before [74]. To the best of our knowledge, no previous work has proposed a dynamic access control methodology that resulted in an access control model for cross-organisation data sharing in acute care scenarios.

4.3 STAKEHOLDERS OF ELECTRONIC MEDICAL RECORDS DURING ACUTE CARE

This section introduces stakeholders of an acute care EMR system and their roles in the ABAC paradigm. We specifically consider the situation when a patient is treated in an Emergency Session (ES), covering the time window since the patient requests emergency treatment until discharge.

Table 4.1: Comparison of relevant state-of-the-art works.

Research Works	Non-ABAC	ABAC	XACML	Methodology	EMR System	Acute care
[66]	✓					
[67]	✓					
[68]	✓					✓
[69]	✓				✓	✓
[70]	✓					✓
[71]	✓			✓	✓	
[72]		✓		±[1]	✓	✓
[73]	✓		✓		✓	
[74]		✓	✓			
[75]		✓			✓	
[76]	✓				✓	
[77]		✓	✓		✓	±[2]
(this work)		✓	✓	✓	✓	✓

[1] Methodology for conflict resolution.
[2] Partially supports acute care.

We describe the acute stroke care case involving professionals from the emergency call centre, ambulance services and hospitals. The professionals with different roles are organised in teams in each organisation. The time interval in which the teams participate in the patient's ES is the team's Episode of Care (EC), according to FHIR standard concept [78]. An EC starts when a team is invited to the ES and ends when a team finishes the treatment. After that, access to the data is revoked. During the EC, the team members can read and update the patient's EMR.

We identified three recurrent scenarios in which the stakeholders interact with the patient's EMR during acute care: In the first, the patient or someone on behalf of the patient contacts the call centre and requests an ambulance that takes the patient to the hospital. In the second scenario, the patient or someone on behalf of the patient contacts the call centre, which informs to which hospital the patient needs to be taken by their own transportation means. In the third scenario, the patient goes directly to the nearest hospital, where the ES is started and can be extended to other teams if necessary. Figure 4.1 illustrates the three scenarios with episodes of care by the call centre, ambulance and hospital teams. For each team, the figure presents the starting and ending times of their involvement in the ES. It also shows when a team invites another team to join the ES. In all situations, the teams will have access to the patient's EMR from the moment of their involvement in the ES until their task in the treat-

ment is completed. Below, we describe the EC for the call centre, ambulance, and hospital teams.

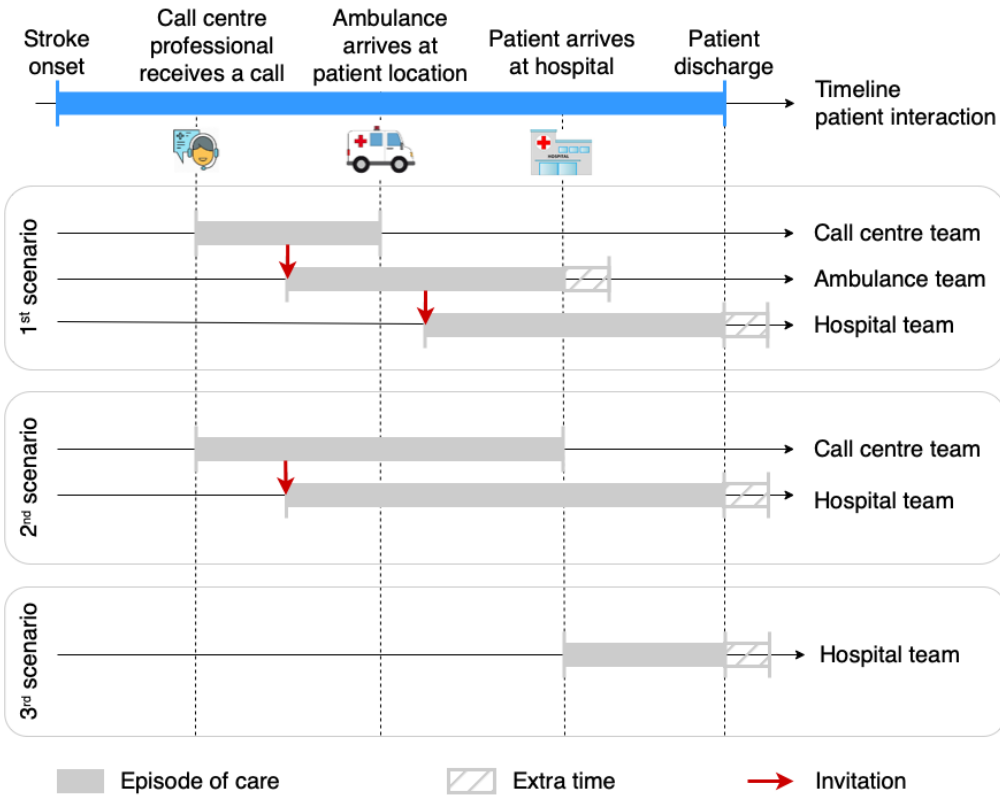


Figure 4.1: Timeline of the teams' involvement during acute care from the stroke onset until patient discharge in the three scenarios. Each team is involved during the periods marked in grey, corresponding to the Episode of Care, during which the team members have access to the patient's medical record. The hatched marks indicate extra time given for completing the report in the medical record after the involvement with the patient ends.

CALL CENTRE TEAM The phone call event is the beginning of the patient's ES in the first and second scenarios. An emergency call centre professional receives the call from the patient or someone on behalf of the patient. During the phone call, the professional follows a triage protocol and needs to read the patient's EMR and adds new information about the patient's current condition. In the first scenario, the professional decides to request an ambulance team to pick up the patient. The ambulance team that accepted the request then also becomes involved in the patient's ES. In both cases, as soon as the patient is under treatment by one of the acute care teams, the call centre professional leaves the ES and should no longer have access to the patient's EMR.

AMBULANCE TEAM Ambulance acute care team professionals must have access to a patient's EMR from the emergency request until the patient's delivery at the hospital. First, the ambulance professionals notify the EMR system that the patient was picked up. Then, following the triage, the ambulance team requests an adequate hospital to receive the patient. After the request and acceptance by the hospital, the ambulance starts the transportation. Finally, after delivering the patient to the hospital, the ambulance professionals have extra time to complete data into the patient's EMR.

HOSPITAL TEAM As soon as the hospital team is involved in the patient's ES, its members should read the patient's EMR to better prepare for the treatment. The hospital is requested by an ambulance team to receive the patient or receives a patient that comes directly to the hospital using private transportation. During the treatment, the acute care team can add new records to the patient's EMR. In the case of transfer to another hospital, a second ambulance and hospital teams become involved in the ES and access the patient's EMR. The ES and the ability to read the patient's EMR terminate when the patient is transferred or discharged. However, the team members should have extra time to complete the treatment record after the ES is over.

Note that each team member data processing actions must be recorded in the audit logs at the user level, as this creates full responsibility for the user and his actions undertaken during an emergency.

4.4 ATTRIBUTE-BASED ACCESS CONTROL ARCHITECTURE FOR ELECTRONIC MEDICAL RECORDS

ABAC defines an access control paradigm in which access rights are granted to the requester using policies consisting of logical combinations of contextual attributes. Figure 4.2 presents the main architecture entities and their direct communication flow following the ABAC model's reference implementation using the eXtensible Access Control Markup Language [79]. XACML is an OASIS [80] standard that describes both a policy language and an access control decision request/response language. Both languages use XSD [81] notations; hence, policy definition and request/response elements are serialised as XML elements. The standard defines five main components that handle access decisions, namely Policy Enforcement Point (PEP), Policy Administration Point (PAP), Policy Decision Point (PDP), Policy Information Point (PIP), and a Context Handler. In our previous work, we extended this reference implementation by providing the appropriate integration hooks to external systems to facilitate integration. We also presented a policy editing component named ASCLEPIOS

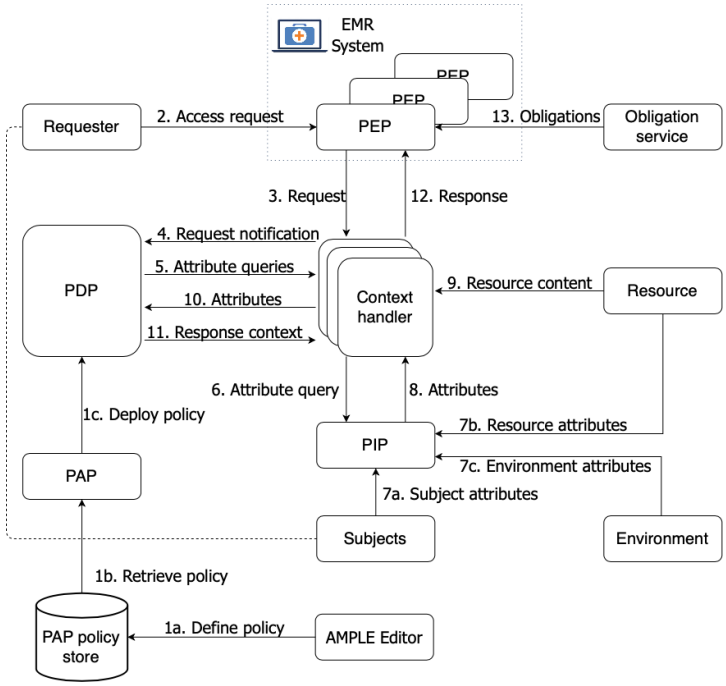


Figure 4.2: Context-Aware Attribute-based Access Control System architecture and communication flow

Models and PoLicies Editor (AMPLE) [82] for managing policies and multiple context handlers of different complexity.

Our system includes the architectural components depicted in Figure 4.2 and briefly described below.

Electronic Medical Records (EMR) system is a web system responsible for the management and storing of the encrypted EMR and respective cryptography keys. Moreover, the EMR system offers the data persistence layer and the services to process data, e.g. create, read, update and delete (CRUD), controlled through PEPs.

Subjects refer to any entity that can interact with the EMR system to request data access. A subject has one or more attributes for characterisation in the system. We consider two different types of subjects: patients and healthcare professionals.

Resource is a data, service or system component that needs to be protected through access control. Each resource offers specific actions that require data processing. A request action refers to the subjects' intended action (e.g. read or update) over a specific resource. For example, the patient data in the EMR system are represented in encrypted form as resources protected through ABAC.

Also, the respective cryptography keys could be managed by the EMR system and protected with ABAC.

Environment elements provide contextual information about the requester or resource. This information can come from the access requester, such as the timestamp, IP address and geolocation, or from external smart devices.

A *Requester* embodies the user application. The subject sends access requests through the user application to process any resource of the EMR system.

The *AMPLE editor* is a graphical web tool that allows the data controller to create, persist and update XACML-based access control policies. These policies are context-aware because they imply evaluating the contextual attributes of subjects, resources, actions and the environment before yielding a permit or deny decision.

The *Policy Administration Point (PAP)* stores a database used for persisting policies and access request decisions. The PAP provides access to the pool of defined policies that have been deployed and activated through the AMPLE editor. Nevertheless, these policies may have static or dynamic parameters that can be updated even at run-time to be immediately enforced.

The *Policy Enforcement Point (PEP)* constitutes the integration hook to any external system such as an EMR system. It manages and serves incoming access requests for processing the patient's EMR. Different PEPs may be used in various sub-components of the EMR system, or they can be appended to the application server used (e.g., Tomcat). A PEP can receive access requests and freeze the execution workflow until a decision is yielded. At the same time, it propagates the requests and attributes to the ABAC system's decision-making components.

The *Policy Decision Point (PDP)* is the core decision place for any incoming access request intercepted by a dedicated PEP. It collects all the necessary contextual information and yields an access control decision, permit or deny, according to the defined policies.

The *Policy Information Point (PIP)* is responsible for retrieving the necessary attributes for the policy evaluation from several external or internal entities. The attributes aggregated to the PIP may be retrieved from the resource to be accessed (partial or complete EMR), the environment, subjects and the intended action. The attribute values refer mainly to raw information.

Context Handlers are responsible for semantically uplifting the raw information received by a PIP and producing the appropriate level of contextual information for the policies. Thus, they enable the aggregation of dynamic attributes to context-aware policies.

The *Obligation service* is a directive from the PDP to the PEP on what must be carried out before or after the access request is approved. If the PEP is unable to comply with the directive, even the approved access request must not be

realised. The augmentation of obligations eliminates the gap between formal requirements and policy enforcement. An example of an obligation could be sending the "purpose of access" declaration for all types of access requests. If the PEP does not receive a valid value, the directive does not comply, and the access request is denied.

Note that the Context-Aware ABAC integrated with the EMR system can dynamically digest new policies at run-time. This means that, upon proper authentication, the PAP storage can be updated with new policies to be enforced, and the mapping of context handlers for inferring the context attribute values can be changed on the fly without interfering with the current policies in the system.

4.5 METHODOLOGY FOR DYNAMIC AND FINE-GRAINED ACCESS CONTROL MODEL

This section proposes a methodology that leverages a fine-grained access control mechanism to the patient's EMR based on the ABAC paradigm. Figure 4.3 depicts the methodology phases described below, namely *Preparation*, *Analysis*, *Development*, *Policies definition* and *Policies enforcement*.

In the *Preparation phase*, we prepare a template to register access control policies and the respective stakeholders for each use case scenario. The template can be found on supplemental material and involves a short description of the objectives and resources that must be protected. Moreover, it provides placeholders for expressing context-driven access control rules through its tabular format, i.e., to list the requester, action, resource, environment, logical operators that combine rules and the desired access control decision. During the interview with each stakeholder, we fill the template with all the relevant emergency procedures specified concerning the need to access the EMR. Thus, the template constitutes the base for extracting the appropriate contextual information that should bind the access control decisions (i.e., the ABAC policies).

The *Analysis phase* involves analysing the filled templates by investigating the required access control rules from the requester, the intended actions, and the resources to be accessed. The purpose is to enumerate the rules that must be used along with the contextual attributes considered per rule. Thus, a significant part of the analysis phase involves determining whether the contextual information needed for access control can be acquired or inferred from the EMR system. The selection of the appropriate contextual attributes is of critical importance for defining dynamic access control policies. In our previous work, we have defined CASM [75], an ontology that serves as a basis for creating ABAC policies. That ontology is used here to map the contextual attributes involved in access control policies. For a specific policy to be enforced on an incoming

access request, the system needs to acquire values for the contextual attributes involved, which happens through context handlers.

The *Development phase* refers to dedicated software (i.e., context handlers) that can leverage raw data to semantically enriched information. If the contextual attributes cannot be acquired or inferred, then the access control rules are revised. Any missing context handlers should be developed and enabled before using the corresponding context to access control policies.

During the *Policies definition phase*, the data controller of the EMR system defines the context-aware policies using the AMPLE policy editor. Through AMPLE, the policies are defined based on context-aware rules and then they are serialised in XACML, which is an appropriate format for enforcement.

The *Policies enforcement phase* is activated once the policies expressed in XACML are deployed in a dedicated ABAC engine. The ABAC enforcement engine retrieves all the related contextual attributes, infers the missing ones by invoking the relevant context handlers and yields a permit or deny decision according to the policies evaluation.

4.6 ATTRIBUTE-BASED ACCESS CONTROL MODELLING FOR ACUTE CARE

This section describes the Acute Care Attribute-Based Access Control (AC-ABAC) modelling resulted from applying the proposed methodology.

Furthermore, AC-ABAC follows the GDPR requirement for data confidentiality and privacy. According to Art.6 of the GDPR - "Lawfulness of processing: Processing shall be lawful only if and to the extent that at least one of the following applies: ...d) processing is necessary in order to protect the vital interests of the data subject or of another natural person". Therefore, Art. 6 imposes that data access during acute care must be granted for those professionals involved in the treatment, and only during the treatment, and then revoked when the treatment is over.

AC-ABAC protects any resource that makes the electronic medical records available for the acute care teams. For instance, the resource could be the patients' encrypted records and the respective cryptographic keys. We believe that a cryptography scheme combined with dynamic access control would provide medical systems with the confidentiality and data privacy required. The implementation of such encryption protocols is no trivial matter, which others have explained [33]. For simplicity, we kept the cryptographic part out of the scope of this paper so that we focus on the access control modelling.

AC-ABAC consists of policies and contextual attributes definitions for coping with dynamic and efficient access control needed in acute care situations. Note that policies considering professionals' roles, IP address, and secure connection are essential and have been explored in previous research [82]. Our focus here

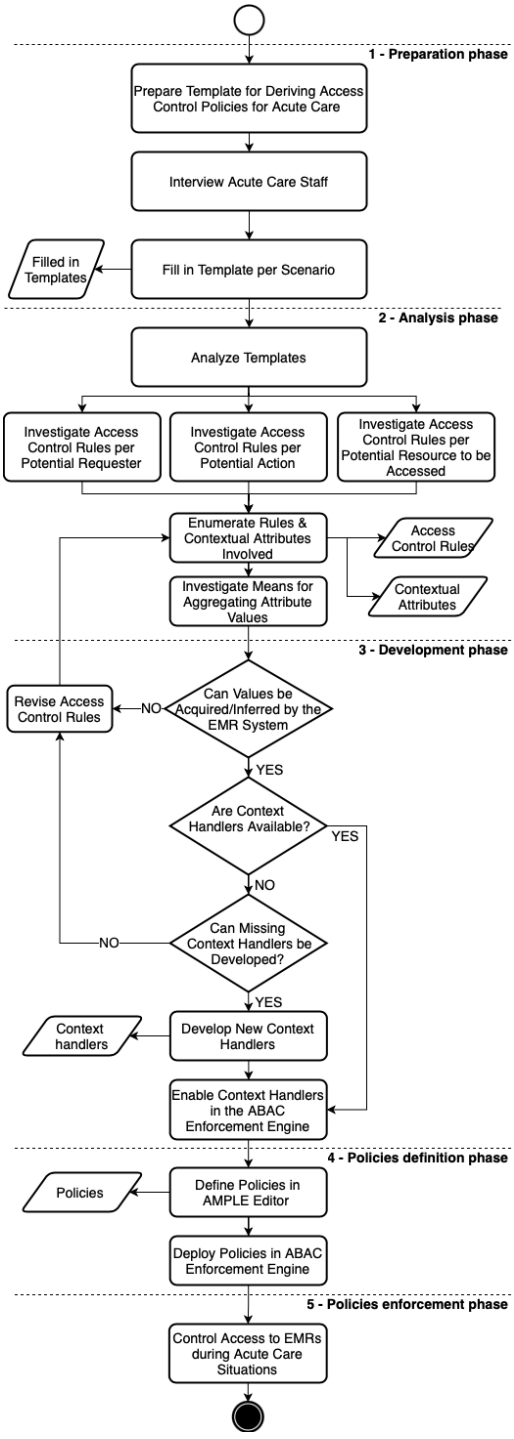


Figure 4.3: Methodology for defining context-aware ABAC policies

is to legitimate access to patient data during an emergency session for the acute care teams involved in the patient treatment. In addition, we consider the interaction between the professionals and teams from multiple organisations as an anchor of trust for access control modelling. The results are presented following each phase of the methodology (see Figure 4.3).

4.6.1 *Preparation phase*

We interviewed professionals from the Amsterdam UMC hospital that work close to the call centre and ambulance service. The template presented in section 4.5 was filled to collect information regarding *subjects*, *actions*, *resources*, and *contextual attributes* (see also [83]). For members of call centre, ambulance and hospital teams, we listed an entry in the template for reading and updating the patient EMR. Then, we determined the immutable and dynamics attributes related to each type of subject, action and resource, and the respective *expected outcome* (permit or deny).

4.6.2 *Analysis phase*

We observed that the *subjects* are the active healthcare professionals in the acute care teams involved in the patient's ES. Therefore, the *resource* should be the patient EMR, and the *actions* should be limited according to the involvement of teams during the treatment timeline. Healthcare professionals should be able to read the EMR as soon as they are involved in the ES. Still, they only should be able to write data on the EMR after they start treating the patient. Two exceptions are the call centre professional and the hospital, who interact with the patient by phone or directly in the emergency department, and can read and update data from the beginning of the call. Moreover, after the treatment is over, the professionals involved should have extra time to add new data about the recent EC. Every EC in the ES is limited by a timeout value that varies according to the acute care team type.

The combination of *contextual attributes* legitimates the patient's ES. These characterise the patient, the healthcare professionals, the acute care team involved in the ES and the duration time of each team's EC on the ES. Table 4.2 lists the contextual attributes that are dynamically assigned to professionals and acute care teams during the ES. $\text{Starter}_{\text{ID}}$ is the member of the first team who creates the ES_{ID} and associates it to the $\text{Patient}_{\text{ID}}$. Every team has a $\text{Team}_{\texttt{tag}}$, where $\texttt{tag}$ characterises the different types of teams for call centre (c), ambulance (a) and hospital (h). Every team member can request access to read or update data on the patient EMR and invite another team to participate in the ES. However, only Team_{c} and Team_{h} can start an ES, only Team_{a} and Team_{h}

can have extra time to update after revoke time, and only Team_h can discharge patient.

Table 4.2: Contextual attributes, definitions and the subject of the attribute.

Contextual attribute	Definition	Belongs to
Patient_{ID}	Identification of patient under emergency treatment.	User
User_{ID}	Healthcare professional identification.	User
Team_{ID}	Team identification within an acute care team.	Team
Team_{tag}	Team type, where $tag \in [c, a, h]$.	Team
Starter_{ID}	Identification of healthcare professional who started ES_{ID}.	ES
ES_{ID}	Emergency session identification.	ES
$t_{startshift}$	Timestamp of when the professional starts the shift.	User
$t_{endshift}$	Timestamp of when the professional ends the shift.	User
$t_{request}$	Access request timestamp.	User
t_{invite}	Invitation timestamp in EC of the Team_{ID} to attend a patient's ES.	Team
t_{treat}	Starting treatment timestamp in the EC of the Team_{ID} in the ES.	Team
t_{revoke}	Revocation timestamp in the EC of the Team_{ID} in the ES.	Team

Table 4.3 enumerates the rules and contextual attributes values involved per entity. The request for reading or updating data must contain the following attributes: $t_{request}$, requester User_{ID}, Team_{ID} and Patient_{ID}. When a healthcare professional joins a team, it is created an entry in the TeamMembers table, which contains the Team_{ID} and the User_{ID} of the professional. When an ES is started, it creates an entry in the ES table with the Patient_{ID} and the Starter_{ID}. When a team joins an ES, it is created an entry in the EC table with the Team_{ID} that is responsible of the episode and the ES_{ID} that it belongs to. Moreover, each entry on the EC table contains the $t_{request}$, t_{invite}, t_{treat} and t_{revoke} attributes describing the team participation timeline in the ES. These attributes are evaluated according to these tables and the contextual attributes defined in Table 4.2.

4.6.3 *Development phase*

Following the steps in phase 2, Figure 4.3, we investigated means to aggregate the contextual attribute values through context handlers. The context handlers must be able to dynamically either infer the attribute values from the request or acquire them from the environment. Following the methodology, we have developed the necessary context handlers to aggregate each contextual attribute from PIP since none were available. Note that user interactions with the EMR System aggregate the specific contextual attributes listed in Table 4.2. After an action is taken, the contextual attributes' values are created or updated in the

Table 4.3: Modelling rules, request's attributes and contextual attributes involved for each entity.

Entity	Rule	Description	Logical representation
Subject	R1	The healthcare professionals are working on their shifts.	$(t_{request} \geqslant t_{startshift}) \wedge (t_{request} \leqslant t_{endshift})$
	R2	The healthcare professional must be an active member of an acute care team.	$User_{ID} \in TeamMembers$
Resource	R3	Only the EMR of the patient under ES must be available to the acute care team active in the ES.	$(Patient_{ID} \in ES\ table) \wedge (Team_{ID} \in EC\ table) \wedge (ES_{ID} \in EC\ table)$
Action	R4	The acute care team has the right to read data as soon as they are involved in the emergency session.	$t_{request} \geqslant t_{invite}$
	R5	The acute care team has the right to read data until they are revoked from the emergency session.	$t_{request} \leqslant t_{revoke}$
	R6	The acute care team has the right to add data as soon as they are in the presence of the patient.	$t_{request} \geqslant t_{treat}$
	R7	The acute care team has the right to add data until a predefined extra time after the treatment.	$t_{request} \leqslant (t_{revoke} + extra)$
	R8	The healthcare professional from call centre or hospital acute care team has the right to start the ES.	$(Team_{tag} = Team_c) \vee (Team_{tag} = Team_h)$
	R9	The healthcare professional from the hospital acute care team has the right to end the ES, unless the healthcare professional was who started the ES.	$(Team_{tag} = Team_h) \wedge (User_{ID} \neq= Starter_{ID})$

PIP, often by the EMR System. Therefore, during policy evaluation, the context handlers acquire the contextual attribute values from the PIP.

The following actions trigger the changes in the PIP: When the healthcare professional starts and ends the work shift, it creates $t_{startshift}$ and $t_{endshift}$. When the organisation's administrators manage teams by adding or removing members, the PIP updates the teams, indexing with $Team_{ID}$ in the TeamMembers table. The ES attributes and the involved teams are updated when the teams act on the EMR System. For example, when $Starter_{ID}$ initiates the ES, it creates an entry on the ES table with ES_{ID}, and also associates the $Patient_{ID}$ and $Starter_{ID}$, so that all information about the ES becomes available on the PIP through the appropriate context handler. Every team that participates in the ES has an EC that starts with t_{invite}, has t_{treat} and ends with t_{revoke}. The 'previous' and 'next' teams are coined regarding the acute care timeline. For example, when the ambulance picks up the patient, it may revoke access to the previous team, which is probably the call centre.

The PIP is responsible for engaging the appropriate context handlers to aggregate the relevant contextual attributes values. This is performed based on $Patient_{ID}$, which indexes the resource EMR. From the $Patient_{ID}$, the PIP can retrieve the active patient's ES_{ID}, teams involved in the ES and their timestamps,

and members of each team. After acquiring the contextual attributes' values, the context handler sends them to the PDP for policy evaluation.

4.6.4 *Policies definition phase*

Following the steps in phase 4, Figure 4.3, we created policies based on the rules expressed in Table 4.3 to protect against non-legitimate requests for accessing the patient's EMR. Table 4.4 summarises the policies created to read and update the patient EMR and authorise the start and end of an ES. The rules combination algorithm of each policy is defined as PERMIT unless DENY, which means that if any rule yields a DENY, the policy outcome decision will be denied. Figure 4.4 represents the hierarchy of the rules on a policies decision tree.

We used AMPLE to create the rules and define the policies. Moreover, we manually defined the dynamic parameters that the context handlers use to evaluate each rule since APAM does not support this definition yet, where we create rules that both sides of the equation are parameters that the values of the contextual attributes will replace. This is obtained as follows. Consider a rule $t_{request} \geqslant X$, where X represents a dynamic value. The context handler will replace the parameter X for the contextual attribute value of some context value, for example, t_{invite}, which can be acquired from the PIP. Therefore, the dynamism of the access policy is introduced by design. The rule does not involve any static values since this can only be known and enforced at run-time.

Table 4.4: Policy is a combination of enumerated rules according to the requested action (see Table 4.3)

Action	Policy
Read	R1 $\wedge$ R2 $\wedge$ R3 $\wedge$ R4 $\wedge$ R5
Update	R1 $\wedge$ R2 $\wedge$ R3 $\wedge$ R6 $\wedge$ R7
Start ES	R1 $\wedge$ R2 $\wedge$ R8
End ES	R1 $\wedge$ R2 $\wedge$ R3 $\wedge$ R6 $\wedge$ R9

4.6.5 *Policies enforcement phase*

Following the steps in phase 5, Figure 4.3, we deployed the PDP policies and added the PEP to the EMR System. After receiving the required contextual attributes and the policy evaluation, the PDP yields a decision to the context handler: PERMIT or DENY. The context handler then notifies the PEP about the decision, and – if it is a PERMIT – the PEP allows the data access request on the EMR System.

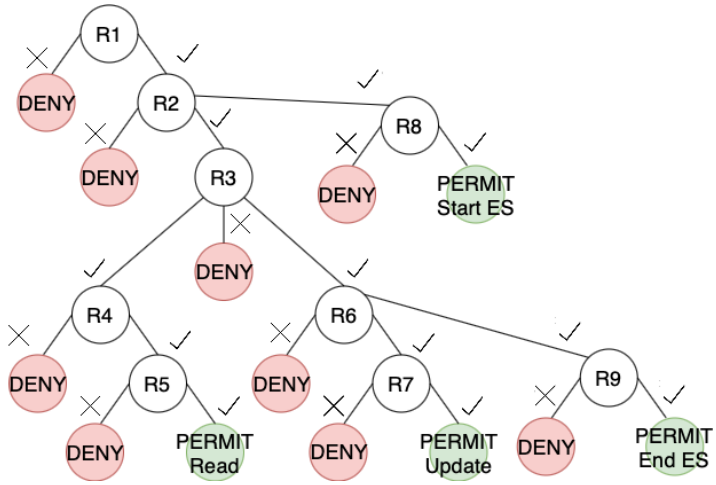


Figure 4.4: Policies decision tree

4.7 IMPLEMENTATION AND EVALUATION

In this section, we present the implementation of a prototype and the evaluation of the policies defined in section 4.6. First, we present the acute care information workflow of the AC-ABAC model. Then, through simulations, we validate the defined policies' correctness and analyse the request evaluation performance in different scenarios.

The prototype includes Contextual-Aware ABAC deployment presented in Figure 4.2, a web application simulating the EMR system to be protected and a custom database that serves as PIP with a REST-API. The EMR system simulation is a web application with available endpoints that allows read, update, start ES and end ES requests. The PIP contains the attributes values that are aggregated from the EMR System. However, in this prototype, we populated the PIP database to generate attribute values from simulated interactions between users and the EMR System under emergency. The context handler uses the REST-API to retrieve and process the contextual attributes stored on the PIP. Both web application and the PIP's REST-API were developed with the Django Framework [59].

Regarding ABAC, the open-source WSO2 Balana engine [84] was used as an implementation of the XACML access control. The context handlers were developed and connected to the PIP and enabled in the ABAC Enforcement Engine. With all ABAC components set up, the PEP is invoked whenever an incoming access request to a protected resource is detected, and the evaluation process begins. The Docker image of the prototype, the defined policies and context handlers of the AC-ABAC model, and the results of the experiments can be found on Github [65].

4.7.1 *Acute care information workflow*

Here we describe the acute care information workflow used in the AC-ABAC model. Guided by the methodology, we understood when and which information we infer during the acute care workflow. Figure 4.5 presents an ES flow where the call centre team starts the ES and invites an ambulance team, the ambulance team invites a hospital team, and the hospital team ends the ES. Each team has a starting point that represents the moment when the team enters the ES. The acute care teams interact with the ABAC engine to obtain access rights, and a team member notifies the EMR system about the events on the patient's ES. The access permissions are granted for the teams during a period of time and are updated according to the subsequent events on the patient's ES. Leave ES represents when an EC of the team ends, while an End ES represents the end of the entire ES. The use case can be extended to support more teams participating in the ES, for example, when the patient needs an ambulance transfer to a second hospital.

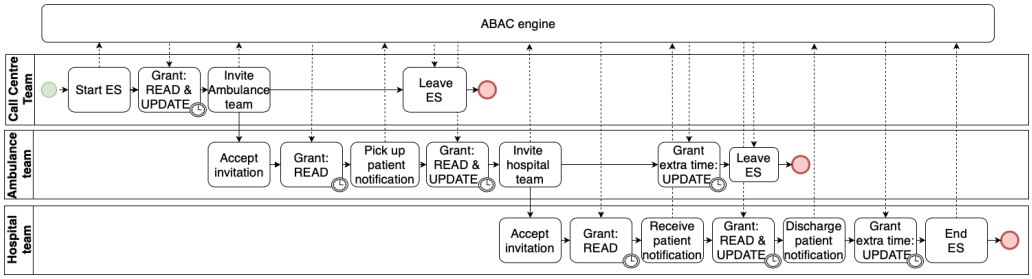


Figure 4.5: Emergency session flow presented as Business Process Model and Notation (BPMN) diagram. The teams have access rights granted to read and update according to the timeline of events.

4.7.2 *Correctness evaluation*

In this section, we present the simulation results to demonstrate the correctness of the policies defined in section 6, table 4.4. This is done by evaluating the policy implementation with a test input (i.e., access request) and validating the corresponding output (i.e., PERMIT or DENY).

We have simulated legitimate and non-legitimate requests in different scenarios in the prototype to evaluate the policies' correctness. Table 4.5 presents the description of fifteen scenarios that were evaluated (S1-15), and their expected and obtained outcomes are summarised in Table 4.6. Some scenarios may be unrealistic because we evaluated each policy's different rules and outcomes to demonstrate that the security mechanism works.

Table 4.5: Description of each scenario tested in the experiments.

Scenarios	Description
S1	A team member in the ES requests to read the EMR when $t_{invite} \leqslant t_{request} \leqslant t_{treat}$
S2	A team member in the ES requests to read the EMR when $t_{treat} \leqslant t_{request} \leqslant t_{revoke}$
S3	A team member in the ES requests to update the EMR when $t_{treat} \leqslant t_{request} \leqslant t_{revoke}$
S4	A team member in the ES requests to update the EMR when $t_{revoke} \leqslant t_{request} \leqslant (t_{revoke} + extratime)$
S5	A professional is not active on the shift, but is a team member participating in the ES
S6	A professional is active on the shift, but is not currently a team member
S7	A professional is active on the shift and is a team member, but the team is not part of the ES
S8	A professional is active on the shift, is a team member, is part of the ES, but requests to another patient's EMR.
S9	A team member in the ES requests to read the EMR when $t_{request} > t_{revoke}$
S10	A team member in the ES requests to update the EMR when $t_{request} < t_{treat}$
S11	A team member from the ES requests to update the EMR when $t_{request} > (t_{revoke} + extratime)$
S12	A team member from a call centre or hospital team requests to start an ES
S13	A team member from a ambulance team requests to start an ES
S14	A team member from the hospital team requests to end an ES and $User_{ID} \neq Starter_{ID}$
S15	A team member from the hospital team requests to end an ES and $User_{ID} = Starter_{ID}$

Scenarios 1-4 consist of must-be-permitted access requests to the patient's EMR. Scenarios 5-11 are must-be-denied access requests to the patient's EMR. Finally, scenarios 12-15 describe the requests to start and end an ES. Table 4.6 presents the policies per action as the combination of rules evaluated in each scenario. It also indicates the rules that fail in each of the must-be-denied scenarios. Table 4.6 also presents the outcomes obtained with the simulation, which corresponded to the expected values in all cases.

We acknowledge that false-positive and false-negative results might happen due to race conditions. For example, while the PIP updates the contextual attributes, the evaluation might consider outdated contextual attributes. Note, however, that this hardly occurs since the database updates in a matter of milliseconds.

4.7.3 *Performance evaluation*

Using our application simulating the EMR system, we implemented a Python script that simulates requests issued by a "Requester" through the application client to the EMR system server. The Context-Aware ABAC prototype intercepts all the requests to the EMR system server and yields a response. Here we assess the performance of the AC-ABAC model implementation regarding the time needed to evaluate an access request, evaluate the policy, and deliver the response (permit or deny).

The requests implemented the fifteen scenarios presented in Section 4.7.2. Table 4.5 and repeated 100 times. We measured the total time since the PEP received the request until it delivered the response (permit or deny) and the

Table 4.6: Policies correctness evaluation in different scenarios. The outcome represents the obtained decision after the policy evaluation.

Scenario	Rules									Action	Outcome
	R1	R2	R3	R4	R5	R6	R7	R8	R9		
S1	✓	✓	✓	✓	✓	-	-	-	-	Read	PERMIT
S2	✓	✓	✓	✓	✓	-	-	-	-	Read	PERMIT
S3	✓	✓	✓	-	-	✓	✓	-	-	Update	PERMIT
S4	✓	✓	✓	-	-	✓	✓	-	-	Update	PERMIT
S5	X	✓	✓	-	-	-	-	-	-	Read & Update	DENY
S6	✓	X	✓	-	-	-	-	-	-	Read & Update	DENY
S7	✓	✓	X	-	-	-	-	-	-	Read & Update	DENY
S8	✓	✓	X	-	-	-	-	-	-	Read & Update	DENY
S9	✓	✓	✓	✓	X	-	-	-	-	Read	DENY
S10	✓	✓	✓	-	-	X	✓	-	-	Update	DENY
S11	✓	✓	✓	-	-	✓	X	-	-	Update	DENY
S12	✓	✓	-	-	-	-	-	✓	-	Start ES	PERMIT
S13	✓	✓	-	-	-	-	-	X	-	Start ES	DENY
S14	✓	✓	✓	-	-	✓	-	-	✓	End ES	PERMIT
S15	✓	✓	✓	-	-	✓	-	-	X	End ES	DENY

time dedicated inside the PDP for the policies' evaluation. The experiments were executed in a DELL PowerEdge R630 server with Intel Xeon ES-2640 v4 Processor and 256GB of RAM at the cloud of the University of Westminster [85].

Figure 4.6 presents the total time to process the request in each scenario, as well as the portion dedicated to policy evaluation (average and standard deviation calculated over 100 runs). As expected, in grey, the time for evaluating the policy dominates the total request evaluation time. This happens because the time spent inside the PDP includes the waiting time for retrieving the policy from the PAP and the contextual attributes from the context handlers. Each scenario presents a different request evaluation time because they use different policies. Moreover, the policies are defined as "permit unless denied", and the rules are evaluated in sequence, as listed in Figure 4.4. So, when one of the rules results in a denial, it stops the execution of the subsequent rules. Note that the scenarios that evaluated the same rules presented similar average request evaluation times (see details in Table 4.6). The exceptions are scenarios S5-S8 and S10, which did not evaluate all the policy rules. In each of these scenarios, a different rule causes the request to be denied, so a different number of rules, and the necessary attributes, are evaluated - this results in different processing times. Figure 4.7 shows how the number of evaluated contextual at-

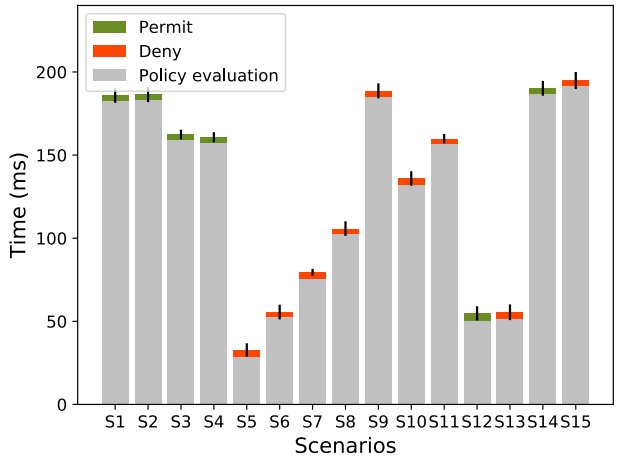


Figure 4.6: Request and policy evaluation time in each scenario (average time for 100 repetitions; error bars represent the confidence interval of 95% around the averages). Must-be-permit scenarios are in green, and must-be-deny in red, policy evaluation times are in grey.

tributes can affect policy evaluation time. Note that the policy evaluation time increases when more attributes need to be evaluated, as expected. However, the result also shows that the number of attributes is not the only factor affecting policy evaluation time. For example, although in S1 and S3, the number of evaluated attributes is the same (fifteen), policy evaluation takes longer in S1. This indicates that different contextual attributes might require different efforts to be inferred.

Regarding the results, the longest average time to evaluate a request was 194.89 ms for S14 and S15. In both scenarios, the user requests to end an ES, which is the most complex policy of our modelling because its evaluation involves the validation of five rules that combine sixteen contextual attributes. However, start and end ES requests happen only once per patient while reading and updating requests happen more frequently. The average time to evaluate a request to read was 185.82 ms in S1, and 162.36 ms a request to update in S3. We suggest that these times to process such complex requests are acceptable, particularly considering the security improvement added to the system when using more fine-grained and context-aware access control policies. This performance is also acceptable with other security-related delays that may include data encryption and decryption [86]. Finally, the obtained performance is also similar to other ABAC approaches [87]. For example, a simple WSO2 Balana-based execution of ABAC policies can reach 187 ms on average for yielding an access control decision. Although more sophisticated approaches rely on XACML and

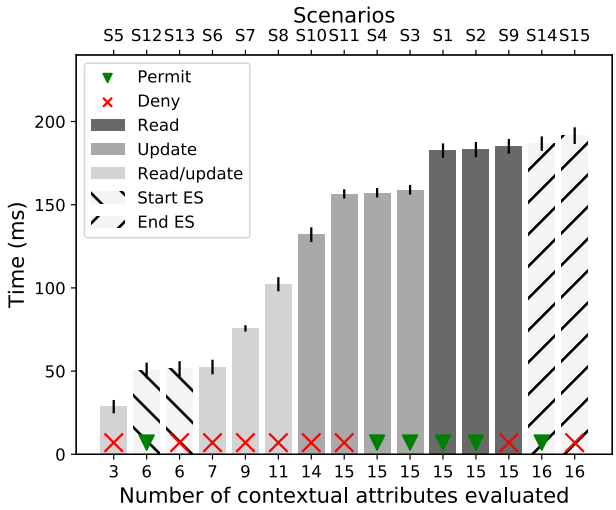


Figure 4.7: Policy evaluation time per number of attributes evaluated in each scenario (average time for 100 repetitions; error bars represent the confidence interval of 95% around the averages). The policies are differentiated per action.

semantic inferencing for yielding access control decisions, they can even exceed 8000 ms, as the policy evaluation time exponentially increases depending on the number of contextual attributes involved in the deployed ABAC rules [87].

4.8 DISCUSSION

Through the proposed methodology, we followed the steps needed to understand the access control dynamism required for this acute care application. The preparation phase of the methodology facilitated collecting the requirements and understanding the stakeholders in a structured manner imposed by the templates. In the second analysis phase, we had to make a choice of contextual attributes to be used to create the access control policies. To guarantee the availability of a patient's EMR at all times, we decided to leave out contextual attributes regarding location, such as GPS coordinates and IP addresses. The iterative approach adopted in the development phase helped refine realistic rules and contextual attributes. In the policies definition phase, we noticed that the AMPLE editor does not define rules with both parameters as contextual attributes. Therefore, to create the rules, we manually modified the XACML rules after creating them on the AMPLE editor. We plan to explore the possibility of direct creating dynamic parameters for those rules on the AMPLE editor in future work. Finally, during the policies enforcement phase, we observed that race conditions regarding outdated security tokens might occur. Such inconsis-

tencies can be minimised by regenerating tokens frequently after modifying the team composition.

Regarding patient identification, AC-ABAC model assumes that the patient is registered beforehand, so there is a $\text{Patient}_{\text{ID}}$. However, it is possible that the patient can not be found or identified during the emergency (e.g. unconscious patient). In such cases, the EMR system should give a temporary identification (e.g. 'John Doe' or 'Joanna Doe') to the patient so that during the ES, the teams can share the collected information. The proper registration or attribution of the episodes of care can be done after the ES has ended.

Furthermore, we highlight that the defined policies can dynamically change at run-time without any need to re-compile or restart the authorisation engine. For example, imagine that during the COVID-19 pandemic, there was a shortage of ambulances due to many people going to the hospital. In such a case, the paramedic teams of the military forces could provide emergency response. The proposed model could be instantly updated with a policy to add the military teams without compromising the rest of the operational policies in the system.

Regarding the experiment results, the different times found for the request evaluation in the various scenarios indicate that the access control system could be susceptible to timing attacks [88]. In a timing attack, the attacker tries to discover vulnerabilities in the security of a system by studying the variation in its response time to different input parameters. In the case of ABAC, an attacker could analyse the response time according to his attributes and discover which are correct because they lead to a longer response time. Moreover, the attacker could identify the policy by knowing which correct attributes and perform exploitation on the access control. A solution for this is simply to answer all the requests with nearly-constant time.

4.9 CONCLUSIONS

This paper presented an access control modelling that keeps patient data confidential without compromising the data availability for the legit acute care teams. Firstly, we introduced a step-by-step methodology that leveraged fine-grained access control modelling using the Context-Aware Attribute-Based Access control (ABAC) model in an EMR system. We understood which rules and contextual attributes should and should not be used to legitimate access to the patient's EMR for the acute care teams during an emergency session through an analysis guided by the proposed methodology. Secondly, we developed the Acute Care Attribute-Based Access Control (AC-ABAC) model, which has access policies and contextual attributes to enable granting and revoking access to legit healthcare professionals. Finally, we developed a prototype of the EMR system integrated with the Contextual-Aware ABAC engine to explore multi-

ple scenarios and evaluate the correctness and performance of the AC-ABAC model.

Furthermore, we implemented a prototype of AC-ABAC model in a use case. Finally, we evaluated the prototype in multiple scenarios to check the policies' correctness. In all scenarios, the outcome resulted as expected. Using the prototype, we simulate requests for various scenarios and evaluate the performance of our AC-ABAC modelling. The results show that the longest average time to process a request was 194.89 ms, which we consider reasonable when comparing ABAC with other security-related delays, including data encryption and decryption. Moreover, we suggest that the time added to the overall request process is worthwhile, considering the security added to the system. The proposed AC-ABAC model enables the patient's EMR availability for the acute care teams considering a real-case emergency scenario and the complexity of multiple team collaboration without neglecting the patient's EMR security and privacy requirements.

5

PERCEPTIONS OF A SECURE CLOUD-BASED SOLUTION FOR DATA SHARING DURING ACUTE STROKE CARE: QUALITATIVE INTERVIEW

Acute stroke care demands fast procedures through the collaboration of multiple professionals across multiple organisations. Cloud computing and the wide adoption of electronic medical records (EMR) enable healthcare systems to improve data availability and facilitate sharing among professionals. However, designing a secure and privacy-preserving EMR cloud-based application is challenging because it must dynamically control the access to the patient's EMR according to the needs for data during treatment. We developed a prototype of a secure EMR cloud-based application. This research aimed to collect impressions, challenges and improvements for the prototype when applied to the use case of secure data sharing among acute care teams during emergency treatment in the Netherlands. We performed fourteen semi-structured interviews with medical professionals with four prominent roles in acute care. We employed thematic analysis of interview transcripts. The results reinforced the current challenges for patient data sharing during acute stroke care. Moreover, from the user point of view, we expressed the challenges of adopting the prototype in a real scenario and suggestions for improving the proposed technology's acceptability. This explorative study identified several significant barriers and improvement opportunities for the future acceptance and adoption of the proposed system. Moreover, the study results highlight that the desired digital transformation should consider integrating existing systems instead of requesting migration to a new centralised system.

This Chapter is based on:

Marcela Tuler de Oliveira, Lúcio Henrik Amorim Reis, Henk Marquering, Aeilko Having Zwinderman and Sílvia D. Olabarriaga *"Perceptions of a secure cloud-based solution for data sharing during acute stroke care: qualitative interview study"*, in Journal of Medical Internet Research - JMIR Formative Research [89].

5.1 INTRODUCTION

A stroke is a medical condition that occurs when the blood supply to part of the brain is suddenly interrupted, classified as ischaemic, or when a blood vessel in the brain bursts, spilling blood into the spaces surrounding brain cells, classified as hemorrhagic [12]. Fast access to information is essential in acute stroke care. During an emergency, healthcare professionals from different organisations need to evaluate the patient's condition, identify the type of stroke and severity, decide upon the treatment, transport the patient to the adequate care centre, and perform the required intervention. Researchers have shown that the sooner the treatment is given, the better the outcomes for the patient are [13, 90]. Moreover, patient transportation at the highest priority and hospital notification before patient arrival were associated with faster stroke care and better outcomes [2]. Finally, data availability through electronic medical records (EMR) would improve decision-making and, ultimately, quality of care [1], leading to a substantial reduction of unnecessary investigations and optimised communication among the acute stroke care teams involved in the treatment.

Emergency treatment of a patient usually requires cross-organisational collaboration: professionals at the emergency call centres, ambulance services, hospitals and general practitioners' clinics. In the Netherlands, those healthcare organisations are independent and have different policies and systems for patient data sharing. However, from the first call to the emergency call centre, all the professionals involved need to exchange information while treating the patient. Currently, this information is exchanged orally or by phone, as there is no unified EMR that all professionals can share during treatment. Such conventional information-sharing methods consume time and effort, being also fault-prone. Therefore, the need for a system that enables acute care professionals to share patient data throughout the treatment process is evident, despite the organisation where the professionals work. Such data also represent valuable sources of evidence for medical research afterwards.

Cloud storage services provide an environment matching the needs for remote and ubiquitous access to the patient's EMR [2]. However, security and privacy challenges impede the wide adoption of cloud services since these are susceptible to privacy and security threats[86]. Patients and healthcare organisations are afraid of losing control over the EMR when storing it on untrusted third-party clouds [91]. And finally, besides handling the privacy and security threats in cloud environments, cloud-based EMR applications must also comply with the legal requirements regarding privacy and security imposed by the General Data Protection Regulation (GDPR) [8]. The GDPR attests that healthcare professionals and organisations are not obliged to systematically ask for patients' consent before they can use the data contained in the EMR. However,

the professionals are bound by all the principles described in Article 5 of the GDPR, which ensures the exemption from consent is proportionate and limited to what is necessary for the patient's treatment. Therefore, in the case of acute care, professionals are allowed to access the patient's EMR only through their involvement in the treatment [14], requiring a solution that can dynamically grant and revoke access to the data.

A few solutions have been proposed to improve data availability and communication among professionals during acute care. Munich et al. [92] presented a smartphone application to facilitate tracking the patient's location during an ambulance transfer between organisations. Nam et al. [93] also proposed a smartphone application based on the Cincinnati Prehospital Stroke Scale to aid self-screening and hospital decisions. However, these applications do not provide access to the patient's previous EMR.

Several studies have attempted to protect patient privacy in EMR cloud-based systems. Privacy-preserving approaches for e-Health clouds are classified as cryptographic and non-cryptographic [14]. Various cryptographic approaches have been proposed to encrypt data on the cloud [94, 95]. Seol et al. [77] proposed a combination of approaches using Attribute-Based Access Control and encrypted files to share medical records stored on the cloud. However, these works do not mention how to dynamically grant and revoke access to the encrypted data, which would be necessary to comply with GDPR fully.

Regarding dynamic access solutions, some systems offer 'break-glass access', which embodies the idea that, under certain conditions, a user can break the glass and explicitly override a denied access request [28]. Although some proposals approach break-glass access to encrypted EMR [21–23], access revocation after the emergency situation is still a problem. Thus, besides using encryption and access control to secure the data on the cloud, it is necessary to use modern techniques to address all the requirements adequately in acute care.

5.1.1 *ASCLEPIOS Acute Stroke Care Application*

ASCLEPIOS (Advanced Secure Cloud Encrypted Platform for Internationally Orchestrated Solutions in Healthcare) is a project funded by the Horizon 2020 program [96]. The project developed the ASCLEPIOS eHealth Cloud-based framework, which deploys several modern cryptographic and access control mechanisms for protecting corporate and personal sensitive data. The framework enables and facilitates the development of cloud-based eHealth applications that can protect the data subjects' privacy and prevent internal and external attacks. It combines dynamic index-based symmetric searchable encryption (DSSE) [43] and attribute-based encryption (ABE) [32] to protect data in the cloud and to enable granting and revoking access to a user without interfering

with the other users. These modern techniques allow dynamic management of encryption key access, therefore enabling more flexible access control that is important for acute care data sharing. Furthermore, the framework offers attribute-based access control (ABAC) based on flexible and configurable policies and attributes as an extra security layer to the encrypted data [82]. Only the users who behold the correct attributes can fulfil the policy and interact with the framework to access the data. Our organisation participated in the ASCLEPIOS project and implemented a demonstrator exploring the framework for the acute stroke care case.

The ASCLEPIOS Acute Stroke Care demonstrator is a secure EMR cloud-based application that leverages the ASCLEPIOS framework to share data among the acute care teams in a cross-organisational paradigm. In particular, it ensures that a team only has access to the patient's data under emergency conditions [20, 40]. It relies on a unified EMR stored in the cloud in encrypted form to improve data accessibility during an emergency. Figure 5.1 shows the EMR data model, which follows the Fast Healthcare Interoperability Resources (FHIR) standard [78]. Moreover, Figure 5.1 also shows the management entities and relations that the system uses to store necessary data, such as organisations, teams, etc. Note that the EMR is encrypted with a unique key for each patient, and healthcare professionals can get access to the key and the encrypted data only while treating this patient.

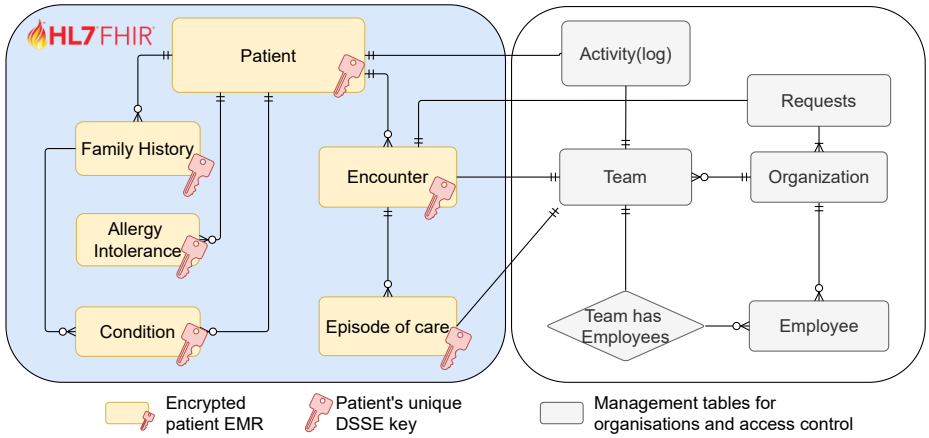


Figure 5.1: EMR data model represented as entities relations of the Acute Stroke Care demonstrator following the Fast Healthcare Interoperability Resources (FHIR) standard.

At the beginning of the project, we collected requirements from the potential stakeholders: call centre, ambulance service and hospital professionals[97]. Besides regulations and security requirements, we also asked about the current information workflow in which professionals are involved and how long they

Table 5.1: Summary of requirements of the Acute Stroke Care demonstrator. Extracted from [98].

Requirement	Description
Availability	EMR should always be available for access by legitimate users.
Confidentiality	Only authorised users should access the EMR.
Integrity	The accuracy and consistency of the EMR should be assured.
Non-repudiation	The professional cannot deny what she/he has done.
Auditability	For every action, it must be possible to know who did it, what, when, where, why and how the action occurred.

should have access to the patient's EMR. Table 5.1 summarises the requirements for the acute stroke care demonstrator.

We implemented a Web-based application to address the requirements from Table 5.1, with functionality to strengthen users' trust and comply with the GDPR. The EMR data are encrypted using a combination of DSSE to protect the data and ABE to protect and manage the DSSE keys. The implemented ABAC policies grant and revoke the access of healthcare professionals according to their participation in the patient's acute stroke care timeline and present the EMR through the professionals' user interfaces.

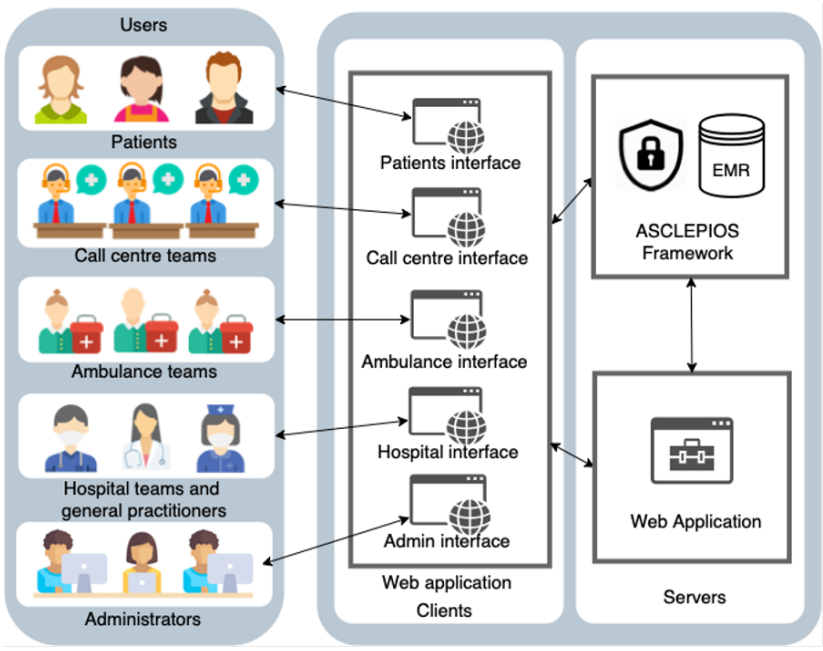


Figure 5.2: Diagram of the Acute Stroke Care demonstrator architecture with the ASCLEPIOS framework and the stakeholders involved.

Figure 5.2 shows the diagram of the architecture of the Acute Stroke Care demonstrator with the ASCLEPIOS framework and the stakeholders involved (patients and healthcare professionals). Patients and healthcare professionals have their own interface where they can interact with the system in different ways.

Figure 5.3 shows an example of the patient interface with the list of organisations that treated him in a past emergency. For each organisation, there are timestamps from when the organisation joined, started and finished acute care. For each role in each organisation, there is an interface where, during an emergency, the professionals can access the patient's EMR and request for other teams to join the emergency. Figure 5.4 shows an example of the call centre interface used to treat the patient. The call centre can input relevant information and request another team (e.g. ambulance team), and on the right side, the EMR of the patient is presented. The interfaces for the ambulance and hospitals are similar to the interface shown in Figure 5.4. More information about the application can be obtained in our previous work [98] and videos in the Multimedia Appendix.

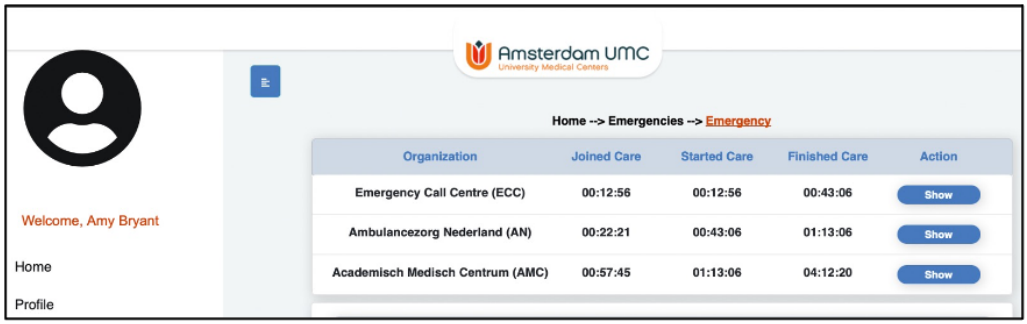


Figure 5.3: Example of the patient interface showing the organisations that treated him in some past emergency.

Figure 5.5 illustrates the information flow considered in the application during an emergency session, starting when the patient has a stroke until treatment completion at a hospital. An emergency session holds all access to the patient's EMR during acute care. The teams involved in the treatment become part of the emergency session for a period of time and leave the session when their task is completed. In this case, the patient calls the emergency call centre for help. From this moment, the call centre professional searches for the patient's identification in the EMR system and starts an emergency session for this patient. Next, the call centre professional requests an ambulance team to participate in this emergency session. After the ambulance arrives at the patient's location, the ambulance team performs triage and decides on the hospital to take the patient for treatment. Once they know which hospital to go to, the hospital team

also becomes involved in the emergency session of the patient. After arrival at the hospital, the hospital team confirms or invalidates that the patient suffers from an ischemic stroke and performs adequate treatment. The patient is finally discharged and returns home. The same procedure happens if the call centre cannot identify the patient in the system. In such a case, a temporary identification is used to store and share the patient's data during the emergency, and later the data are merged into the patient's EMR.

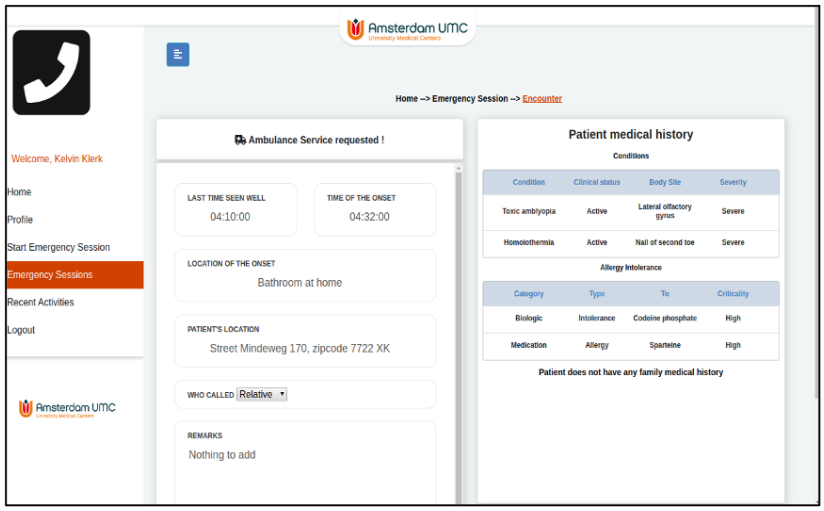


Figure 5.4: Example of call centre interface treating a patient during an emergency.

Figure 5.5 highlights that the healthcare professionals of each organisation are involved only for a limited period, and access to the patient EMR must happen only when necessary, complying with the GDPR. In an acute stroke care scenario, an involved healthcare team requests the participation of another team in the treatment, e.g. the call centre requests an ambulance to pick up the patient. Given the urgency, for adequate preparation, it would be necessary for the new team to have access to read the patient's EMR even before meeting the patient, e.g. the requested ambulance team can read the patient's history during displacement. Moreover, the teams should have extra time to add data that could not be input during the treatment. Finally, access to the EMR must be revoked for any team that no longer participates actively in the patient's treatment; for example, access by the call centre team is revoked after the ambulance team picks up the patient.

5.1.2 *Significance*

It is essential to gain user input early in technology development to improve applications according to users' needs [99]. In this study, we presented to stake-

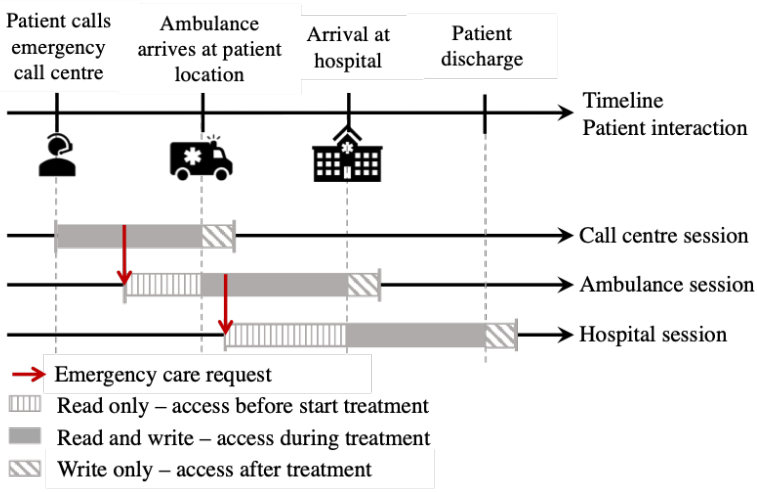


Figure 5.5: Example of an acute stroke care timeline involving multiple healthcare organisations.

holders a web application designed to facilitate patient data sharing among acute care professionals using a secure cloud solution. We also explained how this application would be used during a simulated scenario of acute stroke care. This presentation served to disseminate a new vision for secure data exchange during a medical emergency, where the data is encrypted and decrypted locally in the user's device before being sent to the cloud. Moreover, access to patients' data is granted and revoked dynamically to the professionals according to their participation in the treatment. Furthermore, this study aims to raise awareness and attract stakeholders' interest in this type of service. Finally, the stakeholders' impressions and feedback further validate the ASCLEPIOS acute stroke care application concept, providing valuable input for further technology development.

5.1.3 *Objective*

The goal of the interviews was twofold. First, the goal was to show the application usage to the main stakeholders: professionals from emergency call centres, ambulance services, emergency hospitals and general practitioners. Second, to collect their impressions of how the application would fit into their daily acute care workflow.

5.1.4 *Research questions*

With this study, we aim to answer the following questions:

RQ1. What are the current challenges for patient data sharing during acute stroke care?

RQ2. What are the participants' impressions of the proposed ASCLEPIOS Acute Stroke Care application?

RQ3. What would be the challenges and suggestions for adopting the ASCLEPIOS Acute Stroke Care application in a real-life scenario?

5.2 METHODS

5.2.1 *Overview*

We conducted an in-depth interview-based study with the main acute stroke care stakeholders. We started recruiting the participants and requesting their consent to record the interviews. The interviews were divided into three parts. First, we asked about the participants' familiarity with cybersecurity tools for data sharing in questionnaire part A. Second, we presented the ASCLEPIOS framework concepts and a simulation of the usage of the ASCLEPIOS Acute Stroke Care application during acute stroke care and by the patient. Third, we asked the participants' impressions regarding the use of the application in questionnaire part B. We tailored the in-depth interview following the given answers to the questionnaire, and the discussion evolved in light of emerging findings. We applied a qualitative thematic analysis of the data collected through the questionnaires and transcriptions of the interviews.

5.2.2 *Recruitment*

Participants were recruited from four groups, namely representatives of: (G1) emergency call centres, (G2) ambulance services, (G3) emergency hospitals and (G4) general practitioners. We started recruiting potential participants by email based on a contact person from the Amsterdam UMC. Each message introduced the project and requested an interview. Interviews were scheduled with those who replied and provided informed consent to participate. After an interview, we always asked if the participants could indicate other potential participants from the four groups. We sent a total of 16 invitations. A follow-up email was sent to non-responders one week later. When we did not get any reply, we

stopped any further contact with non-responders assuming they had no interest in participating.

The recruitment process and interview happened in three phases from September 2021 to August 2022: the first with six participants, the second with five participants and the third with three participants. We stopped recruitment when we reached thematic saturation and had a similar representation of the four main stakeholders and potential users of the application. Our study's theoretical saturation refers to the point in data collection when no additional themes or insights are identified, and data begin to repeat so that further data collection is redundant, signifying that an adequate sample size is reached [100] . During the second phase, we reached thematic saturation. In the third phase, we validated the saturation once the participants did not bring any new themes or suggestions in addition to the suggestions already put forward by participants in the previous phases.

5.2.3 *Data collection*

Two of the co-authors (MTO and LHAR) interviewed each participant individually. Eight participants were interviewed in person and three online. In general, the interviews took around 45-60 minutes. We interviewed participants from various healthcare acute care organisations in North Holland, South Holland and Utrecht, the Netherlands. During the interviews, we collected data of two types: the answers to the structured questionnaire parts A and B implemented using Google Forms [101] (see Table 1) and the audio recordings of the interviews done with a cell phone. All the data collected for demographics is stored in a private file. Table 5.2 summarises the demographic information about the interviewees.

5.2.4 *Data management*

After the interviews, we transferred the recordings via a secure virtual private network to the otter.ai service to automate the transcription process [102]. The treatment of the transcriptions was performed according to the six steps proposed by Azevedo et al. [103]. Interview transcripts, notes, and answers to the questionnaires were pseudo-anonymised using the same identifiers and divided into four groups. For example, 'Participant 1 from G1' is a professional from an emergency call centre. The audio recordings were stored in an encrypted digital audio recorder maintained in a local machine. Only the pseudonymised transcripts were shared with other co-authors. The audio recordings are retained for one year after the end of the ASCLEPIOS project (June

Table 5.2: Demographics

Variable	Count	%
Gender		
Male	9	64,3%
Female	5	35,7%
Role in acute care		
Emergency call centre professional	3	21,4%
Ambulance nurse	4	28,6%
Emergency/Neurologist physicians hospital	4	28,6%
General practitioner	4	21,4%
Year of experience in acute care		
0-4	2	14,3%
5-9	4	28,6%
10-14	1	7,1%
15-19	3	21,4%
20-25	1	7,1%
25 or more	3	21,4%
Region in the Netherlands		
North Holland	9	64,3%
Utrecht	3	21,4%
South Holland	2	14,3%

2023), and the transcripts and answers to the questionnaires will be retained for five years after the end of the project.

5.2.5 *Data Analysis*

Data were analysed following the four steps from the principles of qualitative study and systematic text condensation [104]. This procedure consists of the following steps. First, we read the transcripts and the answers from the questionnaires to get an overall impression and identify preliminary themes as responses to the research questions of this study. The preliminary themes are directly related to the questionnaires. Second, we defined the coding that represents the themes and sub-themes. Then we read all the transcripts and the answers once again and assigned the themes and subthemes to the transcripts, with the support of MAXQDA software [105]. Third, we condensed the transcripts and answers as themes and sub-themes. Finally, we synthesised the descriptions of the participants' impressions and their feedback as quotations.

5.2.6 *Ethical Considerations*

All participants were asked to give written consent based on oral and written information about the study, and only those who gave their consent were included (n=14). The study did not collect or otherwise handle patient or health-related data. All the data collected in the questionnaires through Google Forms were pseudo-anonymised and correlated to the transcripts through the timestamps. Moreover, only the authors (MTO and LHAR) have the sharing permissions to access the data in Google Forms. The ASCLEPIOS project's ethics advisory committee and data protection officer assessed the study design and informed consent forms. They concluded that a more rigorous ethical review was unnecessary because the study did not collect any sensitive or personal data.

5.3 RESULTS

total of 14 participants were interviewed. They classified their roles as professionals from the call centre (3), ambulance (4), hospital (4) and general practitioners' clinics (3). We represent the four groups to show the diversity of the participants according to their roles in acute care. In general, the interviewees were very interested in understanding the vision proposed by the application and were excited to give feedback.

We identified five themes in the data analysis, namely (T1) current challenges, (T2) quality of the shared EMR data, (T3) EMR data integrity and auditability, (T4) application usefulness and functionality, and (T5) trust and acceptance of the technology. In the analyses phase, we did not observe any significat correlation between the groups and answers, and no theme was only mentioned by a specific group. Because of that, the results are not presented per group, and we only use the groups in the citation because it gives more context to participants' quotations. See an overview of identified themes and subthemes in Textbox 1. Table 3 presents the relationship between the identified subthemes, the questions from the questionnaires parts A and B, and this study's research questions. The results presented in the following subsections use the questionnaire part and the number of the question; for example, A1 is the answer to questionnaire part A, question 1.

Table 5.3: Questions from questionnaires Part A and Part B, and how they are related to the research questions of this study and the identified sub-themes

	Questions	Research questions	Sub-themes
A	1. Do you use any electronic medical record (EMR) system to share patient data?	RQ1	T1.1; T1.2
	2. Is the EMR system cloud-based?	RQ1	T1.1;T1.2; T1.3
	3. Is the patient data encrypted in the EMR system?	RQ1	T1.1; T1.3
	4. Would you be willing to share encrypted patient data in a cloud-based solution across multiple healthcare organisations?	RQ1	T1.1; T1.3
	5. How important is it to keep the patients' data confidential andonly available to the healthcare professionals involved in their treatment?	RQ1	T1.3
	6. How much would a patient data leakage affect the patient's life?	RQ1	T1.3
B	1. How would information such as medical conditions, allergies/intolerances, and family history, as informed by the patient in the demo, be useful in case of emergency?	RQ2	T2.1
	2. How much would the availability of patient data before the treatment improve the decision-making during treatment?	RQ2	T2.2
	3. Do you believe that a digital system, such as the demo, could prevent data loss?	RQ2	T3.1;T3.2
	4. The demo considers accountable the professional, the team,and the organisation who added new data to the patient record during treatment. Who do you think should be accountable?	RQ2	T3.2
	5. Do you think that healthcare professionals should be able to add or edit the patient's data after the treatment ends?	RQ2	T3.1;T3.2; T3.3
	6. Do you think a system like this demo could be useful in a real situation?	RQ2	T3.4
	7. What would be needed to improve the usefulness of a system like the demo?	RQ3	T4.1;T4.2; T4.3
	8. Would you trust using a system like this demonstrator in your daily tasks?	RQ3	T5.1
	9. What would be needed to increase your trust in a system like this demo?	RQ3	T5.1
	10. How likely would your organisation be to accept adopting a system like this demo in a real situation?	RQ3	T5.2; T5.3
	11. What would be needed to improve your organisation's acceptability of a system like this demo?	RQ3;	T5.2; T5.3
	12. Do you think a system like this demo could make patients feel safer about providing their data to your organisation?	RQ3	T5.4

Textbox 1: Overview of themes and subthemes

- T1 Current challenges
  - T1.1 The current systems lack standardisation and structure of data
  - T1.2 Non-interoperability of systems hampers the exchange of data
  - T1.3 Achieve Professionals' awareness of security and privacy with the patients' data
- T2 Quality of data
  - T2.1 Reliability of the data provided by the patient
  - T2.2 Reliability of the data provided by other teams
- T3 Integrity and accountability
  - T3.1 Prevention of data loss
  - T3.2 Accountability of the data added and edited during the treatment
  - T3.3 Duration of the extra time to add and edit data after the end of treatment
  - T3.4 How to handle unknown patients during an acute care
- T4 Usefulness and functionality
  - T4.1 Integration of the application with other (exiting) systems as data sources
  - T4.2 Granularity of access control to parts of the EMR
  - T4.3 Information about the patient's condition after the treatment for learning purposes
- T5 Trust and acceptance of the technology
  - T5. Professionals' training to use the system
  - T5.2 Extend the system to include all types of stakeholders of an EMR system
  - T5.3 Merge current systems instead of proposing a new one
  - T.5.4 Increase patient trust and awareness

5.3.1 *Current challenges for patient data sharing during acute stroke care*

The first theme (T1) emerged when the participants answered questionnaire part A. All participants told us about how they share patient data during acute

care and their difficulties. Ten out of eleven said they use EMR systems to share patient data and feel comfortable with them (A1). One third use cloud solutions, one third do not use the cloud, and the other third do not know how the system stores the data (A2). Most participants use different systems in different organisations, and these systems usually do not communicate directly with each other (T1.1). In the Netherlands, the call centre and ambulance professionals can share data about the emergency. Still, these professionals do not have access to previous medical records, only about the ongoing acute care event. The hospitals usually do not communicate directly with the ambulance systems, and the data are generally duplicated when shared. Moreover, in North Holland, the ambulance team can print out the information collected during patient transportation and give the paper to the hospital team on arrival. A participant expressed this as follows:

> "... now we are still working in such an old fashion with paper. Even after the team types the information inside the ambulance, I will receive a paper printed out or a PDF document when I receive the patient. Then I need to manually extract what I think is relevant information and insert it into another system with 10-15 words, and this is the medical report in the patient file." [Participant of G3]

The lack of interoperability was also mentioned as a big challenge because, even if they have access to other systems, they usually cannot merge the patient data into a single EMR (T1.2). The general practitioners have to merge the records manually when following up on the patient's treatment:

> "As a GP (general practitioner), when my patient calls and I suspect that there is a stroke, I will request an ambulance, and I will receive a notification when the patient arrives at the "hospital x" and receives the treatment. But I can't see anything more. So I need to ask them for the treatment records, and I receive a PDF file again, and I need to insert the information again into the GP system. This is really annoying!" [Participant of G4]

The participants told us about their awareness of security and privacy responsibilities regarding the patients' data (T1.3). Six out of eleven do not know if their EMR system stores the patient data in encrypted form (A3). Still, all participants are willing to share encrypted patient data in a cloud-based solution across multiple healthcare organisations (A4). Also, they all agree that it is important to keep the patient data confidential and make them available only to the healthcare professionals involved in their treatment (A5). Thirteen out of fourteen believe that a patient data leakage would affect the patient's life (A6). Some of them also criticised the current data management approaches, which

usually offer break-glass buttons that bypass the conventional access control mechanism of the system to any professional that has access to the system:

> "When I need to access some data that I usually don't have access to, a "break-glass" pop-up appears, and if I click yes, I have access to the data." [Participant of G3]

5.3.2 *Participants' impressions of the proposed application*

The second theme (T2) emerged when the participants answered questionnaire part B with their impressions of the application after seeing it in use.

The application enables the patient to input some information into the system, such as medical conditions, allergies, intolerances and family history. Therefore, we asked how such information could be useful in an emergency case. Ten out of eleven believe that it would be very much useful (B1). However, all the participants commented on the doubts about sufficient quality and reliability of the information provided by the patient for acute care decision-making (T2.1):

> "Usually, when patients add medical information to their files, that is not the type of information that a doctor is looking for. For example, if patients add that they have a tumour, they cannot say the location of the tumour nor describe it as the doctor will do. Thus, the information is not that useful, but it is better than nothing." [Participant G3]

> "As a doctor, I don't think that the data the patient inputs to the system is 100% reliable. I would trust it more if another doctor had added the information." [Participant of G4]

Although all participants agree that the availability of the data before the treatment starts could improve decision-making (B2), some types of data are double-checked and input into the system again when the patient is delivered to another healthcare team, for example, when the ambulance delivers a patient at the hospital (T2.2):

> "Having access to what the teams (call centre and ambulance) added about the patient can save a lot of effort and make the treatment faster. However, suppose the patient comes from another hospital and has already done some imaging. Nowadays, the next hospital team usually remakes the images exams even if they have access to the previous exam." [Participant of G2]

The third theme (T3) emerged when we asked the participants' perspectives on how much a system like our application could prevent data loss (T3.1). In

the T1, the participants mentioned that the lack of interoperability makes them re-write essential data, and much information is lost in this process. During the interview, all mentioned that using a centralised system would prevent data loss (B3).

> "... prevent data loss? The central system on itself? Yes, absolutely." [Participant of G1]

> "... we can prevent this (data loss) when we all use one platform, and it is secure like a cloud (referring to our application)." [Participant G2]

Moreover, we asked the participants who should be accountable for the data added to the EMR of the patient when a team treats the patient (T3.2). They all agreed that the person who added the data is accountable, but six out of eleven think that the whole team should also be responsible and traceable for what happens to the patient, as proposed in the demonstrator (B4).

> "The accountability of the data is what makes the doctor remake the image exams. They do not trust that the image was made correctly in another hospital, so they need to double-check before deciding or giving a diagnostic and writing it down." [Participant of G4]

> "All the professionals who participate in the treatment should be accountable, but the professional who wrote the data must be responsible for it." [Participant of G2]

Furthermore, we asked how long the access to patient data should still be possible after the treatment is over, for example, to input data that could not be added before due to the urgency of the treatment or other responsibilities (T3.3). All participants agreed that the data should be added as soon as possible to be useful in the acute care for other teams, but they also agree that sometimes the extra time is fundamental to complete and edit all the forms (B5).

> "At the end of our shift, my colleagues and I always go back to the reports. We write any information that we haven't added because of the hurry. So, I believe 24 hours is a good extra time, more than that is too much." [Participant of G3]

> "This is a difficult question. Because when I look into my practice, sometimes it happens that we arrive at the hospital, we deliver the patient. And then they call us again, and we have cardiac arrests around the corner, then we don't have time... Of course, it is not a

> standard procedure, but this happens quite often. So, I think if the team needs extra time, they should click the button saying that they need to keep the session open until the end of their shift and close it as soon as possible." [Participant of G2]

> "Because we make mistakes when we type the information, we should be able to fix them when we have time. But I think that access after the treatment is over must be logged as editing data." [Participant of G2]

We asked if the participants thought that a system like our demonstrator would be useful in their daily tasks. They all responded that it would be useful, and eleven out of fourteen said it would be very useful (B6).

> ...the cloud solution itself will be very useful. All the (user) interfaces not, but for the cloud solution, definitely yes. [Participant of G1]

Five participants highlighted that sometimes the patient could not be rapidly identified to look for existing medical records in the system. They were very interested in the application function that enables the system to store the data generated in the treatment using the crypto scheme and later merge these data to the patient's EMR (T3.4).

> "...sometimes when there is a tourist, for example, it takes some time to find their ID or passport or whatever. So then, it would be handy to be able to merge that (patient data) afterwards." [Participant of G2]

5.3.3 *Challenges and suggestions for the adoption of the application*

The fourth theme (T4) emerged when we asked what would be needed to improve the usefulness of a system. The participants made various suggestions to enhance the usefulness and functionality of the application (B7).

The participants suggested that the application should be able to comprise other types of care, such as regular doctor appointments, which would require the admission of more types of users for the application and extend the access control model to cover their requests. At least, the system should be able to exchange data with other (exiting) systems (T4.1):

> "I think one of the things that I missed is that you can push information to your base to the local EMR system." [Participant of G4]

The participants gave feedback regarding the granularity of access control to parts of the EMR (T4.1). Five participants suggested that the system should support splitting the patient EMR into two parts; one part of data that is shared with the patient and another part of data that is shared among the healthcare professionals. These four participants believe that the patient should not read all the annotations that the healthcare professionals create. They mentioned that doctors write information about triage that needs further investigation to remember what was done before the diagnosis. According to them, such information should only be shared among the healthcare professionals involved with the treatment. They affirmed that this type of information could create misunderstanding and unnecessary stress for the patients. On the other hand, all participants agreed that patients should be able to read about the diagnosis and procedures done in the treatment.

> "Nowadays, the patients have access to part of the data. I add to the EMR only the diagnostics and measurements. I also add some notes to the patient. However, I have another place to add my comments as a doctor. For example, if a suspect that the patient has cancer, I do not add this in his report directly. First, I ask for exams, but I need to keep this note to remember the patient's case with more details." [Participant of G4]

Three other participants said that patients should be able to read all the data about their treatment and inform them as much as possible.

> "So now (in the demonstrator), the patients can see anything I type. So now, I think I will sometimes be very careful. On the other hand, if you type it down, you can also say to the patient. If you can't say it to the patient, so maybe you shouldn't write it down. If you say, if you write down the patient is maybe faking it, you should also tell the patient that you think he is faking it. So yeah, I think anything I typed down is also something I would tell the patient. Yeah. I don't know if other doctors think otherwise. This is kind of a regulation thing. I believe. The patient has some will on this." [Participante of G3]

Four participants suggested that the application should include more data sharing opportunities for learning purposes (T4.2). These participants said that they are interested in performance measurement, such as aggregated metrics about the organisations. Others are more interested to know more about what happens after they leave the patient under the care of other teams mainly to learn if their decision was correct or not.

> "...can you get aggregated metrics, for example? Because this is what we need to report, some hospitals and departments, like the entry of the emergency departments. Or, for instance, for ambulances, to report how fast they were for every patient with stroke because this is like a quality metric that we have to show to improve the quality of the service." [Participante of G3]

> "You're not a taxi when you transfer the patient in an ambulance. I believe that the professionals involved in the treatment should see what happens with the patient even after their task is done because it is part of the learning process." [Participant of G2]

In the fifth and last theme (T5), we analysed the trust and acceptability of the application among the participants and what would be the challenges regarding its adoption in a real scenario. All participants said they would 'much' and 'very much' trust using the application in their daily tasks (B8). Seven out of eleven participants highlighted the need to train healthcare professionals to use a digital system like the demonstrator (T5.1). Once the professionals understand how the system works and its security scheme, they will trust and be motivated to use it (B9).

> "...the point is that human errors happen pretty often because the professionals are not able to interact with the (current) system. When things go wrong in the hospital (system), that affects the patients negatively. Thus, the professionals must be trained to use the system correctly." [Participant of G3]

Thirteen out of fourteen participants believed their organisation would adopt a system like this (B10). To improve the acceptance by the healthcare organisations (B11), six participants suggested that our application should include more types of users beyond the acute care teams and offer opportunities for data sharing among all of them (T5.2).

> "This system should be able to comprise other types of access, so we extend the security measures that you created for the acute care to include the conventional and all the other types." [Participant of G4]

The feedback from nine out of eleven participants was to think about integrating the existing EMR system with the ASCLEPIOS framework (T5.3). They all seem to value the application, but they also reinforced that the acceptance of a new centralised national EMR system would be far-fetched. So, the recommendation was to consider using the framework as an interoperability layer between the existing systems:

> "The organisation is very sceptical about new systems, so this can be a barrier to the organisation's acceptance. But if we prove that the system works properly and if it could be interoperable with the existing system, it would help the process." [Participant of G1]

> "... if you want all the acute care workers to work in the same system, that won't be easy. But if they would work in their systems and connect all those systems with APIs or anything else we did with this cloud solution that will be there, then there is a fair chance that it can work." [Participant of G2]

Finally, all participants answered that patients would feel safer about sharing their data (B12). Still, seven participants said that most patients are not aware of the privacy risks related to EMR leakage. Because of that, two participants suggested that healthcare organisations should be more transparent about the patients' data processing and create awareness about the privacy risks (T5.4).

> "I think most of the patients are not thinking at this level. Most of the patients are not thinking about their privacy risks or if their data is available in case of an emergency. They usually think about it after something happens." [Particpant of G1]

5.4 DISCUSSION

5.4.1 *Principal findings*

This research aimed to validate the security concepts of a cloud-based medical data sharing application for acute stroke care that exploits the ASCLEPIOS framework. During the interviews with healthcare professionals, it became evident that they experience - daily - the lack of a properly connected and secure information infrastructure for patient data exchange across organisations. The application was well received and considered relevant by all. However, a large number of non-interoperating systems are used in practice, and replacing them with a new system – like the developed application – did not seem realistic to them. An alternative path to be explored involves developing an interoperation layer for a cloud-based secure and trusted data exchange that could bridge legacy systems with the newly developed technology.

Another interesting finding is that the participants were excited to give feedback when we said that we would demonstrate the usefulness of our project in a simulation to support acute stroke care. We simulated the workflow, emphasising that the professionals from each team could access the patient EMR only from the moment when they were invited to participate in treatment until their

tasks were done. Thus, they could see the added value that the proposed solution could bring to facilitate data sharing among all the professionals involved. Furthermore, the received feedback also validates the access control model implemented in the application.

Finally, we highlight three suggestions that the participant gave to increase the usefulness of the system and what we could achieve using the ASCLEPIOS framework. The first suggestion is to expand the system to support all types of access to EMRs. The second one is to create more granularity of access control for different types of data contained in the EMR, which would require separating the data that is shareable with the patient from what is shared only among the healthcare professionals. The third suggestion is about consulting aggregated metrics from all the EMRs stored for learning purposes. All these suggestions provide valuable feedback that will be explored in future works.

5.4.2 *Limitations*

One limitation of the study is that demonstrating the usage of application interfaces can be a double-edged sword. In addition to seeing how the system would work and better understanding the solution behind the screen, the participants might also be distracted by the interfaces presented during the simulation. We anticipated this effect, so we stimulated participants to give feedback beyond the user interface. Nevertheless, we still received suggestions about interface content and design modifications, which were not relevant to this study's research questions but that could be useful in a future application design.

Another limitation was the small number of professionals we could interview due to the COVID 19 pandemic. To perform the in-depth interviews, we preferred to have in-person meetings and let the participant interact with the application. However, acute care professionals are very busy, and even more so due to the pandemic, so it was even harder than anticipated to involve more professionals. Moreover, there were multiple lockdowns during the study, so we had to use online meetings to prevent cancelling the already confirmed interviews. For these online interviews, we realised that, unfortunately, the communication and the interaction were limited because they could not directly visualise the application being used. Besides this limitation, the five participants gave valuable feedback during the online meetings.

5.4.3 *Comparison with Prior Work*

Researchers have successfully adapted similar sociotechnical qualitative interviews to collect the stakeholders' perceptions and validate the concept of innovative technological solutions for healthcare. Murry et al. [106] interviewed

senior managers and medical staff to explore and understand the experiences of implementing e-health initiatives and their assessment of factors that promote the integration of e-health initiatives. Hasselgren et al. [107] interviewed medical students and analysed their perceptions of a blockchain-based decentralised work for keeping professional history and credentials portfolio. Brandt et al. [108] interviewed overweight patients to identify drivers of importance for long-term personal lifestyle changes from a patient perspective when using a collaborative e-health tool. Azode et al. [109] conducted a qualitative interview study to investigate the opportunities and challenges of using data from wearable sensor devices in healthcare. Georgiou et al. [110] also used a qualitative interview study to assess the impact of introducing new health IT initiatives for medical imaging processing. Woodward et al. [111] explored the personal experiences of healthcare professionals using e-health innovations for data sharing in selected post-conflict situations. Inspired by these works, we used similar methods, and we acknowledge the importance of gaining stakeholders' input for e-health technology development for further improvement and acceptability of new technologies.

Our previous work [97] collected and analysed the perspectives of medical staff about healthcare and data privacy requirements for the e-health cloud using a qualitative interview. At that time, we collected requirements that would guide the design of the demonstrator. Moreover, we investigated the participants' understanding of cloud services and how they envision using the ASCLEPIOS solution in their daily tasks. At that point, we did not have the acute stroke care application ready to present to the clinicians. In this study, besides validating the requirements discussed in [98], showing the participants a working application allowed them to go deeper into the matter and ask questions related to the actual usefulness and acceptance of the ASCLEPIOS solution for the cross-organisation acute stroke care data sharing.

5.5 CONCLUSIONS

This study validated the need for a cross-organisation data-sharing solution that offers the security and privacy required when patient data is processed. The participants emphasised that our cloud-based application would solve the data sharing problems, such as the duplication of data, lack of information and standardisation. Still, it would not be realistic to propose that all the organisations involved in acute care migrate to a unique cloud-based application. Future work should investigate opportunities to update the system according to these inputs and further explore the ASCLEPIOS framework as a secure and interoperable layer for patient data sharing. The concept validation and feed-

back presented in this study incite the desire for a digital transformation in healthcare systems.

6

SMARTACCESS: ATTRIBUTE-BASED ACCESS CONTROL SYSTEM FOR MEDICAL RECORDS BASED ON SMART CONTRACTS

Cross-organisation data sharing is challenging because all the involved organisations must agree on 'how' and 'why' the data is processed. Due to a lack of transparency, organisations need to trust that others comply with the agreements and regulations. We propose to exploit blockchain and smart contracts technologies to define an Attribute-Based Access Control System for cross-organisation medical records sharing, coined SmartAccess. SmartAccess offers joint agreement over access policies and dynamic access control besides blockchain transparency and auditability. We leverage the Attribute-Based Access Control model to implement smart contracts. We deploy and test them on a private and permissioned blockchain, transforming the access control process into a distributed smart contract execution. This paper proposes the SmartAccess system and its application in two healthcare use cases. We introduce the threat model and perform a security analysis of the system. To demonstrate the feasibility of our proposal, we implement a proof-of-concept of the smart contracts, written in Solidity language, with a size-efficient policy representation, and analyse the complexity and scalability of the contracts' functions. Furthermore, we present performance results, measuring the latency and throughput of the transactions to execute the access control functions with different blockchain network consensus setups. We also compare the performance of the SmartAccess system against two open-source solidity implementations of access control, Role-based Access Control (RBAC) and Access Control List (ACL). Finally, we discuss the strengths and drawbacks of our proposal. SmartAccess requires the overhead of a decentralised system, but the trade-off is transparency, regulation compliance and auditability for complex cross-organisation data sharing.

This Chapter is based on:

- **Marcela T. de Oliveira**, Lúcio H. A. Reis, Yiannis Verginadis, Diogo M. F. Mattos and Sílvia D. Olabarriaga *"SmartAccess: Attribute-Based Access Control System for Medical Records based on Smart Contracts"* [112], in IEEE Access.

6.1 INTRODUCTION

In the last decades, organisations have increasingly processed personal data; however, ensuring that data are leveraged only for legitimate purposes remains a significant challenge. Organisations must implement adequate access control mechanisms to safeguard legitimate data access when handling personal data, particularly when sharing data across organisations. Although there is plenty of data and value to be exploited, various data sharing barriers exist, including legal data protection regulations, which might differ across the various organisations.

According to the European General Data Protection Regulation (GDPR) [9], *"Personal data means any information relating to a data subject, an identified or identifiable natural person"*. Besides the data subject, the GDPR also defines the two prominent roles and responsibilities of data processing: the data controller is *"who determines the purposes for which and how personal data is processed"* and the data processor is *"who processes personal data only on behalf of the controller"* [17]. Moreover, to support cross-organisation data processing, the GDPR also defines joint controllers who *"together with one or more organisations jointly determine 'why' and 'how' personal data should be processed"* [113]. The organisations usually agree on the terms of data sharing and sign a legally-binding document, coined the data processing agreement (DPA)[18]. Unfortunately, the compelling practical and ethical justifications usually defined in the DPA for the 'why' are not universally understood by all parts. Moreover, the auditing process to check compliance with the DPA relies on information technology applications records that usually fail to inform about 'why' and 'how' the personal data has been processed transparently.

Healthcare is a typical case of cross-organisational data sharing because healthcare professionals from multiple organisations need to process patients' data to perform their tasks. The patient's data here means the electronic medical records (EMR), which contain personal data about the patient. The patient's data are usually spread among hospitals and clinics that have treated the patient at least once. Thus, each organisation holds only some part of the data. In many cases, parts of the data remain unreachable, even in emergencies, because they are located in information systems outside the treating organisation's boundaries. This limitation could be addressed by using shared medical record systems that exploit cloud solutions [74, 87, 114] or distributed database systems, such as the InterPlanetary File System (IPFS) [115]. These shared systems promise to provide the required data consistency and availability for healthcare purposes. The healthcare organisations then act as joint controllers, defining and enforcing strict cross-organisation access control to patients' data. A successful access control system must be dynamic and granular to support

the complex nature of cross-organisational data sharing in healthcare. For example, a doctor must have access to a patient's data during treatment, but the access must be revoked when the doctor finishes the care. Moreover, all the access logs should be available for the audit process and regulatory compliance.

The Attribute-Based Access Control (ABAC) model offers a dynamic and fine-grained access control approach to protect personal data [82]. ABAC defines policies as combinations of rules and attributes that can be as granular as necessary. ABAC also uses context expression and contextual attributes, which gives dynamicity to the policy evaluation. Thus, ABAC defines not only "why" and "how" but also "by whom", "when" and "where" the personal data can be processed. A drawback of current systems that use the ABAC model is that it usually delegates access control management to the data storage system administrator, where the policies are defined, managed, evaluated, and enforced. Hence, the data controllers and subjects must trust that the storage system administrator will follow the defined policies and not allow any processing of personal data that does not comply with the respective policies.

Regarding transparency, the data controllers also depend on the data storage systems to keep and disclose records about data processing activity on the personal data. Recently, blockchain and smart contracts concepts have been proposed to facilitate transparency over data access control [116–118]. In these proposals, the data processing is logged as transactions of the blockchain, which are immutable and transparent for auditing. However, to the best of our knowledge, none of the proposals considers the complexity and dynamicity required for healthcare cross-organisation data sharing. In most proposals, the so-called 'data owner' defines the access control policies, but the 'data owner' of the patient's data is an unclear role [119] and it is not defined in the GDPR. Moreover, relying on the patients to define specific access control policies for their data is unrealistic.

Our research focuses on exploiting smart contracts and blockchain technology through the ABAC model for dynamic access control of personal data across organisations. Thus, we must overcome the following challenges: First, healthcare organisations should agree and comply with common access control policies for patient data processing as joint data controllers. Second, the access control system should guarantee a valid purpose for data processing. Third, policy decisions and enforcement should not depend on a centralised trustworthy party. Finally, the logs of data access activity should be transparent and auditable.

The paper proposes the SmartAccess system that follows the reference architecture defined by the XACML standard [79] and implements the ABAC components as smart contracts. In the blockchain network, the data controllers jointly define and manage the policies, in consensus, with the rest of the data

controllers in the blockchain network. The data processors and the context provide attributes validated by their data controllers to justify and legitimate the data processing activities. Data processing is only allowed if the data processor runs the smart contracts and has the right attributes to comply with the policy's rules, therefore, yielding a permit decision. Data processors run the access control smart contracts locally in their computers without relying on a central party. Every function executed in the smart contracts generates an auditable transaction published in the blockchain. Therefore, any node of the network can always search for the access requests that have been performed to given data, the values of the attributes at that time, the policy enforced during the access evaluation, and the resulting access decision. Moreover, we analyse the security threats of the SmartAccess system and demonstrate the proposed solution in different use cases healthcare scenarios: access with patient consent and access during acute care. Finally, we demonstrate the feasibility of the solution through the implementation of a size-efficient policy representation and experiments that evaluate the complexity and scalability of the smart contract functions. Moreover, we test the performance of the SmartAccess system in different blockchain network setups, measuring the latency and throughput of the transaction and compare the performance of SmartAccess with two implementations in solidity of the Role-based Access Control (RBAC) and Access Control List (ACL) mechanisms.

6.2 BACKGROUND AND RELATED WORK

This section presents the background of the concepts adopted in the SmartAccess system, namely blockchain, smart contracts and the Attribute-based Access Control Model. We also present related work and summarise their characteristics in a table.

6.2.1 *Blockchain and smart contracts*

Blockchain technology is defined by two essential elements: the data structure of the blocks and the peer-to-peer network composed of the participant nodes [120]. Blockchain allows running distributed applications without needing a trusted third party while meeting security requirements such as integrity, authenticity, non-repudiation, and accountability [121].

The blockchain data structure comprises blocks concatenated to the previous one via hash values. Except for the first (genesis block), each subsequent block contains valid transactions and the hash of the content of the previous block. Hence, the hash concatenation between the blocks ensures the integrity and immutability of the transactions stored in each block since these cannot be tam-

pered with without breaking the hash chaining. The integrity and immutability of the transactions rely on the peer-to-peer network. Each node of the network has a copy of the hash chaining, and because of the network consensus mechanism, all the nodes have the same global view of the blockchain. Nodes can order and package validated transactions into a candidate block to be inserted into the blockchain. The successful insertion of a block in the chain is known as 'mining', and the nodes eligible to insert blocks in the blockchain are the 'miners'. The selection of miners depends on the consensus mechanism [122].

The adopted consensus mechanism depends on the type of network and the roles of the nodes. The blockchain network can be private or public, and the nodes' roles can be permissioned or permissionless [123]. In a public blockchain, everyone can join and leave the network; on the other hand, in a private blockchain, the nodes must be added to the blockchain and known by the entire network. In a permissionless network, all nodes play the same role, i.e., every node in the network can mine blocks and participate in the consensus. The network is called permissioned when the nodes are divided into groups according to their role, e.g., miner nodes and nodes that only generate transactions. Usually, in a permissioned blockchain, the role of a node is assigned when the node joins the network [124].

Blockchain was initially proposed to support financial transactions of virtual cryptocurrency in the Bitcoin network [120]. A *transaction* is a message sent from one node address to another. It can include binary data, which is called 'payload'. The technology has evolved, and it currently can be used to implement more complex transactions, known as smart contracts, first implemented in the Ethereum network [121]. *Smart contract* is a code with executable functions and storage space for its data, of which all nodes have a replica locally.

When a smart contract is deployed, the contract owner broadcasts a transaction carrying the payload as a code linked to a public address. Once this transaction is mined, all nodes store a replica of the smart contract [125]. To execute a smart contract function, the 'sender node' broadcasts a transaction to the network with the smart contract's public address, carrying the function input arguments inside the transaction payload. The sender node then waits for the transaction to be validated and mined in a block. Meanwhile, from the point of view of all other nodes, the transaction has not yet happened. Once the transaction is mined and broadcast to the network, the transaction is considered executed. The miner node sends a 'transaction receipt' to the address of the sender node, confirming that the transaction was mined. Therefore, the 'event' is emitted at that point. It means that the sender node executes the smart contract function with the input arguments of the transaction and emits the event that the user front-end can then process [126]. Although the function is executed locally, any node can verify the event transactions executing the

smart contracts with the same inputted arguments. A smart contract function can also execute functions from other contracts as a process. In order to send transactions and execute functions, it may be required for the sender to pay a fee. In Ethereum, the fee is referred to as 'gas'. In Ethereum Virtual Machine (EVM) based blockchain, gas is an excellent way to evaluate the complexity of the smart contracts since the gas usage increases with the complexity of the transactions. The smart contracts' global state can be seen as a virtual machine running all the code on the blockchain.

6.2.2 *Attribute-Based Access Control model*

The ABAC model defines an access control paradigm by which access rights are granted to the data requester by using policies that consist of logical combinations of attributes. ABAC policies, requests and responses are expressed in the XACML language, an OASIS [80] standard. A policy is a combination of rules that the requester must obey. We achieve this by combining algorithms at the policy set level (i.e. policy combining algorithms) or at the policy level (i.e. rule combining algorithms). Each algorithm defines how to properly merge the evaluation of the different requests to produce a unique decision. The policies are associated with the 'targets', which can be a resource, a type of action, a context expression, or a combination of these. A context expression describes the circumstances under which access should be allowed. For example, a policy protects a combination of targets: a patient's data (resource) to be read (type of action) in case of an emergency (context expression). When a request is issued, the rules expressed in the policies are evaluated, exploiting the attribute values to return a response. The responses contain the decision concerning the request. A response can be 'Permit', 'Deny', 'Indeterminate' (in case of errors or missing values) or 'Not applicable' (the request does not regard any of the policies).

The XACML standard defines five main components that handle access decisions, namely the Policy Administration Point (PAP), Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Information Point (PIP), and a Context Handler (CH). The PAP stores and manages a persistent pool of policies associated with the target identifiers. The PEP constitutes the integration hook to any system, where the resources to be secured are stored and managed. A PEP receives the access requests and freezes the execution workflow until a decision is yielded. At the same time, it propagates the requests to the PDP, which is the core decision place for any incoming access request. The PDP retrieves all the necessary attributes and contextual information from the PIP, evaluates the defined policies, and yields a decision accordingly. The PIP is responsible for retrieving and storing attribute values. The Context Handler (CH) is responsible for deriving the context of a certain request and, in some recent efforts [82] for

semantically uplifting the attribute values stored in the PIP to infer additional context.

6.2.3 *Related work*

Here we discuss the most relevant literature proposing solutions for access control to medical data using blockchain technology. In Table 6.1 we classify the related work according to XACML's five main components. Even if the access control proposed in the paper is not based on the XACML standard, we evaluate if the proposed solution uses blockchain to handle each part of the access control process as defined in XACML. We mark with ✓in the respective column when a solution uses the blockchain to store or manage the access control policies (PAP), to handle the requests and deliver the decision (PEP), to evaluate the policies (PDP), to store and manage the attributes used to evaluate the policies (PIP), and to store and manage contextual attributes used to evaluate the policies (CH). Moreover, we also mark with ✓if an implementation is presented in the papers. Bellow, we present a summary of the proposed solutions and how SmartAccess differs from existing literature.

Table 6.1: Classification of related works indicating if the blockchain covers the components defined by the XACML standard and if the proposal was implemented. See also text

Ref	PAP	PEP	PDP	PIP	CH	Implementation
[127]	✓					
[128]		✓				
[129]		✓				
[130]		✓				✓
[131]		✓				
[132]	✓	✓				
[133]				✓		✓
[134]	✓	✓	✓			✓
[116]	✓	✓	✓	✓		✓
[117]	✓		✓	✓		✓
[118]		✓	✓			✓
This work	✓	✓	✓	✓	✓	✓

According to Dias et al. [127], current systems that attempt to share access control policies between healthcare entities are prone to system and network faults and do not assure the integrity of policies. The authors propose using a consortium blockchain, where the ABAC policies are stored off-chain, the point-

ers are stored as transactions, and the different entities know all the parties that can act over the e-Health resources. Although the approach allows entities to maintain the consensus about the policies, it is limited to sharing policies. SmartAccess goes beyond and uses smart contracts technology to manage, evaluate and enforce access control policies in a distributed system.

Fan et al. [128] propose the Medblock, a data-sharing framework with an access control mechanism based on a signature scheme. The sensitive data and the pointers to the patient EMR are encrypted with the multi-signature scheme inside the blockchain. The access control mechanism traverses the blocks until it finds the right block by comparing the signature with the signature collection on the ledger. Whether the user can see the encrypted content on the block depends on the comparison result. Zhang et al. [129] propose the FHIRChain for data sharing among clinicians and researchers based on the FHIR standard. FHIRChain addresses five key interoperability requirements: user identifiability and authentication, secure data exchange, permissioned data access, consistent data formats, and system modularity. The data access control is based on a smart contract that results in an access token. The access tokens are defined for each data transaction, where it uses asymmetrical encryption to protect the data pointers off-chain. This design uses the users' digital health identities to encrypt content so that only users holding the correct digital identity private keys can decrypt the content. Daraghmi et al. [130] propose MedChain, an incentive consensus mechanism that leverages the degree of the reputation of health providers regarding their efforts in maintaining medical records and creating new blocks. The access control contract includes all permissions-related information specific to every record based on smart contracts. It lists the Ethereum addresses for all users who have access permissions on the record. This contract specifies the level of access and the symmetric key encrypted with each user's public key. Although [128–130] promise an effective access control, none offer dynamic access control for daily healthcare needs. In those proposals, the access permissions need to be listed or specified beforehand the need for data.

Azaria et al. [131] propose MedRec, a decentralised EMRs management system using blockchain technology. MedRec is a modular design that manages permissions, authorisation, and data sharing between participants. The authors highlight the ability of MedRec to encrypt outside data and preserve hash pointers to patients' health records along with their access permissions in the blockchain. Like MedRec, Dagher et al. [132] propose Ancile, an Ethereum-based blockchain for a record management system that utilises smart contracts for heightened access control and data obfuscation. Ancile keeps the patients' medical records in the existing databases of providers, and reference addresses to these records and their permissions to each record are stored in the smart contract. It was designed to store the Ethereum addresses of all nodes that may

interact with a record, a level of access, and a symmetric key encrypted with each node's public key. Dubovitskaya et al. [133] propose an EHR system where the patients specify the access control policies and permissions to each healthcare professional. Rajput et al.[134] propose a system that allows the 'owner' of the records to assign the rules for an emergency team or staff member who can obtain permission to access the current information from the resource by considering the time restriction. The access control is based on a list of pre-registered emergency teams and physicians.

Rauhani et al.[117] proposed an ABAC system for EMR data sharing. It is different from ours because it does not consider context expression, which is fundamental for healthcare applications where the access rights are dynamic and depend on the patient's needs. Maesa et al.[116] also proposed an ABAC system using blockchain. However, the values of attributes cannot be changed since they are stored on the blockchain. They are auditable since their updates can be executed only through blockchain transactions, which are recorded on the blockchain. Both proposals [116, 117] do not consider that the attributes must be authenticated by the organisations where the processor works every time they interact with the access control system. As an asynchronous system, the blockchain would require that the organisations update the attributes of their professionals on the blockchain continuously, which would be a burden for the dynamic attributes of the healthcare professionals. On the other hand, [118] proposed to keep the user attributes out of the blockchain and rely on the trusted authorities to maintain a list of users associated with their verified attributes. Then using a smart contract, these authorities authenticate the user's attributes during the user request for data. SmartAccess uses an authentication token as the source of the attributes for professionals and the environment. Thus, every time the professional performs a request, the professional must provide an access token signed by the organisation.

The proposals [127, 129, 131, 134] use PoW and incentivise health providers and medical researchers to participate in mining by earning Ether, an Ethereum-based currency unit, to fund their activities' continuation. In other words, they participate in mining to get beneficiaries from the network, although most of the current healthcare systems are welfare-oriented with no intent to involve any monetary value. Because of that, SmartAccess uses non-monetary consensus mechanisms based on voting, Byzantine Fault Tolerance (BFT) consensus [135]. Furthermore, all the proposals [131–133] based on a pre-defined list of users cannot be used in emergency situations since a patient under an emergency cannot know the specific teams or professionals that will treat them. Moreover, when the authors refer to 'data owners', they are referring to 'patients', the data subjects who usually have no control over the data collected about them. Although these proposals give more control to the data subject, in the healthcare

scenario according to GDPR, the healthcare organisations are responsible for the access control to the EMR.

6.3 SMARTACCESS: ATTRIBUTE-BASED ACCESS CONTROL BASED ON SMART CONTRACTS

This section proposes SmartAccess, a new blockchain-based access control system to secure personal data across organisations. First, we present an overview and a description of the network and smart contracts. Then, we describe the access control flow, showing how the network nodes interact with the smart contracts.

6.3.1 *Overview*

SmartAccess is based on a blockchain peer-to-peer network and smart contracts. The network nodes use the smart contracts to define access control policies, request access permission and verify whether the request is permitted or not. Following the XACML, we denote 'resource' any data that can be requested using SmartAccess and 'target' as the combination of context expression, action and resource requested. Moreover, any network node can audit the requests, targets, decisions, and policy enforcement logs that are transparent in the blockchain.

We assume that nodes run user applications that would enable them to be authenticated, search for the resource identifiers and request for the resource stored off-chain. The actual user application is considered out of the scope of this paper because it would be specific to a use case and perhaps to a blockchain framework. We also assume that the usage of cryptography to protect the resources in the storage would be done. The cryptography and management of cryptography keys used to encrypt the data are also considered out of the scope of this paper. Instead, we focus on a generic access control system provided by smart contracts to iterate with the off-chain storage and retrieve resources.

Fig. 6.1 shows an overview of the network and the smart contracts. The network comprises data controller, storage and processor nodes, which have different roles according to their function regarding the GDPR. The smart contracts running on the nodes offer functions to manage the private network and provide access control to an off-chain resource. The four smart contracts follow the XACML model: Enforcement Smart Contract (PEPSC), Decision Smart Contract (PDPSC), Policies Smart Contract (PAPSC) and Information Smart Contract (PIPSC).

Fig. 6.2 shows how SmartAccess components interact, including the nodes of the network and their communication with the four smart contracts. The controller nodes propose and manage the access policies to protect a common

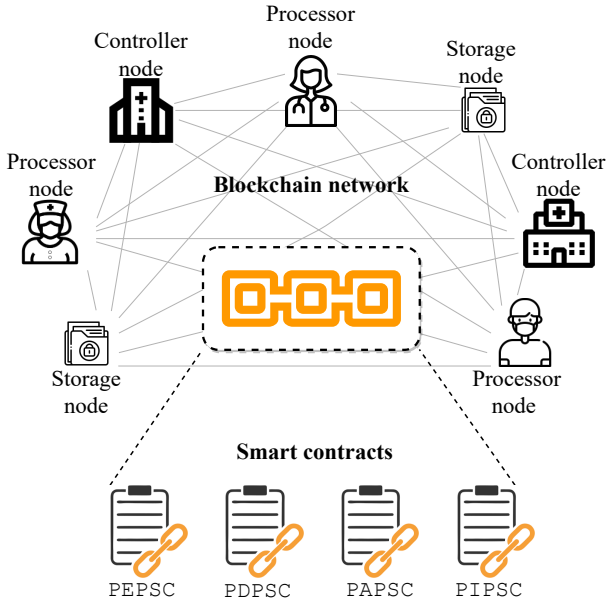


Figure 6.1: Overview of SmartAccess, which is composed of a peer-to-peer blockchain network composed of controller, processor and storage nodes, and four smart contracts: Enforcement Smart Contract (PEPSC), Decision Smart Contract (PDPSC), Policies Smart Contract (PAPSC) and Information Smart Contract (PIPSC).

resource, and the policies represent what they jointly agreed upon. Once a policy is mined, it is stored in PAPSC (step 0.1). Controller nodes also store and remove the attributes of processor nodes at PIPSC (step 0.2). When PEPSC receives a request from a processor node for a target (step 1), it requests an evaluation by PDPSC (step 2). PDPSC then gets the policy from PAPSC (step 3) and attributes from PIPSC (step 4), evaluates the policy and saves the decision at PEPSC (step 5). Finally, the processor node requests off-chain the storage node for the resource (step 6). The storage node verifies the decision in PEPSC (step 7), and if this request has been 'Permit', the storage node sends the resource off-chain to the processor node (step 8).

The network nodes, smart contracts and flow are explained in further detail respectively in subsections 6.3.2, 6.3.3 and 6.3.5.

6.3.2 *Network*

The proposed blockchain network is private and permissioned, where nodes are added and removed from the network and have roles and permissions. A node is a connected device or server from where the user can send transactions,

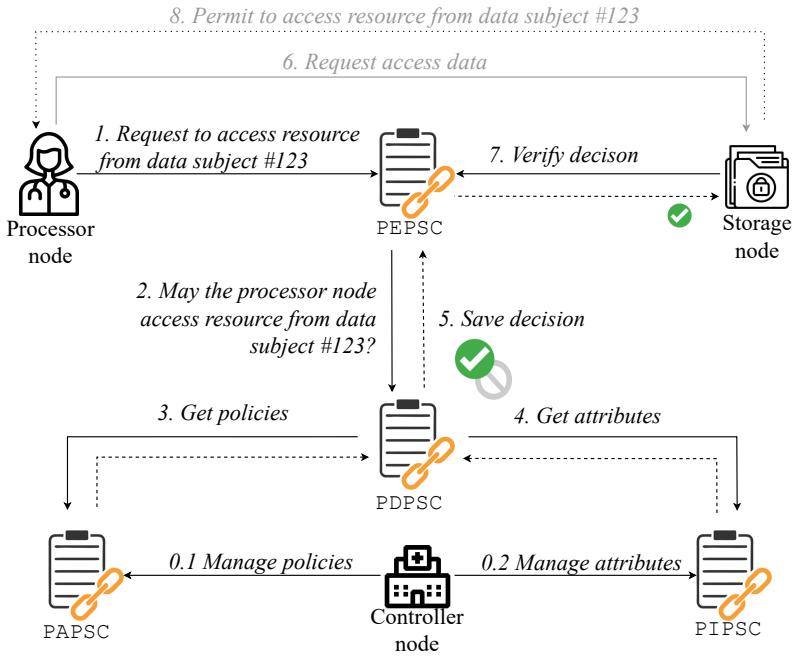


Figure 6.2: Overview of SmartAccess flow, showing how the nodes interact with smart contracts when there is a request to access a target. See text for details.

execute smart contracts and store the blockchain. A node also runs an application implementing the business logic layer. Every node has a pair of asymmetric keys used to generate its address and digital signature. Below, we describe how each node type maps to the GDPR roles and explain their responsibilities in the blockchain network.

Controller node (n_c): represents an organisation with the data controller role in GPDR. Each organisation runs its own node, which participates in the consensus mechanism, generates blocks, and manages the network and the smart contracts. A controller node defines and deploys access control policies and validates the proposed policies through the consensus. It can also add new controllers, processors and storage nodes, as well as manage the attributes of their processor nodes, which are the professionals that work for the organisation. Controller nodes are also responsible for authenticating the processors and providing them with an authentication token that carries valid attributes for further interaction with the system. In a healthcare scenario, hospitals and clinics are examples of controllers that define policies and manage the access of professionals and patients to the resources.

Processor node (n_p): represents the data processor role in GDPR. A processor node is a requester in the ABAC model, and who runs the smart contracts to request access to a resource. In a healthcare scenario, the processor nodes

are the healthcare professionals or the patients (data subject). A data processor only processes the resource on behalf of the data controller. Each n_p is added to the blockchain network by some n_c, and it must be authenticated by the same n_c before requesting data access.

Storage node (n_s): represents a persistent data storage system, such as a cloud storage provider or IPFS, serving as a gateway between a resource and the n_c and n_p. Storage nodes are added to the network by some n_c and, after that, they can receive requests from any n_p. When a request to access a resource is received, the storage node runs the smart contracts to verify if n_p has permission to access the data. In the ideal scenario, the n_c nodes that are hospitals or contain patient resources should also be part of the network as n_s nodes. Hence creating decentralised storage for the patients' data.

6.3.3 *Smart Contracts*

The smart contracts (SC) implement the main components of the ABAC model according to XACML: PEPSC for policy enforcement, PDPSC for yielding decisions, PAPSC for policies management and PIPSC for attributes storage. The XACML Context Handler (CH) is distributed into the four SC because all requests and policies are driven by context expression. This means that SC handles the requests according to the circumstances and the purpose of the access and applies the appropriate policy.

Table 6.2: Definitions of arguments used in the SC.

Term	Description
$\mathtt{Actions}$	Possible actions on the data: Create, Read, Update, Delete
$\mathtt{Policy}$	Logical representation of an access policy
$\mathtt{Attributes}$	Logical representation of users' attributes
$\mathtt{ContextAttr}$	Logical representation of contextual attributes
$\mathtt{ContextExp}$	Statement that expresses the context of the data access
$\mathtt{Decision}$	Possible decisions are Permit or Deny
$\mathtt{Policy}_{\mathrm{ID}}$	Identifier of a policy
$\mathtt{Subject}_{\mathrm{ID}}$	Identifier of a data subject
$\mathtt{Resource}_{\mathrm{ID}}$	Identifier of the data (e.g. URI) related to a $\mathtt{Subject}_{\mathrm{ID}}$
$\mathtt{PK}_n$	Public key of a node, where $n \in \{c,p,s\}$
$\mathtt{Addr}_n$	Network address of a node, where $n \in \{c,p,s\}$
$\mathtt{Auth}_p^c$	Authorisation token of n_p, signed by n_c
$\mathtt{Addr}_{signer}$	Address of the n_c that signed $\mathtt{Auth}_p^c$

Table 6.3: Overview of the functions implemented by each SC, with sender, input arguments and result of the function when successfully executed.

SC	Function	Sender	Input	Result
PEPSC	RequestAccess	n_p	$Subject_{ID}$, $Resource_{ID}$, ContextExp, Actions, Attributes, $Auth_p^c$	Calls PDPSC:EvaluateRequest
	SaveDecision	PDPSC	Decision, $Subject_{ID}$, $Resource_{ID}$, ContextExp, Actions, PK_p	Saves Decision in PEPSC associated to PK_p and requested target
	RevokeAccess	n_c, n_p PIPSC	$Subject_{ID}$, $Resource_{ID}$, ContextExp, Actions, PK_p	Revokes Decision of PDPSC associated to PK_p and requested target
	SaveRequestLog-Obligation	n_s	ContextExp, $Subject_{ID}$, $Resource_{ID}$, PK_p agente, country, city, timestamp, action, outcome	Saves request header attributes from off-chain request in PEPSC
	VerifyDecision	n_s	$Subject_{ID}$, $Resource_{ID}$, ContextExp, Actions, PK_p	Verifies if PK_p has a Decision Permit to the requested target
PDPSC	EvaluateRequest	PEPSC	$Subject_{ID}$, $Resource_{ID}$, ContextExp, Actions, Attributes, $Auth_p^c$	Calls RetrievePKp, IsAddrOfController, LoadPolicy and GetContextAttr. Then evaluate the request and yields Decision calling SaveDecision
PIPSC	AddNode	n_c	$Addr_n$,PK_n	Adds a new node to the network, saving $Addr_n$ in PIPSC
	RemoveNode	n_c	$Addr_n$, PK_n	Removes a node from the network, deleting $Addr_n$ from PIPSC
	RetrievePKp	PDPSC	$Addr_c$, $Addr_p$	Searches for the addresses and retrieves PK_p from PIPSC
	IsAddrOfController	PDPSC	$Addr_{signer}$	Verifies if the signer is the legit n_c for n_p by checking whether the $Addr_p$ is associated to the signer $Addr_c$ in PIPSC
	AddContextAttr	n_c,n_p	ContextAttr,ContextExp, $Resource_{ID}$, $Subject_{ID}$,$Addr_p$	Adds ContextAttr in PIPSC
	RevokeContextAttr	n_c,n_p	ContextAttr,ContextExp, $Resource_{ID}$, $Subject_{ID}$,$Addr_p$	Revokes ContextAttr in PIPSC and calls PEPSC:RevokeAccess
	GetContextAttr	PDPSC	ContextExp, $Resource_{ID}$, $Subject_{ID}$, $Addr_p$	Searches and gets ContextAttr in PIPSC
PIPSC	CreatePolicy	n_c	$Policy_{ID}$, Policy, ContextExp, Actions, $Subject_{ID}$, $Resource_{ID}$	Adds Policy in PAPSC mapped to ContextExp, Actions,$Subject_{ID}$ and $Resource_{ID}$
	ChangePolicy	n_c	$Policy_{ID}$ or ContextExp, Actions, $Subject_{ID}$, $Resource_{ID}$	Changes Policy or the map relations
	RemovePolicy	n_c	$Policy_{ID}$ or ContextExp, Actions, $Subject_{ID}$, $Resource_{ID}$	Removes Policy from PAPSC
	LoadPolicy	PDPSC	ContextExp, Actions, $Subject_{ID}$, $Resource_{ID}$	Loads Policy from PAPSC

Table 6.2 describes the input arguments for the SC functions, and Table 6.3 summarises SC and their functions (denoted ⟨smart contract name⟩:⟨function name⟩). A sender is a node or smart contract allowed to emit the transaction to run the function. There is a checking condition inside each function to verify if the sender is allowed to run the function, and if the required condition is true, the node is allowed to run the function. The four SC contain the functions code and a relational database where each argument is saved and managed accordingly to each function execution. The functions can add, remove, load, change, and save a variable in the relational database of the smart contract. For example, the database contained in PAP stores the policies for each target in the following format ($\mathtt{ContextExp}$, $\mathtt{Actions}$, $\mathtt{Resource}_{\mathrm{ID}}$), for example ('emergency', 'read', '123'). The result of the functions listed in the Table 6.3 describes what happens in the system when the function is successfully executed; otherwise, the execution stops without any change for the system.

Every transaction that executes a function is validated and mined in a new block added to the blockchain. The transaction payload logs the arguments inputted, such as the target, attributes, policies and decisions used when executing the corresponding function. Any network node can search for an argument and retrieve all the transactions containing the argument.

Below we describe the SC and their relationships with the XACML model. See Fig. 6.2 for a simplified overview.

*Enforcement Smart Contract (*PEPSC*)*

The smart contract implements the PEP functionality in the ABAC XACML model. It contains four functions: PEPSC:RequestAccess to process the data access requests, and PEPSC:SaveDecision, PEPSC:RevokeAccess and PEPSC: VerifyDecision to manage the corresponding decisions. n_p runs PEPSC: RequestAccess, which triggers a call to PDPSC:EvaluateRequest. After the evaluation, the decision is taken, and PEPSC:SaveDecision is called back. PEPSC:RevokeAccess can be called by n_p, n_c, or by PIPSC:RevokeContextAttr when a contextual attribute is revoked. The PEPSC:VerifyDecision is called by n_s to check if the requesting processor n_p has a valid access permission. Finally, PEPSC:SaveRequestLogObligation is called by n_s to save the attributes related to the off-chain request such as location, origin IP address, etc.

*Decision Smart Contract (*PDPSC*)*

The contract implements the PDP functionality in the ABAC XACML model, being the core decision point for any incoming access request. It contains only the PDPSC:EvaluateRequest function, which evaluates the access requests. This function is called by PEPSC:RequestAccess and requires the same input argu-

ments. The evaluation process requires PDPSC to call functions in the other SC to acquire additional information: PIPSC:RetrievePKp, PAPSC:LoadPolicy, PIPSC:IsAddrOfController, PIPSC:GetContextAttr and PEPSC:SaveDecision.

*Policies Smart Contract (*PAPSC*)*

The contract implements the PAP functionality in the ABAC XACML model, where the policies are defined, managed and stored. In PAPSC, a Policy is related to the $\text{Policy}_{\text{ID}}$, ContextExp, Actions, $\text{Subject}_{\text{ID}}$ and $\text{Resource}_{\text{ID}}$. It contains four functions: PAPSC:CreatePolicy, PAPSC:ChangePolicy, PAPSC:RemovePolicy and PAPSC:LoadPolicy. Only n_c may run the policy management functions of PAPSC to create, change or remove a policy. Also, only PDPSC calls PAPSC:LoadPolicy to retrieve a Policy for evaluation.

*Information Smart Contract (*PIPSC*)*

The smart contract implements the PIP functionality in the ABAC XACML model, gathering the information necessary to evaluate the requests. PIPSC stores the addresses of all nodes. It contains seven functions: PIPSC:AddNode, PIPSC:RemoveNode, PIPSC:RetrievePKp, PIPSC:IsAddrOfController, PIPSC:AddContextAttr, PIPSC:GetContextAttr and PIPSC:RevokeContextAttr. n_c manages the network with PIPSC:AddNode and PIPSC:RemoveNode. PIPSC:IsAddrOfController verifies whether the signer Addr_{signer} of a token Auth_p^c is the same n_c which added n_p to the network. PIPSC also stores the ContextAttr according to the ContextExp. The relational database used to store the ContextAttr differs for each ContextExp supported in the system. The functions to add, get and revoke ContextAttr: PIPSC:AddContextAttr, PIPSC:GetContextAttr, PIPSC:RevokeContextAttr. We present in Section 6.4 examples of two different ContextExp that customise the PIPSC according to the use cases.

6.3.4 *Setup and Maintenance*

At the system setup, the blockchain initial network is created with at least three controller nodes as required in the voting consensus adopted in the SmartAccess. Next, the controller nodes establish the functionality for SmartAccess by running setup functions in the steps illustrated in Fig. 6.3, In step 1, one n_c deploys all the contracts: PEPSC, PDPSC, PIPSC and PAPSC. In step 2, any n_c can add other controller and storage nodes by running the PIPSC:AddNode. Then, in step 3, each n_c adds its processor nodes, which are the professionals that work for the corresponding organisation. Finally, in step 4, any n_c can create a Policy by running PAPSC:CreatePolicy.

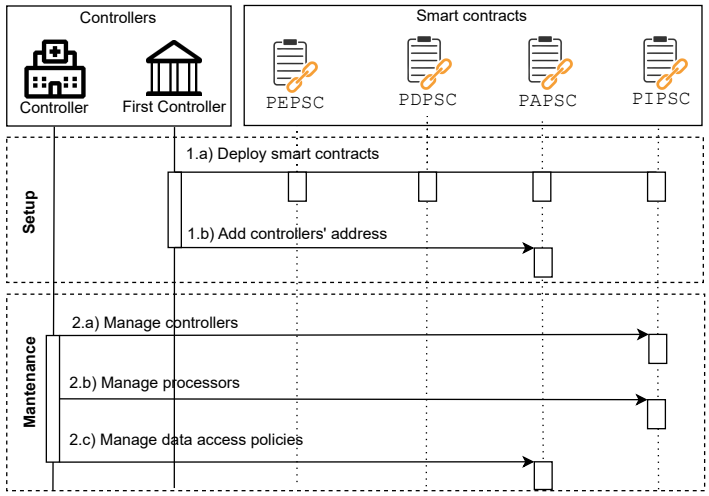


Figure 6.3: Setup and maintenance of the smart contracts. The setup requires an initial n_c to deploy all the smart contracts and add the n_c nodes that will be part of the network. The n_c nodes maintain the network by managing other controllers, processors and data access policies.

Once in the SmartAccess network, all controller nodes participate in the consensus mechanism and maintenance of the network. n_c runs PIPSC:Add Node and PIPSC:RemoveNode to manage the controller and storage nodes. With the same functions, each n_c manages its processor nodes. Moreover, any n_c can run PAPSC:CreatePolicy, PAPSC:ChangePolicy and PAPSC:Remove Policy to manage the policies, which must be in agreement with all the n_c in the network. The policy agreements among controllers are achieved through the consensus mechanism, in which the transactions to run policy management are validated and stored in the blockchain in an auditable manner. See more details about the functions in Table 6.3.

6.3.5 *Access control flow*

Fig. 6.4 shows the interactions between the various components during the evaluation of an access request. The flow is divided into three main steps: authentication, authorisation and data processing.

The processor node's authentication happens off-chain. In step 1.a, a user n_p sends credentials to be authenticated by the organisation (n_c) where n_p works. n_c authenticates and generates a signed authentication token $Auth_p^c$ containing the attributes of user n_p. This token is returned to the user in step 1.b and used in all further communication with the network.

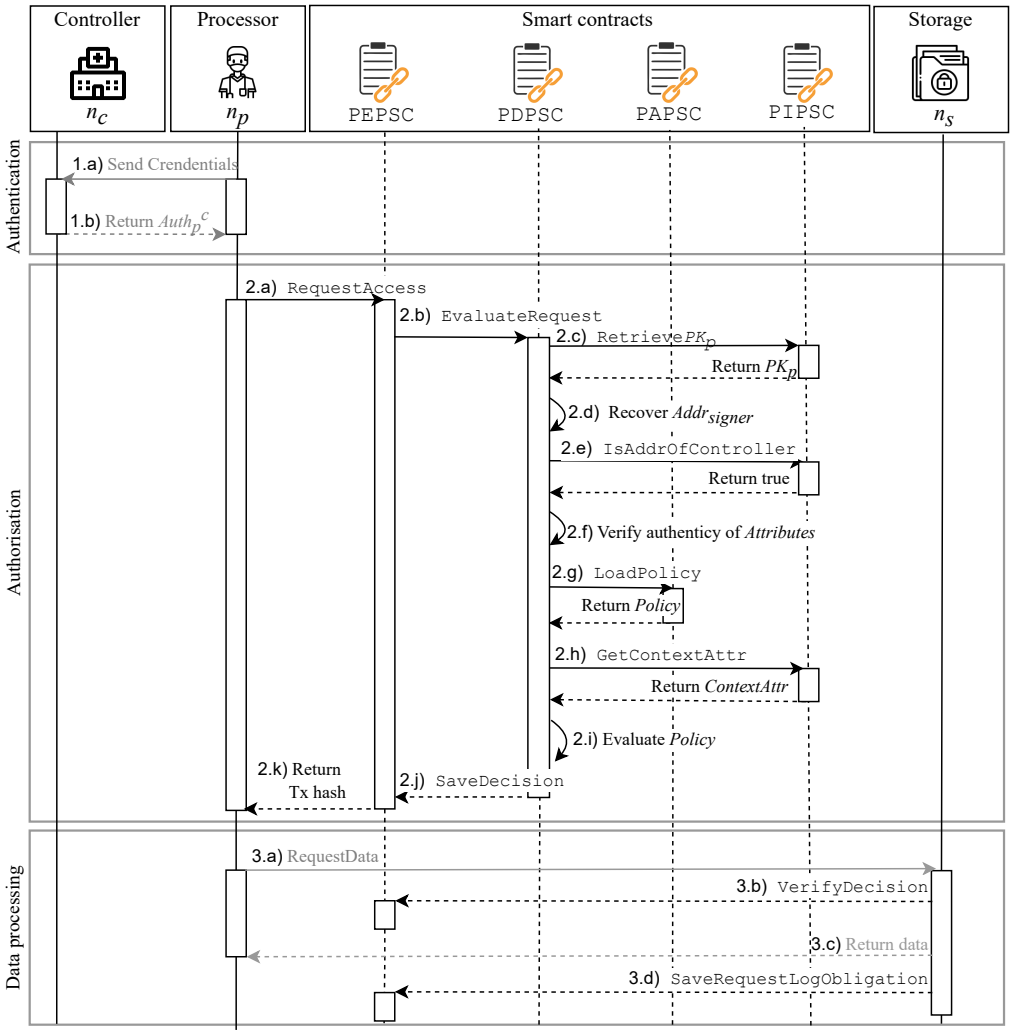


Figure 6.4: Access control flow for n_p. n_p requests read access to some data by running PEPSC:RequestAccess. It triggers communication between the various SC to permit/deny access and save the decision. After obtaining permission, n_p sends a request message off-chain, *RequestData*, to read the resource from n_s. Finally, n_s verifies if the requester is allowed access by running PEPSC:VerifyDecision, then returns the resource to n_p. In grey, are the off-chain requests.

The authorisation step, which is the main focus here, happens in the blockchain. It starts in step 2.a, when a user n_p, in possession of $Auth_p^c$, runs PEPSC:RequestAccess. Then, in step 2.b, PEPSC calls PDPSC:Evaluate Request and waits for a decision. In step 2.c, PDPSC takes the sender's $Addr_p$ and retrieves the respective PK_p from PIPSC. In step 2.d, PDPSC recovers the address of the signer $Addr_s$ of the token $Auth_p^c$. Then, in step 2.e, PDPSC verifies if the signer is

the legit n_c for n_p by checking whether $Addr_p$ is associated with the signer's $Addr_c$ in PIPSC. In step 2.f, PDPSC verifies if the Attributes are legit by performing a hash function of the Attributes and PK_p and comparing the result with the signed hash from $Auth_p^c$. In step 2.g, PEPSC loads the Policy that is mapped to the ContextExp, Actions, $Subject_{ID}$ and $Resource_{ID}$ criteria. In step 2.h, PDPSC gets the ContextAttr required to evaluate the Policy. In step 2.i, PDPSC evaluates if the user has the Attributes and ContextAttr that comply with that Policy. Finally, in step 2.j, PDPSC calls PEPSC:SaveDecisionto save the Decision related to $PK_p$and the requested target in the PEPSC. In step 2.k, the n_p receives the transaction TX hash after the PDPSC:EvaluateRequest finished execution.

The actual data processing occurs in step 3.a, when the user n_p requests to process the resource by sending a message to n_s. This message is encrypted with PKs and signed by n_p. In step 3.b, n_s decrypts the message and calls PEPSC:VerifyDecision to verify if PK_p has a permit or denial decision associated with the target and resource in the PEPSC. If permitted, in step 3.c, n_s returns the resource to n_p, if not, the access will be denied. A similar process is used if n_p requests to create, update or delete data. After the response from n_s to n_p, in step 3.d ns runs the PEPSC:SaveRequestLogObligation to save in the blockchain all the relevant information from the off-chain data processing to enable transparent auditing.

Once the n_p has Decision regarding the specific target, this Decision can be revoked. The revocation can be automatic if we define an expiration time to Decision or if n_p lose an attribute that granted the decision in the first place. We will see the revocation flow in detail in the use cases for each ContextExp.

6.4 USE CASES: ACCESS CONTROL TO ELECTRONIC MEDICAL RECORDS

This section adopts SmartAccess to implement access control to share medical records in two scenarios. In the first use case, data access occurs with explicit patient consent, and in the second use case, access is consented implicitly during acute care. For each use case, we define PIPSC to store the corresponding ContextAttr according to the ContextExp. The blockchain network comprises healthcare organisations (n_c), healthcare professionals and patients (n_p), and a centralised cloud service provider (n_s) with the off-chain Electronic Medical Records (EMR) for both use cases. Fig.6.5 and Fig.6.6 present examples of the operations required to grant and revoke access rights through the management of context attributes - these operations are explained in detail below.

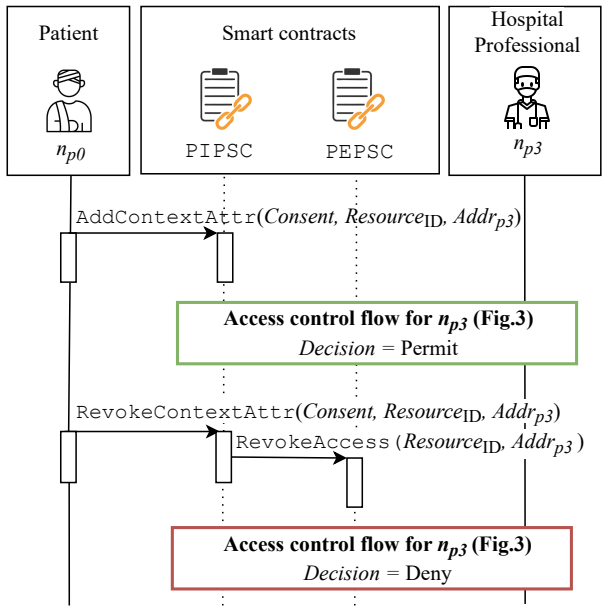


Figure 6.5: Example of ContextAttr flow in the access with the patient consent use case. The patient n_{p0} adds and revokes a Consent attribute to hospital professional n_{p3}. After adding the consent attribute, the n_{p3} performs an access control with the Decision being 'Permit'. After revoking the consent attribute, n_{p3} has the decision 'Deny' from the access control.

6.4.1 *Access with patient consent*

This use case illustrates the specific case that the patient n_{p0} goes to a medical appointment with a healthcare professional n_{p3}. Before the appointment starts, n_{p0} gives consent to n_{p3} to access their EMR. A patient can only give and revoke consent to their own resources. The consent can apply to the entire EMR or only to a part of the EMR by specifying the $Subject_{ID}$ or the $Resource_{ID}$, respectively. We designed the 'ContextExp'= 'consented session', where the Policy has one rule that n_p must have patient consent as ContextAttr saved on PIPSC associated to the $Addr_p$.

Fig. 6.5 shows the interaction of the patient with PIPSC to give access consent to n_{p3}. After authentication, the patient runs PIPSC:AddContextAttr(*Consent*, $Resource_{ID}$, $Addr_{n_{p3}}$) to save in PIPSC the ContextAttr=*Consent* associated to $Resource_{ID}$ and $Addr_{n_{p3}}$. Later, n_{p3} request to access the specific $Resource_{ID}$ following the access control flow described in Fig. 6.4. The access is permitted since the n_{p3} has valid ContextAttr and the other *Attributes* match the *Policy*. Access can have an expiration time or be revoked by the patient as follows: n_{p0} runs the PIPSC:RevokeContextAttr(*Consent*, $Resource_{ID}$, $Addr_{n_{p3}}$), that removes the ContextAttr and then triggers the access permission revocation

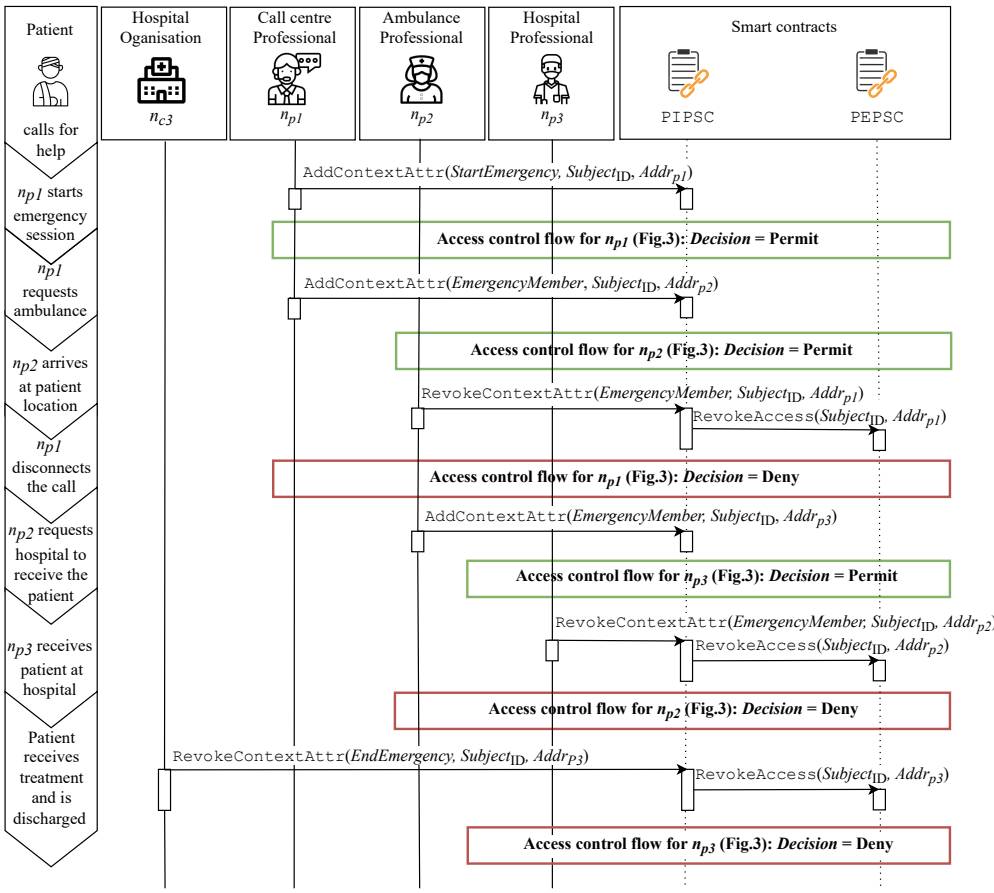


Figure 6.6: Example of ContextAttr flow in the access during acute care emergency use case. Professionals and organisations involved in acute care can start an emergency access session, add and revoke new professionals to the emergency and end the emergency access session according to the data needed for the treatment. The professionals have the decision 'Permit' when also the ContextAttr legitimate their participation in the patient emergency. If the ContextAttr is revoked, the professional has access to 'Deny'.

PEPSC:RevokeAccess($Resource_{ID}$, $Addr_{n_{p3}}$) to remove already-permit accesses. If the n_{p3} tries to access the patient's data after the ContextAttr is revoked, the decision will be 'Deny'.

6.4.2 *Access during acute care*

In this use case, the patient needs acute care, so access to the EMR is implicitly permitted for the patient's sake (fig.6.6). Multiple professionals treat a patient during an emergency, requiring dynamic access control to the patient EMR

across healthcare professionals from different organisations. The entire patient's EMR under emergency is referenced by $\mathtt{Subject}_{\mathrm{ID}}$.

For this use case, we have four main nodes involved in the emergency treatment: healthcare professionals from call centre n_{p1}, ambulance n_{p2} and hospital n_{p3}, and hospital controller n_{c3}. The professionals perform the cross-organisation sharing of data according to the data needed during the treatment. As soon as the professionals finish their task in the patient treatment, the professionals should no longer have access to the data.

Then we created ContextExp=*Emergency session*, which covers the time window since the patient requests emergency treatment until discharge. In PIPSC the professionals are added and revoked to a list of n_p involved in the patient's emergency session. The call centre and hospital professionals may add the ContextAttr=*StartEmergency* for a patient who has the EMR related to $\mathtt{Subject}_{\mathrm{ID}}$. Thus, once becoming part of the emergency session, any n_p can 'add' and 'revoke' another n_p from the emergency session. When the emergency session is over, the hospital organisation uses the ContextAttr=*EndEmergency* to revoke all the accesses of the remaining n_p involved in their emergency.

Fig.6.6 shows the interactions of the nodes with PIPSC. To start the emergency session, n_{p1} runs PIPSC:AddContextAttr (*StartEmergency*, $\mathtt{Subject}_{\mathrm{ID}}$, $\mathtt{Addr}_{p1}$). Then, n_{p1} requests an ambulance and adds n_{p2} to the emergency session, running PIPSC:AddContextAttr function again for the $\mathtt{Addr}_{p2}$. When n_{p2} starts treating the patient, n_{p2} removes n_{p1} from that emergency session by running PIPSC:RevokeContextAttr (*EmergencyMember*, $\mathtt{Subject}_{\mathrm{ID}}$, $\mathtt{Addr}_{p1}$). This triggers PEPSC:RevokeAccess($\mathtt{Subject}_{\mathrm{ID}}$, $\mathtt{Addr}_{p1}$) which also revokes the access permission that was once granted to $\mathtt{Addr}_{p1}$. The same happens for the ambulance and hospital professionals: n_{p2} adds n_{p3}, then n_{p3} revokes n_{p2}. Finally n_c runs PIPSC:RevokeContextAttr(EndEmergency, $\mathtt{Subject}_{\mathrm{ID}}$, $\mathtt{Addr}_{p3}$) to finalise the emergency session.

6.5 THREAT MODEL AND SECURITY ANALYSIS

We consider two types of threats in our system. The first type of threat is inherited from the blockchain, consisting of attacks on the blockchain consensus and ledger, blockchain peer-to-peer (P2P) network, SC and wallets [136]. Regarding consensus attacks, a mitigation approach is to use consensus mechanisms that guarantee Byzantine fault tolerance. The attacks regarding SC and wallets occur due to faulty code development and exploitable vulnerabilities, which is out of this paper's scope. For a future integration of SmartAccess and an EMR application, the SC must be extensively verified against exploitable errors. The second type of threat consists of attempts to bypass access control through impersonation and man-in-the-middle (MITM) attacks [137]. Here we focus on

the threats to access control, where a corrupted n_p node impersonates a user, uses fake attributes or bypasses the authorisation steps. We state the following proposition, and below, we analyse the soundness of SmartAccess considering three attacks: Impersonation Attack, Fake Attributes Attack and Reuse Permission Attack. We denote the adversary as ADV and assume that the signature scheme is EUF-CMA secure[138] and the public key cryptosystem is IND-CCA2 secure[139].

6.5.1 *Impersonation Attack Soundness*

Proposition 1: *Let ADV be a corrupted* n_p *node that listens to the connection between the* n_p*, denoted as B, and* n_c*, denoted as C. B sends its credentials and receives* Auth_B^C *from C. ADV steals B's token and runs* PEPSC:RequestAccess *to access a resource. We assume that ADV has no access to* PK_B*, then ADV has a negligible probability of successfully launching an Impersonation Attack.*

Analyses. ADV launches a Impersonation Attack by running the PEPSC: RequestAccess function with the inputs: ContextExp, Actions, $\mathrm{Resource}_{ID}$, Attributes and $\mathrm{Auth}_B^C = \sigma_C(\mathrm{Hash}(PK_B, \mathrm{Attributes}))$, where σ_C denoted the signature of *C* of the public key and attributes from *B*. PEPSC calls PDPSC: EvaluateRequest, which retrieves the public key of the sender PK_{ADV} and compares $\mathrm{Hash}(PK_{ADV}, \mathrm{Attributes})$ with $\mathrm{Hash}(PK_B, \mathrm{Attributes})$ from the signed Auth_B^C. Given the security of the one-way random Hash function, there is a negligible probability that the hashes result is the same value.

6.5.2 *Fake Attributes Attack Soundness*

Proposition 2: *Let ADV be a corrupted node that reads the policies in* PAPSC *and forges its* Attributes *to match with a* Policy*. Assuming that the signature scheme in* Auth_{ADV}^c *is EUF-CMA secure, the ADV has a negligible probability of successfully launching a Fake Attributes Attack.*

Analysis: ADV launches Fake Attributes Attack after been authenticated and receiving legit $\mathrm{Auth}_{ADV}^c = \sigma_C(\mathrm{Hash}(PK_{ADV}, \mathrm{Attributes}))$. ADV runs PEPSC: RequestAccess using altered $\mathrm{Attributes}_{fake}$ as input. As in Proposition 1, the comparison of the Hash results between the forged $\mathrm{Attributes}_{fake}$ and the Hash from Auth_{ADV}^c has a negligible probability of matching. The ADV can then tamper the Auth_{ADV}^c using the forged $\mathrm{Attributes}_{fake}$ instead of the legitimate Attributes. However, under the assumption of the secure digital signature scheme, ADV fails to forge a valid signature σ_c, and therefore, ADV fails to launch the Fake Attributes Attack.

6.5.3 *Reuse Permit Decision Attack Soundness*

Proposition 3: *Let ADV be a corrupted node that sends a message RequestData off-chain to interact with* n_s *without passing through the authorisation step in the blockchain. We assume that ADV has no valid access* Decision = *Permit for the requested target in the* PEPSC. *Therefore, ADV fails to launch the Reuse Permit Decision Attack.*

Analysis: ADV launches Reuse Permit Decision Attack by performing a request to the n_s consisting of a RequestData message encrypted with public key of storage PK_s and signed by the PK_{ADV}. The n_s decrypts the message and runs PEPSC:VerifyDecision to check if ADV has a valid permit Decision, for the requested target and associated with the PK_{ADV}. Even if there is valid permission, the input arguments in the RequestData message must be the same as the target associated with the valid Decision. Thus, ADV cannot reuse valid permit access given to a different $Resource_{ID}$ or perform other actions than were permitted. Moreover, the PEPSC:Verify Decision function compares the timestamp of the message against the permission expiration time. Thus ADV cannot use an old and expired access permission either. Therefore, ADV cannot successfully perform a Reuse Permit Decision Attack.

6.6 IMPLEMENTATION

We have implemented a prototype of SmartAccess in Solidity [140] language and deployed and tested the prototype in an Ethereum Virtual Machine (EVM)-based blockchain system. Below we describe the implementation of the SC of the SmartAccess, the representation of the XACML policies and attributes using the Solidity language and how the policy evaluation is done.

We have implemented all the SC and each component described in Table 6.3 and Fig. 6.2 using Solidity language. The implementation of PEPSC, PDPSC, PAPSC and PIPSC for the two use cases are available on GitHub [141].

6.6.1 *Policy representation*

In the prototype implementation, the policy is represented in two formats: the Policy representation to evaluate the Attributes of the user and logical operations mapped as code to evaluate the ContextAttr in PIPSC.

First, we express the logical size-efficient policy representation using an array of 8 bits, Policy[i], where the index $i = \{0, ..., 7\}$. The controller nodes define a mask that adds meaning for each bit. Each bit of the array represents a rule of the policy. If the bit is 1, n_p must have the attribute, if the bit is 0, n_p is not required to have the attribute. For example, i=0 represents the attribute

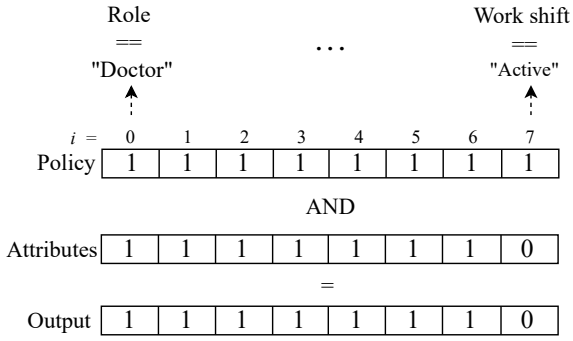


Figure 6.7: Example of a policy mask and Policy AND Attributes evaluation. The output of the evaluation is different from the Policy. Therefore the Attributes do not comply with the Policy.

'Hospital professional', then Policy[0]=1 the user is a hospital professional and Policy[0]=0 means the user is not.

To evaluate the policy, the n_p needs an $Auth_p^c$ to perform an access attempt, which contains an array of 8 bits as the logical representation of the Attributes. Logically, the same mask used for the Policy is used to represent the Attributes. During PDPSC:EvaluateRequest, the decision process executes an AND logical operation between the two arrays, Policy and Attributes. The AND operation between two arrays yields '1', where both indexes are '1'. Thus, n_p needs to have the '1' in the same position in the Attributes as in the Policy.

Fig. 6.7 shows an example of policy mask, where i=0 is the rule *Role == "Doctor"* and i=7 is the rule *Work shift == "Active"*. Moreover, it shows an example evaluation of a policy using the array AND operator. Since the output is different from the Policy, the Decision is Deny.

Second, each ContextExp requires additional evaluation of ContextAttr on PIPSC, as shown in the Section 6.4. For the ContextExp logical operations are mapped as code in PIPSC. For that, PIPSC have additional storage to save, get and delete ContextAttr for each ContextExp.

6.7 EVALUATION

Here we present an initial evaluation of SmartAccess using the prototype (Section 6.6). We started by presenting the setup of the blockchain networks, then we deployed the SC and ran experiments to assess the implementation regarding performance, latency, size and scalability.

6.7.1 *Experimental Setup*

The experiments were conducted on a Macbook Pro with M1 processor and 16GB of RAM. We created three private blockchain networks for the experiments using GoQuorum [142] with three different voting-based consensus mechanisms (Instabul Byzantine Fault Tolerant (IBFT) [143], Quorum Byzantine Fault Tolerant (QBFT) [144] and Reliable, Replicated, Redundant, And Fault-Tolerant (RAFT) [145]). Each network has five n_c, and each one mines a block in the blockchain with an interval of one second. To create and send transactions in the network, we used the workloads from ChainHammer framework [146]. ChainHammer sends multiple transactions using the JSON-RPC 2.0 specification, which is a lightweight remote procedure call (RPC) protocol. The blockchain receives JSON-RPC calls and executes whatever is sent as an argument inside the request in JSON format. We had to adapt the ChainHammer code to send transactions that execute the SC functions of SmartAccess and that yield the results in the format defined by SmartAccess. The adapted scripts can also be found in the Github [141] repository.

6.7.2 *Experiments*

We evaluated the transactions that execute the most crucial functions on SmartAccess, namely PEPSC:RequestAccess
and PEPSC:VerifyDecision. These functions are the most crucial since they are required for any access attempt using the SmartAccess. PEPSC:Request Access is the most complex function because of the number of logical operations and calls to other SC functions. PEPSC:VerifyDecision is simpler and similar to all the other functions of SmartAccess. Moreover, we compared the results of PEPSC:RequestAccess and PEPSC:VerifyDecision with a baseline smart contract that implements two basic functions: one sets a number to a variable (Baseline SET), and the other retrieves that number (Baseline GET). These baseline functions enable us to compare the performance of SC from SmartAccess against the simplest operations possible with a smart contract, in the same network.

We used a different consensus mechanism for each network setup: IBFT, QBFT and RAFT. These are options of consensus mechanisms for private network blockchains. The IBFT and QBFT are Byzantine Fault Tolerant (BFT) consensus, which means that they will reach consensus in the network even if some nodes fail to respond or respond with incorrect information. RAFT is Crash Fault Tolerant (CFT) which means that the network supports nodes failing but fails to recognise malicious behaviour. Then, for each network, we repeated four

experiments to measure: (i) performance, (ii) latency, (iii) size and (iv) scalability.

The performance experiments measure the number of transactions per second (TPS) and the gas consumed to run the transactions. TPS indicates the number of transactions the blockchain system can handle per second. The gas consumed indicates the complexity of the function. We calculated TPS from the average of transactions that were mined on each block. The gas used to run the transactions was retrieved from the transaction receipt in the blockchain.

The latency experiment measures how long it takes, after creating a transaction, for the transaction to be mined and become available on the blockchain. This experiment indicates how long it would take for a user to pass through the access control (PEPSC:RequestAccess and PEPSC:VerifyDecision). The transaction latency is calculated as the difference between the timestamp of the block in which the transaction was mined and the transaction's timestamp.

The size experiments measure the amount of data generated by the transactions and the mined blocks. This experiment provides insight into what to expect regarding growth and storage usage from the blockchain system utilising SmartAccess.

Finally, the scalability experiments measure how the TPS behaves for an increasing number of rules, attributes and contextual attributes that need to be verified during the policy evaluation process. We ran experiments with 64, 128 and 256 attributes and 1, 10 and 100 contextual attributes.

For each experiment, we ran 11 rounds, sending 5000 transactions to each of the three blockchain networks. The first round was used as a warm-up, and the results of this round were excluded from the final analysis. In total, 55000 transactions were created to run the four functions of the smart contracts (PEPSC:RequestAccess, PEPSC:VerifyDecision, Baseline SET and Baseline GET).

An overview of average results is presented in Table 6.4. Fig. 6.8 shows TPS results for the performance experiment. TPS results were similar among the different networks, although RAFT consensus performed slightly better on average. This result is expected since the RAFT consensus performs fewer verifications and is not Byzantine fault tolerant (BFT). The results show that PEPSC:RequestAccess has significantly lower TPS than the baseline contracts. This TPS decrease is also expected since PEPSC:RequestAccess performs several steps to yield a decision. Note that this is different for PEPSC:VerifyDecision, which only needs to verify the access Decision. Overall, the average TPS was 250 for PEPSC:RequestAccess and 290 for PEPSC:VerifyDecision. Furthermore, in table 6.4 we can see a high increase in the gas usage for the PEPSC:RequestAccess, which shows that it performs more computational procedures, while losing around 15% in TPS compared to the baseline functions.

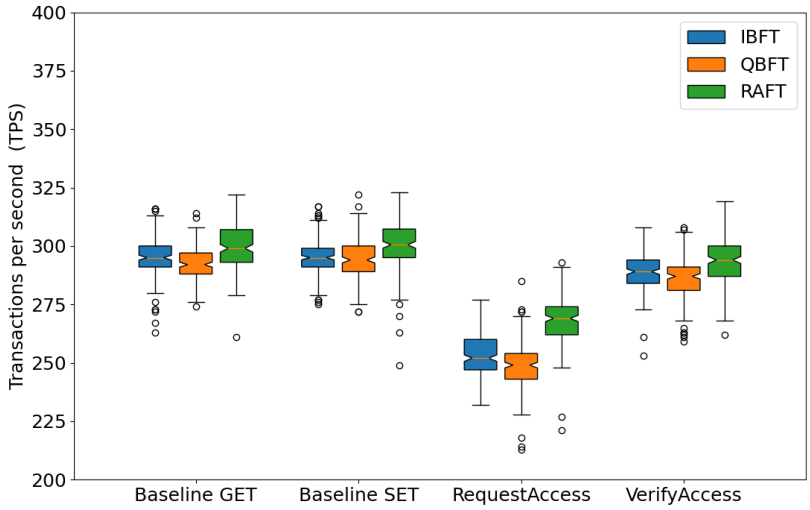


Figure 6.8: Transactions Per Second (TPS) of the baseline and SmartAccess functions with different blockchain network setups (IBTF, QBTF and RAFT consensus).

Table 6.4: Experiment results regarding performance, latency and size for the baseline contracts and the SmartAccess functions with different blockchain deployments. Average for 10 runs with 5000 transactions each. Tx=transaction, TPS=transactions per second.

	Function	Performance			Latency	Size	
		Tx gas	Block gas	TPS	Time (s)	Tx(KB)	Block(KB)
IBFT	Baseline GET	21937	6477k	295	2.00	0.944	41.05
	Baseline SET	26798	7907k	295	2.00	0.946	41.03
	PEPSC:RequestAccess	164122	38843k	252	1.99	6.139	122.73
	PEPSC:VerifyDecision	31765	9141k	288	2.00	1.531	112.83
QBFT	Baseline GET	21937	6370k	292	2.00	0.944	40.49
	Baseline SET	26798	7805k	293	2.00	0.946	40.70
	PEPSC:RequestAccess	164122	40808k	249	1.98	6.139	120.86
	PEPSC:VerifyDecision	31765	9003k	285	2.00	1.531	111.52
RAFT	Baseline GET	21937	6529k	299	1.02	0.944	41.20
	Baseline SET	26798	7975k	300	1.02	0.946	41.28
	PEPSC:RequestAccess	164122	44116k	269	1.04	6.139	130.29
	PEPSC:VerifyDecision	31765	9182k	290	1.02	1.531	113.35

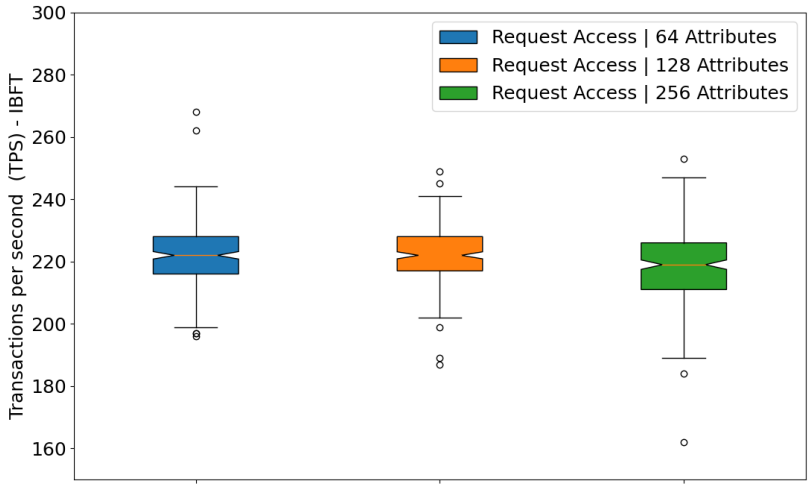


Figure 6.9: Transactions Per Second (TPS) for an increasing number of attributes that are verified for policy evaluation (IBTF consensus network).

The latency results are expressed in seconds in table 6.4. For IBFT and QBFT consensus, the latency is, on average, approximately 2s (1 second spent for each block required to mine the transactions). The results using RAFT consensus show that it takes approximately 1s (one block). There were no differences in latency for the different functions of the baseline SC and SmartAccess contracts.

Table 6.4 shows the average sizes of transactions and blocks for IBFT, QBFT and RAFT consensus. The transactions to execute PEPSC:RequestAccess generate more data than the transactions to run the baseline contract functions. Moreover, from the transactions and block sizes, the growth of the network using the SmartAccess contracts is expected to be bigger than the baseline, following a linear growth.

In the scalability experiments, we deployed only IBFT consensus because byzantine fault tolerant consensus are more secure and widely adopted in private blockchain systems. Moreover. the results for IBFT and QBFT were similar. The results are presented in Figures 6.9 and 6.10. Fig. 6.9 shows the transactions per second (TPS) considering a verification of a Policy and Attributes with 64, 128 and 256 Attributes. The results show that, despite increasing the number of attributes, the TPS remains the same, hence providing scalability regarding the attributes supported by SmartAccess. Fig. 6.10 shows the TPS considering a policy that requires evaluation of 1, 10 and 100 contextual attributes. The results show that the TPS is inversely proportional to the number of ContextAttr. On average, the TPS for 1, 10 and 100 ContextAttr is 225, 213 and 101. For 100 ContextAttr, the TPS decreases more than 50%; however, it supports 100 times more attributes, handling more sophisticated access control policies.

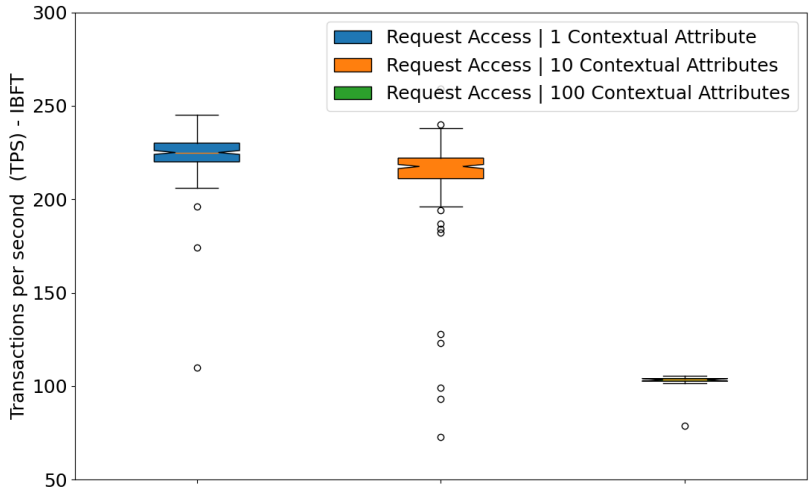


Figure 6.10: Transactions Per Second (TPS) varying the number of contextual attributes that are verified during the access request.

6.7.3 *Comparison with the start-of-the-art*

For the experimental comparisons with state-of-the-art, we could not obtain any open-source code for the related work solutions presented in Section 6.2.3. Nevertheless, we performed a comparison of performance, size and latency between our proposal and two access control functions that are Solidity-based and available as open-source. One is an implementation of the Role-based Access Control Smart Contract (RBACSC)[147], and the other is an implementation of the Access Control List Smart Contract (ACLSC)[148]. In the experiments, we only used the function PEPSC:RequestAccess, because it is analogous to the functions from RBACSC and ACLSC that evaluate a request for data access. Note, however, that these functions, in fact, are different in the three access control solutions. In particular, RBACSC and ACLSC evaluation is much simpler than SmartAccess, which supports complex ABAC policies. Table 4 describes the smart contract functions from RBACSC and ACLSC that we use to perform experiments in comparison with PEPSC:RequestAccess.

RBACSC defines a policy-neutral access-control mechanism based on roles and privileges, where the permissions to perform certain operations are assigned to specific roles. RBACSC implements multiple functions for access control per role. The RBACSC:OnlyAdminCanViewThis function performs access control before executing an action, similarly to PEPSC:RequestAccess. For example, the action could be to read a file. RBACSC:OnlyAdminCanViewThis checks if the sender (requester) has an adequate role before executing the action, in this case, the role of 'administrator'.

ACLSC implements a list of permissions associated with a system resource. The list specifies which users or system processes are granted access to which resources and what operations are allowed on a given resource. ACLSC implements various functions for access control using different lists of permissions. We chose the ACLSC:GetResourceMetadata function for comparison with PEPSC:RequestAccess. The chosen function returns the metadata of a given resource stored in the smart contract storage, and its execution is protected. When this function is called, the contract checks whether the sender (requester) belongs to the group of members with permission to execute ACLSC:GetResourceMetadata. The resource model is defined by another smart contract.

Conceptually, both RBACSC and ACLSC implementations only protect the execution of functions inside the smart contract, therefore for resources stored on-chain. This is similar to the PEPSC:RequestAccess function, which requests permission to process a resource and saves the decision in the smart contract PEPSC. The difference between our proposal and the other two functions is that SmartAccess protects resources stored off-chain, and only the access control is performed on-chain.

Similarly to the experiment's Section 6.7.1, we have deployed the same Quorum network running the IBFT consensus to run analogous PEPSC, RBACSC and ACLSC smart contract functions. For SmartAccess contracts, we used Solidity version 0.8.21, while RBACSC used version 0.4.23, and ACLSC used version 0.8.1. The network has the same number of nodes (5) and time between mining blocks (one second). We ran 11 rounds (1 round for warm-up), sending 5000 ACLSC:GetResourceMetadata and RBACSC:OnlyAdminCan ViewThis transactions to the network. Similarly to the previous experiment, we also measured the performance, latency and size to compare with the PEPSC:RequestAccess from figure 8 with IBFT consensus. See Section 6.7.1 for more details about the experimental setup.

An overview of the average results is presented in 6.5. Fig.6.11 shows the TPS results from the three access control requests. We observe that PEPSC:RequestAccess had a higher TPS median result and a more stable distribution range of results in [232, 277] TPS, in comparison with RBACSC:OnlyAdminCanDoThis and ACLSC:GetResourceMetaData, which presented a dispersion in the results from [167, 267] and from [176, 274] TPS, respectively.

Table 6.5 also shows the results for the gas, latency and size of the transactions and mined blocks. The gas used by PEPSC:RequestAccess is much higher than that used by RBACSC:OnlyAdminCanDoThis and ACLSC:GetResource MetaData. This is expected since the last two functions verify only a single value before yielding the decision. PEPSC:RequestAccess, however, performs extra steps such as verifying the authentication token and evaluating more complex policies, including contextual attribute verification. The size of the transactions

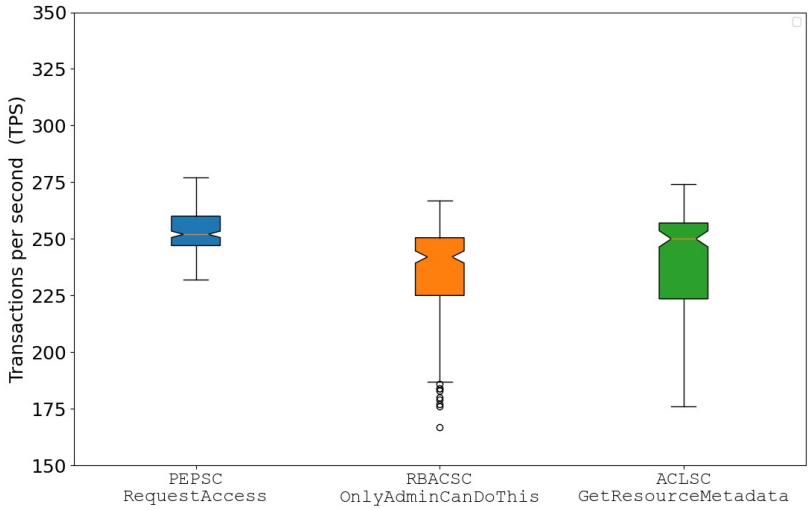


Figure 6.11: Transactions Per Second (TPS) of the three access request functions of PEPSC, RBACSC and ACLSC in 10 runs.

and blocks is also higher for PEPSC:RequestAccess because of the complexity of the transaction, which requires more input parameters and performs more complex operations. However, the latency between the generation of the transaction and the transaction being mined followed the previous results and stayed around 2 seconds for every function because it is bounded by the IBFT consensus mechanism.

6.8 DISCUSSION

Access control systems usually keep their policies secret, arguing that secrecy helps protect the system, the so-called 'security through obscurity'. Nowadays, the practice is frowned upon by the necessity of transparency and auditability to enhance trust in the system. SmartAccess is designed to be transparent to

Table 6.5: Experiments results regarding performance, latency and size for the comparison with PEPSC, RBACSC and ACLSC. Average for 10 runs with 5000 transactions each. Tx=transaction, TPS=transactions per second.

Function	Performance			Latency	Size	
	Tx gas	Block gas	TPS	Time(s)	Tx (KB)	Block (KB
PEPSC:RequestAccess	164122	38843k	252	1.98	6.139	122.73
RBACSC:OnlyAdminCanDoThis	23260	5448k	234	2.00	1.409	25.44
ACLSC:GetResourceMetadata	21561	5171k	239	2.00	0.946	26.03

every network node. We consider that the transparency supported by smart contracts enhances trust in the system and enables real-time auditing of the data processing agreements. Every new policy, or modification in the policies regarding a shared resource, is transparent and is validated through consensus by all nodes.

The evaluation indicated that our access control runs in a reasonable time window, even for the increasing complexity of the policies due to a larger number of attributes to be evaluated. Considering the network setup with IBFT consensus, the overall latency to access the data is 4 seconds. In the case of using contextual attributes, it needs two extra seconds to add the ContextAttrin PIPSC, increasing the overall access control to around 6s.

We acknowledge that our experiment and evaluation have the limitation of running in a notebook with limited resources. We then ran baseline contracts to be the base of comparison for our results. Despite the decrease in performance compared to the baseline contracts, the average throughput of 220 TPS is reasonable to handle an inflow of access control requests. For example, for the acute care use case in the Netherlands, it is expected that roughly 110 emergencies per day [149]. Even in the worst-case scenario, where the 110 emergencies and access requests happened simultaneously and with 100 contextual attributes to be verified by SmartAccess, our experimental setup would handle all the 110 PEPSC:RequestAccess and 110 PEPSC:VerifyDecision transactions in a short latency time. The growth of the blockchain for the 110 accesses would be less than 1 MB per day.

Although the functionalities of PEPSC:RequestAccess being more complex than RBACSC:OnlyAdminCanDoThis and ACLSC:GetResourceMetadata, the TPS of PEPSC:RequestAccess was significantly higher. We hypothesise that this difference might be explainable by the differences in implementations between the contracts and the version of the compiler. The newest compiler version used by SmartAccess has better optimisations than the previous ones. Nevertheless, the results for gas and sizes were consistent with the different complexity of the functions.

Regarding the transparency inherent to the system, some privacy concerns appear firstly because the resources may contain personal data, and secondly, because the transactions may also contain some personal data of the professionals or patients. The off-chain resource containing personal data must be encrypted, and the identifiers of the encrypted files $\mathrm{Resource}_{\mathrm{ID}}$ are related to $\mathrm{Subject}_{\mathrm{ID}}$. The EMR application running on top of the SmartAccess system is responsible for having the relationship between the patient identifier and the $\mathrm{Subject}_{\mathrm{ID}}$ of the resources. Thus, when the n_p sends the request transaction, it only contains the $\mathrm{Subject}_{\mathrm{ID}}$ and $\mathrm{Resource}_{\mathrm{ID}}$, keeping the patient pseudo-anonymous in the blockchain. We designed SmartAccess to address all nodes using the pseudo-

anonymous addresses, which means that a person is linked to a $Addr_p$ and PK_p, but no actual name or the real-world identity is known unless the n_c which added the n_p to the network. It is known as pseudo-anonymous addresses because the n_c must know the processor's real identity, as the organisation knows the identification of its employee. The same applies to the patients, who must be added to the network by a n_c. We assume that a national organisation could represent the n_c and be responsible for the authentication process of the patients' n_p, keeping their real identifiers private from the blockchain. For future work, we will investigate using a federated identity system for blockchain, such as Self-Sovereign Identity from Hyperlegder Indy [150]. The pseudo-anonymous address also means that a combination of attributes available in the requests could be used to de-identify the person behind the node when combined with other sources of personal data. We consider it as a drawback of our proposal. However, any other system that uses pseudo-anonymous identifiers is subject to de-identification and privacy compromise. Thus we believe that using pseudo-anonymous addresses is a fair trade-off when we accomplish a transparent and auditable system.

We also assume that the usage of cryptography to protect the resources in the storage is necessary. Others have proposed innovative mechanisms that combine modern cryptographic schemes to protect data confidentiality with double-layer encryption, one for the data and the other for the cryptography keys [33]. For example, in our previous work [98], we used Dynamic Symmetric Searchable Encryption (DSSE) to enable the search of encrypted data combined with Attribute-based Encryption (ABE) to protect the DSSE keys used to encrypt the data. To interact with the DSSE and ABE, the system first performs access control using a centralised ABAC system. SmartAccess could be used as the access control layer (ABAC) above the cryptographic layers (DSSE and ABE) instead of only protecting access to the data. In future work, we intend to extend SmartAccess to replace the centralised ABAC solution.

Some blockchain proposals to store data processing logs rely on the assumption that n_s will send transactions to the blockchain only to record logs in there [129]. We consider this assumption unrealistic because there is no motivation for n_s to send a transaction for every request-response out of the blockchain. Therefore, SmartAccess logs are only about the access control requests and decisions done by the system. The logs related to data processing are recorded and maintained by the n_s. During SmartAccess design, we could have opted to save information regarding the decision inside the payload of events inside each transaction. However, this way, the n_s would read the decision through the transactions without interacting with the smart contracts, and the verification logs would not be recorded. Because of that, we decided not to save the decision as payload; instead, we added PEPSC:VerifyDecision function

where n_s executes the function to read the save decision inside the PEPSC. This way, n_s cannot know the decision unless it executes the function and generates the verification logs accordingly.

Finally, our solution adds complexity compared with ABAC implemented as a centralised service. However, current centralised solutions lack transparency and trust when sharing medical data across organisations. Moreover, we defined SmartAccess as an access control system to protect an EMR system with modular characteristics, making the system more flexible to changes for specific cases. All the SmartAccess requests have a clear purpose of processing defined by ContextExp. All the logs are associated with the purpose, which enables auditability across organisations and GDPR compliance. The two use cases described in this paper illustrate what can be designed using the SC. In future works, we will investigate how the SmartAccess can fit more use cases outside the healthcare scenario. Moreover, we will compare SmartAccess with traditional centralised access control mechanisms.

6.9 CONCLUSION

This paper proposes SmartAccess, an Attribute-Based Access Control System for medical records sharing. The solution enables cross-organisation joint agreement over access policies, dynamic access control, transparency, and auditability. SmartAccess leverages the Attribute-Based Access Control model to implement four smart contracts that mimic the granularity of the model. We presented SmartAccess applied in two healthcare use cases: access with patient consent and access with implicit consent during acute care. We analysed the threat model and performed a security analysis in the initial evaluation. Through a proof-of-concept implementation of the SC, we demonstrated the feasibility of our proposal by analysing the complexity and scalability of the functions of the contracts and; by measuring the latency and throughput of the transactions to execute the access control functions with different blockchain network consensus setups. The results display expected overheads when executing the SmartAccess functions from the SC. Finally, our implementation's source code [141] is open to the community, which can facilitate further research leading to the adoption of SmartAccess in future applications.

7

DISCUSSION

Ideally, a digital transformation of the healthcare sector will happen when secure, complete and unambiguous electronic medical records (EMR) are available and shareable. EMR are generated and accessible by multiple healthcare professionals who treat the patients during their lifetime, despite the geographic location, healthcare organisation or specialisations [151]. The patient's privacy is always preserved, and no access is allowed unless necessary or if the patient consents to it. Finally, the knowledge that comes from the EMR is used to gain operational efficiency and consequently improve clinical care [152].

Nowadays, the Netherlands has no integrated EMR system, and the patients' EMR are spread and fragmented in different organisations where patients were treated at least once. According to the Ministry of Health, Welfare and Sport of the Netherlands, taking care of people includes taking care of their personal data. All people have the right to respect their privacy, therefore, protecting personal data and processing data in accordance with the law are key elements to enable cross-organisational data sharing [153]. Even without an integrated EMR system, in this thesis, the goal was to use encryption schemes and access control models for potential integrated EMR systems of the future. The thesis presents communication protocols for cross-organisation data sharing in compliance with the general data protection regulations.

Furthermore, the current EMR systems use break-glass procedures to enable data availability in case of emergencies without appropriate authorisation protocols. Another goal of this thesis was to demonstrate that the novel security mechanisms address availability in more secure ways. Therefore, the break-glass procedures would be rare and only used in extreme situations because the EMR system would provide an adequate access control mechanism that supports lawful data access without compromising patients' privacy. In the thesis, the algorithms with the break-glass name refer to the procedure of starting an emergency session for the patient, are retricted for emergency teams and are followed by access control policies. Thus, in case of emergency, do not just break the glass but use a secure mechanism to protect and share data among organisations during acute care.

This thesis presented encryption and access control protocols for cross-organisation data sharing inspired by the acute stroke care case. Our proposals are,

however, applicable and generalisable to other acute care cases. Furthermore, the proposed protocols could also be extended to conventional non-acute cases in which the patient consents to data processing. Specific adjustments, such as policies and attributes, would need to be defined according to the needs and purposes of data access to support additional use cases.

7.1 ANSWERS TO RESEARCH QUESTIONS

The main contributions of this thesis are technical solutions proposed to answer the following research questions.

RQ1: How to enable secure data sharing of confidential patient data stored on untrustworthy cloud-based EMR systems during acute care?

To answer this question, we proposed two protocols through which acute care teams can share confidential EMR. Both proposals aimed to achieve the following: First, to enable access to the patient's EMR to the members of all acute care teams involved in treating the patient. Second, to ensure that health professionals may access EMR if and only if they have a legitimate role on the patient's treatment team. Third, to dynamically grant and recoke access to EMR as demanded by the patient's treatment. Finally, to revoke access without requiring re-encryption and any impact on other legitimate users.

In chapter 2, we proposed the Red Alert Protocol (RAP) as our first answer to RQ1. RAP adopted Ciphertext-Policy Attribute-Based Access Control (CP-ABE) to protect the patient's EMR in a cloud solution. The data is available for all the teams from multiple organisations. The defined policy requires that each involved team proves its participation in the emergency care to the Master Authority (MA), which is a secure entity running in a trustworthy environment and responsible for access control. This is done when the team jointly solves a challenge: the healthcare entity, represented by an attested smart device, and at least two professionals respond to the challenge, proving that they are co-located and working together. After the challenge is solved and the users' attributes are validated, MA generates a CP-ABE emergency key that satisfies the policy. However, direct sharing of the CP-ABE emergency key is not secure enough because getting access to the key would allow anyone to access the patient's EMR at any future moment. Therefore, the MA also generates an access control token for the team. This token has a default expiration time and contains the identity of the professionals in the treatment team. These process the data and interact with the cloud provider using the token and the key. Access

revocation of a team happens as soon as the patient is no longer under the care of this team. To do so, the MA needs to be informed about the current phase of the treatment the patient is, by either a check-in or check-out message from the hospital. Both messages are sent by the attested smart devices of each treatment team and include a timestamp. Thus, even if the token is still valid according to the default expiration time, the cloud provider does not allow any type of access to the data after the revocation.

RAP enables healthcare professionals to decrypt a patient's encrypted EMR by using time-based tokens that are issued during an emergency. After the expiration of the tokens, the users are revoked and can no longer access the patient's EMR. The security of RAP was proven using both simulation-based security analyses as well as analyses of direct attacks on the protocol. Finally, we proved that the execution time for the RAP core functions is acceptable in an emergency situation since the approximate sum of execution times of the RAP functions is below 0.5 seconds. However, the performance of the CP-ABE for encryption and decryption functions depends on the size of the policy, the values and number of attributes attached to the user's key and the size of the EMR files. For example, suppose we use CP-ABE protocol to encrypt and decrypt big files like computed tomography (CT) images. In that case, the execution time increases and potentially compromises the availability of the data.

The scalability limitation of RAP motivated us to propose another protocol presented in Chapter 3. The Access Control for Acute Care (AC-AC) protocol overcomes RAP limitations using a hybrid encryption scheme to protect data and manage access control to the shared data. In AC-AC, data are encrypted using the symmetric key from the Dynamic index-based Symmetric Searchable Encryption (DSSE) scheme before being sent to the cloud provider. The management and sharing of this key are done under the CP-ABE scheme, where the DSSE key is encrypted under policies and decrypted with the secret CP-ABE key generated if the user's attributes match the policies, similarly to RAP. Moreover, we proposed to use an additional mechanism for defining fine-grained access rights coined 'scope', which are controlled by the Revocation Authority (REV). The scope is a one-dimensional array of four bits that represent the access rights (i.e., view, add, delete and revoke access) assigned to the user for each data collection encrypted under the DSSE key. The scope is updated according to the timestamps of the exchanged messages. For example, a message notifying the system that the ambulance has delivered the patient to the hospital. From this time, the right to read data is revoked for the ambulance team. We proved that the AC-AC protocol is resilient to multiple attacks. Moreover, we analysed that the time complexity grows linearly with database size. Furthermore, the expected execution time of the AC-AC algorithms used during an

emergency session is acceptable in an acute care timeline. The most extended algorithm of the protocol had an execution time below 170 ms, and the other algorithms are even faster and should not affect access to EMR or delay revocation.

Although the AC-AC protocol offers an additional access control mechanism on top of the encryption scheme, it only controls the type of actions permitted, considering the timestamps of the messages. We also know that the other contextual attributes could be used for legitimate access during an emergency. These attributes are dynamic, e.g. the professional's work schedule and the IP addresses, but the ABE scheme does not support such dynamic attributes.

RQ2: How to model a dynamic and fine-grained access control mechanism to secure patient data during acute care?

In chapter 4, we proposed a context-aware attribute-based access control model for data sharing during acute care, called AC-ABAC, for a cloud-based EMR system. AC-ABAC implements access control policies that consider contextual attributes for dynamically sharing patient data with healthcare professionals during the timeline of acute care. The AC-ABAC runs in an authorisation engine at a system level, handling all the access requests and enforcing the appropriate policies. We followed a methodology to understand the correct rules and available attributes we could use to model the access control without interfering with the healthcare workflow. The methodology has five phases: preparation, analysis, development, policy definition and policy enforcement.

The preparation phase of the methodology facilitated collecting the requirements and understanding the stakeholders in a structured manner imposed by the templates. In the analysis phase, we had to make a choice of contextual attributes to be used to create the access control policies. To guarantee the availability of a patient's EMR at all times, we decided to leave out contextual attributes regarding location, such as GPS coordinates and IP addresses. The iterative approach adopted in the development phase helped refine realistic rules and contextual attributes. Finally, during the policies enforcement phase, we observed that race conditions regarding outdated security tokens might occur. Such inconsistencies could be minimised by regenerating tokens frequently after modifying the team composition.

Furthermore, we highlight that the defined policies can dynamically change at run-time without any need to re-compile or restart the authorisation engine. For example, we could have a shortage of ambulance services during a pandemic or a natural disaster. In such a case, the paramedic teams of the military

forces could provide emergency response. The AC-ABAC model could be updated on-the-fly with a new policy to add the military teams without compromising the rest of the operational policies in the system.

It is essential to collect the perception of healthcare professionals to gain user input early in technology development to improve the acceptability of applications according to users' needs. To demonstrate the secure mechanisms proposed to answer RQ1 and RQ2, we developed a cloud-based EMR system prototype with an acute stroke care application that combines the AC-AC protocol and AC-ABAC model, presented in chapters 3 and 4. In chapter 5, we presented the results of a qualitative interview with healthcare professionals with prominent roles in acute care in the Netherlands. First, we showed the use of the prototype during simulated acute stroke care, showing through the user interfaces the access being granted and revoked to the teams according to their tasks in the patient's treatment. We used in-depth interviews to capture their perspectives on the application design, its functions and its use in a simulated acute care event. Although our study was designed in the context of a specific project, the challenges for developing an EMR system that supports acute care and the collected feedback about cloud-based systems are possibly applicable in a broader context.

The results of the interviews reinforced that the most relevant challenges for patient data sharing are the lack of interoperability and connectivity between systems from different organisations. Moreover, this study got relevant feedback from every interviewee regarding the period for data availability, accountability, prevention of data loss and how to handle unknown patients during acute care. The interviews also aimed to validate the security concepts of a cloud-based medical data-sharing application for acute stroke care. During the interviews with healthcare professionals, it became evident that they experience the lack of a properly connected and secure information infrastructure for patient data exchange across organisations. The cloud-based EMR system was well received and considered relevant by all. However, as many non-interoperating systems are used in practice, replacing them with a new system – like the developed application – did not seem realistic to the interviewees. An alternative path to be explored involves developing an interoperation layer for cloud-based security and trustworthy data exchange that could bridge legacy systems with the newly AC-AC protocol and AC-ABAC model.

RQ3: How to facilitate data sharing across multiple organisations by providing means to define joint access control policies and enforcement mechanisms in a transparent and auditable manner?

Inspired by the feedback about the need for trustworthiness among the healthcare organisation presented in Chapter 5, we proposed in Chapter 6 the Smart Access system — an attempt to decentralise the access control to patient EMR, which are already decentralised in multiple EMR systems. In Smart Access, healthcare organisations jointly agree on access policies and dynamic access control over shareable data using a blockchain solution. SmartAccess is based on smart contracts to implement the ABAC model for dynamic data access control across organisations. Data processing is only allowed if the healthcare professional runs the smart contracts and has the right attributes to comply with the policy rules. Therefore, policy decisions and enforcement do not depend on a centralised server. Finally, every function executed in the smart contracts generates auditable transaction logs published in the blockchain. Through a proof-of-concept implementation, we experimented with the execution of smart contracts. The consensus mechanism determined the wait time to validate the data access permission. Assuming an IBFT consensus network configuration, the total latency added by Smart Access to process the request and authorise data access is 4 s.

Regarding the transparency inherent to the system, some privacy concerns appear. Firstly, the resources may contain personal data, and secondly, the transactions may also have the personal data of the professionals or patients. To reduce privacy concerns, we designed Smart Access to address all nodes using pseudo-anonymous addresses. No actual name or real-world identity is known except for the organisation that added a professional to the network. The same applies to patients, who must be added to the network by a healthcare organisation. We assume that a national organisation could be responsible for the authentication process of the patients, keeping their real identifiers separated from the blockchain. However, using pseudo-anonymous addresses is not always sufficient, considering that the users' attributes available in the request can be combined with other sources of personal data, therefore identifying the person behind the pseudo-anonymity. We consider the possible combination of attributes to de-identify the person as a drawback of our proposal. However, any other system that uses pseudo-anonymous identifiers is subject to de-identification and privacy risks.

Finally, Smart Access adds complexity compared with AC-ABAC, which is implemented as a centralised service. However, current centralized solutions lack transparency about data processing within the organization. Consequently, they do not provide proof of compliance with regulations[19]. Then, the lack of trustworthiness has been one of the limitations when sharing medical data across organisations. Therefore, although the complexity of a distributed solution is more significant than a centralised solution, it brings as a trade-off the inherent transparency of the system regarding the data processing, the joint

agreement about the access control policies and the potential trustworthiness built among the organisations.

7.2 FUTURE PERSPECTIVES

One of the reasons that the healthcare industry is behind in digitalisation compared to other sectors is the lack of understanding and trust in the technologies by healthcare professionals. Vice-versa, engineers and computer scientists need to comprehend the complexity and requirements of healthcare systems, resulting in incomplete or far-fetched solutions. Therefore it takes quite some awareness, education, and even convincing effort to make new solutions come through in healthcare systems. Open communication with EMR stakeholders, as a diverse audience, could make innovative technical solutions be comprehended and used to solve problems.

I envision transparent and auditable healthcare data systems where the healthcare organisations, which are the data controllers, can build reputation and trustworthiness among themselves. Moreover, I imagine healthcare institutions as safe environments where the patients' data face fewer privacy risks, and healthcare professionals can use the best technology innovations to achieve their tasks to the fullest.

In future works, I will keep investigating the use of blockchain technology and smart contracts to provide distributed management over a shared data source among healthcare organisations. More precisely, I will continue the research done in Smart Access, in Chapter 6, to expand the healthcare use cases and the tests of performance and scalability. Moreover, I will research tools for reading and visualising the logs generated in the SmartAccess blockchain to build a user-friendly dashboard. This dashboard will enable interpretation and consolidate the information and knowledge from the access control and data processing logs, which is very important for transparency, audibility and security awareness. Furthermore, I will research the use of Smart Access solutions to fit use cases outside the healthcare scenario. Smart Access can be helpful in many scenarios where there are joint data controllers and no trustworthy third party available to intermediate the transactions [154]. Another interesting research is to evaluate the privacy risks that come from the transparency of blockchain and how to minimise it. In future research, I will investigate the potential use of blockchain Crosschain Transactions [155] and zero-knowledge proof theories [156, 157] to develop privacy-preserving applications for cross-organisation data sharing.

BIBLIOGRAPHY

[1] Richard Hillestad, James Bigelow, Anthony Bower, Federico Girosi, Robin Meili, Richard Scoville, and Roger Taylor. "Can electronic medical record systems transform health care? Potential health benefits, savings, and costs." In: *Health affairs* 24.5 (2005), pp. 1103–1117.

[2] John Adam Oostema, Mojdeh Nasiri, Todd Chassee, and Mathew J Reeves. "The quality of prehospital ischemic stroke care: compliance with guidelines and impact on in-hospital stroke response." In: *Journal of Stroke and Cerebrovascular diseases* 23.10 (2014), pp. 2773–2779.

[3] Adil Hussain Seh, Mohammad Zarour, Mamdouh Alenezi, Amal Krishna Sarkar, Alka Agrawal, Rajeev Kumar, and Raees Ahmad Khan. "Healthcare data breaches: insights and implications." In: *Healthcare*. Vol. 8. 2. Multidisciplinary Digital Publishing Institute. 2020, p. 133.

[4] *U.S. Department of Health and Human Services Office for Civil Rights Breach Portal: Notice to the secretary of HHS breach of unsecured protected health information*. URL: https://ocrportal.hhs.gov/ocr/breach/breach_report.jsf.

[5] Trustwave. *2020 Trustwave Global Security Report*. URL: https://www.trustwave.com/en-us/resources/library/documents/2020-trustwave-global-security-report/.

[6] *Cost of a Data Breach 2022 Report IBM*. URL: https://www.ibm.com/reports/data-breach.

[7] Accountability Act. "Health insurance portability and accountability act of 1996." In: *Public law* 104 (1996), p. 191.

[8] *European Parliament and Council of European Union Regulation (EU) 2016/679*. https://gdpr-info.eu. Accessed: 16/12/2020. 2016.

[9] *GDPR Art. 6 Lawfulness of processing*. https://gdpr-info.eu/art-6-gdpr/. Accessed: 10/11/2021. 2016.

[10] Anna Ferreira, Ricardo Cruz-Correia, Luis Antunes, Pedro Farinha, E Oliveira-Palhares, David W Chadwick, and Altamiro Costa-Pereira. "How to break access control in a controlled manner." In: *19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06)*. IEEE. 2006, pp. 847–854.

[11] Benjamin Stark, Heiko Gewald, Heinrich Lautenbacher, Ulrich Haase, and Siegmar Ruff. "Misuse of 'Break-the-Glass' Policies in Hospitals: Detecting Unauthorized Access to Sensitive Patient Health Data." In: *Research Anthology on Privatizing and Securing Data*. IGI Global, 2021, pp. 1231–1256.

[12] National Institute of Neurological Disorders, Stroke (US). Office of Communications, and Public Liaison. *Stroke: Hope through research*. 99. The Institute, 1999.

[13] Jeffrey L Saver. "Time is brain–quantified." In: *Stroke* 37.1 (2006), pp. 263–266.

[14] Assad Abbas and Samee U Khan. "A review on the state-of-the-art privacy-preserving approaches in the e-health clouds." In: *IEEE Journal of Biomedical and Health Informatics* 18.4 (2014), pp. 1431–1441.

[15] Daisuke Mashima and Mustaque Ahamad. "Enhancing accountability of electronic health record usage via patient-centric monitoring." In: *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*. ACM. 2012, pp. 409–418.

[16] Rui Zhang and Ling Liu. "Security models and requirements for healthcare application clouds." In: *2010 IEEE 3rd International Conference on cloud Computing*. IEEE. 2010, pp. 268–275.

[17] *What is a data controller or a data processor?* https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/obligations/controller-processor/what-data-controller-or-data-processor. Accessed: 10/11/2021. 2016.

[18] *What is a GDPR data processing agreement?* https://gdpr.eu/what-is-data-processing-agreement/. Accessed: 16/11/2021. 2016.

[19] Stephen O'shaughnessy and Anthony Keane. "Impact of cloud computing on digital forensic investigations." In: *Ifip international conference on digital forensics*. Springer. 2013, pp. 291–303.

[20] Marcela Tuler de Oliveira, Alexandros Bakas, Eugene Frimpong, Adrien ED Groot, Henk A Marquering, Antonis Michalas, and Silvia D Olabarriaga. "A break-glass protocol based on ciphertext-policy attribute-based encryption to access medical records in the cloud." In: *Annals of Telecommunications* (2020), pp. 1–17.

[21] Ming Li, Shucheng Yu, Kui Ren, and Wenjing Lou. "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings." In: *International conference on security and privacy in communication systems*. Springer. 2010, pp. 89–106.

[22] Achim D Brucker, Helmut Petritsch, and Stefan G Weber. "Attribute-based encryption with break-glass." In: *IFIP International Workshop on Information Security Theory and Practices*. Springer. 2010, pp. 237–244.

[23] Yang Yang, Xianghan Zheng, Wenzhong Guo, Ximeng Liu, and Victor Chang. "Privacy-preserving smart IoT-based healthcare big data storage and self-adaptive access control system." In: *Information Sciences* 479 (2019), pp. 567–592.

[24] Alessandra Scafuro. "Break-glass Encryption." In: *IACR International Workshop on Public Key Cryptography*. Springer. 2019, pp. 34–62.

[25] Dean Povey. "Optimistic security: a new access control paradigm." In: *Proceedings of the 1999 workshop on New security paradigms*. ACM. 1999, pp. 40–45.

[26] Achim D. Brucker and Helmut Petritsch. "Extending Access Control Models with Break-glass." In: *Proceedings of the 14th ACM Symposium on Access Control Models and Technologies*. SACMAT '09. Stresa, Italy: ACM, 2009, pp. 197–206.

[27] Tao Zhang, Sherman SM Chow, and Jinyuan Sun. "Password-controlled encryption with accountable break-glass access." In: *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM. 2016.

[28] Srdjan Marinovic, Robert Craven, Jiefei Ma, and Naranker Dulay. "Rumpole: a flexible break-glass access control model." In: *Proceedings of the 16th ACM symposium on Access control models and technologies*. ACM. 2011.

[29] Debby Wallner, Eric Harder, Ryan Agee, et al. *Key management for multicast: Issues and architectures*. Tech. rep. RFC 2627, 1999.

[30] Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas. "Multicast security: A taxonomy and some efficient constructions." In: *IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*. Vol. 2. IEEE. 1999, pp. 708–716.

[31] Sandro Rafaeli and David Hutchison. "A survey of key management for secure group communication." In: *ACM Computing Surveys (CSUR)* 35.3 (2003), pp. 309–329.

[32] John Bethencourt, Amit Sahai, and Brent Waters. "Ciphertext-Policy Attribute-Based Encryption." In: *2007 IEEE Symposium on Security and Privacy (SP '07)*. 2007, pp. 321–334.

[33] Antonis Michalas. "The lord of the shares: Combining attribute-based encryption and searchable encryption for flexible data sharing." In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. 2019, pp. 146–155.

[34] Antonis Michalas. "Sharing in the rain: Secure and efficient data sharing for the cloud." In: *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*. IEEE. 2016.

[35] N. Paladi, C. Gehrmann, and A. Michalas. "Providing User Security Guarantees in Public Infrastructure Clouds." In: *IEEE Transactions on Cloud Computing* 5.3 (2017), pp. 405–419.

[36] Danny Dolev and Andrew C Yao. "On the security of public key protocols." In: *Information Theory, IEEE Transactions on* 29.2 (1983).

[37] Marcela Tuler de Oliveira, Antonis Michalas, Adrien E. D. Groot, Henk A. Marquering, and Sílvia Delgado Olabarriaga. "Red Alert: Break-Glass Protocol to Access Encrypted Medical Records in the Cloud." In: *HealthCom 2019- International Conference on e-health Networking, Applications and Services*. IEEE. 2019.

[38] Victor Costan and Srinivas Devadas. "Intel SGX explained." In: *Cryptology ePrint Archive* (2016).

[39] Alexandros Bakas and Antonis Michalas. "Modern family: A revocable hybrid encryption scheme based on attribute-based encryption, symmetric searchable encryption and SGX." In: *International Conference on Security and Privacy in Communication Systems*. Springer. 2019, pp. 472–486.

[40] Marcela Tuler de Oliveira, Hai-Van Dang, Lúcio HA Reis, Henk A Marquering, and Sílvia Delgado Olabarriaga. "AC-AC: Dynamic revocable access control for acute care teams to access medical records." In: *Smart Health* 20 (2021), p. 100190.

[41] Silke Walter, Panagiotis Kostopoulos, Anton Haass, Isabel Keller, Martin Lesmeister, Thomas Schlechtriemen, Christian Roth, Panagiotis Papanagiotou, Iris Grunwald, Helmut Schumacher, et al. "Diagnosis and treatment of patients with stroke in a mobile stroke unit versus in hospital: a randomised controlled trial." In: *The Lancet Neurology* 11.5 (2012), pp. 397–404.

[42] Benjamin Fabian, Tatiana Ermakova, and Philipp Junghanns. "Collaborative and secure sharing of healthcare data in multi-clouds." In: *Information Systems* 48 (2015), pp. 132–150.

[43] Antonis Michalas, Alexandros Bakas, Hai-Van Dang, and Alexandr Zaltiko. "MicroSCOPE: Enabling Access Control in Searchable Encryption with the Use of Attribute-Based Encryption and SGX." In: *Nordic Conference on Secure IT Systems*. Springer. 2019, pp. 254–270.

[44] Entao Luo, Qin Liu, and Guojun Wang. "Hierarchical multi-authority and attribute-based encryption friend discovery scheme in mobile social networks." In: *IEEE Communications Letters* 20.9 (2016), pp. 1772–1775.

[45] Htoo Aung Maw, Hannan Xiao, Bruce Christianson, and James A Malcolm. "BTG-AC: Break-the-glass access control model for medical data in wireless sensor networks." In: *IEEE journal of biomedical and health informatics* 20.3 (2015), pp. 763–774.

[46] Ana Ferreira and Gabriele Lenzini. "Comparing and Integrating Break-the-Glass and Delegation in Role-based Access Control for Healthcare." In: *ICISSP*. 2016, pp. 63–73.

[47] Wenhai Sun, Shucheng Yu, Wenjing Lou, Y Thomas Hou, and Hui Li. "Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud." In: *IEEE Transactions on Parallel and Distributed Systems* 27.4 (2014), pp. 1187–1198.

[48] WanFen Guo, XiaoLei Dong, ZhenFu Cao, and JiaChen Shen. "Efficient attribute-based searchable encryption on cloud storage." In: *Journal of Physics: Conference Series*. Vol. 1087. 5. IOP Publishing. 2018, p. 052001.

[49] Mukti Padhya and Devesh C Jinwala. "BTG-RKASE: Privacy Preserving Revocable Key Aggregate Searchable Encryption with Fine-grained Multi-delegation & Break-The-Glass Access Control." In: *ICETE (2)*. 2019, pp. 109–124.

[50] Gustav Aagesen and John Krogstie. "BPMN 2.0 for modeling business processes." In: *Handbook on Business Process Management 1*. Springer, 2015, pp. 219–250.

[51] Nico Weichbrodt, Anil Kurmus, Peter Pietzuch, and Rüdiger Kapitza. "AsyncShock: Exploiting synchronisation bugs in Intel SGX enclaves." In: *European Symposium on Research in Computer Security*. Springer. 2016, pp. 440–457.

[52] *Sniffing Attack*. https://en.wikipedia.org/wiki/Sniffing_attack. Accessed: 13/05/2020.

[53] *Man-in-the-middle Attack*. https://en.wikipedia.org/wiki/Man-in-the-middle_attack. Accessed: 13/05/2020.

[54] Ferdinand Brasser, Urs Müller, Alexandra Dmitrienko, Kari Kostiainen, Srdjan Capkun, and Ahmad-Reza Sadeghi. "Software grand exposure:{SGX} cache attacks are practical." In: *11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17)*. 2017.

[55] Karan Gupta, Nitin Rakesh, Neetu Faujdar, and Nidhi Gupta. "IoT Based Solution for Automation of Hospital activities with High Authentication." In: *Smart Systems and IoT: Innovations in Computing*. Springer, 2020, pp. 121–130.

[56] Hu Xiong, Junyi Tao, and Chen Yuan. "Enabling telecare medical information systems with strong authentication and anonymity." In: *IEEE Access* 5 (2017), pp. 5648–5661.

[57] S. Qiu, G. Xu, H. Ahmad, and L. Wang. "A Robust Mutual Authentication Scheme Based on Elliptic Curve Cryptography for Telecare Medical Information Systems." In: *IEEE Access* 6 (2018), pp. 7452–7463.

[58] Catherine McGeoch, Peter Sanders, Rudolf Fleischer, Paul R Cohen, and Doina Precup. "Searching for Big-Oh in the data: Inferring asymptotic complexity from experiments." In: *Lecture Notes in Computer Science: Proceedings of the Dagstuhl Seminar on Experimental Algorithmics*. 2001.

[59] *Django Framework*. https://www.djangoproject.com/. Accessed: 11/03/2021.

[60] *PyJWT 1.7.1*. https://pypi.org/project/PyJWT/. Accessed: 22/05/2020.

[61] *Python Cryptography Toolkit (pycrypto)*. https://pypi.org/project/pycrypto/. Accessed: 22/05/2020.

[62] *AccessControl-AcuteCare-Protocol V1.0*. https://github.com/AMCeScience/AccessControl-AcuteCare-Protocol, Accessed: 29/05/2020.

[63] Marcela T. de Oliveira, Yiannis Verginadis, Lúcio H.A. Reis, Evgenia Psarra, Ioannis Patiniotakis, and Sílvia D. Olabarriaga. "AC-ABAC: Attribute-based access control for electronic medical records during acute care." In: *Expert Systems with Applications* (2022), p. 119271.

[64] Soumitra Sudip Bhuyan, Umar Y Kabir, Jessica M Escareno, Kenya Ector, Sandeep Palakodeti, David Wyant, Sajeesh Kumar, Marian Levy, Satish Kedia, Dipankar Dasgupta, et al. "Transforming healthcare cybersecurity from reactive to proactive: current status and future recommendations." In: *Journal of medical systems* 44.5 (2020), pp. 1–9.

[65] Marcela Tuler de Oliveira, Yiannis Verginadis, Lúcio H. A. Reis, Ioannis Patiniotakis Evgenia Psarra, and Sílvia D. Olabarriaga. *AC-ABAC Github repository*. https://github.com/AMCeScience/AC-ABAC-modelling-public. Accessed: 01/04/2021. 2021.

[66] Barsha Mitra, Shamik Sural, Jaideep Vaidya, and Vijayalakshmi Atluri. "A survey of role mining." In: *ACM Computing Surveys (CSUR)* 48.4 (2016), pp. 1–37.

[67] K Rajesh Rao, Ashalatha Nayak, Indranil Ghosh Ray, Yogachandran Rahulamathavan, and Muttukrishnan Rajarajan. "Role recommender-RBAC: Optimizing user-role assignments in RBAC." In: *Computer Communications* 166 (2021), pp. 140–153.

[68] Ana Ferreira, David Chadwick, Pedro Farinha, Ricardo Correia, Gansen Zao, Rui Chilro, and Luis Antunes. "How to securely break into RBAC: the BTG-RBAC model." In: *2009 Annual Computer Security Applications Conference*. IEEE. 2009, pp. 23–31.

[69] Fatemeh Nazerian, Homayun Motameni, and Hossein Nematzadeh. "Emergency role-based access control (E-RBAC) and analysis of model specifications with alloy." In: *Journal of information security and applications* 45 (2019), pp. 131–142.

[70] Amar Arora and Anjana Gosain. "Dynamic Trust Emergency Role-based Access Control (DTE-RBAC)." In: *International Journal of Computer Applications* 975 (2020), p. 8887.

[71] Mor Peleg, Dizza Beimel, Dov Dori, and Yaron Denekamp. "Situation-based access control: Privacy management via modeling of patient data access scenarios." In: *Journal of Biomedical Informatics* 41.6 (2008), pp. 1028–1040.

[72] Alessio Lunardelli, Ilaria Matteucci, Paolo Mori, and Marinella Petrocchi. "A prototype for solving conflicts in XACML-based e-Health policies." In: *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*. IEEE. 2013, pp. 449–452.

[73] Jorge Calvillo-Arbizu, Isabel Román-Martínez, and Laura M Roa-Romero. "Standardized access control mechanisms for protecting ISO 13606-based electronic health record systems." In: *IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*. IEEE. 2014, pp. 539–542.

[74] Yiannis Verginadis, Antonis Michalas, Panagiotis Gouvas, Gunther Schiefer, Gerald Hübsch, and Iraklis Paraskakis. "PaaSword: A Holistic Data Privacy and Security by Design Framework for Cloud Services." In: *Journal of Grid Computing* 15 (2017), pp. 219–234.

[75] Evgenia Psarra, Yiannis Verginadis, Ioannis Patiniotakis, Dimitris Apostolou, and Gregoris Mentzas. "A Context-Aware Security Model for a Combination of Attribute-Based Access Control and Attribute-Based Encryption in the Healthcare Domain." In: *Workshops of the International Conference on Advanced Information Networking and Applications*. Springer. 2020, pp. 1133–1142.

[76] Mohamed Abomhara, Huihui Yang, and Geir M Køien. "Access control model for cooperative healthcare environments: Modeling and verification." In: *2016 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE. 2016, pp. 46–54.

[77] Kwangsoo Seol, Young-Gab Kim, Euijong Lee, Young-Duk Seo, and Doo-Kwon Baik. "Privacy-preserving attribute-based access control model for XML-based electronic health record system." In: *IEEE Access* 6 (2018), pp. 9114–9128.

[78] *FHIR Episode of care*. https://www.hl7.org/fhir/episodeofcare.html. Accessed: 11/03/2021. 2011.

[79] *eXtensible Access Control Markup Language (XACML) Version 3.0*. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html. Accessed: 08/12/2020.

[80] *OASIS XACML Technical Committee*. https://www.oasis-open.org. Accessed: 08/12/2020.

[81] *W3C XML Schema Definition Language (XSD)*. https://www.w3.org/TR/xmlschema11-1/. Accessed: 08/12/2020. 2012.

[82] Evgenia Psarra, Yiannis Verginadis, Ioannis Patiniotakis, Dimitris Apostolou, and Gregoris Mentzas. "Securing Access to Healthcare Data with Context-aware Policies." In: *11th International Conference on Information, Intelligence, Systems and Applications (IISA)*. IEEE. 2020, pp. 1–6.

[83] Marcela Tuler de Oliveira, Yiannis Verginadis, Lúcio H. A. Reis, Ioannis Patiniotakis Evgenia Psarra, and Sílvia D. Olabarriaga. *AC-ABAC templates*. https://github.com/AMCeScience/AC-ABAC-modelling-public/tree/main/templates. Accessed: 01/04/2021. 2021.

[84] *WSO2 Balana*. https://github.com/wso2/balana. Accessed: 11/03/2021. 2014.

[85] James DesLauriers, Tamas Kiss, Resmi C Ariyattu, Hai-Van Dang, Amjad Ullah, James Bowden, Dagmar Krefting, Gabriele Pierantoni, and Gabor Terstyanszky. "Cloud apps to-go: Cloud portability with TOSCA and MiCADO." In: *Concurrency and Computation: Practice and Experience* 33.19 (2021), e6093.

[86] Alexandros Bakas, Hai-Van Dang, Antonis Michalas, and Alexandr Zalitko. "The Cloud we Share: Access Control on Symmetrically Encrypted Data in Untrusted Clouds." In: *IEEE Access* 8 (2020), pp. 210462–210477.

[87] Yiannis Verginadis, Panagiotis Gouvas, Spyros Mantzouratos, Simeon Veloudis, Thomas Schork Sebastian, Ludwig Seitz, Iraklis Paraskakis, and Gregoris Mentzas. "Context-aware Policy Enforcement for PaaS-enabled Access Control." In: *IEEE Transactions on Cloud Computing* 10.1 (2019), pp. 276–291.

[88] Billy Bob Brumley and Nicola Tuveri. "Remote Timing Attacks Are Still Practical." In: *Computer Security – ESORICS 2011*. Ed. by Vijay Atluri and Claudia Diaz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 355–371.

[89] Marcela Tuler de de Oliveira, Lúcio Henrik A Reis, Henk Marquering, Aeilko Having Zwinderman, and Sílvia D Olabarriaga. "Perceptions of a secure cloud-based solution for data sharing during acute stroke care: qualitative interview study." In: *JMIR Formative Research* (2022), p. 40061.

[90] Michelle P Lin. "Time matters greatly in acute stroke care." In: *Neurologia i neurochirurgia polska* 54.2 (2020), pp. 104–105.

[91] Kui Ren, Cong Wang, and Qian Wang. "Security challenges for the public cloud." In: *IEEE Internet computing* 16.1 (2012), pp. 69–73.

[92] Stephan A Munich, Lee A Tan, Danilo M Nogueira, Kiffon M Keigher, Michael Chen, R Webster Crowley, James J Conners, and Demetrius K Lopes. "Mobile real-time tracking of acute stroke patients and instant, secure inter-team communication-the Join app." In: *Neurointervention* 12.2 (2017), p. 69.

[93] Hyo Suk Nam, JoonNyung Heo, Jinkwon Kim, Young Dae Kim, Tae Jin Song, Eunjeong Park, and Ji Hoe Heo. "Development of smartphone application that aids stroke screening and identifying nearby acute stroke care hospitals." In: *Yonsei medical journal* 55.1 (2014), pp. 25–29.

[94] Hossein Rahmani, Elankovan Sundararajan, Zulkarnain Md Ali, and Abdullah Mohd Zin. "Encryption as a Service (EaaS) as a Solution for Cryptography in Cloud." In: *Procedia Technology* 11 (2013), pp. 1202–1210.

[95] Joseph A Akinyele, Matthew W Pagano, Matthew D Green, Christoph U Lehmann, Zachary NJ Peterson, and Aviel D Rubin. "Securing electronic medical records using attribute-based encryption on mobile devices." In: *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*. 2011, pp. 75–86.

[96] *Advanced Secure Cloud Encrypted Platform for Internationally Orchestrated Solutions in Healthcare, howpublished=*https:https://www.asclepios-project.eu*, note = Accessed: 01/11/2022, key= EU Horizon 2020, year= 2022*.

[97] Taridzo Chomutare, Kassaye Yitbarek Yigzaw, Silvia Delgado Olabarriaga, Alexandra Makhlysheva, Marcela Tuler de Oliveira, Line Silsand, Dagmar Krefting, Thomas Penzel, Christiaan Hillen, and Johan Gustav Bellika. "Healthcare and data privacy requirements for e-health cloud: A qualitative analysis of clinician perspectives." In: *2020 IEEE International Conference on E-health Networking, Application & Services (HEALTHCOM)*. IEEE. 2021, pp. 1–8.

[98] Lúcio HA Reis, Marcela Tuler de Oliveira, Diogo MF Mattos, and Sílvia D Olabarriaga. "Private Data Sharing in a Secure Cloud-based Application for Acute Stroke Care." In: *2021 IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE. 2021, pp. 568–573.

[99] Young Mi Choi. "Utilizing end user input in early product development." In: *Procedia Manufacturing* 3 (2015), pp. 2244–2250.

[100] Monique Hennink and Bonnie N Kaiser. "Sample sizes for saturation in qualitative research: A systematic review of empirical tests." In: *Social Science & Medicine* (2021), p. 114523.

[101] *Google Form*. https://www.google.com/forms/about/. Accessed: 02/11/2022.

[102] *Voice meeting notes & real-time transcription*. https://www.otter.ai/. Accessed: 02/11/2022.

[103] Vanessa Azevedo, Margarida Carvalho, Flávia Fernandes-Costa, Soraia Mesquita, Joana Soares, Filipa Teixeira, and Ângela Maia. "Interview transcription: conceptual issues, practical guidelines, and challenges." In: *Revista de Enfermagem Referência* 4.14 (2017), pp. 159–167.

[104] Kirsti Malterud. "Systematic text condensation: a strategy for qualitative analysis." In: *Scandinavian journal of public health* 40.8 (2012), pp. 795–805.

[105] *Interview Analysis with MAXQDA*. https://www.maxqda.com. Accessed: 02/11/2022.

[106] Elizabeth Murray, Joanne Burns, Carl May, Tracy Finch, Catherine O'Donnell, Paul Wallace, and Frances Mair. "Why is it difficult to implement e-health initiatives? A qualitative study." In: *Implementation Science* 6.1 (2011), pp. 1–11.

[107] Anton Hasselgren, Katina Kralevska, Danilo Gligoroski, Arild Faxvaag, et al. "Medical Students' Perceptions of a Blockchain-Based Decentralized Work History and Credentials Portfolio: Qualitative Feasibility Study." In: *JMIR Formative Research* 5.10 (2021), e33113.

[108] Carl Joakim Brandt, Jane Clemensen, Jesper Bo Nielsen, and Jens Søndergaard. "Drivers for successful long-term lifestyle change, the role of e-health: a qualitative interview study." In: *BMJ open* 8.3 (2018), e017466.

[109] Ijeoma Azodo, Robin Williams, Aziz Sheikh, Kathrin Cresswell, et al. "Opportunities and challenges surrounding the use of data from wearable sensor devices in health care: qualitative interview study." In: *Journal of medical Internet research* 22.10 (2020), e19542.

[110] A Georgiou, M Prgomet, S Lymer, A Hordern, L Ridley, and J Westbrook. "The impact of a health IT changeover on Medical Imaging Department work processes and turnaround times." In: *Applied clinical informatics* 6.03 (2015), pp. 443–453.

[111] Aniek Woodward, Molly Fyfe, Jibril Handuleh, Preeti Patel, Brian Godman, Andrew Leather, and Alexander Finlayson. "Diffusion of e-health innovations in 'post-conflict'settings: a qualitative study on the personal experiences of health workers." In: *Human resources for health* 12.1 (2014), pp. 1–10.

[112] Marcela Tuler De Oliveira, Lúcio Henrik Amorim Reis, Yiannis Verginadis, Diogo Menezes Ferrazani Mattos, and Sílvia Delgado Olabarriaga. "SmartAccess: Attribute-Based Access Control System for Medical Records based on Smart Contracts." In: *IEEE Access* (2022), pp. 1–1.

[113] *Joint controllers*. https://gdpr-info.eu/art-26-gdpr/. Accessed: 10/11/2021. 2016.

[114] Amjad Ullah, Huseyin Dagdeviren, Resmi C Ariyattu, James DesLauriers, Tamas Kiss, and James Bowden. "MiCADO-Edge: Towards an Application-level Orchestrator for the Cloud-to-Edge Computing Continuum." In: *Journal of Grid Computing* 19.4 (2021), pp. 1–28.

[115] *IPFS powers the Distributed Web*. https://ipfs.io. Accessed: 17/10/2021. 2015.

[116] Damiano Di Francesco Maesa, Paolo Mori, and Laura Ricci. "A blockchain based approach for the definition of auditable access control systems." In: *Computers & Security* 84 (2019), pp. 93–119.

[117] Sara Rouhani, Rafael Belchior, Rui S Cruz, and Ralph Deters. "Distributed attribute-based access control system using permissioned blockchain." In: *World Wide Web* 24.5 (2021), pp. 1617–1644.

[118] Amal Ghorbel, Mahmoud Ghorbel, and Mohamed Jmaiel. "Accountable privacy preserving attribute-based access control for cloud services enforced using blockchain." In: *International Journal of Information Security* (2021), pp. 1–20.

[119] Angela Ballantyne. "How should we think about clinical data ownership?" In: *Journal of Medical Ethics* 46.5 (2020), pp. 289–294.

[120] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system." In: *Decentralized Business Review* (2008), p. 21260.

[121] Gavin Wood. "Ethereum: A secure decentralised generalised transaction ledger." In: *Ethereum project yellow paper* 151 (2014), pp. 1–32.

[122] Marcela Tuler de Oliveira, Lúcio H.A. Reis, Dianne S.V. Medeiros, Ricardo C. Carrano, Sílvia D. Olabarriaga, and Diogo M.F. Mattos. "Blockchain reputation-based consensus: A scalable and resilient mechanism for distributed mistrusting applications." In: *Computer Networks* 179 (2020), p. 107367.

[123] Marcela Tuler Oliveira, Gabriel R Carrara, Natalia C Fernandes, Célio VN Albuquerque, Ricardo C Carrano, Dianne SV Medeiros, and Diogo MF Mattos. "Towards a performance evaluation of private blockchain frameworks using a realistic workload." In: *2019 22nd conference on innovation in clouds, internet and networks and workshops (ICIN)*. IEEE. 2019, pp. 180–187.

[124] K. Christidis and M. Devetsikiotis. "Blockchains and Smart Contracts for the Internet of Things." In: *IEEE Access* 4 (2016), pp. 2292–2303.

[125] Dianne Medeiros, Natalia Fernandes, and Diogo M. F. Mattos. "Smart Contracts and the Power Grid: A Survey." In: *2019 1st Blockchain, Robotics and AI for Networking Security Conference (BRAINS) (BRAINS'19)*. Rio De Janeiro, Brazil, Mar. 2019.

[126] *A Guide to Events and Logs in Ethereum Smart Contracts*. https://consensys.net/blog/developers/guide-to-events-and-logs-in-ethereum-smart-contracts/. Accessed: 15/12/2021. 2016.

[127] João Pedro Dias, Hugo Sereno Ferreira, and Ângelo Martins. "A blockchain-based scheme for access control in e-health scenarios." In: *International Conference on Soft Computing and Pattern Recognition*. Springer. 2018, pp. 238–247.

[128] Kai Fan, Shangyang Wang, Yanhui Ren, Hui Li, and Yintang Yang. "Medblock: Efficient and secure medical data sharing via blockchain." In: *Journal of medical systems* 42.8 (2018), pp. 1–11.

[129] Peng Zhang, Jules White, Douglas C Schmidt, Gunther Lenz, and S Trent Rosenbloom. "FHIRChain: applying blockchain to securely and scalably share clinical data." In: *Computational and structural biotechnology journal* 16 (2018), pp. 267–278.

[130] Eman-Yasser Daraghmi, Yousef-Awwad Daraghmi, and Shyan-Ming Yuan. "MedChain: A Design of Blockchain-Based System for Medical Records Access and Permissions Management." In: *IEEE Access* 7 (2019), pp. 164595–164613.

[131] Asaph Azaria, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman. "MedRec: Using Blockchain for Medical Data Access and Permission Management." In: *2016 2nd International Conference on Open and Big Data (OBD)*. 2016, pp. 25–30.

[132] Gaby G Dagher, Jordan Mohler, Matea Milojkovic, and Praneeth Babu Marella. "Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology." In: *Sustainable cities and society* 39 (2018), pp. 283–297.

[133] Alevtina Dubovitskaya, Furqan Baig, Zhigang Xu, Rohit Shukla, Pratik Sushil Zambani, Arun Swaminathan, Md Majid Jahangir, Khadija Chowdhry, Rahul Lachhani, Nitesh Idnani, et al. "ACTION-EHR: patient-centric blockchain-based electronic health record data management for cancer care." In: *Journal of medical Internet research* 22.8 (2020), e13598.

[134] Ahmed Raza Rajput, Qianmu Li, and Milad Taleby Ahvanooey. "A blockchain-based secret-data sharing framework for personal health records in emergency condition." In: *Healthcare*. Vol. 9. 2. Multidisciplinary Digital Publishing Institute. 2021, p. 206.

[135] Miguel Castro, Barbara Liskov, et al. "Practical byzantine fault tolerance." In: *OsDI*. Vol. 99. 1999. 1999, pp. 173–186.

[136] Shubhani Aggarwal and Neeraj Kumar. "Attacks on blockchain." In: *Advances in Computers*. Vol. 121. Elsevier, 2021, pp. 399–410.

[137] Franco Callegati, Walter Cerroni, and Marco Ramilli. "Man-in-the-Middle Attack to the HTTPS Protocol." In: *IEEE Security & Privacy* 7.1 (2009), pp. 78–81.

[138] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. "A digital signature scheme secure against adaptive chosen-message attacks." In: *SIAM Journal on computing* 17.2 (1988), pp. 281–308.

[139] Eike Kiltz and John Malone-Lee. "A general construction of IND-CCA2 secure public key encryption." In: *IMA International Conference on Cryptography and Coding*. Springer. 2003, pp. 152–166.

[140] https://soliditylang.org. *Solidity 0.8.10 documentation*. https://docs.soliditylang.org/en/v0.8.10/. Accessed: 24/11/2021. 2021.

[141] *SmartAccess, Github repository*. https://github.com/AMCeScience/paper-acess-control-blockchain. Accessed: 01/06/2022. 2022.

[142] *GoQuorum*. https://consensys.net/docs/goquorum//en/latest/. Accessed: 01/06/2022. 2022.

[143] *The Istanbul BFT Consensus Algorithm*. https://github.com/ConsenSys/quorum-ibft. Accessed: 29/06/2022. 2020.

[144] *Quorum Genesis Tool*. https://github.com/ConsenSys/quorum-genesis-tool. Accessed: 29/06/2022. 2022.

[145] *Raft consensus*. https://raft.github.io. Accessed: 29/06/2022. 2022.

[146] *ChainHammer*. https://github.com/drandreaskrueger/chainhammer. Accessed: 01/06/2022. 2022.

[147] *Role-based Access Control smart contract (RBAC-SL)*. https://github.com/runningbeta/rbac-solidity. Accessed: 01/06/2022. 2022.

[148] *Access Control List smart contract (ACL-SC*. https://github.com/masaun/ACL-smart-contract. Accessed: 01/06/2022. 2022.

[149] *Hartschting. Numbers and facts about vasculary diseases*. https://www.hartstichting.nl/hart-en-vaatziekten/feiten-en-cijfers-hart-en-vaatziekten. Accessed: 01/06/2022. 2022.

[150] *Hyperledger Indy*. https://www.hyperledger.org/use/hyperledger-indy. Accessed: 01/06/2022. 2022.

[151] Gregory Vial. "Understanding digital transformation: A review and a research agenda." In: *Managing Digital Transformation* (2021), pp. 13–66.

[152] Sascha Kraus, Francesco Schiavone, Anna Pluzhnikova, and Anna Chiara Invernizzi. "Digital transformation in healthcare: Analyzing the current state-of-research." In: *Journal of Business Research* 123 (2021), pp. 557–567.

[153] Government of the Netherlands. *Privacy statement by the Ministry of Health, Welfare and Sport*. URL: https://www.government.nl/ministries/ministry-of-health-welfare-and-sport/privacy.

[154] Svein Ølnes, Jolien Ubacht, and Marijn Janssen. *Blockchain in government: Benefits and implications of distributed ledger technology for information sharing*. 2017.

[155] Peter Robinson, Raghavendra Ramesh, and Sandra Johnson. "Atomic crosschain transactions for ethereum private sidechains." In: *Blockchain: Research and Applications* 3.1 (2022), p. 100030.

[156] Xiaoqiang Sun, F Richard Yu, Peng Zhang, Zhiwei Sun, Weixin Xie, and Xiang Peng. "A survey on zero-knowledge proof in blockchain." In: *IEEE network* 35.4 (2021), pp. 198–205.

[157] Xiaohui Yang and Wenjie Li. "A zero-knowledge-proof-based digital identity management scheme in blockchain." In: *Computers & Security* 99 (2020), p. 102050.

SUMMARY

Healthcare is going through a digital transformation. Through Electronic Medical Records (EMR) systems, sensitive data is collected and shared across organisations for clinical and research purposes. However, data security and privacy are one of the pressing challenges facing the healthcare industry today. All because EMR systems contain vast amounts of sensitive patient data, which makes them attractive targets for malicious purposes.

Although the primary goal of EMR systems is to make data available, this should not mean that patients' data should be available to anyone. Usually, EMR are only available for healthcare professionals that have a treatment relationship with the patient, assuming implicit consent. Any other request to access the data is restricted, and the professionals may request to "break the glass" to proceed. The break-glass procedure typically only requires the professionals to acknowledge that the access is restricted and give the reason for the access. Thus, professionals can access patient data using the break-glass procedure, which inevitably raises suspicion that patients' data privacy might be jeopardised. Even during emergencies, when there is a vital interest for the patient, and access to the data is considered legitimate, only the acute care teams involved in patient care should be allowed to access the patient data. EMR systems must have an appropriate access control mechanism to protect patient privacy and prove compliance with data protection and safety regulations. For that, EMR systems must guarantee data processing confidentiality, data integrity, traceability and auditability.

This thesis presents various secure mechanisms for cross-organisational data sharing during acute stroke care. In such situations, access to the data is urgent. For all the proposals, we assume the patients' EMR are stored in a cloud system to improve accessibility and collection of medical records during the emergency. However, to protect against abuse or internal attacks in the cloud providers, the patient's EMR is stored as ciphertext in the cloud, and the encryption keys are only shared with the involved acute care teams. Additionally, the proposed protocols are used to dynamically grant and revoke access to the patient's EMR for the healthcare teams according to the needs of triage, diagnosis, hospital selection and treatment.

Chapter 2 presents a protocol that allows access to encrypted patients' EMR only during an emergency and only for authorised treatment teams. The Red Alert Protocol (RAP) is based on the Ciphertext-Policy Attribute-Based Encryp-

tion (CP-ABE) scheme to encrypt the EMR associated with policies defined for emergency situations. Additionally, it adopts token authentication to grant and revoke data access during the timeline of acute stroke care. Cryptography happens on the user's side, so the cloud provider only handles encrypted data. Through a messaging protocol, all treatment teams can securely decrypt the patient's EMR and add new information about the patient's treatment. Furthermore, the protocol ensures that acute care teams only access the patient's EMR for the period during which the patient is under their care. However, we observed through experimental results that the CP-ABE solution would present scalability limitations when used to encrypt large amounts of data.

Chapter 3 describes another protocol that overcomes the scalability limitation of CP-ABE faced in RAP, presented in Chapter 2. With the same goal of dynamic sharing of the encrypted patient's EMR among emergency teams, the proposed AC-AC protocol uses a hybrid encryption scheme combining Dynamic index-based Symmetric Searchable Encryption (DSSE) and CP-ABE. The AC-AC protocol uses DSSE, which is more scalable, to encrypt the patients' EMR, and, as a second layer of encryption, the DSSE keys are encrypted with the CP-ABE scheme. Similarly to RAP, we defined emergency policies for the CP-ABE, but instead of protecting the data, we now protected the DSSE key to the data. Therefore, the members of the acute care team involved in the patient's emergency have the adequate attributes to decrypt the DSSE key and then access the patient's EMR. Moreover, we propose to use, on top of the hybrid encryption scheme, a dynamic access control protocol that grants and revokes access to the encrypted keys and the encrypted EMR according to the type of action (i.e., view, add, delete and revoke access). Finally, AC-AC was proven to be resilient to multiple attacks. The expected execution time of the AC-AC algorithms used during an emergency session is acceptable in an acute care timeline.

Chapter 4 presents Context-Aware Attribute-Based Access Control model with dynamic and fine-grained access control policies for patient EMR, coined Acute Care Attribute-Based Access Control (AC-ABAC). Moreover, we applied a step-by-step methodology for modelling the AC-ABAC. AC-ABAC presents the policies and contextual attributes used by acute care teams to legitimate an emergency session for a patient. AC-ABAC is designed as an extra layer of protection for the system on top of cryptography schemes. Any request to access an EMR system must first be authorised by AC-ABAC. Furthermore, we developed a prototype to evaluate the correctness and performance of the model. The results of performance show that the time added by the AC-ABAC evolution policies to the overall request process is worthwhile, considering the security added to the EMR system.

Chapter 5 focuses on gaining the professionals' feedback on technology development to improve acceptability. For that, we developed a prototype of a

secure EMR cloud-based application that combines the AC-AC encryption protocol from Chapter 3 and the AC-ABAC model from Chapter 4. We presented the prototype to acute care professionals and explained how it would be used during a simulated scenario of acute stroke care. This Chapter presents the methods, results and discussion of in-depth interviews with their perspectives on the application. It identifies several significant barriers and improvement opportunities for the future acceptance and adoption of the proposed application. The participants emphasised that our cloud-based application would solve data-sharing problems, such as the duplication of data, lack of information and standardisation. Still, it would not be realistic to propose that all the organisations involved in acute care migrate to a single cloud-based application.

Chapter 6 focuses on organisations' joint agreements over access policies, dynamic access control, transparency, and auditability. We propose the SmartAccess system based on smart contracts and blockchain technology to implement the ABAC model for distributed access control across organisations. In SmartAccess, healthcare organisations can agree and comply with common access control policies for patient data processing. The organisations use the consensus mechanism of the blockchain to manage the policies in agreement with the other organisations. Data access is only allowed if the healthcare professional runs the smart contracts and has the right attributes to comply with the policy rules. Therefore, policy decisions and enforcement do not depend on a centralised server but on a distributed execution of smart contracts. Finally, every function executed in the smart contracts generates auditable transaction logs published in the blockchain. Through a proof-of-concept implementation, we demonstrated the feasibility of our proposal with different blockchain network consensus mechanisms. SmartAccess has the costs of a decentralised system, but the trade-off is transparency, regulation compliance and auditability for complex cross-organisation data sharing.

This thesis presents mechanisms for secure cross-organisational data sharing during acute stroke care. The proposed mechanisms are applicable and generalisable to other acute and non-acute care cases. These mechanisms focus on improving data availability once the security requirements are fulfilled, so the professionals no longer need break-glass procedures, even in emergencies. Instead, access to data happens with lawful purpose, without compromising patient privacy. Furthermore, cross-organisational data sharing requires trustworthiness among organisations, which needs to be built with regulation compliance and transparency. The proposed mechanisms of this thesis may help future healthcare digital systems comply with the regulations and be more transparent, thus improving security and enabling data sharing across organisations.

SAMENVATTING

De zorg maakt een digitale transformatie door. Via Elektronisch patiëntendossier (EPD)-systemen worden gevoelige gegevens verzameld en gedeeld tussen organisaties voor klinischen onderzoeksdoeleinden. Gegevensbeveiliging en privacy zijn echter dringende uitdagingen waarmee de gezondheidszorg tegenwoordig wordt geconfronteerd. En dat allemaal omdat EPD-systemen enorme hoeveelheden gevoelige patiëntgegevens bevatten, waardoor ze een aantrekkelijk doelwit zijn voor kwaadwillende doeleinden.

Hoewel het primaire doel van EPD-systemen is om gegevens beschikbaar te maken, mag dit niet betekenen dat patiëntengegevens voor iedereen beschikbaar worden. Meestal zijn de EPD's alleen beschikbaar voor werknemers in de gezondheidszorg die een behandelrelatie hebben met de patiënt, waarmee verondersteld wordt dat de patiënt impliciet toestemming geeft. Elk ander verzoek om toegang tot de gegevens te krijgen is beperkt, en werknemers moeten toestemming vragen om patiëntdata te gebruiken middels een breekglasprocedure. De breekglasprocedure vereist doorgaans alleen dat de werknemers erkennen dat de toegang tot de data beperkt is, waarnaast ze de reden voor toegang tot de data op moeten geven. Zo hebben alle werknemers toegang tot patiëntgegevens met behulp van de breekglasprocedure, wat onvermijdelijk aanleiding geeft tot het vermoeden dat de gegevensprivacy van patiënten in gevaar kan komen. Zelfs in noodsituaties, wanneer er een vitaal belang is voor de patiënt en toegang tot de gegevens als gerechtvaardigd wordt beschouwd, mogen alleen de acute zorgteams die betrokken zijn bij de patiëntenzorg toegang krijgen tot de patiëntgegevens. EPD-systemen moeten een geschikt mechanisme voor toegangscontrole hebben om de privacy van patiënten te beschermen en te bewijzen dat ze voldoen aan de voorschriften voor gegevensbescherming en veiligheid. Daarvoor moeten EPD-systemen de vertrouwelijkheid van de gegevensverwerking, de gegevensintegriteit, de traceerbaarheid en controleerbaarheid van de gegevensverwerking garanderen.

Dit proefschrift draagt verschillende veilige mechanismen aan voor het delen van gegevens tussen organisaties tijdens acute zorg voor een beroerte. In dergelijke situaties is toegang tot de gegevens van essentieel belang. Voor alle voorstellen gaan we ervan uit dat het EPD van de patiënten wordt opgeslagen in een cloudsysteem om de toegankelijkheid en verzameling van medische dossiers tijdens de noodsituatie te verbeteren. Ter bescherming tegen misbruik of interne aanvallen bij de cloudaanbieders wordt het EPD van de patiënt echter

als versleutelde tekst (Ciphertext) in de cloud opgeslagen en worden de encryptiesleutels alleen gedeeld met de betrokken acute zorgteams. Bovendien worden de voorgestelde protocollen gebruikt om dynamisch toegang tot het EPD van de patiënt te verlenen en in te trekken voor de zorgteams afhankelijk van de behoeften van triage, diagnose, ziekenhuisselectie en behandeling.

Hoofdstuk 2 stelt een protocol voor dat alleen tijdens een noodgeval en alleen voor geautoriseerde behandelteams toegang geeft tot versleutelde EPD van patiënten. Het Red Alert Protocol (RAP) is gebaseerd op het Ciphertext-Policy Attribute-Based Encryption (CP-ABE)-schema om de EMR te coderen dat is gekoppeld aan het beleid dat is gedefinieerd voor noodsituaties. Bovendien gebruikt het token-authenticatie om gegevenstoegang te verlenen en in te trekken tijdens de tijdlijn van acute zorg voor een beroerte. De cryptografie gebeurt aan de kant van de gebruiker, dus de cloudprovider verwerkt alleen versleutelde gegevens. Via een berichtenprotocol kunnen alle behandelteams het EPD van de patiënt veilig ontsleutelen en nieuwe informatie over de behandeling van de patiënt toevoegen. Bovendien zorgt het protocol ervoor dat acute zorgteams alleen toegang hebben tot het EPD van de patiënt gedurende de periode dat de patiënt onder hun hoede is. We hebben echter via experimentele resultaten waargenomen dat de CP-ABE-oplossing schaalbaarheidsbeperkingen zou opleveren bij gebruik om grote hoeveelheden gegevens te versleutelen.

Hoofdstuk 3 beschrijft een ander protocol dat de schaalbaarheidsbeperking van CP-ABE in RAP overbrugt, welke gepresenteerd is in hoofdstuk 2. Met hetzelfde doel van het dynamisch delen van het EPD van de versleutelde patiënt onder noodteams, maakt het voorgestelde AC-AC-protocol gebruik van een hybride versleutelingsschema. een combinatie van op dynamische indexen gebaseerde symmetrische doorzoekbare versleuteling (DSSE) en CP-ABE. Het AC-AC-protocol gebruikt DSSE, dat beter schaalbaar is, om het EPD van de patiënt te versleutelen, en als tweede versleutelingslaag worden de DSSE-sleutels versleuteld met het CP-ABE-schema. Net als bij RAP hebben we noodbeleid voor de CP-ABE gedefinieerd, maar in plaats van de gegevens te beschermen, hebben we nu de DSSE-sleutel voor de gegevens beschermd. Daarom beschikken de leden van het acute zorgteam die betrokken zijn bij de noodsituatie van de patiënt over de juiste attributen om de DSSE-sleutel te decoderen en vervolgens toegang te krijgen tot het EPD van de patiënt. Bovendien stellen we voor om, bovenop het hybride coderingsschema, een dynamisch toegangscontroleprotocol te gebruiken dat toegang tot de gecodeerde sleutels en de gecodeerde EMR verleent of intrekt, afhankelijk van het type actie (d.w.z. bekijken, toevoegen, verwijderen en intrekken van toegang). Ten slotte is bewezen dat AC-AC bestendig is tegen meerdere aanvallen. De verwachte uitvoeringstijd van de AC-AC-algoritmen die tijdens een spoedsessie worden gebruikt is acceptabel in een acute zorgtijdlijn.

Hoofdstuk 4 introduceert het Context-Aware Attribute-Based Access Control-model met dynamisch en fijnmazig beleid voor toegangscontrole voor EMR van patiënten, aangeduid met Acute Care Attribute-Based Access Control (AC-ABAC). Bovendien hebben we een stapsgewijze methodologie toegepast voor het modelleren van de AC-ABAC. AC-ABAC presenteert het beleid en de contextuele kenmerken die door acute zorgteams worden gebruikt om een spoedsessie voor een patiënt te rechtvaardigen. AC-ABAC is ontworpen als een extra beveiligingslaag voor het systeem bovenop cryptografieschema's. Elk verzoek om toegang tot een EPD-systeem moet eerst worden goedgekeurd door AC-ABAC. Verder hebben we een prototype ontwikkeld om de juistheid en prestaties van het model te evalueren. De resultaten van de prestaties laten zien dat de tijd die door het AC-ABAC-evolutiebeleid wordt toegevoegd aan het algehele aanvraagproces de moeite waard is, gezien de beveiliging die het toevoegt aan het EPD-systeem.

Hoofdstuk 5 richt zich op het verkrijgen van feedback van werknemers in een vroeg stadium van de technologieontwikkeling om de acceptatie ervan te verbeteren. Daarvoor ontwikkelden we een prototype van een veilige EMR-cloudge- baseerde applicatie die het AC-AC-coderingsprotocol uit hoofdstuk 3 en het AC-ABAC-model uit hoofdstuk 4 combineert. We presenteerden het prototype aan acute zorgprofessionals en legden uit hoe het gebruikt zou moeten worden tijdens een gesimuleerd scenario van acute zorg voor een beroerte. Dit hoofdstuk presenteert de methoden, resultaten en bespreking van diepte-interviews met hun perspectieven op de toepassing. Het identificeert verschillende belangrijke belemmeringen en verbeteringsmogelijkheden voor de toekomstige acceptatie en goedkeuring van de voorgestelde aanvraag. De deelnemers benadrukten dat onze cloudgebaseerde applicatie problemen met het delen van gegevens zou oplossen, zoals het dupliceren van gegevens, gebrek aan informatie en standaardisatie. Toch zou het niet realistisch zijn om voor te stellen dat alle organisaties die betrokken zijn bij de acute zorg migreren naar één cloudgebaseerde applicatie.

Hoofdstuk 6 richt zich op de gezamenlijke afspraken van organisaties over toegangsbeleid, dynamische toegangscontrole, transparantie en controleerbaarheid. We stellen het SmartAccess-systeem voor op basis van smart contracts en blockchain-technologie om het ABAC-model voor gedistribueerde toegangscontrole over organisaties te implementeren. In SmartAccess kunnen zorgorganisaties met elkaar overeenstemmen en voldoen aan gemeenschappelijk beleid voor toegangscontrole voor de verwerking van patiëntgegevens. De organisaties gebruiken het consensusmechanisme van blockchains om het beleid in overleg met de andere organisaties te beheren. Toegang tot gegevens is alleen toegestaan als de zorgprofessional de smart contracts uitvoert en over de juiste attributen beschikt om aan de beleidsregels te voldoen. Beleidsbeslissingen en

handhaving zijn daarom niet afhankelijk van een gecentraliseerde server maar van een gedistribueerde uitvoering van de smart contracts. Ten slotte genereert elke functie die in de smart contracts wordt uitgevoerd, controleerbare transactielogboeken die in de blockchain worden gepubliceerd. Door middel van een proof-of-concept-implementatie hebben we de haalbaarheid van ons voorstel aangetoond met verschillende consensusmechanismen voor blockchain-netwerken. SmartAccess heeft de kosten van een gedecentraliseerd systeem, maar de compromis biedt transparantie, naleving van de regelgeving en controleerbaarheid voor het delen van complexe gegevens tussen organisaties.

Dit proefschrift draagt verschillende mechanismen aan voor het veilig delen van gegevens tussen organisaties tijdens acute zorg voor een beroerte. De voorgestelde mechanismen zijn toepasbaar en generaliseerbaar naar andere acute en niet-acute zorggevallen. Deze mechanismen zijn gericht op het verbeteren van de beschikbaarheid van gegevens zodra aan de beveiligingsvereisten is voldaan, zodat professionals geen breekglasprocedures meer nodig hebben, zelfs niet in noodgevallen. In plaats daarvan vindt toegang tot gegevens plaats met een wettig doel, zonder de privacy van de patiënt in gevaar te brengen. Bovendien vereist het delen van gegevens tussen organisaties de betrouwbaarheid van organisaties, die moet worden opgebouwd met naleving van de regelgeving en transparantie. De voorgestelde mechanismen van dit proefschrift kunnen toekomstige digitale zorgsystemen helpen om te voldoen aan de regelgeving en transparanter te zijn, waardoor de beveiliging wordt verbeterd en het delen van gegevens tussen organisaties mogelijk wordt gemaakt.

Name PhD student: Marcela Tuler de Oliveira PhD period: 12/2018 - 11/2022 Name PhD supervisor: Dr. S.D. Olabarriaga, Prof. Dr. A. H. Zwinderman and Prof. Dr. Henk Marquering		
PhD training	Year	ECTS
General courses		
The World of Science	2019	0,7
Research Data Management	2019	0,9
E-science	2021	0,7
Didactic skills	2022	1,5
Scientific Writing in English	2022	1,5
Specific courses		
Cryptography 1 - Coursera	2019	1,0
Machine Learning - Udemy	2020	2,0
Deep Learning - Udemy	2021	1,8
Summer school: Real-world crypto and privacy organised by Digital Security group - Radboud University	2022	1,5
Seminars, workshops and master classes		
Three times ASCLEPIOS 'Protecting Vital Assets Workshop' - Secura	2019-2021	1,5
Threat Modeling Webinar: 'Find problems when there's time to fix them' – Secura	2019	0,4
Weekly meetings of the Cardiovascular Engineering	2019-2022	4,0
Weekly seminars of the Clinical Epidemiology, Biostatistics and Bioinformatics	2019-2020	3,0
Bi-weekly seminars Epidemiology and Data Science department	2021-2022	0,5

Presentations		
Paper at IEEE Healthcom 2019 - Bogotá, Colombia	2019	0,2
Four meeting of the ASCLEPIOS project	2019-2021	1,6
Three presentation on Weekly Cardiovascular Engineering Meeting of Biomedical Engineering and Physics	2019-2022	0,8
Three ASCLEPIOS 'Protecting Vital Assets Workshop	2019-2021	1,2
Guest lecture at Faculty of Technology, Policy and Management (TBM) - TU Delft	2022	0,2
(Inter)national conferences		
IEEE Healthcom 2019 - Bogotá, Colombia	2019	1,0
APH Annual Meeting 2019 - Amsterdam, Netherlands	2019	0,2
Health RI - Moving forward - Online	2021	0,2
Health RI - Setting data in motion - Utrecht, Netherlands	2022	0,2
Other		
Presentation chair at the IEEE Healthcom 2019 – Bogotá, Colombia	2019	0.2
Technical program committee of IEEE Global Communications Conference: Communication & Information Systems Security	2019	0,4
Reviewer for Journal Annals of Telecommunication Elsevier	2019	0,2
Review for IEEE International Symposium on Computer-Based Medical Systems (CBMS)	2021-2022	0,4
Reviewer for Journal of Medical Internet Research (JMIR)	2022	0,2
Plenaries of the H2020 ASCLEPIOS project	2018-2022	4,0
Weekly meetings of the H2020 ASCLEPIOS project	2018-2022	4,0
Monthly meetings Bioinformatics & biomedical computing	2021-2022	1,0

Teaching	Year	ECTS
Lecturing and Tutoring		
Advanced Medical Imaging - teaching assistant	2020	0,5
Supervising and Mentoring		
Supervising master thesis	2020	0,5
Supervising Miniscriptie Biomedische wetenschappen	2022	0,5
Mentoring of junior scientific programmer in ASCLEPIOS project	2020-2022	4,0

Parameters of Esteem	Year
Grants	
Co-applicant of the awarded Horizon ExtremeXP project 2023-2026	2022
Awards and Prizes	
Best paper award of 2019 22nd Conference on Innovation in Clouds, Internet and Networks (ICIN)	2020

PUBLICATIONS

IN THIS THESIS

Marcela Tuler de Oliveira*, Alexandros Bakas*, Eugene Frimpong, Adrien ED Groot, Henk A Marquering, Antonis Michalas, and Sílvia D Olabarriaga. "A break-glass protocol based on ciphertext-policy attribute-based encryption to access medical records in the cloud." In: Annals of Telecommunications (2020), pp. 1–17. Springer

Marcela Tuler de Oliveira, Hai-Van Dang, L úcio HA Reis, Henk A Marquering, and Sílvia Delgado Olabarriaga. "AC-AC: Dynamic revocable access control for acute care teams to access medical records." In: Smart Health 20 (2021), p. 100190

Marcela Tuler de Oliveira*, Yiannis Verginadis*, Lúcio H. A. Reis, Evgenia Psarra, Ioannis Patiniotakis, and Sílvia Delgado Olabarriaga. "AC-ABAC: Attribute-Based Access Control for Electronic Medical Records during Acute Care." Accepted for publication in Expert Systems With Applications (2022)

Marcela Tuler de Oliveira, Lúcio Henrik Amorim Reis, L., Henk A. Marquering , Aeilko Having Zwinderman, and Sílvia Delgado Olabarriaga. "Perceptions of a secure cloud-based solution for data sharing during acute stroke care: qualitative interview study". Accepted for publication in Journal of Medical Internet Research JMIR (2022).

Marcela Tuler De Oliveira, Lúcio Henrik Amorim Reis, Yiannis Verginadis, Diogo Menezes Ferrazani Mattos, and Sílvia Delgado Olabarriaga. "SmartAccess: Attribute-Based Access Control System for Medical Records based on Smart Contracts". Accepted for publication in IEEE Access (2022).

OTHER PUBLICATIONS

Marcela Tuler de Oliveira, Lucio HA Reis, Ricardo C Carrano, Flavio L Seixas, Debora CM Saade, Celio V Albuquerque, Natalia C Fernandes, Silvia D Olabarriaga, Dianne SV Medeiros, and Diogo MF Mattos. "Towards a blockchain-based secure electronic medical record for healthcare applications." In: ICC 2019-2019 IEEE International Conference on Communications (ICC). IEEE. 2019, (pp. 1–6).

Marcela Tuler Oliveira, Gabriel R Carrara, Natalia C Fernandes, Célio VN Albuquerque, Ricardo C Carrano, Dianne SV Medeiros, and Diogo Menezes Ferrazani Mattos. "Towards a performance evaluation of private blockchain frameworks using a realistic workload." In: 2019 22nd conference on innovation in clouds, internet and networks and workshops (ICIN). IEEE. 2019, (pp. 180–187).

Marcela Tuler de Oliveira, Antonis Michalas, Adrien E. D. Groot, Henk A. Marquering, and Sílvia Delgado Olabarriaga. "Red Alert: Break-Glass Protocol to Access Encrypted Medical Records in the Cloud." In: HealthCom 2019- International Conference on e-health Networking, Applications and Services. IEEE. 2019.(pp. 1-7).

Marcela Tuler de Oliveira, Lúcio HA Reis, Ricardo C Carrano, Flavio L Seixas, Debora CM Saade, Celio V Albuquerque, Natalia C Fernandes, Sílvia Delgado Olabarriaga, Dianne SV Medeiros, and Diogo MF Mattos. "Towards a blockchain-based secure electronic medical record for healthcare applications." In: ICC 2019 IEEE International Conference on Communications (ICC). IEEE.(pp. 1–6).

Tom Tervoort, **Marcela Tuler De Oliveira**, Wolter Pieters, Pieter Van Gelder, Silvia Delgado Olabarriaga, and Henk Marquering. "Solutions for mitigating Cybersecurity risks caused by legacy software in medical devices: a scoping review." In: IEEE Access 8 (2020),(pp. 84352–84361).

Taridzo Chomutare, Kassaye Yitbarek Yigzaw, Silvia Delgado Olabarriaga, Alexandra Makhlysheva, **Marcela Tuler de Oliveira**, Line Silsand, Dagmar Krefting, Thomas Penzel, Christiaan Hillen, and Johan Gustav Bellika. "Healthcare and data privacy requirements for e-health cloud: A qualitative analysis of clinician perspectives."In: 2020 IEEE International Conference on E-health Networking, Application & Services (HEALTHCOM). 2021. IEEE.(pp. 1–8).

Lúcio HA Reis, **Marcela Tuler de Oliveira**, Diogo MF Mattos, and Sílvia D Olabarriaga. "Private Data Sharing in a Secure Cloud-based Application for Acute Stroke Care." In: 2021 IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS). 2021,(pp. 568–573). IEEE.

Lúcio HA Reis, **Marcela Tuler de Oliveira**, James Bowden, Dagmar Krefting, Sílvia D Olabarriaga and Diogo MF Mattos, "Cryptography on Untrustworthy Cloud Storage for Healthcare Applications: A Performance Analysis," 2021 XI Brazilian Symposium on Computing Systems Engineering (SBESC), 2021, pp. 1-8.

Moritz Platt, Anton Hasselgren, Juan Manuel Román-Belmonte,**Marcela Tuler De Oliveira**, Hortensia De la Corte-Rodríguez, Sílvia Delgado Olabarriaga, E Carlos Rodríguez-Merchán, Tim Ken Mackey. "Test, trace, and put on the blockchain?: A viewpoint evaluating the use of decentralized systems for algorithmic contact tracing to combat a global pandemic." In: JMIR Public Health and Surveillance 7.4 (2021), e26460.

Kassaye Yitbarek Yigzaw et al. "Chapter 14 - Health data security and privacy: Challenges and solutions for the future." In: Roadmap to Successful Digital Health Ecosystems. Ed. by Evelyn Hovenga and Heather Grain. Academic Press, 2022, pp. 335–362.

Lúcio Henrik Amorim Reis, **Marcela Tuler de Oliveira** and Sílvia Delgado Olabarriaga, 2022, July. "Fine-grained Encryption for Secure Research Data Sharing". In 2022 IEEE 35th International Symposium on Computer-Based Medical Systems (CBMS) (pp. 465-470). IEEE.

* These authors contributed equally

ACKNOWLEDGMENTS

First and foremost, I thank God for illuminating my path and for all the blessings and achievements. To some people who accompanied me and were fundamental for realising this dream, I express my sincere gratitude and the importance they had, and still have, in this achievement.

I want to thank my family for their patience and unconditional support. Especially my son Theo, who is the reason for my joy and gives me the strength to move forward. To my husband, Thales, my biggest fan, who supports me in all our adventures. Thank you for moving with me to the Netherlands and starting over our lives here. Without you two, I would not make it.

With the same spirit, I would like to thank my parents, Susana e Marcelo, for helping me throughout my life, supporting my choices and loving me unconditionally. Likewise, I would like to thank my brother Felipe for standing close to me, being my rock, and making my life joyful. To my in-laws for always being present and supporting us. To my grandmother, Maria Tereza, I want to thank you for being my role model for women ahead of your time.

I need to provide a special thanks to my supervisors. Silvia, I am grateful for all the support and guidance. I learned a lot from you about research and life. Thanks for believing in my potential and for your dedication. I am lucky to have you as my mentor. Koos, thanks for being a role model of a professor and always being interested in my research and development. You were always very positive about my work and were responsible for keeping me motivated throughout these years. Henk, thanks for your guidance and for being my constant reminder of the challenges of acceptance of new technical solutions in the medical field. I learned a lot from you about how to think more interdisciplinary way. Thanks for inviting me to your research group, where I was close to my research stakeholders and made great friends.

The chapters of this thesis were carried out after lengthy discussions and collaboration with several people I am proud to call co-authors. Therefore I want to thank Lucio, Yiannis, Diogo, Alex, Hai-Van, Eugine, Antonis, Ioannis, Jenny, Adrien and all my partners in the ASCLEPIOS project.

Last but not least, I want to thank my paranymph, Ricardo and Henk, who, besides friends, dedicated their time to relevant discussions of my work. To all my friends I made in these four years, Nila, Praneeta, Nerea, Manon, Maarten, Ricardo, Lucas, Roel, Neils, Mahsa, Daphnee, Dolly and many others, thanks for all the fun and support; because of you, the PhD life was easier.

ABOUT THE AUTHOR

Marcela Tuler de Oliveira was born in Campos dos Goytacazes, Rio de Janeiro, Brazil, on July 22nd 1991. The daughter of a mathematics teacher, she always liked numbers and logic. Inspired by her older brother, in 2010, she pursued telecommunications engineering as a bachelor at Universidade Federal Fluminense (UFF) in Rio de Janeiro, Brazil. Since the second semester, she has been involved in research projects. In 2013, Marcela was offered a one-year scholarship from the Brazilian government to study at the University of Florida-USA as part of the "Science Without Border" exchange program. At the University of Florida, besides the classes from her original curriculum, she took extra credits to learn about Entrepreneurship, Leadership and Ethics, courses offered by the graduate program. In 2016, she got one of the most desirable and competitive internships at the most prominent television company in the country. After finishing the one-year compulsory internship, Marcela was sure that her future was in academia. Even before graduation, Marcela had already taken half of the classes in the Master's program. Then, she officially started her Master's in August 2017 after graduation. By November 30th, 2018, Marcela received a Master's degree in Electrical and Telecommunications Engineering from the Graduate Program at UFF, with her thesis entitled "Development of trust-based consensus mechanism for private permissioned blockchain". On December 3rd 2018, she started her PhD at Amsterdam UMC, University of Amsterdam. She was hired to participate in the ASCLEPIOS EU H2020 Project (Trusted digital solutions and Cybersecurity in Health and Care), where she researched the usage of data encryption and access control modelling for securing and sharing healthcare and research data. Marcela currently works as an assistant professor at the Technology, Management and Policies Faculty at the Delft University of Technology.