Adatbiztonság kihívásainak feltárása és megoldások kidolgozása a telemedicina adatút edge komponensein

PhD Disszertáció tézisei

Szabó Zoltán
Témavezető: Dr. Bilicki Vilmos

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

Szeged
2023

1 Bevezetés

Az egészségügy és az informatika integrációja folyamatosan erősödik, ám ez a gyorsuló folyamat újabb és újabb kihívásokat és akadályokat támaszt, amelyek jelentős mértékben megnehezítik az olyan technológiai megoldások kidolgozását, melyek javíthatják a páciensellátás minőségét és megkönnyíthetik az egészségügyi dolgozók munkáját.

Disszertációmban bemutatom kutatási eredményeimet, melyeket a telemedicina alkalmazások biztonsága és jogosultságkezelése terén végzett munkám során értem el. Munkámat két részre osztottam, melyek közül az első a felhő és edge computing elemeit kombináló heterogén telemedicina adatútra fókuszál; különösen az edge szegmens két, speciális esetére, a feldolgozó és tároló edge-re, melyek jogosultságkezelés szempontjából kimagaslóan komplex kihívást jelentenek, mivel a felhőtől függetlenül kell érvényesíteniük a hozzáférési szabályokat, hasonló hatékonysággal és minimális késleltetéssel. Formálisan definiáltam egy taxonómiát, melynek révén négy jogosultságkezelési típust határoztam meg az adatút igényeinek lefedésére; hasonlóan tettem a különböző edge típusokkal, illetve egy kutatócsoport tagjaként megvizsgáltam az okostelefon alapú peer-to-peer hálózatok stabilitását és potenciálját, melyek az edge egy speciális esetét képviselik; kidolgoztam egy keretrendszert a Policy Enforcement Point (PEP) koncepciójából kiindulva, mely képes mind a feldolgozó, mind a tároló edge-en megvalósítani a definiált jogosultságkezelési kategóriákat; ehhez a keretrendszer kísérleti implementációkat hoztam létre, melyeken teszteltem a módszertan hatékonyságát; végül kidolgoztam egy nyílt forráskódú szimulációs eszközt, mely képes a betegfolyam modellezésére, és ezáltal validációs paraméterek generálására, melyekkel ellenőrizhető a jogosultságkezelési módszertanom hatékonysága és alkalmazhatósága.

Munkám második felében az adatút végén elhelyezkedő frontend alkalmazásokat vizsgáltam. Analitikai szempontból ezek különösen komplex területet képviselnek, ugyanis a frontend keretrendszerek felépítése, illetve az ilyen webalkalmazások dinamikus, nehezen követhető navigációi jelentős mértékben megnehezítették a forráskód statikus elemzésére tett próbálkozásokat. Emiatt kidolgoztam egy saját módszertant, mely a nagy nyelvi modellek (LLM) képességeire építkezve elemzi és azonosítja az alkalmazások forráskódját, azon belül is azokat a szegmenseket, melyek érzékeny, biztonságkritikus adatokkal dolgoznak, és ahol fennáll a veszély ezek kiszivárgására. Ehhez létrehoztam egy formális kategorizálást az adat érzékenységi szintjeinek megkülönböztetésére, mely három szintből áll az alapján, hogy egy potenciális adatszivárgás mekkora károkat okozna, illetve egy másik kategorizálását, mely az alkalmazások komponenseinek védettségi szintjét kvantifikálja. Az érzékenységi taxonómiám validitását a GPT-4 és GPT-3.5 API-k segítségével végeztem, előbb egy változónevekből összeállított szótár, majd véletlenszerűen kiválogatott, nyílt forráskódú webalkalmazások forráskódjai segítségével. Ezt követi a védettségi taxonómia validálása, majd a két kiértékelés eredményeinek összevonásával a potenciál érzékeny adat-szivárgási pontok detektálása.

2 I. téziscsoport: Jogosultságkezelés a telemedicina adatút peremén

Az alábbi téziscsoportban megvizsgáltam telemedicina adatút követelményeit jogosultságkezelés tekintetében. Meghatároztam azokat a feltételeket, amelyeket az adatút minden pontján teljesíteni kell a biztonsági előírások és a válaszidő összeegyeztetése érdekében. Definiáltam egy négyelemű taxonómiát a szükséges jogosultságkezelési kategóriák számára. Bemutattam az edge-t, mint a jogosultságkezelés kritikus eseteit, meghatároztam két különleges típusát, a feldolgozó edge-t és a tároló edge-t, valamint bemutattam eredményeinket egy speciális edge esetről, az okostelefon alapú peer-to-peer hálózatok stabilitásának elemzéséről. Ezután két kísérleti környezetet hoztam létre, egyet a feldolgozó edge és egyet a tároló edge modellezésére, valamint teszt szabályokat az említett környezetekben a jogosultságkezelési keretrendszer gyakorlati megvalósításának validációjához késleltetés szempontjából. Végül bemutattam a nyílt forráskódú betegfolyam szimulátoromat, amely szimulálja a kórházi osztályokon kezelt betegek áramlását, és felhasználható a kifejlesztett jogosultságkezelési keretrendszer validálási paramétereinek generálására.

A téziscsoporthoz tartozó publikációk: [J1],[J2],[J3],[C5],[C6],[C7],[C8],[F9],[F10], [F11], [F12]

2.1 I/1. tézis: A telemedicina jogosultságkezelés formális definíciói és követelményei

Megvizsgáltam a modern telemedicina alkalmazások komplex követelményeit jogosultságkezelés tekintetében. Definiáltam egy taxonómiát a különböző típusú jogosultságkezelési szabályok és az általam meghatározott TAPE-követelmények formalizálására, amelyek szükségesek annak biztosításához, hogy az implementált jogosultságkezelési megoldás garantálja az infrastruktúra bármely pontján a megfelelő adatvédelem és sebesség közötti egyensúlyt.

A tézishez tartozó publikációk: [J1],[C6],[F9],[F10]

Az egészségügyben a dokumentumok és adatok egyre jelentősebb része válik digitálissá. A trend azonban sajátos kihívásokkal jár, melyek közül a legjelentősebb, hogy ezeknek az elektronikus egészségügyi rekordoknak (EHR) könnyen hozzáférhetőnek, kereshetőnek és értelmezhetőnek kell lenniük, mivel a páciensek szabadon mozoghatnak az egészségügyi ökoszisztémában. Emiatt elvárás a strukturáltság és a szabványosítás, hogy ezáltal elérhető célkitűzés lehessen a klinikai együttműködés, illetve az adatok automatizált feldolgozása.

Az átjárhatóság érdekében az évek során számtalan fejlesztő - illetve kutatócsoport próbálkozott szabványok kidolgozásával, ezek közül a legnépszerűbb és legelterjedtebb a HL7 szabványügyi szervezet által kidolgozott Fast Healthcare Interoperability Resources (FHIR) [16] lett, köszönhetően magas fokú testreszabhatóságának.

A szabvány hivatalos dokumentációja azonban egy tekintetben hiányos - csak ajánlásokat tartalmaz arra vonatkozóan [21], hogyan lehet az adatstruktúrákat kibővíteni annak érdekében, hogy az integrálható legyen olyan klasszikus jogosultságkezelési metodológiákkal, mint a szerepkör alapú RBAC [24], illetve az attribútum alapú ABAC [33]. Az, hogy

ezeket pontosan hogyan kellene alkalmazni, implementálni egy komplex telemedicina rendszerben, hogyan használhatóak kompetenciák, felelősségek definiálására az egészségügyi összetett jogosultságkezelési igényeinek kielégítésére, nincsenek egyértelműen definiálva, mely az FHIR (és konkurensei) talán legnagyobb hátulütője.

A területen végzett kutatómunkám egyik kiindulópontja annak feltérképézése volt, hogy lehetséges-e olyan megoldás kidolgozása, mely a komplex, heterogén adatút bármely pontján alkalmazható lehet, ugyanazokkal az eszközökkel, definíciókkal és konfigurációkkal. Az alapvető jogosultságkezelési művelettípusokat, amelyeket kutatásaim alapján egy ilyen megoldásnak támogatnia kell, az alábbi módon definiáltam a TAPE-követelmények formájában:

- **Transparency (Áttetszőség)**: A jogosultságkezelési megoldásnak olyan minimális, alig kimutatható hatással szabad csak lennie az adatút teljesítményére és áteresztő képességére, amennyire az csak lehetséges.

- **Adaptability (Adaptálhatóság)**: Az érzékeny adatok megfelelő kezeléséhez a telemedicina alkalmazásoknak rugalmas, aprólékosan állítható szabályokra van szüksége. Az ABAC és RBAC önmagukban nem elegendőek ennek ellátásra, mivel a rendszerek közötti átjárhatóság igénye jóval rugalmasabb megoldásokat igényel. A megoldásnak támogatnia kell a legspeciálisabb és legaprólékosabb jogosultságkezelési igényeket is.

- **Portability (Hordozhatóság)**: A megoldásnak az infrastruktúrán belül bárhol alkalmazhatónak kell lennie. Az edge computing legnagyobb erőssége abban rejlik, hogy akkor is képes garantálni a funkcionalitást, amikor a felhő maga elérhetetlen. Ez magával vonja, hogy a jogosultságkezelési megoldásnak alkalmazhatónak kell lennie a felhő és az edge, az edge és végpontok között, a felhőben, és bizonyos esetekben akár magukon a végpontokon is, feltéve ha azok rendelkeznek a jogosultságkezelési feladatok ellátásához szükséges erőforrásokkal.

- **Efficient (Hatékonyság)**: Mivel az adatút mentén több olyan csomópont is előfordulhat, melyek nem rendelkeznek elegendő memóriával és számítási kapacitással az összetettebb transzformációk és elemzések elvégzéséhez, a megoldásnak kímélnie kell őket az ilyen műveletektől és csak a legszükségesebb műveleteket végezheti el ezeken.

A telemedicina jogosultságkezelési elveket az alábbi taxonómia és követelmények szerint definuiáltam. A páciens a dokumentumai elsődleges tulajdonosa, a kezelőorvosa, aki létrehozta, vagy legalábbis asszisztált a létrejöttükben, a másodlagos, más egészségügyi dolgozók, rokonok, családtagok pedig alapértelmezésben nem, vagy csak részlegesen férhetnek hozzá az adathoz. A rendszernek támogatnia kell a közvetett jogosultságot, vagyis olyan eseteket, amikor egy egyén nem önként, hanem valamilyen csoport részeként kap részleges vagy teljes jogosultságot. A jogosultságkezelési szabálykategóriákat a felhasználó saját szerepköre, a csoport szerepköre és a dokumentum elemeinek érzékenysége alapján kell meghatározni.

Bizonyos esetekben szükség lehet kontextuális információkra is a hozzáférés megállapításához. Ezen felül az egészségügy egyik kulcsfontosságú igénye az, hogy a jogosultságkezelés sosem foglalhatja magába alapértelmezésben a teljes dokumentumhoz való hozzáférést. Számos esetben ez magába foglal olyan információkat is, mely lehetővé tenné egy

harmadik félnek, hogy rekonstruálja az érzékeny adatokat is, melyekkel jogosulatlan információkhoz férhet hozzá, vagy akár veszélyeztetheti a pácienst. Az utolsó igény a legspecifikusabb és legnehezebb aspektusa az egészségügyi biztonságnak: a Break-the-Glass olyan eseteket fed le, amelyek során azonnali hozzáférést kell adni a kritikus páciensadatokhoz életmentő beavatkozás céljából. Ekkor érthető módon sem a pácienstől, sem a kezelőorvosától nem várható el, hogy engedélyezzék a rendszeren belül a hozzáférést. Ugyan ezekben az esetekben csak néhány specifikus rekordra és dokumentumra van szükség, ahhoz, hogy a jogosultságkezelési elvek ne sérüljenek, azokat mindenképp intenzív és gyors transzformációknak kell alávetni, hogy csak a legszükségesebb elemeik válhassanak elérhető az adott szituációban.

Ezek alapján az alábbi négy kategóriát határoztam meg a jogosultságkezelési szabályok számára:

- **Role Evaluation**: A szabálynak azt kell eldöntenie, hogy a felhasználó szerepe vagy szerepei alapján kaphet-e részleges vagy teljes hozzáférést az adott dokumentumhoz;
- **Contextual Evaluation**:A szabálynak azt kell megállapítania, hogy a felhasználó szerepkörei, továbbá különböző attribútumok és kontextuális információk alapján kaphat-e részleges vagy teljes hozzáférést;
- **Contextual Modification**: A jogosultság biztosításán felül a szabálynak átalakításokat kell végeznie a dokumentumon, eltávolítani vagy megváltoztatni bizonyos mezők tartalmát;
- **Break-the-Glass**: Az egészségügyi alkalmazások különleges igénye, vészhelyzet esetén a szabályank azonnali hozzáférést kell biztosítania, de közben titkosítania vagy el kell távolítania több mezőt a dokumentumból.

2.2 I/2. tézis: A jogosultságkezelés formális definíciói és kihívásai a telemedicina felhőn túl

Speciális kategóriákat definiáltam a felhőn kívüli edge típusainak megkülönböztetésére, amelyek jogosultságkezelés szempontjából a modern telemedicina infrastruktúra különleges kategóriáját képviselik. Formálisan definiáltam ezeknek a kategóriákat feldolgozó és tároló edgeként. Ezután megvizsgáltam az egyre elterjedtebb okostelefonokon alapuló peer-to-peer hálózatokat, mint a szélsőséges megoldások eseteit, és elemeztem azok potenciálját és stabilitását, hogy önálló edge hálózatokként működhessenek az adatúton.

A tézishez tartozó publikációk: [J1],[J2],[C5],[C7],[C8],[F10],[F11]

Modern hálózati topológiákban, melyek magukba foglalják az IoT-t, okoseszközöket, edge computingot, a nyilvános és privát felhők integrációját, mint például a 1. ábrán látható modellben, a hagyoányos jogosultságkezelési metodológiák nem elégségesek. Ahhoz, hogy az I/1. tézisben kimondott követelmények teljesülhessenk, egy hibrid stratégia kidolgozása szükséges, mely kombinálja az ABAC és RBAC módszertanok erősségeit. Az adat és a feldolgozás érzékeny jellege miatt elengedhetetlen, hogy a jogosultságkezelést megvalósító csomópontok az infrastruktúra bármely pontján elhelyezhetőek legyenek.

Ez a hordozhatóság magába foglalja azokat az eseteket is, amikor a felhőt elérhetetlennek vagy legalábbis korlátozottan elérhetőnek kell tekintenünk. Nem véletlenül terjed a fog, illetve edge computing, amely a felhő és az eszközök közötti adatforgalmat úgy próbálja csökkenteni, hogy az adatút végpontjain különböző, önellátó mikrohálózatokat hoz létre, amelyek a felelősségek megosztásával és a funkciók aggregálásával, ahol csak tudják, felszabadítják a felhő kapacitását, és csak a kritikus, feltétlenül szükséges műveletek esetén kommunikálnak vele. Míg azonban az adatgyűjtés, tárolás és gyorsítótárazás már viszonylag megoldott és lefedett témák az edge computingon belül, a jogosultságkezelési műveletek megvalósítása lényegesen nagyobb kihívást jelent.

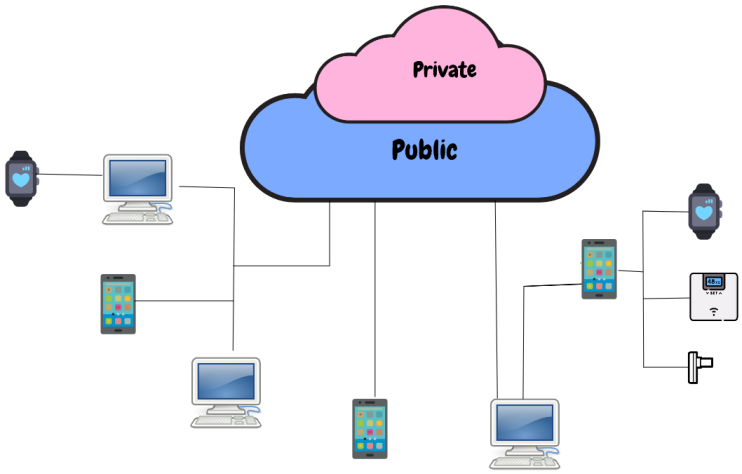


Figure 1: *Felhő és edge integrált telemedicina infrastruktúra vázlata*

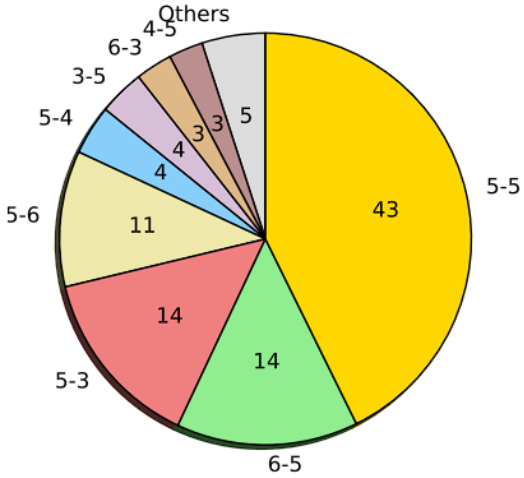
Az ilyen végpontokon az adatok elérhetősége és a hozzáférés sebessége nem sérülhet, de a jogosultságkezelési szabályok szigorú betartatása sem, amelyeket minden más esetben követni kell a telemedicinális adatok esetében. Az edge-en figyelembe kell vennünk az azon elhelyezkedő csomópontok csökkentett erőforrásait, amelyek megnehezítik bizonyos funkciók megvalósíthatóságát.

Kapacitásaik alapján két fő típust különböztetünk meg az edgen belül, amelyeket a disszertációban formálisan is definiáltam. Ezek a következők:

- **feldolgozó edge**: a legerősebb, legnagyobb kapacitású elemek az edgen, amelyek elegendő tárolási és feldolgozási kapacitással rendelkeznek akár 1000 dokumentum feldolgozásához, és képesek mind a négy meghatározott jogosultságkezelési kategória érvényesítésére.

- **tároló edge**: elemek, amelyek kapacitása kisebb, mint a feldolgozóké, de még mindig képesek nagyobb, több százas nagyságrendű dokumentummennyiségek feldolgozására, és képesek legalább a legkritikusabb jogosultságkezelési szabályok, a Role Evaluation és Contextual Evaluation kezelésére.

Az edgen belül az okoseszközök közötti peer-to-peer kapcsolatok különleges esetet jelentenek, mivel sok különböző típusú eszköz keveredik bennük egy dinamikusan változó környezetben. Pontos kapacitásuk feltérképezése érdekében, különös tekintettel a stabilitásra, kutatócsoportunk részt vett egy nagyszabású adatgyűjtési és elemzési kampányban,



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: *Peer-to-peer kapcsolatok száma különböző NAT-típusok mellett*

amelynek során egy saját fejlesztésű, Stunner nevű alkalmazást fejlesztettünk ki, amelyet a kampányban részt vevő felhasználók a telefonjaikra telepítettek. Hozzájárulásukkal különböző kísérleteket végeztünk peer-to-peer kapcsolatok létrehozására, amelyek eredményeit a résztvevő felhasználók engedélyével gyűjtöttük, természetesen személyes adataikat eltávolítva. A kulcsfontosságú mérések olyan elemeket tartalmaztak, mint a készülék hálózati kapcsolatának típusa a mérés időpontjában, a NAT típusa, a létrehozott peer-to-peer kapcsolat sikeressége, és ha sikeres volt, annak hossza. A kampány során összegyűjtött adathalmaz végül elérhetővé vált, létrehozva ezzel az egyik legnagyobb ilyen jellegű adatbázist. A peer-to-peer elemzés egyik eredménye a 2 ábrán látható.

2.3 I/3. tézis: Jogosultságkezelési implementációk a feldolgozó és tároló edge-en

Bemutattam a javasolt jogosultságkezelési megoldásom gyakorlati implementációját, majd tesztkörnyezeteket állítottam fel az edge típusok számára a négy kategóriához tartozó tesztszabályokkal, és ezekben a környezetekben mértem a végrehajtás hatékonyságát növekvő adatmennyiség mellett. A mérések során megvizsgáltam az értékelést végző csomópontok erőforrásigényeit, valamint az adatútra mért késleltetéseket. Az altézisben bemutatom a mért késleltetéseket, amelyek megerősítették, hogy a különböző edge típusokban észszerű men-

nyiségű adat esetében az általam felvázolt megoldás megfelel az előző tézisekben meghatározott követelményeknek.

A tézishez tartozó publikációk: [J1],[J2]

A végrehajtási pont egy népszerű koncepciója, mely feltevésem szerint képes kellett, hogy legyen követelmények teljesítésére, a Policy Enforcement Point (PEP), amelyet az OASIS szabványügyi szervezet az eXtensible Access Control Markup Language szabvány (XACML) [23] részeként fejlesztett ki, és amely a klasszikus ABAC modell kiterjesztése, más néven policy-based access control vagy PBAC. Bár a tervem része egy olyan egyéni, kifinomultabb biztonsági megoldás kifejlesztése volt, amelyre nem vonatkoznak az XACML szabvány szigorú korlátai, a PEP-alapú architektúra koncepciója megfelelt az igényeinknek.

A jogosultságok érvényesítésére vonatkozó koncepciónk teszteléséhez kutatócsoportunk egy ígéretes új megoldást választott, az Open Policy Agent (OPA) [18] nevű, Golang és WebAssembly nyelven elérhető megoldást, ami tökéletesen alkalmassá tette arra, hogy az adatút több pontján - a felhőben, a különböző edge-típusokban vagy akár magukban a webalkalmazásokban - is elhelyezhető legyen. Az OPA azt is lehetővé teszi, hogy a döntéshozatalhoz szükséges információkat JSON formátumban tároljuk, és a különböző irányelveket saját szkriptnyelvén, a Rego-ban definiáljuk, amelyhez az élesítést követően egy jól definiált REST-interfészen keresztül lehet hozzáférni HTTP POST-kéréssel, amely tartalmazza a szűrendő vagy értékelendő kontextuális dokumentumokat (esetünkben az egészségügyi rekordokat).

Ami a validációt illeti, kétféle tesztkörnyezetet hoztam létre: az egyik egy kisebb infrastruktúrát szimulált a feldolgozó edge szimulálásához, míg a másik egy felhővel való kapcsolat nélküli tároló edget szimulált, amelynek a helyben tárolt adatokhoz való hozzáférési igényeket kellett értékelnie. Az ezekre az edgetípusokra vonatkozó feltételezéseim alapján az önálló OPA-változattal szimulált feldolgozó edge esetében úgy véltem, hogy mind a négy értékelési kategória elfogadható késleltetést fog okozni még az ezres nagyságrendű dokumentumméretek esetén is, míg a WebAssembly alapú tároló edge képességei lényegesen korlátozottabbak lesznek, de ugyanakkor képes lesz a Role Evaluation és a Contextual Evaluation kategóriákba tartozó irányelvek hatékony kiértékelésére. A kísérleteket a Szegedi Tudományegyetem Szoftverfejlesztési Tanszékének Inclouded fejlesztőcsapata telemedicina alkalmazásai alapján modellezett FHIR Observation dokumentumokon végeztem. Először 10, majd 20, 50, 100, 200 stb. kérdeztem, egészen 2000-ig, minden egyes lekérdezést többször végeztem el, az értékelésekhez pedig az átlagolt eredményeket használtam.

A 3. ábra a legösszetettebbnek tekintett Break-the-Glass kategória kiértékelésével járó késleltetését mutatja a dokumentumlekérés teljes futási idejéhez viszonyítva a teszteléshez használt infrastruktúrán belül. Jól látható, hogy még 2000 dokumentum esetében is az értékelés átlagosan kevesebb mint 1000 ms, ami a teljes dokumentumlekérés futási idejének kevesebb mint 40%-a.

Ami a WebAssembly futási idejét illeti, a vizsgálat részét képezte annak felmérése, hogy pontosan milyen eszközök lehetnek alkalmasak arra, hogy betöltsék a tároló edge szerepét. E célból a WebAssembly OPA implementációját Contextual Evaluation és Role Evaluation szabályokkal konfiguráltam, és különböző eszközökön futtattam egy webalkalmazásba ágyazva, amely a hozzáférés engedélyezése előtt helyi adatlekérdezést és hozzáférés-kiér-

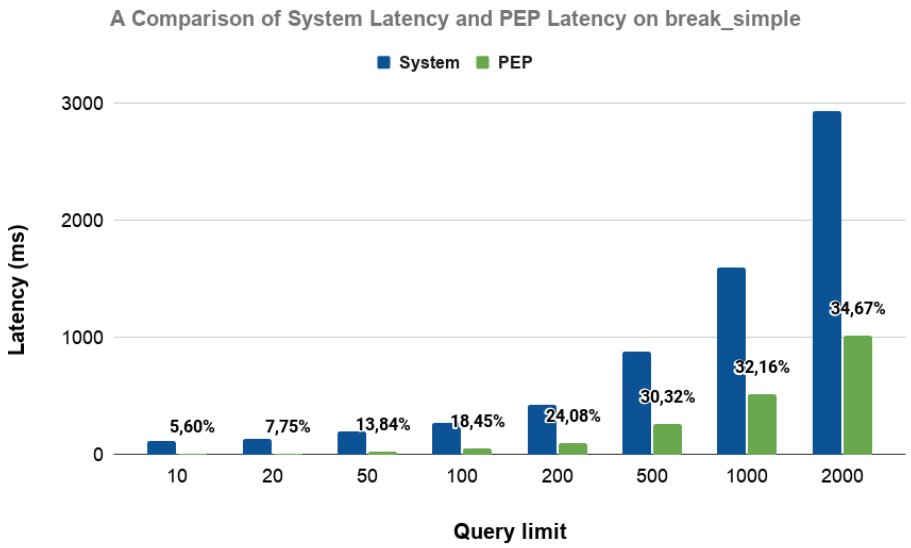


Figure 3: *A rendszer szintű átfutási idő és a jogosultságkezelés által okozott átfutási idő összehasonlítása Break-the-Glass kiértékelés során.*

tékelést végzett. A késleltetések itt már jelentősen nagyobbak voltak az eszközök korlátozott erőforrásai miatt, és ahogy az a 4. ábrán látható összehasonlításból kiderül, míg az asztali PC Chrome böngészőjében futó verzió még nagyobb mennyiségű dokumentum esetén is viszonylag elfogadható eredményeket produkált, addig az okostelefonos és táblagépes verzió már 200 dokumentum felett drasztikusan elkezdte növelni a késleltetést. Ez azt jelenti, hogy 100-200 dokumentumig bármelyik eszköz betöltheti a tároló edge szerepét, de az eredmények alapján elképzelhető, hogy ez az edge típus alkalmatlan nagyobb dokumentumhalmazok hatékony kiértékelésére.

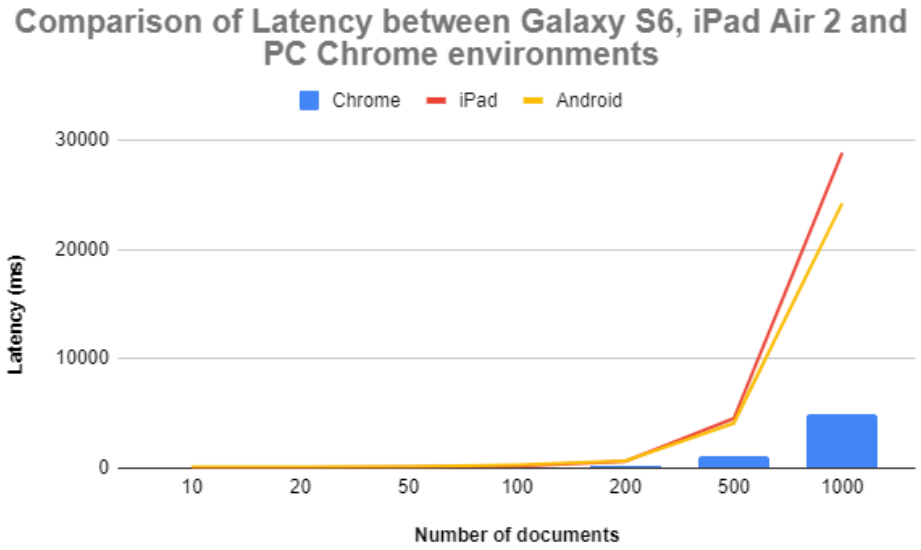


Figure 4: *WebAssembly késleltetések összehasonlítása*

Az eredmények megerősítették, hogy reális adatmennyiség esetén a feldolgozó edge jogosultságkezelése miatti extra késleltetés többnyire 500 ms alatt volt, ami átlagosan a teljes folyamat végrehajtási idejének kevesebb mint 25%-át tette ki. A tároló edge esetében, bár a megvalósítható hozzáférés-szabályozási műveletek száma korlátozott volt, és a végrehajtási idők magasabbak voltak, néhány száz dokumentum mellett átlagosan 1 másodperc alatt maradtak, függetlenül a végrehajtási környezet kapacitásától.

2.4 I/3.1. tézis: Nyílt forráskódú betegfolyam szimulációs eszköz az eredmények validálásához

Kifejlesztettem egy nyílt forráskódú szimulációs eszközt, amely nyílt forráskódú könyvtárakat és eszközöket használ a betegek eloszlásának és várakozási idejüknek - a betegfolyamnak - a modellezésére a kórházi osztályokon. A kifejlesztett szimulációs eszköz felhasználható annak validálására, hogy az előző altézisben mért megnövekedett átfutási idő milyen mértékben lassíthatja a folyamatot a kezelt adatok mennyisége alapján, és ez milyen hatással lehet a telemedicina adatútra, a betegfolyamra és a várakozási időkre az ellátási folyamat kulcsfontosságú pontjain.

A tézishez tartozó publikációk: [J3],[F12]

A betegfolyam [19] azt az időablakot jelenti, amelyet egy beteg az egészségügyi ellátórendszerben tölt az érkezéstől a távozásig. Az alapvető cél az, hogy ezt az időt a lehető legkevesebbre csökkentsük az ellátás minősége érdekében, mely elv az utóbbi években kiemelt szerepet kapott a járványügyi intézkedések miatt, amelyek korlátozták az egy helyen egyidejűleg tartózkodó betegek számát, a járvány elleni védekezés érdekében pedig sokan tettek javaslatokat és próbáltak létrehozni validációs eszközöket [26] arra vonatkozóan, hogy mennyi lehet a maximális várakozási idő, amit egy beteg egy helyen tölthet, hogy a fertőzés esélye minimálisra csökkenjen.

Bár a piacon számos olyan eszköz áll rendelkezésre, amely alkalmas az ilyen szimulációk felépítésére és futtatására, ezek képességei többnyire korlátozottak, és használatuk jelentős tanulási időt és energiabefektetést igényel, ami pont azok dolgát nehezíti meg a legjobban, akik a legtöbb információval rendelkeznek a témában, és képesek lennének a legpontosabb szimulációs modelleket létrehozni. Kutatócsoportunk célja egy olyan szimulációs eszköz létrehozása volt, amely nyílt forráskódú elemekre építve olyan modelleket futtathat, amelyeket csekély programozói tudással rendelkező kutatók vagy akár orvosok is definiálhatnak egy egyszerű vizuális leíró nyelv segítségével. Az eszköz a korábbi tézisek eredményeinek validálásához is kulcsfontosságú, mivel használata lehetőséget adhat annak ellenőrzésére is, hogy a jogosultságkezelés miatt megnövekedett futási idők hogyan befolyásolhatják a várakozási időt és a betegek torlódását, különböző scenáriók esetén.

Az eszköz futtatásához a SpiffWorkflow [20] nevű nyílt forráskódú Python könyvtárat használtuk, amely alkalmas üzleti folyamatmodellek definiálására és futtatására, valamint a Camunda-t [15], az egyik legnépszerűbb üzleti folyamatmodellező eszközt a tényleges modellezésére. A kifejlesztett szoftver különböző fő összetevői a 5. ábrán láthatók.

Az elkészült szimulációs eszköz értékeléséhez egy olasz kutatócsoport [29] modelljét készítettük el újra, amely egy sürgősségi osztályt (ED) ábrázolt, és amelynek felépítése

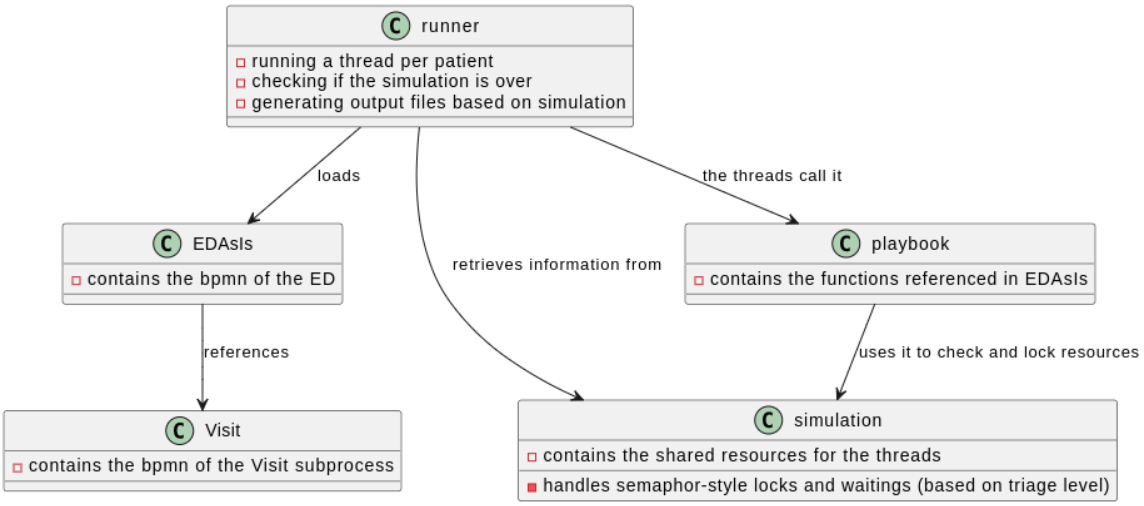


Figure 5: *A betegfolyam szimulátor moduljainak felépítése*

minimális változtatással adaptálható volt magyar kórházak hasonló osztályaira. A szimuláció paraméterezéséhez a Somogy Vármegyek Kaposi Mór Oktató Kórház [32] betegforgalmi adatait használtuk fel különböző szimulációkon keresztül. Az eredmények alapján eszközünk alkalmasnak bizonyult a további használatra összetettebb kiértékelésekben és a validációs paraméterek előállítására.

3 II. téziscsoport: Adatszivárgási lehetőségek detektálása progresszív web alkalmazásokban nagy nyelvi modellek segítségével

A disszertáció második téziscsoportjában a telemedicina adatút egyik kulcsfontosságú elemét elemeztem, az adatút végén fellelhető felhasználói alkalmazásokat. Olyan taxonómiát definiáltam, amely az adatok szivárgásának hatása alapján határozza meg és rangsorolja az alkalmazásokban az érzékeny adatokat, illetve egy másikat, amely meghatározza a alkalmazások komponenseinek védelmi szintjét, oly módon, hogy a meghatározott kategóriák prompt engineering technikák segítségével átadhatóak legyenek LLM-eknek is. Ezután bemutattam eredményeimet előbb egy változónevekből álló szótár elemeinek osztályozásával, majd a nyílt forráskódú frontend alkalmazások érzékeny adatainak detektálásával a GPT-3.5 és GPT-4 API-k segítségével, amellyel megerősítettem azt a hipotézist, hogy az LLM-ek összetett ismerete révén a jelenlegi GPT modellek szintjén a klasszikus értelemben vett tanítás bizonyos esetekben már kiváltható a szükséges tudás prompt alapú átadásával is. Végül összeállítottam egy metodológiát, mellyel elemeztem a válogatott nyílt forráskódú alkalmazásokat az érzékeny adatok, majd a védettségi szintek detektálásával, a két eredmény kombinációjával pedig a potenciális adatszivárgási pontok azonosításával.

A téziscsoporthoz tartozó publikációk: [J4]

3.1 II/1. tézis: Az érzékenység és védettség formális taxonómiája

Definiáltam a frontend alkalmazásokban található érzékeny adatok taxonómiáját, amely három különböző kategóriába sorolja őket a kiszivárgásuk és illetéktelen hozzáférésük által okozott kár mértéke alapján. Ezt követően bemutattam egy további kategorizálást, amely az alkalmazáskomponensek védelmi szintjeit úgy osztja szintekre, hogy azok reprezentálják az érzékenységi kategóriák alapján szükséges védelmet.

A tézishez tartozó publikációk: [J4]

A nagy nyelvi modellek (LLM) elterjedésének és az olyan alkalmazásokban való széles körű használatuknak köszönhetően, mint a ChatGPT, az elmúlt évben jelentősen megnőtt az érdeklődés a mesterséges intelligencia iránt. A programozás, és különösen a statikus programkód értelmezése, elemzése és dokumentálása az egyik legígéretesebb kutatási területté nőtte ki magát a GPT-modellek tanítása során felhasznált nagymennyiségű adat és forrás, valamint az LLM-ek kontextuális értelmezési és elemzési képességei miatt. Ezen képességek miatt feltételeztem, hogy az olyan modellek, mint a GPT-3.5 és a GPT-4 segítségével olyan statikus kódelemzési kihívások is megoldhatók lesznek, amelyek eddig túl bonyolultnak bizonyultak a hagyományos módszerek számára. Az egyik ilyen terület a progresszív webalkalmazások statikus kódelemzése az adatút végén, melyek komplex modularitásuk és navigációjuk miatt nehezen elemezhetőek, eddig főként egyedi fejlesztésű keretrendszerekkel és eszközökkel nyílt erre lehetőség, [25], illetve a lintelés és kiterjedt tesztelés révén.

Ezek a megoldások azonban elsősorban a kód szemantikájára és minőségére összpontosítottak, kizárva olyan lehetséges problémák és sebezhetőségek felfedezését, amelyek az alkalmazások működésének és az általuk kezelt adatok tartalmának mélyebb megértését igénylik. Egy frontend-alkalmazásban idetartozó sebezhetőség például az érzékeny vagy kritikus elemek nem megfelelő izolálása vagy védelme - a Common Weakness Enumeration adatbázis [30] CWE-653: Improper Isolation or Compartmentalization sebezhetőségének változata -, ami azt jelenti, hogy az alkalmazás nyilvános vagy nem védett interfészei képesek és jogosultak hozzáférni érzékeny, biztonságkritikus adatokhoz és műveletekhez. Ha ez szerveroldali hibákkal, szoftverhibákkal vagy a felhasználó rosszindulatú tevékenységével párosul, különböző mértékű adatszivárgáshoz vezethet (az adatbázisban a CWE-200: Exposure of Sensitive Information to an Unauthorized Actor - Érzékeny információk expozíciója illetéktelen felhasználók számára - családdal azonosítva). Az ilyen típusú sebezhetőségek hatékony észleléséhez az első fontos lépés az alkalmazás által kezelt, érzékenynek tekinthető adatok felderítése.

Ennek érdekében mind az adatok érzékenységére, mind a szoftverkomponensek védelmi szintjére vonatkozóan olyan formális kategorizálást definiáltam, amely kompatibilis a prompt engineering technikákkal [22], mint például a ”few shot examples” és a ”chain-of-thought”, hogy ne csak logikus, hanem kompatibilis és használható legyen az LLM-ek értelmezési képességeivel. Az érzékenységi szintek kategorizálásának alapja az adatok kiszivárgása által okozott kár mértéke és az aggregálandó mennyiség volt, amely a felhasználókra vonatkozó személyes, pénzügyi, egészségügyi vagy más hasonló információkhoz való hozzáféréshez szükséges.

Az érzékenységi szintek az alábbiak:

- **Level 1 (Alacsony érzékenység)**: Ezen a szinten nagy mennyiségű adat felhalmozására és összeállítására van szükség a bizalmas információk kikövetkeztetéséhez és a visszaélésre alkalmas lehetőségek megteremtéséhez. Ide tartoznak többek között a felhasználó viselkedési előzményei, a meglátogatott webhelyek listája, a webhelyen található kedvelt termékek és érdeklődési körök, valamint a keresési előzmények.

- **Level 2 (Közepes érzékenység)**: Az ilyen szintű adatok megszerzésével viszonylag egyszerűen lehetséges potenciálisan kritikus információk lekérdezése és összeállítása. Az olyan megoldások, mint a kétfaktoros hitelesítés, a rendszeres e-mail és SMS értesítések, valamint a kimerítő naplózás az érintett alkalmazásokban enyhíthetik egy esetleges behatolás hatásait. Ez a legnagyobb és legkiterjedtebb csoport. Olyan adatokat tartoznak ide, mint például a felhasználónevek, jelszavak, magánjellegű feljegyzések, politikai nézetek, szexuális irányultság, IP-cím, fizikai hely, és időpontfoglalások.

- **Level 3 (Magas érzékenység)**: Az ilyen szintű adatok már önmagukban is érzékenyek, megszerzésük vagy nyilvánosságra hozataluk súlyos jogi következményekkel járhat, és jelentős károkat okozhat. Ilyenek például az egészségügyi adatok, az orvosi kórtörténet, a társadalombiztosítási szám, a jogosítvány és a bankkártyaadatok.

A védettségi skála pedig az alábbi szintekből tevődik össze:

- **0. védettségi szint**: A komponens nem rendelkezik semmiféle védettséggel.

- **1. védettségi szint**: Mezei autentikáció, az alkalmazás csak azt ellenőrzi, hogy a felhasználó bejelentkezett-e az alkalmazásba.

- **2. védettségi szint**: RBAC [24]: A bejelentkezésen felül a felhasználó különböző szerepkörökkel rendelkezik, melyek meghatározzák az alkalmazáson belüli jogosultságait.

- **3. védettségi szint**: ABAC [33]: A bejelentkezésen és opcionális szerepkörök felül egyéb attribútumok is, mint például a fizikai lokáció, az idő, a platform is szerepet játszanak a hozzáférés engedélyezésében.

3.2 II/2. tézis: Webalkalmazások adatainak érzékenységi elemzése

Kidolgoztam egy GPT API alapú módszertant, amely az előző tézisben meghatározott taxonómiát használja a frontend alkalmazásokban található érzékeny adatok felismerésére, és az ilyen adatokon végzett műveletekre összpontosító kódelemek megjelölésére. A kategorizálást egy kiértékeléssel validáltam, amely egy 200 szavas változónév-gyűjtemény elemeit a meghatározott kategóriák egyikébe sorolja. Ezt követően lefuttattam ezt a módszert nyílt forráskódú alkalmazásokból származó, összesen 292 komponensre, hogy azonosítsam bennük az érzékeny adatokat, ezáltal pedig az azok kezelésével foglalkozó serviceket.

A tézishez tartozó publikációk: [J4]

A definiált kategóriák, illetve a GPT-3.5 és GPT-4 API-k elemzési képességeinek validálása érdekében - amelyek a cikk megírásának időpontjában a leghatékonyabb GPT-modellek,

és API hívásokon keresztül elérhetőek voltak - összeállítottunk egy 200 változónévből álló szótárat, amely egyenlően oszlott meg a nem érzékeny, alacsony érzékenységű, közepes érzékenységű és magas érzékenységű kategóriák között. A szótár elkészítésekor szándékosan megnehezítettük a modellek számára a teljes pontosság elérését: néhány változónév szándékos helyesírási hibát tartalmazott, mások idegennyelvűek voltak, például magyar, német vagy olasz, és mások egy érzékenyebb szótő és egy kevésbé érzékeny végződés kombinációját tartalmazták.

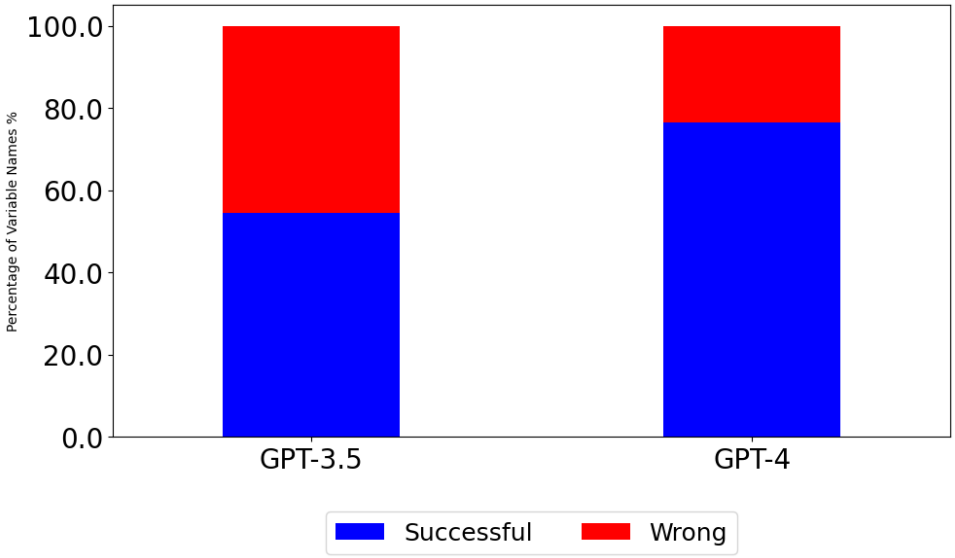


Figure 6: *GPT-3.5 és GPT-4 eredményeinek összehasonlítása a változónevek kiértékelésén*

A validálás eredményei a 6. ábrán láthatók. A GPT-3.5 API a szótár 45,5%-át helytelenül osztályozta, a tévesztett esetek többsége pedig triviális volt. Ezzel szemben a GPT-4 csak az esetek 23,5%-ában hibázott, és a hibák mélyebb vizsgálata során kiderült, hogy csak a kifejezetten nehéz elemeknél hibázott, a hibás esetek több mint felében pedig helyesen azonosította a hibákat érzékeny adatként, csak a pontos kategóriát nem találta el. Bár vizsgálataink során mindkét modellt használtuk, az előzetes eredmények már ezen a ponton azt sugallták, hogy a GPT-4 pontosabb eredményeket fog produkálni, míg a GPT-3.5 a legegyszerűbb esetek kivételével minden hajlamosabb lesz a kudarcra.

Mivel kutatócsoportunknak [27, 28] már jelentős tapasztalata volt az Angular keretrendszerrel [17], ezért a következő kiértékelésekhez is azt választottuk alapként. Metodológiám azon az elképzelésen alapul, hogy az Angular webalkalmazásokban az érzékeny adatok a Service osztályokban összepontosulnak, amelyeket az Angular keretrendszer más osztályai metódushívásokon keresztül, szingletonként érnek el és használnak. Az érzékenységi szintek és sebezhetőségek statikus kód alapján történő felderítéséhez először a GPT-k kontextuskezelő képességeinek segítségével kellett felderítenünk a Component osztályokban érzékenynek tűnő elemeket, és azonosítanunk kellett, hogy ezek közül melyeket kezelik Services-ek.

Kutatócsoportunk a GitHubról több ezer nyilvános Angular projektet gyűjtött össze egy crawler algoritmus segítségével, hogy különböző forráskód-elemzéseket végezhessünk. A projektek TypeScript forrásfájljait címkézéssel, a szóköz karakterek eltávolításával és a sortörések eltávolításával minifikáltuk, majd véletlenszerűen kiválasztottuk a 12 legnagy-

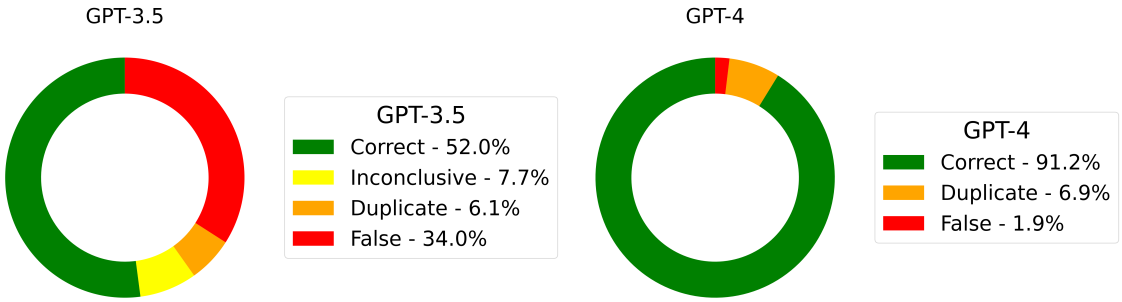


Figure 7: *A modellek pontosságának összehasonlítása érzékeny adatok detektálásában*

obb, legösszetettebb és legnagyobb projektet, amelyek így összesen 292 elemzendő komponenst eredményeztek a további vizsgálatokhoz.

Az első lépés az volt, hogy a 292 komponenst végigiteráljuk, és azonosítsuk az érzékeny adatokat tartalmazó változókat és objektumokat. A 7. ábrán látható eredményekből egyértelmű, hogy a GPT-3.5 a GPT-4-hez képest lényegesen több attribútumot jelölt meg tévesen érzékenynek. A tévesztések közé azok az esetek tartoznak, amikor az érzékelés hibás volt; esetleg egy függvényt vagy egy nem fontos változót jelöltek érzékeny adatnak egy változó vagy objektum helyett; a duplikáltak közé azok az esetek tartoznak, amikor ugyanazt az érzékeny adatot egy fájlban többször is jelölték; és a hiányos adatok közé azok az esetek, amikor az érzékeny adatot helyesen észlelték, de az érzékenység indoklása hiányos vagy helytelen volt.

Figure 8: *A sikeresen, illetve sikertelenül detektált érzékeny Service-k aránya a két modell által*

Ezután ezen attribútumok alapján azonosítottuk a Serviceket, amelyek érzékeny adatok forrásaként vagy célpontjaként szolgáltak. Az eredmények a 8. ábrán láthatók, és bár a GPT-4 itt egyértelműen hatékonyabb volt, jelentős számú hibát vétett az érzékeny

szolgáltatások következetes felismerésében. Az eredmények kiértékelése után kiderült, hogy ez a fejlesztők néhány különösen érdekes rossz gyakorlatának volt köszönhető, amelyek közé tartozott a több felelősséget kezelő monolitikus Servicek használata, illetve az Angular injektálási elvének különböző megkerülései.

3.3 II/3. tézis: Védettségi szintek és sebezhetőség detektálása webalkalmazásokban

Felvázoltam egy GPT alapú statikus forráskód-elemző pipelinet, amely a védelmi szintek taxonomiáját használja a frontend-alkalmazás összetevőinek védelmi szintjének azonosítására, majd a védelmi szint és az érzékenységi szint kiértékelésének eredményeit felhasználja a sebezhetőségek, potenciális adatszivárgások felderítésére, ahol a komponensek nem védettek, vagy amelyek védelmi szintje nem elegendő az általuk kezelt adatok érzékenységéhez képest, összhangban a CWE-653 típusú szoftver sebezhetőséggel.

A tézishez tartozó publikációk: [J4]

Az elemző folyamat promptjait a prompt engineering [31], egy viszonylag új tudományág elveinek felhasználásával, több ciklusos kísérletezéssel fejlesztettém ki; a felhasznált promptok a disszertáció függelékeiben találhatók. A kritériumok közé tartozott a világos megfogalmazás, a kezdeti tesztelés során felfedezett anomáliák újbóli megjelenését tiltó szabályok beillesztése, valamint az elvárt válaszformátumhoz tartozó példák mellékelése. Ennek érdekében nagyobb, bővebb kéréseket használtunk, amelyek lehetővé tették mind a formai hibák elkerülését, mind pedig az olyan prompt engineering technikák bevonását, mint a ”few shot example”, ahol minden kérés legalább egy minta bemenetet és egy megfelelő minta kimenetet tartalmazott, valamint a chain-of-thought, ahol az egyszerű szabályok mellett a kérésekhez társított minta gondolat- és logikai folyamatot is megadtuk, ami segített a helyes levezetésben és a különböző hibák elkerülésében.

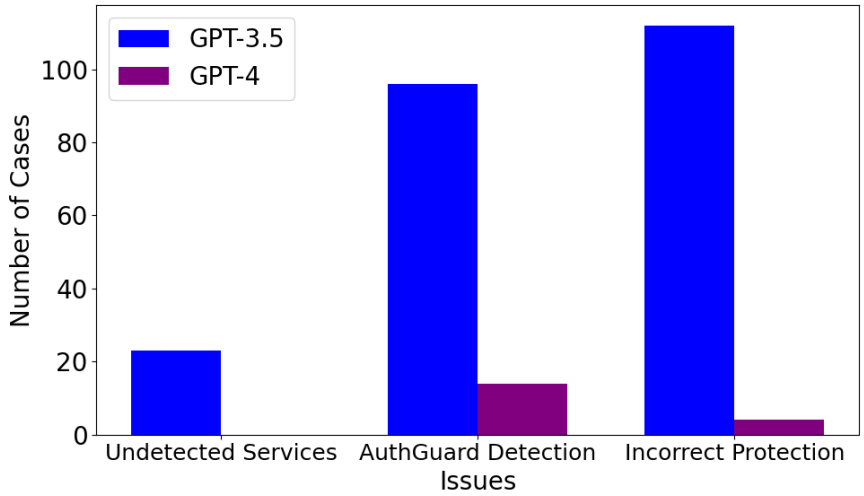


Figure 9: *A komponensek elemzése során vétett hibák összevetése modellenként*

A teljes elemző folyamat az alábbi lépésekből áll:

1. **Kódbázis Minifikálása**: Az Angular alkalmazások .ts forrásfájljait tömörítettük, eltávolítva a sortöréseket és a szóközöket.

2. **Érzékeny Elemek Detektálása**: A tömörített komponensfájlokat egyenként átadtuk a GPT API-nak. Az elemzés először azonosította az érzékeny adatokat, a válaszban kifejtve, hogyan jelennek meg az alkalmazásban, hogyan kerülnek felhasználásra, majd egy aggregációs szkript ezen információk alapján kiválasztotta azokat a Serviceket, amelyek a projektben érzékeny adatokon végzett olvasási vagy írási műveletekben vettek részt. A megjelölt Servicek minden egyes előfordulása esetében az általuk kezelt adatok érzékenységi szintjét is meghatározták.

3. **Projektfájlok JSON Mappelése**: A tömörített komponensfájlokat egyenként átadtuk a GPT API-nak. E lépés eredményeképpen minden projekthez egy JSON fájl jött létre, amelyben az alkalmazás komponensei JSON objektumokká redukálódnak, és csak az aktuális (villetve a tervezett jövőbeli) vizsgálatokhoz szükséges információkat tartalmazzák, például a szülő-gyermek komponensek közötti kapcsolatokat, az injektált Serviceket, valamint az azokból használt adattagokat és műveleteket.

4. **Védettségi Szint Megállapítása**: Az előző lépésből származó JSON-öket, valamint a tömörített Router-konfigurációkat és AuthGuardokat egyenként átadtuk a GPT API-nak, amely hozzáadta a skálánknak megfelelő védelmi szintet. Ezt követően egy Python-szkript egyesítette ezeket a védelmi szinteket, és a gyermek komponensekhez a szülő komponensek védelmi szintjét rendelte.

5. **Sebezhetőségek Detektálása**: A Védettségi Szint Megállapítása és az Érzékeny Elemek Detektálása lépések eredményeinek egy xlsx fájlban történő összesítésével kilistázzuk a tesztelt projektek sebezhetőségeit, azokat a pontokat, amikor egy érzékeny szolgáltatást használó komponens nem rendelkezik elég magas védettségi szinttel.

A 9. ábra a Védettségi Szint Megállapítása lépés eredményeit mutatja be, a GPT-4 esetében a hibás AuthGuard-érzékelési problémák többségét a szülő- és gyermekmodulok alkalmazáson belüli egymásba ágyazása okozta, ami megnehezítette az AuthGuardok pontos érzékelését. A vizsgálatok során azonban az esetek többségében a maximális AuthGuard-szintet helyesen észlelték, így a 292 vizsgált esetből mindössze négy esetben volt nem megfelelő védettségi szint. A GPT-3.5 eredményeinek kiértékelésekor azt tapasztaltuk, hogy a prompt engineering technikák alkalmazása és a modellek hőmérsékletének 0-ra állítása ellenére nehezen megmagyarázható problémákkal találkoztunk, beleértve nemlétező AuthGuardok hallucinálását és az ugyanazon AuthGuardhoz tartozó ellentmondásos védelmi szinteket.

Amint az a 10. ábrán látható, a GPT-3.5-re vonatkozó feltételezéseink beigazolódtak, és az előző lépések gyenge eredményeit felhalmozva összesen csak a sebezhetőségek 1%-át sikerült felismernie. Az érzékeny szolgáltatások azonosítására vonatkozó kezdeti naiv feltételezésünk azonban nagyon súlyos következményekkel járt a GPT-4 esetében is.

A probléma jelentős részét a monolitikus Servicek okozták, ahol a Servicek nem egy jól meghatározott feladatot és szerepet láttak el, hanem a fejlesztők több hasonló eljárást zsúfoltak beléjük anélkül, hogy figyelembe vették volna, hogy a nem érzékeny és a magas érzékenységű adatkategóriák között minden szint megjelenhetett így egyetlen Servicen

Figure 10: *A két modell és a GPT-4 szigorított szabályok melletti pontossága a potenciális szivárgási helyek felderítésében*

belül. Ez a probléma azt eredményezte, hogy a Servicek érzékenységi szintje sok esetben nagyon eltérő volt, amikor egy Servicet egyetlen komponensben közepes vagy magas érzékenységűnek minősítettek, míg egy másikban ugyanazt az érzékeny adatok szempontjából irrelevánsnak nyilvánították.

A detektálás eredmények javítása és e problémák megoldása érdekében a GPT-4-re a szabályok jelentős szigorítása került alkalmazásra. Az eredmények igen látványos javulása a GPT-4 Strict eredményeiként láthatóak.

4 A disszertáció eredményei

Az **első téziscsoportban** formálisan definiáltam a jogosultságkezelési szabálykategóriákat; felvázoltam egy kategorizálást az edge típusai számára; feltártam és elemeztem az okostelefonok képességeit egy stabil peer-to-peer hálózat létrehozására; és értékeltem a jogosultságkezelési megoldásom teljesítményét az edge típusokat szimuláló környezetekben; a részletesebb validációhoz pedig létrehoztam egy nyílt forráskódú szimulációs eszközt. A részletes kifejtés a disszertáció 3. fejezetében található.

I / 1. Megvizsgáltam a modern telemedicina alkalmazások komplex követelményeit jogosultságkezelés tekintetében. Definiáltam egy taxonómiát a különböző típusú jogosultságkezelési szabályok és az általam meghatározott TAPE-követelmények formalizálására, amelyek szükségesek annak biztosításához, hogy az implementált jogosultságkezelési megoldás garantálja az infrastruktúra bármely pontján a megfelelő adatvédelem és sebesség közötti egyensúlyt.

I / 2. Speciális kategóriákat definiáltam a felhőn kívüli edge típusainak megkülönböztetésére, amelyek jogosultságkezelés szempontjából a modern telemedicina infrastruktúra különleges kategóriáját képviselik. Formálisan definiáltam ezeknek a kategóriákat feldolgozó és tároló edge-ként. Ezután megvizsgáltam az egyre elterjedtebb okostelefonokon alapuló peer-to-peer hálózatokat, mint a szélsőséges megoldások eseteit, és elemeztem azok potenciálját és stabilitását, hogy önálló edge hálózatokként működhessenek az adatúton.

I / 3. Bemutattam a javasolt jogosultságkezelési megoldásom gyakorlati implementációját, majd tesztkörnyezeteket állítottam fel az edge típusok számára a négy kategóriához tartozó tesztszabályokkal, és ezekben a környezetekben mértem a végrehajtás hatékonyságát növekvő adatmennyiség mellett. A mérések során megvizsgáltam az értékelést végző csomópontok erőforrásigényeit, valamint az adatútra mért késleltetéseket. Az altézisben bemutatom a mért késleltetéseket, amelyek megerősítették, hogy a különböző edge típusokban észszerű mennyiségű adat esetében az általam felvázolt megoldás megfelel az előző tézisekben meghatározott követelményeknek.

I / 3.1. Kifejlesztettem egy nyílt forráskódú szimulációs eszközt, amely nyílt forráskódú könyvtárakat és eszközöket használ a betegek eloszlásának és várakozási idejüknek - a betegfolyamnak - a modellezésére a kórházi osztályokon. A kifejlesztett szimulációs eszköz felhasználható annak validálására, hogy az előző altézisben mért megnövekedett átfutási idő milyen mértékben lassíthatja a folyamatot a kezelt adatok mennyisége alapján, és ez milyen hatással lehet a telemedicina adatútra, a betegfolyamra és a várakozási időkre az ellátási folyamat kulcsfontosságú pontjain.

A **második téziscsoportban** a népszerű GPT-3.5 és GPT-4 modellek értelmezési és elemzési képességeit vizsgáltam összetett frontend alkalmazások statikus forráskódelemzése során. Taxonómiákat definiáltam az adatok érzékenysége és a komponensek védettsége kategorizálására az esetleges adatszivárgás esetén bekövetkező kár alapján; a kategóriákat a GPT API-k segítségével validáltam előbb egy változónevekből álló szótár, majd egy adag nyílt forráskódú Angular projekt segítségével; végül az eredmények alapján bemutattam a lehetséges adatszivárgási helyek detektálásának hatékonyságát. A részletes kifejtés a disszertáció 4. fejezetében található.

II / 1. Definiáltam a frontend alkalmazásokban található érzékeny adatok taxonómiáját, amely három különböző kategóriába sorolja őket a kiszivárgásuk és illetéktelen hozzáférésük által okozott kár mértéke alapján. Ezt követően bemutattam egy további kategorizálást, amely az alkalmazáskomponensek védelmi szintjeit úgy osztja szintekre, hogy azok reprezentálják az érzékenységi kategóriák alapján szükséges védelmet.

II / 2. Kidolgoztam egy GPT API alapú módszertant, amely az előző tézisben meghatározott taxonómiát használja a frontend alkalmazásokban található érzékeny adatok felismerésére, és az ilyen adatokon végzett műveletekre összpontosító kódelemek megjelölésére. A kategorizálást egy kiértékeléssel validáltam, amely egy 200 szavas változónév-gyűjtemény elemeit a meghatározott kategóriák egyikébe sorolja. Ezt

követően lefuttattam ezt a módszert nyílt forráskódú alkalmazásokból származó, összesen 292 komponensre, hogy azonosítsam bennük az érzékeny adatokat, ezáltal pedig az azok kezelésével foglalkozó serviceket.

II / 3. Felvázoltam egy GPT alapú statikus forráskód-elemző pipelinet, amely a védelmi szintek taxonomiáját használja a frontend-alkalmazás összetevőinek védelmi szintjének azonosítására, majd a védelmi szint és az érzékenységi szint kiértékelésének eredményeit felhasználja a sebezhetőségek, potenciális adatszivárgások felderítésére, ahol a komponensek nem védettek, vagy amelyek védelmi szintje nem elegendő az általuk kezelt adatok érzékenységéhez képest, összhangban a CWE-653 típusú szoftver sebezhetőséggel.

Az 1. táblázat összegzi a tézispontok és a publikációm közötti kapcsolatokat.

Table 1: *A tézispontok és publikációk kapcsolata*

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 A szerzői publikációs listája

Publikációk folyóiratokban

[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, **-**

Konferenciacikkek

[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**

Egyéb releváns publikációk

[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)

A disszertációban felhasználatlan publikációk

[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.

Irodalomjegyzék

[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 Summary

The dissertation explores the challenges and problems generated by the complex access control requirements of modern telemedicine applications and demonstrates different solutions that meet or facilitate the resolution of these challenges.

The scientific results are presented in two groups, which are detailed in the third and fourth chapters of the dissertation.

The first group of theses focuses on the complex data path of heterogeneous elements of the telemedicine cloud-integrated application infrastructure. In it, I have presented the requirements that a policy enforcement solution must fulfil in order to enforce the defined rules in a reliable and efficient way, without hampering the care process. I have described two types of edge beyond the cloud that represent a special case for privilege management, the processing and caching edge, and examined smartphone-based peer-to-peer networks as an alternative, fully distributed edge type from the standpoint of stability and efficiency. I have presented a proposed framework that builds on the concept of a Policy Enforcement Point (PEP) to implement an authorization policy enforcement point that can be placed at multiple points on the data path, capable of partial analysis of the retrieved data and also partial modification or encryption of the data as necessary before granting access. I implemented the proposed framework using Open Policy Agent (OPA) and evaluated its operation and efficiency in test environments simulating two specific edge types. Furthermore, I have developed and evaluated an open source simulation tool which can be used to generate validational constraints for the solution.

In the second set of theses, I investigated the challenges posed by the static analysis of applications at the end of the data path. I defined two taxonomies, one for detecting and ranking sensitive data, and the other for determining the level of protection of application components. I validated the usability of the categories first by classifying a 200-word collection of variant names, and then on a test set of 292 components extracted from open source Angular projects. Based on the sensitive data identified, I examined the effectiveness of identifying the sensitive parts of the software from the static source code. Similarly, I validated the taxonomy of vulnerability levels, also using the 292-element component set, and then analyzed the effectiveness of detecting vulnerabilities defined as inadequate isolation of critical parts identified as CWE-653 vulnerabilities that risk leakage of sensitive data by combining the results of the two tests. In addition to presenting the results, I have also identified the flaws that cause the problems and, based on these, the improvements that can be achieved by tightening the detection principles.

Both areas hold a wealth of further research opportunities and questions, but my work provides an excellent illustration of their potential and can help developers and designers of telemedicine applications to design and implement and validate privilege management solutions.

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 asszisztencia 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 asszisztencia.

- 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 ....

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

DI vezető aláírása