

Debreceni Egyetem
Informatikai Kar
Adattudomány és Vizualizációs Tanszék

Blokklánc alapú biztonsági keretrendszer IoT eszközökre

TDK dolgozat

Konzulens:

Dr. Oláh Norbert
Adjunktus

Szerző:

Nagy Csaba Norbert
Mérnökinformatikus BSc

Debrecen

2023

Bírálatra való beküldéshez hozzájárulok.

Tartalomjegyzék

1.	Bevezető.....	1
2.	Internet of Things.....	3
2.1.	IoT rendszer felépítése.....	3
2.1.1.	IoT rendszer előnyei	5
2.1.2.	IoT rendszer hátrányai	6
2.1.3.	IoT biztonság	8
2.2.	Okosotthon	11
2.2.1.	Meghatározás és célok.....	12
2.2.2.	Okosotthon részei	13
3.	Blokklánc	16
3.1.	Kriptográfia a blokkláncba	16
3.2.	Ethereum	20
3.2.1.	Okos szerződés és Solidity	24
4.	Okosotthon biztonsági keretrendszer	26
4.1.	Aktualitás, sérülékenységek, motiváció	26
4.2.	Rendszer résztvevői.....	28
4.3.	Javasolt megoldás	29
5.	Biztonsági kérdések.....	31
5.1.	Biztonsági követelmények.....	31
5.2.	Biztonsági megoldások és a kriptográfiai primitívek által nyújtott védelem.....	33
6.	Implementáció	36
6.1.	Komponensek.....	37
7.	Összefoglaló	42
8.	Köszönetnyilvánítás.....	44
9.	Irodalomjegyzék	45
10.	Ábrajegyzék	49

1. Bevezető

Az életünkben egyre nagyobb szerepe van a városainkban, kórházainkban és otthonainkban megjelenő okos eszközöknek, amelyek megkönnyítik hétköznapijainkat. Ezek az eszközök jellemzően egy központi egységhez csatlakoznak és kommunikációt folytatnak egymással és a központi egységgel. Az IoT eszközöknek számos alkalmazási területe van (okosváros, okosotthon, drónok és autók hálózata), azonban sok esetben sérülékenyek ezek a rendszerek. A támadóknak elég lehet egyetlen eszközhöz hozzáférniük és onnantól akár az egész hálózatot is kompromittálhatják. Számos incidens mutatja (Mirai, Gafgyt, Hajime, Tsunami) ([1][2]), hogy sokszor az alapvető biztonsági beállítások is hiányoznak (például a jelszó beállítás), ezért az IoT eszközök különösen ki vannak téve a rosszindulatú támadóknak. Emellett a legtöbb IoT eszköz korlátozott erőforrással rendelkezik, így a hagyományos kriptográfiai megoldások nem alkalmazhatóak ezekre az eszközökre. Egy másik releváns probléma az IoT eszközöknél, hogy a gyártók a termék kiadásától bizonyos ideig nyújtanak csak támogatást, így idővel az eszközök szoftverei vagy firmwarei elavulnak és ezáltal válnak sérülékennyé. Fontos továbbá, hogy az IoT eszközök által generált nagy mennyiségű adatot általában felhőkben tárolják.

A dolgozatban egy biztonsági keretrendszert javasolunk blokklánc technológiával okosotthon környezetben használva. A keretrendszer biztosít egy eszközközkezelőt, amely IoT eszközök különböző attribútumait tárolja, mint név, firmware verzió, típus, tulajdonos, állapot. Ehhez egy engedélyköteles blokkláncot alkalmazunk, amely a szenzitív adatokat tartalmazó attribútumokat titkosítva tárolja (pl. verzió, állapot). Az eszközközkezelő része egy okosszerződés. Az okosszerződés egy program, mely a blokkláncon fut és a kód (függvények) valamint az adat (állapot) gyűjteménye. Az implementált okosszerződés célja, hogy ellenőrizhető legyen, ha a felhasználó nem változtatta meg az alapértelmezett jelszót az eszköznél vagy az eszközhöz tartozó szoftver, illetve firmware elavult verzióval rendelkezik. Végezetül a keretrendszer biztonsági mentést nyújt, ahol az okosotthon beállítások és konfigurációk elosztottan kerülnek tárolásra a blokkláncon. A keretrendszer biztonságosnak tekinthető, mivel a webalkalmazás megköveteli a felhasználó hitelesítését, továbbá a felhasználó és a keretrendszer biztonságosan csatornát használ TLS protokollt alkalmazva. A blokklánc biztosítja az elosztott tárolást, illetve az okosszerződés egy elosztott alkalmazás, amely nagyobb rendelkezésre állást biztosít. Így az

okosotthon beállításaival és konfigurációival könnyebb helyreállítani az okosotthont különböző kártékony vagy zsaroló programok fertőzése esetén. A blokkláncon lévő adatok publikusak, ezért a szenzitív adatok titkosítva kerülnek hozzáadásra a blokklánchoz és csak a helyes visszafejtő kulccsal rendelkező résztvevők képesek visszafejteni azokat.

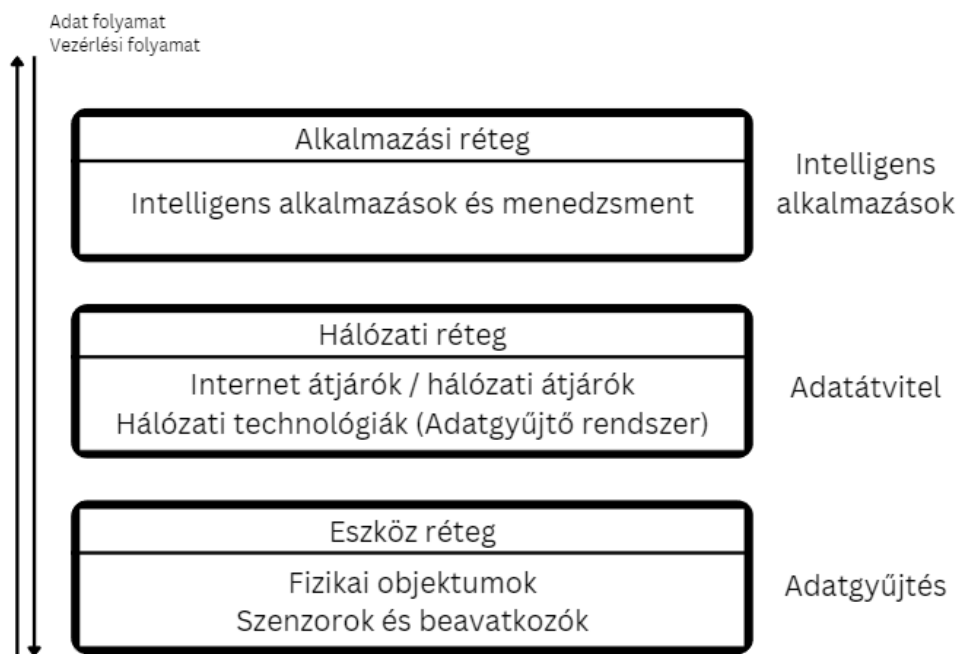
A javasolt rendszerről egy prototípust készítettünk, amely egy webes alkalmazás Javascript nyelven implementálva. Az implementáció kialakítása során teszteltük a rendszer funkcionalitását és a összegyűjtöttük a rendszerünk biztonságát adó megoldásokat és kriptográfiai primitíveket.

2. Internet of Things

1999-ben Kevin Ashton megalkotta a dolgok internete kifejezést, amely kiemelkedő szerepet játszott a dolgok internetének történetében és a technológiai fejlődésben. Ashton a Procter & Gamble számára tartott előadásában az Internet of Thingst (IoT) egy olyan technológiaként fogalmazta meg, amely az 1983-ban hivatalosan megjelent RFID (Radio Frequency Identification) segítségével több eszközt kapcsolt össze ([3]). A dolgok internete egy intelligens rendszernek feleltethető meg, amely rendelkezik átfogó célokkal, megbízható adatátvitellel és intelligens adatfeldolgozással. Az IoT rendszerek az életünket minden téren körül fogják, ahol a főbb alkalmazásuk az okosvárosok, az okosotthonok, az intelligens tömegközlekedés, az egészségügy és az ipari környezetek ([4]). Az IoT technológiában minden eszköz egyedi azonosítóval rendelkezik és könnyen hozzáférhető. Azonban a vezetékes és vezeték nélküli kommunikációs eszközök számának növekedése kihívást jelent, ugyanis a természeti erőforrások végesek, és környezetvédelmi szempontból is hátrányt jelentenek. A Metcalfe-törvény alapján 2020-ra világszerte közel 50 milliárd eszköz csatlakozott a hálózatra és ez a szám exponenciálisan növekszik évről évre ([5]).

2.1. IoT rendszer felépítése

Az IoT ökoszisztéma egy összetett rendszer, melynek szereplőit az egyszerűbb megértés érdekében egy általánosított IoT architektúrákba rendezték, melyekre három-öt-hét rétegű architektúrákat javasoltak ([6]). Azonban tanulmány is vitatja, hogy a rétegek közötti hasonlóságok és különbségek miben rejlenek és mely architektúrát milyen ökoszisztémában érdemes használni. A dolgozatban a háromrétegű architektúrát tárgyaljuk meg [6], amely az alábbi rétegekből épül fel:



2.1.1 ábra IoT architektúra rétegenkénti egységei

- *Eszköz réteg*

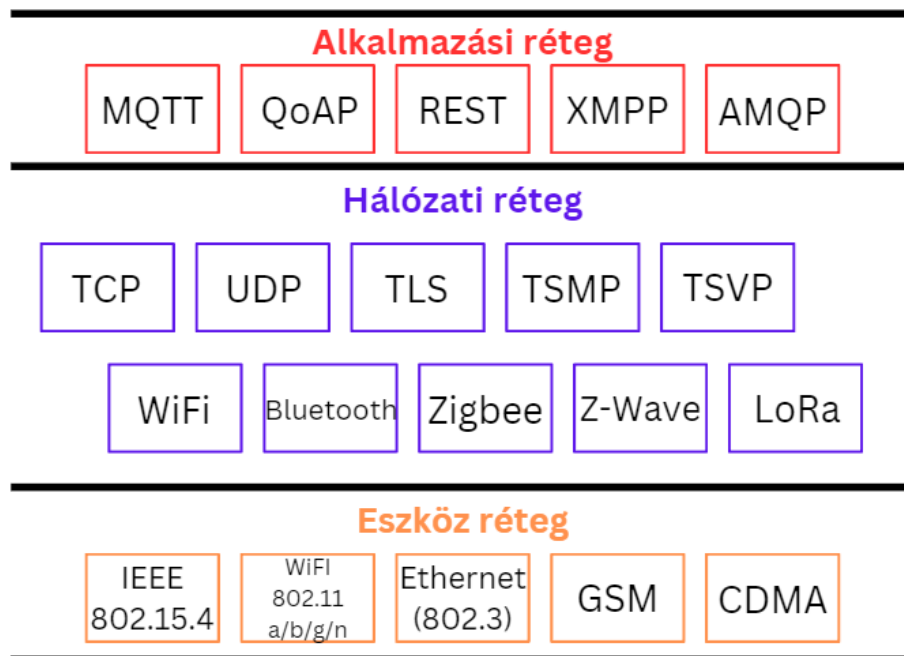
A legalsó réteg a hierarchiában a fizikai réteg, ami magába foglalja azokat a fizikai eszközöket és protokollokat (IEEE 802.15.4, GSM, CDMA, LTE), amelyek érzékelik és gyűjtik az adatot a környezetből és a fizikai világ eseményeire reagáló módosítást teszik lehetővé a beavatkozó eszközök (aktuátorok) által. Feladatkörükbe tartozik az összegyűjtött adatok digitális formába való alakítása és azok továbbítása más rétegek felé. Ilyenek például: a szenzorok, kapcsolók, vezérlő egységek és beágyazott rendszerek ([6]).

- *Hálózati réteg*

A hálózati eszközök között kommunikáció valósul meg, illetve az ehhez szükséges protokollok (IP, 6LoWPAN, TLS, TCP, UDP) és technológiák is itt helyezkednek el, mint például a Wi-Fi (WEP, WPA, WPA2, WPA3), a Bluetooth, a Bluetooth Low-Energy (BLE), a ZigBee, az RFID és a Z-Wave ([7]). IoT környezetben a vezeték nélküli kommunikációs protokollokat gyakrabban használják, mint a vezetékes protokollokat, hiszen vezeték nélküli eszközök módosítását, frissítését lényegesen egyszerűbb kivitelezni. Protokollok kiválasztása a hardver eszközök heterogenitásától, az energiafogyasztástól, az átviteli sebességtől és az átviteli távolságtól függ ([6]).

- *Alkalmazási réteg*

Olyan szoftverekből és alkalmazásokból tevődik össze, amelyek a begyűjtött adatokat kezelik és felhasználják (MQTT, QoAP, REST, XMPP, AMQP). Ebben a rétegben jelennek meg a felhőszolgáltatások is. Ide sorolhatóak továbbá azok a webalkalmazások, amelyek egyaránt hozzáférhetőek nyílt (webes) vagy zárt (belső hálózat) interfészen keresztül, illetve az okosotthoni, a távvezérlő, a blokklánc vagy a mesterséges intelligencia alapú rendszerek.



2.1.2 ábra IoT architektúra rétegenkénti protokollok megjelenései

2.1.1. IoT rendszer előnyei

Az IoT rendszerek számos előnyt nyújtanak a felhasználók számára. Az érzékelők által nagymennyiségű adatgyűjtés valósulhat meg, az összegyűjtött adatokat pedig továbbítják. Ez hozzájárul, a környezetünk és annak változásainak megismeréséhez. A nyitottság és a nagyobb átláthatóság a jobb felügyeletet, illetve a vállalkozások, felhasználók jobb döntéshozatalát segíti elő. Az információk elemzésével a döntéshozatal javul azáltal, hogy a fizikai eszközöktől érkező információk naprakészek, ami lehetővé teszi az aktuális állapotok elemzését és nyomonkövetését. Az automatizálás és ellenőrzés területén a már feldolgozott adatokból származó kimenetek javítják a hatékonyságot és az eredményességet ([8]). A beavatkozók a

megfigyelt közegben várnak, amíg az érzékelők által mért adatokat nem szükséges módosítani, beavatkozni, így hatékony rendszereket tudunk teremteni (például energiatakarékosabbat). Az automatizálás segítségével a felhasználók által kívánt igényeket pontosíthatjuk, gyorsíthatjuk így teremtve meg számukra egy kényelmi állapotot. Az eszközök hálózati interfésszel rendelkeznek, ezáltal hálózati csomóponthoz csatlakoznak, így a távoli vezérlés is megvalósítható, amelyeket korábban csak helyben lehetett elérni. Az IoT rendszereknél fontos a felhasználóbarát platform kialakítása, azaz, hogy a rendszer a felhasználók számára könnyen kezelhető legyen és biztonságérzetet nyújtson. Rendszerek megtervezése és fejlesztése során, az egyik kiemelten fontos alapelv a személyes adatok védelme, így a modern IoT rendszereink nagyobb biztonsági szinttel kell, hogy rendelkezzenek. Fontos, hogy az eszközök hardveresen és szoftveresen is már alapértelmezetten tartalmazzanak védelmi mechanizmusokat, melyek ha szükséges, további biztonsági funkciókkal bővíthetők. Emellett a rendszereknek az érzékelés tulajdonságuk segítségével előrejelzéseket kell generálniuk, hogy ezzel is folyamatos trendelemzést tudjanak biztosítani. A felhasználókat felhatalmazhatjuk különböző műveletek végrehajtására a rendszereken belül, figyelembe véve a jogszabályok minél hatékonyabb betartását. A szolgáltatások hatékonyságának és rugalmasságának javítása elősegíti a költségcsökkentést magas számú eszközzel bíró rendszerek eseteiben.

2.1.2. IoT rendszer hátrányai

Számos előnyös tulajdonság mellett az IoT eszközök számos technikai és szabályozási hiányossággal is rendelkeznek. A szabályozási oldalt tekintve a legfontosabbak az adatok tulajdonjogával kapcsolatos, mint például a személyes adatok védelme és az információk megosztása. Az adatok sérülékenységi szintje döntő szerepet tölt be, amelyet hozzáférés-ellenőrzési mechanizmusoknak figyelembe kell vennie. Az adatkezelés nem megfelelő kezelése akadályozhatja az IoT eszközök által generált adatok biztonságos és etikus felhasználását ([8]). Egy eszköz ára széles skálán mozoghat a piacon. A felhasználók hajlamosak az olcsóbb eszközök megvásárlására, figyelmen kívül hagyva azt a tényt, hogy a drágább eszközök milyen egyéb extra tulajdonságokkal rendelkeznek, mint a hozzá hasonló termékek olcsóbb változata. Funkcionalitást tekintve a drágább és olcsóbb termék is képes lehet ugyanazt a kényelmi szintet nyújtani a felhasználó számára, azonban az olcsóbb eszközök esetében sokszor a biztonsági szempontból kulcsfontosságú hardveres, szoftveres alkotó elemek maradnak ki a gyártás során.

A hátrányok mellett az eszközök darabszáma folyamatos növekszik, így komplex rendszereket eredményez heterogén eszközökkel.

Komplex rendszerek esetében nehéz kezelni és karbantartani az eszközöket, sokszor rövid élettartammal bírnak, cseréjük, frissítésük nehéz és magas költséggel jár, valamint az eszközök folyamatos működése magas energiafogyasztással jár. Az eszközök sokfélék lehetnek a kommunikációs módszereket és képességeket tekintve, hiszen a számítási és tárolási teljesítmény, az energia ellátás, az alkalmazkodóképesség, a mobilitás is befolyásoló tényező lehet. Ezek miatt egy teljes összetett rendszert kiépíteni magas költséggel jár, mivel sok heterogén eszközből és rendszerből állnak, amelyek eltérő követelményekkel rendelkeznek. Egy példa egy ilyen rendszer kialakítására a Walmart amerikai vállalat, ahol egy RFID technológián alapuló teljes IoT rendszer implementációjára 1-3 millió dollárt költöttek ([8]). Más piacokhoz képest azonban ez egy hatalmas piaci előnyt képzett még úgyis, hogy elengedhetetlenül fontos a rendszerek és eszközök folyamatos karbantartása és frissítése.

Szoftver tekintetében, a nagymennyiségű adatgyűjtés és továbbítás adatbiztonsági kockázatokkal jár az IoT eszközök kommunikációja során. A bizalmasság szempontját felvetve, sokszor könnyen korrumpálhatók az eszközök, ezáltal érzékeny adatok kerülhetnek illetéktelen tulajdonba. Azonban sok esetben a megfelelő tudás hiányában, esetlegesen adatkezelés során is felmerülhetnek olyan problémák, amelyek mérsékelt adatbiztonsági kockázattal járhatnak.

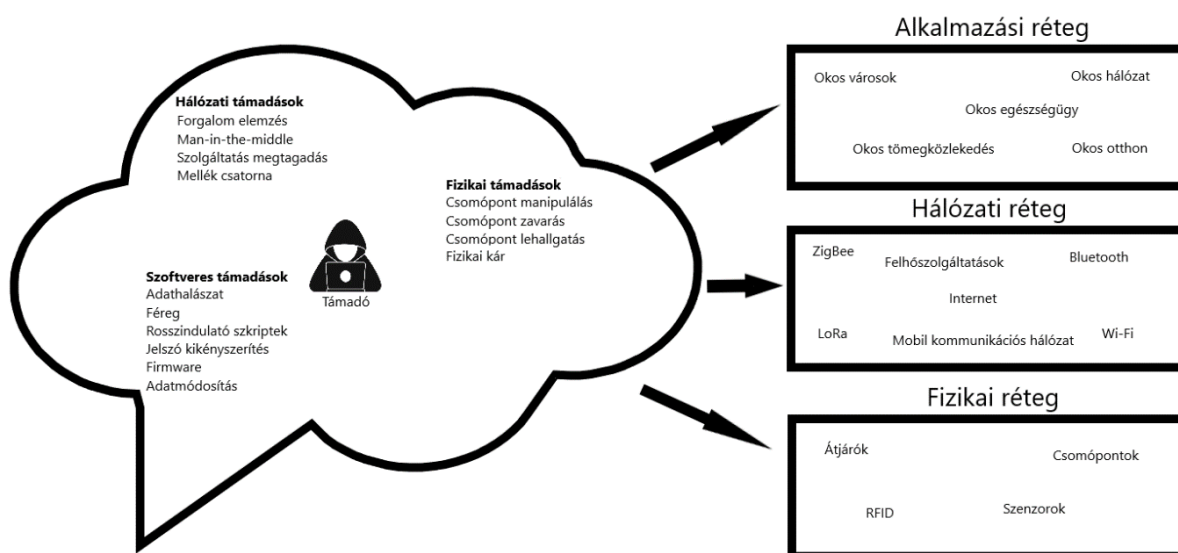
Az alábbi táblázat az IoT rendszerek számos előnyeit és hátrányait stratégiai, taktikai és operatív szempontok szerint adja meg ([8]).

	Előnyök	Hátrányok
Stratégiai	Fejlesztett előrejelzés és trendelemzés, Átláthatóság biztosítása, A felhasználók felhatalmazása	Adatbiztonság, Ellentétes piaci erők, Szabályozások be nem tartása
Taktikai	Jobb tervezés, Jogszabályok hatékony végrehajtása, Fejlett egészségügyi és biztonsági intézkedések, Költségcsökkentés	Átjárhatóság és integráció, Bizalmassággal kapcsolatos kérdések

Operatív	Szolgáltatások hatékonyságának, rugalmasságának javítása, Valós idejű nyomonkövetés	A megfelelő ismeretek hiánya, Adatkezelési problémák, Hibás adatok
-----------------	---	--

2.1.3. IoT biztonság

Az IoT biztonság az IoT rendszerek és eszközök megfelelő védelmét jelenti a külső fenyegetésekkel szemben. Magába foglalja a hardveres és szoftveres védelmi mechanizmusokat, az adatvédelmet és a felhasználói jogosultságok kezelését. Az IoT biztonság több szempontból hatással van az IoT rendszerekre, eszközökre, adatokra és az emberekre is egyaránt. A kockázati tényezők többek között az adatok védelme és a jogosulatlan hozzáférés megakadályozása, az eszközök közötti kommunikációt biztosító protokollok megbízhatósága, a rendszerek és az eszközök integritásának és rendelkezésre állásának biztosítása. IoT környezet minden résztvevőjét rendszeres ellenőrzése alá kell vetni, és folyamatos frissítést igényelnek, az újonnan megjelenő biztonsági kockázatok és fenyegetések elkerülése miatt. A biztonságnak többretegűnek kell lennie a támadások megelőzése érdekében. Fizikai (hardveres) támadásoktól kezdve a logikai (szoftveres) támadásokig széles skálán vannak feljegyezve a támadás típusok és az általuk támadni kívánt rétegek az IoT architektúrában, amelyet a 2.1.3.1 ábra mutat.



2.1.3.1 ábra IoT ökoszisztémában lévő rétegek elleni támadások

A biztonsági célokat megvalósító mechanizmusok feladata az, hogy széleskörben lefedjenek minden olyan sérülékenységi pontot, amely bármilyen formában kompromittálható egy IoT

környezetben. Például a gyenge hitelesítési adatoknak köszönhetően ment végbe a Mirai botnet kibertámadás ([4]). Az IoT ökoszisztémák körébe tartoznak biztonsági követelmények, ezeknek a teljesülését különböző mechanizmusok (algoritmusok, ügyviteli, fizikai, logikai szabályok) biztosítják. A biztonsági követelményeket három kategóriába lehet sorolni: hardveres, szervezeti és egyéni, technikai követelmények.

Hardveres követelmények közé tartozik, amikor az eszközt egy speciális biztonsági chippel látják el. Ez a chip a biztonsági modulon keresztül lehetővé teszi a biztonságos rendszerindítási folyamatot, véletlen számok generálását és az eszközzel szembeni hamisítási kísérlet érzékelését, erre egy gyakorlati példa az Intel Firmware Guard, ami lehetővé teszi a firmware verzió frissítését valamint firmware hiba vagy támadás esetén annak helyreállítását. Hasonló biztonsági funkciókat lát még el az Intel BIOS Guard és Intel Boot Guard.

A szervezeti és egyéni követelmények a humánerőforrások körét foglalja magába. A követelmények betartása a rendszerek tudatos és biztonságos használatát segíti elő, az egyének folyamatos képzésével, biztonsági rések és incidensek rendszerinti kezelésével.

Végül a technikai követelmények, melyeknek célja az IoT rendszerekben csökkenteni az különböző elemekhez tartozó lehetséges kockázatokat. Erre példa az eszközmenedzsmenthez kapcsolódó rendszer kialakítása, fenntartása abból a célból, hogy az eszközök azonosíthatók és hitelesíthetők legyenek, illetve a kockázatok és fenyegetések azonosítása és értékelése az eszköztől az alkalmazásig, annak érdekében, hogy belső, illetve harmadik féltől származó kockázatok redukálhatóak legyenek. Biztonsági rendszerek tekintetében elengedhetetlen az AAA fogalom elméleti és gyakorlati használata, melynek elemei a következők:

- Hitelesítés (Authentication) – Az eszközökhöz hozzáférni kívánt felhasználó kilétéről való meggyőződés és a megbízható hozzáférést jelenti. Számos hitelesítési eljárás létezik, többek között egy és kétfaktoros hitelesítési megoldások. Erre egy példa a felhasználó név és jelszó egyfaktoros hitelesítés alkalmazása, melynél több fontos beállítás is megkövetelt. Ide tartoznak a jelszavak és PIN (Postal Index Number) kódok magas entrópiája, ahol fontos, hogy egyediség jellemezze, minimum 10-12 karakterből álljon és tartalmazzon számot, speciális karaktereket.
- Jogosultságkezelés (Authorization) – Az a folyamat, amely az adott jogosultsági szinteket és azokhoz kapcsolódó jogokat határozza meg. Az információbiztonság egyik legfontosabb eleme, mivel a jogosultságok megfelelő kezelése növeli a rendszer biztonságát. Például az engedélyezett műveletek végrehajtásának korlátozása, a

hozzáférési mechanizmusok beállítása és naprakészen tartása, szerepkör-alapú hozzáférés-kezelés alkalmazása.

- Naplózás (Accounting) – Rögzíti a felhasználóhoz köthető hitelesítéseket, a felhasználói fiókok és hozzáférések kezelését, szabályok módosítását és a rendszer működésével kapcsolatos tevékenységeket.

Az IoT rendszerben lévő adatok és információk bizalmasságának, hitelességének, integritásának és rendelkezésre állásának védelme érdekében indokolt a kriptográfia függvények és algoritmusok megfelelő és hatékony használata az adatok tárolásánál és továbbításánál egyaránt. Az IoT eszközök és rendszerek biztonságának növelése érdekében számos javaslat szerepel a tudományos irodalmakban. Ennek érdekében különböző kulcscsere protokollok tervezése és biztonsági elemzése tartozik. A protokoll tervezési folyamatában kiemelt fontosságú a megfelelő jelszóhasználat beállítása és a végpontok közötti biztonságos kommunikáció biztosítása. A javasolt protokoll a [9] cikkben egy méretezhető és robusztus séma, amelyben a sikeres szótártámadáshoz a támadónak $k - 1$ darab okos otthoni eszközt (k a jelszó küszöbszám) kell tudnia kompromittálni. A [10] cikkben Bagherzandi egy jelszóval védett titokmegosztási (PPSS - Password-protected secret sharing) és küszöbszámon alapuló megoldást ismertetett. A [11] publikációban Jarecki egy egykörös optimális PPSS-sémát javasol, amely mindösszesen két üzenetküldést tartalmaz. Azonban ezek a megoldások nem skálázhatóak. Isler és Küpcü a [12][13] publikációjukban hasonló szempontokat vesznek figyelembe (skálázhatóság, robusztusság, jelszóhasználat stb.), azonban az általuk javasolt protokollok jobban alkalmazhatóak felhő környezetben és tartalmazzak tárolószolgáltatásokat. A [9] megoldás okos otthoni környezetben való használatra lett tervezve, ahol $n \geq 10$ eszköz és $o \geq 5$ küszöbszám esetén (o a hitelesítéshez szükséges IoT eszközök darabszáma) jobb hatékonysági eredményt nyújtanak. Mivel az IoT eszközök nagy része korlátozott erőforrással rendelkezik, a hagyományos tanúsítványokra alapozott kriptográfiai megoldások nem mindig alkalmazhatóak. Előnyösebb lehet egy alternatív megoldás használata, mint például az identitás alapú kriptográfia (Identity-based cryptography) és a bilineáris leképezések alkalmazása. A [14] cikkben egy platformfüggetlen implementációra tesznek javaslatot, melynek célja, hogy hatékony és nyílt forráskódú IBC implementációt biztosítson asztali, mobil és IoT platformok számára. A CryptID biztonságosan felgyorsítja a kód futtatást a különböző beágyazott környezetekben, mind a kliens-, mind a szerveroldalon, a WebAssembly használatával. A jelszó beállításának fontossága szintén kiemelendő, mivel különböző lassú hash algoritmusok

(Bcrypt, Argon2) segítségével védhetők meg a tárolt jelszavak a különböző offline támadások ellen. A [15] cikk egy új megközelítést mutat be, amely az identitás alapú kriptográfiában használt bilineáris leképezések többkörösítésével éri el a tárolt jelszavak megfelelő védelmét az offline támadások ellen (szótár és szivárványtábla támadások). Fontos megemlíteni, hogy az IoT eszközök által generált nagy mennyiségű adatot általában felhőkben tárolják. A [16] cikk szerzői egy olyan protokollt javasoltak, amely egy többszerveres jelszó alapú hitelesített kulcscsere sémát valósít meg a kliens és a felhőszolgáltató szerverei között. Ellentétben más, küszöbszámon alapuló titokmegosztási algoritmusokat alkalmazó és jelszó alapú protokollokkal, mint például a [10],[11],[13],[17], ebben az esetben bár a jelszóinformációkat megosztjuk a szerverek között, nem kell rekonstruálni a titkot a titokrészekből annak ellenőrzése érdekében, hogy a felhasználó hiteles-e. Egy IoT rendszer beállításánál fontos, hogy erős kulcsok és szabványos, biztonságos titkosítási algoritmusok használata legyen előírás és tiltani kell a nem biztonságos protokollok használatát. Szükséges a rendszeres ellenőrzések és felülvizsgálatok bevezetése. Nélkülözhetetlen a releváns események és integritási hibák felfedezése, valamint a rosszindulatú szoftverek felderítése és a rendszer rendellenességeinek nyomon követése és javítása. A védelem hatékonyságának érdekében szükséges a folyamatos monitorozás.

2.2. Okosotthon

A dolgozatban említett IoT eszközök menedzselésére egy olyan alkalmazást ajánlunk, amely okosotthon ökoszisztémában elhelyezett IoT eszközökre fókuszál. Összegyűjti, ellenőrzéseket végez rajtuk és a felhasználó számára információkat közöl az eszközök aktuális állapotáról. Az okosotthon olyan rendszer, amelyben az IoT eszközök összekapcsolódnak egy központi vezérlőegységgel, és ez lehetővé teszi többek között az otthoni környezet távoli vezérlését. A rendszer résztvevői többek között távolról vezérelhetik a világítást, fűtést, hűtést, a biztonsági rendszereket és minden egyéb elektromos készülékeket, amelyek rendelkeznek hálózati eléréssel. Aldrich és munkatársai az okosotthont a következőképpen határozta meg:

Olyan lakóhely, amely számítástechnikai és információs technológiával van felszerelve, reagál és kiszolgálja a lakók igényeit, javítva a felhasználók komfortérzetét és kényelmét. A biztonságot és szórakozást a technológia irányításával az otthonon belüli technológia és a külvilággal való kapcsolat révén biztosítja ([18]).

Az okosotthon rendszerek kiépítése és alkalmazása egyre népszerűbbek az otthonainkban és számos előnnyel rendelkeznek. Célja az otthoni életünk könnyebbé, kényelmesebbé, biztonságosabbá és hatékonyabbá tétele.

2.2.1. Meghatározás és célok

A dolgozatban az okosotthon alatt a következő meghatározást tekintjük. Az okosotthon több IoT eszköz és rendszer összekapcsolása, melyek egymással kommunikációt folytatnak. Adatot gyűjtenek, cserélnek és dolgoznak fel úgy, hogy a felhasználóknak lehetőségük legyen kezelni azokat manuálisan vagy automatikusan.

IoT eszközök típusait tekintve két csoportot alkothatunk:

- Korlátozott erőforrással rendelkező eszközök – amelyek biztonsága az alacsony erőforrásuk miatt (CPU – Central Processing Unit, memória, akkumulátor) korlátozott lehet.
- Nagy erőforrással rendelkező eszközök – melyek jellemzően hálózati tápellátással rendelkeznek. Képesek a szükséges kriptográfiai eljárások (például TLS protokoll) használatára, mivel hardver paraméterük (CPU, memória) jelentős számítási kapacitást biztosít.

Ahhoz, hogy biztonságos hálózatot építsünk, ismernünk kell a hálózatunk összetevőit. A hálózat egyik tagja az otthoni internetszolgáltató egy ISP (Internet Service Provider) által van biztosítva, amely lehet országos méretű, például Telekom vagy Digi, vagy kisebb szolgáltatók, amelyek csak egy megyében vagy néhány településen érhetők el ([19]). Az eszközök csatlakozhatnak vezetékiesen illetve vezeték nélkül hálózatunkra, ezáltal különböző hálózatokba sorolhatjuk őket.

Az otthoni eszközök és érzékelők kapcsolódhatnak helyi hálózathoz vagy ebből eredő otthoni alhálózathoz (HAN – Home Area Network).

- Nagysebességű hálózatok, általában Wi-Fi hálózatok (például: beltéri egység vagy mobil eszköz által biztosított Hotspot).
- Személyi hálózatok (PAN – Personal Area Network) vagy több eszköz által kialakított Ad-hoc hálózat, melyek alacsony sebességű és energiaigényű hálózatok (például: Bluetooth, Zigbee).

Ezen felül kapcsolódhatnak nagy kiterjedésű hálózatokhoz (WAN – Wide Area Network).

- Nagysebességű, magas energiafogyasztású hálózatok, melyek az internethez való hozzáférést biztosítják internetszolgáltató vagy mobilhálózatüzemeltető hálózaton keresztül.
- Kis sebességű, alacsony energiafogyasztású hálózatok (LPWAN – Low Power Wide Area Network), amely WAN kapcsolatot biztosít alacsony energiaigényeket megkövetelve a rendszer eszközeitől (például: LoRaWAN, Sigfox).

Eszközök által biztosított szolgáltatások és az okosotthon rendszer fő célja a felhasználó hétköznapi életének megkönnyítése és biztonságosabbá tétele. A berendezéseket egy vagy több eszközről tudjuk irányítani és kezelni (például távirányító vagy okostelefon segítségével) ezzel is biztosítva a kényelmes és felhasználóbarát irányíthatóságot. Távoli vezérlés egyfajta biztonságérzetet biztosít a felhasználónak, abban az esetben például, ha elfelejtette bezárni az ajtót vagy beélesíteni a riasztót, esetleg nem kapcsolt ki valamilyen magas fogyasztással rendelkező berendezést. Mindezeket figyelembe véve egy hatékony rendszer valósítható meg az energiatakarékosság és a biztonság szempontjából.

2.2.2. Okosotthon részei

Az okosotthoni rendszerek különböző formában jelennek meg. Az egyre növekvő IoT eszközök által több lehetőség áll rendelkezésre a felhasználók számára, ami az alábbi tanulmányból került összegzésre ([18]).

1. Virtuális asszisztens rendszerek: A rendszerek hang alapú interakciókkal irányíthatóak, melyekhez különböző feladatok ellátására alkalmas hangvezérlő eszközök tartoznak. A virtuális asszisztens rendszerek képesek más eszközök (termosztát, riasztó, beágyazott rendszerek) vezérlésére. Napjainkban a legnépszerűbb és legelterjedtebb virtuális rendszerek közé tartoznak az Amazon Alexa, a Google asszisztens, az Apple Siri és a Microsoft Cortana.

Telefonról, tabletről vagy számítógépről vezéreljük a televíziókon a csatornaválasztást és a zeneszámokat. A rendszer által a telefon kihangsúlyozza a lakásban lévő hangszórókat, és mozgásérzékelők segítségével a megfelelő helyiség hangszóróin játssza le a zenét. Az ébresztési funkció felébreszti a lakókat azáltal, hogy elkezd felhúzni a redőnyöket

és növeli a hangszórók hangerejét. A rendszer hangzónákat is támogat, így lehetőség van arra, hogy különböző helyiségekben eltérő zene szóljon.

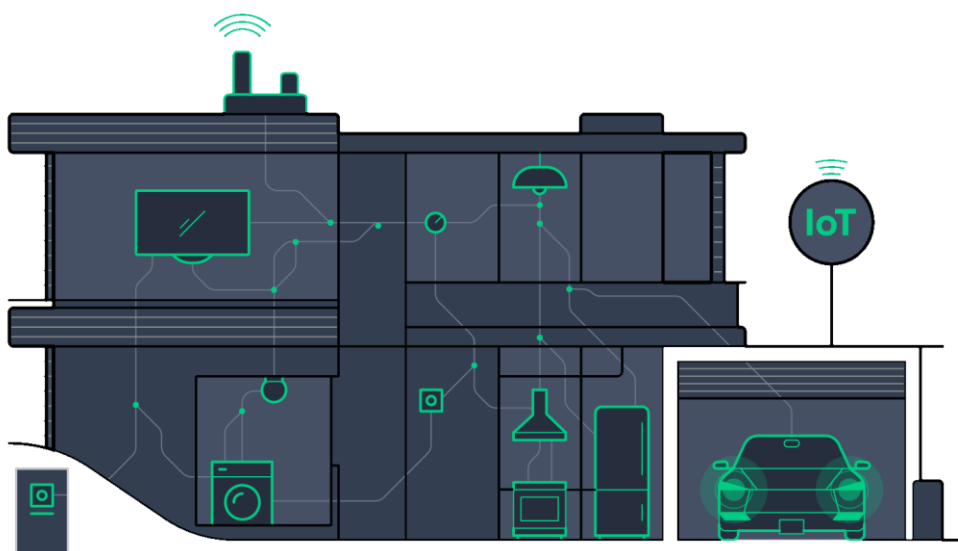
2. Biztonsági rendszerek: A már korábbiakban említett távoli vezérlés lehetővé teszi a felhasználók számára, hogy távolról is megbizonyosodjanak az ajtózáruk, riasztórendszerek és biztonsági kamerák megfelelő működéséről.

Az okosotthon biztonsági rendszerének részei közé tartozik a telepített biztonsági kamerák távoli megfigyelésének lehetősége, valamint központi zárrendszer az ablakon és ajtón. A kamerák vezérlése, mozgatása és zoomolása távolról is irányítható. A mozgásérzékelők észlelik a behatolót és riasztást küldenek a tulajdonosnak a biztonsági rendszeren vagy mobiltelefonon keresztül. Integrálhatók olyan érzékelő rendszerek, mint például a tűz- és füstjelző, a gázszivárgás-, a szén-monoxid-, illetve vízszivárgást észlelő szenzorok. A személyi orvosi riasztó rendszerek lehetővé teszik a sérült vagy beteg otthontartózkodók számára a segítségkérést.

3. Energiahatékonysági rendszerek: A termosztátok és a világítási berendezések által használni kívánt energia csökkentését valósítják meg, ezzel alacsonyabb költségeket biztosítva.

Az interneten keresztül vezérelhető termosztát, hozzájárul a távoli hőmérséklet-szabályozáshoz. A rendszer automatikusan vezérli a nyílászárók nyitását és zárását a friss levegő beáramlásának biztosítása érdekében, illetve az árnyékolók (redőnyök, rolók) mozgatását. Az intelligens fűtési rendszerek hőérzékelők segítségével mérik a helyiségek hőmérsékletét, és ennek alapján szabályozzák a fűtést. A világítási rendszerben olyan funkciók építhetők be, mint az energiahatékony világítás, amely automatikusan ki- vagy bekapcsol, ha valaki belép vagy kilép a szobából. A világítást távirányítóval vagy interneten keresztül lehet vezérelni. A természetes fény érzékelése alapján a rendszer szabályozza az árnyékolók és függönyök helyzetét, az optimális természetes fény beáramlásáért.

4. Egészségügyi rendszerek (telemedicina): A vérnyomásmérők, pulzusmérők, okoskarkötők, okosórák biztosítják a felhasználó számára az egészségügyi állapotuk folyamatos megfigyelését. Szükség esetén visszajelzést adnak külső szervezetek számára, hogy egészségügyi incidens esetén időben reagálhassanak.
5. Háztartási rendszerek: A háztartási eszközök (mosógép, kávéfőző) autonómok, programozhatók vagy alkalmazásokon keresztül távolról vezérelhetők.



2.2.2.1 ábra IoT Okosotthon szemléltetése ([20])

3. Blokklánc

1991-ben Stuart Haber és W. Scott Stornetta fogalmazták meg először a blokklánc alapötletét, melynek célja, hogy egy olyan számítással megvalósítható megközelítést kínáljanak, ami lehetővé teszi a digitális dokumentumok időbélyegzését, hogy megakadályozzák azok módosítását vagy meghamisítását ([21]). Létrehoztak egy rendszert, amely egy kriptográfiailag biztonságos blokkláncot használ a dokumentumok időbélyegzett tárolására. 2008-ban Satoshi Nakamoto megvalósította az elosztott blokkláncok elvét. Az architektúrát továbbfejlesztette úgy, hogy blokkokat adott hozzá a kezdeti lánchoz, anélkül hogy azokat megbízható felek által aláírták volna. Az adatátvitel során a tranzakciók biztonsági előzményeit több csomópont között tárolják. A résztvevők hálózaton keresztül időbélyegezik és ellenőrzik minden egyes tranzakciót ([21]).

A blokklánc egy egyenrangú, elosztott adatbázis (distributed database), mely biztosítja két fél között a közvetlen, kriptográfiailag biztonságos tranzakció végbemenetelét központi hatóság bevonása nélkül. A blokklánc blokkok sorozata, melyeket kriptográfiai hash függvényekkel kötnék össze, ezáltal a blokklánc teljes története megváltoztathatatlan (immutable).

3.1. Kriptográfia a blokkláncba

A kriptográfia az információ biztonságának és titkosításnak tudományága, mely kulcsfontosságú szerepet játszik a digitális világban. A kriptográfia magába foglalja a rejtjelező algoritmusokat, amelyeket adatok titkosítására vagy visszafejtésére használunk, illetve egyéb kriptográfiai eljárásokat, amik biztosíthatják a bizalmasságon (confidentiality) túl olyan tulajdonságokat, mint a sértetlenség (integrity), a hitelesítés (authentication), a letagadhatatlanság (nonrepudiation) vagy a nyomonkövethetőség (accountability).

A blokklánc technológiában három kriptográfiai sémát szoktunk jellemzően felhasználni ([22]):

1. Hash függvények – Egy $H: \{0, 1\}^* \rightarrow \{0, 1\}^n$, kriptográfiai hash függvény tetszőleges véges hosszúságú ösképtérbeli bináris sorozatot rögzített n hosszúságú bináris sorozatba képez, mely függvénynek az alábbi tulajdonságai vannak:
 - Egyirányú függvény

A H függvény egyirányú, ha a következő két feltétel teljesül:

- Könnyű kiszámítani: létezik olyan hatékony T algoritmus, amely x bemenetre a $H(x)$ kimenetet adja,
- Nehéz invertálni: tetszőleges hatékony T' algoritmus, és tetszőleges $p(n)$ polinom, valamint elegendően nagy n esetén

$$\Pr \{H(T'(H(x), 1^n)) = H(x)\} < \frac{1}{p(n)}.$$

ahol x egyenletes eloszlással választott őskép. T' invertáló algoritmusnak megadjuk az x képét bemenetnek azzal, hogy adjon meg egy ősképtérbeli elemet, amelynek azonos a képe. Egyirányú függvény esetén ezt a feladatot csak elhanyagolhatóan kis valószínűséggel képes megoldani bármely, a feltételeknek megfelelő algoritmus.

- Gyengén ütközés-ellenálló hash függvény
A H hash függvény gyengén ütközés-ellenálló vagy második őskép ellenálló x bemenetre, ha nehéz feladat előállítani egy $x' \neq x$ értéket (második ősképet), amelynek azonos a lenyomata, azaz melyre $H(x') = H(x)$.
- Ütközés-ellenálló hash függvény
A H hash függvény ütközés-ellenálló, ha nehéz olyan $x, x', x \neq x'$ párt találni, amelyre $H(x') = H(x)$.

Számos hash függvény létezik ([22]).

- SHA (Secure Hash Algorithm), több verzióban érhető el, többek között SHA-0, SHA-1, SHA-2, SHA-3. Manapság a 2, 3-as hash függvényeket használják biztonsági funkciójuk miatt (például SHA-256, SHA-384, SHA-512). A függvény egy szöveges üzenetet, mint például jelszót vagy adatot egy rövid, fix hosszúságú karakterlánccá alakít, amit hash értéknek nevezünk. A bemeneti üzenetet fix méretű blokkokra bontja, majd minden egyes blokkot feldolgoz matematikai műveletek segítségével, hogy egyedi hash értéket hozzon létre az adott blokkokhoz. A hash értékeket összefűzi, hogy a teljes bemeneti üzenet végső hash értékét előállíthassa. SHA-t széles körben használják biztonsági kommunikációs protokollokban, mint például a SSL/TLS, a VPN és az IPsec ([24]).
- MD5 (Message Digest 5), egy hash függvény, amely a szöveges üzenetből egy rövid, 128 bites hash értéket képes generálni. Az MD5 nem teljesíti a

kriptográfiai hash függvények jellemzőit, ezért napjainkban már nem használják ezt az algoritmust. 2004 márciusában indított MD5CRK nevű elosztott projekt keretein belül születésnap (birthday attack) támadással egy ütközés megtalálása volt a cél, ezzel is bizonyítva, hogy az MD5 nem biztonságos kriptográfiai hash függvény. A projekt 2004. augusztus közepén ért véget, amikor Xiaoyun Wang, Dengguo Feng, Xuejia Lai és Hongbo Yu bemutattak egy MD5 ütközést. Analitikus támadások jelentései szerint mindössze egy órát vett igénybe az ütközés megtalálása IBM p690 klasztereken ([24][25][26]).

2. Szimmetrikus kulcson alapuló titkosítási sémák – $E : \{0, 1\}^n \times \{0, 1\}^k \rightarrow \{0, 1\}^n$ transzformáció, ami egy n bites nyílt blokkot és a k bites kulcsot az n bites titkosított blokkba képezi le. Rögzített kulcs mellett, az E transzformációnak természetesen invertálhatónak kell lennie, különben a rejtjeles blokkból nem tudnánk egyértelműen visszaállítani a nyílt blokkot. Az előbb meghatározott blokkrejtjelezőt tehát úgy is definiálhatjuk, mint az $E_K : \{0, 1\}^n \rightarrow \{0, 1\}^n$ invertálható leképezések egy családját, ahol a család egyes elemeit $K \in \{0, 1\}^k$ kulcs választja ([22]). A küldő és a fogadó fél az üzenettitkosításhoz és a visszafejtéshez ugyanazt a kulcsot használják (vagy polinom időben kiszámítható egyik a másiktól). A kulcsok biztosítják a biztonságos adattovábbítást a felek között, mivel csak azok, akik a kulcs tulajdonában vannak, képesek visszafejteni a titkosított üzeneteket. Gyorsabb, hatékonyabb és kevesebb számítási erőforrást igénylő titkosítási eljárás, mint az aszimmetrikus titkosítás ([21][23]).

- AES (Advanced Encryption Standard), egy blokk-alapú titkosítási algoritmus, amely Rijndael algoritmuson alapul. A NIST (National Institute of Standards and Technology) 2001-ben felhívást tett közzé az előző DES (Data Encryption Standard) algoritmus lecserélésének érdekében. Feladata a bizalmas információk rejtjelezéssel történő védelme. Az AES tiszta algebrai struktúrája $GF(2^8)$ aritmetikát alkalmaz, ahol bájt a legkisebb kezelt egység. Az AES titkosítás folyamatának során egy nemlineáris egyenletrendszerre van szükség, amelyet $GF(2^8)$ felett kell megoldani. Az egyenletrendszer a bemeneti üzenet és a kulcs alapján adja meg a titkosított üzenetet. Az üzenet 32 bites (4 bájt) szóból állnak, ennek megfelelően az AES blokkmérete 128, 192 vagy 256 bites

kulcsokkal használják, ami lehetővé teszi a bemeneti adatok blokkonkénti titkosítását ([22][23]).

Titkosításkor rendszerint folyam- vagy blokk kódolást használunk. A blokk kódolás során a nyílt szöveget előre meghatározott hosszúságú blokkokra bontjuk, és ezeket a blokkokat külön-külön titkosítjuk a megfelelő algoritmus segítségével. Így a titkosítás blokkonként történik, és nem karakterenként.

- GCM (Galois/Counter Mode) az AES-el együtt használatos blokk titkosítási mód biztonságos kommunikáció kialakításának érdekében, amely hitelesített titkosítást (Authenticated Encryption) nyújt, és biztosítja az elküldött adatok sértetlenségét és ellenőrzését. A hitelesített titkosításhoz négy bemenet áll rendelkezésre, amik a titkos kulcs, az inicializálási vektor, a nyílt szöveg és az opcionális hitelesítési adatok. Ezen túl két kimenettel rendelkezik, a titkosított adattal és egy hitelesítő címkével, amit üzenethitelesítési kódznak (MAC) is neveznek, amely segíti a címzettet a küldő ellenőrizhetőségében, hogy az adat nem sérült a küldés és továbbítás során ([23]).

3. Aszimmetrikus kulcson alapuló kriptográfiai eljárások – A kriptográfiai algoritmusokban a szimmetrikus kulcsú eljárásokkal ellentétben az aszimmetrikus kriptográfiánál egy kulcspárt használnak, melyek egymástól eltérnek (polinom időn belül nem számítható ki egyikből a másik). A kulcspárt egy privát és publikus kulcs alkotja, ahol a publikus kulcs nyilvánosan ismert, tulajdonos által széles körben terjeszthető, és ezzel titkosítható a küldeni kívánt adat, míg a privát kulcs csak a kulcspár tulajdonosa által ismert, és az adatok visszafejtésére szolgál. Az aszimmetrikus kriptográfiai sémáknak a titkosítás mellett egy másik felhasználási területe a digitális aláírások. A dokumentumokat privát kulcs segítségével írják alá, majd a nyilvános kulcs használatával ellenőrzik azokat. A digitális aláíráshoz használt kulcspár szolgál az adatok integritásának biztosítására, illetve a küldő fél hitelesítésére. Emellett biztosítja még a letagadhatatlanság tulajdonságát. A titkosításhoz és a digitális aláíráshoz egyaránt külön kulcspárt alkalmazunk. Az aszimmetrikus kulcson alapuló titkosítási eljárásokban biztosítva van a kulcsmegosztási mechanizmus a szimmetrikus titkosítási eljárásokkal szemben. Az aszimmetrikus kulcsú titkosítás esetén a nyilvános kulcsú infrastruktúra (PKI) biztosítja a kulcsok hitelesítését és megosztását, míg a szimmetrikus kulcsú titkosítás esetén a kulcsokat előzetesen meg kell osztani a

kommunikáló felek között. Fontos eltérés a szimmetrikus titkosítási sémáktól az, hogy az aszimmetrikus kulcsú titkosítás lassabb és erőforrásigényesebb, mint a szimmetrikus kulcsú titkosítás ([21][22][23]).

- RSA (Rivest-Shamir-Adelman), 1977-ben a három tervezőjéről kapta nevét (Ron Rivest, Adi Shamir és Leonard Adelman). Az egész számok faktorizálásnak problémáján alapul az RSA módszer. Három algoritmusból tevődik össze, amelyek a kulcsgenerálás, a titkosítás és a visszafejtés. Nyilvános kulcsából a privát kulcs nem kiszámítható. Az algoritmus biztonsága függ az algoritmushoz választott prímszám méretétől, ami lehet az ajánlott 2048 bites kulcs, de hosszabb távú tárolásra 4096 bites kulcsot használnak. Támadások ellen széles körben képes megfelelő biztonságot nyújtani ([22][27]).
- ECDSA (Elliptic Curve Digital Signature Algorithm), az ECC (Elliptic curve cryptography) és a DSA (Digital Signature Algorithm) elvein alapuló módszer. Az ECDSA a diszkrét logaritmus problémán alapul. Bitcoin és Ethereum blokkláncokon használatos az adatok hitelesítésére és adatintegritási szolgáltatások elősegítésére. A rendszerben a digitális aláírás előállításához privát kulcsot, míg annak ellenőrzéséhez publikus kulcsot használnak. Az RSA sémához hasonlóan három algoritmusból tevődik össze, mint a kulcsgenerálás, az aláírás és az ellenőrzés. Az ECDSA esetében a kulcsok rövidebbek lehetnek, mint az RSA esetében, amelyek hasonló biztonságot képesek nyújtani ([23]).

3.2. Ethereum

Az Ethereum (ETH) egy nyílt forráskódú, decentralizált blokklánc platform, amely okosszerződést (smart contract) használ. Az Ethereum láncban olyan csomópontok vannak amelyek EVM-eket (Ethereum Virtual Machine) futtatnak és ezek lehetővé teszik az Ethereum állapotgépeinek folyamatos, megszakítás nélküli és változatlan működését, valamint az okos szerződések futtatását. A konszenzus mechanizmus biztosítja, hogy az összes csomópont ugyanazon az állapoton dolgozzon, és elfogadja az egyetlen, hiteles változatot az állapotról. Ezáltal lehetséges az egyének közötti egyetértés megvalósítása. Aki részt vesz ebben az Ethereum hálózatban, annak az EVM állapotáról rendelkeznie kell egy másolattal, ezen felül bárki küldhet kérést az EVM részére, hogy tetszőleges számításokat végezzen rajta. Ilyen fajta

kérések esetében a hálózat többi résztvevőjének ellenőrizni, validálni és végrehajtani kell a kért számításokat. A számítási kéréseket tranzakciós kéréseknek nevezzük, ahol az összes tranzakció és az EVM állapotának nyilvántartása mind a blokkláncon szerepelnek, amelyet az összes csomópont elosztva tárol és egyeztet. A platform natív kriptovalútája az Ether. A Bitcoin (BTC) után az Ethereum a második legnagyobb és legaktívabban használt kriptovaluta a piacon. Egy pont-pont közötti (peer-to-peer) hálózat, ahol a csomópontok a blokklánc működtetésért felelnek konszenzus mechanizmus segítségével. Az Ethereumra jellemző a Turing teljesség, ami egy Alan Turing által meghatározott szó. A Turing teljesség azt jelenti, hogy egy adott számítási rendszer vagy modell képes szimulálni bármely más számítási modellt vagy rendszert. Egy számítási modell csak akkor tekinthető Turing teljesnek, ha képes szimulálni minden olyan algoritmust, amely kiszámítható egy Turing-gépen ([29]). A hálózatok a követelmények és a felhasználás alapján két rétegbe sorolhatók ([21][28][29]).

Publikus hálózatok

- Főhálózat (Mainnet) – Jelenlegi élő és elsődleges nyilvános hálózat, ahol a tranzakciók történnek az elosztott főkönyvben. Hálózati azonosítója és lánc azonosítója 1, ezáltal tudjuk a tényleges főhálózatot beazonosítani. A <https://etherscan.io/> oldal nyújt segítséget a blokkok részletes adatainak, mérőszámainak és maga az Ethereum blokkláncához való felfedezéshez.
- Teszthálózatok (Testnets) – Ethereum tesztelésére létrehoztak teszt alhálózatokat, amelyet protokoll vagy intelligens szerződések fejlesztői használnak. Céljuk az okosszerződések és a Dappok (Decentralized applications) tesztelhetősége még nem éles környezetben. A legtöbb tesztet egy engedélyköteles PoA (Proof-of-Authority) konszenzus mechanizmust használ, ami a csomópontok előre meghatározott listáján alapul, amelyek hitelesítik a tranzakciókat, azonban a PoS (Proof-of-Stake) mechanizmus is használható bizonyos teszteteken, ami tranzakciók feldolgozására és új blokkok létrehozására szolgál a blokkláncban. A PoA gyors tranzakcióátvitelt biztosít, mivel nem kell várni a bányászokra, hogy megerősítsék a tranzakciókat. Gyakran használt teszthálózatok a Ropsten, a Kovan és a Rinkeby voltak azonban napjainkban a Goerli és a Sepolia teszthálózatok használatosak ([29][30]). Az Ethereum tesztetnek teljesen ingyenesek és nem rendelkeznek valós értékkel, azonban Ethereummal való interakcióhoz ETH-ra van szükség. A legtöbb ember a tesztet ETH-t faucetekből szerezheti be, amelyek olyan webalkalmazások, ahol a felhasználók egy

címet adnak meg, amelyre az ingyenes ETH-t küldik. A faucetek bárki által hozzáférhetőek költségmentesen, és lehetővé teszik az Ethereum teszhálózatának kipróbálását, fejlesztését és tesztelését egy biztonságos és kedvező környezetben.

Privát hálózatok

- Fejlesztési hálózatok – Ethereum alkalmazás fejlesztésére szolgál, hogy az alkalmazás kiadását megelőzően az aktuális problémákat, hibákat felderíthessék. Lényegesen gyorsabb iterációt biztosít, mint egy nyilvános teszhálózat.
- Konzorciumi hálózatok – A konszenzus folyamatot a csomópontok előre meghatározott csoportja irányítja. A nyilvános Ethereum hálózat olyan, mint egy nyilvános internet, a konzorciumi hálózat pedig olyan, mint egy privát intranet.

Az Ethereum rendszer fő alkotó elemei ([21][29]):

- Fiókok (Accounts) – Az Ethereum fiók ether (ETH) egyenleggel rendelkező entitás, amely tranzakciót tud küldeni az Ethereum hálózaton. A fiókok típusait tekintve lehetnek felhasználó által kezelt vagy okos szerződést alkalmazók. Képesek ether és tokenek fogadására, tartására és küldésére, illetve interakcióba léphetnek a telepített okosszerződésekkel. Minden fiók egy 20 bájtos címmel és egy hozzátartozó állapottal rendelkezik.
- Fiók állapot (Account state) – Négy összetevőből épül fel. A nonce (number only used once) ami egy véletlen szám, amely védelmet nyújt a visszajátszásos támadás ellen, az egyenleg (balance) a címhez tartozó wei darabszáma, a codeHash ami az EVM-en lévő fiók kódjára utal és a tárolási gyökér (storageRoot), amely a fiók tárolni kívánt tartalmát kódolja.
- Tranzakciók (Transactions) – Alapfogalomként meghatározva a tranzakció egy kriptográfiailag aláírt utasítás. Megjelenhet üzenethívás vagy szerződéslétrehozás formájában. Legegyszerűbb tranzakció az ETH átutalása egyik számláról a másikra. Egy tranzakció több információt is tartalmaz, mint például a feladó címe, a fogadó címe, az aláírás, a nonce érték, a küldeni kívánt adat és a tranzakció végrehajtásához szükséges gáz és díjak.
- Blokkok (Blocks) – Minden tranzakció csoportosítva szerepel blokkokban tárolva, amelyek a lánc előző blokkjának hash értékével vannak ellátva, mivel a kriptográfiai hash függvény biztosítja az egyes blokkok sértetlenségét. Ezáltal bármely blokk

módosítása az egész lánc történetére hatással van, így megakadályozhatóak a különböző támadások a lánc manipulációja ellen.

- Gáz és díjak (Gas and fees) – Minden számítás, amely tranzakció formájában megy végbe az Ethereum hálózaton, az díjat von maga után. A díjat „gáz”-nak nevezett címetben adják meg. A gázárakat gwei-ben mérik, 1 gwei 0.000000001 ETH-nak felel meg. A gázon felül három különböző díjat használnak Ethereum hálózatokon.
 - Alapdíj (Base fee) – Minden blokknak van egy alapdíja, ami tartalékként szolgál. Ahhoz, hogy egy blokk bekerülhessen a láncban, a gázonkénti felajánlott árnak legalább az alapdíjjal kell megegyeznie.
 - Elsőbbségi díj (Priority fee or tip) – Egyfajta borraválóként szolgáló díj a bányászok számára, annak érdekében, hogy az üres blokkok bányászata helyett olyan blokkokat bányásszanak, amelyet előnybe kell részesíteni a konkurens tranzakciók helyett.
 - Maximális díj – A felhasználók által megadott maximális limit, amelyet hajlandók kifizetni egy tranzakció végrehajtásáért. Ez az opcionális paramétert maxFeePerGas-nak nevezik.
- Bányászat (Mining) – Az a folyamat, amely során, a hálózaton dolgozó számítógépek egymással versenyeznek az új blokkok létrehozásáért és a tranzakciók validálásáért.
- Konszenzus mechanizmus – Az ötletek és protokollok teljes halmazát jelenti, amelyek lehetővé teszik, hogy a csomópontok elosztott halmaza megegyezzen a blokklánc állapotáról. Az Ethereum konszenzus mechanizmust használ, ami 2022 előtt a PoW (Proof-of-Work) volt azonban napjainkban már a PoS (Proof-of-Stake) használatos ([29]). A mechanizmus váltásra azért volt szükség, mert a PoS gazdasági szempontból biztonságosabb, hatékonyabb és kevesebb erőforrást igényel, mint a PoW mechanizmus. Kezeli a korrump résztvevőket, mivel a hálózatot azok a résztvevők biztosítják, akik a legtöbb Ethereum token-t birtokolják. Az Ethereum hálózatán belül a legtöbb tőkével rendelkező résztvevők nagyobb érdekeltséggel rendelkeznek a hálózat biztonsága és fenntarthatósága iránt. Ennek következtében kevésbé valószínű, hogy rosszhiszeműen viselkednek, vagy kárt okoznak a hálózatban. Emellett pedig kriptográfiai szempontból is hatékonyabb, mivel a PoS-alapú hálózatokkal szembeni támadásokhoz sokkal nagyobb számítási erőforrásokra lenne szükség, mint a PoW-alapú hálózatokkal szembeni támadásokhoz. A PoS algoritmus használata lehetővé teszi

a hálózat számára, hogy kevesebb számítási erőforrást használjon, és ezzel csökkentse a támadásokkal szembeni sebezhetőségét.

- Proof-of-Stake – A validátorok blokkokat hoznak létre, ahol minden körben véletlenszerűen kiválasztunk egy validálót, aki blokkot javasolhat. Ekkor a PoS egy tranzakciócsomagot kér a párosított végrehajtó kienstől. Ezt konszenzus adatokba csomagolják, hogy egy új blokkot alkothassanak a láncra. Ezt a blokkgyártást ETH-ban jutalmazák. Kriptogazdasági szempontból biztonságos mivel a támadónak, hatalmas mennyiségű ETH-t kellene megsemmisítenie ahhoz, hogy átvegye az irányítást a lánc felett.

3.2.1. Okos szerződés és Solidity

Az okosszerződés egy önmagát végrehajtó számítógépes program, amely képes végrehajtani egy szerződés feltételeit a felek között. Ezek a szerződések sok esetben blokklánc technológián alapulnak, amely egy biztonságos és decentralizált környezetet biztosít a tranzakciók végrehajtásához és tárolásához. Az EVM az Ethereum blokklánc egy olyan alapvető technológiája, amelynek fontos szerepe van okosszerződéseknel. Ez a virtuális környezet lehetővé teszi, hogy a blokkláncon tárolt szerződések végrehajthatók legyenek az Ethereum hálózaton. Az EVM pontosabban egy biztonságos futtató környezet, amely garantálja a kódok megbízható végrehajtását, mindeközben kihasználja a blokklánc előnyeit. Széleskörű felhasználási lehetőséget kínálnak, beleértve a pénzügyi tranzakciókat, a beszerzési láncok kezelését és a digitális azonosítások ellenőrzését. Az okosszerződések elosztott számítógépes programok, amelyek tartalmazzák a szabályokat és a feltételeket. Ha a feltételek teljesülnek a szerződés automatikus végrehajtja a megállapodott intézkedéseket anélkül, hogy emberi beavatkozásra vagy harmadik felek bevonására lenne szükség. Mivel az okosszerződés blokkláncon működik, így megváltoztathatatlanok és manipulálhatatlanok a hash függvényeknek köszönhetően, ezáltal a szerződés feltételei is ugyanezeket a tulajdonságokat örökli. Az EVM-et használó okosszerződések a Solidity programozási nyelvet használják. A Solidity által megvalósulhat az Ethereum blokkláncra való fejlesztés. A Solidity hasonlóan népszerű programozási nyelv, mint a Java vagy a C++, azonban nagyon hasonlít a Javascript nyelvhez. A szerződések programozói által írt kódokat a Solidity Compiler fordítja le az EVM által is értelmezhető Bytecode-ra, amelyet ezután az Ethereum hálózaton lehet futtatni. A Solidity funkcionálisan és szintaktikailag nagyon hasonlít más programozási nyelvekre,

azonban vannak fontos különbségek az Ethereum blokklánc és az okoszerződések környezete miatt. Ide tartoznak az Ethereum-szabványos típusok, mint például az uint, az uint256, a keccak256 és az address típusok, valamint a contract konstruktor, illetve a gáz (gas) mérése, amely a szerződések futtatásához szükséges erőforrást jelöli ([21][28]).

4. Okosotthon biztonsági keretrendszer

Az okosotthoni rendszerek számának folyamatos növekedése meghatározó napjainkban, mivel a rendszerben számos heterogén eszköz vesz részt így nehéz egyesével nyilvántartani az eszközök állapotát és biztonságát. Emiatt szükség van olyan megoldásra, amely egy átfogó információt szolgáltat a rendszerben fellelhető összes eszközzel a felhasználó számára. Kihívást jelent, hogy minden IoT elem rendelkezik valamilyen alap biztonsági mechanizmussal, amely gyártás során hardveresen és szoftveresen már beépítésre került, viszont figyelembe kell venni azt a tényt, hogy az eszközök alapértelmezett beállításai már weben keresztül bárki számára elérhetőek, emiatt az eszközök a kiadástól számítva kompromittálhatónak tekinthetőek. Ilyen például az routerekre való belépéshez szükséges IP cím és felhasználónév jelszó párosa. Ezen tények tudatában az eszközök biztonsági fokát magasabb szintre kell fejleszteni, méghozzá olyan megoldást kell alkalmazni, amely összetett rendszerek esetében képes sokszínű, egységes biztonságot nyújtani. A következőekben az aktuális problémákról és sérülékenységi pontokról, illetve hozzájuk társuló biztonsági javaslatokról fog szó esni.

4.1. Aktualitás, sérülékenységek, motiváció

Tanulmányok feldolgozták az okosotthonokban található IoT eszközök által elérhető biztonsági funkciókat, ahol kutatások során megállapították, hogy nem volt olyan pontja az eszközöknek, amelyet ne lehetett volna hardveresen vagy szoftveresen kompromittálni ([31][32]). Mivel az eszközök egy belső vezérlőegységhez csatlakoznak így a hálózat tagjainak tekinthetőek, innentől a támadó számára elég egyetlen sérülékenységi pontot felfedezni és onnantól átveheti a vezérlést akár az egész rendszer felett. Okosotthonok IoT támadásai következményekkel járnak, amelyek a kibertér határait is túllépik, ezáltal hatással vannak a felhasználók életére és otthonaik biztonságára. Sok esetben nem az okosotthonokat tekintik fő célpontnak, hanem a bennük található bizonyos eszköztípust, egy meghatározott hálózati szolgáltatást vagy felhasználót. A támadások elkerülése végett biztonsági csapatok, etikus hackerek elkezdtek az okosotthon rendszerek behatolás vizsgálatát, ami egy jól ismert, napjainkban is alkalmazott technika. A behatolás vizsgálat magába foglalja az eszközök vizsgálatát, hogy szimulálhassák,

milyen kihívásokkal állnak szemben egy rendszer feltörése során. Ez a módszer segít megérteni és átlátni az okosotthonban lévő IoT eszközökben rejlő hibákat ([31]).

Ilyen behatolás vizsgálatot hajtottak végre a GHOST (Safe-Guarding Home IoT Enviroments with Personalised Real-time Risk Control) projektben. A GHOST [33] a 2020-as évben az Európai Unió által végrehajtott kutatási projekt nemzetközi érdekeket szolgálva, amelynek célja egy kiberbiztonsági réteg kialakítása IoT okosotthonok számára. A rendszer különböző IoT eszközök alkalmazásával kapcsolatos biztonsági és adatvédelmi aggályokat vizsgálja, ahol az okosotthon környezetet kombinálták blokklánc-technológiával. A GHOST fejlesztése és tesztelése két erőforrás korlátozott platformon történt, egy IoT átjárón és egy Raspberry Pi eszközön. A két eszköz használatával több különböző IoT protokoll elemzését tették lehetővé, mint például a Wi-Fi, a Bluetooth Low Energy és a Z-Wave. Ezeken felül különböző beépített komponenseket használ, mint például kiberbiztonsági tudásbázis, blokklánc védelmi infrastruktúra, anomáliaérzékelési keretrendszer és megosztott adattárolás. Laboratórium tesztelések során a lehetséges hibák számát csökkentették a rendszer zökkenőmentes működése érdekében, majd funkcionális tesztkörnyezeteket alakítottak ki, ezzel is szimulálva egy okosotthont, ahol kezdetben 15 majd 25 eszközt (mint például: okoszár, okoslámpa, robotporszívó stb.) csatlakoztattak és felügyeltek a GHOST segítségével. A tesztkörnyezetek lehetővé tették, hogy megvizsgálják az eszközök közötti csomagszintű adatok áramlási folyamatait, az eszközök sérülékenységi pontjait és a beérkező támadásokat. Az okosotthonokat fenyegető potenciális veszélyek kategóriákba sorolhatóak a következők alapján: fizikai támadások, káresemények, katasztrófák (természeti/környezeti), informatikai eszközök károsodása vagy elvesztése, meghibásodások, üzemzavarok, lehallgatások, visszaélések ([33]). A felhasználók csoportjának fontos szerepük van az okosotthon biztonságának megteremtésében. Kiemelt figyelmet kell fordítani az okosotthonok biztonságára, amikor okoseszközöket választunk és konfigurálunk. Az okoseszközök, vezérlőalkalmazások, vezeték nélküli hálózatok és felhőszolgáltatások kiválasztása során a biztonsági szempontoknak elsőbbséget kell élvezniük. A biztonsági intézkedések közé tartozik a biztonságos jelszavak használata és a kétlépcsős azonosítás alkalmazása az okosotthoni rendszerekhez való hozzáférés során. Sérülékenységi pontnak tekinthető a felhasználók nem megfelelő tudása az okosotthon rendszerekben. A felhasználók nem minden esetben vannak tisztában az otthonukban fellelhető IoT eszközök működőképességével és beállításával. Az eszközökhöz tartozó használati utasítás (papír alapú vagy QR-kód által elérhető) részletes leírást biztosít a

felhasználóknak az eszközök megfelelő alkalmazásához és beállításához, azonban sok esetben a rendszereket telepítők nem osztják meg a felhasználókkal ezeket az információkat. Az adathalászat és pszichológiai manipuláció általi veszteségek elkerülése érdekében, a magánszemélynek és a cégeken belül alkalmazottaknak is már bevált gyakorlat a továbbképzés a különböző rendszerek biztonságos és tudatos használata érdekében. A támadások számos formában megjelenhetnek, mint például a spam, phishing, spearphishing, vishing, smishing, vagy a technikai támogatási csalások.

Az elavult eszközök egy másik nagy csoportot képeznek, amely krízis pontot tölt be az okosotthon rendszerekben. Ezek az eszközök hardverparaméterei illetve szoftver specifikációjukat tekintve is sérülékenynek mondhatóak. A gyártók az eszközeiket a gyártástól számítva bizonyos ideig támogatják. Ezen eszközök elromlása üzemzavart is okozhat a hálózatban, ami csekély problémának tekinthető azzal ellentétben, ha egy eszköz elavult szoftvere kompromittálva lesz. Érdemes az eszközöket naprakészen tartani, azaz folyamatos karbantartást és frissítést biztosítani. Gyártó által támogatott eszközök esetében rendszeresen érkeznek frissítések az eszközökhöz, amelyekben javítják a szoftveres biztonsági réseket és hibákat. Ezen felül az okosotthon biztonsági keretrendszerben elengedhetetlen a rendszer folyamatos ellenőrzése és tesztelése, hiszen ezek elősegítik a biztonsági problémák, hibák azonosítását és lehetővé teszik a megfelelő intézkedéseket a védelem javítása érdekében.

Az okosotthoni rendszerek megbízható és biztonságos működésének kulcsa a biztonságos tervezés és telepítés. A megfelelő rendszerarchitektúra kialakításakor fontos a hálózatbiztonság, a rendszerek közötti kommunikáció és az adatok biztonsága. Rendszeres biztonsági mentések készítése szintén kiemelt fontosságú az okosotthoni eszközök és beállítások védelme érdekében. Ha bármilyen biztonsági probléma merül fel, a biztonsági mentés könnyű visszaállítást és eszközkonfigurációt tesz lehetővé. Az eszközök közötti biztonságos kommunikáció érdekében érdemes biztonsági protokollokat és szabványokat alkalmazni, mivel ezek lehetővé teszik a titkosítást, a hitelesítést, az adatintegritást és az adatvédelmet.

4.2. Rendszer résztvevői

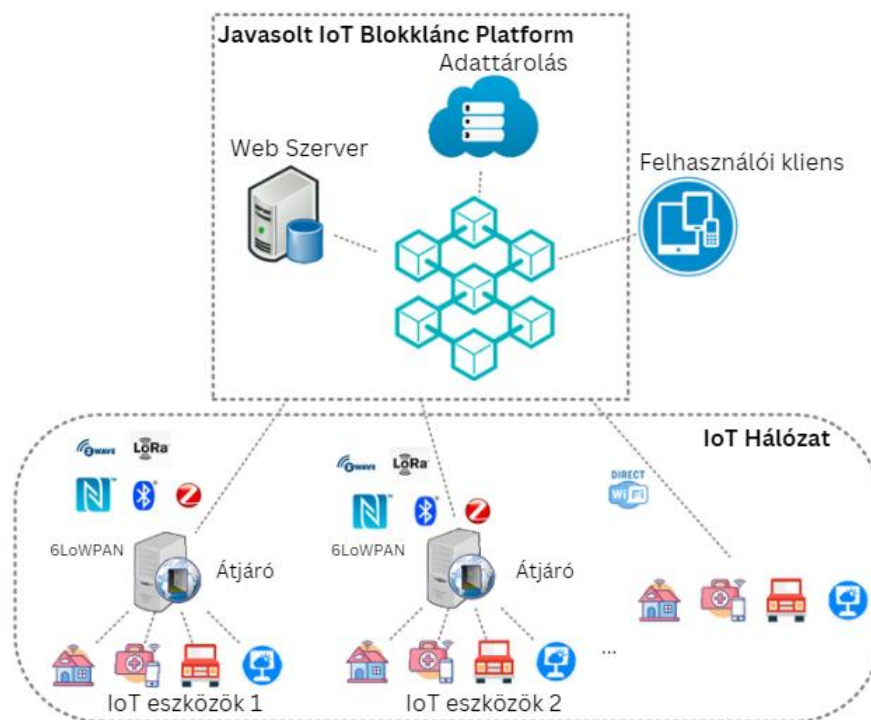
Az okosotthon rendszereknek több résztvevője is van, és minden félnek megvan a maga feladatköre, ami elengedhetetlen egy okosotthon biztonságos működése érdekében:

- **Eszközgyártók** – Az eszköz gyártói felelősek az IoT eszközök tervezéséért, azok biztonsági funkcióinak beépítésért és a biztonsági szabványok betartásáért.
- **Rendszertelepítők** – A telepítők feladata az otthoni rendszerek kivitelezése és konfigurációja, amelyek során biztosítaniuk kell az eszközök biztonsági funkcióinak megfelelő működését, ezen felül feladatuk még a felhasználók tájékoztatása a rendszerük működéséről és egyéb specifikus beállításairól.
- **Szolgáltatók** – A szolgáltatók felelősek az ügyfelüknek nyújtott szolgáltatások biztonságáért, mint például a Nest, amely a fűtés, hűtés, világítás és biztonsági rendszerek automatizálására specializálódott (hasonló szolgáltatók még a SmartThings, Vivint). Feladatuk az adatvédelem, a felhasználói azonosítás és az eszközök védelme.
- **Okoseszközök** – Lehetővé teszik az okosotthon rendszer működését. Az eszközöket megkülönböztethetjük, például típus szerint lehetnek érzékelők vagy beavatkozók, illetve korlátozott vagy magas kapacitással rendelkező eszközök, stb.
- **Központi vezérlő egység** – Ez az eszköz irányítja a többi okoseszközt otthonunkban. Ez lehet egy hétköznapi modem vagy akár egy hangvezérléses okosasszisztens.
- **Felhasználók** – Az okosotthoni rendszer legfontosabb résztvevői. A rendszertelepítők mellett a felhasználók is felelősek az eszközök megfelelő konfigurálásáért, erős jelszavak használataért, az eszközök szoftverfrissítéseinek időben történő telepítéséért és az általános biztonsági előírások betartásáért.

4.3. Javasolt megoldás

Az okosotthonok biztonsági keretrendszerére egy olyan megoldást javasolunk, amely az okos eszközök kiberbiztonsági szintjét magasabb szintre emeli és a hálózatunkban megjelenő okoseszközöket összegyűjti, és folyamatos monitorozást végez rajtuk. Ehhez a keretrendszerhez egy eszközkezelőt javasolunk, amely az IoT eszközök különböző attribútumait egy engedélyköteles blokkláncon elosztott módon tárolja, figyelve a szenzitív adatok titkosított tárolására. Az okosotthon megteremtéséhez elengedhetetlen az IoT eszközök használata, ezért ezeknek aktuális problémáit és sérülékenységeit vizsgáltuk az eddigiekben. Az IoT előnyeit és hátrányait figyelembe véve a blokklánc technológiát alkalmaztuk a biztonság magasabb szintre emelése érdekében, mint például az érzékeny adatok megfelelő tárolására, kezelésére és

továbbítására. A blokklánc lehetővé teszi az adatok elosztott, biztonságos tárolását harmadik fél bevonása nélkül. Megváltoztathatatlan tulajdonsága miatt a blokklánc védi az adatbázist a különböző manipulációk ellen. Az eszközök egyszerű nyomonkövethetőség érdekében kifejlesztettünk egy webes API-t, amely az okosotthon rendszerben fellelhető eszközökhöz tartozó információkat tárolja, mint például az eszköz neve, a MAC címe, a firmware verziója, hálózati ellátás típusa és ezen felül csatolható a rendszerhez az eszközről egy kép és konfigurációs fájl a biztonsági mentés alkalmazása érdekében. Blokklánc használata segítségével olyan funkciókat valósít meg, mint az eszközökhöz tartozó alapértelmezett jelszavak megváltoztatásának ellenőrzése, valamint az eszközök firmware és szoftver verzióinak ellenőrzése. A keretrendszerünk célja, hogy növelje az okosotthonok biztonsági szintjét és figyelmeztesse a felhasználókat, ha az eszközökhöz új frissítések érkeznek. Továbbá a frissítések végrehajtása automatikusan is megtörténhet a rendszer elemein. Ezek mellett a blokkláncon elosztott módon tárolva vannak az okosotthonhoz tartozó biztonsági mentések, ahol az eszközök beállításai és konfigurációi rendelkezésre állnak a különböző kártékony programok ellen. Mivel több helyen és rendelkezésre állnak ezek a biztonsági mentések, amennyiben a rendszer sérül az könnyedén visszaállítható korábbi állapotába. A felhasználói kliens olyan eszköz, amely lehetőséget biztosít a végfelhasználók számára, hogy adatokat olvassanak vagy írjanak a blokklánc hálózatára. Ezek az eszközök lehetnek például okostelefonok, laptopok és számítógépek, amelyeken keresztül a végfelhasználók a blokklánc hálózatra kapcsolódnak. Az IoT eszközök számára számos kommunikációs protokoll áll rendelkezésre, mint például a Bluetooth, ZigBee, Wi-Fi és a 2G/3G/4G mobil hálózatok. Az IoT alapú helyi hálózatokban található átjárók az említett kommunikációs technológiákon keresztül csatlakoznak a szerverhez, és szolgáltatásközvetítőként is működnek a hálózatban lévő eszközök számára.



4.3.1 ábra IoT eszközök és blokklánc kapcsolata

5. Biztonsági kérdések

A biztonsági kérdések vizsgálatának célja, hogy egy képet kapjunk a rendszerünk biztonságához kapcsolódó kérdésekről és a hozzájuk társuló megoldásokról. Az előző fejezetek során azonosítottuk az IoT ökoszisztéma és blokklánc-technológia elemeit és megértettük működésüket. Figyelembe vettük a hozzájuk társuló veszélyforrásokat és fenyegetéseket, amelyek hatással lehetnek a rendszerünk működésében. A következő fejezetben összegzésre fog kerülni a dolgozathoz készült implementáció biztonsági követelményei, hogy mely tulajdonságoknak kell teljesülnie blokklánc alapú IoT rendszerek esetében, illetve a biztonság milyen formában valósul meg a rendszerünkben.

5.1. Biztonsági követelmények

A megfelelő biztonsági szint elérése érdekében szükséges követelmény rendszert alkotni, amely minden esetben igyekszik a felhasználó és a rendszer között azokat megfeleltetni. Az

egyszerűbb átláthatóság érdekében szempontok szerint igyekszünk ismertetni a szükséges követelményeket a [34] kutatást alapul véve.

- *Hitelesítés*

A rendszerhez hozzáférni kívánt felhasználó kilétéről való meggyőződés és a megbízható hozzáférést a hitelesítés biztosítja. Lehetővé teszi a felhasználók azonosítását, ezzel megakadályozva a jogosulatlan hozzáférést. Hitelesítés többféle módszer segítségével megvalósítható, ahogy a korábbi 2.1.3 fejezetben is említésre került az egy és kétfaktoros hitelesítési metódusok, a biometrikus azonosítás vagy a digitális okmányok használata. A hitelesítésnek a webalkalmazáshoz és ezáltal eszközökhöz való csatlakozásban, illetve ezeken való műveletvégrehajtásokban is kiemelt szerepe van.

- *Bizalom*

A személyes adatok és az eszközök attribútumainak védelmét biztosítani kell az adatok tárolása, feldolgozása és továbbítása során. Az adatok csak azok számára hozzáférhetőek, akiknek jogosultságuk van, így fontos a hozzáférési jogosultságok megfelelő kezelése, ezen felül az érzékeny adatok titkosítása. Adatok titkosított tárolása miatt csak azok a felhasználók érhetik el ezeket az adatokat, akik blokkláncnak tagjai és rendelkeznek a megfelelő visszafejtő kulccsal.

- *Integritás*

Az integritás fontos az adatok és rendszerek megfelelő működése érdekében, teljességre és megbízhatóságra utalva. Az adatoknak a tárolás, a feldolgozás és a továbbítás fázisaiban is sértetlennek kell maradniuk, a rendszereknek pedig az elvártnak megfelelően, illetéktelen módosításoktól mentesen kell működniük. Fontos az adatok rendszeres ellenőrzése a tárolás, a továbbítás és a küldés folyamataiban is. Az adatokat megfelelő tárolása és védelme érdekében biztonsági mentéseket szükséges készíteni.

- *Rendelkezésre állás*

Biztosítja az arra jogosult felhasználók számára a rendszerben elérhető szolgáltatások elérhetőségét és az adatok rendelkezésre állását a felhasználó által alkalmazni kívánt időpontban, az esetleges hibák vagy leállások esetén is. Ezen felül biztosítja a megfelelő redundanciát és mentéseket a rendszer és adatok számára. A rendelkezésre állás biztosításának érdekében szükséges a rendszerek időszakos karbantartása, frissítése és ellenőrzése a hibák és sebezhetőségek feltárása érdekében.

- *Letagadhatatlanság*

A rendszerben résztvevő felhasználók vagy illetéktelen belépők által végrehajtható műveletek nyomon követése, annak érdekében, hogy bármilyen rendellenesség esetében bizonyíték álljon rendelkezésünkre. A letagadhatatlanság megvalósításával a behatolások detektálása naplózható és megelőző intézkedések kapcsolhatóak hozzá (például rosszindulatú felhasználók kizárása a rendszerből). Az eszközök működésének és a rendszerben történő eseményeknek a naplózása során is kritikus az így keletkezett log fájlokban lévő adatok forrásához kapcsolódó letagadhatatlanságot biztosítani.

5.2. Biztonsági megoldások és a kriptográfiai primitívek által nyújtott védelem

A rendszerünk alkotóelemeinek biztonságát és a biztonsági követelmények teljesülését az alábbi biztonsági megoldások és kriptográfiai primitívek biztosítják.

- *Hitelesítés*

Az implementációnkhoz készült webalkalmazás megköveteli a hitelesítést, amely megakadályozza az illetéktelen belépéseket, illetve lehetővé teszi a rosszindulatú támadások elleni védelmet. Az implementációnkban az adott felhasználói fiók csak a megfelelő felhasználó név és jelszóval érhető el. Emellett a felhasználónak a blokklánc hálózat tagjának kell lenni ahhoz, hogy hozzáférjen a blokkokhoz, amely megköveteli a regisztrációt és az azt követő autentikációt egyaránt.

- *Bizalom*

Az implementációnk elősegíti az IoT eszközök monitorozását a biztonsági kockázatok csökkentése érdekében. A bizalom biztosítása érdekében a felhasználók és a webszerver közötti kommunikációhoz TLS (Transport Layer Security) protokollt alkalmazunk. A blokklánc hálózatban minden tranzakció nyilvánosan látható. Ezért a blokkláncban lévő tranzakciókat, amelyek érzékeny adatokat tartalmazó attribútumokat tartalmaznak AES-GCM blokktitkosítási algoritmussal titkosítunk. Az adatok titkosságát így csak a hitelesítés után a visszafejtő kulccsal rendelkező résztvevők képesek megismerni.

- *Integritás*

A rendszerünk az eszközök frissítésére és karbantartására, illetve az alapértelmezett jelszó megváltoztatására hívja fel a felhasználó figyelmét. Ha egy támadó képes az adatok átvitele során vagy tárolásánál ezen adatok manipulálására, akkor súlyos biztonsági incidenseket képes okozni, például nem megfelelő jelszóbeállítást vagy elavult szoftver vagy eszközkomponensből fakadó hiányosságokat kihasználni. Ahhoz, hogy megelőzzük az illetéktelen módosításokat az implementációban több kriptográfiai eljárás is alkalmazásra kerül, többek között a digitális aláírás az előbb említett TLS protokollnál vagy a tranzakciók aláírásánál (a tranzakciók az okosotthonhoz kapcsolódó különböző funkciókat és adatokat tartalmazza). Emellett fontos megemlíteni az SHA-256 hash függvény használatát a blokklánc elemeinek módosíthatatlanságának eléréséért.

- *Rendelkezésre állás*

Az IoT eszközök rendelkezésre állása a megbízható és stabil hardver- és szoftverkomponensek alkalmazásán alapul. Az általunk javasolt keretrendszerben az eszközök és a hálózatok redundanciája és a keret skálázhatósága segít abban, hogy a rendszer folyamatosan működjön, még akkor is, ha egyes eszközök vagy hálózati elemek meghibásodnak. A rendelkezésre állás fontos eleme a blokklánc-hálózat, amely támogatja az elosztott tárolást és alkalmazást, illetve magába foglalja az adatok tárolását és az okosotthon futtatását több csomópont között. Ennek célja az adatok és az okosotthon funkciók redundáns tárolása és a rendelkezésre állásának biztosítása a hálózatban. Mivel az elosztott tárolás és alkalmazás lehetővé teszi az adatok és függvények szétszórását a hálózatban ezért lehetőség van arra, hogy a hálózaton belüli replikációkból rövid idő alatt helyreállítsuk azokat, például, ha egy adat elveszik vagy kártékony program miatt elérhetetlenné válik. Ezáltal az elosztott tárolás és alkalmazás segít biztosítani a rendelkezésre állást és minimalizálni a leállásokat. Az általunk javasolt keretrendszer a blokklánc alkalmazásával emellett növeli a védelmét a kártékony programokkal szemben. Mivel az adatok és a függvények elosztottan vannak tárolva a blokkláncon, így hiába fertőzi meg a kártékony program az okosotthon rendszerünket és akár több csomópontot egyidőben, a konfigurációs fájlok felhasználásával gyorsan visszaállíthatóvá válik a rendszer a korábbi, fertőzés előtti állapotába.

- *Letagadhatatlanság*

Az IoT esetében a letagadhatatlanság biztosítása érdekében a rendszernek nyomon kell követnie az eszközök és a felhasználók által generált adatokat. Az adatokat meg kell erősíteni és hitelesíteni, hogy a rendszerben részt vevő valamennyi fél biztos lehessen az adatok helyességében és hitelességében. A blokklánc megváltoztathatatlan tulajdonságának fontos része, hogy egyúttal biztosítja a letagadhatatlanságát is. Hitelesített felhasználók által végrehajtott műveletek a lánc főkönyvében kerülnek tárolásra, ezáltal minden tevékenység visszakereshető a lánc történetében. A blokklánc technológia a digitális aláírások segítségével tehát biztosítja az adatok integritás védelme mellett az adateredetintegritást is. Ezen kívül a blokklánc hálózatban a konszenzus mechanizmusok által biztosítva van, hogy a tranzakciók csak akkor hajthatók végre, ha az megbízható résztvevők egyetértenek, így elkerülhető, hogy a támadók manipulálják a hálózatot és hamis okosotthoni beállításokat és funkciókat állítsanak be.

6. Implementáció

A dolgozatomban célja egy olyan összetett rendszer megalkotása volt, amely egy új blokklánc alapú biztonsági keretrendszert valósít meg IoT eszközökre okosotthon környezetben. Felmértük az IoT eszközök tulajdonságait és számba vettük a blokklánc technológia előnyeit, majd egy webes alkalmazást készítettünk, amely lehetővé teszi az okosotthonokban fellelhető IoT eszközök nyilvántartását és karbantartását. A webes API segíti a felhasználókat az eszközeik folyamatos monitorozásában és naprakészen tartásában anélkül, hogy rendszergazdai vagy telepítői beavatkozásra lenne szükség. A rendszerhez hozzárendelhető az otthonunkban fellelhető összes IoT eszköz és az eszközökhöz tartozó összes alapértelmezett információ, amely a webes felületen egy oldalon az eszközök egymástól elkülönítve megtalálhatóak. Ezen felül a rendszer képes összevetni az eszköz által jelenleg telepített firmware azonosítóját az interneten található jelenleg legfrissebb firmware azonosítóval és automatikusan vagy szükség esetén kézi vezérléssel frissíti azt.

Fontos még megemlíteni a rendszerünkben használt blokklánc-technológiához kapcsolatos konszenzus mechanizmust. Lehetővé teszi a csomópontok számára, hogy a résztvevők halmaza megegyezzen a blokklánc állapotáról. A mi rendszerünk által alkalmazott teszhálózat Proof-of-Work konszenzust használ, de egy új privát blokklánc kialakítása esetén érdemes megfontolni az FBA (Federated Byzantine Agreement) konszenzus mechanizmus használatát, hiszen jól ismert az alacsony költségű tranzakcióiról, skálázhatóságáról és nagy áteresztőképességéről. Működését tekintve a BFT-hez (Byzantine Fault Tolerance) hasonlít, azaz, ha a csomópontok tetszőleges adatokat generálnak a BFT abban az esetben is megoldja a konszenzus elérésének problémáját. A BFT és ezáltal a FBA is garantálni tudja a rendszer biztonságát a tranzakciók folyamatos ellenőrzésével, hitelességüknek és adatintegritásuk védelmével. A csomópontok hibás vagy rosszhindulatú működése esetén is képes 33%-ig tolerálni a felek közötti ellenkezéseket. A csomópontokat előre ismerni és ellenőrizni kell, mielőtt a felhasználó bármilyen titkosítást, digitális aláírást vagy hash függvényt kér az FBA-tól. A jegyző csomópont (notary node) kiválasztja azokat a csomópontokat, amelyekben megbízik, ezáltal létrehozza a csomópontok csoportját kialakítva az FBA hálózatot.

Összehasonlításképpen az FBA-hoz képest a PoW mechanizmus 50%-os toleranciával rendelkezik, míg a PoS 33%-al, azonban általánosan elmondható mind PoW és PoS mechanizmusokról, hogy megvalósításuk és fenntartásuk bonyolult és nehéz folyamat. A PoW

esetében résztvevő bányászoknak számítógépes hardverekre van szükségük, amelyek jelentős költséggel járnak. Emellett a bányászoknak jelentős mennyiségű elektromos energiára is szükségük van a számítási feladatok elvégzéséhez, ami további költségeket jelent. A PoS esetében a résztvevőknek bizonyos összeget kell letétbe helyezniük, amely biztosítja az ellenőrzési jogokat a hálózaton. Ez a folyamat szintén bonyolult lehet, mivel biztosítani kell, hogy a letétbe helyezett összeg megfelelően legyen kezelve, és a validátorok által megfelelően legyen ellenőrizve ([30]).

6.1. Komponensek

A rendszer és a hozzá társuló forráskód több alapvető részből épül fel, amelyek közül bármely technológia hiányában a rendszer nem alkotna teljes biztonsági keretrendszert. A forrás kódot alkotó technológiák:

- *Node.js*

A Node.js egy nyílt forráskódú szoftver platform, amely skálázható internetes alkalmazások, webszerverek készítésére hoztak létre. A Node.js egy V8 Javascript motorra épül, amely eseményalapú, aszinkron bemenettel és kimenettel rendelkezik a túlterhelés minimalizálása és a skálázhatóság maximalizálása érdekében. Az NPM (Node Package Manager) nevű eszköz segítségével a fejlesztők könnyen telepíthetnek és frissíthetnek modulokat és függőségeket ([35]).

- *Solidity*

A Solidity egy objektum-orientált programozási nyelv. A Solidity egy olyan programozási nyelv, amely az Ethereum blokklánc platformon történő okos szerződések programozására szolgál. C++ és Javascript nyelvekből épül fel, amely lehetővé teszi a fejlesztők számára az Ethereum láncra való alkalmazások fejlesztését. Az okos szerződések programozható logikát és állapotokat tartalmaznak, mint például a változók, a függvények, az objektumok, az öröklődés és az interfészek ([36]).

- *React*

A React egy nyílt forráskódú, deklaratív frontend fejlesztésére használatos nyelv. A webes API felhasználói felületének megjelenítéséhez és kinézetének testre szabásához a React JavaScript könyvtárat alkalmaztuk, ami által különböző komponenseket jeleníthetünk meg, mint például gombok, szöveg dobozok, űrlapok ([37]).

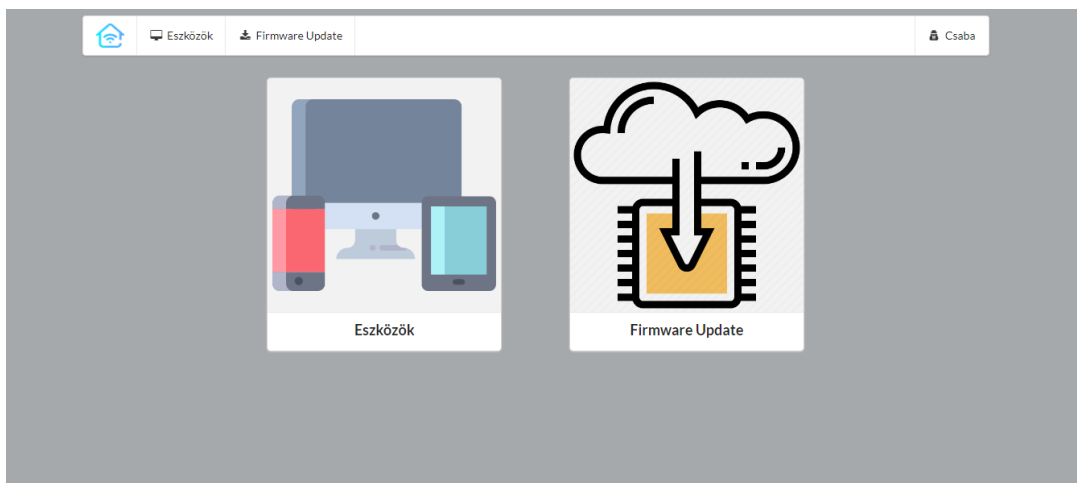
- *Next.js*

A Next.js egy nyílt forráskódú webfejlesztői keretrendszer, amely lehetővé teszi a React alapú web API-k alkalmazását szerveroldali rendelések és lokális webhelyek generálása érdekében. Támogatja az automatikus kódgyorsítást és adatlekérdezést, amely által a React alapú alkalmazások számára a hatékonyabb és gyorsabb működést biztosít ([38]).

- *Metamask*

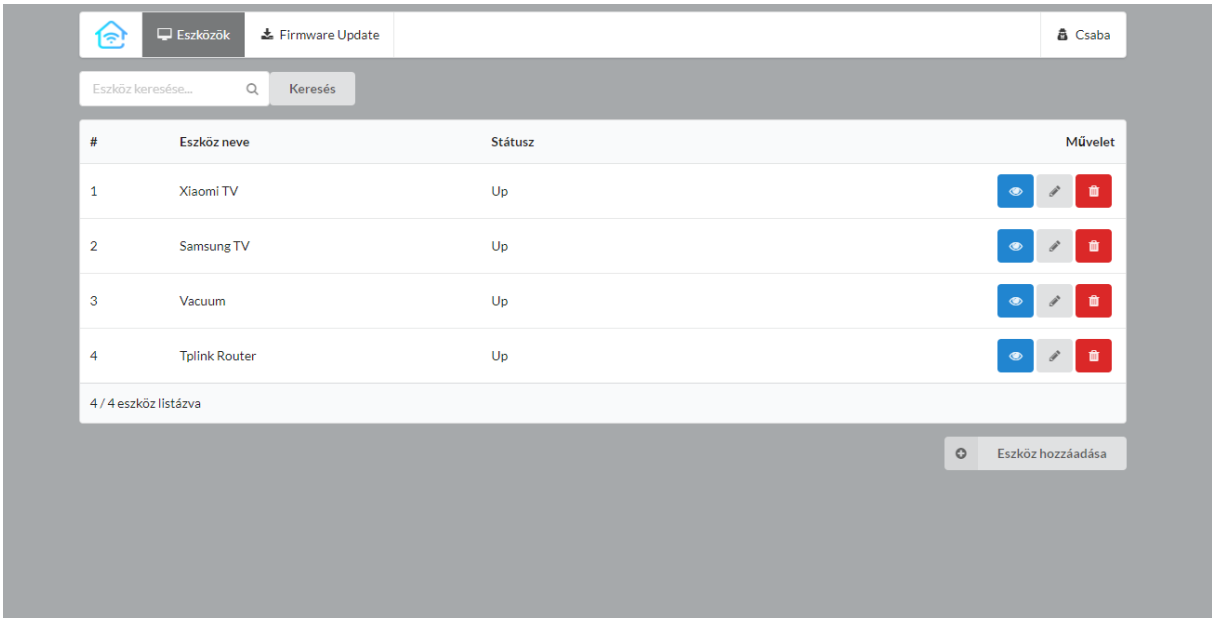
A Metamask egy böngésző bővítmény formában elérhető, ingyenes kriptovaluta pénztárca. A Metamask a felhasználók számára a fiókcímek tárolását és kezelését, valamint az Ethereum alapú kriptovaluták és tokenek küldését és fogadását segíti elő. Emellett a Metamask biztonságos csatlakozást biztosít a decentralizált alkalmazásokhoz webböngészőn vagy mobilalkalmazások beépített böngészőjén keresztül. Legfontosabb előnye, hogy nem igényel technikai ismeretet, így bárki által könnyen hozzáférhető és alkalmazható ([39]).

A különböző technológiákat felhasználva, készült el a dolgozathoz tartozó prototípusalkalmazás. Az alkalmazást a NPM könyvtáron belüli `npm run dev` paranccsal indíthatjuk el, ezt követően `http://localhost:3000/` címen keresztül férünk hozzá a felhasználói felülethez. A Metamask fiók hitelesítése után megtörténik a bejelentkezés ahol 6.1.1 ábrán látható felületet érjük el. Erről az oldalról elérhetjük az Eszközök, illetve az eszközökhöz tartozó Firmware Update oldalakat.



6.1.1 ábra Bejelentkezési felület

Az Eszközök oldalra belépve tekinthetjük meg a hálózatunkra felcsatlakoztatott eszközöket, amelyek monitorozás alatt állnak a rendszerünk által. A rendszerhez statikusan adhatóak hozzá az eszközök, amelyet az egyszerű demonstráció érdekében négy próbaeszközzel valósítottunk meg, amelyek két darab okostelevízió, egy okosporszívó és egy router. Az oldalon megtekinthetjük az eszköz nevét és státuszát, hogy a hálózatunkban jelenleg aktív (UP) vagy passzív (DOWN) állapotot töltenek be. Ezeken felül az eszközökhöz tartozik három műveleti egység, amik a megtekintés, a szerkesztés és a törlés. A megtekintés oldalon részletesebb információkat tudhatunk meg, mint például az eszközről egy kép, a MAC cím, a Firmware azonosító, a konfigurációs fájl, a napló fájl és az eszköz tulajdonosának Metamask címe. A szerkesztés fülön az eszközök adatait vagyunk képesek módosítani. Illetve a Törlés fül, amely által az adott eszközt képesek vagyunk eltávolítani rendszerünkéből. Az Eszközök oldalt a 6.1.2 ábrán tekinthetjük meg.



#	Eszköz neve	Státusz	Művelet
1	Xiaomi TV	Up	  
2	Samsung TV	Up	  
3	Vacuum	Up	  
4	Tplink Router	Up	  

4 / 4 eszköz listázva

Eszköz hozzáadása

6.1.2 ábra Eszközök felület

Fontos már ezen a ponton megjegyezni, hogy a rendszer által biztosított műveletek mindegyike (hozzáadás, módosítás, törlés) esetében a műveletek okosszerződés által biztosított tranzakciók formájában történnek és minden tranzakciónak költsége van. Ebből kifolyólag a Metamask fiókunknak rendelkeznie kell Ethereum egyenleggel. A rendszert az Ethereum teszt hálózatán valósítottuk meg a Sepolia teszhálózat segítségével. A hálózatra a <https://sepoliafaucet.com> oldal segítségével vagyunk képesek saját erőforrásunk segítségével teszt Ethereumot bányászni, ami egy kisebb átfutási idővel jóváírásra kerül a Metamask fiókunkban.

Amennyiben a rendszerben valamilyen módosítást végeztünk arról egy tranzakció készül, amit a <https://sepolia.etherscan.io> oldalon keresztül tudunk nyomon követni. Egy példa 6.1.3 ábrán, ahol új eszköz hozzáadásának tranzakciója tekinthető meg.

The screenshot shows the Etherscan interface for a Sepolia Testnet transaction. The transaction is successful and has 22 block confirmations. The transaction hash is 0x52eb27f3fcbcf02f2390c7ff7d208447b343cfbe790d0f07b1629983575ce01. The transaction was sent from 0x92874F71A920821006F018be4086Ab6CDd2aF6f to 0x8ec3825Ab797417b591f3D1785Fb8Fb4A5b39a. The value is 0 ETH (\$0.00) and the transaction fee is 0.000161677500970065 ETH (\$0.00). The gas price is 2.500000015 Gwei (0.000000002500000015 ETH).

Field	Value
Transaction Hash	0x52eb27f3fcbcf02f2390c7ff7d208447b343cfbe790d0f07b1629983575ce01
Status	Success
Block	3161651 (22 Block Confirmations)
Timestamp	4 mins ago (Mar-25-2023 04:55:12 PM +UTC)
From	0x92874F71A920821006F018be4086Ab6CDd2aF6f
To	0x8ec3825Ab797417b591f3D1785Fb8Fb4A5b39a
Value	0 ETH (\$0.00)
Transaction Fee	0.000161677500970065 ETH (\$0.00)
Gas Price	2.500000015 Gwei (0.000000002500000015 ETH)

6.1.3 ábra Etherscan oldalon található tranzakció részletei

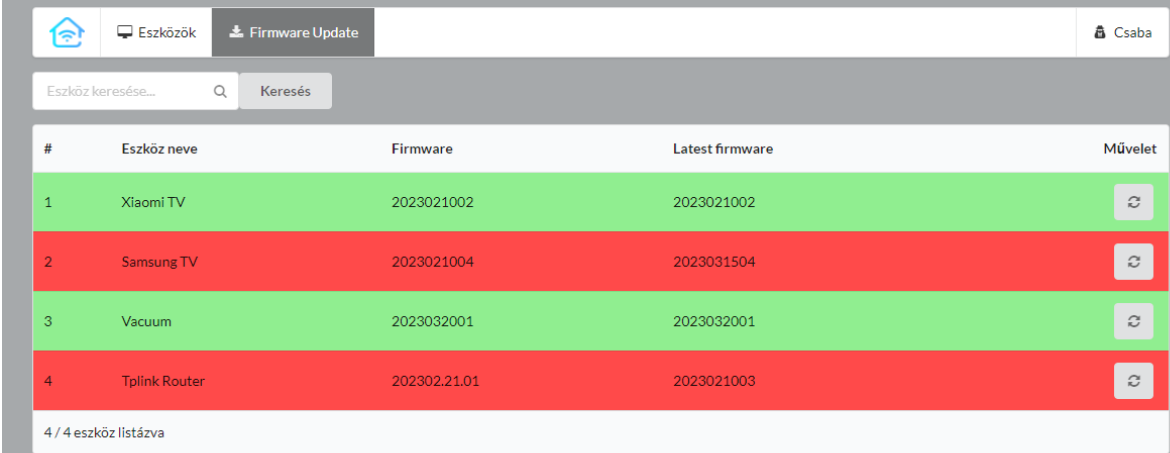
A kezdő oldalról tovább juthatunk még a Firmware Update oldalra, ahol az eszközökhöz tartozó nevet, jelenlegi firmware azonosítót és az interneten a gyártók által megadott legfrissebb firmware verziót láthatjuk, amelyet a 6.1.4 ábra mutat.

#	Eszköz neve	Firmware	Latest firmware	Művelet
1	Xiaomi TV	2023021002	2023021002	
2	Samsung TV	2023021004	2023031504	
3	Vacuum	2023.02.30	2023032001	
4	Tplink Router	202302.21.01	2023021003	

4 / 4 eszköz listázva

6.1.4 ábra Firmware Update felület

Az eszközök sorában lévő szín jelzések figyelmeztetik a felhasználót az eszközeik állapotáról. A zöld jelzés mutatja, hogy az eszköz jelenleg a legfrissebb firmware verzióval, míg a piros jelzés arra utal, hogy az eszköz nem a legújabb firmware verzióval rendelkezik. A firmware verziók frissítéséhez tartozik egy művelet gomb, amelynek megnyomása esetén a firmware verzió eszközönként felülíródik a 6.1.5 ábrán is látható Latest firmware verzió számával.



The screenshot shows a web interface for firmware updates. At the top, there are tabs for 'Eszközök' (Devices) and 'Firmware Update'. Below the tabs is a search bar with the text 'Eszköz keresése...' and a 'Keresés' button. The main content is a table with the following columns: '#', 'Eszköz neve' (Device name), 'Firmware', 'Latest firmware', and 'Művelet' (Action). The table contains four rows of data, each with a green or red background indicating the update status. The 'Művelet' column contains a circular arrow icon for each device.

#	Eszköz neve	Firmware	Latest firmware	Művelet
1	Xiaomi TV	2023021002	2023021002	
2	Samsung TV	2023021004	2023031504	
3	Vacuum	2023032001	2023032001	
4	Tplink Router	202302.21.01	2023021003	

4 / 4 eszköz listázva

6.1.5 ábra Firmware Update felület eszköz frissítését követően

A dolgozatban szereplő implementáció forráskódja megtalálható a <https://github.com/ncsn/SmarthomeBlockchain> oldalon.

7. Összefoglaló

A dolgozatom célja az volt, hogy egy blokklánc alapú biztonsági keretrendszer demót mutasson be IoT eszközökre alkalmazva. A dolgozat második fejezetében feltérképezésre került az Internet of Things rendszerek és eszközök széleskörű alkalmazási lehetőségeit, amelyek lefedik az életünk minden területét. Végigvettük az IoT rendszerek architektúrális felépítését, az IoT rendszerek előnyeit és hátrányait. Elengedhetetlen volt még az IoT eszközök biztonságát tárgyalni, hiszen a kisebb és a komplex rendszerek is sok kis különböző eszköz összeségéből épülnek fel. Ezáltal a legkisebb alkotó elem biztonságát is magas szinten kell kezelni, azonban a legtöbb IoT eszköz korlátozott erőforrással rendelkezik, ami miatt nem alkalmazható hagyományos kriptográfiai megoldások. Ahogy számos incidens is mutatta (Mirai, Gafgyt, Hajime, Tsunami), sokszor alapvető biztonsági beállítások hiányoznak (például jelszó beállítás), ezért az IoT eszközök már a gyártástól való kiadásuktól kompromittáltak lehetnek. Meghatároztuk az okosotthonok követelményeit és céljait, továbbá példák keretein belül ismertettük az okosotthon ökoszisztémát. Felmértük a blokklánc-technológia fontosságát napjainkban. Egy engedélyköteles blokkláncot alkalmaztunk, ahol az érzékeny adatokat tartalmazó attribútumokat titkosítva tároljuk. A keretrendszer megalkotása során egy eszközkezelőt valósítottunk meg, amely az IoT eszközök különböző attribútumait tárolja, mint az eszköz neve, a szoftver/firmware verzió, a típus, a tulajdonos és az állapot. Ezek mellett ismertetésre került az Ethereum és okosszerződések alkalmazási és működési feltételei a biztonsági keretrendszerünkben. Átfogó képet alkottunk a napjainkban fellelhető aktuális problémákról és sérülékenységekről. Taglaltuk az okosotthon rendszer résztvevőit és javaslatot készítettünk a saját implementációnkról más rendszerekkel összevetve. Megadtuk a biztonsági követelményeket, illetve az azokat biztosító biztonsági megoldásokat és kriptográfiai primitíveket. A blokklánc alapú keretrendszer biztonságos, mivel megköveteli a felhasználó hitelesítését és a biztonságos csatornát a felhasználó és a rendszer között a TLS protokoll alkalmazásával. A blokklánc használata biztosítja az adatok elosztott tárolását, és az okosszerződések alkalmazása lehetővé teszi az elosztott alkalmazások nagyobb rendelkezésre állását. Ennek köszönhetően az okosotthon beállításai és konfigurációi könnyen helyreállíthatók kártékony vagy zsaroló programok fertőzése esetén. A blokkláncon tárolt adatok publikusak, de a szenzitív adatok titkosítva vannak, és csak a megfelelő visszafejtő kulccsal rendelkező résztvevők képesek hozzáférni azokhoz. A hatodik fejezetben bemutatásra

került, a dolgozathoz készített implementáció, amelyben áttekintést kapunk a rendszerünk működéséről, céljáról és az azt alkotó komponenseiről.

8. Köszönetnyilvánítás

Szeretnék köszönetet mondani a konzulensemnek, Dr. Oláh Norbertnek, aki a szakkollégiumi csatlakozásom óta folyamatos támogatásával, bizalmával és munkájával végig kísért engem ezen az úton, és általa egy számomra teljesen új környezetbe nyerhettem betekintést. Nagyon hálás vagyok, minden perc felém szánt ideje miatt.

Tisztaszívvvel köszönöm a családomnak a sok gondoskodást, a lelki és anyagi támogatást és a folyamatos biztatást, amely végig kísért egyetemi tanulmányaim során. Hálásan köszönöm barátnőm kitartó türelmét, biztatását, gondoskodását és a mindennapi feladatokban nyújtott segítségét.

Külön köszönöm barátaimnak, hogy egyetemi éveim során végtelen kitartással és buzdítással támogatták céljaim elérését, és nélkülözhetetlen tanácsokkal valamint segítséggel hozzájárultak a dolgozat megírásához. Mindez nagymértékben hozzájárult a sikeres dolgozat létrejövetelében.

9. Irodalomjegyzék

- [1] Kambourakis, G., Kolias, C., & Stavrou, A., The mirai botnet and the iot zombie armies. In *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*, (2017, October), IEEE, pp. 267-272.
- [2] Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J., DDoS in the IoT: Mirai and other botnets. *Computer*, 50(7), (2017), pp. 80-84.
- [3] Pradyumma G., Omkar B. & Sagar B., Introduction to IoT. *Internation Advanced Research Journal in Science, Engineering and Technology (IARJSET)*, (2018), pp. 1-2.
- [4] Eryk S., Andy A., Jara F., Jonathan S., Michael Z. & Burkhard S., Landscape of IoT security. *Communication Systems Group CSG, Department of Informatics IfI, University of Zürich*, (2022).
- [5] Ola S., Imad E., Ali C., & Ayman K., Software Defined IoT Security Framework. *American University of Beirut*, (2020), pp. 1-2.
- [6] Apostolos G., Leandros M., Mohamed A. & Nick A., IoT, Communication protocols and security threats, (2023).
- [7] Microsoft Azure, IoT-technológiák és –protokollok, Forrás: <https://azure.microsoft.com/hu-hu/solutions/iot/iot-technology-protocols/>, (2023).
- [8] Paul B. & Marijn J., Effects of the Internet of Things (IoT), Systematic review of the benefits and risks. *Delft University of Technology, Delft, The Netherlands*, (2015).
- [9] A. Huszti, Sz. Kovács, N. Oláh, Scalable, password-based and threshold authentication for Smart Homes, *Int. J. Inf. Secur.* 21, (2022), pp. 707-723.
- [10] A. Bagherzandi, S. Jarecki, N. Saxena, Y. Lu, Password-protected secret sharing. In: *ACM Conference on Computer and Communications Security* (2011), pp. 433-444.
- [11] S. Jarecki, A. Kiayias, H. Krawczyk, Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In *International Conference on the Theory*

and Application of Cryptology and Information Security, Springer, Berlin, Heidelberg. (2014), pp. 233-253.

- [12] D. Isler, A. Kúpcú, Threshold single password authentication. ESORICS Data Privacy Management, Cryptocurrencies and Blockchain Technology, (2017), pp. 143-162.
- [13] D. Isler, A. Kúpcú, Distributed Single Password Protocol Framework. IACR Cryptol. ePrint Arch., 976., (2018).
- [14] A. Huszti, N. Oláh Provably Secure Scalable Distributed Authentication for Clouds. Lecture Notes In Computer Science 12579., Chapter 10, (2020), pp. 188-210.
- [15] Á. Vécsi, A. Bagossy, A. Pethő, Cross-platform identity-based cryptography using WebAssembly. Infocommunications journal, 11(4), (2019), pp 31-38.
- [16] Cs. Bertók, A. Huszti, T. Kádek, Zs. Jámboor, A multi-round bilinear-map-based secure password hashing scheme. In 2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS), IEEE, (2022), pp. 28-33
- [17] X. Boyen, Hidden credential retrieval from a reusable password. In: *Proceedings of the 4th International Symposium on Information, ACM*, (2009), pp. 228-238.
- [18] Davit M., Savvas P. & Eleftherios A., A systematic review of the smart home literature: A user perspective. *New Castle University Business School, United Kingdom*, (2019).
- [19] Nemzeti Kibervédelmi Intézet (NKI), Otthoni hálózatok biztonsága, (2022), pp. 5.
- [20] Designers: Home Smart IoT Home: Domesticating the Internet of Things, Forrás: <https://www.toptal.com/designers/interactive/smart-home-domestic-internet-of-things>, (2023).
- [21] Md Ashraf Uddin, Introduction to Blockchain Technology. *Federation University Australia, Jagannath University*, (2021), pp. 1-2, 4, 14-20.
- [22] Buttyán L., Vajda I., Kriptográfia és alkalmazásai, Typotex kiadó, (2005).

- [23] Imran Bashir, Mastering Blockchain. *Packt Publishing Ltd. Birmingham*, (2020), pp. 63-78.
- [24] John B., Cochran M. & Trevor H., A Study of the MD5 Attacks: Insights and Improvements. (2006).
- [25] Philip H., Michael P. & Gregory G., Musings on the Wang et al. MD5 Collision. *Qualcomm Australia*, (2004).
- [26] Bishop Fox, Fast MD5 and MD4 Collision Generators. *BishopFox*, (2013).
- [27] Simin Du & Guodong Ye, ITW and RSA based asymmetric image encryption algorithm. *Alexandria University, Alexandria Engineering Journal*, (2022).
- [28] Gavin Z., Longxiang G., Liqun H. & Jian G., Ethereum Smart Contract Development in Solidity. Forrás:
https://books.google.hu/books?id=OGn6DwAAQBAJ&lpg=PR7&ots=_g2xRs3S7I&dq=smartcontract%20and%20solidity&lr&hl=hu&pg=PA3#v=onepage&q&f=false,
 (2021).
- [29] Ethereum, Forrás: <https://ethereum.org/en/developers/docs/intro-to-ethereum/>, (2023).
- [30] Sumaira J., Naveed A., Warda A., Haitham C. & Amad D.: Research and Applied Perspective to Blockchain Technology: A Comprehensive Survey. *Department of Computer Science, University of Peshawar, Pakistan*, (2021).
- [31] Luís C., Joao P. & Miguel T., Vulnerabilities in IoT Devices for Smart Home Environment. *Center of Technology and Systems (CTS), UNINOVA, Portugal*, (2023).
- [32] Stefan B., Ondrej G., Roderik P., Peter S. & Pavol Z., IoT Security Challenges: Cloud and Blockchain, Postquantum Cryptography, and Evolutionary Techniques. *Slovak University of Technology in Bratislava*, (2021), pp. 2-5.
- [33] Charalampos S., Konstantinos M., Konstantinos V., Dimitrios T., Anastasija C., Neils A., Dimitri K., Georgios S., Pankaj P. & Sokratis K., Implementing a Forms of

Consent Smart Contract on an IoT-based Blockchain to promote user trust. *Information Technologies Institue, Greece*, (2018).

- [34] Ciprian P., Rares C. & Alin S., Advancing security and data protection for smart home systems through blockchain technologies. *University of Bucharest, Romania*, (2022).
- [35] Node.js, Forrás: <https://nodejs.org/en/about>, (2023).
- [36] Solidity, Forrás: https://dev.to/envoy_/history-and-origin-of-solidity-2mhl, (2023).
- [37] React, Forrás: [https://en.wikipedia.org/wiki/React_\(JavaScript_library\)](https://en.wikipedia.org/wiki/React_(JavaScript_library)), (2023).
- [38] Next.js, Forrás: <https://vercel.com/home>, (2023).
- [39] Metamask, Forrás: <https://docs.metamask.io/guide/>, (2023).

10. Ábrajegyzék

- 2.1.1 ábra IoT architektúra rétegenkénti egységei; 6. oldal
- 2.1.2 ábra IoT architektúra rétegenkénti protokollok megjelenései; 7. oldal
- 2.1.3.1 ábra IoT ökoszisztémában lévő rétegek elleni támadások; 10. oldal
- 2.2.2.1 ábra IoT Okosotthon szemléltetése; 17. oldal
- 4.3.1 ábra IoT eszközök és blokklánc kapcsolata; 33. oldal
- 6.1.1 ábra Bejelentkezési felület; 40. oldal
- 6.1.2 ábra Eszközök felület; 41. oldal
- 6.1.3 ábra Etherscan oldalon található tranzakció részletei; 42. oldal
- 6.1.4 ábra Firmware Update felület; 42. oldal
- 6.1.5 ábra Firmware Update felület eszköz frissítését követően; 43. oldal