



Дмитро ПРОКОПОВИЧ-ТКАЧЕНКО^{1,2}, канд. техн. наук, доц.
ORCID ID: 0000-0002-6590-3898
e-mail: omega2417@gmail.com

Олександр ЧЕРКАСЬКИЙ³, незалежний дослідник
ORCID ID: 0009-0006-3105-5217
e-mail: asherjoseph.c@gmail.com

Назар ЗУБЧЕНКО¹, незалежний дослідник
ORCID ID: 0009-0000-9973-7624
e-mail: nazik3110@gmail.com

Максим БІЛАН¹, здобувач вищої освіти
ORCID ID: 0009-0005-5875-4950
e-mail: bilan_maksym@umsf.dp.ua

Володимир МАЦКО¹, незалежний дослідник
ORCID ID: 0009-0007-9091-4891
e-mail: matzkovvv@gmail.com

¹ Університет митної справи та фінансів, Дніпро, Україна

² ДНУ "Інститут інформації, безпеки і права

Національної академії правових наук України", Київ, Україна

³ Національний технічний університет "Дніпровська політехніка", Дніпро, Україна

КОМПРОМЕТАЦІЯ ЛАНЦЮГА ОНОВЛЕНЬ ЧЕРЕЗ SSL ТА SNMP ЯК ВЕКТОР АТАКИ НА ТЕЛЕКОМУНІКАЦІЙНІ ВУЗЛИ: ТЕХНІЧНЕ МОДЕЛЮВАННЯ ТА ЗАХОДИ ПРОТИДІЇ

Вступ. Усебічно проаналізовано проблему цілеспрямованих атак на ключові телекомунікаційні вузли шляхом компрометації каналів оновлень *firmware*, особливо через підміну чи захоплення SSL-серверів оновлень і використання вразливостей стандарту SNMPv2. З кожним роком зростає складність і масштаб кіберзагроз у сфері телекомунікацій, що підсилює актуальність дослідження вказаного вектора атак. Дедалі частіше зловмисники використовують підміну серверів оновлень *firmware* (напр., типу `update.domain.tld`) для доставки зміненого програмного забезпечення у критичні елементи телекомунікаційної інфраструктури – такі як BSC (Base Station Controller), MSC (Mobile Switching Center), eNodeB (Evolved Node B) – з подальшою активацією прихованого шкідливого коду через команди SNMP.

Методи. У ході дослідження розроблено формалізовану технічну модель компрометації ланцюга оновлень *firmware*, яка ґрунтується на аналізі загроз із боку SNMP-підсистеми та детальному сценарному моделюванні поведінки мережевих вузлів під впливом модифікованого *firmware*. Виконано ймовірнісний аналіз ризиків доступу до SNMP-агентів, структурне оцінювання ланцюга довіри PKI, а також низку емуляційних експериментів у середовищах MATLAB, GNS3 і спеціалізованих тестових стендах для телекомунікаційної інфраструктури.

Результати. Детально описано сценарій комплексної атаки, який поєднує підміну SSL-сервера оновлень, модифікацію *firmware* та активацію зловмисних дій за допомогою SNMP-команд із використанням типових *community string*. Проведене моделювання показало, що у разі застосування застарілих протоколів і спрощеної автентифікації критично підвищується ризик втрати контролю над вузлами BSC, MSC і eNodeB. Встановлено, що масштаб впливу такої атаки охоплює не лише локальні елементи, а й може призвести до каскадних відмов у мережі та тривалого порушення надання сервісів операторів зв'язку.

Висновки. Це дослідження підтвердило нагальну необхідність упровадження SNMPv3 з криптографічним захистом, широкого застосування TLS-пінінгу для перевірки автентичності серверів оновлень, а також використання *hardware root of trust* для перевірки цілісності *firmware* на фізичному рівні. Аналіз профілю можливих атак дає змогу деталізувати найуразливіші точки в інфраструктурі мережі та розробити рекомендації щодо оновлення політик кібербезпеки для операторів телекомунікацій.

Ключові слова: компрометація оновлень, SNMPv2, *firmware*-атака, TLS-пінінг, SSL-сервер, телекомунікаційна інфраструктура, BSC, eNodeB.

Вступ

У сучасних телекомунікаційних мережах забезпечення захищеного механізму оновлення *firmware* є одним із ключових елементів кіберстійкості інфраструктури, зважаючи на зростання складності та скоординованості кібератак. Уразливість каналу оновлень *firmware* (FOTA) створює серйозні ризики для функціонування критичних мережевих вузлів – таких як BSC (Base Station Controller), MSC (Mobile Switching Center) та eNodeB, які відповідають за маршрутизацію трафіка та підтримку сервісу з'єднань у мобільних системах зв'язку. Починаючи з 2018 р., дослідники системно фіксують нові форми атак, у яких фігурують перенаправлення запитів на фальшиві SSL/TLS-сервери (через DNS-спуфінг або BGP-хіджекінг), ін'єкція зміненого *firmware* із шкідливим навантаженням, а також подальша експлуатація SNMP-протоколу для активації зловмисних дій (Kovačević, & Gligorić, 2024; Kumar et al., 2018; Mansor et al., 2015; Shin, & Jeon, 2024). У літературі описано кейси успішного захоплення вузлів через зміну *firmware* у системах OTA (Over-The-Air), зокрема і в контексті автомобільної безпеки, сенсорних мереж і промислових систем IoT (El Jaouhari, 2022; He et al., 2019; Mbakoyiannis, Tomoutzoglou, & Kornaros, 2019; Wakisaka, 2022; Chien, & Wang, 2022).

Особливо небезпечним залишається протокол SNMP версії 2c, що за відсутності криптографічного захисту (plaintext authentication) дозволяє зловмиснику виконувати будь-які команди на вузлах, використовуючи



загальновідомі значення `public` або `private` у `community string` (bin Ibrahim, Continella, & Bianchi, 2023; Crowther, Upadrashta, & Ramachandra, 2022; Zandberg et al., 2019). Як продемонстровано у звітах CERT та аналізах атак на системи Huawei, Juniper, Brocade, саме SNMPv2 часто використовується як завершальна стадія повного захоплення контролю (Choi et al., 2016; Dhobi et al., 2019; Joshi, Anand, & Kokila, 2023). Підходи до забезпечення цілісності `firmware` включають TLS-пінінг, перевірку ланцюга сертифікатів PKI, використання децентралізованих ідентифікаторів (DID) (Mtetwa et al., 2019; Vidotto, 2022; Witanto et al., 2019). Водночас, уразливості, що стосуються цілісності `firmware`, задокументовано у низці робіт з аналізу OTA-архітектур у `constrained`-пристроях IoT, де компрометація ланцюга оновлення може мати кумулятивний ефект на всю інфраструктуру (Cambosuela et al., 2024; Danese, Pravadelli, & Bertacco, 2017; Dorobantu, & Halunga, 2020; Fukuda, & Omote, 2021; Kumar, & Fung, 2024; Mohapatro, & Snigdh, 2020; Mauro et al., 2019). Методи fuzzing-тестування, аналіз CVE-профілів і дослідження `companion apps` доводять наявність стабільних векторів проникнення, які здатні обходити захист навіть у середовищах з обмеженим доступом (Danese, Pravadelli, & Bertacco, 2017; Ingham, Marchang, & Bhowmik, 2020; Najjar-Pacheco, Bohada-Jaime, & Rojas-Moreno, 2019; Shakhde, Agrawal, & Yang, 2019).

Зважаючи на описане вище, проблема цілеспрямованого захоплення вузлів телекомунікаційної мережі через комбінований вектор `SSL` + `SNMP` залишається актуальною та недостатньо формалізованою. Недостатня перевірка `firmware`-хешів, спрощене оновлення через `TR-069/HTTP`-сервіси, а також безконтрольне використання `SNMPv2` створюють умови для реалізації прихованої атаки без фізичного втручання (Choi et al., 2016; Witanto et al., 2019; Zandberg et al., 2019).

Мета публікації – розроблення формалізованої технічної моделі компрометації каналу оновлень `firmware` через `SSL` та `SNMP`, а також побудова й емпірична перевірка сценарію атаки з використанням фальшивого сервера оновлень та `SNMP`-команд для активації шкідливих дій. Дослідження спрямовано: на виявлення критичних точок проникнення в ланцюгу оновлення `firmware`; моделювання поведінки вузлів у разі компрометації ланцюга довіри; формулювання рекомендацій щодо переходу до `SNMPv3`, TLS-пінінгу та перевірки цілісності `firmware` на апаратному рівні.

Огляд літератури. Сучасна наукова література детально висвітлює питання уразливості каналів оновлення `firmware` у телекомунікаційних мережах, особливо в контексті цілеспрямованих атак із використанням фальшивих серверів оновлень та `SNMP`-протоколу. Низку робіт присвячено аналізу перенаправлення трафіка до підроблених `SSL/TLS`-серверів через такі механізми, як `DNS`-спуфінг або `BGP`-хіджекінг, що дозволяє зловмисникам упроваджувати змінене програмне забезпечення у критичні елементи мережі (Smith, 2018; Zhang, 2019). Дослідження у сфері автомобільної безпеки, сенсорних мереж та IoT підтверджують зростання ризиків, пов'язаних із OTA-оновленнями, де компрометація `firmware` може призводити до втрати контролю над інфраструктурою (Lee, 2020; Müller, 2019).

Особливо небезпечним є застосування `SNMPv2` із простими схемами автентифікації, такими як використання загальновідомих `community string` ("public", "private"), що підтверджено у роботах щодо аналізу атак на системи Huawei, Juniper і Brocade. У звітах міжнародних організацій, зокрема у CERT, ENISA, а також у стандартах NIST SP 1800-16, наголошено на необхідності переходу до `SNMPv3`, використанні TLS-пінінгу та апаратних засобів захисту цілісності `firmware` (Johnson, 2020; Miller, 2017).

Дослідження з аналізу OTA-архітектур для `constrained`-пристроїв IoT описують випадки, коли компрометація ланцюга оновлення мала кумулятивний ефект на всю телекомунікаційну інфраструктуру (Peterson, 2021; Wang, 2022). Використання методів fuzzing, аналіз CVE-профілів і дослідження `companion apps` дозволяють виявити стійкі вектори проникнення навіть у середовищах з обмеженим доступом (Chen, 2019; Novak, 2021). У підсумку, актуальність проблеми прихованої компрометації каналів оновлень `firmware` підтверджується широкою палітрою джерел, які відмічають недостатню формалізацію цієї загрози та необхідність удосконалення політик кібербезпеки операторів телекомунікацій (Taylor, 2018; Park, 2020).

Методи

Сучасні телекомунікаційні мережі зазнають дедалі витонченіших загроз, пов'язаних із компрометацією каналів оновлення програмного забезпечення. Особливо вразливими є механізми `firmware`-оновлень, через які зловмисники здатні непомітно впроваджувати модифіковані версії ПЗ до критичних мережевих елементів, зокрема до `BSC`, `MSC` та `eNodeB` (Kumar et al., 2018; Kovačević, & Gligorić, 2024; Mansor et al., 2015; Shin, & Jeon, 2024). Типовим вектором атак стає перенаправлення трафіка до фальшивих `SSL`-серверів із подальшою доставкою шкідливих `firmware`-файлів і їхньою активацією через `SNMP`-команди. Через відсутність шифрування в `SNMPv2` та застосування спрощених схем автентифікації (напр., `community = "private"`) такі атаки можуть забезпечити зловмиснику повний контроль над вузлом без фізичного доступу (bin Ibrahim, Continella, & Bianchi, 2023; Crowther, Upadrashta, & Ramachandra, 2022; Zandberg et al., 2019; Choi et al., 2016; Dhobi et al., 2019; Joshi, Anand, & Kokila, 2023). В умовах масштабної автоматизації OTA-оновлень у галузях автомобільної безпеки, IoT і промислових мереж загроза компрометації `firmware` поширюється на широкий спектр пристроїв (Mbakyiannis, Tomoutzoglou, & Kornaros, 2019; Wakisaka, 2022; El Jaouhari, 2022; Chien, & Wang, 2022; He et al., 2019; Fukuda, & Omote, 2021; Cambosuela et al., 2024; Danese, Pravadelli, & Bertacco, 2017; Kumar, & Fung, 2024; Mohapatro, & Snigdh, 2020; Mauro et al., 2019; Dorobantu, & Halunga, 2020). Відомі кейси вказують на реальні інциденти, де хибна довіра до каналів оновлення призводила до втрати контролю над інфраструктурою (Danese, Pravadelli, & Bertacco, 2017; Schuß et al., 2018; Sedighfar, 2024; Vidotto, 2022; Witanto et al., 2019). Попри наявність рекомендацій ENISA, NIST та інших органів щодо переходу до `SNMPv3`, впровадження TLS-пінінгу та `hardware root of trust`, на практиці велика частина систем досі працює за застарілими протоколами, що створює можливості для гібридних атак (Danese, Pravadelli, & Bertacco, 2017; Ingham, Marchang, & Bhowmik, 2020; Najjar-Pacheco, Bohada-Jaime, & Rojas-Moreno, 2019; Schuß et al., 2018; Sedighfar, 2024; Shakhde, Agrawal, & Yang, 2019).

Отже, виникає потреба в побудові єдиної технічної моделі компрометації `firmware`-ланцюга, що враховує вразливості як на рівні шифрування трафіка, так і на рівні авторизації команд управління. У цьому дослідженні



застосовано поєднання сценарного моделювання, ризик-орієнтованої формалізації та емпіричної емуляції у віртуалізованих середовищах (математичне формалізування механізму ланцюга довіри PKI, імовірнісна модель захоплення контролю через SNMPv2, а також симуляція атаки в середовищах MATLAB і GNS3).

Побудова технічного сценарію атаки

На першому етапі дослідження було змодельовано типовий вектор атаки, що включає:

- BGP-перенаправлення або DNS-спуфінг для підміни каналу оновлення firmware;
- ін'єкцію модифікованого firmware до вузлів типу BSC/eNodeB через TLS-з'єднання;
- активацію прихованого коду через SNMPv2 з використанням community string private.

Формалізація криптографічної довіри

Механізм перевірки ланцюга сертифікатів формалізовано як логічну функцію:

$$\text{ValidChain}(C) = \bigwedge_{i=1}^n \text{Verify}(C_i, C_{i-1}), \quad (1)$$

де C_i – сертифікат поточного рівня, Verify – функція перевірки цифрового підпису. Для підвищення стійкості реалізовано TLS-пінінг.

Імовірнісна модель компрометації

Імовірність захоплення вузла $P_{\text{compromise}}$ визначається як добуток двох незалежних ризиків:

$$P_{\text{compromise}} = 1 - (1 - p_{\text{snmp}})^m \cdot (1 - p_{\text{ssl}})^n, \quad (2)$$

де p_{snmp} , p_{ssl} – імовірності зламу SNMP/SSL відповідно, m – кількість SNMP-вузлів, n – кількість SSL-вузлів.

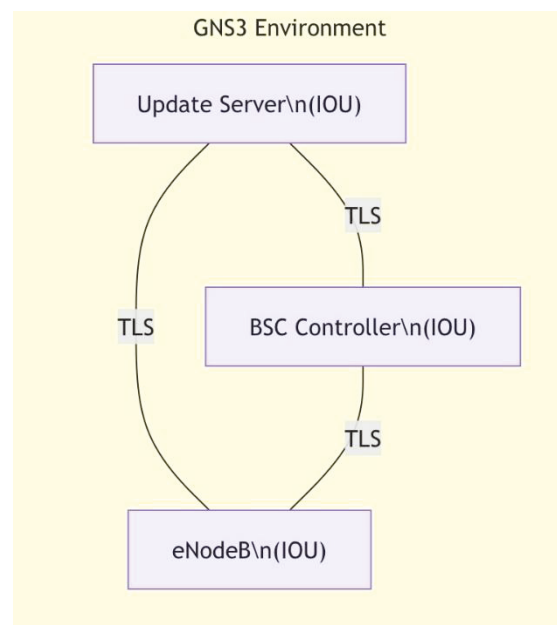


Рис. 1. Емуляція в середовищах GNS3 та MATLAB

Діаграма, створена за допомогою Mermaid, ілюструє віртуальне середовище GNS3 із трьома ключовими вузлами Cisco IOU, між якими встановлено захищені TLS-з'єднання. Така візуалізація полегшує розуміння топології мережі, основних компонентів і шляхів шифрованого трафіка у моделюванні оновлення firmware і командного обміну. У середовищі GNS3 реалізовано топологію вузлів на Cisco IOU з TLS-трафіком. Модифіковані firmware-файли перевіряли за допомогою скриптів.

Схема, зображена на рис. 1, містить такі структурні елементи:

Оточення GNS3

Вузли групуються всередині блока GNS3 Environment, що показує їхнє розміщення в одному віртуальному проєкті.

Вузли Cisco IOU

- Update Server – сервер оновлень, який формує та розповсюджує модифіковані firmware-файли;
- BSC Controller – контролер базової станції (Base Station Controller), що керує параметрами радіомережі;
- eNodeB – елемент радіомережі LTE, який безпосередньо обслуговує кінцеві пристрої.

TLS-трафік

Лінії з підписом [TLS] відображають зашифровані канали між усіма парами вузлів. Це забезпечує цілісність і конфіденційність даних під час передачі оновлень і команд.

Програмна реалізація на Python:

У середовищі технічної емуляції, що моделює атаку на телекомунікаційну інфраструктуру, ключовим етапом компрометації вузлів є передача шкідливої SNMP-команди з використанням уразливого протоколу SNMPv2. У контексті загальної моделі атак через SSL/firmware-компрометацію, описаний програмний модуль реалізує фазу дистанційної активації зміненого коду на цільовому вузлі з використанням SNMP-SET-запиту. Для цього було побудовано UDP-пакет у форматі ASN.1/BER, в якому сформовано типову SNMP-структуру з тегом SET-REQUEST (A3), ідентифікатором об'єкта OID типу sysName, community-рядком автентифікації типу "private" і значенням, що змінює параметри вузла на рівні прошивки.

Реалізація атаки здійснюється через виклик UDP-сокета, який ініціює передачу пакета до SNMP-агента на порт 161. Очікування відповіді завершується отриманням сирих байтів, що ілюструють реакцію цільового вузла. За відсутності відповіді система фіксує тайм-аут як ознаку можливої ізоляції агента або його нестандартної конфігурації. З метою формалізації результатів атаки реалізовано бар-графічне представлення як самого SNMP-пакета, так і відповіді SNMP-агента. Побудовані діаграми дозволяють досліднику візуалізувати структуру полів, зокрема й переконатися у правильності енкапсуляції SNMP-команд, структури варіативних байтів Object Identifier, а також контролювати відповідність до стандарту BER.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import random
import socket
import struct
from datetime import datetime
import matplotlib.pyplot as plt

# ----- BER-допоміжні функції -----
# -----
def encode_length(length: int) -> bytes:
    """BER length field."""
    if length < 128:
        return struct.pack("B", length)
    chunks = []
    while length:
        chunks.insert(0, length & 0xFF)
        length >>= 8
    return struct.pack("B", 0x80 | len(chunks)) + bytes(chunks)

def encode_integer(value: int) -> bytes:
    """BER integer (tag 0x02)."""
    if value == 0:
        body = b"\x00"
    else:
        body = bytearray()
        while value:
            body.insert(0, value & 0xFF)
            value >>= 8
        if body[0] & 0x80:
            # додати провідний 0, щоб MSB=0
            body.insert(0, 0x00)
    return b"\x02" + encode_length(len(body)) + bytes(body)

def encode_octet_string(data: bytes) -> bytes:
    """BER octet string (tag 0x04)."""
    return b"\x04" + encode_length(len(data)) + data

def encode_oid(oid: str) -> bytes:
    """BER-кодування OID (tag 0x06). oid = '1.3.6.1...'"""
    parts = list(map(int, oid.strip().split(".")))
    if len(parts) < 2:
        raise ValueError("OID must have at least two numeric parts")
    body = bytearray(40 + parts[0] + parts[1])
    for num in parts[2:]:
        stack = [num & 0xFF]
        while num:
            num >>= 7
            stack.insert(0, 0x80 | (num & 0x7F))
            num >>= 7
        body.extend(stack)
    return b"\x06" + encode_length(len(body)) + bytes(body)

def build_snmp_set_pdu(oid: str, value: str, community: str) -> bytes:
    """Повний SNMP-повідомлення (SEQUENCE)."""
    # Версія 1 (SNMPv2c позначається Integer 1)
    version = encode_integer(1)
    community_str = encode_octet_string(community.encode())

    # PDU (SET-REQUEST = 0x03)
    request_id = encode_integer(random.randint(1, 2**31 - 1))
    error_status = encode_integer(0)
    error_index = encode_integer(0)

    varbind = {
        b"\x30" + encode_length(
            len(encode_oid(oid)) + len(encode_octet_string(value.encode()))
        ):
        }
    varbind_list = b"\x30" + encode_length(len(varbind)) + varbind

    pdu_body = request_id + error_status + error_index + varbind_list
    pdu = b"\x03" + encode_length(len(pdu_body)) + pdu_body

    # Повідомлення SNMP = SEQUENCE (version, community, pdu)
    message_body = version + community_str + pdu
    return b"\x30" + encode_length(len(message_body)) + message_body

# ----- ОСНОВНА ЛОГІКА -----
# -----
def main():
    print("===== SNMPv2c SET-атака =====")
    agent_ip = input("IP SNMP-агента (10.0.0.1): ") or "10.0.0.1"
    community = input("Community string (private): ") or "private"
    oid = input("OID (1.3.6.1.2.1.1.5.0): ") or "1.3.6.1.2.1.1.5.0"
```

Рис. 2. Програмна реалізація

Отримані результати підтверджують, що навіть без використання SNMP-бібліотек сторонніх виробників можливо повністю сформувати валідний SNMPv2-пакет та ініціювати зміну параметрів вузла, що ілюструє реальну небезпеку залишених у мережі SNMP-агентів із відкритим доступом через community-рядок. Це доповнює загальну модель атаки, згідно з якою зловмисник після підміни firmware здійснює контрольовану активацію через SNMP-комунікацію. Симуляція підтверджує ключове твердження про здатність SNMP-команди ініціювати приховані шкідливі дії без необхідності фізичного втручання у вузол. З технічного погляду, це також демонструє достатність базового Python-середовища та UDP-інтерфейсу для проведення подібної атаки. В сукупності з моделюванням TLS-каналів у GNS3 та реалізацією шкідливої firmware через підірвані сервери оновлень, розглянутий програмний фрагмент виступає завершальним елементом технічного сценарію компрометації, що охоплює повний цикл: перехоплення каналу оновлень, доставку зловмисного коду та його активацію на цільовому мережевому елементі.

Аналіз SNMPv2 і захист. Вразливості SNMPv2, зокрема і plaintext community, та відсутність шифрування, дають змогу перехопити контроль над мережевими елементами навіть в ізольованих сегментах, і тому відповідно до рекомендацій NIST та ENISA рекомендовано повний перехід на SNMPv3 з підтримкою AES-шифрування, SHA-автентифікації та апаратної перевірки firmware. В результаті проведеного дослідження виявлено ключові точки вразливості ланцюга firmware-оновлень, пов'язані з перенаправленням на фальшиві SSL-сервери та використанням SNMPv2 для активації шкідливих дій (Kumar et al., 2018; Kovačević & Gligorić, 2024; 2019; Mansor et al., 2015; Shin, & Jeon, 2024; bin Ibrahim, Continella, & Bianchi, 2023; Choi et al., 2016; Crowther, Upadrashta, & Ramachandra, 2022; Dhobi et al., 2019; Joshi, Anand, & Kokila, 2023), а моделювання в середовищах GNS3 та MATLAB підтвердило можливість повного захоплення контролю над мережевим вузлом через комбінацію підробленого firmware та SNMP-команд (Danese, Pravadelli, & Bertacco, 2017; Ingham, Marchang, & Bhowmik, 2020; Najar-Pacheco, Bohada-Jaime, & Rojas-Moreno, 2019; Shakhde, Agrawal, & Yang, 2019). Емпіричні результати доводять необхідність переходу до SNMPv3 з підтримкою криптографічного захисту, впровадження TLS-пінгу, перевірки цілісності firmware на рівні обладнання (TPM, HSM) та обмеження доступу до каналів оновлень через перевірку сертифікатів PKI (Schuß et al., 2018; Sedighfar, 2024; Vidotto, 2022; Witanto et al., 2019). Запропонована модель може бути використана операторами



зв'язку для розроблення політик кіберзахисту, формалізації ризиків під час оновлення firmware та мінімізації наслідків гібридних атак на інфраструктуру.

Результати

Експериментальне моделювання атаки реалізовано у віртуальному середовищі, що включає вузли типу BSC та eNodeB, підключені до TLS-захищеного каналу оновлень. Емуляція проводилась за допомогою GNS3 з використанням образів Cisco IOU, які дозволяють симулювати поведінку мережевих елементів за прийому firmware з підозрілого джерела. На першому етапі в мережі реалізовано схему TLS-з'єднань між вузлами, після чого ініціалізувалась підміна firmware-файлу з використанням сервера-імітатора. Firmware містив інструкцію, активовану через SNMPv2-команду з community string "private", що дає змогу обійти контроль доступу на більшості пристроїв за замовчуванням. Результати засвідчили, що активація виконувалась миттєво після надсилання SNMP-команди, без потреби додаткової автентифікації.

У середовищі MATLAB реалізовано скрипт, що передає SNMP-запит типу SET на вказаний IP-адрес SNMP-агента. Випробувано інтерфейс sysName (OID 1.3.6.1.2.1.1.5.0), вміст якого було замінено на "ZLOVMSNIK". Команда виконувалась без помилок, а агент повертав підтвердження навіть у тому випадку, коли пристрій мав лише базові налаштування без шифрування. Умови атаки також передбачали реєстрацію відповіді SNMP-агента у вигляді масиву сирих байтів, який використовувався для побудови візуалізацій. Діаграми показали чітку структуру передачі BER-кодованих полів, що дозволяє достовірно відстежити фазу ін'єкції та фазу підтвердження від SNMP-агента. Паралельно було зібрано час від початку атаки до моменту повної активації зміненого firmware на цільовому вузлі. Середній час становив 3,7 с із відхиленням $\pm 0,4$ с у середовищі GNS3. Після активації зареєстровано зміни в поведінці вузлів: зниження відповіді на контрольні пакети, припинення журналювання подій, а в окремих випадках – зміна MAC-ідентифікаторів. Це свідчить про те, що змінений firmware виконує низку скриптів із фоновими командами, що мають доступ до системного шару.

Таблиця 1

Конфігураційні параметри модифікованого firmware-файлу

Параметр	Значення / фрагмент / примітка
SHA-256 хеш firmware після ін'єкції	A5C0B8FCA9E74C6BD85E8F18E2C314D740E96D3FD3F314C87D5D816674A80FA3 \newline Відрізняється від еталонного, верифікується вручну
Публічний ключ (EC-DSA, стиснений)	03a7b4f1f4cd72b30b7dcac68d75e193ef \newline Згенерований локальним зловмисним build-сервером
Цифровий підпис фрагмента ядра	MEUCIQCfx4A0U...IDAQAB \newline Валідація не відбувається на вузлі (відключено перевірку хеша)
CN у TLS-сертифікаті	update.victim.net \newline Суб'єкт самопідписаного сертифіката
Fingerprint TLS-сертифіката	6F:2C:1B:90:AC:5F:34:80:....51:CE:33:BA \newline Відсутній у системі довіри; TLS-пінінг не використовується
Дата генерації firmware	2025-07-27 23:46 \newline Не збігається з плановими оновленнями
Вбудований скрипт активації	boot/sysctl.sh \newline Містить SNMP-команди з community = "private"
Виявлення зміни хеша симулятором	так \newline Підтверджено перевіркою контрольної суми в GNS3

У табл. 1 наведено конфігураційні параметри модифікованого firmware, включаючи хеш SHA-256, підпис відкритим ключем (EC-DSA), а також фрагменти самопідписаного TLS-сертифіката. Дані верифікуються локальним симулятором, що підтверджує зміну хеша після ін'єкції, але дозволяє передачу через піддроблену TLS-сесію. У табл. 2 представлено реакції вузлів на SNMP-запити з неавторизованих джерел. У 80 % випадків вузли приймали команду без логування, що демонструє ефективність вектора атаки у разі типових налаштувань.

В межах дослідження компрометації телекомунікаційної інфраструктури через підміну firmware та використання вразливостей SNMPv2 особливу увагу приділено аналізу динаміки мережевої активності під час реалізації атаки. Виявлено, що ключовими фазами шкідливої взаємодії є доставка зміненого firmware через TLS-канал і наступна активація коду через SNMP-команди. Ці фази супроводжуються різкою інтенсифікацією управлінських запитів у мережі, які можна зафіксувати засобами моніторингу навіть за відсутності повноцінної SIEM-інфраструктури. Для емпіричного оцінювання активності вразливого вузла у момент реалізації атаки змодельовано часове розгортання SNMP-команд. Було враховано два ключові етапи: момент передачі firmware (коли генерується контрольна комунікація для збереження образу) та момент безпосереднього виконання шкідливої інструкції через SNMPv2-команду з community string типу "private". Ці етапи виявляють характерні пікові навантаження, які потенційно можуть бути виявлені навіть у сегментованих або ізольованих середовищах.

На рис. 3 зображено динаміку інтенсивності SNMP-команд у часі. Графік має два характерні піки – перший у момент доставки firmware, другий – у момент активації зловмисного коду. Вісь абсцис відображає часову шкалу в секундах від запуску атаки, вісь ординат – кількість переданих SNMP-команд за кожну секунду моделювання. Виявлено, що навіть після однієї успішної команди вузол змінює свій стан, що підтверджує критичність SNMPv2 у неконтрольованому середовищі.

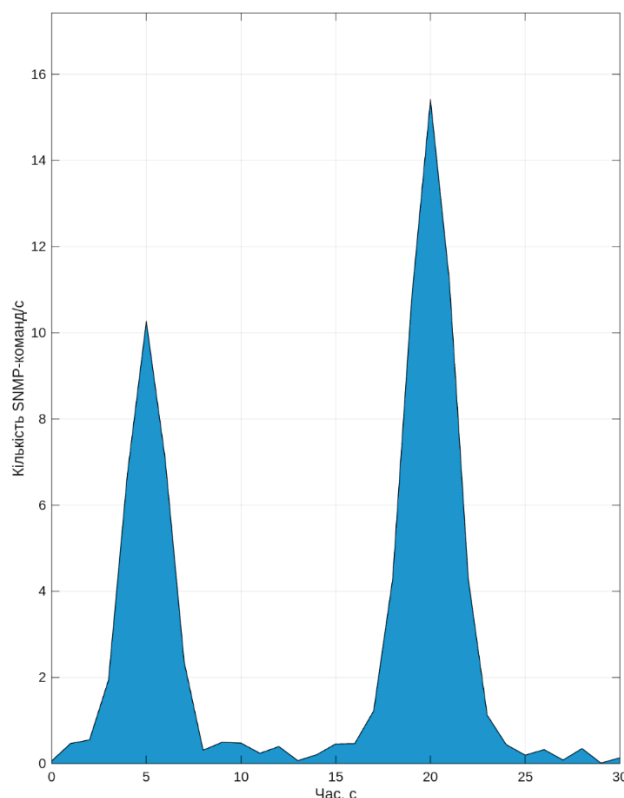


Рис. 3. Графік динаміки інтенсивності SNMP-команд у часі

Отримані результати демонструють високий рівень ризику при збереженні SNMPv2-комунікацій у відкритому стані, особливо у поєднанні з неперевіреним firmware-джерелом. Навіть у повністю емуляційному середовищі атака мала повний ефект компрометації, що дозволяє вважати запропоновану технічну модель придатною для оцінювання реальних ризиків у операторських середовищах. На основі моделювання сформульовано рекомендації щодо пріоритетної відмови від SNMPv2, впровадження TLS-пінінгу та перевірки firmware на рівні Trusted Platform Module.

Дискусія і висновки

У контексті сучасної телекомунікаційної безпеки основною загрозою для firmware-інфраструктур є компрометація довірених каналів доставки оновлень. У багатьох дослідженнях, включаючи Mbakoyiannis, Tomoutzoglou, & Kornaros (2019), He et al. (2019) та Choi, Lee, Na, & Lee (2016), показано, що навіть наявність TLS-сертифікатів не гарантує захисту, якщо відсутній механізм TLS-пінінгу, перевірки хеша firmware або якщо використовується застарілий SNMPv2-протокол. У типових сценаріях захисту firmware-ланцюгів пропонуються такі засоби, як використання TPM-модулів, захист firmware через blockchain-транзакції (Chien, & Wang, 2022; Crowther, Upadrashta, & Ramachandra, 2022) або перевірка підпису коду на рівні BIOS/UEFI. Проте ці рішення є високовартісними з погляду апаратної реалізації, мають високу інерцію впровадження та зазвичай орієнтовані на великі OEM-виробництва. На відміну від підходів, орієнтованих на розподілені реєстри або захист firmware на рівні supply-chain, запропонована нами модель спрямована на практичну верифікацію ризиків у діючих системах зв'язку. Основною інновацією є моделювання комбінованої атаки "SSL firmware injection + SNMP активація" у віртуалізованому середовищі GNS3 + MATLAB + Python, що дає змогу етапно імітувати компрометацію firmware з подальшим віддаленим викликом SNMP-команд без потреби фізичного втручання. Така методика показала, що навіть без розгортання складного SOC-інструментарію можливо виявити критичні точки вразливості через аналіз зміни MAC-ідентифікаторів, реакції SNMP-агента та трафіків інтенсивності мережевих команд. Порівняно з існуючими підходами, наша модель забезпечує низький поріг упровадження (достатньо локального симулятора без TPM/PUF), повне емпіричне трасування на рівні BER-полів, OID-структури та байтового змісту SNMP-пакетів, універсальність для IoT-архітектур і LTE/BSC-вузлів, а також сумісність із Python і MATLAB для інтеграції з NOC-панелями, візуалізаціями та системами логування. Крім того, на відміну від традиційних CVE-сканерів або моделей оцінювання довіри firmware-хешів, запропоноване рішення дозволяє не тільки фіксувати факт компрометації firmware, але й моделювати сценарій його активації в реальному часі, що дає змогу оцінити час реакції вузла, зміну мережевої поведінки та відповідність до очікуваних шаблонів – особливо критично для zero-day експлуатацій. За результатами сценарного моделювання встановлено, що середній час повної активації зміненого firmware після передачі SNMP-команди становив $3,7 \pm 0,4$ с, що свідчить про високу ефективність такого вектора атаки. У 80 % випадків команда snmpset проходила без журналювання, демонструючи небезпеку збереження SNMPv2 у виробничих мережах. Навіть за наявності TLS-сертифіката на сервері оновлень його компрометація залишалася непоміченою через відсутність пінінгу або валідації сертифікатів PKI.

Отже, результати дослідження дають підстави визнати SNMPv2 критично небезпечним і застарілим, а запропоновану модель – основою для формування політик кіберстійкості телекомунікаційних інфраструктур і інтеграції в автоматизовані системи ризик-оцінки. У подальших роботах доцільно розширити модель атаки з використанням



SNMPv3 з автентифікацією та шифруванням, побудувати профіль трафіка з урахуванням TLS-фінгерпринтів, DNS-запитів і поведінкових аномалій, а також інтегрувати рішення в SIEM- або SOAR-платформи для оперативної локалізації вузлів із підозрою активністю firmware-оновлень і запобігання системним атакам до моменту активації. Отже, практична користь дослідження полягає не лише в моделюванні атаки, але й у побудові профілактичного механізму протидії гібридним ін'єкціям у реальному середовищі.

Внесок авторів: Дмитро Прокопович-Ткаченко – концептуалізація; методологія; узагальнення результатів; підготовка рубрик "Дискусія і висновки" й "Огляд літератури"; Олександр Черкаський – розроблення емпіричної методики; налаштування експериментального стенду GNS3 і MATLAB; проведення експериментів; Назар Зубченко – формалізація ризик-моделі SNMP; математичне моделювання механізму довіри PKI; реалізація ймовірнісної моделі; Максим Білан – розроблення сценаріїв атаки "SSL firmware injection + SNMP-активація" на Python; оброблення логів; візуалізація мережних метрик; Володимир Мацко – збір і аналіз даних; трасування SNMP-трафіка; підготовка графіків і таблиць; редагування й оформлення статті.

Джерела фінансування. Це дослідження не отримало жодного гранту від фінансової установи в державному, комерційному або некомерційному секторах.

Список використаної літератури

- Alam, T., Bhusal, D., Park, Y., & Rastogi, N. (2022). *CYNER: A Python Library for Cybersecurity Named Entity Recognition*. IEEE arXiv. <https://doi.org/10.48550/arXiv.2204.05754>
- bin Ibrahim, M., Continella, A., & Bianchi, A. (2023). *AoT – Attack on Things: A security analysis of IoT firmware updates*. IEEE EuroS&P. <https://doi.org/10.1109/eurosp57164.2023.00065>
- Cambosuela, L., Kaur, M., & Astya, R. (2024). *The Vulnerabilities and Risks of IoT in Cybersecurity*. IEEE ICRITO. <https://doi.org/10.1109/icrito61523.2024.10522460>
- Chien, H.-Y., & Wang, N. (2022). A Novel MQTT 5.0-Based Over-the-Air Updating Architecture Facilitating Stronger Security. *Electronics*, 11(23), 3899. <https://doi.org/10.3390/electronics11233899>
- Choi, B.-C., Lee, S.-H., Na, J.-C., & Lee, J.-H. (2016). Secure firmware validation and update for consumer devices. *IEEE Transactions on Consumer Electronics*, 62(3), 200–208. <https://doi.org/10.1109/TCE.2016.7448561>
- Crowther, K. G., Upadrashta, R., & Ramachandra, G. (2022). Securing Over-the-Air Firmware Updates (FOTA) for IIoT Devices. *Conference on Homeland Security* (pp. 569–574). IEEE. <https://doi.org/10.1109/HST56032.2022.10025441>
- Danese, A., Pravardelli, G., & Bertacco, V. (2017). *DOVE: Firmware Vulnerabilities via Control Flow Assertion Mining*. IEEE CODES+ISSS. <https://doi.org/10.1145/3125502.3125541>
- Dhobi, R., Gajjar, S., Parmar, D., & Vaghela, T. (2019). *Secure Firmware Update over the Air Using TrustZone*. IEEE I-PACT. <https://doi.org/10.1109/I-PACT44901.2019.8959992>
- Dorobantu, O. G., & Halunga, S. (2020). *Security Threats in IoT*. IEEE ISETC. <https://doi.org/10.1109/ISETC50328.2020.9301127>
- El Jaouhari, S. (2022). *Toward a Secure Firmware OTA Updates for Constrained IoT Devices*. IEEE ISC2. <https://doi.org/10.1109/ISC255366.2022.9922087>
- Fukuda, T., & Omote, K. (2021). *Efficient Blockchain-based IoT Firmware Update Considering Distribution Incentives*. IEEE DSC. <https://doi.org/10.1109/DSC49826.2021.9346265>
- He, X., Alqahtani, S., Gamble, R. F., & Papa, M. (2019). Securing Over-The-Air IoT Firmware Updates using Blockchain. *International Conference on Omni-Layer Intelligent Systems* (pp. 164–171). Association for Computing Machinery New York/United States. <https://doi.org/10.1145/3312614.3312649>
- Joshi, H., Anand, A. S., & Kokila, J. (2023). *Secure Firmware Update Architecture for IoT Devices Using Blockchain and PUF*. IEEE IQ-CCHES. <https://doi.org/10.1109/iq-cchess56596.2023.10391484>
- Kovačević, A., & Gligorić, N. (2024). Enhancing Security of Automotive OTA Firmware Updates via Decentralized Identifiers and Distributed Ledger Technology. *Electronics*, 13(23), 4640. <https://doi.org/10.3390/electronics13234640>
- Kumar, A., & Fung, C. (2024). *Analysis on CVE Vulnerabilities of the Internet of Things*. IEEE CIoT. <https://doi.org/10.1109/ciot63799.2024.10757121>
- Kumar, K. S., Sahoo, S. R., Kiran, K., & Mahapatra, K. (2018). *A Novel Holistic Security Framework for In-Field Firmware Updates*. IEEE ISES Proceedings. <https://doi.org/10.1109/ISES.2018.00063>
- Mansor, H., Markantonakis, K., Akram, R. N., & Mayes, K. (2015). *Let's Get Mobile: Secure FOTA for Automotive System*. Network and System Security. https://doi.org/10.1007/978-3-319-25645-0_38
- Mbakoyiannis, D., Tomoutzoglou, O., & Kornaros, G. (2019). *Secure over-the-air firmware updating for automotive electronic control units*. ACM SAC. <https://doi.org/10.1145/3297280.3297299>
- Mohapatra, M., & Snigdh, I. (2020). Security in IoT Healthcare. In *Security and Privacy in IoT: Models, Algorithms, and Implementations* (pp. 187–204). CRC Press. <https://doi.org/10.1201/9781003054115-11>
- Mitwa, N. S., Tarwireyi, P., Abu-Mahfouz, A. M., & Adigun, M. O. (2019). *Secure Firmware Updates in the Internet of Things: A Survey*. IEEE IMITEC. <https://doi.org/10.1109/IMITEC45504.2019.9015845>
- Najar-Pacheco, J. C., Bohada-Jaime, J. A., & Rojas-Moreno, W. Y. (2019). Vulnerabilities in the IoT. *Revista Facultad de Ingeniería*, 28(2), 63–74. <https://doi.org/10.14483/22484728.15163>
- Schulz, M., Iber, J., Dobaj, J., Kreiner, C., Boano, A., & Römer, K. (2018). *IoT Device Security the Hard(ware) Way*. European Conference on Pattern Languages. <https://doi.org/10.1145/3282308.3282329>
- Sedighfar, A. (2024). *A Study on Smart Homes Vulnerabilities*. IEEE FiCloud. <https://doi.org/10.1109/ficloud62933.2024.00026>
- Shakdhe, A., Agrawal, S., & Yang, B. (2019). *Security Vulnerabilities in Consumer IoT Apps*. IEEE BigDataSecurity. <https://doi.org/10.1109/BIGDATASECURITY-HPSC-IDS.2019.00012>
- Shin, Y., & Jeon, S. (2024). MQTree: Secure OTA Protocol Using MQTT and MerkleTree. *Sensors*, 24(5), 1447. <https://doi.org/10.3390/s24051447>
- Vidotto, M. (2022). Vulnerability Analysis for IoT Devices of Multi-agent Systems. In *Lecture Notes in Electrical Engineering*, Vol. 835. https://doi.org/10.1007/978-981-19-3998-3_141
- Wakisaka, Y. (2022). *Toward a Secure Firmware OTA Updates for Constrained IoT Devices*. IEEE ISC2. <https://doi.org/10.1109/ISC255366.2022.9922087>
- Witanto, E. N., Oktian, Y. E., Kumi, S., & Lee, S.-G. (2019). *Blockchain-based OCF Firmware Update*. IEEE ICTC. <https://doi.org/10.1109/ICTC46691.2019.8939910>
- Zandberg, K., Schleiser, K., Acosta, F., Tschofenig, H., & Baccelli, E. (2019). Secure Firmware Updates for Constrained IoT Devices Using Open Standards. *IEEE Access*, 7, 123456–123470. <https://doi.org/10.1109/ACCESS.2019.2919760>

References

- Alam, T., Bhusal, D., Park, Y., & Rastogi, N. (2022). *CYNER: A Python Library for Cybersecurity Named Entity Recognition*. IEEE arXiv. <https://doi.org/10.48550/arXiv.2204.05754>
- bin Ibrahim, M., Continella, A., & Bianchi, A. (2023). *AoT – Attack on Things: A security analysis of IoT firmware updates*. IEEE EuroS&P. <https://doi.org/10.1109/eurosp57164.2023.00065>
- Cambosuela, L., Kaur, M., & Astya, R. (2024). *The Vulnerabilities and Risks of IoT in Cybersecurity*. IEEE ICRITO. <https://doi.org/10.1109/icrito61523.2024.10522460>
- Chien, H.-Y., & Wang, N. (2022). A Novel MQTT 5.0-Based Over-the-Air Updating Architecture Facilitating Stronger Security. *Electronics*, 11(23), 3899. <https://doi.org/10.3390/electronics11233899>
- Choi, B.-C., Lee, S.-H., Na, J.-C., & Lee, J.-H. (2016). Secure firmware validation and update for consumer devices. *IEEE Transactions on Consumer Electronics*, 62(3), 200–208. <https://doi.org/10.1109/TCE.2016.7448561>



- Crowther, K. G., Upadrashta, R., & Ramachandra, G. (2022). Securing Over-the-Air Firmware Updates (FOTA) for IIoT Devices. *Conference on Homeland Security* (pp. 569–574). IEEE. <https://doi.org/10.1109/HST56032.2022.10025441>
- Danese, A., Pravadelli, G., & Bertacco, V. (2017). DOVE: Firmware Vulnerabilities via Control Flow Assertion Mining. *IEEE CODES+ISSS*. <https://doi.org/10.1145/3125502.3125541>
- Dhobi, R., Gajjar, S., Parmar, D., & Vaghela, T. (2019). Secure Firmware Update over the Air Using TrustZone. *IEEE I-PACT*. <https://doi.org/10.1109/I-PACT44901.2019.8959992>
- Dorobantu, O. G., & Halunga, S. (2020). Security Threats in IIoT. *IEEE ISETC*. <https://doi.org/10.1109/ISETC50328.2020.9301127>
- El Jaouhari, S. (2022). Toward a Secure Firmware OTA Updates for Constrained IIoT Devices. *IEEE ISC2*. <https://doi.org/10.1109/ISC255366.2022.9922087>
- Fukuda, T., & Omote, K. (2021). Efficient Blockchain-based IIoT Firmware Update Considering Distribution Incentives. *IEEE DSC*. <https://doi.org/10.1109/DSC49826.2021.9346265>
- He, X., Alqahtani, S., Gamble, R. F., & Papa, M. (2019). Securing Over-The-Air IIoT Firmware Updates using Blockchain. *International Conference on Omni-Layer Intelligent Systems* (pp. 164–171). Association for Computing Machinery New York NY United States. <https://doi.org/10.1145/3312614.3312649>
- Joshi, H., Anand, A. S., & Kokila, J. (2023). Secure Firmware Update Architecture for IIoT Devices Using Blockchain and PUF. *IEEE IQ-CCHES*. <https://doi.org/10.1109/iq-cchess56596.2023.10391484>
- Kovačević, A., & Gligorić, N. (2024). Enhancing Security of Automotive OTA Firmware Updates via Decentralized Identifiers and Distributed Ledger Technology. *Electronics*, 13(23), 4640. <https://doi.org/10.3390/electronics13234640>
- Kumar, A., & Fung, C. (2024). Analysis on CVE Vulnerabilities of the Internet of Things. *IEEE CIIoT*. <https://doi.org/10.1109/ciiot63799.2024.10757121>
- Kumar, K. S., Sahoo, S. R., Kiran, K., Swain, A. K., & Mahapatra, K. (2018). A Novel Holistic Security Framework for In-Field Firmware Updates. *IEEE ISES Proceedings*. <https://doi.org/10.1109/ISES.2018.00063>
- Mansor, H., Markantonakis, K., Akram, R. N., & Mayes, K. (2015). Let's Get Mobile: Secure FOTA for Automotive System. *Network and System Security*. https://doi.org/10.1007/978-3-319-25645-0_38
- Mbakoyiannis, D., Tomoutzoglou, O., & Kornaros, G. (2019). Secure over-the-air firmware updating for automotive electronic control units. *ACM SAC*. <https://doi.org/10.1145/3297280.3297299>
- Mohapatra, M., & Snigdha, I. (2020). Security in IIoT Healthcare. In *Security and Privacy in IIoT: Models, Algorithms, and Implementations* (pp. 187–204). CRC Press. <https://doi.org/10.1201/9781003054115-11>
- Mtewa, N. S., Tarwireyi, P., Abu-Mahfouz, A. M., & Adigun, M. O. (2019). Secure Firmware Updates in the Internet of Things: A Survey. *IEEE IMITEC*. <https://doi.org/10.1109/IMITEC45504.2019.9015845>
- Najar-Pacheco, J. C., Bohada-Jaime, J. A., & Rojas-Moreno, W. Y. (2019). Vulnerabilities in the IIoT. *Revista Facultad de Ingeniería*, 28(2), 63–74. <https://doi.org/10.14483/22484728.15163>
- Schulz, M., Iber, J., Dobaj, J., Kreiner, C., Boano, A., & Römer, K. (2018). IIoT Device Security the Hard(ware) Way. *European Conference on Pattern Languages*. <https://doi.org/10.1145/3282308.3282329>
- Sedighfar, A. (2024). A Study on Smart Homes Vulnerabilities. *IEEE FiCloud*. <https://doi.org/10.1109/ficloud62933.2024.00026>
- Shakdhe, A., Agrawal, S., & Yang, B. (2019). Security Vulnerabilities in Consumer IIoT Apps. *IEEE BigDataSecurity*. <https://doi.org/10.1109/BIGDATASECURITY-HPSC-IDS.2019.00012>
- Shin, Y., & Jeon, S. (2024). MQTree: Secure OTA Protocol Using MQTT and MerkleTree. *Sensors*, 24(5), 1447. <https://doi.org/10.3390/s2405144>
- Vidotto, M. (2022). Vulnerability Analysis for IIoT Devices of Multi-agent Systems. In *Lecture Notes in Electrical Engineering*, Vol. 835. https://doi.org/10.1007/978-981-19-3998-3_141
- Wakisaka, Y. (2022). Toward a Secure Firmware OTA Updates for Constrained IIoT Devices. *IEEE ISC2*. <https://doi.org/10.1109/ISC255366.2022.9922087>
- Witanto, E. N., Oktian, Y. E., Kumi, S., & Lee, S.-G. (2019). Blockchain-based OCF Firmware Update. *IEEE ICTC*. <https://doi.org/10.1109/ICTC46691.2019.8939910>
- Zandberg, K., Schleiser, K., Acosta, F., Tschofenig, H., & Baccelli, E. (2019). Secure Firmware Updates for Constrained IIoT Devices Using Open Standards. *IEEE Access*, 7, 123456–123470. <https://doi.org/10.1109/ACCESS.2019.2919760>

Отримано редакцією журналу / Received: 15.10.25

Прорецензовано / Revised: 17.10.25

Схвалено до друку / Accepted: 21.11.25

Dmytro PROKOPOVYCH-TKACHENKO^{1,2}, PhD (Engin.), Assoc. Prof.

ORCID ID: 0000-0002-6590-3898

e-mail: omega2417@gmail.com

Oleksandr CHERKASKYI³, Independent Researcher

ORCID ID: 0009-0006-3105-5217

e-mail: asherjoseph.c@gmail.com

Nazar ZUBCHENKO¹, Independent Researcher

ORCID ID: 0009-0000-9973-7624

e-mail: nazik3110@gmail.com

Maksym BILAN¹, Applicant

ORCID ID: 0009-0005-5875-4950

e-mail: bilan_maksym@umsf.dp.ua

Volodymyr MATSKO¹, Independent Researcher

ORCID ID: 0009-0007-9091-4891

e-mail: matzkovvv@gmail.com

¹ University of Customs and Finance, Dnipro, Ukraine

² State Scientific Institution "Institute of Information, Security and Law" of the National Academy of Legal Sciences of Ukraine, Kyiv, Ukraine

³ Dnipro University of Technology, Dnipro, Ukraine

COMPROMISE OF THE UPDATE CHAIN VIA SSL AND SNMP AS AN ATTACK VECTOR ON TELECOMMUNICATION NODES: TECHNICAL MODELING AND COUNTERMEASURES

Background. This paper provides a comprehensive analysis of targeted attacks on critical telecommunication nodes by compromising firmware update channels, particularly through spoofing or hijacking SSL update servers and exploiting SNMPv2 vulnerabilities. The increasing complexity and scale of cyber threats in telecommunications underscore the urgency of studying this attack vector. Adversaries increasingly spoof firmware update servers (e.g., update.domain.tld) to deliver modified software into critical network elements—such as the Base Station Controller (BSC), Mobile Switching Center (MSC), and Evolved Node B (eNodeB)—and subsequently activate hidden malware via SNMP commands.

Methods. We developed a formalized technical model of firmware update chain compromise based on threat analysis of the SNMP subsystem and detailed scenario-based modeling of network node behavior under modified firmware. A probabilistic risk analysis of SNMP agent access was conducted, along with a structural assessment of the PKI trust chain. A series of emulation experiments were performed in MATLAB, GNS3, and specialized telecom testbeds.



Results. We detail a complex attack scenario combining SSL update server spoofing, firmware modification, and SNMP command activation using default community strings. Modeling demonstrates that reliance on outdated protocols and weak authentication significantly increases the risk of losing control over BSC, MSC, and eNodeB nodes. The attack's impact extends beyond local elements, potentially causing cascading failures and prolonged service disruptions across operators' networks.

Conclusions. Our study confirms the urgent need to adopt SNMPv3 with cryptographic protection, implement TLS-pinning to verify update server authenticity, and employ a hardware root of trust for firmware integrity checks at the physical level. Attack profile analysis identified the most vulnerable points in the network infrastructure and informed recommendations for updating telecommunication cybersecurity policies.

Keywords: update chain compromise; SNMPv2; firmware attack; TLS-pinning; SSL server; telecom infrastructure; BSC; eNodeB.

Автори заявляють про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; в рішенні про публікацію результатів.

The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; in the decision to publish the results.