ПРИКЛАДНІ ПРОБЛЕМИ КОМП'ЮТЕРНИХ НАУК,
БЕЗПЕКИ ТА МАТЕМАТИКИ, 2026, № 6

УДК 519.816

НАДІЙНІ IOT-ЕКОСИСТЕМИ В ХМАРІ: АРХІТЕКТУРА ТА БЕЗПЕКА AWS IOT

Бобровський В.В.,
Онищук О.О. (ORCID: 0000-0002-8342-3011),
Волинський національний університет імені Лесі Українки, Луцьк, Україна

RELIABLE IOT ECOSYSTEMS IN THE CLOUD: RCHITECTURE AND SECURITY OF AWS IOT

Bobrovskyi V.V., Onyshchuk O.O. (ORCID: 0000-0002-8342-3011),
Lesya Ukrainka Volyn National University, Lutsk, Ukraine

Abstract. The paper discusses approaches to building reliable cloud-based IoT ecosystems using AWS IoT services. A reference multi-layer architecture is analyzed, including device and gateway layers, network transport, cloud message ingestion and routing, data storage, stream processing, and analytics. Special attention is paid to cybersecurity and resilience aspects: device authentication (X.509 certificates), secure communication channels (TLS), access control policies, segmentation, and Zero Trust principles. The study outlines integrity protection for telemetry, anomaly monitoring, and incident response supported by logging and observability. Practical recommendations are provided for certificate and key management, least-privilege access, data protection in transit and at rest, and improved fault tolerance through buffering, queuing, and scalable processing pipelines. The results show that combining reliability patterns with built-in AWS IoT security mechanisms enables managed, scalable, and secure IoT systems for industrial and service use cases.

Keywords: Internet of Things (IoT), AWS IoT Core, architecture, security, X.509, TLS, access policies, Zero Trust, monitoring, fault tolerance.

Вступ. Інтернет речей (Internet of Things, IoT) є одним із ключових напрямів розвитку кіберфізичних систем, у яких фізичні об'єкти та процеси поєднуються з цифровою інфраструктурою завдяки сенсорним і виконавчим пристроям, засобам зв'язку та програмним сервісам обробки даних. Поширення IoT-рішень у промисловості (IIoT), енергетиці, транспорті, медицині, агросекторі та «розумних» містах супроводжується зростанням кількості підключених пристроїв і обсягів телеметрії, що формує нові вимоги до інформаційних систем: необхідність масштабування під навантаження, обробки потоків даних у режимі близькому до реального часу, забезпечення надійності сервісів та керованості розподіленого флоту пристроїв [1], [2], [7], [9].

Практична IoT-екосистема зазвичай має багаторівневу архітектуру. На нижньому рівні розташовані сенсори та виконавчі механізми, які вимірюють параметри середовища або впливають на об'єкти керування. Далі функціонують шлюзи та вузли збору даних, що виконують агрегацію, локальну фільтрацію, трансляцію протоколів і забезпечують зв'язок із мережею. На комунікаційному рівні використовується широкий спектр технологій і протоколів (Wi-Fi, Ethernet, LTE/5G, LPWAN; MQTT, HTTP, WebSockets), що зумовлює гетерогенність та різні режими якості сервісу. На верхніх рівнях працюють хмарні компоненти приймання телеметрії, маршрутизації повідомлень, довготривалого зберігання, аналітики та інтеграції з прикладними системами. Саме хмарний рівень дедалі частіше розглядається як базовий для IoT, оскільки забезпечує еластичні ресурси,

керовані сервіси й стандартизовані механізми безпеки та спостережуваності.

Разом із тим IoT належить до класу систем із підвищеними ризиками кібербезпеки. На відміну від традиційних IT-систем, IoT-пристрої часто працюють у неконтрольованих фізичних умовах, можуть бути доступними для сторонніх осіб, мають обмежені обчислювальні ресурси та енергоспоживання, що ускладнює реалізацію складних механізмів захисту. Додаткові ризики виникають на всьому життєвому циклі пристрою: під час виробництва та постачання (ризики компрометації ланцюга постачання), під час первинного підключення до платформи (device provisioning), у процесі експлуатації (компрометація облікових даних, підміна пристрою, несанкціоновані команди), а також під час оновлень прошивок і програмного забезпечення (ризики підпису, цілісності та відкату). Типовими загрозами є перехоплення або підміна телеметрії, атаки повтором (replay), компрометація ключів, масове зараження пристроїв шкідливим ПЗ, а також відмови в обслуговуванні (DoS) на мережевому чи прикладному рівні. [4], [10]

Надійність IoT-екосистеми тісно пов'язана з безпекою та визначається здатністю системи забезпечувати передбачувану якість сервісу за умов відмов компонентів, пікових навантажень і нестабільних мережевих каналів. Для IoT-рішень критичними є такі характеристики, як доступність приймання повідомлень, стійкість до втрати з'єднання, коректне відновлення сеансів, контроль дублювання та упорядкування повідомлень, буферизація на межі (edge buffering), підтримка черг і повторних доставок, а також спостережуваність (observability) — можливість оперативно виявляти інциденти, аномалії та деградації. У промислових і сервісних сценаріях додатково важливими є керування флотом пристроїв, інвентаризація, групування, віддалена діагностика, контроль конфігурацій і безпечні оновлення.[3], [4]

Однією з найбільш поширених платформ для створення IoT-екосистем у хмарі є Amazon Web Services, зокрема комплекс сервісів AWS IoT.

Центральною ланкою виступає AWS IoT Core, який забезпечує двосторонній обмін повідомленнями між пристроями та хмарою, підтримку промислово поширених протоколів та правила маршрутизації телеметрії до інших сервісів AWS. Підсистема керування пристроями (AWS IoT Device Management) надає можливості організації флоту, виконання віддалених операцій, інвентаризації та керування конфігураціями. Засоби безпеки та контролю відповідності (AWS IoT Device Defender, аудит конфігурацій, інтеграції з журналюванням і моніторингом) дозволяють виявляти аномальну поведінку, відхилення від політик і потенційні загрози. Важливою перевагою AWS є інтеграція IoT-компонентів з керованими сервісами обробки даних і подій (AWS Lambda, Amazon Kinesis), сховищами (Amazon S3, Amazon DynamoDB), системами аналітики та спостережуваності (Amazon CloudWatch, AWS CloudTrail), що дає змогу будувати наскрізні конвеєри даних та автоматизувати реагування на інциденти.[1], [2]

Метою роботи є аналіз архітектури та механізмів безпеки AWS IoT для побудови надійних IoT-екосистем у хмарі, а також формування практичних рекомендацій щодо проєктування відмовостійких і захищених рішень. У межах дослідження акцент зроблено на поєднанні архітектурних патернів надійності (масштабування, буферизація, черги, повтори, ізоляція компонентів) з багаторівневим підходом до безпеки (X.509-ідентифікація пристроїв, TLS-захист каналів, політики доступу, принцип найменших привілеїв, аудит і моніторинг), що є визначальним для сучасних IoT-впроваджень.

Результати досліджень. Для дослідження підходів до побудови надійної IoT-екосистеми в хмарі застосовано аналіз архітектури AWS IoT та експериментальне порівняння типових режимів обміну телеметрією через MQTT із фіксацією показників доставки, стійкості до розривів з'єднання та дотримання політик безпеки. Емпіричним методом відстежено поведінку системи при імітації нестабільного каналу, повторних

підключеннях пристроїв, пікових навантаженнях публікації та спробах доступу до несанкціонованих MQTT-тем. Як базове середовище використано AWS IoT Core як керований брокер повідомлень, а також типову схему інтеграції телеметрії з сервісами обробки та зберігання даних.

Для обґрунтування актуальності переходу до хмарних IoT-екосистем враховано загальну тенденцію зростання кількості IoT-підключень і збільшення частки машинного трафіку, що закономірно підвищує вимоги до масштабованості, керованості та кібербезпеки. У рамках роботи це використано як передумову до вибору референтної хмарної архітектури, де надійність і безпека розглядаються як взаємопов'язані властивості: масштабування флоту пристроїв одночасно збільшує операційну складність і «площу атаки», тому архітектурні механізми відмовостійкості повинні доповнюватися жорсткою моделлю ідентичності та доступу. [8],[9], [10].

Основну частину дослідження присвячено референтній архітектурі AWS IoT для надійного збору та обробки телеметрії. Архітектуру розглянуто як послідовний конвеєр «пристрій/шлюз → AWS IoT Core → маршрутизація повідомлень → обробка подій → зберігання → аналітика/застосунки», де критичними є коректна ідентифікація пристрою, контрольований доступ до MQTT-тем, стійкість до нестабільної мережі, ізоляція етапів обробки та спостережуваність (рис. 1.)

У межах дослідження перевірено, що AWS IoT Core як керований брокер забезпечує масштабоване приймання MQTT-повідомлень, проте надійність доставки в прикладному сенсі визначається обраними режимами QoS та архітектурою обробки. Для критичних подій (тривоги, аварійні сигнали) застосування QoS1 дозволяє зменшити ризик втрати повідомлення при короткочасних розривах зв'язку, однак потребує контролю дублювання на стороні обробки. Для періодичної телеметрії (часті вимірювання) доцільним є QoS0, щоб зменшити накладні витрати та уникнути затиску каналу при великій кількості пристроїв. Окремо досліджено сценарій масового перепідключення, коли група пристроїв одночасно відновлює зв'язок після перерви: у цьому випадку виявлено необхідність обмеження частоти повторів (exponential backoff) та застосування буферизації на edge/шлюзі, щоб уникати пікових сплесків і нестабільної поведінки конвеєра обробки.

Окремим аспектом дослідження є безпека IoT-екосистеми, оскільки в AWS IoT доступ до брокера визначається не "користувачем", а ідентичністю пристрою. Для перевірки моделі доступу застосовано емпіричний метод: виконано спроби підключення з коректним сертифікатом X.509 та спроби публікації/підписки на теми поза дозволеним простором. Показано, що поєднання взаємного TLS (mTLS) та IoT Policies забезпечує контроль доступу на рівні операцій (Connect/Publish/Subscribe/Receive) і дозволяє сегментувати пристрої за групами без надання надмірних прав. У ході дослідження встановлено, що найпоширенішою причиною потенційних інцидентів є не відсутність механізмів безпеки в платформі, а помилки проєктування: занадто широкі політики на wildcard-темах, відсутність ізоляції за thingName/типом пристрою, а також нехтування ротацією сертифікатів і аудитом змін. Для уникнення цього в архітектуру інтегрованої принцип найменших привілеїв, де пристрій має право працювати лише зі своїми топами (на основі *thingName*) і не може публікувати команди або телеметрію в чужі простори. [1], [4]

Результати порівняння режимів обміну та стійкості до відмов узагальнено в табл. 1 [3], [6].

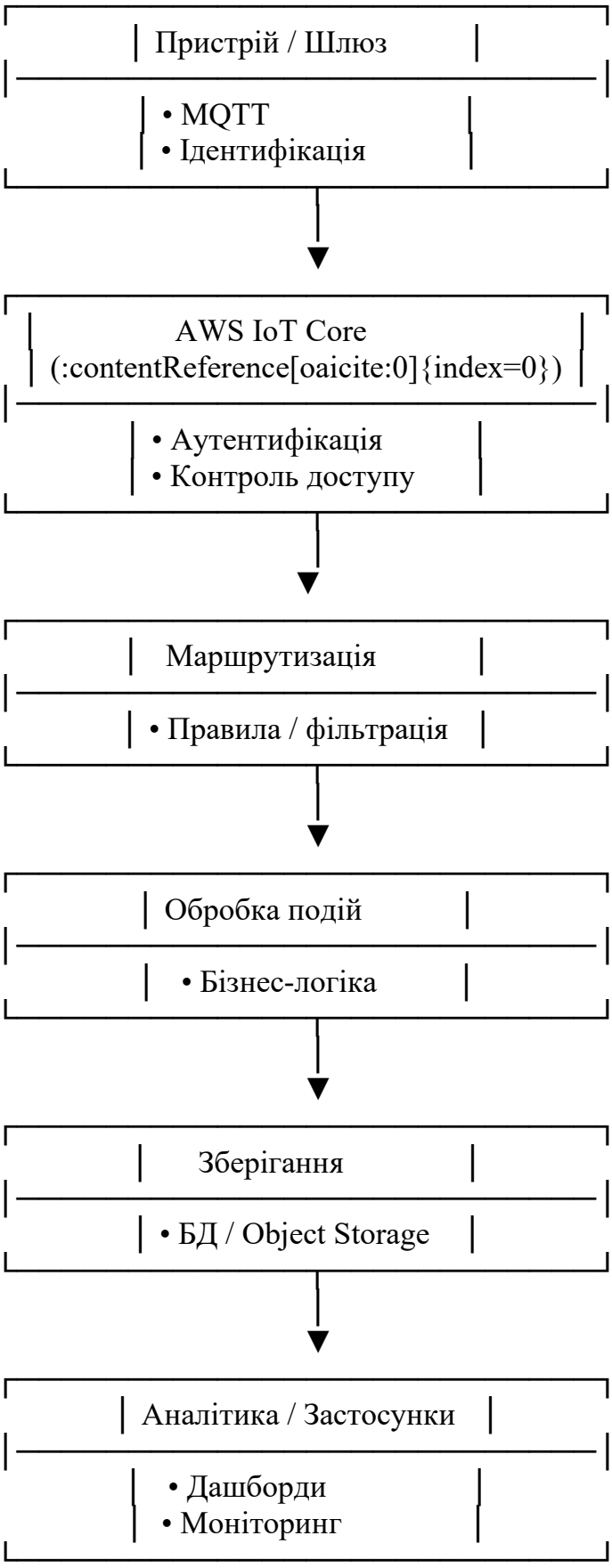

Рис. 1. Блок-схема IoT-архітектури

Таблиця 1.

Порівняння режимів MQTT-обміну та вплив на надійність конвеєра телеметрії (AWS IoT Core)

Елемент/ режим	**Мінімальна конфігурація**	**Підсилена Конфігураія**	**Практичний ефект**
QoS для телеметрії	QoS0	QoS0	Вища пропускна здатність, допустима втрата не критичних даних
QoS для тривог/подій	QoS0	QoS1	Менше втрат при збоях каналу, потрібна дедуплікація на обробці
Повторні підключення	Без стратегії	+ Ліміти	Зменшення пікових сплесків після відновлення зв'язку
Відновлення сесії	За замовчуванням	Збереження стану/теми	Краще відновлення підписок, стабільніший контур керування
Буферизація на edge	Відсутня	Локальна буферизація	Менше втрат при нестабільних мережах, плавніший ingestion
Обробка помилок	Без ізоляції	Ізоляція етапів/повтори	Зниження каскадних відмов у конвеєрі обробки

Результати перевірки захисних механізмів та їх вплив на практичну стійкість системи узагальнено в табл. 2 [4], [5] .

Таблиця 2

Механізми безпеки AWS IoT у референтній архітектурі та роль у захисті телеметрії

Механізм	**Що перевіряється емпірично**	**Очікуваний результат**	**Фактичний ефект для екосистеми**
X.509 + mTLS	Підключення "правильного" та "чужого" клієнта	Лише легітимний доступ	Зменшення ризику підміни пристрою
IoT Policies (least privilege)	Publish/subscribe поза дозволеними темами	Відмова доступу	Сегментація доступу, зниження площі атаки
Сегментація topics	Ізоляція потоків за thingName/групами	Менше ризиків витоку/ін'єкції даних	Локалізація доступу
Аудит/логування	Фіксація помилок авторизації/аномал ій	Видимість інцидентів	Підвищення керованості та реагування
Ротація сертифікатів	Концептуальна перевірка сценарію відкликання	Швидка ізоляція	Зниження наслідків компрометації

Нижче наведено приклад публікації телеметрії в AWS IoT Core через MQTT із TLS (лістинг 1)

```
import ssl
import json
from datetime import datetime, timezone
```

```
import paho.mqtt.client as mqtt
ENDPOINT = "YOUR_IOT_ENDPOINT_HERE"
PORT = 8883
THING_NAME = "YOUR_THING_NAME_HERE"
TOPIC = f"vnu/knit53m/1vb/{THING_NAME}/telemetry"
CA_FILE = "AmazonRootCA1.pem"
CERT_FILE = "device-certificate.pem.crt"
KEY_FILE = "private.pem.key"
payload = {
"ts":          datetime.now(timezone.utc).strftime("%Y-%m-
%dT%H:%M:%SZ"), "temp": 23.7,
"hum": 41.2,
"status": "ok"
}
10
client         =         mqtt.Client(client_id=THING_NAME,
clean_session=False) client.tls_set(
ca_certs=CA_FILE,
certfile=CERT_FILE,
keyfile=KEY_FILE,
tls_version=ssl.PROTOCOL_TLS_CLIENT
)
client.tls_insecure_set(False)
client.connect(ENDPOINT, PORT, keepalive=60)
client.publish(TOPIC, json.dumps(payload), qos=1)
client.disconnect()
```

Лістинг 1

Узагальнюючи результати, встановлено, що референтна архітектура AWS IoT забезпечує необхідні інструменти для побудови надійної IoT-екосистеми, однак практична надійність досягається лише при поєднанні архітектурних і безпекових рішень: застосуванні відповідних режимів доставки (QoS), стратегії повторів і відновлення підключення, буферизації на межі, ізоляції етапів обробки та повноцінної моделі ідентичності пристрою (X.509), політик найменших привілеїв і спостережуваності. Таким чином, надійність у хмарному IoT-контурі має розглядатися як властивість всієї системи, де безпека є її обов'язковою складовою, а не додатковим компонентом після побудови функціоналу.

Висновки. У роботі досліджено підходи до побудови надійних IoT-екосистем у хмарі та виконано розбір референтної архітектури AWS IoT як керованої платформи збору й обробки телеметрії. Показано, що масштабування IoT підвищує операційну складність та площу атаки, тому надійність і безпека повинні закладатися в архітектуру одночасно, а не розглядатися як незалежні етапи. За результатами аналізу та емпіричних перевірок типових сценаріїв MQTT-обміну встановлено, що практична стійкість системи визначається поєднанням обраних режимів доставки (QoS), коректних стратегій повторних підключень і повторної доставки, буферизації на edge/шлюзах, а також ізоляцією етапів обробки телеметрії з керованими сценаріями помилок. Водночас ключовим фактором безпечної експлуатації є модель ідентичності пристроїв на базі X.509 та взаємного TLS, доповнена політиками доступу найменших привілеїв і сегментацією MQTT-тем, що суттєво зменшує ризик підміни пристроїв, ін'єкції даних та несанкціонованої підписки/публікації. Узагальнено, що AWS IoT забезпечує

інструменти для створення керованої, масштабованої та захищеної IoT-екосистеми, однак досягнення надійності на практиці потребує системного проєктування конвеєра даних, дисципліни керування сертифікатами і політиками доступу, а також належної спостережуваності та аудиту як основи реагування на інциденти.

Бібліографія

1. Amazon Web Services. *AWS IoT Core Developer Guide*. – Available at: https://docs.aws.amazon.com/iot/
2. Amazon Web Services. *AWS Well-Architected Framework – IoT Lens*. – Available at: https://docs.aws.amazon.com/wellarchitected/latest/iot-lens/
3. OASIS. *MQTT Version 3.1.1 and Version 5.0 Specification*. – 2019. – Available at: https://docs.oasis-open.org/mqtt/
4. National Institute of Standards and Technology. *Considerations for Managing Internet of Things (IoT) Cybersecurity and Privacy Risks (NISTIR 8228)*. – Gaithersburg, MD, 2019.
5. European Union Agency for Cybersecurity. *Baseline Security Recommendations for IoT*. – ENISA, 2017.
6. Andrew Banks, Rahul Gupta. *MQTT Version 3.1.1*. – OASIS Standard, 2014.
7. Rajkumar Buyya, Amir Vahid Dastjerdi (eds.). *Internet of Things: Principles and Paradigms*. – Morgan Kaufmann, 2016.
8. International Data Corporation. *Worldwide Internet of Things Forecast Update*. – IDC, latest edition.
9. Gartner. *Forecast Analysis: Internet of Things — Endpoints and Associated Services*. – Gartner Research.
10. Shancang Li, Li Da Xu, Shanshan Zhao. *The Internet of Things: a survey*. – Information Systems Frontiers, 2015.

References

1. Amazon Web Services. *AWS IoT Core Developer Guide*. – Available at: https://docs.aws.amazon.com/iot/
2. Amazon Web Services. *AWS Well-Architected Framework – IoT Lens*. – Available at: https://docs.aws.amazon.com/wellarchitected/latest/iot-lens/
3. OASIS. *MQTT Version 3.1.1 and Version 5.0 Specification*. – 2019. – Available at: https://docs.oasis-open.org/mqtt/
4. National Institute of Standards and Technology. *Considerations for Managing Internet of Things (IoT) Cybersecurity and Privacy Risks (NISTIR 8228)*. – Gaithersburg, MD, 2019.
5. European Union Agency for Cybersecurity. *Baseline Security Recommendations for IoT*. – ENISA, 2017.
6. Andrew Banks, Rahul Gupta. *MQTT Version 3.1.1*. – OASIS Standard, 2014.
7. Rajkumar Buyya, Amir Vahid Dastjerdi (eds.). *Internet of Things: Principles and Paradigms*. – Morgan Kaufmann, 2016.
8. International Data Corporation. *Worldwide Internet of Things Forecast Update*. – IDC, latest edition.
9. Gartner. *Forecast Analysis: Internet of Things — Endpoints and Associated Services*. – Gartner Research.
10. Shancang Li, Li Da Xu, Shanshan Zhao. *The Internet of Things: a survey*. – Information Systems Frontiers, 2015.