Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Majkut Kristóf, Soós Sarolta

DIFFERENTIAL PRIVACY-ALAPÚ ANONIMIZÁLÁS K-ANONIMITÁSSAL

KONZULENS

Dr. Dudás Ákos

BUDAPEST, 2019

Tartalomjegyzék

Összefoglaló

Az Európai Unió által 2018. tavaszán bevezetett Általános Adatvédelmi Rendelet nagy terhet ró minden olyan személyre és szervezetre, akik valamilyen módon adatokkal dolgoznak. A GDPR szigorúan szabályozza a különböző adatok kezelését, a meg nem felelés pedig súlyos szankciókkal járhat. A rendelet azonban nem ad pontos iránymutatást arra vonatkozóan, hogy miként kell az említett szabályokat betartani. Erre kínálunk mi egy olyan lehetséges megoldást, melynek alapja az irodalomban is jól ismert k-anonimitás. Mivel mindent az adatrögzítés helyén anonimizálunk, mind az adatkezelő, mind pedig az adatfeldolgozó mentesül a GDPR-előírások alól. Megmutatjuk, hogy minimális változtatásokkal a javasolt algoritmus kiterjeszthető úgy, hogy megfeleljen a Differential Privacy modell által támasztott követelményeknek is.

Abstract

The General Data Protection Regulation of the European Union that entered into force in the spring of 2018 entails a heavy burden on any person or organisation working with data in some way. GDPR strictly controls handling different kinds of data and not meeting the standards might imply serious sanctions. The regulations however do not provide exact guidance on how to observe the aforesaid rules. We propose a possible solution to this, which is based on k-anonymity, well-known from literature. Since we anonymize everything on the data-recording spot, both the data-handler and the data-processor are exempt from the GDPR obligations. We show that with minimum modifications the proposed algorithm can be extended so that it also meets the demands placed by the Differential Privacy model.

1 Bevezetés

A nemzetközi jogban a második világháború után találkozhattunk először az általános, illetve a személyes adatok védelmének fogalmával. A számítógépek és az internet széleskörű elterjedésére volt szükség ahhoz, hogy ezek újra középpontba kerüljenek. Az elsősorban célzott hirdetések kiszolgálására alkalmas ún. *tracking* funkciók, valamint a közösségi hálózatok megszületése tette lehetővé a vállalatok számára azt, hogy egyre több információt gyűjtsenek és tároljanak rólunk, amelyek felett megfelelő törvénykezés hiányában az emberek semmiféle kontrollal nem rendelkeztek. Ugyan az Európai Parlament a ’90-es évek közepén elfogadott egy irányelvet, azt minden tagállam a maga legjobb belátása szerint vezette be a törvénykezésébe, mely által a problémáknak csak egy igen csekély hányadát sikerült kiküszöbölni.

Különböző statisztikai vagy éppen kutatás-fejlesztési célokból, illetve az egyre népszerűbb gépi tanulásnak köszönhetően folyamatosan nőtt az igény a kórházak, pénzintézetek és egyéb szervezetek által gyűjtött adatok publikálására. A személyiségi jogok védelmének érdekében ugyan megkísérelték az egyes rekordokhoz tartozó egyének beazonosíthatóságának megelőzését, mivel azonban az anonimizálás ekkor még igencsak gyerekcipőben járt, számos esetben derült ki, hogy a nyilvánosságra hozott adathalmazokból könnyedén kitudódhat, hogy melyik rekord pontosan kit rejt.

Az egyre több forrásból táplálkozó adatgyűjtés és a publikált adathalmazokból származó folyamatos adatszivárgás mellett az adatlopások is mindinkább elszaporodtak. A kis cégektől kezdve egészen a legnagyobb vállalatokig történtek betörések az informatikai rendszerekbe, melyek hatalmas mennyiségű ügyfélinformáció letöltésével is jártak.

A minél inkább elharapódzó helyzet nyilvánvalóvá tette, hogy beavatkozásra van szükség. Az Európai Unió válasza végül 2016-ban érkezett meg az Általános Adatvédelmi Rendelettel, azaz a GDPR-ral.

1.1 A rendelet

Ugyan a jogalkotási eljárás már 2012-ben megindult, csak négy évvel később hirdették ki és lépett hatályba a GDPR. Az igen szigorú szabályok miatt a vállalatoknak kétéves türelmi időszakot biztosítottak, hogy megfelelően fel tudjanak készülni és az

előírt követelményeknek eleget tudjanak tenni. A rendeletet 2018 nyara óta kell kötelezően, teljes egészében és közvetlenül alkalmazni az Európai Gazdasági Térség minden tagállamában.

Az előírás a személyes adatokon végzett bármely művelettel (adatkezeléssel) kapcsolatosan három kulcsszereplőt említ: az adatkezelőt, az adatfeldolgozót, illetve az érintettet. Adatkezelőnek számít bármely természetes vagy jogi személy, aki meghatározza, hogy mely személyes adatokat, milyen céllal és milyen módon kívánja gyűjteni. Az adatfeldolgozó ezek alapján hajtja végre az előírt műveleteket. Következésképpen az adatkezelőnek szigorúbb előírásoknak kell megfelelnie, hiszen ő felelős a teljes, személyes adatokat tartalmazó adathalmazért. Az adatfeldolgozón valamivel kisebb a teher, számára az a legfontosabb, hogy kizárólag olyan műveleteket hajtson végre, amelyeket az adatkezelő meghatározott. Az érintett pedig nem más, mint maga az adatkezelés alanya, akiről a személyes információkat gyűjtik.

A következő példa segítségével helyezzük megfelelő kontextusba a fentieket: Tételezzük fel, hogy egy bank felbérel egy IT szolgáltatásokat kínáló céget az adatai archiválására. Ebben a szcenárióban a pénzügyi intézet lesz az adatkezelő, hiszen ő határozza meg az adatkezelés pontos célját. Ezt követően az IT cég a követelmények alapján elvégzi a szükséges műveleteket, így válik ő adatfeldolgozóvá. Az érintettek - más néven adatalanyok - pedig a bank ügyfelei.

Fontos tisztázni a rendelet szövegében felbukkanó adatfogalmakat is. Bizalmas adatnak számít minden olyan információ, amelynek kereskedelmi értéke van, azaz amelynek publikálása, megváltoztatása vagy elvesztése negatív hatással lehet az adat tulajdonosának piaci versenyhelyzetére. Ilyenek a szerződések, üzleti folyamatok vagy például egy forráskód. A bizalmas adatok egy részhalmazát alkotják a személyes adatok. Ide tartozik egy azonosított vagy azonosítható, természetes személyre vonatkozó bármilyen információ, így a név, születési dátum vagy irányítószám is. Az érzékeny információk pedig a faji vagy etnikai származásra, politikai véleményre, vallási vagy világnézeti meggyőződésre, szexuális irányultságra utaló személyes adatok, valamint genetikai, biometrikus és egészségügyi azonosítók. Az érzékeny adatok feldolgozására különösen szigorú előírások vonatkoznak.

A GDPR érvényes az összes EU-központú adatkezelőre és adatfeldolgozóra, illetve minden olyan, az Unión kívülire is, amely az EU-ban tartózkodó emberek személyes adatait gyűjti vagy dolgozza fel. A rendelet célkitűzése tehát a személyes

adatok védelmének biztosítása az Európai Unió minden polgára számára, illetve azon jogaik kiterjesztése, hogy rendelkezhessenek az adataikról. Ennek megfelelően a GDPR többek között minden érintett számára biztosítja a jogot, hogy tájékoztatást kérjen az adatkezelésről, hogy bármikor hozzáférjen a róla tárolt adatokhoz, ezeket bármikor módosíthassa, valamint törlésüket kérhesse. Az irányelvek között megjelenik például az adatminimalitás és az elszámoltathatóság is. Az előbbi szerint az adatkezelők a lehető legkevesebb, ténylegesen csak az adatkezeléshez szükséges információt tárolhatják; az utóbbi pedig kifejezi és megerősíti az adatkezelő központi szerepét a teljes adatkezelési folyamat jogszerű lebonyolításában, illetve a vonatkozó adatvédelmi és adatbiztonsági szabályoknak való megfelelésben.

1.1.1 Nehézségek

Mivel az elmúlt pár évtizedben csekély számú adatvédelmi szabályozás lépett érvénybe, illetve az ezirányú próbálkozások is erőtlennek bizonyultak, nem állnak rendelkezésre kiforrott gyakorlatok. Ennek köszönhetően várható volt, hogy folyamatos finomításokra lesz szükség annak érdekében, hogy a rendelet apró pontatlanságai, felbukkanó kiskapui kiküszöbölhetővé váljanak.

Bármennyire is hosszúnak és megengedőnek tűnik a kétéves türelmi időszak, az annak letelte előtti pár hétben sokfelől hallhattunk aggódó hangokat, miszerint a nagy többségnek nem sikerült teljeskörűen eleget tenni a rendelet által előírt követelményeknek. Ennek általános oka az lehetett, hogy a határozatnak való megfelelés koránt sem annyira egyértelmű. A GDPR ugyanis jogi oldalról közelíti meg az adatvédelem kérdéskörét és ennek megfelelően építi fel a követelményrendszerét is, amit azonban technikai oldalról nem egyszerű megfogni.

A személyes adat definíciójában találkozunk az azonosíthatóság fogalmával. Egy természetes személy azonosítható, ha létezik olyan plauzibilis támadás, amely lehetővé teszi az érintett azonosítását, és észszerű esély van arra, hogy ez a támadás sikerrel is jár. A plauzibilitás itt annyit jelent, hogy a támadónak van elég motivációja, hogy elindítson egy ilyen offenzívát. A támadás sikerének becsléséhez olyan tényezőket kell figyelembe venni, mint a technika jelenlegi állapota, a támadás pénzbeli és időbeli költségei, valamint, hogy a támadó rendelkezhet-e az offenzíva kivitelezéséhez szükséges technológiai tudással és felszereléssel. A definícióegyüttes segítségével is rávilágíthatunk a fent említett problémára: Míg jogi kontextusban a GDPR egy precízen megfogalmazott

törvénynek tűnhet, addig technológiai oldalon ezen elvont fogalmak nehezen interpretálhatók, hiszen mérésre kevéssé alkalmasak.

A „technológia jelenlegi állapota” nevezetű követelmény egy másik komoly komplikációra mutat rá a GDPR-ral kapcsolatban: Nem elég ugyanis, hogy egy vállalat egyszeri nagyobb ráfordítással kiépíti a rendeletben előírtak szerint az infrastruktúráját és az adatkezelési gyakorlatát. Mivel a technológiai megoldások rendkívül gyorsan változnak és az IT szektor is folyamatosan fejlődik, a támadók is egyre kifinomultabb eszközökkel próbálnak behatolni a különböző informatikai rendszerekbe. Ebből kifolyólag egy nagyobb kezdeti beruházással kiépített rendszer tud ugyan jó alapként szolgálni, azonban az út a GDPR-konformitáshoz itt nem ér véget, hanem végigkíséri a vállalatot annak egész életciklusán. A technikai változásokat és újításokat követni, az adatvédelmi infrastruktúrát pedig folyamatosan naprakészen tartani, fejleszteni szükséges.

Annak érdekében, hogy egy cég lépést tudjon tartani az újabb és újabb informatikai innovációkkal - ezáltal pedig maximális adatvédelmet biztosíthasson - az előírás javasolja az adatvédelmi szakértőkre való támaszkodást. Az ő feladatkörükbe tartozik a tájékoztatás és tanácsadás, az adatvédelmi eljárások ellenőrzése, illetve a felügyeleti hatóságokkal való kapcsolattartás. A nehézség abban rejlik, hogy egy ilyen szakértőnek mind a technológia, mind pedig a jog világában otthonosan kellene mozognia. Mivel azonban a rendelet előtt ez a munkakör nem létezett, illetve erre irányuló képzések sem voltak, összességében nem egyszerű manapság ilyen kvalitásokkal rendelkező szakembert találni.

1.1.2 Az anonimizálás

A korábban említettek alapján jól látható, hogy a GDPR-megfelelés kiépítése és fenntartása valóban nagy kihívás elé állította az EU területén üzemelő, illetve az európai polgárok adatait kezelő és feldolgozó vállalatokat. A jó hír, aminek hallatán sokak szeme felcsillanhat, hogy a rendelet egyáltalán nem vonatkozik az anonimizált adatokra, így amennyiben csak azokkal dolgozunk, teljesen figyelmen kívül hagyhatjuk a GDPR-t. Rossz hír viszont, hogy a valódi anonimitás elérése koránt sem triviális feladat.

A határozat szerint egy adat anonim, amennyiben általa az érintett nem, vagy többé nem azonosítható. Azonosíthatóság alatt a már említett, igen absztrakt módon definiált fogalmat értjük, a technológia jelenlegi állapotával és a plauzibilitással. Ezzel

pedig rögtön rámutattunk, hogy hol kezdődnek a gondok az anonimizálással, hiszen visszatértünk a jog és a technológia konfliktusához. Fontos kiemelni, hogy amennyiben sikerül az anonimizálást tökéletesen kivitelezni, az így kapott adathalmaz akkor és csak akkor esik a GDPR hatáskörén kívülre, ha az eredeti adathalmaz véglegesen törlésre kerül. Ha ugyanis egy anonim adathalmazt publikálunk, az mindaddig személyes adatnak számít - és az adatkezelő felelősséggel tartozik érte - ameddig az eredeti adathalmaz rendelkezésre áll, hiszen segítségével adatlopás esetén akár az egész anonim adatbázis visszafejthetővé válik.

A definícióból származó nehézségek mellett az anonimitás gyakorlati megvalósításánál is több problémába ütközünk. Mint azt a számos adatvédelmi incidens is bizonyítja, az anonimnak hitt adathalmazok gyakran szivárogtatnak személyes információkat. Külön-külön ártalmatlannak tűnő, „anonim" adatbázisok kombinálásával ugyanis kiderül, hogy mégis azonosítható néhány érintett; majd pedig minél több adathalmazt használunk fel, úgy tárulnak fel sorban a személyazonosságok. A *Big Data* kora ebből a szempontból hatalmas kihívás elé állítja a kutatókat, hiszen nem elég, hogy napról napra egyre több adat áll rendelkezésre, bennük a legkülönfélébb információkkal, de a rekordok magas dimenzionalitása miatt egyre nehezebb az érintettek kilétének elfedése is [1]. Azonban, mint azt egy, az Egyesült Államokban végzett kutatás megmutatta [2], nincs is feltétlenül szükség a nagy dimenziókra, hiszen az USA népességének 87%-át be lehet azonosítani csupán az ötszámjegyű irányítószám, a nem és a születési dátum segítségével.

Az anonim adatbázisok egy fontos tulajdonsága a használhatóság, azaz *utility*. A tökéletes anonimitást akkor érjük el, ha lényegében semmilyen információt nem adunk ki, ekkor viszont semmit hasznát nem vesszük az adathalmaznak, a *utility* nulla. A legértékesebb maga az eredeti adathalmaz, ahol ugyan a *utility* maximális, az érintettek anonimitása viszont nulla. Így az anonimizálás nem más, mint kompromisszum az adatalanyok anonimitása és az adatok használhatósága között. A cél minden esetben a személyek minél magasabb fokú elrejtése, mialatt az adathalmaz információtartalma a lehető legnagyobb. Ez pedig komoly mérnöki feladat.

Napjainkban az anonimizálás egyre aktívabban kutatott terület, egyrészt valószínűleg a GDPR-nak, másrészt pedig az adatvédelmi mozgalom fellendülésének köszönhetően. Rengeteg kész implementáció és termék van már a piacon. A legjelentősebb nehézségek általában a megfelelő anonimitás bizonyításakor merülnek fel.

Meglehetősen sokféle modellel, megközelítéssel találkozhatunk, melyekhez javasolt alkalmazási terület, illetve ismert támadások is társulnak. Az anonimizáláshoz sokféle eszköz használatos, például az adatok zajosítása, permutálása vagy más módon történő randomizálása, bizonyos attribútumok általánosítása vagy elnyomása.

Gyakori hiba a pszeudoanonimitás és az anonimitás fogalmak összekeverése. Ugyan a pszeudoanonimizálás is adatvédelmi technika, sőt, a GDPR is definiálja a fogalmát és javasolja a használatát, a pszeudoanonim adathalmazok azonban továbbra is a rendelet hatáskörébe tartoznak. Pszeudoanonim adatok esetén az adatalany beazonosításához további információ szükséges, amelyet külön tárolnak annak érdekében, hogy biztosítsák a személyes adatok védelmét.

Az általunk használt anonimizálási modell a k-anonimitás, amivel számos kutatásban találkozhatunk [3] [4] [5] [6] [7]. Egy k-anonim adatbázisban minden rekordhoz létezik legalább *(k-1)* másik rekord ugyanazokkal a kvázi azonosító értékekkel. Egy másik - általunk alkalmazott - megközelítés pedig a gyakorlatban sokat használt Differential Privacy, mely valójában egy adatbázistulajdonság, ami biztosítja, hogy a lekérdezések ne szivárogtathassanak semmiféle személyes információt.

1.2 A dolgozat célkitűzése

Bár sokféle komplex követelménynek kell eleget tennünk, amennyiben sikerül implementálnunk egy működőképes eljárást, az anonimizálással valóban mentesülhetünk azon terhek alól, amit a GDPR ró a személyes adatok kezelőire és feldolgozóira. Felmerül azonban még egy probléma: Az idáig elkészült, napjainkban használt eljárások és termékek csak abban az esetben képesek elvégezni az anonimizálást, ha kéznél van az eredeti adathalmaz is. Ez viszont nem jelent mást, minthogy legalább az anonimizálási folyamat végéig a rendelet igenis vonatkozik az anonimizálást végző szervezetre. Addig a pillanatig, ameddig az eredeti adathalmaz végleges törlése meg nem történik, a GDPR által előírt kötelezettségeknek meg kell felelni, tehát ugyanúgy ki kell építeni az adatvédelemhez szükséges infrastruktúrát és *policy*-rendszert.

A dolgozatban e problémakörnek az orvoslására teszünk javaslatot kliensoldali anonimizálás segítségével. Nézzünk egy konkrét példát: Napjainkban rohamos ütemben terjednek a legkülönfélébb okosotthon eszközök. Sokféle okosporszívóval találkozhatunk a piacon, amelyek takarítás közben bejárják és feltérképezik az egész házat annak érdekében, hogy minél jobban optimalizálni tudják működésüket. A porszívó által

gyűjtött információk, vagy éppen az esetleges hibaüzenetek a terméket gyártó cég számára is értékesek lehetnek, hiszen ezek alapján az eredeti algoritmus finomítható és továbbfejleszthető. Ebben az esetben az eszköz maga a kliens - egy autonóm ágens -, aki adatokat gyűjt, majd küld fel egy központi szervernek, ahol ezeket eltárolják és elemzik. Ugyan az e célra történő adatgyűjtéssel általában megbarátkoznak az emberek, mégsem szeretné senki, hogy eközben bármiféle személyes információ kiszivárogjon és a vállalat kezébe kerüljön. Ez korántsem valamiféle paranoia: 2018-ban egy robotporszívó gyártó cég már gyanúba keveredett, hogy a termékhasználók otthonáról gyűjtött információkat nagyobb vállalatoknak tervezte eladni.

A fenti problémára kínál megoldást az általunk kidolgozott protokoll, amely már az adatokat azok felküldése előtt, kliens oldalon anonimizálja. Ezzel nem csupán azt érjük el, hogy az adatgyűjtő fél ténylegesen mentesül minden, GDPR-ral kapcsolatos kötelezettség alól, hanem egy ezzel a protokollal kommunikáló termék a vásárlókban is nagyobb bizalmat kelthet, az adatvédelemtudatos emberek számára egy vonzó alternatívát nyújthat.

Szerveroldalon tehát k-anonim adatbázist építünk. E modellt megannyi kutatásban vizsgálták már az elmúlt évek folyamán, melynek köszönhetően azzal is tisztában vagyunk, hogy milyen gyengeségei vannak, illetve mely támadásokra érzékeny [8] [9]. Annak érdekében, hogy ezen sérülékenységeket kiküszöböljük és a támadási felületet tovább redukáljuk, a k-anonim adathalmazon olyan módosításokat végzünk, melyek következtében az már a Differential Privacy által támasztott követelményeknek is képes lesz eleget tenni.

1.3 A dolgozat felépítése

A dolgozat következő fejezetében részletesen bemutatjuk a k-anonimitás működését és az ismert támadási modelleket. Emellett ismertetjük a Differential Privacy fogalmát, illetve azt, hogy mikor és hogyan érdemes használni.

A harmadik fejezetben prezentáljuk az általunk kidolgozott protokollt. Megmutatjuk, hogyan történik a különböző attribútumok kategorizálása és kezelése, egy már meglévő adatbázis használata vagy éppen egy új inicializálása, illetve, hogy milyen támadási modellekben gondolkodtunk.

Mindezek után, a negyedik fejezetben elvégezzük a protokoll anonimizálási mechanizmusának kiértékelését. Az anonimizálás szimulációja után a kapott adathalmazon folytatunk vizsgálatokat különböző metrikákat felhasználva.

Az ötödik fejezetben az alkalmazott algoritmuson, illetve a felhasznált adathalmazon végzett módosításokat mutatjuk be, melyeknek köszönhetően már a Differential Privacy modell által előírt elvárásoknak is meg tudunk felelni.

2 Kapcsolódó irodalom

Az alábbiakban ismertetjük a dolgozat további részeiben is felhasználandó fogalmakat. Bemutatjuk a k-anonimitás és a Differential Privacy témaköreit vizsgáló kutatások legfrissebb eredményeit, illetve mindkét koncepció esetén kitérünk a korlátokra, valamint a biztonságosság és a támadhatóság kérdéseire is.

2.1 A k-anonimitás

Elsőként a k-anonimitás fogalmával és a k-anonimizáló algoritmusok adatvédelemben betöltött szerepével foglalkozunk.

Az adatokat kezelő szervezetek és vállalatok általános célja úgy nyilvánosságra hozni információt, hogy a kiadott adathalmazban található egyének közül senkit se lehessen egyértelműen beazonosítani. Az anonimizálás olyan adatfeldolgozási technika, melynek célkitűzése a személyazonosításra alkalmas adatok eltávolítása vagy módosítása annak érdekében, hogy a névtelen ismertetők már egyetlen konkrét személyhez se legyenek társíthatók. Azt viszont mindenképpen szem előtt kell tartani, hogy csak azon adatok kerüljenek törlésre, melyek e cél eléréséhez feltétlenül szükségesek, máskülönben az eredeti tendenciák már nem lesznek megfigyelhetők és az adathalmaz elértéktelenedhet.

A k-anonimitás fogalmát az alábbi példa adathalmaz segítségével vezetjük be.

Név	Születési idő	Nem	Irányítószám	Betegség
Kovács János	1967.12.15.	Férfi	8156	Influenza
Kis Ernő	1971.02.21.	Férfi	1287	Lábtörés
Tóth Piros	1980.04.10.	Nő	7650	Leukémia
Németh Mária	1982.05.30.	Nő	7689	Hepatitis
Nagy András	1973.11.19.	Férfi	1200	Tüdőrák
Gál József	1962.09.09	Férfi	8120	Májciszta

2.1.1. táblázat: Betegadatok

Naiv hozzáállással csak az egyértelmű azonosítókat - név, személyi igazolvány szám - távolítjuk el. A példa táblázatban így a *Születési idő*, *Nem*, *Irányítószám* és a *Betegség* oszlopok maradnak. Amennyiben egy támadó csupán ehhez - a már módosított - adathalmazhoz fér hozzá, háttérinformációk nélkül egyetlen személyt sem ismerhet fel. Ha viszont feltételezzük, hogy az alábbi adatbázis is publikusan elérhető, a két

táblázatban egyaránt szereplő *Születési idő, Nem* és *Irányítószám* oszlopok összevetésével egy nemkívánatos harmadik fél könnyen juthat azon információ birtokába, miszerint Nagy András tüdőrákban szenved. E sérülékenység az irodalomban „összekapcsolás támadás" (*linking attack*) néven vált ismertté.

Név	Születési idő	Nem	Irányítószám
Végh Jenő	1983.08.07.	Férfi	8213
Pál Annamária	1959.10.18.	Nő	9210
Tóth Zsófia	1990.02.01.	Nő	1110
Nagy András	1973.11.19.	Férfi	1200

2.1.2. táblázat: Szavazópolgárok jegyzéke

Ahogy azt a tudományos írásában *Sweeney* is belátta, az ily módon publikált adathalmazok nem nevezhetők anonimnak [10].

A k-anonimitás segítségével az előzőekben említett támadás kivédhető. Ahhoz viszont, hogy a k-anonimitás fogalmát bevezethessük, először az általa használt attribútumtípusokkal kell foglalkoznunk.

- Egyértelmű azonosító: Egy adott személy kizárólagos tulajdonsága; olyan adat, mely e személy kilétét egyértelműen meghatározza (pl. név, személyi igazolvány szám).
- Kvázi azonosító: Olyan adat, mely önmagában nem, más információkkal közösen viszont már képes egy adott személy egyértelmű azonosítására (pl. születési idő, nem).
- Szenzitív adat: Egy adott személyről érzékeny információt hordozó ismertető (pl. betegség, politikai nézetek).

A fentiek ismeretében a k-anonimitás a következőképpen definiálható: Legyen $RT(A_1, \ldots, A_n)$ egy adattáblázat, QI_{RT} pedig a hozzá kapcsolódó kvázi azonosítók. Az *RT* tábla akkor és csak akkor elégíti ki a k-anonimitás feltételeit, ha az $RT[QI_{RT}]$-ben található összes értékszekvencia az $RT[QI_{RT}]$-ben legalább *k*-szor megjelenik. Más szavakkal tehát egy adathalmaz akkor k-anonim, amennyiben az egyes emberek a róluk tárolt információ alapján nem különböztethetők meg legalább *(k-1)* másik olyan egyéntől, akikről szintén tárol információkat az adatbázis [3]. Következésképpen a k-anonimitás képes az adatvédelem fokának kifejezésére.

Példaként nézzük a már korábban látott betegadatok k-anonim változatát, ahol $k = 2$, a kvázi azonosítók pedig a *Születési idő, Nem* és *Irányítószám* ismertetők.

Anonimizálás során az egyértelmű azonosítók mindig eltávolításra kerülnek, a szenzitív adatok pedig érintetlenül maradnak. A táblázatban bármely kvázi azonosító esetén minimum kettő egyforma sor található, így a módosított adathalmaz $k = 2$ értékre tényleg kielégíti a k-anonimitás elvárásait.

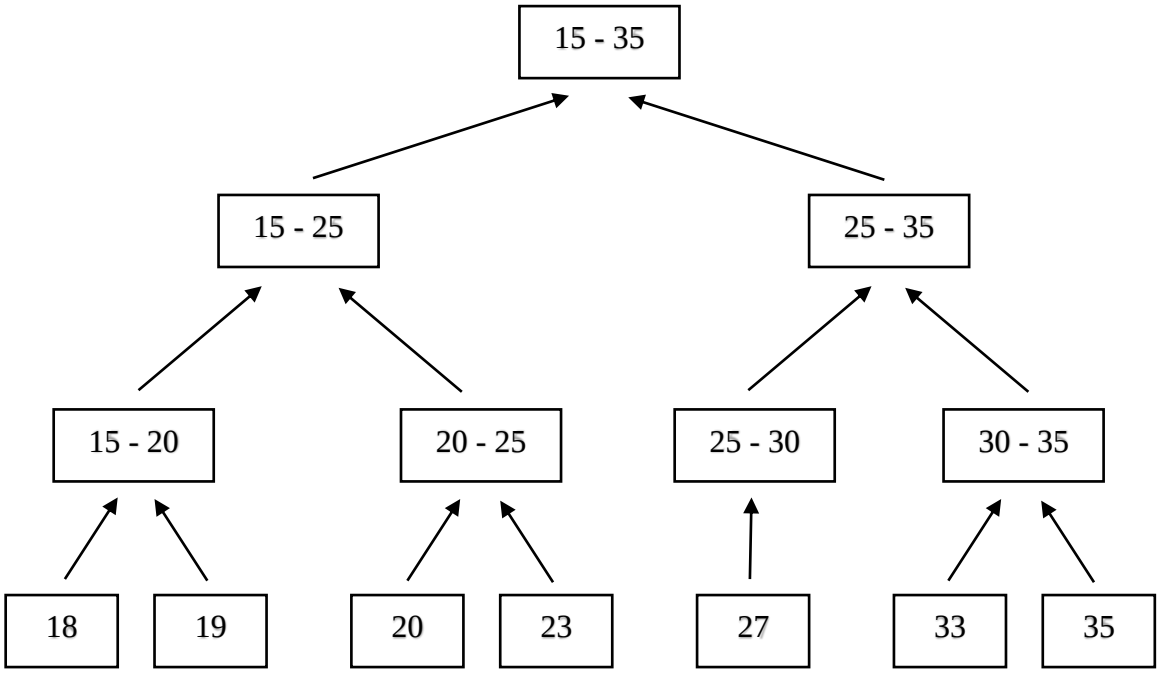
Születési idő	Nem	Irányítószám	Betegség
1960-as évek	Férfi	81**	Influenza
1970-es évek	Férfi	12**	Lábtörés
1980-as évek	Nő	76**	Leukémia
1980-as évek	Nő	76**	Hepatitis
1970-es évek	Férfi	12**	Tüdőrák
1960-as évek	Férfi	81**	Májciszta

2.1.3. táblázat: Betegadatok 2-anonim változata

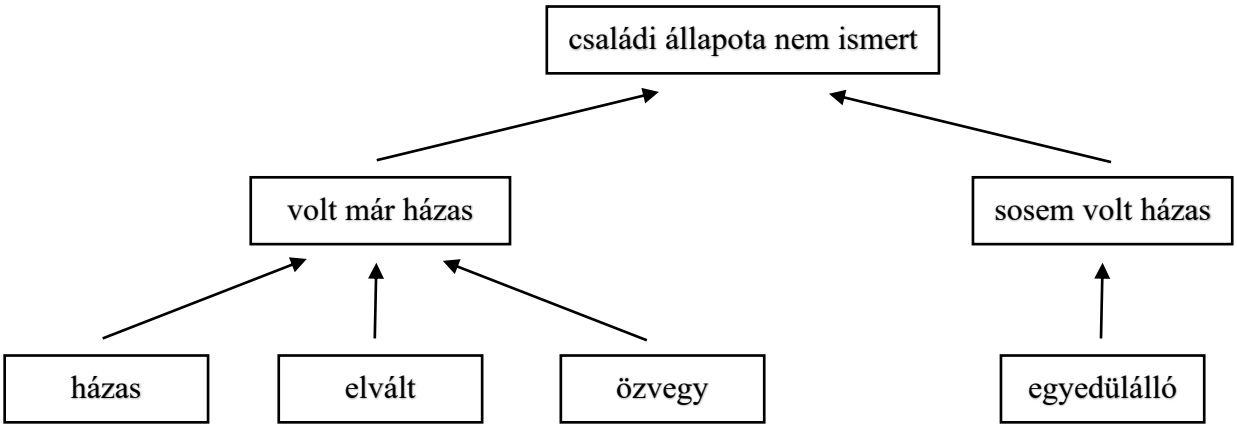
Beláthatjuk, hogy ily módon a már említett *linking*-támadás nem vihető végbe, mivel nem állapítható meg teljes bizonyossággal, hogy az egyező sorok közül melyik vonatkozik az adott személyre.

2.1.1 Generalizálás

Egy adathalmaz k-anonim formára alakítása többféleképpen történhet. A két legfontosabb megközelítés a kvázi azonosítók általánosítása, illetve elnyomása. Generalizáláskor az adatot valamilyen hierarchia segítségével egy kevésbé specifikus, általánosabb formára hozzuk. Szöveges adatok esetén a hierarchia az információ formájától függ, számoknál viszont általában magától értetődően egyre tágabb intervallumokat használunk. Az alábbi ábrákon konkrét példákat mutatunk az általánosítás működésére [3].



2.1.1. ábra: Életkorok generalizálása



2.1.2. ábra: Családi állapotok generalizálása

2.1.2 Elnyomás

Amennyiben k-anonimizálás során a kvázi azonosítók értékének elnyomása mellett döntünk, akkor az attribútumokat a táblázatból eltávolítjuk, egyáltalán nem hozzuk őket nyilvánosságra. Az elnyomás abban az esetben használatos, ha számottevően sok olyan adatsor található az adathalmazban, melynek előfordulási gyakorisága kisebb, mint *k*, így pedig a k-anonimitás feltételeinek kielégítéséhez nagy mennyiségű általánosításra lenne szükség. Az eljárás jókora odafigyelést igényel, mivel hatására az elnyomott attribútum által tartalmazott információ elvész és hiába anonim az adathalmaz, ha semmiféle következtetést nem lehet belőle levonni.

2.1.3 Támadási modellek

Önmagában a k-anonimitás nem nyújt 100%-os biztonságot az adatok számára. A következőkben példák segítségével bemutatunk két olyan lehetséges támadási formát, melyek a k-anonimitás sebezhetősége miatt vihetők végbe [11].

- Homogenitás támadás: Először nézzük az alábbi táblázat első két sorát. Még ha tudomásunk is van arról, hogy a két 12 éves fiú Adam és Bob, csupán 50%-os bizonyossággal feltételezhetjük, hogy Adam vagy Bob ért el 79 pontot a teszten. Másrészről viszont, ha pontos ismeretünk van arról, hogy a két lány Anna és Bea, akkor 100%-os bizonyossággal tudjuk, hogy mind Annának, mind pedig Beának 99 pontja lett a teszten. Általánosan fogalmazva tehát a homogenitás támadás végrehajtható, ha személy-leleplező jegyek egy adott kombinációjához mindig ugyanaz az érzékeny adat társul.
- Háttértudás támadás: Tegyük fel, hogy ismerjük az osztály két 12 éves fiúját, Adam-et és Bob-ot, illetve Bob-ról még azt is biztosan tudjuk, hogy ha 90 pont felett teljesít egy dolgozatban, akkor a szülei csokoládéval jutalmazzák. Ha Bob-ot a tesztpontszámok kihirdetésének napján csokoládéval a kezében látjuk, az extra tények ismeretében nem csak arra következtethetünk, hogy Bob 98 pontot ért el a teszten, hanem arra is, hogy Adam pedig 79-et. A példa segítségével beláthatjuk, hogy külső forrásból származó információ k-anonim adathalmazzal való kombinálása is adatszivárgáshoz vezethet.

	Életkor	**Nem**	**Tesztpontszám**
1.	12	Férfi	79
2.	12	Férfi	98
3.	13	Nő	99
4.	13	Nő	99
5.	14	Férfi	82
6.	14	Férfi	85

2.1.4. táblázat: 2-anonim tesztpontszámok

Az említett sérülékenységek kiküszöbölésére a k-anonimitás továbbfejlesztéseként már több - az irodalomból szintén ismert - modellt is létrehoztak, mint például az *l-diversity* [12], vagy a *t-closeness* modellek [13].

2.2 Differential Privacy

Amennyiben k-anonimizálással dolgozunk, ahhoz, hogy megtaláljuk a megfelelő *k*-t, először valamilyen módon ki kell találnunk azt, hogy mik lehetnek egy esetleges támadó céljai és adottságai, illetve mennyit ismerhet az adathalmazból. A gyakorlatban ez a legtöbbször nem kivitelezhető, csupán feltételezésekkel élhetünk. A Differential Privacy modell felhasználásával azonban bármilyen információ megvédhető, és nem számít az sem, hogy a támadó mit tud az adatokról.

A Differential Privacy fogalmát egy példa segítségével vezetjük be. Az alábbi hipotetikus táblázat felhasználók jelenlegi pozícióját tartalmazza. A 'Hány felhasználó tartózkodik a hármas pozícióban?' lekérdezésre a helyes válasz a 3. Amennyiben viszont az 5. felhasználó nem szerepelne az adatbázisban, a helyes válasz a 2 lenne, ezáltal pedig egy potenciális támadó arra a következtetésre juthatna, hogy az 5. felhasználó a hármas pozícióban tartózkodik [14].

A Differential Privacy modell fő célja tehát, hogy egy nemkívánatos harmadik személy ne lehessen biztos abban, hogy egy adott illető jelen van-e az adathalmazban, avagy sem.

	1.pozíció	**2.pozíció**	**3.pozíció**	**4.pozícó**
1.felhasználó	0	0	1	0
2.felhasználó	1	0	0	0
3.felhasználó	1	0	0	0
4.felhasználó	0	1	0	0
5.felhasználó	0	0	1	0
6.felhasználó	0	0	1	0

2.2.1. táblázat: Felhasználók jelenlegi helyzete

Egy lekérdezés tehát abban az esetben rendelkezik a Differential Privacy tulajdonsággal, ha semmi, vagy csupán nagyon kevés múlik azon, hogy az adatbázis tartalmazza-e egy adott személy adatait. A megközelítés zajt ad a lekérdezés eredményéhez az adatok védelmének érdekében. Ez a zaj gondoskodik arról, hogy egy válasz valószínűsége - amikor az adott felhasználó is az adathalmaz tagja - majdnem megegyezzen ugyanezen válasz valószínűségével akkor, amikor az adott felhasználó nem szerepel az adatbázisban [14]. Matematikai jelekkel írva: $P(válasz_n \mid felhasználó\ szerepel\ az\ adathalmazban) \approx P(válasz_n \mid felhasználó\ nem\ szerepel\ az\ adathalmazban)$. Mivel a

’majdnem megegyezik’ kifejezés nem egyértelmű, a koncepció egy $\mathcal{E}$ paraméter segítségével adja meg, hogy a két valószínűség mennyire közelítse meg egymást.

A módszer szemléltetéseképpen a fentebb már említett példát kissé elnagyolva folytatjuk. Ehhez legyen *t* a lekérdezés eredménye, *c* pedig a helyes válasz. A mechanizmus így a $t = c + zaj$ értéket adja. Tegyük fel, hogy a ’Hány felhasználó tartózkodik a hármas pozícióban?’ lekérdezésre a $t = 6$ válasz érkezik, a támadó pedig az 5. felhasználó kivételével mindenkinek a helyzetét ismeri. Ekkor a következő lehetőségek állnak fent: Az 5. felhasználó a hármas pozícióban található és a zaj értéke 3, avagy a felhasználó nem a hármas pozícióban tartózkodik és a zaj értéke 4. A támadó egyik eshetőségben sem lehet 100%-ig biztos.

Formálisan megfogalmazva, egy *A* algoritmus minden *t*-re kielégíti az $\mathcal{E}$-Differential Privacy követelményeit, amennyiben minden két szomszédos adatbázis (*D* és *D’*) esetén teljesül a következő egyenlőtlenség [15]:

$$\frac{\Pr\left(A(D) = t\right)}{\Pr\left(A(D') = t\right)} \leq \exp\left(\mathcal{E}\right)$$

Két adatbázis abban az esetben mondható szomszédosnak, ha maximum egy sorukban különböznek.

Az $\mathcal{E}$ paraméter megválasztásakor a használhatóság és az anonimitás mértéke között kell kompromisszumot keresni - hiába tűnik vonzónak egy olyan csekély $\mathcal{E}$, mint a 0,01 vagy a 0,1, egy ilyen kicsi érték - főleg kis méretű adathalmazok esetén - könnyen a hasznosság rovására mehet [16].

2.2.1 Alkalmazott zaj típusa

A Differential Privacy mechanizmus alkalmazásakor a lekérdezés eredményeként olyan visszajelzést várunk, ami nagy valószínűséggel van közel a helyes válaszhoz. Az $\mathcal{E}$ paraméter megjelenése miatt az állandó zaj használata nem lehetséges. Tudományos munkájában *Dwork* a Laplace-zaj alkalmazását javasolja [17]. Tegyük fel, hogy egy támadó egy *f(x)* lekérdezést intéz az *X* adathalmaz irányába, amely kérdésre a helyes válasz *a*. Továbbá tegyük fel azt is, hogy az említett adatbázisra nézve egy - a Differential Privacy tulajdonsággal rendelkező - $k_f()$ mechanizmus már implementálva van; a rendszer válasza pedig *R*. *Dwork* tétele szerint a maszkolt válasz $k_f(X) = R = a + y$ lesz, ahol *y* a 0 átlaggal és $b = \Delta f / \mathcal{E}$ skálázási paraméterrel rendelkező Laplace-eloszlásból származó

zajt reprezentálja. A skálázási paraméter egyenletében Δf az *f(X)* értékben jelentkező maximális eltérést képviseli, amennyiben az *X* halmazban pontosan egy bemenet változik. Így fordulhat elő például, hogy a támadó által megszerzett adat pontosan egy rekordban különbözik az *X* által tárolt adatoktól.

Az *f(X)* lekérdezésre nézve tehát Δf az ún. globális szenzitivitást reprezentálja, amely valós számmal fejezi ki, hogy egy adott személy az *f* függvény kimenetét milyen mértékben befolyásolja [16]. Amennyiben a maximumszámítást az összes olyan *x* és *y* adathalmazokra végezzük, melyek pontosan egyetlen bejegyzésükben térnek el, a globális érzékenység képlete a következő:

$$\Delta f = \max | f(x) - f(y)|$$

A Laplace-zaj alkalmazása mellett többször találkozhatunk még a Gauss-féle mechanizmus használatával is, mely a lekérdezés eredményéhez Gauss-eloszlásból származó zajt ad, és a Laplace-konstrukcióval ellentétben nem $\mathcal{E}$-Differential Privacy-t, hanem csupán közelítő, $(\mathcal{E}, \delta)$-Differential Privacy-t ér el, ahol δ azt a valószínűséget jelzi, hogy néhány egyén esetén nagyobb lehet az adatszivárgás mértéke, mint másoknál.

A Differential Privacy modell eredeti formájában nem alkalmazható szöveges vagy kategorikus adatokon, ekkor más megközelítésre van szükség. A különböző zajok hozzáadása helyett ilyen esetekben a mechanizmus elődjét, a *Warner* által bevezetett ún. randomizált válasz módszert használjuk, melyet egy példa segítségével mutatunk be [18].

Tegyük fel, hogy ki szeretnénk deríteni, hány diák csalt a teszten. Legyen a csalási arány *p*. Ha csak egyszerűen megkérdezzük, nem fogunk őszinte válaszokat kapni, így más szisztéma szerint járunk el: Mindenkit megkérünk, hogy dobjon fel egy *C* érmét, ahol $P(C = fej) = \theta$ és $P(C = írás) = 1 - \theta$. Ahhoz, hogy az egyes emberek adatai védve maradjanak, a következőt kérjük: Amennyiben a dobás eredménye fej, válaszolják azt, hogy IGEN, ha viszont írás, válaszoljanak őszintén arra a kérdésre, hogy „Csaltál-e már valaha teszten?”. Ezáltal az *Y* megfigyelésünk $Y = (1 - C) + C * Z$ formájú lesz, ahol $Z = 1$, ha már csaltak, $Z = 0$ különben. Így pedig $\pi \equiv P(Y = 1)$ kiszámítható a $\pi = (1 - \theta) + \theta * p$ képlettel, amiből a $p = (\pi - 1 + \theta) / \theta$ egyenletet kapjuk. Ezek után pedig a *p* arány megbecsülhető a π megtippelésével.

2.2.2 Támadási modellek

Beláthatjuk tehát, hogy a Differential Privacy megközelítés segítséget nyújt a különbözőség-támadás elleni védekezésben, amikor is aggregált statisztikai adatok között egy támadó könnyen találhat két olyan csoportot, melyek pontosan egy személyben térnek el, és a két társaság értékei közötti különbség felhasználásával már következtethet az adott egyén értékére. A módszer többek között még a k-anonimitásnál említett linking-támadás ellen is hatásos lehet, habár több hátulütője is van: A véletlenszerű zaj hozzáadásával működő algoritmusok gyengéje például, hogy amikor ugyanarra a statisztikára több különböző zajos választ adnak, a zaj kiátlagolható, s így a valós adat visszanyerhető, illetve, hogy még a torzítás ellenére is - ha túl nagy mennyiségű statisztika kerül nyilvánosságra - rekonstrukciós támadással az adatok visszaállíthatók.

3 A javasolt algoritmus

Az irodalomban eddig ismert algoritmusok esetén a teljes eredeti adathalmaz rendelkezésre kell, hogy álljon ahhoz, hogy az anonimizálást el lehessen végezni. Ez pedig maga után vonja azt is, hogy az anonimizálást végző félnek meg kell felelnie a GDPR által előírtaknak. Ennek kiküszöbölésére javaslunk az alábbiakban egy olyan protokollt, amely lehetővé teszi a kliens (ahol az adat generálódik) számára azt, hogy a saját adatait már helyben anonimizálja. Így már csak olyan adatokat fog felküldeni a szervernek feldolgozásra, amelyek alapján őt nem lehet beazonosítani; ez pedig mindkét fél számára előnyös: Egyrészről, a szervert mentesíti az adatkezelési kötelezettségek alól, másrészről a kliens pedig nagyobb biztonságban tudhatja az adatait, hiszen az adatfeldolgozó sem tudja felhasználni a megszerzett információt a kliens kárára. Emellett érdemes azt is figyelembe venni, hogy egy adatlopás is alacsonyabb kockázatokkal jár. A GDPR rendelet hatáskörén kívül esnek a helyesen anonimizált adathalmazok, így azok szabadon publikálhatók. Ez azt is jelenti, hogy ha anonim adathalmazból történik az adatlopás, a megtámadott vállalat nem vonható felelősségre személyes információk kiszivárogtatása miatt. Az anonimitás okán pedig a klienseknek sem kell aggódniuk, hogy róluk bármiféle konkrét információ kitudódik.

3.1 Használt fogalmak

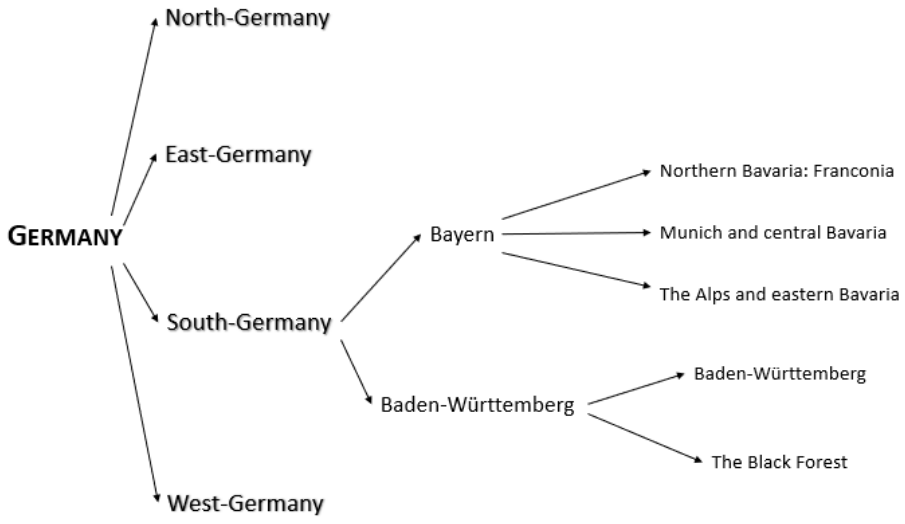
A modellünk a következőképpen néz ki: A kliensek autonóm ágensek, akik folyamatosan adatokat gyűjtenek, majd ezeket felküldik egy központi szervernek, ahol az adatokat tárolják és elemzik. A szerveren k-anonim adatbázis épül. A kliens az információk felküldése előtt megpróbálja elhelyezni magát az szerveren tárolt adatbázis megfelelő ekvivalencia-osztályában, és csak abban az esetben küldi fel az érzékeny adatait a szervernek, ha sikerrel járt.

Mint azt a k-anonimitás ismertetésekor láttuk, háromféle adatbázis attribútumot különböztetünk meg: Az egyértelmű- és kvázi azonosítókat, illetve az érzékeny információt tartalmazó attribútumokat. Anonimizálás során az egyértelmű azonosítókat természetesen teljesen el kell távolítanunk. A kvázi azonosítók értékeit elnyomva vagy generalizálva kapjuk meg az úgynevezett ekvivalencia-osztályokat, amelyek mindegyike legalább k elemet kell, hogy tartalmazzon a k-anonimitás teljesülésének érdekében. Az

érzékeny információkat érintetlenül hagyjuk, mivel lényegében azok adják az anonim adathalmaz értékét.

A protokoll működésének megértéséhez további három definícióra lesz szükségünk, amelyek segítségével a kvázi azonosítókat tudjuk kategorizálni.

- Kategorikus attribútum: Értéke tovább nem finomítható; vagy teljesen elnyomjuk, vagy az eredeti formájában megőrizzük. Ilyen attribútumra példa a „Férfi" vagy „Nő", mint nem megjelölés.
- Intervallum-attribútum: Az eredeti értékeket általánosítással rejtjük el, amely során intervallumokban helyezzük el azokat. Ezen intervallumok aztán valamilyen logika alapján - például az intervallum elfelezésével - könnyedén finomíthatók. Intervallum lehet például a születési dátum, 1989 helyett *[1980 - 1990]* intervallumra generalizálva.
- Hierarchikus attribútum: A generalizálás és a későbbi finomítás nem feltétlenül egyértelmű, nem programozható le olyan egyszerűen, mint azt az intervallum-attribútum esetén láttuk, így a két művelet egy előre definiált hierarchia fa szintjei között felfele, illetve lefele mozgással végezhető el. Erre egy példa a földrajzi helyek általánosítása, amelyet a *3.1.1 ábrán* szemléltetünk. Valamilyen absztrakció definiálható ugyan, de az általánosítás finomítását nem tudjuk leírni könnyen, kénytelenek vagyunk erre a célra valamilyen logika szerint egy hierarchia fát építeni.



3.1.1. ábra: Német régiókból épített hierarchia fa

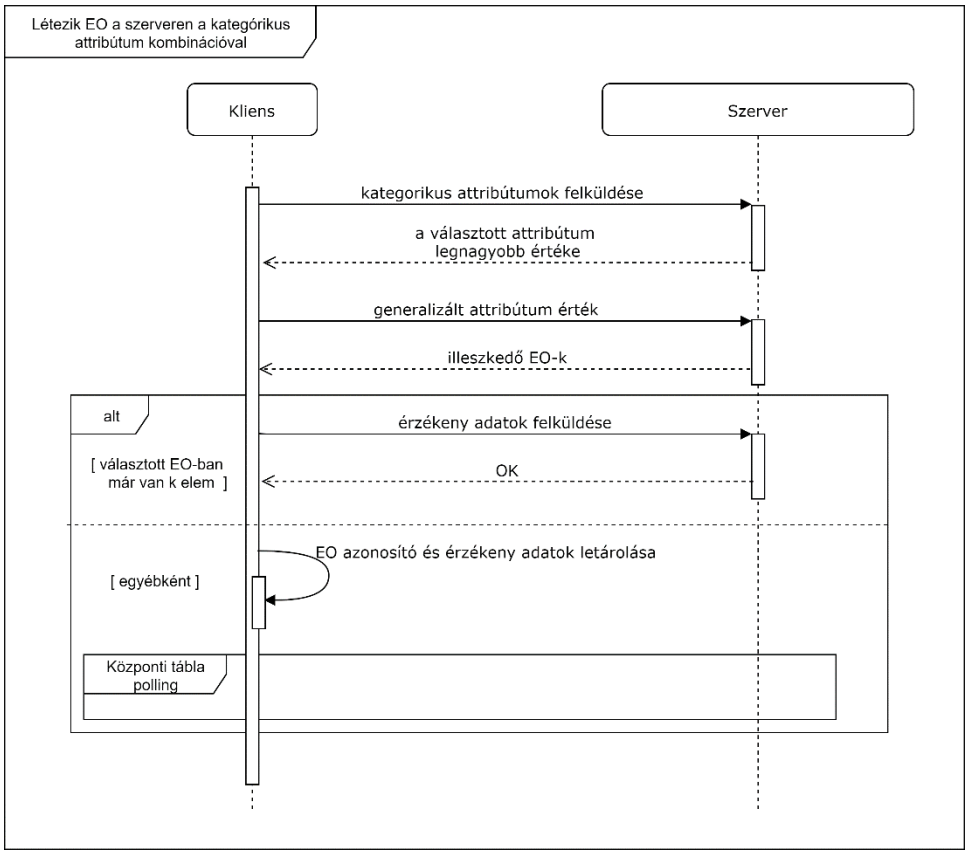
3.2 A javasolt protokoll

Ha a kliensnek rendelkezésére áll valamilyen összegyűjtött adat, a kommunikációt az ehhez tartozó kategorikus attribútumok felküldésével kezdheti meg. Ezt mindenféle kockázat nélkül megteheti, mivel ezekben a mezőkben szereplő adatok semmiképpen sem általánosíthatók, azaz minden esetben az eredeti formájukban kerülnek felküldésre. A szerver az érkező kategorikus attribútum értékeket fogja felhasználni az ekvivalencia-osztályok első körös szűréséhez. Amennyiben ez alapján a szóba jövő ekvivalencia-osztályok száma egy általunk definiált korlát alatt van, ezek mindegyikét elküldjük a kliensnek. Ellenkező esetben két lehetőség fordulhat elő: Több iterációban leküldjük a kliensnek az összes szóba jövő ekvivalencia-osztályt, vagy valamilyen módon megpróbáljuk csökkenteni ezeknek a számát.

Mivel az összes lehetséges kategorikus attribútumérték-kombináció száma valószínűleg nem lesz kiemelkedően magas, nagyobb adathalmazok esetén a szűrés után kapott ekvivalencia-osztályok leküldése továbbra is igen nagy adatforgalmat jelentene. Célszerű lenne tovább szűkíteni ezt a halmazt, amelyhez az intervallumváltozókat hívhatjuk segítségül. Ehhez ki kell jelölnünk előre egy, esetleg több intervallum-attribútumot, amelyekről további információt fogunk kérni a kliensektől.

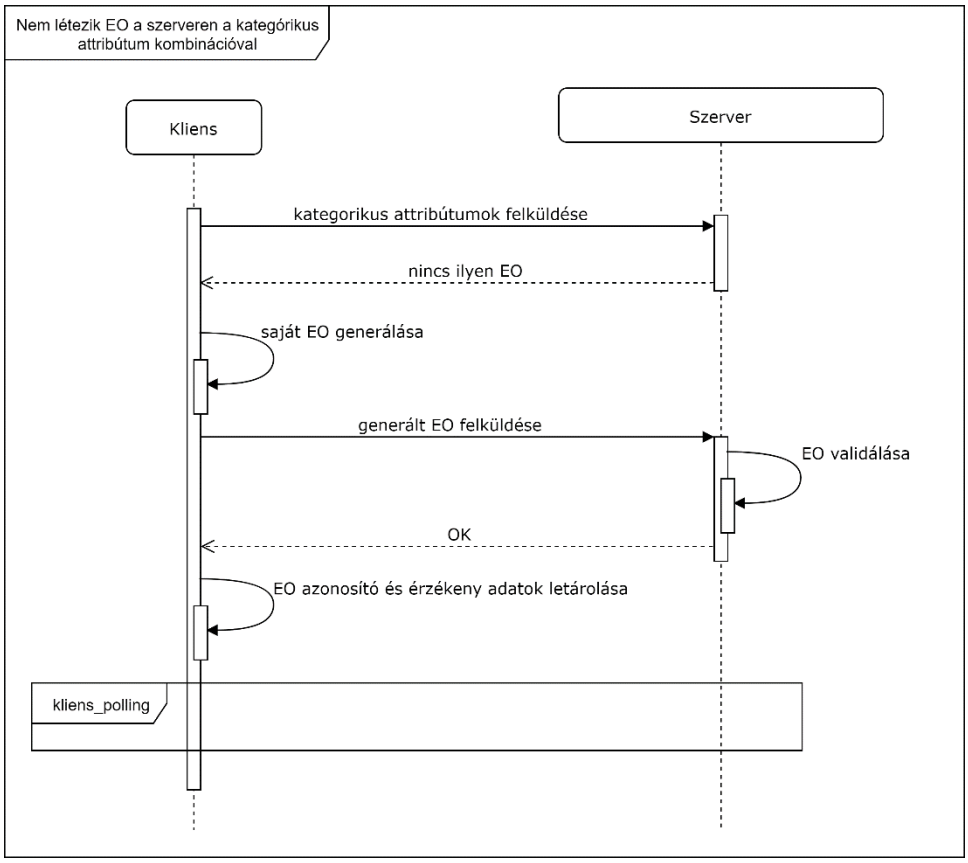
Egy adott kategorikus attribútumérték-kombinációhoz tartozó ekvivalencia-osztályokban különböző méretű intervallumok állhatnak elő az általánosítás során egy-egy tulajdonságra. Annak érdekében, hogy a kliens számára a legnagyobb biztonságot nyújthassuk, a kijelölt intervallum-attribútum legnagyobb intervallumát fogjuk mindig számon tartani a különböző kategorikus attribútumérték-kombinációk mellett. A kommunikáció második lépésében pedig ezt az intervallumméretet küldjük le a kliensnek, hogy az előre egyeztetett attribútumának értékét ez alapján általánosítsa, azaz helyezze el azt egy legalább ekkora méretű intervallumban.

A kliens valamilyen logika szerint elvégzi a kért általánosítást, majd a kapott intervallumot visszaküldi a szervernek. Ez alapján egy újabb szűrést tudunk végezni a szerveroldalon: Megkeressük az összes olyan ekvivalencia-osztályt, amelynek az egyeztetett attribútumában lévő érték metszi a klienstől kapott intervallumot. Ezen a ponton újra dönthetünk, hogy a szóba jövő ekvivalencia-osztályok száma elég alacsony-e már, azaz leküldjük-e őket, vagy további iterációkban próbáljunk tovább szűkíteni.



3.2.1. ábra: Kliens-szerver kommunikáció abban az esetben, ha létezik a szerveroldalon a kliens adatira illeszkedő ekvivalencia-osztály

Amennyiben a szerver leküldi az ekvivalencia-osztályokat, a kliens ezeken végigmegy mindaddig, amíg nem talál egy, a saját adataira illeszkedő osztályt. Ha megtalálja, akkor a választott ekvivalencia-osztállyal együtt felküldi az érzékeny információit. Ellenkező esetben jelzi a szerver felé a sikertelen műveletet, majd generál magának egy megfelelő ekvivalencia-osztályt, és azt küldi fel. Az érzékeny adatokat azonban ekkor még nem adja ki, hanem letárolja, és vár, ameddig az adatok felküldése már biztonságos lesz számára - azaz megvárja az új ekvivalencia-osztály feltöltődését *(3.2.2 ábra)*.



3.2.2. ábra: A javasolt protokoll működése a kliensre illeszkedő ekvivalencia-osztály hiányában

3.3 Az ekvivalencia-osztályok feltöltése

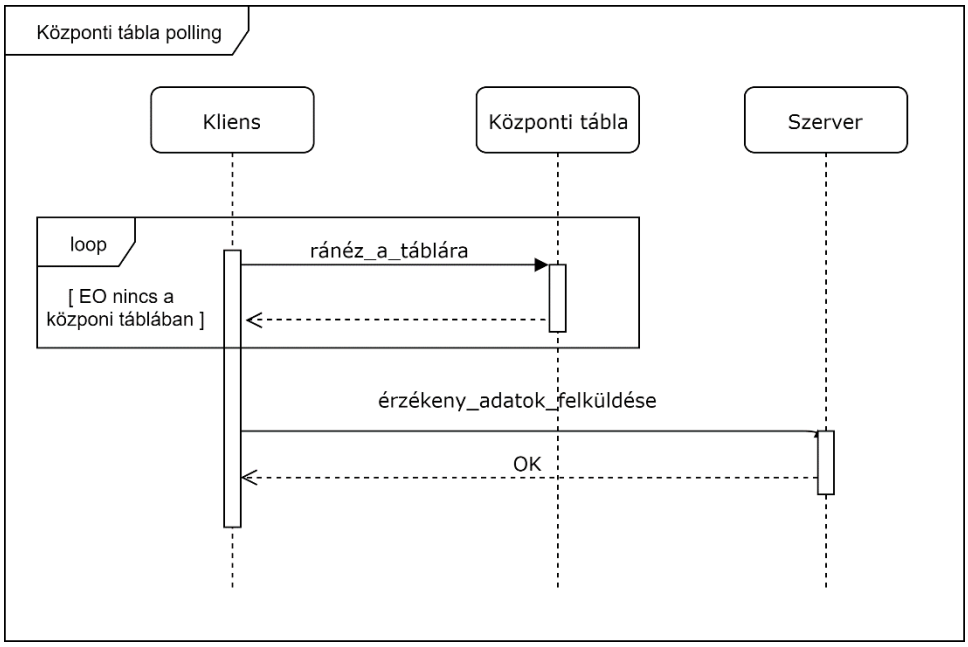
Felmerül a kérdés, hogy ameddig még nincs meg *k* darab elem egy adott ekvivalencia-osztályban és a kliensek más-más időpontokban küldik fel az adataikat, addig hogyan tudjuk garantálni a k-anonimitást.

Nyilvánvaló, hogy ha a kliensek folyamatosan küldenék fel az adataikat attól függetlenül, hogy a rájuk illeszkedő ekvivalencia-osztály hány elemet tartalmaz, akkor az idő nagy részében sérülne a k-anonimitás feltétele. Ezt úgy tudjuk kiküszöbölni, ha a rendszer működése során egy ekvivalencia-osztályban végig vagy nulla, vagy legalább *k* darab elem van. Az azonos ekvivalencia-osztályba tartozó klienseknek tehát egyszerre kellene felküldeniük az érzékeny adataikat, amihez valamiféle szinkronizációra van szükség. Ezt egy ún. központi táblával, illetve az ekvivalencia-osztályok *flag*-elésével tudjuk megoldani.

A központi tábla publikusan elérhető, a szerver ide azokat - a még üres ekvivalencia-osztályokat jegyzi be, amelyekbe már legalább *k* darab kliens jelezte, hogy küldene adatot. A táblában egy bejegyzés tartalmazza az ekvivalencia-osztály

azonosítóját, illetve egy időpontot, amikor a szerver az ide tartozó érzékeny adatokat várja a kliensektől. Így biztosítható az előírt adatvédelmi garanciák teljesítése, megfelelő szinkronizáció esetén ugyanis másodperceken belül *k* eleműre telik az ekvivalencia-osztály. Emellett ez kényelmes módja lehet a forgalomszabályozásnak is, amire sok kliens esetén nagy szükségünk lesz.

Az ekvivalencia-osztályok *flag*-elése megmutatja a klienseknek, hogy a választott osztályba biztonságos-e felküldeni az érzékeny információkat. Ha a *flag* értelmében nincs meg a *k* elem a szerveren, akkor a *3.2.1 ábra* szerint a kliens letárolja a választott ekvivalencia-osztály azonosítóját, illetve a saját érzékeny adatait. Innentől kezdve bizonyos időközönként ránéz a központi táblára, és amint megjelenik a számára érdekes ekvivalencia-osztály azonosítója, a mellette szereplő időpontban felküldi az érzékeny adatait. Ezt szemlélteti a *3.3.1. ábra*.



3.3.1. ábra: A kliens folyamatosan figyeli a központi tábla állapotát, ameddig a hozzá tartozó ekvivalencia-osztály azonosítója meg nem jelenik, és az érzékeny adatait biztonságban felküldheti

3.4 Az ekvivalencia-osztályok finomítása

A feltöltés menetének tisztázása után a következő kérdés az, hogy amint egy ekvivalencia-osztály elérte a *k* elemszámot, meddig töltjük tovább, mikor és hogyan fogjuk ezeket finomítani.

Ha az érzékeny adatok felküldése megtörtént minden olyan klienstől, aki jelzett, a k-anonimitás garanciája rögtön teljesül. Ez pedig azt is jelenti, hogy nincs értelme jóval

tovább növeszteni az adott ekvivalencia-osztályt, hiszen az csak túlanonimizáláshoz, azáltal pedig a kapott adathalmaz *utility*-jének romlásához vezet. Az ekvivalencia-osztályunkat tehát mihamarabb finomítani kell.

Mint azt a szekció elején ismertetett definíciók mutatják, finomítani az intervallum- vagy hierarchia attribútumok mentén tudunk. Javasolt valamilyen sorrendet felállítani az egyes attribútumok között, és eszerint finomítani a megtelt ekvivalencia-osztályokat. Az *3.4.1. táblázat* mutatja be, hogy egy egyszerű struktúrájú kiinduló ekvivalencia-osztály esetén hogyan történik a csiszolás. A felállított sorrend az *Életkor - Lakhely*, azaz először az életkor attribútumot finomítjuk, ezután a lakhelyet, majd újra az életkort, és így tovább. A *Lakhely* attribútum esetén az *3.1.1. ábrán* szemléltetett hierarchiát használjuk fel.

Amint elkészültek a finomított ekvivalencia-osztályok, az eredetit befagyaszthatjuk. Ez azt jelenti, hogy az klienseknek ezt már soha nem fogjuk elküldeni, csak a belőle leszármaztatott, újonnan létrehozottakat. Ez nagyban hozzájárul az anonimizált adathalmaz információtartalmának növeléséhez.

Ekvivalencia-osztály	Életkor	Nem	Lakhely
1	30-40	Férfi	Dél-Németország
1.1	30-35	Férfi	Dél-Németország
1.2	35-40	Férfi	Dél-Németország
1.1.1	30-35	Férfi	Észak Bajorország: Frankföld
1.1.2	30-35	Férfi	München és Közép-Bajorország
1.1.3	30-35	Férfi	Alpok és kelet Bajország
1.1.4	30-35	Férfi	Baden-Württemberg
1.1.5	30-35	Férfi	Fekete-erdő
1.1.1.1	30-32	Férfi	Észak Bajorország: Frankföld
1.1.1.2	32-35	Férfi	Észak Bajorország: Frankföld

3.4.1. táblázat: Ekvivalencia-osztályok finomítása intervallum- és hierarchia attribútumok mentén

3.5 A protokoll felkonfigurálása

A protokoll alkalmazása előtt még van néhány paraméter, amelynek értéket kell adnunk. Legelőször a k-anonimitáshoz szükséges *k*-nak kell egy megfelelő számot választanunk. Arról, hogy ezt hogyan érdemes megtenni, már több cikkben is értekeztek [19]. Emellett még a következőkről kell döntést hoznunk:

- A kommunikáció során - második lépésben melyik intervallum-attribútum mentén végezzük a szűréseket?

A kommunikáció során szükséges szűréshez az intervallum-attribútumok közül logikusan azt érdemes választani, amelyiknek a legnagyobb az értékkészlete, hiszen ebben az esetben tudjuk a szóba jövő ekvivalencia-osztályok halmazát a legjobban szűkíteni. Ha úgy ítéljük meg, erre a célra kijelölhetünk akár több attribútumot is.

- Hány elem beérkezése után tesszük ki az ekvivalencia-osztály azonosítóját a központi táblába?

Kézenfekvőnek tűnik, hogy amint k darab kliens jelentkezett az ekvivalencia-osztályba, rögtön publikáljuk is azt a központi táblába. Azonban ebben az esetben - valamilyen apró hálózati- vagy kliensoldali hiba esetén, ha nem érkezik meg pár adat, nem tartható a k-anonimitás garanciája sem. Érdemes tehát valamilyen $k + \varepsilon_1$ számú klienst összevárni, hogy ezt a hibalehetőséget kiküszöböljük.

- Mikor finomítunk egy ekvivalencia-osztályt?

Felmerül a kérdés, hogy mi legyen az $k + \varepsilon_2$ szám, ami után az ekvivalencia-osztályunkat finomítjuk. Amennyiben elég biztosak vagyunk abban, hogy fenti $k + \varepsilon 1$ számú kliens esetén a garanciánk teljesülni fog, kézenfekvő lehet egy $\varepsilon = \varepsilon 1 = \varepsilon 2$ választás. Ez azt jelenti, hogy amint megkapjuk $k + \varepsilon$ számú klienstől a visszajelzést, az adott ekvivalencia-osztályt rögtön tovább finomítjuk, ezzel maximalizálva az anonim adathalmaz *utility*-jét. Ebben az esetben soha nem lesz olyan, hogy egy kliens rögtön fel tudja küldeni az érzékeny adatait, hiszen a szerveren kizárólag vagy üres vagy befagyasztott ekvivalencia-osztályok lesznek. Más követelmények esetén az ε_2 növelésével ezen változtathatunk.

- Hogyan érdemes felépíteni a szerveroldali adatbázist?

A protokoll leírásából adódik, hogy érdemes külön tárolnunk a kategorikus attribútumérték-kombinációkat a többi kvázi azonosítótól. Erre azért van szükség, mert ezen értékekhez a kommunikáció elején rögtön vissza kell adnunk a választott intervallum-attribútumhoz tartozó legnagyobb intervallum értékét. Emellett szükségünk lesz egy táblára a többi kvázi azonosító számára; egyre az érzékeny attribútumoknak, illetve még egyre a központi táblának, amelyet majd valamilyen formában publikálunk. Érdemes lehet a befagyasztott ekvivalencia-osztályok

számára is létrehozni egy külön táblát, így a kommunikáció folyamán a kereséseket kisebb térben kell végeznünk.

3.6 A protokoll alkalmazása

A protokoll használata előtt két lehetőségünk van: Teljesen üres adatbázissal indulunk, amit nulláról kell felépítenünk, vagy egy már meglévő k-anonim adatbázishoz kezdünk el új rekordokat hozzáadni. Utóbbi esetben könnyebb a dolgunk, mivel csupán a protokoll működéséhez szükséges táblákat kell feltöltenünk a meglévő adatbázis adataival, és már használható is a rendszerünk.

Teljesen üres adatbázis esetén azonban eleinte minden kliensnek saját ekvivalencia-osztályokat kellene generálnia egy darabig. Ez többletmunka a kliensnek és a szervernek egyaránt. A szervernek ugyanis illene a kliensek által gyártott ekvivalencia-osztályokat valamiféleképpen validálni. Egy üres adatbázis mellett pedig a szerver semmilyen segédinformációval nem tud szolgálni a kliensnek, például, hogy milyen méretű intervallumokkal anonimizáljon, vagy hogyan készítsen el egy saját adataira illeszkedő ekvivalencia-osztályt. Észszerűnek tűnik tehát ebben az esetben valamilyen logika mentén kezdeti ekvivalencia-osztályokat generálni a szerveroldalon, mielőtt a protokollt alkalmazzuk. Ez viszont további kérdéseket vet fel.

Ha túl nagy ekvivalencia-osztályokat definiálunk, kiterjedt lesz a kezdeti adatvesztés is, túl részletes osztályok esetén pedig sok olyan ekvivalencia-osztályt gyártunk, amelyek talán soha nem fognak megtelni. Megpróbálhatjuk akár az egész teret lefedni, hiszen ekkor a klienseknek soha nem lenne szükségük saját ekvivalencia-osztály definiálására, ehhez viszont rengeteg osztályt kéne legenerálni, ráadásul ahogy a sémánk dimenzionalitása, illetve az attribútumok értékkészlete nő, úgy a lehetséges ekvivalencia-osztályok tere is egyre gyorsabban bővül, amit lefedni egyre nehezebb lesz. Másik alternatíva, hogy amennyiben lehetőség van rá, támaszkodhatunk a generálás során korábbi statisztikai adatokra is. Ennek segítségével csak a várhatóan megjelenő ekvivalencia-osztályokat fogjuk előre legyártani. Fontos kiemelni, hogy ugyan a kezdeti ekvivalencia-osztályok minősége eleinte erősen befolyásolja a formálódó anonimizált adatbázis minőségét, ez hosszú távon akár elnagyolt osztályok esetén is a maximális *utility*-hez fog konvergálni a folyamatos finomításoknak köszönhetően.

Ha a fenti alternatívák közül választottunk egyet, és a kezdeti ekvivalencia-osztályokat sikerült definiálni, onnantól kezdve a protokoll működésében már semmilyen

eltérés nincs ahhoz az esethez képest, amikor egy már meglévő k-anonim adatbázisra alapozva indítjuk el a rendszerünket.

3.7 A protokoll működése

Egy példa segítségével nézzük meg, hogyan is történik a szerver-kliens kommunikáció, hogyan jön képbe a központi tábla, illetve miként készít egy kliens új ekvivalencia-osztályt az adatai számára.

A korábbiakban leírtak szerint a példában is a protokoll felkonfigurálásával indítunk. A szemléletesség kedvéért egy 2-anonim ($k = 2$) kimeneti adathalmazzal megelégszünk. A szerver egészen addig *flag*-eli az ekvivalencia-osztályokat (azaz jelzi a klienseknek, hogy egyelőre ne küldjenek érzékeny adatokat), ameddig az ekvivalencia-osztályba legalább 3-an nem jelentkeztek ($\varepsilon_1 = 1$). Finomításra pedig csak akkor kerül sor, ha már legalább 4 darab kliens érzékeny információi megérkeztek a szerverre ($\varepsilon_2 = 2$).

Adott tehát egy 2-anonim adatbázis, amelybe a *3.7.2 táblázatban* látható ekvivalencia-osztályok, illetve ezekhez a *3.7.3 táblázatban* található érzékeny információk kerültek be ezidáig. A három klienst, akinek az adatküldését végig fogjuk követni, a *3.7.1 táblázatban* tüntettük fel.

Név	Kor	Nem	Szemszín	Betegség
Péter	21	Férfi	Kék	Tüdő
László	29	Férfi	Kék	Máj
Ferenc	60	Férfi	Kék	Szív

3.7.1. táblázat: A három kliens

ID	Kor	Nem	Szemszín	Elemszám	Aktív-e
1	20 - 30	Férfi	Kék	2	I
2	25 - 30	Nő	Kék	3	I
3	30 - 40	Férfi	Barna	3	I

3.7.2. táblázat: Az ekvivalencia-osztályok

ID	EO_ID	Betegség
1	2	Vese
2	2	Tüdő
3	3	Tüdő
4	2	Máj
5	3	Máj
6	3	Máj

3.7.3. táblázat: Az érzékeny adatokat tároló tábla

Tegyük fel, hogy Péter kezd elsőként kommunikálni a szerverrel, méghozzá 11:55-kor. Felküldi a kategorikus attribútumait, azaz, hogy férfi és kékszemű. Mivel egyetlen ilyen aktív ekvivalencia-osztály áll rendelkezésre a szerveroldalon, az 1-es azonosítójút rögtön vissza is kapja a kliens. Mivel az ekvivalencia-osztályba eddig még csak két kliens jelentkezett, a szerver *flag*-elve küldi el. Péter látja, hogy rá éppen illeszkedik ez az ekvivalencia-osztály, amit jelez is a szerver felé, viszont a *flag* miatt egyelőre eltárolja az adatait, és várja, hogy az aktuálisan üres központi táblában megjelenjen az 1-es azonosító. Amikor a szerver megkapja a választ, az ekvivalencia-osztály elemszáma már 3 lesz, így kiírja a központi táblába, hogy az 1-es ekvivalencia-osztályba tartozók küldjék az adataikat 12:00-kor.

EO_ID	Időpont
1	12:00

3.7.4. táblázat: A központi tábla a szerver és Péter kommunikációja után

Ha legalább két adat megérkezik az 1-es ekvivalencia-osztályba, azaz a 2-anonim garancia teljesül, a szerver leveszi az osztályról a *flag*-et. Így, ha László 12:02-kor fel szeretné küldeni az adatait és a kategorikus attribútumai miatt szintén az 1-es osztályt kapja válaszul, ő rögtön küldheti a szervernek az érzékeny információit. Az elemszám ekkor 4-re nő, így a szerver befagyasztja az 1-es ekvivalencia-osztályt és azt finomítva két újat állít elő. A finomításhoz az egyetlen intervallum-attribútumhoz tud nyúlni; a *[20, 30]* intervallumot két kisebbre bontva létrehozza az 4-es és 5-ös azonosítóval rendelkező osztályokat.

A harmadik kliensünk kommunikációja az előző kettőhöz hasonlóan indul, 12:04-kor. A *(Férfi, Kékszemű)* értékpárt felküldve visszakapja a két, újonnan létrehozott, 4-es és 5-ös azonosítójú ekvivalencia-osztályt. Ferencnek azonban a kora miatt ezek egyike sem megfelelő, így megpróbál másikat kérni a szervertől. Mivel az adott kategorikus attribútumérték-kombinációval nincs több ekvivalencia-osztály, a szerver megkéri

Ferencet, hogy generáljon magának egy sajátot, ami illeszkedik az adataira. Ehhez elküldi neki, hogy a szerveren a legnagyobb intervallumméret 10, tehát a saját korát erre alapozva generalizálja. Ferenc ekkor felküldi az *(55 - 65, Férfi, Kék)* ekvivalencia-osztályt, amely a 6-os azonosítót kapja, majd adatait eltárolva elkezdi figyelni a központi táblát.

12:05-kor, miután mindhárom kliens befejezte a kommunikációját a szerverrel, a központi tábla üres, a fenti táblázatok pedig az alábbiak szerint módosultak:

ID	Kor	Nem	Szemszín	Elemszám	Aktív-e
1	20 - 30	Férfi	Kék	4	N
…	…	…	…	…	…
4	20 - 24	Férfi	Kék	0	I
5	25 - 30	Férfi	Kék	0	I
6	55 - 65	Férfi	Kék	1	I

3.7.5. táblázat: Az ekvivalencia-osztályok állapota 12:05-kor

ID	EO_ID	Betegség
1	2	Vese
…	…	…
7	1	Szív
8	1	Szív
9	1	Tüdő
10	1	Máj

3.7.6. táblázat: Az érzékeny adatokat tároló tábla állapota 12:05-kor

3.8 Támadási modellek

A javasolt - kliens oldalon anonimizáló - módszer segítségével sikeresen kiküszöböltük a k-anonimitás eléréséhez jelenleg használt algoritmusok legnagyobb sérülékenységét, miszerint az anonimizálás megkezdéséhez a teljes eredeti adatbázisnak egy az egyben rendelkezésre kell állnia, s ezáltal az információk könnyen egy nemkívánatos harmadik személy kezébe kerülhetnek. Ennek ellenére azonban az általunk bemutatott mechanizmus sem nyújt 100%-os adatbiztonságot. Több lehetséges támadási modellt is kidolgoztunk, melyeket a következőkben mutatunk be.

3.8.1 Szerver oldalon tárolt adatbázis sérülékenységei

Elsőként azzal az esettel foglalkozunk, amikor az anonimizált adathalmaz semmilyen formában nem kerül publikálásra, csupán a szerver oldalon tárolódik.

Egy - kifejezetten a szerver ellen irányuló - támadás eredményeként megeshet, hogy a teljes adatbázis egy nemkívánatos harmadik fél kezébe kerül. Mivel a javasolt mechanizmus végső soron egy k-anonim adathalmazt állít elő, így ebben az esetben a sérülékenységek a k-anonimitásnál látottakkal egyeznek meg: Potenciálisan homogenitás-támadástól, illetve háttértudás-támadástól kell tartanunk. Szintén a k-anonimitás kapcsán, generalizáláskor lehet probléma a kiugró érték, mivel így az általánosítás után az intervallum egyik széle feltűnően alacsonyra, illetve magasra kerülhet. Ha pedig csupán egyetlen kiugró érték szerepel az adathalmazban, annak gazdája könnyen visszakereshető. Példaként tegyük fel, hogy egy adatbázis egy város lakóinak nettó keresetét rögzíti. Továbbá tegyük fel azt is, hogy köztudottan egyetlen olyan ember lakik a városban, akinek a nettó jövedelme meghaladja a 10 milliót. Ha az anonimizált adatok között találunk egy olyan (pl. $k = 20$-as) csoportot, ahol a generalizált nettó jövedelem érték *[900 ezer, 35 millió]*, akkor legalább annyi levezethető, hogy a leggazdagabb helybéli ebbe a csoportba tartozik, és az ő keresete 35 millió [20]. Ez utóbbi problémára a k-anonimitás és a Differential Privacy kombinálása megoldást nyújthat; erre a későbbiekben még visszatérünk.

3.8.1.1 Hálózati átvitel során előforduló támadások

Egy nagyobb problémakör az adatok hálózaton történő átviteléhez, az információcsere folyamatához kapcsolható.

Amennyiben a mechanizmus két résztvevője között a kommunikáció nem anonimizáltan zajlik, a szerver oldal könnyen olyan ismertetőjelek birtokába juthat az egyes klienseket illetően (például IP-cím), melyek segítségével akár be is tudja őket azonosítani. Ezen felül, ha a kliens-szerver interakció során nem ellenőrizzük, hogy a kliens valódi-e, illetve, hogy valódi adatokkal szolgál-e, előfordulhat az is, hogy az adatbázis fals információkkal telik meg.

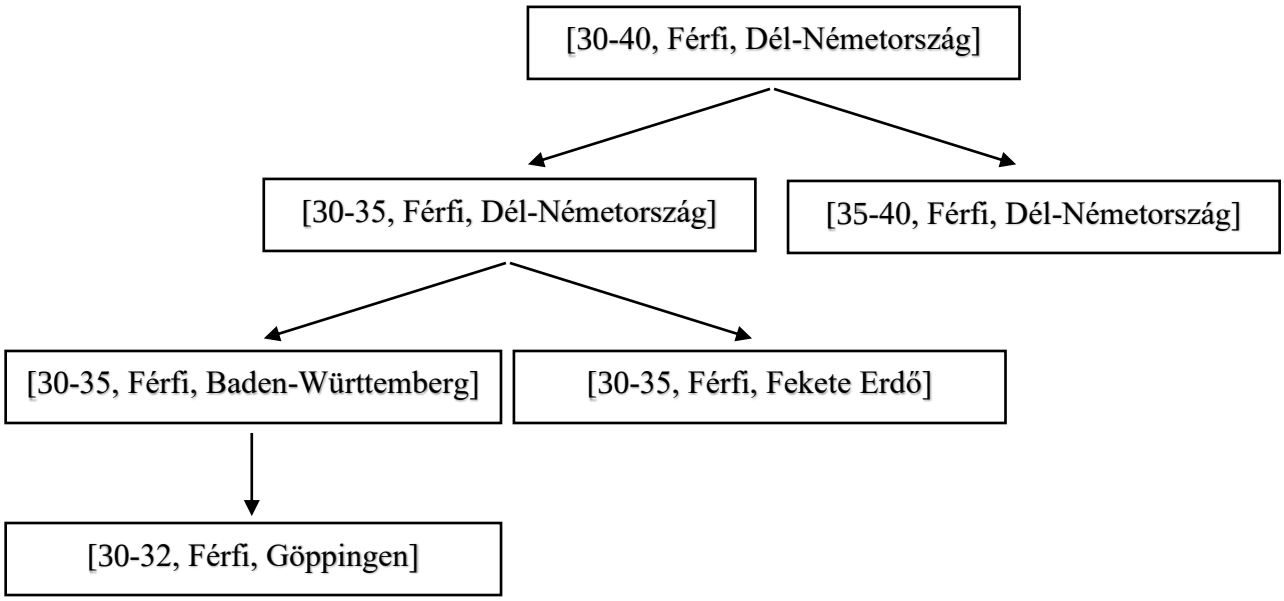
Ezen sebezhetőségek kiküszöbölhetők, ha maga az adatátvitel is anonimizált formában történik, illetve, ha a kommunikáció megkezdésekor a kliens átesik egy validációs folyamaton.

3.8.1.2 Az ekvivalencia-osztályok kezeléséből adódó támadások

Egy másik fontos problémakör az anonimizáló algoritmus ekvivalencia-osztály létrehozásához, kezeléséhez és megosztásához kapcsolódik.

Abban az esetben, amikor egy új kliens egyik létező ekvivalencia-osztályba sem sorolható be, a mechanizmus működése értelmében a szerver egy teljesen újat készíttet, melyben értelemszerűen ekkor még csupán egyetlen kliens adatai találhatók meg. Ugyan a szerver oldali központi táblázatba csakis akkor kerülnek be a szenzitív adatok, illetve az új ekvivalencia-osztály, mikor már a k-anonimitás feltételei teljesülnek, ha a köztes állapotban a szerver által megjelölt osztályokhoz mégis hozzáfér egy nemkívánatos harmadik fél, minimum annyi információ a birtokába juthat, hogy az adott kliens melyik ekvivalencia-osztályhoz fog tartozni, ezáltal pedig később - a k-anonim adathalmazból - már nagyobb bizonyossággal megismerheti a felhasználó érzékeny adatait.

Ugyancsak a módszer működéséből adódóan előfordulhat, hogy egyes ekvivalencia-osztályokban gyorsan elérjük a maximum kapacitást, melynek következményeképpen - ahogy az algoritmus leírásában megmutattuk - az osztályok finomítására van szükség. Ha sokat csiszolunk rajtuk, egy idő után túlzásba eshetünk, ami ugyan a hasznosságot növeli, de az adatbiztonság rovására mehet. Tegyük fel, hogy az ekvivalencia-osztályok a *[Kor, Nem, Lakhely]* formát veszik fel. Az alábbi ábrán látható, hogy a finomítások során a *Kor* attribútum esetén az intervallum szignifikánsan beszűkült, illetve a *Lakhely* attribútum értéke is jobban behatárolja az adott ekvivalencia-osztályba tartozó egyes felhasználók által lakott területet.



3.8.1. ábra: Az ekvivalencia-osztályok finomítása

További támadási felületet biztosít, hogy a kliens az összes létező ekvivalencia-osztályhoz hozzáférhet, illetve a szerver által tárolt központi táblázatot is elérheti, mely

által pedig például azt is láthatja, hogy egy-egy ekvivalencia-osztály milyen gyorsan telik fel.

3.8.2 Publikált adatbázis sérülékenységei

Abban az esetben is több sebezhetőséggel találkozhatunk, ha az adatbázist valamilyen módon publikálni is szeretnénk, s nem csupán a szerveren való tárolás a cél. Megfontolandó például, hogy milyen gyakran frissítsük a közzétett adatbázist. Ha egymás után, túl sűrűn aktualizálunk, fennáll a *man-in-the-middle* támadás lehetősége, ellenkező esetben viszont az adatbázis messze nem lesz naprakész, ami pedig a használhatóság rovására megy. További problémaként merülhet fel még az is, hogy miképpen tegyük elérhetővé az adatokat: Publikáljuk az egész táblázatot, csak aggregált formában adjuk ki az információkat, vagy esetleg hozzunk létre egy megfelelő API-t, melyen keresztül szabályozni tudjuk, hogy milyen lekérdezések engedélyezettek az adathalmaz irányába. A kérdésre adott válasz értelemszerűen az adatok felhasználásának módjától, illetve céljától függ.

4 Validáció és kiértékelés

Kérdéses, hogy a protokoll működése során milyen lesz az anonim adathalmaz minősége azon algoritmusokhoz képest, melyek az eredeti adathalmazon végzik az anonimizálást. Ennek felderítése érdekében kiemeltük a protokollból az anonimizálás folyamatát, ezt szimuláltuk, és az így kapott adathalmazon értékeltünk ki bizonyos metrikákat.

A k-anonimizáló algoritmusok esetén a már bevett *UCI Adult Dataset*-et használtuk fel. Eltávolítottuk a hiányos rekordokat, illetve a jövedelem oszlopot - ez mindössze annyi információt tárolt, hogy az adott illető 50 000-nél többet, vagy kevesebbet keres -, így egy 8 attribútumot tartalmazó, 30 000 sorból álló adathalmazt kaptunk. Az attribútumok mindegyike kvázi azonosító, érzékeny információt nem tartalmaznak, tehát közvetlenül felhasználhatók az ekvivalencia-osztályok előállítására.

Az összehasonlítás alapjául egy *open source Mondrian* implementáció szolgált [21] [22]. Ezt felparaméterezhettük különböző *k* értékekkel, illetve futtathattuk az adathalmaz különböző méretű szeleteire.

4.1 Metrikák

Adathalmaz anonimitásának mérésére számos metrikát definiáltak az elmúlt években. Köszönhetően a standardizált mérőszámok hiányának, illetve annak, hogy a napjainkban az anonimizálás aktívan kutatott téma, különböző algoritmusokhoz különböző metrikákat kezdtek el definiálni az emberek. Ugyan ezeket a [23] cikkben sikerült összegyűjteni, azonban még így sem egyszerű megtalálni a megfelelőt, a számunkra legbeszédesebbet. Az előbbi cikk 6 dimenziót említ, amelyek mentén érdemes vizsgálni az anonim adathalmazok minőségét. Ezek közül mi az információveszteséget mérő metrikákra koncentráltunk, hogy megnézzük, mennyiben számít a teljes, eredeti adathalmaz hiánya.

Komplexitásban és kifejezőerőben számottevő a szórás. A legegyszerűbbek közt az ekvivalencia-osztályok méretére alapozó mérőszámok szerepelnek. Előnyük, hogy gyorsan és könnyen számíthatók, viszont az adathalmaz minőségéről csupán elnagyolt képpel tudnak szolgálni. Ilyen például az adathalmazban szereplő személyek felfedésének

legmagasabb kockázatát mérő szám. Ez k-anonimitás esetén *1 / k*, vagy annál kisebb szám kell, hogy legyen [23]:

$$\theta_{max} = \frac{1}{\min_j |EO_j|}$$

A következő, az ekvivalencia-osztályok átlagos méretére alapozó metrika esetén a $|T|$ az eredeti adathalmaz mérete, az $|EOk|$ az ekvivalencia-osztályok száma, *k* pedig az anonimizáláshoz használt paraméter [24]. Célunk a C_{avg} minimalizálása, ahol a minimum az 1. Ezt akkor érjük el, ha minden ekvivalencia-osztály éppen *k* méretű. Az ekvivalencia-osztályok méretével együtt a mérőszámunk is elkezd növekedni. Ez pedig nem jelent mást, minthogy az adathalmazunkat túlanonimizáljuk, azaz az előírt garanciákon felüli anonimitást biztosítunk. Ezzel együtt pedig felesleges információvesztés, és *utility*-csökkenés is történik.

$$C_{avg}(T') = \frac{|T|}{|EOk| * k}$$

Sok mérőszám kalkulál a teljesen elnyomott rekordok számával. Erre épít a megkülönböztethetőségi metrikák (*Discernability Metric*) egyike is, amely az ekvivalencia-osztályok méretének és az elnyomott rekordok számának függvényében határoz meg egy globális „büntetést” [24]:

$$DM(T') = \sum_{\forall EO:|EO| \geq k} |EO|^2 + \sum_{\forall EO:|EO| < k} |T| * |EO|$$

Az információvesztés mérésére szolgáló, valamivel komplexebb metrika a *Global Certainty Penalty (GCP)*. A *Normalized Certainty Penalty (NCP)* minden ekvivalencia-osztályhoz rendel egy büntetést annak függvényében, hogy az mennyire generalizálja az egyes attribútumértékeket. Ennek megfelelően az intervallumváltozók esetén az intervallumok nagysága, hierarchiaváltozók esetén pedig a rejtett értékek száma szerint fog büntetni. A *Global Certainty Penalty* ezeket az *NCP*-ket összegzi a teljes anonim adathalmazra, majd normalizálja. A mérőszám nagy előnye, hogy a normalizálásnak köszönthetően különböző kardinalitású és dimenzionalitású táblák összehasonlítását is megengedi, emellett pedig a *[0; 1]* értékkészlet könnyen értelmezhetővé teszi. A cél nyilvánvalóan az információvesztés, így a metrika minimalizálása [25]:

$$GCP(T') = \frac{\sum_{\forall EO} |EO| * NCP(EO)}{d * N}$$

$$NCP(EO) = \sum_{k=1}^{d} w_i * NCP_{A_i}(EO)$$

$$NCP_{A_{num}}(EO) = \frac{max_{A_{num}}^{EO} - min_{A_{num}}^{EO}}{max_{A_{num}} - min_{A_{num}}}$$

$$NCP_{A_{cat}}(EO) = \begin{cases} 0, & card(u) = 1 \\ \frac{card(u)}{|A_{cat}|}, & egyébként \end{cases}$$

4.2 A szimuláció és a Mondrian-implementáció

Az anonimizálás szimulációját *Pythonban* valósítottuk meg. Ennek folyamán bejárjuk az eredeti *UCI* adathalmazt, mintha az adatok folyamatosan, egyesével érkeznének, és a nálunk lévő, már anonim adathalmaz alapján kezeljük le őket. A korábbi protokollban leírtak szerint építünk egy adathalmazt az aktív, illetve egyet a lezárt ekvivalencia-osztályok számára. Amennyiben egy aktív ekvivalencia-osztály eléri a *k* elemet, akkor finomítjuk egy intervallum- vagy hierarchiaváltozó mentén, a belőle újonnan generáltakat pedig elhelyezzük az aktív ekvivalencia-osztályok között, míg az eredetit a lezártak közé tesszük át. Ezt futtattuk le többször, különböző szcenáriókban. A szimulációt elindítottuk úgy, hogy semmilyen kezdeti ekvivalencia-osztályt nem definiáltunk, hagytuk, hogy ezeket az elejétől fogva a „kliensek" generálják le. A másik esetben valamilyen logika mentén mi gyártottunk le előre ekvivalencia-osztályokat. Ezt a két esetet hasonlítottuk össze a *Mondrian*-algoritmus kimenetével, az eredeti adathalmazt különböző mérettel mintavételezve, illetve a teljes adathalmazra *k = 5, 10* és *20* esetén.

A szimuláció kimenete tehát a lezárt és az aktív ekvivalencia-osztályokat tartalmazó adathalmazok. A lezártban szereplő adatok esetén küldhették volna fel a kliensek biztonságban az érzékeny adataikat, ekkor teljesül a k-anonimitás feltétele. Ezzel ellentétben az aktív adathalmazba tartozó kliensek a szimuláció végén még helyben tárolták volna az érzékeny adatokat, és folyamatosan figyelték volna a központi táblát. Mivel a protokoll működése során folyamatosan érkező adatokat feltételezünk, a használt, relatíve kis méretű adathalmazra futtatva a szimulációt az aktív ekvivalencia-osztályok száma jellemzően elég magas maradt. A metrikák számítása során csak a lezárt ekvivalencia-osztályokra végeztük el a kalkulációkat.

Mit jelent mindez a metrikáink szempontjából?

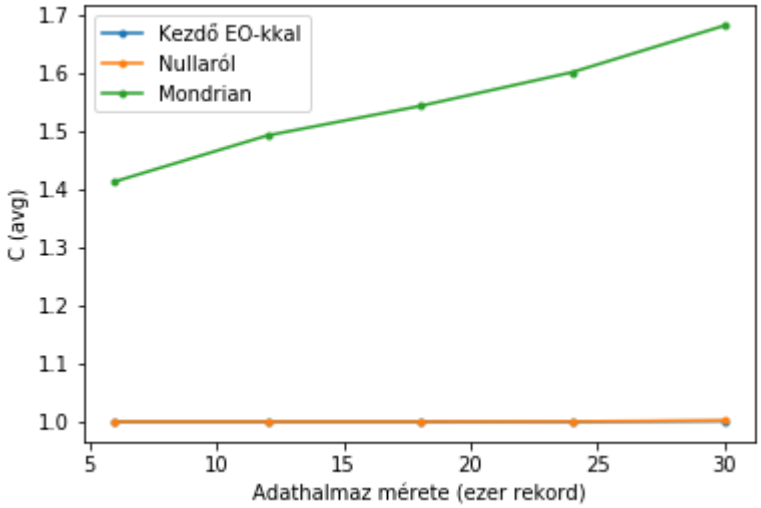
- θ_{max} : Mivel az anonimizálás során mi minden ekvivalencia-osztályt k eleműre töltünk, az adathalmazunk maximális kockázata minden esetben a legkisebb adathalmazunk mérete szerint *1 / k* lesz.
- $C_{avg}(T')$: Egy ekvivalencia-osztály mérete csupán abban az esetben kezdhet el nőni, ha azt minden attribútuma mentén maximálisan finomítottunk, ellenkező esetekben az ekvivalencia-osztályok mérete a szimulációban pontosan *k*, gyakorlatban $k + \varepsilon$. Így számíthatunk arra, hogy a C_{avg} egy 1-hez nagyon közeli szám lesz. Fontos kiemelni azonban, hogy ez a mi eljárásunkra ellentétesen fog működni, mint várnánk. Ugyanis, míg eleinte ez egy konstans, 1-től alig nagyobb szám lesz, úgy az anonimizált adatok *utility*-jének növekedésével elkezdhet nőni. Ennek oka éppen a maximális *utility* elérése, azaz egy ekvivalencia-osztály maximális finomítása. Ebben az esetben már nem tudunk újabb, több információt hordozó osztályokat előállítani, hanem az eredetit fogjuk tovább növelni, aminek így *k* fölé fog nőni az elemszáma, ezzel folyamatosan növelve a metrikánk értékét is.
- $DM(T')$: Adatot soha nem nyomunk el teljes mértékben, csupán eltároljuk a kliensen, aki kivárja az információira illeszkedő ekvivalencia-osztály feltöltődését. Az erre vonatkozó metrikáknak tehát itt kevéssé vesszük hasznát. Számolhatunk az elnyomott érték helyett az aktív ekvivalencia-osztályokban ragadt adatpontok számával. Be kell látnunk azonban, hogy ezek az óriási büntetés értékek csak kis mértékben használhatók. Csupán arra képesek rávilágítani, hogy melyik esetben jobb az aktív - lezárt ekvivalencia-osztályok aránya, ehhez viszont felesleges e számolást végigvezetni.

A *Mondrian*-algoritmus esetén a maximális kockázat ugyancsak *1 / k* körüli érték lesz, az elvártak szerint működik és beszédesebb a C_{avg} érték is. Az elnyomásnak ebben az esetben sem vesszük nagy hasznát, az implementáció ugyanis minden értéket elhelyez valamilyen ekvivalencia-osztályban. Mind a szimuláció, mind pedig a *Mondrian* esetén a *GCP* bizonyult igazán beszédes metrikának.

4.3 Az eredmények

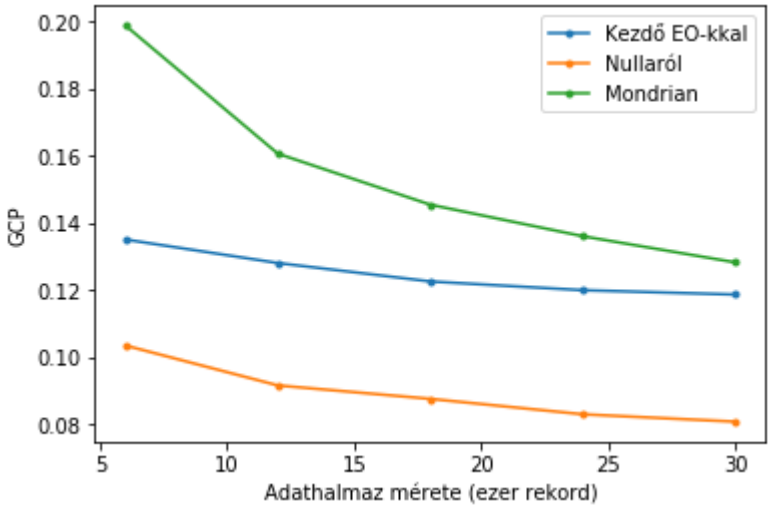
A fejezetben elsőként az ekvivalencia-osztályok méretének alakulását vizsgálatuk a C_{avg} metrika segítségével. A protokoll működéséből adódik, hogy e szám valóban egy

1-hez nagyon közeli értéket vesz fel minden esetben. Az adathalmaz méretének növekedésével minimális növekedést megfigyelhetünk, amely - az előzőeket megismételve - annak köszönhető, hogy abban a néhány maximálisra finomított ekvivalencia-osztályban az elemszám elkezd növekedni. Ezzel szemben a *Mondrian* egyre nagyobb adathalmazokra futtatása során bekövetkezett C_{avg} növekedés éppen *utility*-csökkenést fog jelenteni a metrika definíciójának megfelelően. Ennek oka, hogy egyre több a *k*-nál nagyobb méretű ekvivalencia-osztály, amely túlanonimizálással és információvesztéssel jár.



4.3.1. táblázat: A C_{avg} értékének változása az adathalmaz méretének növekedésével

Mindezek után az eredeti adathalmazt hasonlóan mintavételezve, majd a teljes adathalmazra különböző *k* értékek mellett számoltuk ki a *GCP* értékeket.



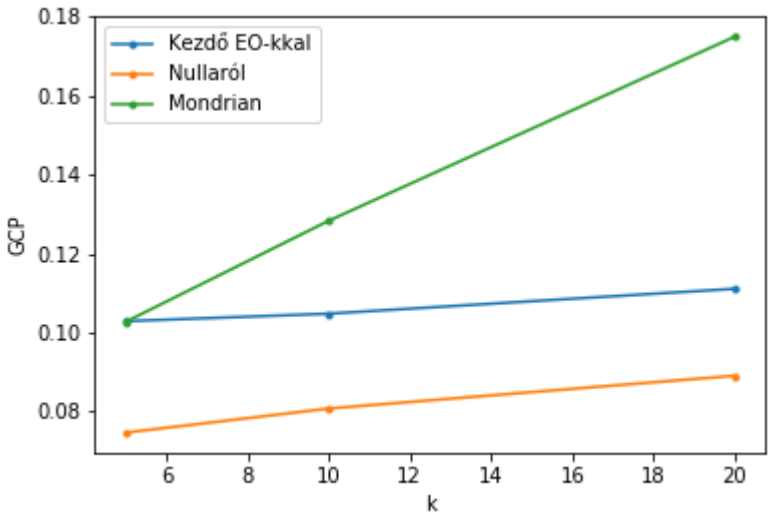
4.3.2. táblázat: A *GCP* értékének változása az adathalmaz méretének növekedésével

Az eredmények azt mutatják, hogy az adathalmaz növekedésével az adatvesztés folyamatosan csökken, azaz egyre jobb minőségűek lesznek az ekvivalencia-osztályaink. Láthatjuk, hogy a szimuláció jobban teljesít a *Mondrian*-algoritmusnál. Tudjuk, hogy a saját anonimizálási módszerünk kezdetekor az ekvivalencia-osztályok minősége lényegesen rosszabb, amit azonban kompenzálni tud azzal, hogy szigorúan tartja magát a *k* elemszám utáni finomításhoz, melynek köszönhetően egészen gyorsan konvergálhat jelentősen jobb *utility*-vel rendelkező ekvivalencia-osztályok felé.

A kétféle szimuláció szerint a jobb eredményt akkor kapjuk, ha teljesen nulláról indulunk, azaz semmilyen ekvivalencia-osztályt nem definiálunk előre. Ekkor ezek teljes mértékben az érkező kliensadatok szerint alakulnak ki, várhatóan magasabb információtartalommal. Ez a megközelítés azonban a gyakorlatban korlátokba ütközhet, a kezdeti adatbázisépítés, az új ekvivalencia-osztályok generálása során megnövekedett mértékű kommunikáció, illetve az új osztályok validációja túl nagy terhelést jelenthet a szerver számára.

Jóllehet a szimuláció kimenetén csak a lezárt ekvivalencia-osztályokon végezzük a méréseket, viszont az aktív ekvivalencia-osztályok nagy része már finomított, így a fenti számokon csak tovább javítana. Ugyanakkor ez rámutat arra is, hogy miért nem használhatjuk mégsem a *Mondrian* helyett is az általunk bemutatott anonimizálást. A *Mondrian* valamivel több információvesztéssel jár, de egyetlen rekordot sem nyom el, ebből adódó adatvesztés tehát egyáltalán nincs. Ezzel szemben a mi eljárásunk egy teljes, kéznél lévő adathalmazra futtatva csak annak a töredékét, például a teljes *Adult Dataset* esetén a rekordok számának csupán 65%-át képes elhelyezni lezárt ekvivalencia-osztályokban. Ez a *Mondrian* 0%-os elnyomásához képest 35%-ot jelentene, ami nyilván nem megengedhető.

Végezetül különböző *k* értékek esetén vizsgáltuk a *GCP* értékét. Nem meglepő módon a paraméter növelésével az információveszteség is elkezd növekedni, hiszen egyre elnagyoltabb ekvivalencia-osztályokkal dolgozunk. Érdekes azonban, hogy a *Mondrian* sokkal érzékenyebben reagál az ekvivalencia-osztály méretek növekedésére.



4.3.3. táblázat: A *GCP* értékének változása a *k* paraméter növelésével

Összességében megállapíthatjuk, hogy az anonimizáló algoritmusunk valóban konvergál a magas *utility*-vel rendelkező ekvivalencia-osztályok irányába, ezzel az információvesztést pedig elég alacsonyan tudjuk tartani. Szükség van azonban a folyamatosan érkező, nagy mennyiségű adatokra, hogy ezt a konvergenciát minél jobban fel tudjuk gyorsítani, és a klienseink mihamarabb küldhessék a szervernek az érzékeny adataikat.

5 A Differential Privacy és a k-anonimitás találkozása

Ahogy azt már a k-anonimitást bemutató részben is beláttuk, a harmadik fejezetben javasolt - kliens oldalon anonimizáló, a k-anonimitáson alapuló - módszer sem nyújt 100%-os adatbiztonságot. A következőkben az általunk ismertetett algoritmus sebezhetőségei közül kifejezetten csak a k-anonimitás kapcsán felmerülő támadásokkal foglalkozunk, a szerver-kliens közötti kommunikáció biztonságossá tételére, illetve az ekvivalencia-osztályok megfelelő kezelésére most nem térünk ki.

Ebben a fejezetben arra a kérdésre keressük a választ, hogy vajon a k-anonimitás kombinálható-e a Differential Privacy modellel, illetve, amennyiben igen, kiküszöbölhetők-e ezáltal a k-anonimitás tipikus sérülékenységei. Szemléltetésképpen a korábbiakban javasolt algoritmuson és az annak alkalmazásával kapott k-anonim adathalmazon végzünk különböző módosításokat, melyeknek segítségével a Differential Privacy modell által támasztott követelményeknek is szeretnénk megfelelni.

5.1 A módszerek társítása az irodalomban

A k-anonimitás és a Differential Privacy legtöbbször, mint egymástól távol álló koncepciók tűnnek fel különböző tudományos munkákban. Míg a k-anonimitás szintaktikus, a Differential Privacy inkább szemantikus módon biztosítja az adatok védelmét. Mindezek ellenére azonban - ahogy azt a *Purdue* Egyetemen lefolytatott kutatás [20] is megmutatta - létezik a gyakorlatban is használható összekötő kapocs a két mechanizmus között. Ahhoz, hogy ezt az összefüggést ismertethessük, először szükség van a közelítő $(\mathcal{E}, \delta)$-Differential Privacy, a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy, illetve a biztonságos k-anonimizálás fogalmak bevezetésére.

- Egy véletlenszerűsített A algoritmus minden t-re kielégíti a közelítő $(\mathcal{E}, \delta)$-Differential Privacy követelményeit, amennyiben minden két szomszédos adatbázis (D és D') esetén teljesül a következő egyenlőtlenség [20]:

$$\mathrm{P}(A(D) = t) \leq \exp(\mathcal{E}) * \mathrm{P}(A(D') = t) + \delta$$

- Egy A algoritmus akkor, és csak akkor elégíti ki a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy követelményeit, amennyiben $\beta > \delta$ és az A^{β} algoritmus kielégíti az $(\mathcal{E}, \delta)$-Differential Privacy követelményeit, ahol A^{β} egy olyan módszer, mely előbb β valószínűséggel mintavételez (minden rekordot β

valószínűséggel tartalmaz a bemeneti adatbázisból), majd pedig a mintákból álló adathalmazra alkalmazza *A*-t [20].

- Egy k-anonimizáló algoritmus biztonságos, ha olyan leképzést használ, amely nem függ túlságosan egy-egy adott adatsortól - tehát azok az eredmények, melyek túl érzékenyek egy rekord megváltozására, eltávolításra kell kerüljenek. Mint azt már a második fejezetben is láthattuk, egy k-anonimizáló *A* algoritmus bemenete egy *D* adathalmaz és egy *k* érték, kimenete pedig egy *S = A(D)* adathalmaz. A biztonságos k-anonimizáló algoritmus formális definíciójához minden *A* anonimizáló algoritmust két lépésben kell specifikálnunk: Elsőként A_m egy *g: D → T mapping*-függvényt bocsát ki, ahol *T* az összes lehetséges adatsor halmaza. A második lépés a *D* összes adatsorára alkalmazza *g*-t, azaz *A(D, k) = Apply*(A_m*(D, k), D, k)*, ahol *Apply* a következőképpen működik [20]:

Apply(*g, D, k*)
S ← *0*
for *(minden t ∈ D-re)*
    S ← *S* ∪ *g(t)*
end for
for *(minden s ∈ S-re)*
    if *(s kevesebb, mint k-szor jelenik meg S-ben)*
      s összes megjelenését eltávolítani S-ből
    end if
end for
return *S*

A fenti definíciók felhasználásával az alábbiakban a gyakorlatban is bevezetjük a k-anonimitás és a Differential Privacy közötti összefüggést. Megmutatjuk, hogy az irodalomból ismert elméleti modell [20], mely szerint a biztonságosan végrehajtott k-anonimizálás minden $\delta < 1$-re kielégíti a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy követelményeit, a gyakorlatban is helytálló. Célunk tehát demonstrálni, hogy amennyiben egy biztonságos k-anonimizáló algoritmus végrehajtását megelőzi egy véletlenszerű mintavételező lépés, akkor az algoritmus eleget tesz az $(\mathcal{E}, \delta)$-Differential Privacy kritériumainak. Ez az eredmény pedig akár egy alternatív megközelítést is kínálhat arra

vonatkozóan, hogy miként feleljünk meg a Differential Privacy modell által támasztott elvárásoknak a kimenetek manipulálása nélkül. Fontos megjegyezni még azt is, hogy a mintavételezés β valószínűségének csökkentésével az adatvédelem mértéke növelhető; ezzel a későbbiek folyamán még foglalkozunk.

5.1.1 Erősen biztonságos k-anonimizálás

Mint azt már az előző részekben is láttuk, sem a k-anonimitás, sem pedig a jelenleg létező k-anonimizálási algoritmusok nem nyújtanak megfelelő szintű adatvédelmet. Ahhoz, hogy a különböző sérülékenységeket kiküszöbölhessék, tudományos munkájukban *N. Li, W. Qardaji* és *D. Su* lépésenként egyre erősebb feltételeket fogalmaztak meg a k-anonimitás használatát illetően [20].

A korábban megismert biztonságos k-anonimizálási algoritmus alkalmazásával gátat szabhatunk a generalizálás utáni kiugró értékek problémájának (*lásd 3.3 fejezet*); az erősen biztonságos k-anonimizálás pedig még intenzívebb védelmet garantál. Formálisan megfogalmazva egy *A* k-anonimizáló algoritmus akkor és csak akkor erősen biztonságos, amennyiben az $A_m(D, k)$ függvény a *D* megváltozása esetén is konstans marad, azaz a *g* leképzés nem függ a bemeneti adathalmaztól. Ez azt jelenti, hogy ha egy személy adatsora publikálásra kerül, akkor a bemeneti adatbázisban léteznie kell még minimum $(k-1)$ másik olyan rekordnak, amelyek a felvétel sémájában megegyeznek a nyilvánosságra hozott adatsorral. Továbbá a felvételi séma (*recording scheme*) nem függ az adathalmaztól és csupán a felvétel eredménye látható. Így a bemeneti adatbázisban minden egyén egy legalább *k* méretű tömegben rejtőzik. Értelemszerűen az adatvédelem szintje a *k* értékével arányosan növekszik. Mindezek ellenére bizonyított, hogy az erősen biztonságos k-anonimizálási algoritmusok semmilyen $\delta < 1$ esetén sem elégítik ki az $(\mathcal{E}, \delta)$-Differential Privacy elvárásait, viszont kis δ mellett a *k* és a β paramétereknek már megadhatók olyan értékek, melyekre a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy feltételei teljesülnek.

Amennyiben az $f(j; n, \beta)$ jelölést használjuk a binomiális eloszlás valószínűségi súlyfüggvényére és az $F(j; n, \beta)$ jelölést a halmozott valószínűségi súlyfüggvényre, a következő tétel fogalmazható meg: Bármely erősen biztonságos k-anonimizáló algoritmus kielégíti a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy követelményeit minden $0 < \beta < 1, \mathcal{E} \geq -\ln(1 - \beta)$ és $\delta = d(k, \beta, \mathcal{E})$ esetén, ahol a *d* függvény definíciója az alábbi [20]:

$$d(k,\beta,\mathcal{E}) = \max_{n:n \geq \left\lceil \frac{k}{\gamma}-1 \right\rceil} \sum_{j>\gamma n}^{n} f(j;n,\beta), \text{ahol } \gamma = \frac{(e^{\varepsilon}-1+\beta)}{e^{\varepsilon}}$$

A binomiális eloszlás valószínűségi súlyfüggvénye *f(j; n, β)* azt a valószínűséget fejezi ki, miszerint *n* próbálkozásból - ahol minden próbálkozás *β* valószínűséggel lesz sikeres - pontosan *j* darab lesz eredményes. A halmozott valószínűségi súlyfüggvény pedig az $F(j;\ n,\beta)\ = \sum_{i=0}^{j} f(i;\ n,\beta)$ képlettel írható le.

Fontos megjegyezni, hogy a tételben szereplő négy paraméter közül $\mathcal{E}$ és δ definiálja az adatvédelem szintjét, *k* és *β* pedig az anonimizált adat minőségét befolyásolja. Ahhoz, hogy a négy paraméter közötti összefüggést megvizsgálhassuk, a *d* függvénnyel kell foglalkoznunk. Elsőként figyeljük meg, hogy $\gamma > \beta$, mivel $\gamma - \beta = \frac{(e^{\varepsilon}-1+\beta)}{e^{\varepsilon}} - \beta = \frac{(e^{\varepsilon}-1)(1-\beta)}{e^{\varepsilon}} > 0$. Ezáltal a *d* függvényben megjelenő szumma a binomiális eloszlás valószínűségek azon részeit adja össze, melyek γn-nél nagyobbak. A nagy számok törvénye alapján arra következtethetünk, hogy minél nagyobb az *n* értéke, annál kisebb lesz e rész valószínűsége. Így tehát a lehető legkisebb *n*-t választva $(n = \left\lceil \frac{k}{\gamma} - 1 \right\rceil)$ maximalizálhatjuk a *d* függvényben szereplő formulát.

5.1.2 Ɛ-biztonságos k-anonimizálás

A gyakorlatban az erősen biztonságos k-anonimizáló algoritmusok kimenete általában kevéssé használható. Ennek elkerülése érdekében a követelmények legtöbbször nem ennyire szigorúak: A generalizáló séma függhet a bemeneti adathalmaztól, viszont nem függhet túlságosan egyetlen adott rekordtól sem. Formálisan kifejezve, egy *A* k-anonimizáló algoritmus akkor és csak akkor $\mathcal{E}$-biztonságos, ha az A_m függvény kielégíti az $\mathcal{E}$-Differential Privacy kritériumait. Szintén bizonyított tétel, hogy az $\mathcal{E}$-biztonságos k-anonimizálás megfelel a mintavételező *(β, $\mathcal{E}$, δ)*-Differential Privacy által támasztott elvárásoknak: Bármely $\mathcal{E}_1$-biztonságos k-anonimizáló algoritmus kielégíti a mintavételezéses *(β, $\mathcal{E}$, δ)*-Differential Privacy követelményeit, ahol $\mathcal{E} \geq -\ln(1-\beta) + \mathcal{E}_1$ és a *d* függvény definíciója az alábbi [20]:

$$d(k,\beta,\mathcal{E}) = \max_{n:n \geq \left\lceil \frac{k}{\gamma}-1 \right\rceil} \sum_{j>\gamma n}^{n} f(j;n,\beta) = \delta, \text{ahol } \gamma = \frac{(e^{\varepsilon-\varepsilon_1}-1+\beta)}{e^{\varepsilon-\varepsilon_1}}$$

5.1.3 Eredmények

Az aktuális alfejezetben tehát elméletben ismertettük, hogy amennyiben egy biztonságos k-anonimizáló algoritmus végrehajtását megelőzi egy véletlenszerű mintavételező lépés, megadhatók olyan paraméter-értékek, melyek mellett a közelítő (ε, δ)-Differential Privacy feltételei teljesülnek. Ezzel egy alternatív megoldást is találtunk az (ε, δ)-Differential Privacy által támasztott követelmények kielégítésére. A legtöbb jelenlegi módszer zajt ad az adatokhoz a globális vagy lokális érzékenység alapján; a k-anonimitás és a Differential Privacy modell összekapcsolásával viszont más opció is szóba jöhet: A kimenet torzítása helyett először hajtsunk végre véletlenszerű mintavételezést, a mintákból álló adatbázison dolgozzunk tovább, majd pedig az eredményhalmazból tüntessük el azon részleteket, melyek túl érzékenyek egy-egy adatrekord megváltozására.

Mindent összevetve beláthatjuk, hogy a k-anonimitás és a Differential Privacy közötti kapcsolatot alapvetően a mintavételezés biztosítja. Ezen állítást a harmadik fejezetben bemutatott algoritmus segítségével, az alábbiakban egy gyakorlati példán keresztül bizonyítjuk be. Tudomásunk szerint az irodalomban még nem volt precedens arra, hogy egy mechanizmus egyszerre feleljen meg a k-anonimitás, illetve a Differential Privacy által elvárt feltételeknek is.

5.2 Módosítások a javasolt algoritmuson és annak kimeneti adathalmazán

Ahhoz, hogy az előzőekben vázolt felvetéseket szemléltetni tudjuk, ebben az alfejezetben megmutatjuk, milyen változtatásokat kell elvégezni, hogy az általunk javasolt - kliens oldalon anonimizáló - módszer alkalmazásával a k-anonimitás fenntartása mellett a Differential Privacy modell által támasztott követelményeket is teljesítsük. A kiindulási alap minden esetben a nyers, még anonimizálatlan adat.

5.2.1 Első lépés: Mintavételezés

Mint ahogyan azt a korábbi szekciókban is tárgyaltuk, a k-anonimitás és a Differential Privacy közötti kapcsolat megteremtéséhez elsőként a k-anonimizáló algoritmus végrehajtását meg kell, hogy előzze egy véletlenszerű mintavételező lépés. Tegyük fel, hogy a nulladik pillanatban az alábbi - még anonimizálatlan - adathalmaz áll a rendelkezésünkre.

Név	Nem	Nettó jövedelem	Választott párt
Kovács János	Férfi	350 000	A
Kis Ernő	Férfi	670 000	A
Tóth Piros	Nő	520 000	C
Németh Mária	Nő	180 000	A
Nagy András	Férfi	880 000	B
Gál József	Férfi	1 740 000	B

5.2.1. táblázat: Kliensek adatai (eredeti adatbázis)

Legyen $\beta \leq 1$ a mintavételezési valószínűség. A szemléletesség kedvéért dolgozzunk most a $\beta = 0{,}7$ értékkel. Ez azt jelenti, hogy az eredeti adatbázisból minden egyes adatsor 70%-os valószínűséggel kerül be a minták közé. Így a mintavételezés végeztével az adathalmazunk például a következőképpen is alakulhat:

Név	Nem	Nettó jövedelem	Választott párt
Kovács János	Férfi	350 000	A
Tóth Piros	Nő	520 000	C
Németh Mária	Nő	180 000	A
Gál József	Férfi	1 740 000	B

5.2.2. táblázat: Kliensek adatai (mintavételezett adatbázis)

Annak érdekében, hogy végül mind a k-anonimitás, mind pedig a Differential Privacy modell által előírt feltételek teljesülhessenek, a k-anonimizáló mechanizmus bemenete a mintákból álló adatbázis kell, hogy legyen.

A mintavételezési valószínűség (β) értékének változtatásával a hasznosság és az adatvédelem mértéke közötti arány is módosítható. Értelemszerűen minél kisebb valószínűséggel operálunk, annál nagyobb az adatvédelem foka, mivel annál kevesebb információt használunk fel egyáltalán. Ez viszont az ún. *utility* rovására megy; az eredetihez képest igen csekély számú mintából álló adathalmazban kevéssé figyelhetők meg a trendek, illetve következtetéseket is nehezebben tudunk belőle levonni.

5.2.1.1 Folyamatosan érkező adatok

Az általunk ajánlott algoritmus előfeltételeiből és működéséből adódóan nem fordulhat elő a fent bemutatott szituáció, amikor egy nagyobb - még nyers adatokból álló - adathalmaz érhető el az eljárás alkalmazását megelőzően. A gyakorlatban a kliensek egymástól függetlenül, eltérő időpontokban kezdeményezik a kommunikációt a szerverrel, így általában folyamatosan érkező adatokkal kell dolgoznunk.

A mintavételezés menete ebben az esetben is a korábbiakban látottakhoz hasonló: Legyen $\beta \leq 1$ újfent a mintavételezési valószínűség, a mostani példában pedig használjuk ugyancsak a $\beta = 0{,}7$ értéket. Tegyük fel, hogy a jelenlegi időpillanatban egyetlen kliens szándékozik információt küldeni magáról, és e kliens az alábbi adatokkal rendelkezik:

Név	Nem	Nettó jövedelem	Választott párt
Nagy Béla	Férfi	710 000	C

Ekkor a mintavételezés értelmében 70% annak a valószínűsége, hogy a fenti adatsort megtartjuk, a minták közé beválogatjuk, s a harmadik fejezetben látott protokoll felhasználásával, kliens oldalon anonimizálva a szerveren eltároljuk. Ellenkező esetben a teljes adatsor eldobásra kerül, további teendő nincs vele. Ennek a valószínűsége a példa alapján a β értékét figyelembe véve 30%.

A mintavételező lépést az összes olyan kliensen alkalmazzuk, mely a szerverrel kommunikációt kezdeményez. Így, ha kliens oldalon elvégezzük a mintavételezést, az megfelel annak, mintha az egész adathalmaz birtokában soronként eszközölnénk azt.

5.2.2 Második lépés: Kiugró értékek eltávolítása

Második lépésben célunk a mintákból álló adathalmazra felhasznált k-anonimizáló algoritmus biztonságossá alakítása. Ehhez el kell távolítanunk azon eredményeket, melyek túl érzékenyek egy-egy adott rekord megváltozására.

Vezessük végig egy példán keresztül hogyan is zajlik a protokoll, illetve nézzük meg, hogy alapvetően biztonságos-e a módszer, és ha nem, milyen módosításokkal tehetjük mégis biztonságossá. Vegyük a klienseket az előző pontban létrejött, négy adatsorból álló mintavételezett adatbázisból *(5.2.2 táblázat)*, s kövessük a kommunikációjukat a szerverrel. Annak, hogy a kliensek egyidejűleg, vagy eltérő időpontokban küldenek magukról információkat, ebben a lépésben nincs jelentősége.

A harmadik fejezetben bemutatott mechanizmus alkalmazása előtt, teljesen üres adatbázis esetén szükség van valamilyen logika alapján szerver oldalon kezdeti ekvivalencia-osztályok generálására. Intervallum-attribútumoknál a protokoll felkonfigurálásakor a legnagyobb intervallum méretét is meg kell adnunk. Természetesen ez az ekvivalencia-osztályok finomítása során később változhat.

Tegyük fel, hogy a szerveren jelenleg az *5.2.3 táblázatban* látható ekvivalencia-osztályok léteznek, a legtágabb intervallum mérete pedig 200 000. A kimeneti adatbázis 3-anonim.

ID	Nem	Nettó jövedelem	Elemszám	Aktív-e
1	Nő	100 000 - 300 000	3	I
2	Férfi	300 000 - 500 000	4	I
3	Férfi	500 000 - 700 000	3	I

5.2.3. táblázat: Ekvivalencia-osztályok a szerveren (kommunikáció előtt)

Ilyen esetben a fent említett négy kliens közül kettő azonnal talál megfelelően illeszkedő ekvivalencia-osztályt. Az 520 ezer nettó jövedelmű hölgy számára egyik létező ekvivalencia-osztály sem felel meg, így a szerver arra kéri őt, hogy generáljon magának egy újat. Ehhez segítségképpen elküldi neki, hogy a szerveren a legnagyobb intervallum mérete 200 000, a nettó bért tehát ez alapján kell általánosítani. Az új ekvivalencia-osztály ebből adódóan például a következőképpen is alakulhat: *[Nő, 500 000 - 700 000]*. A milliós jövedelemmel rendelkező férfi esete hasonló: Az általa generált új ekvivalencia-osztály például *[Férfi, 1 600 000 - 1 800 000]* lehet. A kommunikáció végeztével mindezek után a szerveren az alábbi ekvivalencia-osztályok lesznek tehát megtalálhatók:

ID	Nem	Nettó jövedelem	Elemszám	Aktív-e
1	Nő	100 000 - 300 000	4	I
2	Férfi	300 000 - 500 000	5	I
3	Férfi	500 000 - 700 000	3	I
4	Nő	500 000 - 700 000	1	I
5	Férfi	1 600 000 - 1 800 000	1	I

5.2.4. táblázat: Ekvivalencia-osztályok a szerveren (kommunikáció után)

Vizsgáljuk meg ezen ekvivalencia-osztályokat; mikor merülhetne fel az ún. kiugró érték probléma? Az egyszerű k-anonimitás értelmében megtehetnénk, hogy a milliós nettó bérű férfi információcserébe való bekapcsolódásakor a 3-as azonosítójú ekvivalencia-osztályt kiterjesztjük, s a nettó jövedelem intervallumát például *500 000 - 1 740 000*-re változtatjuk. Ebben az esetben az új kliens az érzékeny adatainak megadásával be is kerülhetne a központi k-anonim táblába. Ha azonban egy potenciális támadó rendelkezik legalább annyi háttérinformációval, hogy az adott választókerületben csupán egyetlen olyan ember él, akinek a havi bevétele az 1 milliót meghaladja, máris egyértelművé válik számára, hogy ez a személy melyik ekvivalencia-osztályba tartozik, ami alapján pedig már a szenzitív adatokra is egyszerűbben tud következtetni.

Ahogy azt az előzőekben meg is mutattuk, az általunk javasolt algoritmus viszont nem engedélyezi az intervallumok kiterjesztését. Amennyiben a szerveren nem található megfelelően illeszkedő ekvivalencia-osztály, új kliens érkezésekor a mechanizmus új ekvivalencia-osztály generálását írja elő. Felmerülhet a kérdés, hogy ez miként segíti az említett probléma kiküszöbölését, hiszen a külső tények birtokában lévő támadó így is azonnal tudja, hol keresse az adott egyén szenzitív információit. A válasz a következő: Az új osztályokat a szerver megjelöli, s mindaddig úgy is tartja, míg nincs bennük elegendő elem a k-anonimitás biztosításához. A jelölt osztályokba tartozó személyek ez idő alatt nem is küldhetik fel a szerver központi táblájába az érzékeny adataikat. Így - még ha tudomásunk is van róla, hogy csak egyetlen milliós jövedelmű ember van az adott kerületben - a kiugró értékeket tartalmazó ekvivalencia-osztályok nem kerülnek publikálásra, pontosan azon okból kifolyólag, hogy az elemszámuk sosem éri el a *k*-t. Ugyan a kommunikációt kezdeményező kliensek megkaphatják, hogy létezik ilyen attribútumokkal ekvivalencia-osztály, a hozzá tartozó szenzitív adatok viszont az előbbi állítás miatt nem kerülnek be a szerver központi táblájába, tehát a kiugró értékkel rendelkező személy adatai védve maradnak.

5.2.3 Eredmények

A fentiek alapján elmondható, hogy az általunk javasolt kliens oldali k-anonimizáló algoritmus - működéséből adódóan - alapvetően biztonságos, mivel segít elkerülni a kiugró értékeket kihasználó támadásokat. Már említettük a bizonyított tételt, miszerint a biztonságosan végrehajtott k-anonimizálás minden $\delta < 1$-re kielégíti a mintavételezéses $(\beta, \mathcal{E}, \delta)$-Differential Privacy követelményeit, feltéve, hogy $\beta > \delta$ is teljesül. Következésképpen mi az alábbi tételt vezettük le:

Tétel: A harmadik fejezetben bemutatott mechanizmus alkalmazása előtt elegendő egy véletlenszerű mintavételező lépés beiktatása ahhoz, hogy - megadott paraméterek használatával - a módszer a k-anonimitás fenntartása mellett a közelítő $(\mathcal{E}, \delta)$-Differential Privacy által előírt elvárásoknak is megfeleljen.

A bizonyítást a korábbi fejezetek alapján csupán vázlatosan adjuk meg.

1. A dolgozatban ismertetett kliens oldali anonimizáló algoritmus k-anonim kimeneti adathalmazt állít elő *(3.7 fejezet)*.
2. A mechanizmus bemeneti adatbázisán minden esetben végrehajtható egy véletlenszerű mintavételező lépés *(5.2.1 fejezet)*.

3. A k-anonimizáló algoritmus - a protokoll működéséből adódóan - automatikusan teljesíti a biztonságosság feltételeit *(5.1* és *5.2.2 fejezet)*.
4. Több kutatásban is alátámasztották már a fenti tézist [20] [26], mely kimondja, hogy a biztonságosan elvégzett k-anonimizálás minden $\delta < 1$ és $\beta > \delta$ esetén kielégíti a mintavételezéses $(\beta, \varepsilon, \delta)$-Differential Privacy követelményeit.
5. Mivel beláttuk, hogy az általunk javasolt k-anonimizáló mechanizmus biztonságos, az előzőleg említett tétel alkalmazásával kimondhatjuk, hogy megfelel a mintavételezéses Differential Privacy előírásainak, azaz $\delta < 1$, $\beta \leq 1$ és $\beta > \delta$ paraméterek mellett az algoritmus használata előtt elegendő egy véletlenszerű mintavételező lépés ahhoz, hogy a módszer a közelítő (ε, δ)-Differential Privacy követelményeit is teljesítse.

Mindezzel tehát alátámasztottuk az elképzelésünket, miszerint a k-anonimitás és a Differential Privacy közötti kapcsolatot a mintavételezés biztosítja.

6 Konklúzió

Manapság az adat az egyik legértékesebb erőforrás. A vállalatok az információkat általában üzletfejlesztési és tanulási célból őrzik meg, azonban gyakran előfordul, hogy egy-egy cég túlzásba viszi az adatgyűjtést. Ennek próbál gátat szabni a 2018. májusában hatályba lépett Általános Adatvédelmi Rendelet, mely olyan irányelveket határoz meg, amiknek minden európai intézménynek meg kell felelnie. A GDPR ugyan jogi előírás, valójában viszont a technológiai cégek és mérnökeik feladata megoldani, hogy a vállalatok teljesíteni tudják a rendeletben foglaltakat.

A dolgozatban egy olyan - k-anonimitáson alapuló - GDPR-kompatibilis megoldást vezettünk be, melynek felhasználásával az adatkezelők automatikusan megfelelnek az előírásoknak. Mivel az anonimizált adatok nem esnek a rendelet hatálya alá, ha mindent az adatrögzítés helyén anonimizálunk, s később is csak anonim formában tároljuk az információkat, mentesülhetünk a GDPR-követelmények alól. Megmutattuk többek között, hogy a kliens oldalon anonimizáló algoritmus hogyan kategorizálja az adatokat, hogyan inicializál egy üres adatbázist, illetve hogyan használ fel egy már meglévő adathalmazt.

Ugyan az ismertetett módszer a GDPR körüli nehézségeket kiküszöböli, önmagában mégsem nyújt 100%-os adatbiztonságot. Ennek alátámasztásaként több potenciális támadási modellt is megvizsgáltunk, illetve közülük néhányra megoldási lehetőségeket is kínáltunk.

A mechanizmus szélesebb körű kiértékeléséhez - az anonimizálás szimulációja után - az eredmény adathalmazon többféle metrika alapján végeztünk méréseket.

Végül, annak érdekében, hogy a különböző sebezhetőségeket minél nagyobb arányban kivédhessük, utánajártunk azon feltételezésünknek, miszerint a k-anonimitás és a Differential Privacy modell kombinálható. A bemutatott eljáráson és annak bemeneti adathalmazán olyan minimális változtatásokat ejtettünk, melyekkel a k-anonimitás fenntartása mellett az algoritmus a Differential Privacy által támasztott követelményeknek is megfelel. A gyakorlatban is hasznosítottuk az irodalomból már ismert elméleti sémákat, aminek eredményeképpen levezettük, hogy a k-anonimitás és a Differential Privacy között a kapcsolatot alapvetően a mintavételezés biztosítja.

7 Köszönetnyilvánítás

A dolgozatban ismertetett eredmények a Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar Balatonfüredi Hallgatói Kutatócsoport szakmai közössége keretében jöttek létre a régió gazdasági fejlődésének elősegítése érdekében. Az eredmények létrehozása során figyelembe vettük a balatonfüredi központú Rendszertudományi Innovációs Klaszter által megfogalmazott célkitűzéseket, valamint a párhuzamosan megvalósuló EFOP 4.2.1-16-2017-00021 pályázat támogatásával elnyert „BME Balatonfüredi Tudáscentrum" térségfejlesztési terveit.

A kutatás az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.2-16-2017-00013, Innovatív Informatikai és Infokommunikációs Megoldásokat Megalapozó Tematikus Kutatási Együttműködések).

8 Irodalomjegyzék

[1] C. C. Aggarwal, „On k-Anonymity and the Curse of Dimensionality,” in *31st VLDB Conference*, 2005.

[2] L. Sweeney , „Simple Demographics Often Identify People Uniquely,” in *Carnegie Mellon University*, Pittsburgh, 2000.

[3] P. Samarati, L. Sweeney, „Protecting Privacy when Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression,” SRI computer science laboratory, Palo Alto, CA, 1998.

[4] M. E. Nergiz és C. Clifton, „Thoughts on k-Anonymization,” in *22nd International Conference on Data Engineering Workshops*, 2006.

[5] R. J. Bayardo és R. Agrawal, „Data Privacy Through Optimal k-Anonymization,” in *21st International Conference on Data Engineering*, 2005.

[6] A. Gionis, A. Mazza és T. Tassa, „k-Anonymization Revisited,” in *Yahoo! Research*, Barcelona, Spain, 2008.

[7] S. Ren-jie , L. Zhong-yue és F. Liang-tao, „An Improved K-anonymity Algorithm Model,” in *The 1st International Conference on Information Science and Engineering*, 2009.

[8] A. Basu, T. Nakamura, S. Hidano és S. Kiyomoto, „k-anonymity: risks and the reality,” in *IEEE Trustcom/BigDataSE/ISPA*, 2015.

[9] A. Meyerson és R. Williams, „General k-Anonymization is Hard,” in *Carnegie Mellon University*, Pittsburgh, 2003.

[10] L. Sweeney, „k-anonymity: a model for protecting privacy,” *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems,* pp. 557-570., 2002.

[11] O. Angiuli, „Quora,” [Online]. Available: https://www.quora.com/What-are-the-weaknesses-of-k-anonymity. [Hozzáférés dátuma: 21. szeptember 2019.].

[12] A. Machanavajjhala, J. Gehrke, D. Kifer, M. Venkitasubramaniam, „l-Diversity: Privacy Beyond k-Anonymity," Department of Computer Science, Cornell University.

[13] N. Li, T. Li, S. Venkatasubramanian, „t-Closeness: Privacy Beyond k-Anonymity and l-Diversity," Department of Computer Science, AT&T Labs – Research, Purdue University.

[14] Kellaris, Szerző, *Differential Privacy.* [Performance]. Georgios.

[15] J. D. Cook, „John D. Cook Consulting," 6. november 2018. [Online]. Available: https://www.johndcook.com/blog/2018/11/06/what-is-differential-privacy/. [Hozzáférés dátuma: 23. szeptember 2019.].

[16] Dr. Hector Page, Charlie Cabot, Professor Kobbi Nissim, „Differential privacy: an introduction for statistical agencies," Georgetown University, 2018.

[17] C. Dwork, A. Roth, „The Algorithmic Foundations of Differential Privacy," in *Foundations and Trends in Theoretical Computer Science*, USA, 2014, pp. 211-407.

[18] „Carnegie Mellon University, Statistics & Data Sciences," [Online]. Available: http://www.stat.cmu.edu/~larry/=sml/diffpriv.pdf. [Hozzáférés dátuma: 25. szeptember 2019.].

[19] S. Kiyomoto és Y. Miyake, „How to find an appropriate *k* for *k*-anonymization," in *Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2014.

[20] N. Li, W. Qardaji, D. Su, „On Sampling, Anonymization, and Differential Privacy: Or, k-Anonymization Meets Differential Privacy," Lafayette, IN 47907, USA, 2011.

[21] K. LeFevre, D. J. DeWitt és R. Ramakrishnan, „Mondrian Multidimensional K-Anonymity," in *22nd International Conference on Data Engineering*, 2006.

[22] Q. Gong és L. Kun, *Python Implementation for Mondrian Multidimensional K-Anonymity,* 2017.

[23] I. Wagner és D. Eckhof, „Technical Privacy Metrics: a Systematic Survey," in *ACM Computing Surveys, Vol 51, No 3, Article 57,* 2018.

[24] V. Ayala-Rivera, P. McDonagh, T. Cerqueus és L. Murphy, „A Systematic Comparison and Evaluation of k-Anonymization Algorithms for Practitioners," in *Transactions on Data Privacy,* 2014.

[25] G. Ghinita, P. Karras, P. Kalnis és N. Mamoulis, „Fast Data Anonymization with Low Information Loss," in *Association for Computing Machinery*.

[26] N. Li, W. Qardaji, D. Su, „Provable Private Data Anonymization: Or, k-Anonymity Meets Differential Privacy," Lafayette, IN 47907, USA.