Szakdolgozat vázlat

Téma:
IoT security: támadási felületek azonosítása és védelmi mechanizmusok fejlesztése Wokwi-alapú szimulációval

Címlap, nyilatkozatok, tartalomjegyzék

- Címlap az egyetemi sablon szerint
- Hallgatói nyilatkozat (saját munka, plágiummentesség)
- Köszönetnyilvánítás (opcionális)
- Tartalomjegyzék, ábra- és táblázatjegyzék

1. Bevezetés (1–3 oldal)

1.1 Téma háttere és aktualitása

- IoT eszközök robbanásszerű terjedése (okosotthon, ipari IoT, egészségügy).
- Mirai-szerű botnet támadások, sebezhető kamerák, routerek, stb.
- Miért kritikus, ha egy IoT-rendszerben hamis szenzoradatok jelennek meg / lehallgatható a forgalom.

1.2 A szakdolgozat célkitűzései

- IoT támadási felületek és tipikus sebezhetőségek rendszerezett bemutatása.
- Biztonságos hálózati protokollok (főleg MQTT + TLS 1.3) és védelmi elvek (Zero Trust, defense-in-depth) ismertetése.
- Egy **Wokwi + Mosquitto + Python** alapú szimulációs környezet megtervezése, megvalósítása és mérése:
    - baseline (titkosítatlan MQTT),
    - TLS 1.3,
    - később mTLS + ACL.

1.3 Kutatási kérdések és hipotézisek

Példa (testre szabhatod):

- **H1:** A titkosítatlan MQTT-kommunikáció kevés számítási erőforrást igényel, de súlyos biztonsági kockázatokat hordoz (lehallgatás, hamis publish).
- **H2:** A TLS 1.3 bevezetése mérhető, de kezelhető overheadet okoz az ESP32-alapú rendszerben.
- **H3:** A kölcsönös tanúsítványalapú hitelesítés (mTLS) és az ACL-ek kombinációja hatékonyan blokkolja a jogosulatlan klienseket és topic-hozzáféréseket.
- **H4:** A Zero Trust és defense-in-depth elvek IoT-re szabott alkalmazása laboratóriumi környezetben is demonstrálható.

1.4 Módszertan rövid áttekintése

- Szakirodalmi kutatás (IoT security, TLS/mTLS, OWASP IoT Top 10, szabványok).
- Saját szimulációs környezet felépítése Wokwi + Mosquitto + Python alapon.
- Kvantitatív mérések (üzenetszám, ráta, hibák), logelemzés.

1.5 A dolgozat felépítése

2. fejezet: IoT technológiai alapok
3. fejezet: támadási felületek, szabványok
4. fejezet: biztonságos protokollok, védelmi architektúrák
5. 5–7. fejezet: saját szimulációs környezet, implementáció, mérések és értékelés
8. fejezet: összegzés, jövőbeli munka

2. IoT technológiai alapok és biztonsági kihívások (4–6 oldal)

2.1 Az IoT fogalma és tipikus architektúrái

- Peremeszközök (szenzorok, aktuátorok), gateway, felhő/backend.
- Példák: okosotthon, ipari felügyelet, okos város.

2.2 IoT-kommunikációs protokollok áttekintése

- MQTT: publish/subscribe, topic struktúra, QoS szintek.
- CoAP: REST-szerű, UDP alapú, DTLS-szel védhető.
- HTTP/HTTPS, WebSocket, egyéb (LwM2M, AMQP – csak megemlítve).

2.3 Erőforrás-korlátozott eszközök sajátosságai

- CPU, memória, energia-korlátok.
- Firmware frissítés kockázatai.
- Miért kihívás az erős kriptográfia kis mikrokontrollereken.

2.4 IoT biztonsági kihívásai

- Fizikai hozzáférés, eszköz ellopása, JTAG, debug interfészek.
- Gyári alapértelmezett jelszavak, gyenge/hiányzó hitelesítés.
- Titkosítatlan kommunikáció, hamis szenzoradatok, man-in-the-middle.
- Frissítési lánc (OTA update) sebezhetőségei.

3. IoT támadási felületek, tipikus sebezhetőségek és szabványok (3–5 oldal)

3.1 Támadási felületek kategorizálása

- **Eszköz/szenzor szint:** firmware hibák, debug portok, fizikai manipuláció.
- **Hálózati szint:** titkosítatlan protokollok, DoS, ARP spoofing, rogue AP.

- **Felhő/backend:** sérülékeny API-k, jogosultságkezelési hibák.
- **Felhasználó / menedzsment:** hibás konfiguráció, social engineering.

3.2 Tipikus sebezhetőségek és támadástípusok

- OWASP IoT Top 10 rövid bemutatása (pl. Insecure Web Interface, Insecure Network Services, Insecure Cloud Interface, stb.).
- Konkrét példák: Mirai botnet, sebezhető IP-kamerák, gyártói „backdoor" hozzáférések.

3.3 Ipari szabványok és ajánlások

- NIST, ENISA, ETSI EN 303 645 fő ajánlásai.
- OWASP IoT ajánlások: secure by design, secure update, logging, stb.
- Következtetés: miért indokolt biztonságos protokollok és architektúrák használata.

4. Biztonságos protokollok és védelmi architektúrák IoT-ben (6–8 oldal)

4.1 TLS 1.3 IoT környezetben

- TLS 1.3 újdonságai: rövidebb handshake, PFS, modern cipher suite-ok.
- Előnyök: bizalmasság, integritás, szerverhitelesítés.
- Korlátok: handshake overhead, cert-kezelés erőforrás-korlátozott eszközökön.

4.2 MQTT biztonsági modellje

- MQTT over TLS 1.3: broker szerepe, titkosított csatorna.
- Hitelesítés: felhasználónév/jelszó, kliens-tanúsítvány.
- ACL-ek: publish/subscribe jogosultságok topic-szinten.

4.3 Egyéb protokollok (elméleti áttekintés)

- DTLS: UDP-alapú TLS, CoAP védelem.
- OSCORE: alkalmazásszintű titkosítás CoAP felett.
- Röviden megemlíteni, hogy a dolgozat gyakorlati része ezekkel nem foglalkozik, csak kontextus.

4.4 Zero Trust és defense-in-depth IoT-ben

- Zero Trust alapelvek (never trust, always verify; identity everywhere; least privilege; folyamatos monitorozás).
- Hogyan illeszkedik ez egy IoT-architektúrába (eszköz-identitás, szegmentálás, broker ACL-ek, monitorozás).

4.4.1 IoT-specifikus Zero Trust megközelítés

- Szenszorok mint „identity-vel rendelkező kliensek".
- Broker, mint bizalmi határ, ACL-ekkel.

4.4.2 Többrétegű védelem (defense-in-depth)

- Eszköz-szint, hálózat-szint, alkalmazás-szint, monitorozás.

4.4.3 Korlátok és további bővítési lehetőségek

- Ez a rész **már megvan** nálad:
    - laboratóriumi környezet korlátai,
    - CoAP/DTLS/OSCORE csak elméletben,
    - későbbi kutatási irányok (fizikai eszközök, MI-alapú detekció).

5. A szimulációs környezet és kutatási módszertan (4–6 oldal)

(Ez gyakorlatban az, amit már elkezdtünk: 5.1–5.4)

5.1 A szimulációs környezet felépítése

- Ez **nálad már kész**:
    - Wokwi-s ESP32 szenzorok,
    - Mosquitto broker (1883, 8883),
    - Python backend (collector, analyzer),
    - hálózati topológia, topic struktúra, JSON payload.

5.2 MQTT-alapú mérési backend megvalósítása

- Szintén **megírtuk**:
    - collector.py (baseline),
    - collector_tls.py (TLS),
    - pub_test / pub_test_tls,
    - analyze_measurements*.py.

5.3 Mérési módszertan és vizsgált szcenáriók

- Szintén **kész vázlat**:
    - három brokerkonfiguráció (A: plaintext, B: TLS, C: mTLS+ACL – C majd később implementálva),
    - mérőszámok (N_msg, T_run, λ, átlag T, N_denied),
    - mérési eljárás lépései,
    - támadási szcenáriók (jogosulatlan publish, TLS melletti támadás, mTLS+ACL blokkolás).

5.4 Mérési eredmények és értékelés – I. (baseline + TLS)

- Már írtunk egy jó vázlatot:
    - 5.4.1 Baseline eredmények táblázat

- 5.4.2 TLS-es eredmények táblázat
- 5.4.3 összehasonlító értékelés (overhead kicsi, biztonság nő, TLS önmagában nem elég).

6. A fejlesztett (szimulált) IoT-biztonsági rendszer bemutatása (6–8 oldal)

Itt fókuszáltan az **implementáció részletes bemutatása** jön:

6.1 Architektúra logikai felépítése

- Rajz(ok) a teljes rendszerről: Wokwi ESP32-k → Mosquitto → Python backend → (opcionális dashboard).
- Támadó kliens pozíciója (hol csatlakozik a brokerhez).

6.2 ESP32-alapú szenzor node-ok implementációja

- Wokwi projekt leírása (sensor1, sensor2).
- Kódvázlat: WiFi connect, MQTT connect, JSON payload összeállítása, publish ciklikusan.
- A random „temperature” generálása és időzítés.

6.3 Mosquitto konfigurációk

- Alap config (1883 – plaintext).
- TLS config (8883 – ca.crt, server.crt, server.key).
- mTLS config vázlata:
  - client cert generálás,
  - require_certificate true, use_identity_as_username true.
- ACL fájl: mely kliens mely topicokra publish/subscribe-olhat.

6.4 Python backend és támadó kliens

- collector / collector_tls részletesebb bemutatása (belső logika, exception kezelés).
- attacker kliens(ek):
  - „normál” tesztpublisher,
  - módosított változat, ami jogosulatlan topicra lő (pl. más szenzor nevében publish).

6.5 Nem funkcionális követelmények és megfigyelések

- Késleltetés, CPU terhelés (inkább kvalitatív).
- Stabilitás, reconnect logika.

7. Támadási szcenáriók, tesztelés és mérési eredmények – II. (4–6 oldal)

Itt már **konkrét számokkal, logrészletekkel** mutatod be a támadásokat és a védelmi mechanizmusok hatását.

7.1 Normál működés biztonságos konfigurációban

- Röviden: TLS + (később mTLS+ACL) mellett hogyan néz ki egy „egészséges” futás.
- Grafikonok: üzenetküldési ráta idősor, hőmérséklet eloszlás.

7.2 Jogosulatlan hozzáférési kísérletek titkosítás nélkül

- Támadó kliens publish-el iot/lab/sensor1/temperature topicra.
- Mutatod, hogy a collector CSV-je kevert adatot tartalmaz; átlaghőmérséklet eltolódik.
- Rövid logrészlet a támadó kódból + magyarázat.

7.3 Jogosulatlan publish TLS mellett

- Ugyanaz a támadó kliens, de TLS-es kapcsolaton.
- Demonstrálod, hogy a TLS nem oldja meg a jogosultságkezelést, a támadás ugyanúgy sikeres.

7.4 Jogosulatlan publish mTLS + ACL mellett

- Kliens cert nélkül → broker elutasítja (log: „Client connection from X failed, not authorized”).
- Kliens certtel, de tiltott topic → ACL hiba.
- Broker logrészletek, N_denied statisztika.

7.5 Eredmények összefoglalása

- Táblázat: baseline vs TLS vs mTLS+ACL (üzenet ráta, hibaszám, elutasított kísérletek).
- Rövid értékelés:
    - TLS overhead ≈ kicsi;
    - mTLS+ACL lényegesen szűkíti a támadási felületet;
    - kompromisszumok (bonyolultabb cert-menedzsment, config).

8. Összegzés és kitekintés (1–3 oldal)

8.1 Eredmények összefoglalása

- Mit értél el:
    - támadási felületek rendszerezése,
    - TLS 1.3 és Zero Trust elvek IoT kontextusban,
    - Wokwi + Mosquitto + Python szimuláció sikeres felépítése,
    - baseline vs TLS vs (tervezett) mTLS+ACL mérések.

8.2 Hipotézisek kiértékelése

- H1–H4 röviden: melyiket igazolták a mérések, melyik részben.

8.3 A prototípus és a kutatás korlátai

- Szimulált környezet,
- Korlátozott eszközszám,
- CoAP/OSCORE csak elméleti szinten,
- Nincs valós ipari környezet.

8.4 Jövőbeli fejlesztési lehetőségek

- Fizikai ESP32/STM32 eszközök bevonása.
- CoAP + DTLS/OSCORE gyakorlati tesztelése.
- MI-alapú anomáliadetekció integrálása az MQTT-forgalom monitorozásába.
- Automatizált deploy (Docker, Kubernetes, stb.) nagyobb skálán.

8.5 Irodalomjegyzék és függelékek

- Irodalomjegyzék az intézeti forma szerint (APA vagy MSZ, ahogy kérik).
- Függelék(ek):
    - fontosabb kódrészletek (ESP32 sketch, collector, attacker),
    - Mosquitto TLS/mTLS/ACL konfigurációk,
    - Wokwi projekt linkje,
    - mérési táblázatok részletesen (ha nem férnek el a törzsszövegben).