Identification of Data Privacy Challenges and Development of Solutions for the Edge Components of the Telemedicine Datapath

PhD Thesis

Zoltán Szabó
Supervisor: Dr. Vilmos Bilicki

A THESIS SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
OF THE UNIVERSITY OF SZEGED

Szeged
2023

1 Introduction

Integration of the healthcare sector and IT is increasing and escalating, but this acceleration is creating new types of challenges and obstacles that can make it extremely difficult to provide cutting-edge solutions that significantly improve patient care and the work of healthcare professionals.
In the following dissertation, I present my findings from an analysis of the security and accessibility of telemedicine applications. My work is divided into two parts, with the first thesis focusing on the analysis of a heterogeneous data path integrating cloud and edge computing, specifically on two largely neglected components of the edge, the processing and storage edge types, which pose a unique challenge from the perspective of access and authorization control, as they must enforce access control rules independently of the cloud, with similar efficiency and minimal latency. I present formal definitions of my own taxonomy used to distinguish between access control policy categories; a categorization of the various edge types and an examination of smartphone-based peer-to-peer networks, as a special type of the edge; a framework based on a customized concept of the Policy Enforcement Point (PEP) framework that implements access control principles for both the processing and the caching edges; my results with experimental implementations of the PEP-based framework on processing and storage edges; and finally an open-source, proprietary simulation tool designed to model patient pathways and generate the necessary parameters to validate the access control framework's results.
In my second thesis, I examine the security of frontend applications at the conclusion of data paths. This is a challenging area from an analytical standpoint, as the nature of frontend frameworks and the dynamic nature of navigation between components have made static code analysis solutions exceedingly difficult to implement. For this reason, I have devised a novel method for my own research using large language models (LLM) that can identify, based on the static source code of an application, the sections where sensitive, safety-critical data leakage may occur. To accomplish this, I first present my own categorization to differentiate sensitive data, which consists of three categories based on the level of possible harm caused by the disclosed data, and a second categorization to quantify the safety levels of application components. Then, I demonstrate the validity of the taxonomy used to quantify data sensitivity with the GPT-4 and GPT-3.5 APIs, using a list of variable names and random sample applications for each. This is followed by the validation of the second taxonomy on identifying the safety levels of components and the efficacy of detecting problematic code segments by integrating the results of the two taxonomy evaluations.

2 Thesis Group I: Data Privacy at the Edge of the Telemedicine Datapath

In the following thesis group, I examined the requirements of the telemedicine data path for access management. I defined the conditions that must be met at each designated point of the telemedicine data path in order to reconcile security requirements and responsiveness. I defined a four-element taxonomy for the categories of policy enforcement to be implemented. I presented the edge as a critical case for access control, its two primary categories for access control, the processing edge and the storage edge, and our results on a special case of the *edge, the analysis of the stability of smartphone peer-to-peer networks. Then, I defined two experimental environments, one for the processing edge and one for the storage edge, as well as my experimental results in said environments using a practical implementation of the access control framework in terms of latency. Lastly, I presented my patient flow simulator built with open source libraries, which can simulate the throughput of patients treated in a hospital ward and can be used to generate validation parameters for the developed access control framework in order to validate the system's delay tolerance.*

Publications related to this thesis: [J1],[J2],[J3],[C5],[C6],[C7],[C8],[F9],[F10],[F11],[F12]

2.1 Thesis I/1: Formal Definitions and Requirements of Telemedicine Access Control

I explored the complex requirements of modern telemedicine applications in terms of access control. I defined a taxonomy to formalize the different types of access control policies and the TAPE requirements necessary to ensure that the implementation of the defined policies can guarantee a balance between data privacy compliance and responsiveness at any point in the telemedicine infrastructure.

Publications related to this thesis: [J1],[C6],[F9],[F10]

In the healthcare industry, patient data are progressively becoming digital. The most significant challenge of this trend is that electronic health records (EHRs) must be easily accessible, searchable, and intelligible as patients migrate through the healthcare ecosystem. In addition, data must be structured and standardized to facilitate automated clinical decision support and other forms of machine processing.

To achieve interoperability, a number of established standards have emerged over the years, the most dominant and widely used being the Fast Healthcare Interoperability Resources (FHIR) [16], which was developed by the Health Level 7 (HL7) standards organization and owes much of its popularity to its high level of customizability.

However, the official documentation of the FHIR standard [21] contains only guidance on how the data structure could be extended to be compatible with traditional access management methods such as ABAC [33] and RBAC [24]. How precisely these access management methods should be implemented in a complex telemedicine system, and how they can be used to define competencies and meet the complex access management requirements of the healthcare industry, were not discussed or clearly established, emphasizing a significant issue with the FHIR and other standards.

One of the starting points of my research in this area was to determine if it would be possible to develop a solution that could be applied to any point along a complex, heterogeneous data path, with the same tools, definitions, and configurations. I defined the basic requirements that this solution had to meet as the TAPE requirements, which are as follows:

- **Transparency**: The access control solution should have as little of an effect as feasible on the data path throughput and performance of telemedicine applications.

- **Adaptability**: To safeguard sensitive data, the field of telemedicine requires stringent, very specific policies. ABAC and RBAC alone are insufficient because interoperability and interchangeability requirements require a significantly more dynamic and refined approach. The developed solution must also facilitate the formulation and evaluation of the most specific requirements.

- **Portability**: The developed solution must be deployable anywhere in the infrastructure. Edge computing's greatest asset is its ability to provide functionality even when the cloud is unavailable. This also implies that it must be able to operate between the edge and the cloud, between the edge and the endpoints, in the cloud, and in certain scenarios, even on the endpoints if they have the required resources.

- **Efficient**: Since many elements of the infrastructure lack sufficient memory and CPU capacity to execute the more complex transformations and data analysis, the developed implementation should spare them from more demanding operations.

I have defined the telemedicine access control principles based on the following taxonomy and requirements. The patient is the primary owner of the document, the physician who wrote or contributed to its creation is the secondary owner, and other physicians and family members have only partial access to the document. The system must also accommodate indirect access, in which the requester attempts to access the file as a group member. These corresponding categories of access should be determined by a combination of user roles, role groups, and document attribute information.

In certain instances, context-specific information is also required to ascertain the extent of access. Furthermore, a key requirement in the healthcare industry is that access does not imply complete access to all document elements. In many instances, access to information that could enable a third party to reconstruct highly sensitive events and elements is rigorously prohibited. The final requirement is also the most particular and difficult aspect of healthcare safety. Break-the-glass necessitates an access-control paradigm that provides immediate access to vital patient data to guarantee the delivery of life-saving care. When life-saving intervention is required and neither the patient nor the physician recording and processing their medical data is available to provide access, this is essentially what occurs. Typically, only a few documents are required for the healthcare provider in such a scenario, but in this instance, extremely complex transformations must be applied.

Based on this, I have defined the four categories of access control rules that follow:

- **Role Evaluation**: The policy has to determine whether based on the user's role or roles in the system, partial or full access should be provided;

- **Contextual Evaluation**: The policy has to determine whether the combination of the user's role, various attributes and contextual information form the basis for partial or full access;

- **Contextual Modification**: Aside from providing access, the policy should also transform the data, removing or altering specific fields;

- **Break-the-Glass**: A specific requirement of a healthcare application. In the case of an emergency, the policy should provide immediate access, while also encrypting or removing sensitive information.

2.2 Thesis I/2: Formal Definitions and Challenges of Access Control Beyond the Telemedicine Cloud

I proposed special categories of edge instances beyond the cloud that represent a special category of modern telemedicine infrastructure from the perspective of access management. I formally defined a categorization of these into storage and processing edges. I then discuss the increasingly prevalent smart device based peer-to-peer networks as an extreme case of edge solutions, and analyse their potential and stability to function as stand-alone edge networks.

Publications related to this thesis: [J1],[J2],[C5],[C7],[C8],[F10],[F11]

In contemporary network topologies that incorporate IoT, smart devices, edge computing, the integration of private and public cloud platforms, such as the one showcased in Figure 1, traditional access control methods are insufficient. To fulfill the requirements defined in Thesis I/1., it is necessary to devise a hybrid strategy that combines the benefits of the two traditional approaches, ABAC and RBAC. To manage the sensitive nature of the data and its processing requirements, it is essential that the evaluation points of access control have the ability be located anywhere within the infrastructure.

This portability includes cases where we have to treat the cloud as inaccessible or at least as having limited availability. It is no coincidence that fog, or edge computing, is proliferating, which attempts to reduce the data traffic between the cloud and the devices by creating different, self-sufficient micro-networks at the endpoints of the data path, which, by sharing responsibilities and aggregating functionality where they can, free up the cloud's capacity and communicate with it only for critical, absolutely necessary operations. However, while data collection, storage, and caching are already relatively solved and covered topics within edge computing, implementing access control operations is a significantly more challenging task.

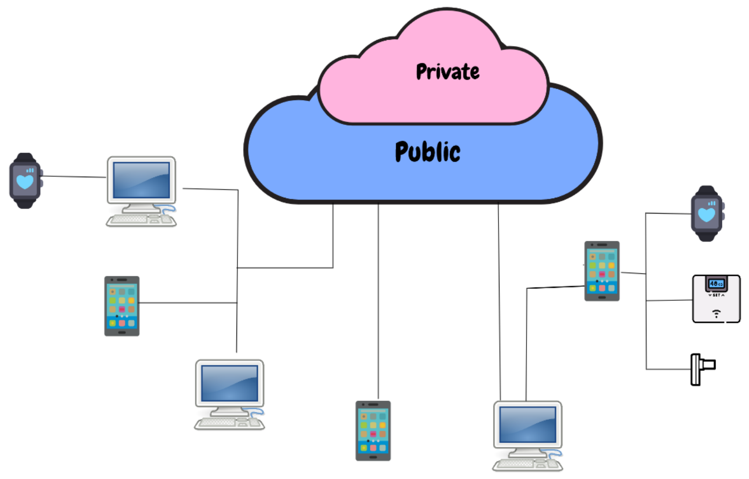
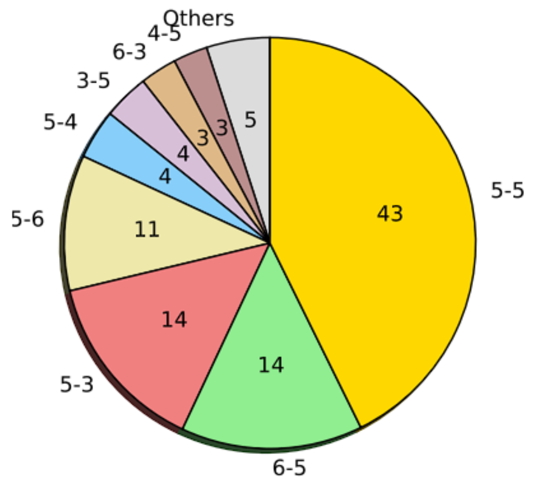


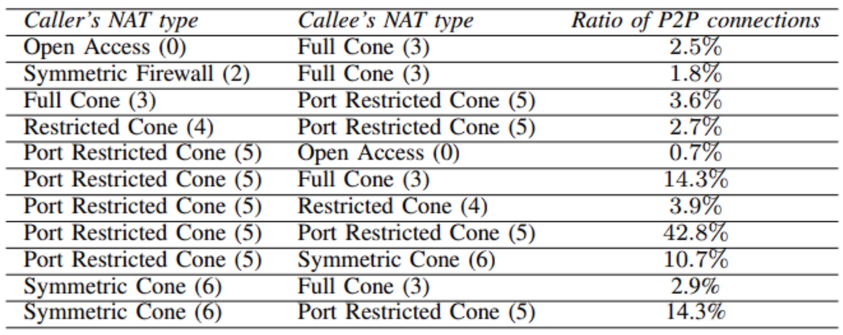
Figure 1: *An overview of a cloud- and edge integrated telemedicine infrastructure*

In such endpoints, data availability and speed of access cannot be compromised, but neither can the strict privilege management rules that must be followed in all other cases with telemedicine data. In the case of edge cases, we must take into account their reduced resources, which make it difficult to make certain functions more feasible.

Based on their capacities, we distinguish two major types of devices within the edge, which I have formally defined in the thesis. These are:

- **processing edge**: the strongest, highest capacity elements on the edge, with sufficient storage and processing capacity to process documents of up to 1000, and capable of evaluating all four defined entitlement management principles.
- **caching edge**: elements with a lower capacity than processing, but still capable of processing larger document volumes in the order of hundreds, and capable of handling at least the most critical privilege management rules, role evaluation and contextual evaluation.





Caller's NAT type	*Callee's NAT type*	*Ratio of P2P connections*
Open Access (0)	Full Cone (3)	2.5%
Symmetric Firewall (2)	Full Cone (3)	1.8%
Full Cone (3)	Port Restricted Cone (5)	3.6%
Restricted Cone (4)	Port Restricted Cone (5)	2.7%
Port Restricted Cone (5)	Open Access (0)	0.7%
Port Restricted Cone (5)	Full Cone (3)	14.3%
Port Restricted Cone (5)	Restricted Cone (4)	3.9%
Port Restricted Cone (5)	Port Restricted Cone (5)	42.8%
Port Restricted Cone (5)	Symmetric Cone (6)	10.7%
Symmetric Cone (6)	Full Cone (3)	2.9%
Symmetric Cone (6)	Port Restricted Cone (5)	14.3%

Figure 2: *Number of peer-to-peer connections with different NAT types.*

Within the edge, peer-to-peer connections between smart devices are a special case, as they mix many different types of devices in a dynamically changing environment. In order to map their exact capacities, with a particular focus on stability, our research team participated in a large-scale data collection and analysis campaign, during which we developed a proprietary application called Stunner, which was installed on the phones of the users participating in the campaign, and with their consent, performed various experiments to establish peer-to-peer connections, the results of which were collected with the permission of the participating users, naturally removing their personal data information. The key metrics included elements such as the type of network connection of the device at the time of measurement, the type of NAT, its load, the success of the peer-to-peer connection established and, if successful, its length. The dataset collected during the campaign was eventually made available, creating one of the largest databases of its kind. One result of the peer-to-peer analysis can be seen on Figure 2.

2.3 Thesis I/3: Policy Enforcement Implementations for the Processing and the Caching Edge

I presented the implementation of the proposed access control solution, then I will set up test environments to represent both edge types, and sample rules to validate the effectiveness of the implementation under increasing data volumes. During the measurements, I examined the resource requirements of the nodes performing the evaluation, as well as the latencies measured for the data retrieval processes. I present the measured latencies which confirmed, that for reasonable amounts of data at the various edge types my policy categories met the requirements established in the previous theses.

Publications related to this thesis: [J1],[J2]

A popular concept for an enforcement point, which I hypthosized to be able to meet the requirements is the concept of Policy Enforcement Point (PEP), developed by the standards organization OASIS as part of its eXtensible Access Control Markup Language standard [23], an extension of the classic ABAC model, also known as policy-based access control or PBAC. While we were committed to developing a custom, fine-grained security solution that is not subject to the strict limitations of the XACML standard, the concept of the PEP-based architecture was found to be well suited to our needs.

To test our concept of policy enforcement, our research group chose a promising new solution called Open Policy Agent (OPA) [18], available in Golang and WebAssembly, which made it perfectly suitable to be placed at multiple points of the datapath - in the cloud, in the various edge types or even in web applications. OPA also permit us to store the information necessary for decision making in JSON format and define the various policies in its own scripting language, Rego, which can later be accessed via a well-defined REST interface with an HTTP POST request containing the contextual information to be filtered or evaluated (in our case, the medical records).

As for the evaluations, I created two types of testbeds for the concept: one simulated a smaller infrastructure with a more capable processing edge module, while the other simulated a caching edge without a connection to the cloud, which had to evaluate access to the locally stored data. Based on my assumptions regarding these edge types, for the processing edge, simulated with the standalone OPA version, I assumed that all four evaluation categories will cause acceptable latency even for document sizes on the order of one thousand, whereas the capabilities of the caching edge, simulated with the WebAssembly runtime will be significantly more limited, but should be able to evaluate policies in the role evaluation and contextual evaluation categories efficiently. The experiments were conducted on realistic FHIR Observation documents derived from a database of telemedicine applications by the Inclouded Development Team of the University of Szeged's Department of Software Engineering. First, 10 documents were retrieved, then 20, 50, 100, 200, etc., all the way up to 2,000, and each retrieval was performed multiple times, with the averaged results being used for evaluations.

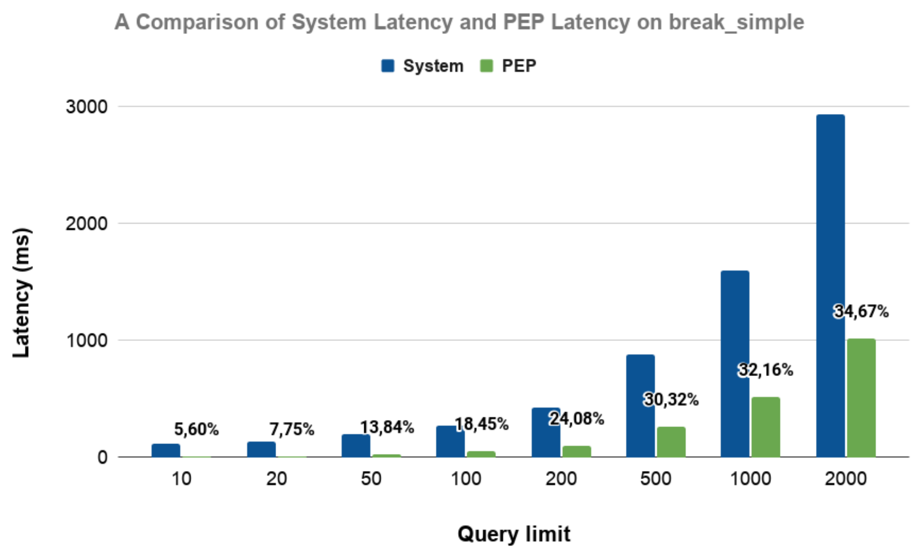


Figure 3: *Comparison of System Latency and policy evaluation latency on a Break-the-Glass policy.*

Figure 3 shows the latency of the Break-the-Glass policy category, which is considered the most complex, in relation to the total runtime of the document request within the infrastructure used for testing. It can be clearly seen that even for 2000 documents, the evaluation averaged less than 1000 ms, which is less than 40% of the total document request runtime.

As far as the WebAssembly runtime is concerned, part of the investigation there was to assess exactly what tools on the edge might be suitable to fulfil the role of caching edge. To this end, the WebAssembly OPA implementation was configured with contextual evaluation and role evaluation policies and run on different devices embedded in a web application that performed local data retrieval and access evaluation before granting access. The latencies have already been significantly greater here due to the limited resources of the devices, and as shown in the comparison shown in Figure 4, while the version running in the desktop PC Chrome browser produced relatively acceptable results even with larger amounts of documents, the smartphone and tablet version has already begun to dramatically increase the latency over 200 documents. Meaning that up to 100-200 documents any of the devices could play the role of the caching edge, but based on the results, it is possible that this edge type is unsuitable for the effective evaluation of larger document sets.

The results confirmed that for a realistic data volume the extra latency due to access control for processing nodes was mostly below 500 ms, which was on average less than 25% of the execution time of the whole process. For storage nodes, while the number of implementable access control operations was limited and the execution times were higher,

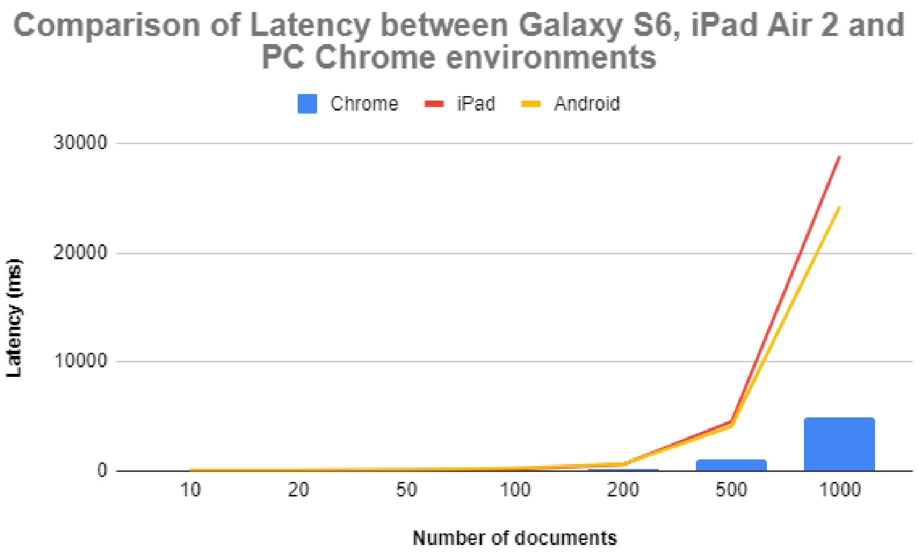


Figure 4: *Comparison of WebAssembly runtime latencies*

the execution times averaged less than 1 second regardless of the capacity of the execution environment.

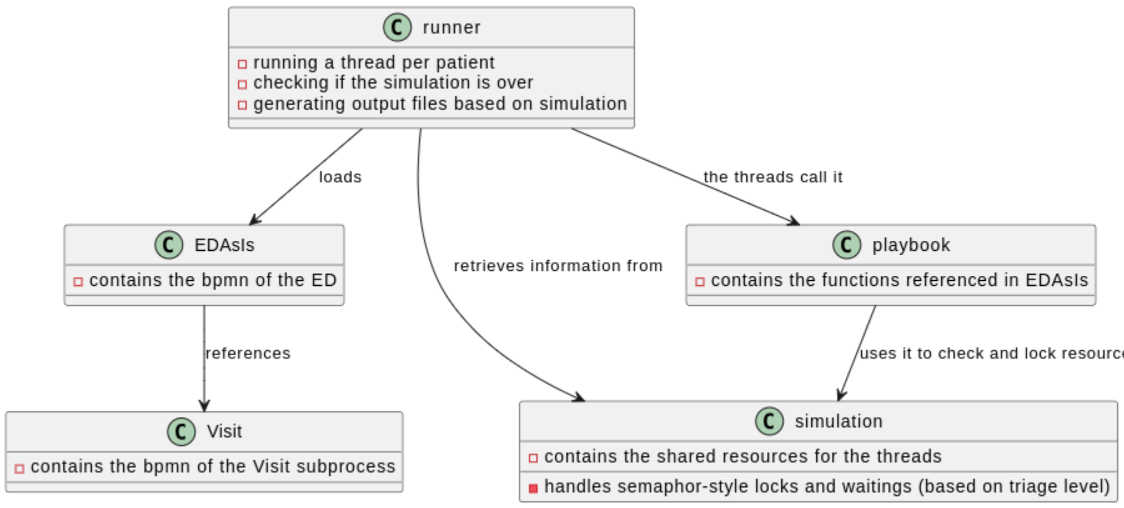
2.4 Thesis I/3.1: An Open Source Patient Flow Simulation Tool for Validation of Results

I developed a simulation tool using open source libraries and tools that can simulate the distribution of patients and their waiting times in a hospital ward. The developed simulation tool can be used to validate the extent to which the increased waiting times due to the access control implementation from the previous theses might slow down the process based on the amount of handled data, and the impact this can have on the telemedicine data path, the patient flow churn and waiting times at crucial parts of the care process.

Publications related to this thesis: [J3],[F12]

Patient Flow [19] refers to the time a patient spends in the healthcare system from arrival to departure. The basic objective is to minimize this time as much as possible in order to ensure quality of care, and this principle has only been emphasized in recent years by epidemiological measures, which have limited the number of patients staying in one place at the same time, and to protect against the epidemic, many have made suggestions and validation tools [26] about how much maximum waiting time a patient can spend in one location, so as to minimise the chance of infection.

Although there are various tools available on the market that are suitable for defining and running such simulations, their capabilities are mostly limited and their use requires considerable learning time and energy investment, making it difficult for those who have the most information on the subject and would be able to define the most accurate simulation models. The aim of our research team was to create a simulation tool by creating open source elements that run models that could be defined by external researchers or even doctors using a simple visual descriptive language with an easy to customize simulation. The tool is also crucial for validating the results of previous theses, as its use has also offered the opportunity to check how the increased run times with authority management evaluations have impacted waiting times and patient churn, in fact, even in different scenarios and circumstances.



runner
running a thread per patient
checking if the simulation is over
generating output files based on simulation
loads
the threads call it
retrieves information from
EDAsIs
contains the bpmn of the ED
playbook
contains the functions referenced in EDAsIs
references
uses it to check and lock resources
Visit
contains the bpmn of the Visit subprocess
simulation
contains the shared resources for the threads
handles semaphor-style locks and waitings (based on triage level)

Figure 5: *Structure of the Patient Flow Simulator*

To run the tool, we used the open-source Python library called SpiffWorkflow [20], which is suitable for defining and running business process models, and for modeling the Camunda [15], one of the most popular business process modeling tools. The various main components of the developed software are shown in Figure 5

For the evaluation of the finished simulation tool, we re-created the model of an Italian research group [29], which depicted an Emergency Department (ED), whose structure could be adapted with minimal changes to similar departments of Hungarian hospitals. For the parameterization of the simulation we used the patient traffic data of Somogyi Kaposi Mór Practicing Hospital [32] through various simulations. Based on the results, our tool proved to be suitable for continued use in more complex evaluations and for the production of validation parameters.

3 Thesis Group II: Potential Data Leak Detection of Progressive Web Applications with Large Language Models

In the subsequent group of theses, I focused on a crucial component of the telemedicine infrastructure, namely the front-end applications at the end of the data path. I defined a taxonomy that defines and ranks sensitive data in applications based on the impact of data leaks, and another taxonomy to determine the protection level of applications, in such a way that the defined categories can be transferred to LLMs using prompt engineering techniques. Then, I presented my results in classifying the elements of a variable-name dictionary and then in detecting sensitive data from open-source front-end applications via the GPT-3.5 and GPT-4 APIs, with which I have validated the hypothesis that through the complex knowledge of the

LLMs at the level of the current GPT models, machine learning in the classical sense can in some cases already be derived by passing the necessary knowledge in the form of well-written prompts to the LLMs. Lastly, I discussed my results in static code-based detection of potential application vulnerabilities by evaluating the application protection level classification and then combining it with the results of the sensitivity detection.

Publications related to this thesis: [J4]

3.1 Thesis II/1: Formal Categorization of Sensitivity and Protection Levels

I defined a taxonomy of sensitive data in front-end applications, which categorises them into three different categories based on the level of damage caused by their leakage and unauthorised access. I then presented another taxonomy, which divides the protection levels of the application components into categories such that they are representative of the protection required for the sensitivity categories.

Publications related to this thesis: [J4]

Due to the proliferation of large language models (LLMs) and their widespread use in applications such as ChatGPT, there has been a significant increase in interest in AI over the past year. Programming, and specifically the process of static interpretation, analysis, and documentation program code, is one of the most promising fields due to the large amount of data and sources the GPT models were trained on and their capabilities of contextual interpretation and analysis. Due to these capabilities, I have hypothesized that using models such as the GPT-3.5 and GPT-4, it could now be possible to tackle types of static code analysis that proved to be too complex until now. One of these areas is the static code analysis of progressive web applications at the end of the data path, which, due to their complex modularity and navigational design, were claimed to be hard to analyze, and the only methods so far that focused on their analysis were custom-developed frameworks and tools [25], linting, and extensive unit testing.

These solutions, however, focused on the semantics and quality of the code, excluding possible problems and vulnerabilities that require a deeper understanding of how the applications work and what kind of data they handle. In a frontend application, a vulnerability that fits into this description is the improper isolation or protection of sensitive or critical elements—a variation of the CWE-653: Improper Isolation or Compartmentalization vulnerability from the Common Weakness Enumeration database [30]—meaning that public or unprotected interfaces of the application have the capacity and authority to access data and operations. If this gets combined with server-side errors, software bugs, or malicious activity on the user's part, it could lead to various degrees of data leakage (identified as the CWE-200: Exposure of Sensitive Information to an Unauthorized Actor family in the database). To efficiently detect this type of vulnerability, the first crucial step is to detect the data handled by the application that could be considered sensitive.

To achieve this, I have defined a categorization for both the sensitivity of the data and the protection level of the software components in a way that is compatible with prompt engineering techniques [22], such as ”few shot examples” and ”chain-of-thought”, to make them not only logical but also compatible and usable with the interpretation capabilities of LLMs. The basis for the sensitivity level categorization was the level of impact caused by the leakage of the data and the amount of said data required to access crucial personal, financial, healthcare, or other similar information about the users.

The categories are defined as follows::

- **Level 1 (Low Sensitivity)**: The accumulation and compilation of large quantities of data at this level is required to infer confidential information and create abuse opportunities. Sensitive information includes, among other things, a user's behavior history, a list of websites visited, products and topics of interest on a website, and search history.
- **Level 2 (Medium Sensitivity)**: By obtaining data at this level, it is possible to retrieve and compile potentially exploitable information. Solutions such as two-factor authentication, regular email and SMS notifications, and exhaustive logging in affected applications can mitigate the effects of a potential breach. This is the largest and most extensive group. It includes data such as, usernames, passwords, private records, political views, sexual orientation, IP address, physical location, and hectic schedules are examples of sensitive information.
- **Level 3 (High Sensitivity)**: The data at this level is sensitive in and of itself, and its acquisition or disclosure can have severe legal repercussions and cause extensive harm. Examples include health information, medical history, social security number, driver's license, and credit card information.

And the scale of protection level is the following:

- **Protection Level 0**: The component has no protection at all.
- **Protection Level 1**: Bare-bones authentication: the application checks whether the user trying to access the resource is logged in to the application.
- **Protection Level 2**: RBAC [24]: In addition to being logged in, the user has different roles that define their scope of privileges within the application.
- **Protection Level 3**: ABAC [33]: In addition to the login and possible role scopes, other attributes such as time, physical location, or type/id of device used are checked before granting access.

3.2 Thesis II/2: Sensitivity Analysis of Web Application Data and Components

I developed a GPT-enhanced methodology that uses the categorization defined in the previous thesis to detect sensitive data in front-end applications and, based on this data, to tag the code elements that focus on operations on such data. I validated the categorisation using an artificial intelligence-based evaluation that classifies the elements of a 200-word collection of variable names into one of the defined categories. I followed this by analyzing a total of 292 components from open source applications to detect sensitive data and based on that, identify the services handling sensitive information in the application.

Publications related to this thesis: [J4]

To validate the categories developed and the analysis capabilities of the GPT-3.5 and GPT-4 APIs, which at the time of writing were the most efficient GPT models available and promptable via API, we compiled a dictionary of 200 variable names, equally divided among the Non-Sensitive, Low Sensitivity, Medium Sensitivity, and High Sensitivity categories. In preparing the dictionary, we intentionally made it difficult for the models to achieve perfect accuracy: some variant names had deliberate misspellings, others were in foreign languages such as Hungarian, German, or Italian, and others contained a combination of a more sensitive word and a less sensitive ending.

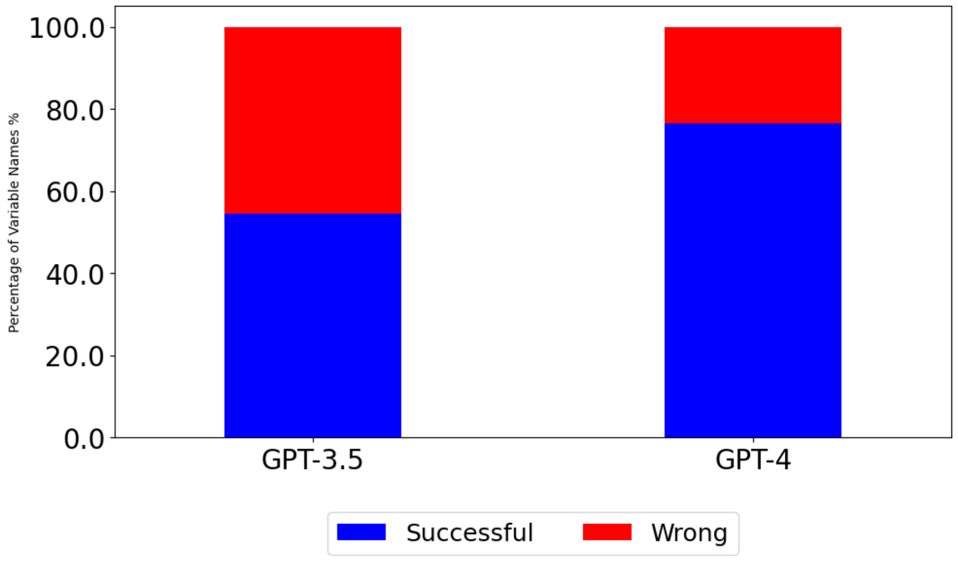


Figure 6: *Comparison of GPT-3.5 and GPT-4 on the variable name evaluation*

The results of this validation can be seen in Figure 6. The GPT-3.5 API incorrectly classified 45.5% of the dictionary, with the majority of cases being trivial. In contrast, GPT-4 only made errors in 23.5% of the cases, and a deeper examination of the errors revealed that it only failed on the explicitly difficult terms, and in more than half of the cases, it correctly identified the errors as sensitive data but incorrectly categorized the exact category. Although both models were used throughout our investigations, preliminary results suggested that GPT-4 might be able to produce more accurate results by analyzing codes, whereas GPT-3.5 appeared more prone to failure in all but the most straightforward cases.

Due to our research team's experience [27, 28] with the Angular framework [17] we chose it for the following evaluations as well. Our methodology is predicated on the notion that in Angular web applications, sensitive data is centralized in Services, which are accessed and utilized as singletons by Angular framework classes via their methods. To detect sensitivity levels and vulnerabilities based on static code, we had to first detect the elements that appear to be sensitive in the Component classes using the context handling capabilities of the GPTs and identify which of these elements are handled by Services.

Our research team collected thousands of public Angular projects from GitHub using a crawler algorithm in order to conduct various source code analyses. We minified the

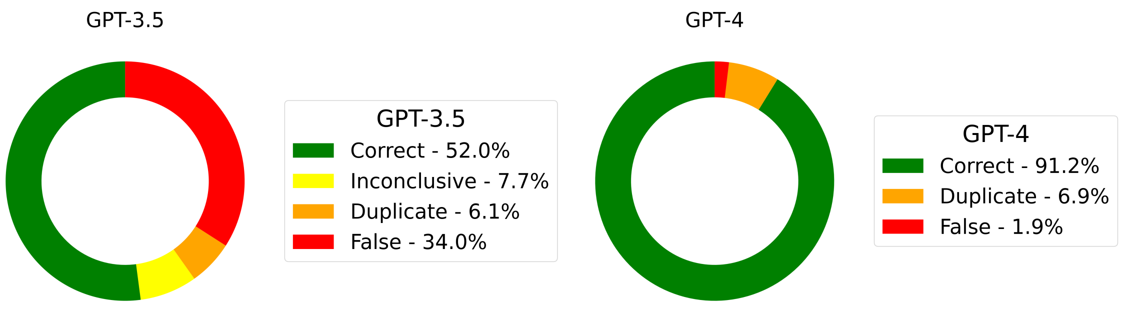


Figure 7: *Comparing the performance of models in detecting sensitive data*

TypeScript source files of the projects by labeling, removing whitespace characters, and removing line breaks, and then randomly sampled 12 of the largest, most complex, and largest projects, which yielded a sum of 292 Components to be analyzed with the sensitivity and protection level categories to detect vulnerabilities.

The first step was to iterate through the 292 Components via prompts, and identify variables and objects that contain sensitive data. From the results seen in Figure 7 it is clear that GPT-3.5 marked significantly more attributes wrongly as sensitive compared to GPT-4. False counts cases where the detection was incorrect; perhaps a function or an unimportant variable was marked as sensitive data instead of a variable or object; Duplicate counts cases where the same sensitive data was marked more than once within the same file; and Inconclusive counts cases where the sensitive data was correctly detected but the justification for the sensitivity was incomplete or incorrect.

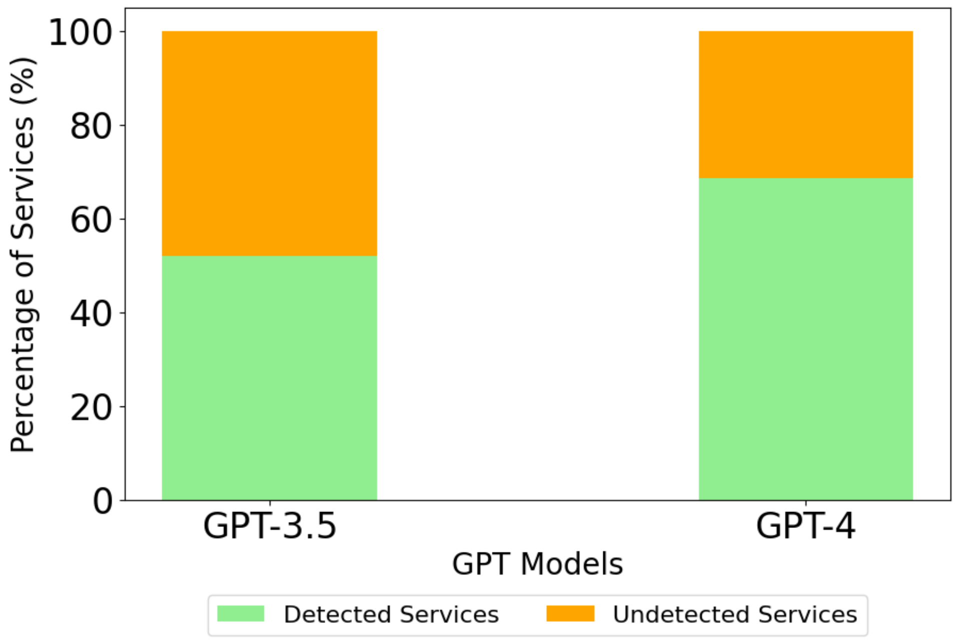


Figure 8: *Proportions of sensitive services successfully and unsuccessfully detected by the models*

Next, based on these attributes, the Services that served as the source or target for the sensitive data were identified. The results can be seen in Figure 8, and although GPT-

4 was clearly more efficient here, it made a significant number of errors in consistently detecting sensitive Services. After an evaluation of the results, it turned out that it was due to some particularly interesting bad practices on the part of the developers, which included the design of monolithical Services handling multiple responsibilities, or various circumventions of the injection principle of Angular.

3.3 Thesis II/3: Protection Level Analysis and Vulnerability Detection for Web Applications

I defined a GPT-supported static source code analysis pipeline that uses the protection level categorisation to identify the protection level of components in a frontend application and then uses the results from the protection level and sensitivity level evaluations to detect vulnerabilities where components are not protected or whose protection level is insufficient for the sensitivity of the data they handle, in line with the CWE-653 type software vulnerability.

Publications related to this thesis: [J4]

The pipeline's prompts were developed using the principles of prompt engineering [31], a relatively new discipline, through multiple cycles of experimentation; the used prompts are included in the Appendices of the dissertation. Criteria included plain wording, the inclusion of rules prohibiting the reappearance of anomalies discovered during initial testing, and the inclusion of examples pertinent to the expected response format. To accomplish this, we used larger, more comprehensive prompts, which allowed us to avoid both format errors and the inclusion of uptake prompt engineering techniques such as ”few shot example”, where each prompt was provided with at least one sample input and a corresponding sample output, and chain-of-thought, where we provided the thought and logic flow in the prompts in addition to simple rules, which helped to provide the correct deductions and avoid various errors.

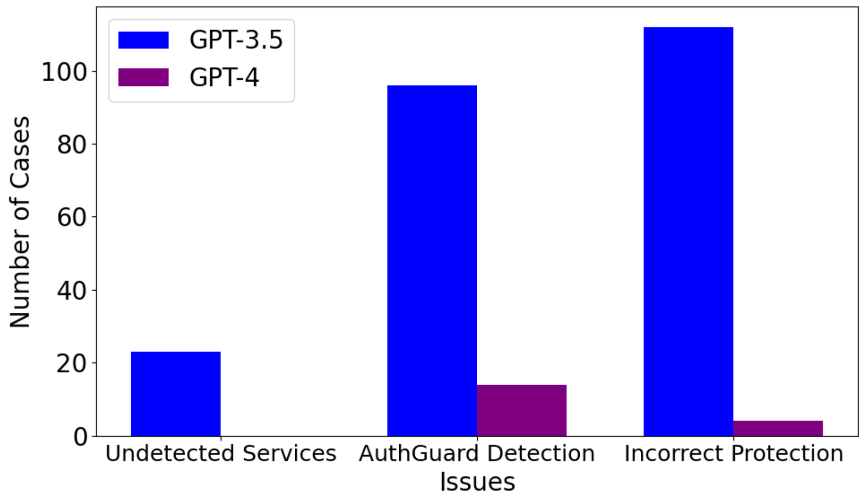


Figure 9: *Comparison of errors made by the models while analyzing the Components*

The complete analysis process consists of the following steps:

1. **Minifying Code Base**: The .ts source files of Angular applications have been minified, removing line breaks and whitespace characters.

2. **Sensitive Element Detection**: The minified Component files were passed one by one to the GPT API. The analysis first identified the sensitive data, explaining how it appears in the application, how it is used, and then an aggregation script used this information to select the Services that were involved in read or write operations on sensitive data in the project. For each occurrence of the tagged Services, the sensitivity level of the data they handle is also determined.

3. **JSON Mapping of Project Files**: The minified Component files were passed one by one to the GPT API. As a result of this step, a JSON file is created for each project, in which the components of the application are reduced to JSON objects, containing only the information needed for the current (or planned future) tests, such as the parent-child component relationships, the injected Services, and their used data members and operations.

4. **Protection Level Discovery**: The JSONs resulting from the previous step, along with the minified Router configurations and AuthGuards, were passed one by one to the GPT API, which added the protection level corresponding to our scale. A Python script then unified these protection levels, assigning the child components the protection level of their parent components.

5. **Vulnerability Detection**: By aggregating the results of the Protection Level Discovery and Sensitive Element Detection steps in an xlsx file, the vulnerabilities from the tested projects are detected, the cases where a Component using a sensitive Service does not have a high enough protection level.

The results of Component analysis steps are presented in Figure 9, with the majority of Incorrect AuthGuard Detection issues for GPT-4 being caused by the nesting of parent and offspring modules within an application, which made it difficult to detect AuthGuards accurately. During the investigations, the maximum level of AuthGuard was detected in the majority of cases, resulting in only four cases of inadequate protection level out of 292 investigated cases. When evaluating the results of GPT-3.5, we have found, that despite employing prompt engineering techniques and setting the temperature of the models to 0, we encountered difficult-to-explain problems, including the appearance of non-existing AuthGuards and inconsistent protection levels for the same AuthGuard.

As can be seen in Figure 10, our assumptions about GPT-3.5 have been confirmed, accumulating the poor results of the previous steps and resulting in a total of only 1% of vulnerabilities successfully detected. However, our initial naive assumption about the identification of sensitive Services had very serious consequences for GPT-4 in the first round. A significant part of the problem was identified to be caused by the monolithic Services, where Services did not have one well-defined task and role but rather developers crammed several similar procedures into them without any consideration that all levels from Non-Sensitive to High Sensitivity data categories could appear within a single Service. This problem caused the sensitivity level of the Services to vary enormously in many cases, where a Service was identified as having either Medium or High Sensitivity in a single Component, while in another it was declared irrelevant to sensitive data.

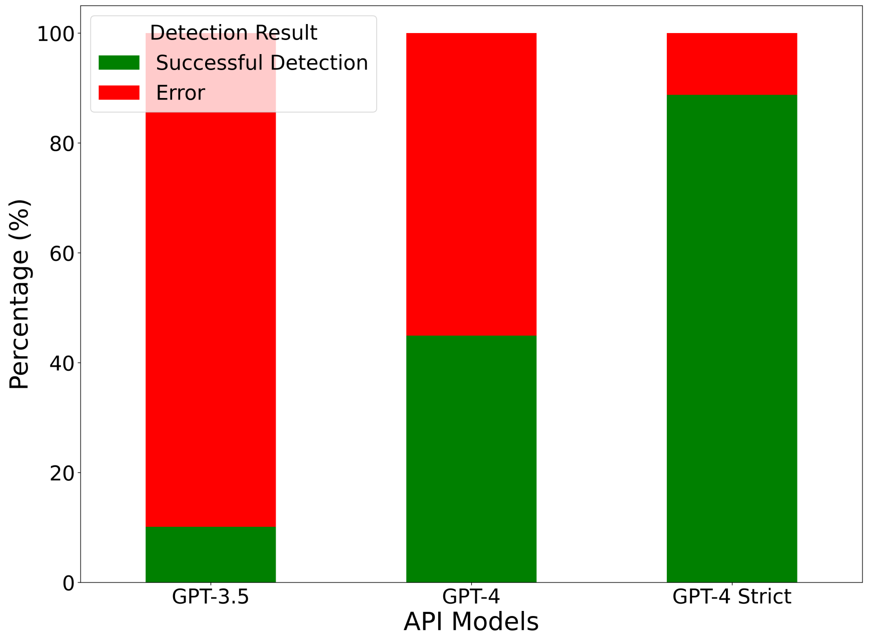


Figure 10: *Success rates of the two models and the strictened detection rules*

To improve the detection results somewhat and address these issues, a major strickten-ing of vulnerability detection rules was applied to GPT-4. A very spectacular improvement in the results can be seen as the GPT-4 Strict results.

4 Contributions of the thesis

In the **first thesis group**, I defined formal access control policy categories for telemedicine applications; proposed a categorisation of the edge beyond the cloud based on their capabilities for access control; explored the capabilities of smartphones to form a stable peer-to-peer network; and evaluated my policy categories using test environments simulating the various edge types. I developed an open source patient flow simulation tool, which can be used to generate validational constraints for the developed solution. Detailed discussion can be found in Chapter 3.

I / 1. I explored the complex requirements of modern telemedicine applications in terms of access control. I defined a taxonomy to formalize the different types of access control policies and the TAPE requirements necessary to ensure that the implementation of the defined policies can guarantee a balance between data privacy compliance and responsiveness at any point in the telemedicine infrastructure.

I / 2. I proposed special categories of edge instances beyond the cloud that represent a special category of modern telemedicine infrastructure from the perspective of access management. I formally defined a categorization of these into storage and processing edges. I then discuss the increasingly prevalent smart device based peer-to-peer networks as an extreme case of edge solutions, and analyse their potential and stability to function as stand-alone edge networks.

I / 3. I presented the implementation of the proposed access control solution, then I will set up test environments to represent both edge types, and sample rules to validate the effectiveness of the implementation under increasing data volumes. During the measurements, I examined the resource requirements of the nodes performing the evaluation, as well as the latencies measured for the data retrieval processes. I present the measured latencies which confirmed, that for reasonable amounts of data at the various edge types my policy categories met the requirements established in the previous theses.

I / 3.1. I developed a simulation tool using open source libraries and tools that can simulate the distribution of patients and their waiting times in a hospital ward. The developed simulation tool can be used to validate the extent to which the increased waiting times due to the access control implementation from the previous theses might slow down the process based on the amount of handled data, and the impact this can have on the telemedicine data path, the patient flow churn and waiting times at crucial parts of the care process.

In the **second thesis group**, I explored the comprehensive and analytical capabilites of the popular models GPT-3.5 and GPT-4 to detect vulnerabilities in complex frontend applications using only static code analyis. I defined taxonomies of data senstivity and application component protection, based on the impact of data leakage; evaluated said categorizations using the GPT APIs first with a dictionary of variable names, then with a set of open source Angular projects; and finally, based on the results I've showcased the effectiveness of vulnerability detection using these definitions and techniques. Detailed discussion can be found in Chapter 4.

II / 1. I defined a taxonomy of sensitive data in front-end applications, which categorises them into three different categories based on the level of damage caused by their leakage and unauthorised access. I then presented another taxonomy, which divides the protection levels of the application components into categories such that they are representative of the protection required for the sensitivity categories.

II / 2. I presented a GPT-enhanched methodology that uses the categorization defined in the previous thesis to detect sensitive data in front-end applications and, based on this data, to tag the code elements that focus on operations on such data. I validated the categorisation using an artificial intelligence-based evaluation that classifies the elements of a 200-word collection of variable names into one of the defined categories. I followed this by presenting the results obtained on the sensitive data detection from a total of 292 components from open source applications.

II / 3. I defined a GPT-supported static source code analysis pipeline that uses the protection level categorisation to identify the protection level of components in a frontend application and then uses the results from the protection level and sensitivity level evaluations to detect vulnerabilities where components are not protected or whose protection level is insufficient for the sensitivity of the data they handle, in line with the CWE-653 type software vulnerability.

Table 1 summarizes the relation between the thesis points and the corresponding publications.

Table 1: *Relation between the thesis points and the corresponding publications*

Publication	Thesis point									
	Credit	IF	SJR	I/1	I/2	I/3	I/3/1	II/1	II/2	II/3
[J1]	0.75		Q3	●	●	●				
[J2]	1		Q3		●	●				
[J3]	0.60		Q4				●			
[J4]	-	3.4	Q1					●	●	●
[C5]	0.48				●					
[C6]	0.48			●						
[C7]	0.50				●					
[C8]	0.60				●					
[F9]	-			●						
[F10]	-			●	●					
[F11]	-				●					
[F12]	-						●			

5 The author's publications on the subjects of the thesis

Journal papers

[J1] **Szabó, Z.**, Bilicki, V. (2021). Evaluation of EHR Access Control in a Heterogenous Test Environment. *Acta Cybernetica*, 25, 485-516. SJR: Q3, **0.75 credits**

[J2] **Szabó, Z.** (2021). Evaluation of a policy enforcement solution in telemedicine with offline use cases. *Pollack Periodica,* 17(1), 12-17. SJR: Q3, **1 credits**

[J3] **Szabó, Z.**, Hompoth, E. A., Bilicki, V. (2023). Patient Flow Analysis with a Custom Simulation Engine. *Acta Cybernetica*, – accepted, under publication SJR: Q4, **0.60 credits**

[J4] **Szabó, Z.**, Bilicki, V. (2023). A new Approach to Web Application Security: Utilizing GPT Language Models for Source Code Inspection. In *Future Internet* MDPI. SJR: Q1, **-**

Full papers in conference proceedings

[C5] **Szabó, Z.**, Bilicki, V., Berta, Á., & Jánki, Z. R. (2017). Smartphone-based data collection with stunner using crowdsourcing: lessons learnt while cleaning the data. *In the Proceedings of ICCGI17* **0,48 credits**

[C6] Jánki, Z. R., **Szabó, Z.**, Bilicki, V., Fidrich, M. (2017, November). Authorization solution for full stack FHIR HAPI access. In *2017 IEEE 30th Neumann Colloquium (NC)* (pp. 000121-000124). IEEE. **0,48 credits**

[C7] **Szabó, Z.**, Téglás, K., Berta, Á., Jelasity, M., & Bilicki, V. (2019). Stunner: A smart phone trace for developing decentralized edge systems. In *Distributed Applications and Interoperable Systems: 19th IFIP WG 6.1 International Conference, DAIS 2019*, Held as Part of the 14th International Federated Conference on Distributed Computing Techniques, DisCoTec 2019, Kongens Lyngby, Denmark, June 17–21, 2019, Proceedings 19 (pp. 108-115). Springer International Publishing. **0,50 credits**

[C8] Berta, Á., **Szabó, Z.**, & Jelasity, M. (2020, December). Modeling Peer-to-Peer Connections over a Smartphone Network. In *Proceedings of the 1st International Workshop on Distributed Infrastructure for Common Good* (pp. 43-48). **0,60 credits**

Further related publications

[F9] **Szabó, Z.**, Bilicki, V. (2018, June). A FHIR-based healthcare system backend with deep cloud side security. In *THE 11TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 184).

[F10] **Szabó, Z.**, Bilicki, V. (2020, June). EHR Data Protection with Filtering of Sensitive Information in Native Cloud Systems. In *THE 12TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 164).

[F11] **Szabó, Z.** Policy Enforcement in Telemedicine with the Deployment of Multiple Enforcement Points. In *The 16th Iványi Miklós International PhD & DLA Symposium,2020.*

[F12] **Szabó, Z.**, Hompoth, E. A., Bilicki, V. (2022, June). Evaluation of a Custom Patient Flow Modeling Framework for Hospital Simulation. In *THE 13TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 202)

Further publications

[13] Nagy, Á., Dombi, J., Fülep, M. P., Rudics, E., Hompoth, E. A., **Szabó, Z.**, ... & Szendi, I. (2023). The Actigraphy-Based Identification of Premorbid Latent Liability of Schizophrenia and Bipolar Disorder. MDPI, *Sensors*, 23(2), 958.

[14] Rudics, E., Nagy, Á., Dombi, J., Hompoth, E. A., **Szabó, Z.**, Horváth, R., ... & Szendi, I. (2023). Photoplethysmograph Based Biofeedback for Stress Reduction under Real-Life Conditions in Healthcare Frontline. MDPI, *Applied Sciences*, 13(2), 835.

Other references

[15] About Modeler — Camunda Platform 8 Docs — docs.camunda.io. `https://docs.camunda.io/docs/components/modeler/about-modeler/`. [Accessed 15-Sep-2022].

[16] Index - fhir v4.0.1. `https://www.hl7.org/fhir/`. (Accessed on 09/16/2020).

[17] Introduction to the angular docs. https://angular.io/docs. Last accessed on 2023-07-20.

[18] Open policy agent official site. `https://www.openpolicyagent.org/`. (Accessed on 09/16/2020).

[19] What Is Patient Flow? — catalyst.nejm.org. `https://catalyst.nejm.org/doi/full/10.1056/CAT.18.0289`. [Accessed 29-Sep-2022].

[20] SpiffWorkflow 1.1.6 documentation — spiffworkflow.readthedocs.io. `https://spiffworkflow.readthedocs.io/en/latest/`, 2014. [Accessed 22-Sep-2022].

[21] Marcus Andrew. Security in fhir at devdays redmond 2019. `https://tinyurl.com/ryk9zlu`, 2019. (Accessed on 07/20/2020).

[22] Sidong Feng and Chunyang Chen. Prompting is all you need: Automated android bug replay with large language models, 2023.

[23] David Ferraiolo, Ramaswamy Chandramouli, Rick Kuhn, and Vincent Hu. Extensible access control markup language (xacml) and next generation access control (ngac). In *Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control*, pages 13–24, 2016.

[24] David Ferraiolo, Janet Cugini, D Richard Kuhn, et al. Role-based access control (rbac): Features and motivations. In *Proceedings of 11th annual computer security application conference*, pages 241–48, 1995.

[25] Md Rakib Hossain Misu and Kazi Sakib. Fantasia: A tool for automatically identifying inconsistency in angularjs mvc applications. 10 2017.

[26] Jose L. Jimenez and Zhe Peng. Covid-19 airborne transmission tool available. `https://cires.colorado.edu/news/covid-19-airborne-transmission-tool-available`, Nov 2020.

[27] Zoltán Richárd Jánki and Vilmos Bilicki. Rule-based architectural design pattern recognition with gpt models. *Electronics*, 12(15):3364, Aug 2023.

[28] Grácián Kokrehel and Vilmos Bilicki. The impact of the software architecture on the developer productivity. *Pollack Periodica*, 17(1):7–11, Mar 2022.

[29] Di Leva and Emilio Sulis. A business process methodology to investigate organization management: A hospital case study. *WSEAS Transactions on Business and Economics*, 14:100–109, 2017.

[30] Bob Martin, Mason Brown, Alan Paller, Dennis Kirby, and S Christey. Cwe. *SANS top*, 25, 2011.

[31] Shima Rahimi Moghaddam and Christopher J. Honey. Boosting theory-of-mind performance in large language models via prompting, 2023.

[32] Csaba Varga, Zsuzsanna Lelovics, Viktor Soós, and Tibor Oláh. Betegforgalmi trendek multidiszciplináris sürgősségi osztályon. *Orvosi Hetilap*, 158(21):811–822, 2017.

[33] E. Yuan and J. Tong. Attributed based access control (abac) for web services. *IEEE International Conference on Web Services (ICWS'05)*, 2005.

6 Összefoglalás

Az értekezés feltárja a modern telemedicina alkalmazások komplex jogosultságkezelési igényei által generált kihívásokat és problémákat, és különböző megoldásokat demonstrál, amelyek teljesítik vagy megkönnyítik az említett kihívások megoldását.

A tudományos eredményeket két csoportra osztva mutattam be, melyek az értekezés harmadik és negyedik fejezetében kerültek részletes bemutatásra.

Az első téziscsoport a telemedicina alkalmazások infrastuktúrájának komplex, heterogén elemekből álló adatútjára koncentrál. Bemutattam benne a követelményeket, amelyeket egy jogosultságkezelési megoldásnak teljesítenie kell annak érdekében, hogy megfelelően és hatékonyan, az ellátási folyamat akadályozása nélkül érvényesíthesse a definiált szabályokat. Felvázoltam a felhőn kívül edge két, jogosultságkezelés szempontjából speciális esetet jelentő típusát, a feldolgozó és tároló edge-t, illetve megvizsgáltam az okostelefon alapú peer-to-peer hálózatokat stabilitás és hatékonyság szempontjából egy alternatív, teljesen elosztott edge típusként. Bemutattam egy tervezett keretrendszert, mely a Policy Enforcement Point (PEP) koncepciójára építve implementál egy, az adatút számos pontján elhelyezhető jogosultságkezelési pontot, mely képes a lekért adatok részleges elemzésére és azok szükségszerű részleges módosítására vagy titkosítására is a hozzáférés biztosítása előtt. A tervezett keretrendszert implementáltam az Open Policy Agent (OPA) segítségével, működését, hatékonyságát pedig kiértékeltem az két speciális edge típust szimuláló tesztkörnyezetekben. További validáció érdekében kifejlesztettem egy nyílt forráskódú szimulációs eszközt, mely alkalmas arra, hogy validációs paramétereket generáljon a kidolgozott jogosultságkezelési megoldás számára.

A második téziscsoportban az adatút végén elhelyezkedő alkalmazások statikus elemzése támasztotta kihívásokat vizsgáltam. Definiáltam két kategorizálást, az egyiket az érzékeny adatok detektálása és rangsorolása, a másikat az alkalmazás komponensek védettségi szintjének megállapítására. A kategóriák használhatóságát validáltam előbb egy 200 szavas változónév-gyűjtemény besorolásával, majd egy nyílt forráskódú Angular projetekből kinyert, összesen 292 komponenst számláló teszthalmazon. Az így azonosított érzékeny adatok alapján megvizsgáltam, mennyire eredményesen azonosíthatóak a statikus forráskód alapján a szoftverek érzékeny elemekkel foglalkozó részei. Hasonlóan validáltam a védettségi szintek taxonómiáját is, szintén a 292 elemű komponenshalmazzal, majd a két vizsgálat eredményének összesítésével elemeztem, mennyire hatékonyan detektálható a CWE-653-as sérülékenységként azonosított, kritikus részek nem megfelelő izolációjaként definiálható sebezhetőség, mely érzékeny adatok kiszivárgását kockáztatja. Az eredmények bemutatása mellett feltártam a problémákat okozó hibákat is, majd azok alapján a detektálási elvek szigorításával elérhető javulást.

Mindkét terület rengeteg további kutatási lehetőséggel és kérdéssel kecsegtet, munkám azonban kiválóan szemlélteti a bennük rejlő lehetőséget, és segítheti a telemedicina alkalmazások fejlesztőit és tervezőit a jogosultságkezelési megoldások tervezésében és implementálásában és validálásában.

7 Nyilatkozat

Szabó Zoltán "Identification of Data Privacy Challenges and Development of Solutions for the Edge Components of the Telemedicine Datapath" című PhD disszertációjában a következő eredményekben **Szabó Zoltán** hozzájárulása volt a meghatározó:

I/1. tézispont [J1] Telemedicina alkalmazások jogosultságkezelésével kapcsolatos elvárások és igények felderítése, szakirodalom feldolgozása, jogosultságkezelési szabálykategóriák meghatározása és formális definiálása.

- **Szabó, Z.**, Bilicki, V. (2021). Evaluation of EHR Access Control in a Heterogenous Test Environment. *Acta Cybernetica*, 25, 485-516. SJR: Q3, **0.75 credits**

[C6] Telemedicina terület jogosultságigényeinek felmérése, ABAC, RBAC alapú megoldások problémáinak felmérése.

- Jánki, Z. R., **Szabó, Z.**, Bilicki, V., Fidrich, M. (2017, November). Authorization solution for full stack FHIR HAPI access. In *2017 IEEE 30th Neumann Colloquium (NC)* (pp. 000121-000124). IEEE.

[F9] Biztonsági modul tervezése kidolgozása és kiértékelések futtatása a vizsgált telemedicina alkalmazásokban a jogosultságkezelési szabályok érvényesítéséhez.

- **Szabó, Z.**, Bilicki, V. (2018, June). A FHIR-based healthcare system backend with deep cloud side security. In *THE 11TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 184).

[F10] Kezdetleges telemedicina jogosultságkezelési igények felmérése, taxonómia kidolgozása és validációs követelmények megtervezése.

- **Szabó, Z.**, Bilicki, V. (2020, June). EHR Data Protection with Filtering of Sensitive Information in Native Cloud Systems. In *THE 12TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 164).

I/2. tézispont [J1] Telemedicina adatút feltárása, teljesítendő igények és követelmények meghatározása.

- **Szabó, Z.**, Bilicki, V. (2021). Evaluation of EHR Access Control in a Heterogenous Test Environment. *Acta Cybernetica*, 25, 485-516. SJR: Q3, **0.75 credits**

[J2] Valós telemedicina alkalmazások adatainak feldolgozása napi adatforgalom meghatározásához. Offline vizsgálati esetek összegyűjtése.

- **Szabó, Z.** (2021). Evaluation of a policy enforcement solution in telemedicine with offline use cases. *Pollack Periodica*, 17(1), 12-17. SJR: Q3, **1 credits**

[C5] Szakirodalom összegyűjtése, elemzése, saját eredményeink elhelyezése, összevetése a state-of-the-arttal, az adatgyűjtő alkalmazás működésének elemzése és az anomáliákat okozó mérési komponensek detektálása, majd eltávolítása, az adatgyűjtő alkalmazással kapcsolatban közölt változtatások implementációja.

- **Szabó, Z.**, Bilicki, V., Berta, Á., & Jánki, Z. R. (2017). Smartphone-based data collection with stunner using crowdsourcing: lessons learnt while cleaning the data.*In the Proceedings of ICCGI17*

[C7] Szakirodalom összegyűjtése, elemzése, saját eredményeink elhelyezése, összevetése a state-of-the-arttal, valamint assziszténcia a P2P keresési - és kapcsolódási folyamat során felmerült hibák, kihívások és problémaforrások azonosításában és elemzésében.

- **Szabó, Z.**, Téglás, K., Berta, Á., Jelasity, M., & Bilicki, V. (2019). Stunner: A smart phone trace for developing decentralized edge systems. In *Distributed Applications and Interoperable Systems: 19th IFIP WG 6.1 International Conference, DAIS 2019*, Held as Part of the 14th International Federated Conference on Distributed Computing Techniques, DisCoTec 2019, Kongens Lyngby, Denmark, June 17–21, 2019, Proceedings 19 (pp. 108-115). Springer International Publishing

[C8] Szakirodalom összegyűjtése, elemzése; adatgyűjtő alkalmazás és adatbázis működésének felügyelete, felmerülő hibák elhárítása.

- Berta, Á., **Szabó, Z.**, & Jelasity, M. (2020, December). Modeling Peer-to-Peer Connections over a Smartphone Network. In *Proceedings of the 1st International Workshop on Distributed Infrastructure for Common Good* (pp. 43-48).

[F10] Open Policy Agent (OPA) kiértékelése implementációs lehetőségként. Kísérleti környezet megvalósítása, és kezdetleges szabály-kiértékelések az OPA segítségével lokális, izolált környezetben.

- **Szabó, Z.**, Bilicki, V. (2020, June). EHR Data Protection with Filtering of Sensitive Information in Native Cloud Systems. In *THE 12TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 164).

[F11] Offline felhasználási esetek felmérése és OPA keretrendszer implementáció adaptálása offline esetekhez.

- **Szabó, Z.** Policy Enforcement in Telemedicine with the Deployment of Multiple Enforcement Points. In *The 16th Iványi Miklós International PhD & DLA Symposium,2020.*

I/3. tézispont [J1] Kísérleti jogosultságkezelési szabályok kidolgozása, implementálása, validációs környezet felállítása. Mintaadatok generálása, tesztesetek megtervezése. Tesztek futtatása, kiértékelése és konklúziók levonása.

- **Szabó, Z.**, Bilicki, V. (2021). Evaluation of EHR Access Control in a Heterogenous Test Environment. *Acta Cybernetica*, 25, 485-516.

[J2] Validációhoz használt PWA alkalmazások fejlesztése, WebAssembly integrációval. Kísérletek futtatása és eredmények elemzése.

- **Szabó, Z.** (2021). Evaluation of a policy enforcement solution in telemedicine with offline use cases. *Pollack Periodica*, 17(1), 12-17.

I/3.1. tézispont [J3] SpiffWorkflow szimulátor bővítése és hibajavítása; szimuláció szcenáriók implementálása; szimulációk kiértékelésében assziszténcia.

- **Szabó, Z.**, Hompoth, E. A., Bilicki, V. (2023). Patient Flow Analysis with a Custom Simulation Engine. *Acta Cybernetica*, – accepted, under publication SJR: Q4, **0.60 credits**

[F12] Szimulációs lehetőségek kiértékelése. Saját szimulációhoz szükséges eszközök kidolgozása és eszköz megtervezése. Kezdetleges szimulációk összeállítása, futtatása és asszisztencia az eredmények kiértékelésében.

- **Szabó, Z.**, Hompoth, E. A., Bilicki, V. (2022, June). Evaluation of a Custom Patient Flow Modeling Framework for Hospital Simulation. In *THE 13TH CONFERENCE OF PHD STUDENTS IN COMPUTER SCIENCE* (p. 202)

II/1. **tézispont** [J4] Szakirodalom feldolgozása, GPT-4 API használatának és a prompt engineering domain felderítése. Taxonómia definiálása az érzékenységi szint és a védettségi szint számszerűsítésére.

- **Szabó, Z.**, Bilicki, V. (2023). A new Approach to Web Application Security: Utilizing GPT Language Models for Source Code Inspection. In *Future Internet* MDPI. SJR: Q1, ~~**0.75 credits**~~

II/2. **tézispont** [J4] Elemző pipeline tervezése és implementálása. Prompt kidolgozása érzékeny adattagok tagelésére, és azokon keresztül az ezekkel foglalkozó Servicek detektálására. Változónév-szótár kidolgozása. Kiértékeléshez használt projektek kiválogatása és kiértékelése. Elemzés futtatása és eredmények manuális validálása.

- **Szabó, Z.**, Bilicki, V. (2023). A new Approach to Web Application Security: Utilizing GPT Language Models for Source Code Inspection. In *Future Internet* MDPI. SJR: Q1, ~~**0.75 credits**~~

II/3. **tézispont** [J4] JSON formátum kidolgozása Angular komponensek statikus elemzéséhez. Prompt kidolgozása a JSON mappelése automatizálására GPT API által. Köztes feldolgozóműveletek implementálása. Védettségi szint meghatározását végző promptok kidolgozása. Kísérletek futtatása és elemzése. Összesített eredmények elemzése. Szigorított detektáló szabályok definiálása és az alapján kapott eredmények elemzése, összevetése az eredeti csoporttal.

- **Szabó, Z.**, Bilicki, V. (2023). A new Approach to Web Application Security: Utilizing GPT Language Models for Source Code Inspection. In *Future Internet* MDPI. SJR: Q1, ~~**0.75 credits**~~.

Ezen eredmények Szabó Zoltán PhD disszertációján túl más tudományos fokozat megszerzésére nem használhatóak fel.

Kelt: 2023. augusztus 18., Szeged

Szabó Zoltán .... [illegible]

Jelölt aláírása

Témavezető alárírása

Az Informatika Doktori Iskola vezetője kijelenti, hogy jelen nyilatkozatot minden társszerzőhöz eljuttatta, és azzal szemben egyetlen társszerző sem emelt kifogást.

Szeged, 2023. augusztus 28.

Dátum, kelt

SZEGEDI TUDOMÁNYEGYETEM INFORMATIKA DOKTORI ISKOLA SZEGED

[illegible]

DI vezető aláírása