Advancing IoT-Oriented Network Simulation: Challenges, Gaps, and Opportunities

Mr. Ramanuja Nayak
School of Computer Applications,
MITS
Rayagada,Odisha.
ramanuja.nayak@gmail.com

Dr. Tapaswini Nayak
School of Computer Applications,
MITS
Rayagada,Odisha.
nayak_roma@yahoo.co.in

Ganta Vasanth Kumar
School of Computer Applications,
MITS
Rayagada,Odisha.
devrajpujari67@gmail.com

Abstract-In Today's busy lifestyle, pet owners face many problems in ensuring regular and balanced feeding for their beloved pets ensuring pets receive the right amount of food at the right time. This Smart pet feeder addresses this issue by employing a based device to offer a fully automated smart feeder. The convenience of the owner and the well-being of the pets are both enhanced by this innovative system incorporates various technologies and devices. When feeding time arrives, a speaker with pre-recorded voice messages gently calls the pet to the food area, ensuring they are aware it's time to eat, in case the pet does not eat food immediately or pet leaves after eating a small portion of food then the system will call pet with a periodic reminder, encouraging them to eat food at their own pace. The device closes the pet's food bowl automatically after the pet leaves the pet feeder to guard against contamination and food spoiling and maintain the quality of the food. It includes real-time food level monitoring, and calling the pet at a scheduled time. It ensures that pets receive timely and sufficient meals, reduces food waste, and safeguards food quality.

Keywords: Microcontroller, Load Sensor, Low-cost Moisture, ESP-32, Embedded System

I. INTRODUCTION

The Internet of Things (IoT), fueled by rapid advances in connectivity and the backing of major technology players, is steadily shaping a future where billions of devices interact seamlessly in a "smart society." Yet, despite its promise, IoT still faces significant challenges before widespread, reliable, and secure adoption can be realized. A key enabler of IoT research and development is the use of simulation tools, which provide controlled environments for testing protocols, designing communication systems, and validating architectures without the prohibitive costs of deploying large-scale hardware testbeds. Such tools play essential roles in three areas: first, in protocol development for academia and standards bodies, where new algorithms and communication mechanisms are tested; second, in industrial evaluation, where simulations help answer critical "what-if" questions such as energy consumption or scalability under real-world constraints; and third, in education, where simulators allow future engineers to study and experiment with IoT systems without the need for expensive infrastructures. These diverse applications underline the importance of simulation frameworks in the IoT ecosystem.

Despite these advantages, IoT simulation remains a challenging domain. Open-source tools such as ns-3 have made significant contributions, yet they fall short in several respects. Current limitations include incomplete support for important standards like IEEE 802.15.4e TSCH, Zigbee, LoRaWAN, and Matter; insufficient energy and security modeling; weak integration with real hardware for hybrid experiments; and a lack of user-friendly graphical interfaces. The fast-paced evolution of IoT protocols, combined with fragmented adoption in real deployments, makes it difficult for simulators to remain relevant without consistent updates and long-term maintenance. Furthermore, while simulators are often praised for flexibility, many fail to capture fine-grained real-world phenomena such as nonlinear battery discharge, cryptographic overhead, and multi-stack interoperability. This gap between simulation fidelity and hardware reality risks undermining the value of simulation in both academia and industry.

Addressing these shortcomings requires moving beyond perspective-style discussions and into concrete, validated implementations. While prior works have highlighted the importance of simulation in IoT research, there remains an evident lack of empirical validation and reproducibility frameworks. Few studies have systematically calibrated simulation models against real-world traces, and even fewer provide open, community-maintained artefacts such as benchmark suites, scenario descriptors, or validated energy/security models. Bridging this gap is crucial, as standardized benchmarks and reference implementations could significantly accelerate both academic innovation and industrial adoption. By producing reproducible, validated, and extensible modules, researchers can add tangible value beyond identifying problems.

To this end, future research should focus on actionable improvements. Developing a realistic TSCH (802.15.4e) module for ns-3 and validating it against testbed traces would address one of the most pressing standardization gaps. Similarly, designing ns-3–to–hardware emulation bridges via UART/SPI shims would enable mixed simulation–hardware testing, enhancing fidelity and practical relevance. Another priority is energy modeling: creating nonlinear battery and

radio state-based models validated through laboratory experiments would allow for more accurate assessments of device lifetime. Security modeling also deserves deeper exploration, particularly in quantifying the performance and energy costs of cryptographic suites like AES-CCM or DTLS. Finally, usability improvements such as GUI-based floorplan importers and JSON-based scenario descriptors would democratize simulator use for both researchers and students.

Equally important is the need for sustainability in the IoT simulation ecosystem. Without consistent maintenance, even well-designed modules risk becoming obsolete. Community-driven benchmark suites, automated regression testing, and open trace repositories could provide the foundation for long-term reliability and comparability across simulators. Such measures would not only support reproducibility but also enable fair evaluations between competing standards and stacks, ultimately guiding both academia and industry toward more informed decisions.

In this research, we aim to build on these gaps and opportunities by proposing and implementing validated improvements to IoT simulation. Our work will move beyond perspectives into practical contributions, delivering open artefacts, reproducible benchmarks, and experimentally verified models that can help close the divide between simulation and reality.

II. LITERATURE REVIEW

The rapid growth of the Internet of Things (IoT) has created a pressing demand for reliable, scalable, and reproducible evaluation platforms that support academic research, industrial experimentation, and standards development. Network simulators such as ns-3 have emerged as vital tools, enabling researchers to test protocol performance and validate designs before deployment. The paper *"Perspectives on IoT-oriented network simulation systems"* highlights the evolving needs of IoT simulation, pointing out strengths such as ns-3's support for IEEE 802.15.4, 6LoWPAN, and ongoing LoRaWAN efforts, while also underlining gaps in fidelity, usability, and long-term support. Although these insights are valuable, the work is primarily a perspective piece with limited empirical validation, emphasizing what should be prioritized but not demonstrating concrete implementations or benchmark-driven evaluations.

Existing studies often discuss protocol-specific implementations or isolated improvements to simulators, but few take a holistic view of how simulation frameworks can be systematically extended and validated against real-world IoT deployments. The reviewed paper identifies the need for flexible and well-maintained simulation environments but does not provide standardized benchmarks, reproducible workflows, or validated tools for features like energy modeling, emulation with real hardware, or security overhead representation. This leaves researchers without concrete methodologies to ensure that simulations match the complexity and constraints of practical IoT scenarios.

Addressing these limitations presents a significant research opportunity. Gaps include the absence of validated trace-based calibration frameworks, the lack of accurate nonlinear battery and energy consumption models integrated with transceiver state machines, and the missing representation of cryptographic overheads in simulations. Furthermore, despite calling for improved usability through graphical interfaces and reproducibility through exportable experiment descriptors, no tangible prototypes are offered. There is also a notable lack of community-driven benchmark suites and regression processes to sustain simulator modules over time. Building upon these shortcomings, future research can contribute by moving from conceptual priorities to concrete, validated prototypes — such as developing an ns-3 TSCH (802.15.4e) module, designing emulator bridges to real hardware, or creating open-source GUI and floorplan-import tools that generate reproducible experiment scenarios. By doing so, new research would not only fill the highlighted technical gaps but also provide empirical evidence, open datasets, and reusable artefacts that enhance reproducibility, usability, and practical relevance of IoT-oriented simulation systems.

III. DISCRETE EVENT NETWORK SIMULATIONS FOR IoT DEVELOPMENT

Ideally, IoT-oriented discrete event simulators should not only capture the communication dynamics of low-power wireless networks but also reflect real-world performance trade-offs such as energy consumption, security overhead, and protocol interoperability with a measurable level of fidelity. Depending on their objectives, users may demand different levels of granularity. Some users are content with basic connectivity models to approximate throughput and latency, while others require detailed representations that integrate complex features such as time-slotted channel hopping (TSCH), multi-protocol coexistence, or nonlinear battery discharge effects.

An equally important consideration in IoT simulation is the balance between precision and usability. While accurate models validated against real-world traces can offer scientific rigor, the complexity of setting up and configuring such scenarios often imposes steep learning curves. Many open-source simulators, including ns-3, are powerful but primarily text-based, requiring programming expertise and domain knowledge to be effectively used. The absence of intuitive graphical user interfaces or scenario editors further discourages adoption among novice researchers or practitioners.

Proprietary tools attempt to address usability by offering graphical interfaces, floorplan importers, and drag-and-drop scenario builders. However, their closed-source nature limits the ability to extend models or introduce new IoT standards,

which is a major drawback for researchers focused on emerging protocols such as Matter, Thread, or Zigbee. Furthermore, these tools rarely provide transparent energy models or the ability to account for the computational cost of cryptographic operations—two aspects increasingly critical for IoT research.

Thus, for IoT development, discrete event simulators must evolve along two parallel lines: improved fidelity through validated models of MAC protocols, energy, and security; and improved accessibility through GUIs, reproducible benchmarks, and integration with real hardware via emulation bridges. Research in this area can fill these gaps by producing open-source modules, calibration toolkits, and user-friendly interfaces that both expand functionality and lower the entry barrier for the IoT research community.

3.1. THE NS-3 NETWORK SIMULATOR

Ns-3 is a discrete event-driven simulator widely used for evaluating computer networks and emerging IoT systems. It offers modular models that enable protocol-level experimentation, making it an essential tool for both academic research and standards development. IoT-related networks are mainly represented in its LR-WPAN and 6LoWPAN modules, which allow simulation of low-power wireless communications. LR-WPAN supports the 2.4 GHz ISM band with O-QPSK modulation and implements IEEE 802.15.4 (2003–2011) MAC features such as association, scanning, and beaconed or beaconless operation. These features allow researchers to emulate typical IoT device initialization and operation within low-power networks.

However, several advanced behaviors introduced in the 2015 revision of the standard remain absent. For example, ns-3 currently lacks support for Guaranteed Time Slots (GTS), has only partial implementation of indirect transmissions, and omits deterministic scheduling mechanisms such as Time Slotted Channel Hopping (TSCH) and the Deterministic and Synchronous Multi-Channel Extension (DSME). Similarly, Received Signal Strength Indication (RSSI) is not provided as part of the PHY model, reducing fidelity in scenarios where link quality estimation is important. On the physical layer side, the simulator only supports O-QPSK at 2.4 GHz, whereas the standard defines a variety of modulations and sub-GHz bands that could improve the realism of IoT simulations.
Beyond LR-WPAN, the 6LoWPAN module enables IPv6 communication over constrained networks, offering packet compression, fragmentation, and mesh forwarding features. This makes it possible to investigate Low-Power and Lossy Networks (LLNs) with realistic constraints on header size and packet forwarding. While 6LoWPAN was designed primarily for LR-WPAN, ns-3 also allows its use with Wi-Fi and other link technologies, extending its utility across heterogeneous IoT networks. Despite this flexibility, official support for higher-layer IoT stacks such as Zigbee, Thread, or Matter is absent, and existing third-party contributions often suffer from limited maintenance.

A further challenge for ns-3 is usability. Although its source code is well-documented, the lack of graphical interfaces for building network scenarios poses a steep learning curve for beginners and limits adoption by practitioners. Graphical tools that allow scenario creation, floorplan import, and export of standardized configuration files would substantially improve accessibility. Likewise, ns-3's current inability to interface directly with IoT hardware restricts opportunities for hardware-in-the-loop experiments. Extending emulation capabilities through UART/SPI bridges to real IoT devices would allow validation of simulation results against physical testbeds.

Critical research opportunities include enhancing ns-3 with realistic energy consumption models, incorporating cryptographic overheads to reflect IoT security trade-offs, and developing calibration frameworks to align simulations with traces from real-world deployments. By addressing these gaps, future work can transition ns-3 from a perspective tool to a validated, reproducible testbed for both academia and industry, adding tangible value to IoT research.

IV. IoT PROTOCOLS: STANDARDS VS. MARKET ADOPTION

The number of protocols standardized for IoT communication is significant and continues to expand. These are broadly categorized as short-range networks such as wireless personal area networks (WPAN) and long-range technologies such as low power wide area networks (LPWAN). Examples include the well-established IEEE 802.15.4 for WPAN and LoRa/LoRaWAN for LPWAN. While these standards define interoperability, their market adoption is often fragmented, with legacy devices still implementing older revisions and proprietary extensions. Beyond these, Thread, Zigbee, and Matter are emerging to unify smart home ecosystems, whereas SigFox and NB-IoT extend LPWAN coverage. However, practical adoption faces challenges: incomplete or outdated simulator support, limited energy and security modeling, and a lack of validated benchmarks hinder real-world translation of standards into deployed solutions. In the following sections, we will consider these limitations as opportunities. Specifically, addressing validation frameworks, energy-aware models, and usability improvements in simulators can bridge the gap between standardized protocols and their effective market adoption.

4.1.*LoRA and LoRAWAN*

Long Range (LoRa) is a physical layer (PHY) technology developed by Semtech, based on Chirp Spread Spectrum (CSS) modulation. Operating in unlicensed sub-GHz frequency bands, primarily 868 MHz and 915 MHz, it is widely valued for its ability to achieve long-range communication with

low energy consumption, which makes it particularly suitable for large-scale deployments in rural and open-area environments. Despite these advantages, LoRa links remain highly sensitive to deployment density and propagation environments. In scenarios with high node concentration or the presence of obstacles such as buildings, attenuation, multipath fading, and interference effects can significantly reduce performance and reliability.

On top of the PHY layer, the LoRa Alliance defined the LoRaWAN protocol, which serves as the link (MAC) layer. LoRaWAN employs an ALOHA-based medium access method, which, while simple, limits scalability due to increased collision probabilities under heavy traffic conditions. Unlike other IoT stacks, LoRa/LoRaWAN currently lacks an official specification for mesh networking, though research efforts have explored mesh and hybrid extensions. Furthermore, LoRaWAN simulation support remains fragmented, with models available for OMNeT++ and ns-3 but typically developed externally and not actively maintained. Addressing this research gap, validated and officially supported simulation models, energy-aware extensions, and security-overhead representations are urgently needed to improve reproducibility, industry relevance, and real-world applicability.

4.2. *IEEE 802.15.4 std.*

The IEEE 802.15.4 standard defines the foundation for low-rate wireless personal area networks (LR-WPANs) and is regarded as the de facto standard for Internet of Things (IoT) applications, particularly in smart homes and industrial monitoring. The standard specifies two layers: the Physical (PHY) layer and the Medium Access Control (MAC) layer. While IEEE 802.15.4 offers flexibility in protocol stack integration, higher-layer stacks such as Zigbee, Thread, and 6LoWPAN typically build on these two layers. This modularity makes it a core enabler of IoT communication. To date, the IEEE 802.15.4 standard has undergone five major revisions (2003, 2006, 2011, 2015, 2020), each adding new frequency bands, modulation schemes, and MAC features.

Despite these revisions, market adoption lags behind standardization. Most commercially available devices continue to rely heavily on the 2006 revision, supporting the 2.4 GHz ISM band with 250 kbps O-QPSK modulation. Even devices considered state-of-the-art, such as the ESP32-H2 released in 2023, have only integrated features up to the 2015 revision. Our survey of hardware manufacturers shows that adoption of newer revisions is slow, often trailing by nearly a decade. This mismatch between standardization and implementation constrains innovation in IoT systems, especially for applications requiring advanced MAC modes, regional band diversity, or higher reliability. Table 1 summarizes key IEEE 802.15.4-compliant hardware and their supported revisions.

Similarly, simulation frameworks present a parallel limitation. Although network simulators such as ns-3, OMNeT++, and Cooja provide IEEE 802.15.4 models, they often support only partial subsets of the standard. For instance, ns-3's LR-WPAN module remains incomplete, missing critical features such as Guaranteed Time Slots (GTS), Time-Slotted Channel Hopping (TSCH), and DSME support. Simulations frequently fail to capture the breadth of PHY revisions, with most focusing exclusively on the 2.4 GHz band. Table 2 shows a comparison of commonly used simulators and the revisions they support.

Table 1. IEEE 802.15.4 Standard-Compliant Hardware and Supported Revisions

Device / Chipset	Release Year	Supported Revision	Supported Bands (MHz)	Notes on Adoption
Texas Instruments CC2530	2010	2006	2400	Widely used in Zigbee nodes
Microchip SAMR21	2015	2011	2400	Limited PHY features
NXP JN5169	2016	2011	2400, 868/915	Regional band support
Silicon Labs EFR32MG12	2018	2015	2400	Partial MAC support
Espressif ESP32-H2	2023	2015	2400	Latest hardware, lacks 2020 revision

Table 2. Network Simulators and Their Supported IEEE 802.15.4 Implementations

Simulator	Supported Revisions	PHY Bands Supported	MAC Features Included	Missing Features / Gaps
ns-3	2006 (partial), 2011	2400, 868, 915 (limited)	CSMA/CA, Indirect TX (partial)	GTS, TSCH, DSME
OMNeT++/INET	2006, 2011	2400	CSMA/CA	Limited PHY revisions
Cooja/Contiki	2006	2400	Beaconless MAC	No TSCH/DSME extensions
LoRaSim (3rd-party)	N/A (LoRa specific)	Sub-GHz bands	LoRa PHY only	No IEEE 802.15.4 MAC
QualNet	2006,	2	CSMA/C	Outdated

	2011	400	A, basic beacon mode	modules

From a research perspective, this incompleteness reveals important gaps. Hardware and simulation implementations do not fully align with the IEEE 802.15.4 revisions, which can mislead evaluations and limit reproducibility. For example, simulations may omit beacon-enabled MAC modes or newer PHY options, while actual hardware may exclude energy-saving mechanisms or security overhead models. Addressing these gaps offers clear opportunities for impactful research.

Future directions should focus on bridging this disparity through prototype implementations and validated models. Developing accurate energy consumption frameworks, integrating nonlinear battery models, and modeling the computational overhead of cryptographic security are critical steps. Similarly, emulator bridges between ns-3 and real IoT hardware could ensure closer fidelity and reproducibility. A further research extension lies in usability: creating graphical user interfaces (GUIs) to configure IEEE 802.15.4 scenarios with floorplan importers and scenario export features would make simulators more accessible and foster wider adoption.

Overall, while IEEE 802.15.4 continues to anchor IoT connectivity, both hardware and simulation implementations remain fragmented and incomplete. By addressing these shortcomings through empirical validation, trace-based calibration, and open-source prototype contributions, research can add significant value. Such contributions not only close the gap between standards and practice but also provide the industry and academia with robust, reproducible tools for future IoT system design and evaluation.

4.3 THE THREAD STANDARD

Thread is an open standard designed for **low-power and low-data rate devices**, providing **IPv6-based secure connectivity** for IEEE 802.15.4 devices. Unlike simpler link-layer implementations, Thread represents a **protocol stack** composed of multiple communication layers that collectively extend the native capabilities of IEEE 802.15.4 radios. While IEEE 802.15.4 was initially conceived for mesh networking, the devices themselves cannot independently establish multi-hop networks. Instead, higher-layer protocols, such as Thread, are required to enable reliable mesh capabilities. Within IEEE 802.15.4, all devices are organized in a **Personal Area Network (PAN)** through a coordinator–end device relationship, where coordinators manage address allocation and access control. However, in Thread, these **MAC-layer management responsibilities are shifted upwards**, with Thread itself providing address assignment, network joining, and security services.

The Thread protocol stack introduces several **non-standard or modified components** when compared to IETF-defined solutions. A notable example is its adaptation of the **Mesh Link Establishment (MLE)** protocol, which is customized to support functions such as asymmetric link management, neighbor discovery, and network administration. Similarly, Thread employs **6LoWPAN header compression**, but replaces the typical 6LoWPAN-ND-based addressing scheme with **DHCPv6 for IPv6 address allocation**. Its routing solution is a lightweight, custom-built protocol loosely based on **RIP (Routing Information Protocol)**, optimized for constrained low-power devices. These deviations highlight Thread's pragmatic focus on efficiency and simplicity at the expense of full alignment with IETF standards, a design choice that has both advantages and drawbacks for long-term interoperability.

One of the most widely used implementations of Thread is **Google's OpenThread**, which supports both older IEEE 802.15.4-2006 devices and newer radios such as the nRF528xx, ESP32-H2, and JN5189. OpenThread supports flexible deployment models. In a **System-on-Chip (SoC)** design, the complete protocol stack and user applications coexist within the same integrated circuit, enabling highly compact IoT devices. In contrast, **co-processor architectures** offload certain functions: (1) the **Radio Co-Processor (RCP)** design, where the host processor runs the Thread stack and applications, communicating with the 802.15.4 radio via UART or SPI; and (2) the **Network Co-Processor (NCP)** design, where the Thread stack resides on the co-processor, while the host processor focuses solely on applications. This modularity offers flexibility in device architecture but also complicates simulation and testing.

In the context of **simulation**, two pathways have been proposed for supporting Thread within the ns-3 network simulator. The first is a **tight integration of OpenThread with ns-3**, using ns-3 to model the IEEE 802.15.4 PHY and MAC layers while deferring protocol logic to OpenThread. This approach is straightforward but limits researchers' ability to modify or extend individual protocols, such as replacing DHCPv6 with 6LoWPAN-ND or enhancing the custom routing protocol. The second pathway involves **directly implementing Thread's higher-layer protocols in ns-3** (e.g., CoAP, MLE, and RIP-like routing). While this approach is far more complex, it offers flexibility and fine-grained control, enabling validation of protocol modifications and extensions that are otherwise constrained in the integrated approach.

However, research gaps remain. Current ns-3 support for Thread is limited, with **energy consumption models, security overhead representations, and trace-based calibration frameworks** underdeveloped. Moreover, there is no standardized benchmark suite for comparing simulation fidelity with hardware-based deployments. Addressing these challenges provides clear research opportunities: for example, developing **nonlinear energy models** for Thread devices,

designing **trace-calibrated propagation tools**, or implementing **Thread's TSCH scheduling extensions** within ns-3. Such contributions would not only bridge simulator limitations but also add reproducible, validated artefacts to the community, thus moving beyond the current perspective-driven discourse.

In conclusion, the Thread standard represents a significant step in enabling **secure, low-power IPv6 mesh networking** for constrained devices, balancing efficiency with pragmatic protocol modifications. Yet, its complexity and partial misalignment with IETF standards pose challenges for **simulation, validation, and interoperability studies**. By focusing on implementing and empirically validating underexplored aspects of Thread within simulators such as ns-3, researchers can contribute both **technical advancements and reproducible tools**, ensuring that Thread remains a robust and practical standard for the rapidly evolving IoT ecosystem.

4.4. ZIGBEE

The CSA's (Connectivity Standards Alliance) Zigbee protocol stack has been specifically designed to operate on top of IEEE 802.15.4. It builds upon and complements the limitations of the underlying PHY and MAC, particularly in areas such as network bootstrap, security, and multi-hop routing. Unlike Thread, which is strictly IP-based, Zigbee leverages both 16-bit short addresses and 64-bit extended addresses already specified in IEEE 802.15.4. Zigbee's longevity—over two decades of iterations—constitutes its strongest advantage, making it one of the most established and widely deployed IoT full stacks in the global market. The stack can be roughly divided into three parts: the network layer (NWK), the application support sublayer (APS), and the application layer (APL) (see Fig. 1).

The **NWK layer** extends the association and joining features of the MAC by enabling network bootstrap and providing tree as well as mesh routing capabilities. Zigbee's mesh routing borrows its foundation from the Ad hoc On-Demand Distance Vector Routing (AODV) protocol but introduces significant modifications, including robust neighbor discovery and dynamic link quality management. This makes Zigbee particularly resilient in dense IoT environments where topologies may shift frequently due to device mobility or energy constraints.

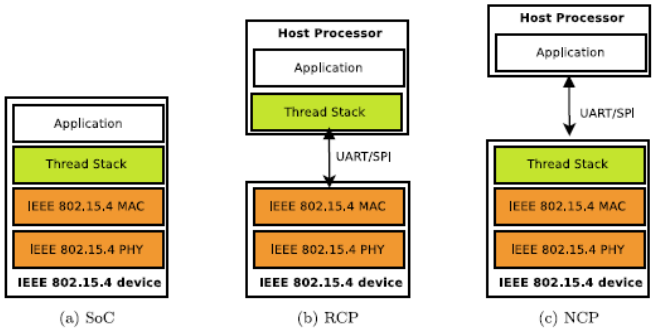

Fig. 1. OpenThread Designs.

The **APS layer** manages bindings between applications, oversees message routing among devices, and handles tasks such as fragmentation and reassembly. Crucially, the APS also provides the address mapping functions required to transition between Zigbee's extended and short addressing schemes. This layer is particularly important in large-scale IoT networks, where thousands of nodes may interact under constrained resources.

The **APL layer** defines the runtime environment for applications, organized through profiles, clusters, and attributes. This structure inspired the development of the Matter application layer, also under CSA's governance. Despite their conceptual overlap, Zigbee APL and Matter are not interoperable because they are designed to operate over different protocol stacks. Nevertheless, the historical continuity between them underscores the relevance of Zigbee in shaping modern IoT standards.

From an implementation standpoint, **ZBOSS** remains the only licensed full open-source Zigbee stack. Its initial public version (1.0) is freely available, while more advanced versions require membership in the ZBOSS Public Initiative, limiting accessibility for researchers. Commercial variants are widely adopted by vendors such as Nordic Semiconductor and Espressif. Alternative solutions like **Zigpy** exist, but their reliance on an RCP design and Python back-end reduces their suitability for high-fidelity simulation and large-scale evaluation.

Support for Zigbee in **discrete-event network simulators** is extremely limited. Previous attempts included Riverbed's Opnet and a discontinued implementation in ns-2 (based on the 2006 specification). At present, there is no official Zigbee module in ns-3, and this represents a significant research gap. Furthermore, even existing prototypes do not provide validated models for **energy consumption, security overhead, or trace-based calibration against real hardware**, which are essential for reproducibility. Addressing these gaps would move research beyond theoretical perspectives towards **prototype-driven validation**, enabling reliable comparisons between Zigbee, Thread, and Matter. For instance, an ns-3 Zigbee NWK module extended with a **nonlinear battery model** and a **cryptographic overhead representation** would provide realistic insights into performance trade-offs in constrained environments. Similarly, coupling Zigbee simulations with **emulation shims** to real hardware (e.g., via

UART/SPI) could foster hybrid testbeds that are more reflective of industrial use cases.

In conclusion, while Zigbee remains one of the most proven IoT stacks, its underrepresentation in current simulation ecosystems limits reproducibility and comparative research. Future work should prioritize validated implementations of Zigbee in ns-3, coupled with trace-driven calibration, energy and security modeling, and usability improvements such as scenario GUIs. Such contributions would directly address the pressing challenges outlined by the community and add tangible value to both academia and industry.

4.5. THE MATTER STANDARD

Matter [x] is a recent application layer standard launched by the Connectivity Standards Alliance (CSA), the same consortium responsible for Zigbee. Its objective is to provide a **common API (application programming interface)** that allows developers to create applications for a wide variety of communication hardware technologies such as IEEE 802.15.4-based Low-Rate Wireless Personal Area Networks (LR-WPAN), Wi-Fi, and Bluetooth Low Energy. Before the introduction of Matter, most vendors relied on proprietary APIs at the application layer, which led to **fragmentation and incompatibility** between devices. This situation forced IoT developers to maintain multiple versions of the same application, increasing development costs and user frustration. With Matter, devices "speak the same language," offering greater interoperability and reducing time-to-market for IoT solutions without substantial vendor-specific adjustments (Fig. 1).

Despite this vision, Matter faces **serious backward compatibility challenges**. Millions of IoT devices are already deployed worldwide using legacy protocol stacks such as Zigbee or Z-Wave. Since Matter depends on Thread (an IPv6-based mesh networking protocol running on top of IEEE 802.15.4) for low-power devices, Zigbee-based networks cannot directly interoperate with Matter systems. The CSA has suggested the use of "bridge" devices that can communicate in both Thread and Zigbee domains, but these guidelines are still vague and require additional hardware. As a result, **the longevity of existing Zigbee deployments is uncertain**, even though Zigbee 3.0 revisions such as R23 continue to support millions of consumer products. Manufacturers could, in theory, push firmware updates to migrate devices to Thread stacks, but due to **limited processing power, memory constraints, and low economic incentives**, this solution is rarely feasible. In many cases, the only practical alternative is to rely on a network co-processor (NCP) to provide interoperability between legacy and Matter-compatible systems (Fig. 2).

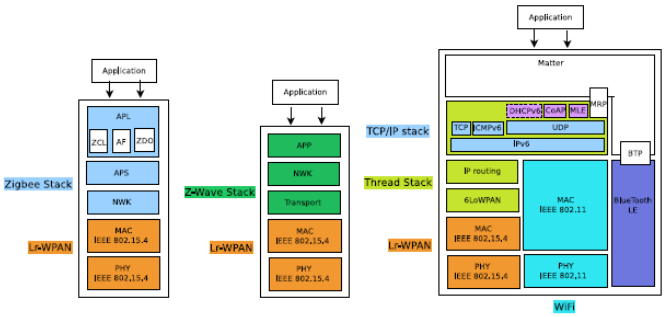

Fig. 2. IoT protocol stacks comparison.

For **home automation use cases**, Matter holds clear advantages. It simplifies application portability, allows for standardized commissioning of devices, and has strong backing from large technology companies such as Apple, Google, Amazon, and Samsung. However, the **stack overhead** of Matter is nontrivial: a device must run Matter alongside Thread and TCP/IP. For small, resource-constrained IoT nodes with only a few kilobytes of memory and strict energy budgets, this overhead is prohibitive. After all, the original intent of IEEE 802.15.4 was to enable low-cost, low-data-rate, low-power devices, where the entire protocol stack is typically embedded in a single chip. In such cases, Matter's design philosophy risks excluding precisely the class of constrained devices that defined the early IoT landscape.

Another critical limitation is related to **privacy and security**. While Matter builds on Thread's strong security features, these protections introduce further computational and memory costs. Users in sensitive domains may also prefer solutions independent from IP networks, which can provide enhanced privacy. Thus, Matter may be less suitable for non-IP-dependent IoT ecosystems, industrial networks, or highly constrained sensor nodes where security must be balanced against efficiency.

In light of these issues, **research opportunities** remain open. One important direction is to explore **lightweight implementations** of Matter for constrained devices, perhaps through modular stacks that allow partial adoption without the full TCP/IP overhead. Another extension involves developing **trace-based validation frameworks** to quantify the actual impact of Matter's overhead on memory, energy consumption, and latency across diverse IoT hardware. Research can also examine **bridge design architectures** that provide robust interoperability between Zigbee and Matter networks, moving beyond CSA's broad guidelines to tested prototypes. Additionally, future studies should investigate **security-cost models**—capturing the trade-offs between cryptographic overhead, battery life, and application performance in Matter-enabled devices (Fig. 3).

To summarize, Matter represents a **significant step toward unifying the IoT ecosystem**, particularly in consumer smart home markets. Its standardized API improves device compatibility and developer efficiency, but it **does not solve**

the backward compatibility problem for the vast installed base of legacy devices. Furthermore, its reliance on Thread and IP networking may limit its adoption in highly constrained or privacy-sensitive applications. By addressing energy models, security overheads, and interoperability prototypes, researchers can add critical value and extend Matter's applicability beyond its current consumer-focused scope.

V. NETWORK SIMULATIONS CHALLENGES

IoT network simulation faces challenges arising from fragmented standards and diverse protocol stacks such as Zigbee, LoRaWAN, Thread, and Matter. Simulators like ns-3 struggle to keep pace with evolving revisions, leaving gaps in MAC/PHY fidelity, stack integration, and usability. The disparity between academic needs for accuracy, industrial demand for scalability, and standardization requirements strains simulator design.

Further challenges include weak support for energy and security models, lack of empirical validation frameworks, and limited emulation with real hardware. Reproducibility, maintenance, and usability remain underexplored. Addressing these issues through validated prototypes, benchmark suites, and accessible GUI tools can bridge critical research gaps.

5.1. PHYSICAL AND MAC LAYERS (L1 AND L2)

The physical (PHY) layer represents the lowest layer of IoT communication stacks and plays a critical role in determining the realism of network simulations. Link-layer simulators can capture PHY behavior at a very fine granularity, including encoding, decoding, and waveform effects, but they are computationally expensive and therefore impractical for large-scale network-level studies. Discrete event simulators, such as ns-3, instead approximate the PHY by employing statistical propagation and interference models. While these models are efficient and extensible, they fall short of reproducing the full range of wireless channel dynamics experienced in real deployments, such as non-linear fading, hardware imperfections, and cross-technology interference. This gap highlights the need for validated propagation and energy models that are trace-calibrated against empirical measurements, enabling more credible simulations without incurring prohibitive computational cost.

The medium access control (MAC) layer is simulated with greater precision since it governs critical mechanisms such as channel access, scheduling, and collision avoidance. However, its implementation in commercial IoT hardware is typically closed and fixed, limiting opportunities for modification or experimentation in practice. Simulators therefore provide unique value by enabling researchers and standards bodies to prototype novel MAC techniques, such as time-synchronized channel hopping (TSCH) or improved contention algorithms. Yet, many simulators remain incomplete in their MAC implementations, omitting features like GTS, DSME, or hardware-specific timing constraints. Moreover, the lack of energy-aware and security-overhead models further restricts the ability of current tools to evaluate trade-offs between performance, reliability, and resource consumption. Addressing these limitations would strengthen simulation fidelity and bridge the gap between academic proposals and real-world device behavior.

A promising direction for future research is to design PHY and MAC simulation models that balance abstraction with validation. Incorporating non-linear battery discharge models tied to transceiver states, along with lightweight representations of cryptographic overhead, can make simulations more realistic for constrained IoT nodes. In parallel, trace-based calibration toolkits and hardware-in-the-loop emulation bridges can provide the necessary ground truth to refine PHY and MAC models. Finally, usability improvements such as GUI-based scenario builders and standardized benchmarks would ensure that these enhanced models are more accessible and reproducible. By implementing and validating these enhancements, new research can move beyond perspective-based discussions and deliver tangible artefacts that advance both simulator fidelity and practical IoT system design.

5.2. NETWORK AND TRANSPORT LAYERS (L3 TO L6)

As shown in Section 3, Fig. 2, protocol stacks can be quite different from each other. As a consequence, there are two different sets of needs.

5.2.1. SINGLE-CHOICE STACKS

We define *single-choice* stacks as those where protocol design choices are tightly bound to specifications, leaving little scope for modification. Examples include LoRaWAN, Zigbee, Matter, and Z-Wave, where compliance with mandated APIs ensures interoperability but restricts protocol-level experimentation. While such rigidity is essential for industrial adoption, it also limits opportunities to explore enhancements in areas like energy modeling, emulation fidelity, or security overheads. For instance, researchers cannot alter the underlying MAC or PHY without deviating from the standard. Nevertheless, introducing controlled flexibility—through validated extensions, benchmark-driven calibration, or emulation bridges with hardware—can serve both research and standardization. This balance enables deeper investigations into performance, energy efficiency, and security tradeoffs while preserving conformance with existing specifications.

5.2.2. MULTI-CHOICE STACKS

Multi-choice stacks provide researchers and developers with the flexibility to configure and experiment with different components of the IPv6-based IoT protocol stack. Unlike single-choice stacks, which are tightly bound to a predefined specification, multi-choice stacks enable custom selection of routing algorithms, transport protocols, and application layers. This flexibility is especially valuable in research environments where the impact of protocol alternatives—such as RPL variations, lightweight transport mechanisms, or security overhead—needs to be critically assessed. By supporting these customizable combinations, simulators allow researchers to evaluate tradeoffs between performance, energy efficiency, and scalability in diverse IoT deployments.

However, existing simulators often lack fully validated implementations, making reproducibility and fidelity difficult. Addressing these gaps requires calibrated models, energy-aware extensions, and integration of security costs. Building practical tools, such as GUIs for scenario creation or trace-to-simulation calibration frameworks, ensures that multi-choice stacks not only remain flexible but also deliver reliable, user-friendly experimentation environments.

5.3. APPLICATION LAYER (L7)

The application layer plays a critical role in IoT simulations, as it defines how devices and users interact through higher-level protocols. Unlike lower layers, modeling L7 is complex because it must capture human-driven behaviors such as web client requests or smart home command patterns, which are often unpredictable and heterogeneous. To reduce this complexity, standardized models derived from organizations like IETF or OCF are frequently employed. These models abstract user interactions into reproducible patterns (e.g., HTTP request–response sizes, MQTT publish–subscribe flows, or CoAP transactions), allowing simulators to mimic real traffic with reasonable fidelity.

Building on identified research gaps, future work should focus on validated, trace-driven application models that integrate security and energy costs, while ensuring usability through no-code scenario configuration tools. For example, coupling standardized traffic generation with calibration against real-world IoT datasets could provide both reproducibility and realism. By addressing these issues, simulation platforms can evolve beyond perspective papers into practical, validated tools that better support industry, academia, and standardization.

VI. DEVELOPMENT PRIORITIES AND OPPORTUNITIES

Future IoT-oriented simulation systems must evolve beyond descriptive perspectives and incorporate validated implementations that bridge simulation and real-world performance. Priorities include accurate modeling of emerging standards such as IEEE 802.15.4e TSCH, LoRaWAN, and Matter, alongside the integration of nonlinear energy and security overhead models. Strengthening emulator bridges with real hardware and ensuring empirical calibration using trace-driven approaches are equally essential for improving fidelity and reproducibility.

Equally important are usability and sustainability. Developing GUI-based scenario builders with floorplan importers and standardized benchmark suites will lower entry barriers, while community-driven maintenance ensures long-term reliability. Together, these priorities create actionable research opportunities that transform limitations into practical, validated tools for academia, industry, and standardization.

6.1. ACCESSIBILITY AND USER INTERFACES

The usability of network simulation systems has long been considered a bottleneck for their broader adoption in both academia and industry. While these simulators offer powerful mechanisms to evaluate protocol performance, test “what-if” scenarios, or support standardization efforts, they often remain accessible only to users with strong programming backgrounds. Most simulators, such as ns-3, are heavily text-driven and require familiarity with scripting languages, module configuration, and networking standards. This steep learning curve discourages newcomers, especially students or practitioners from non-computer science domains who wish to use simulators as educational tools. Accessibility, therefore, emerges as one of the most pressing gaps in current IoT-oriented simulation systems, where user interaction should be treated as a first-class feature rather than an afterthought.

Graphical User Interfaces (GUIs) are central to improving accessibility. A well-designed GUI can significantly reduce the barrier to entry by enabling users to set up simulations through intuitive workflows, thereby eliminating repetitive and error-prone coding tasks. In the context of IoT simulation, GUIs could allow users to define topologies, select protocol stacks, and visualize outcomes without writing complex scripts. Such features would not only minimize menial tasks like configuring protocol parameters or node positions but would also complement text-based approaches rather than replacing them. The analogy to Unreal Engine is apt: in that ecosystem, GUIs (blueprints) work alongside traditional programming to allow for complex environment creation. A similar dual-mode paradigm could transform IoT network simulators into tools that balance accessibility and depth.

Beyond reducing setup complexity, GUIs can enhance the realism of simulated scenarios by supporting advanced features such as floorplan importation. IoT deployments frequently occur in indoor environments where walls, furniture, and other obstacles affect propagation and topology dynamics. By enabling users to import building layouts, position nodes, and automatically infer connectivity across multiple protocol layers, GUIs could provide richer and more accurate representations of real-world deployments. The ability to

export these setups into portable formats such as SVG, JSON, or XML would further promote reproducibility and allow researchers to share and iterate on scenarios. Although ns-3 offers ConfigStore for storing parameters, it remains underutilized and could be integrated into a unified GUI to support seamless experiment replication.

However, designing such interfaces is non-trivial. A critical requirement is a descriptive intermediate language that acts as a bridge between graphical interfaces and the simulator's internal models. OMNeT++'s NED language provides one example, but its complexity illustrates the tension between completeness and usability. If too detailed, description files risk becoming as daunting as raw code; if too simplistic, they may lack the expressiveness required for meaningful experiments. Thus, the challenge lies in striking the right balance: GUIs should enable effortless repetition of common tasks such as Monte Carlo simulations, while still supporting more advanced users who require detailed configurations. This balance is essential to attract both entry-level users and seasoned researchers.

Prior efforts to add GUIs to ns-3 highlight the difficulty of sustaining such tools. Several proposals have emerged in the past, but most failed to gain long-term traction due to scope creep, maintenance issues, or incompatibility with evolving simulator architectures. By contrast, official ns-3 visualizers like NetAnim and NetSimulyzer have achieved moderate success, but they remain limited to post-simulation visualization rather than full scenario creation. For future proposals to succeed, maintenance must be a guiding principle. Lightweight, modular designs, open governance, and community support mechanisms could help ensure that GUI tools do not suffer the fate of abandonment.

From a research perspective, accessibility extends beyond usability to encompass validation and reproducibility. One of the main gaps in current simulators is the lack of trace-based calibration frameworks. GUIs could provide workflows for importing empirical traces from testbeds, fitting propagation or energy models automatically, and embedding those into simulations. Similarly, integration with hardware co-processors via emulation bridges could be made more approachable through GUI wizards, lowering the barrier for researchers who wish to combine simulation with real devices. These extensions would not only make tools easier to use but also ensure that simulations remain grounded in real-world behavior, thereby enhancing credibility.

Another underexplored area is the representation of cross-cutting concerns such as energy and security. While the literature identifies these as critical for IoT, current simulators lack accessible ways to configure or visualize them. A GUI capable of presenting, for example, per-node energy consumption curves or the computational cost of cryptographic operations would provide researchers with actionable insights. By abstracting away the complexity of integrating nonlinear battery models or CPU overheads, such interfaces could democratize access to advanced analyses that are currently restricted to experts. This aligns with the broader goal of making simulators both user-friendly and scientifically rigorous.

Finally, the value of enhanced accessibility lies in its potential to expand the audience and impact of IoT network simulators. By reducing learning curves, offering richer scenario creation, and enabling reproducibility, GUIs would make these tools useful not just for protocol designers but also for educators, standards bodies, and industry practitioners. Where the existing literature provides perspectives, future research can deliver prototypes, benchmarks, and validated evidence that transform these perspectives into practice. Developing a GUI that integrates descriptive languages, calibration workflows, and cross-cutting models would thus not only address a clear research gap but also provide lasting value by bridging the divide between theoretical simulations and real-world IoT experimentation.

6.2. EMULATION AND REPRODUCIBILITY

A central challenge in advancing IoT network simulation systems lies in bridging the gap between simulation models and real-world hardware implementations. As IoT protocols diversify and hardware vendors introduce proprietary variations, ensuring that simulations capture realistic device behavior becomes increasingly difficult. Simulation tools such as ns-3 and OMNeT++ have made proposals to extend emulation capabilities, enabling crossovers where simulator stacks interact directly with hardware stacks. Such capabilities are critical because reproducibility in simulation is only meaningful when results can be validated against actual device performance. Without this crossover, simulation studies risk producing results that are internally consistent but fail to reflect the complexities of real deployments.

IoT-specific characteristics make emulation especially challenging. Unlike traditional networking protocols, which are often well-standardized and implemented uniformly, IoT devices are heterogeneous, with deviations from specifications common across vendors. Configuration values described in standards may be unsuitable in practice or replaced by vendor-specific optimizations, creating inconsistencies that hinder interoperability. Moreover, while standards are publicly available, most hardware protocol stacks are closed-source, monolithic, and non-interchangeable across devices. This restricts the ability of simulators to replicate exact hardware behaviors, leading to mismatches in timing, energy consumption, or security overheads. Overcoming these obstacles requires simulation platforms to offer mechanisms that can "speak the same language" as real devices while remaining adaptable to deviations in proprietary stacks.

In the case of ns-3, emulation is traditionally enabled through the FdNetDevice abstraction, which allows the

simulator to exchange packets via Linux file descriptors with real host devices. However, this approach is limited in IoT contexts, since many IoT protocol stacks are executed directly on microcontrollers rather than integrated into the Linux kernel. For such scenarios, a more appropriate method involves bridging simulator and hardware through UART or SPI protocols, effectively creating a shim layer between the host computer and the MCU-based device. This technique has been exemplified in Thread protocol implementations and, although not yet available in the current ns-3 release (v3.42), it represents a viable path for future expansion. As suggested in Fig. 3 of the reference, such shim-based crossovers would allow simulators to forward traffic to actual devices, enabling validation of simulated scenarios with real-world device responses.

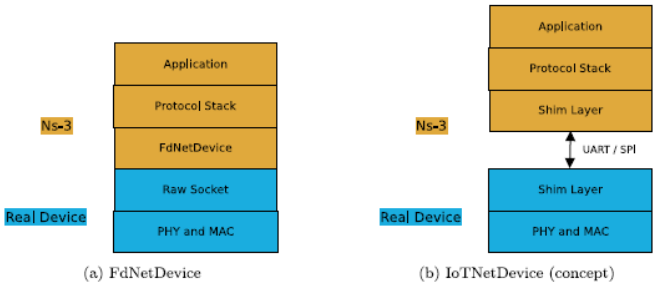

Fig. 3. Ns-3 emulation capabilities examples.

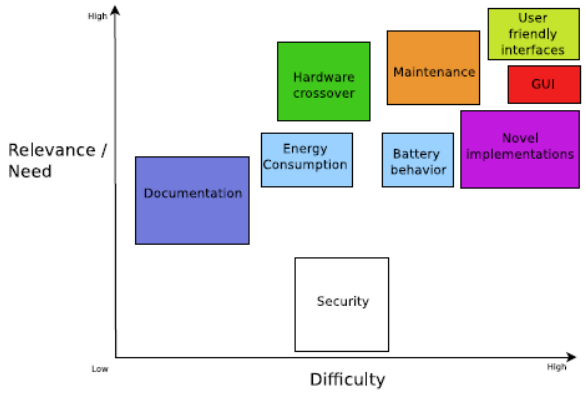

Fig. 4. Ns-3 IoT simulation system priorities and needs.

Despite its promise, emulation remains underdeveloped in current IoT-oriented simulation platforms, pointing to a significant research gap. Existing proposals lack standardized benchmarks for validating emulated interactions, making it difficult to quantify fidelity across devices. Additionally, simulation platforms have not yet integrated advanced energy models, nonlinear battery discharge patterns, or CPU load representations that reflect the cost of cryptographic operations. Without these extensions, reproducibility remains incomplete, as simulated outcomes may diverge significantly from hardware behavior under realistic energy and security constraints. Usability also remains a barrier: researchers often lack tools to easily configure and reproduce mixed simulation–hardware experiments. Addressing these issues requires not only new technical features but also the establishment of community-driven validation frameworks and regression suites.

Future research can provide value by turning perspective-based recommendations into implemented, validated solutions. One promising direction involves developing a standardized ns-3 ↔ MCU shim layer that supports UART/SPI connectivity, combined with open benchmark datasets to calibrate simulations against real deployments. Such a contribution would improve reproducibility and enable cross-tool validation, benefiting academia, industry, and standardization efforts alike. Similarly, extending simulators with realistic energy and security overhead models would allow researchers to explore tradeoffs between performance, efficiency, and robustness in IoT designs. Beyond technical accuracy, usability improvements such as graphical interfaces with floorplan imports and JSON-based scenario descriptors would further democratize access to reproducible research. By addressing these gaps, future work can bridge the divide between simulator abstraction and hardware reality, moving IoT network simulation systems toward tools that are not only predictive but also experimentally verifiable.

6.3. MAINTENANCE

One of the most pressing challenges in IoT-oriented network simulators is the issue of maintenance. As highlighted in prior work, many simulation modules become “abandonware,” with limited updates, poor documentation, and little alignment with evolving protocol specifications. This stagnation leaves researchers relying on outdated or incomplete implementations, which undermines reproducibility and the reliability of findings. Given the rapid pace at which new standards such as Matter, LoRaWAN extensions, and 802.15.4e emerge, sustained maintenance is essential to ensure simulators remain representative of real-world IoT environments.

Addressing this gap requires not only continuous updates to protocol models but also the creation of community-driven maintenance frameworks. Establishing regression test suites, trace-to-simulation calibration toolkits, and versioned benchmarks can encourage long-term usability and provide a shared foundation for validating updates. Equally important is usability — tools like GUI-based scenario designers and exportable JSON descriptors reduce the learning curve, motivating broader adoption and, by extension, stronger community support. When combined with modular code design, these practices help ensure that simulation systems evolve rather than stagnate.

From a research perspective, moving beyond perspectives toward validated prototypes directly adds value. By developing energy models, emulation bridges, or security overhead frameworks alongside open documentation, researchers contribute reusable artefacts that both extend

functionality and promote sustainability. Such contributions transform one-off implementations into long-lived, community-maintained assets, closing the gap between simulation research and industrial or standardization needs.

6.4. ENERGY MODULES DEVELOPMENT

The development of robust **energy modules** is pivotal for advancing IoT-oriented network simulation systems, as energy efficiency directly influences the longevity and viability of battery-powered devices. Current simulators, such as ns-3, offer only limited and often linear energy models, failing to capture the intricate dynamics of transceiver state transitions and the non-linear discharge patterns of real batteries. Addressing this research gap requires designing extensible modules that integrate fine-grained radio state machines with validated nonlinear battery models, enabling accurate estimation of device lifetimes under varying protocol and workload conditions. Furthermore, incorporating security and cryptographic overheads, frequently neglected in existing frameworks, is critical to realistically assessing energy costs in secure IoT communications. Beyond modeling, calibration with hardware testbed measurements and trace-driven validation would ensure fidelity and reproducibility, while delivering open artefacts, benchmark traces, and configuration scripts would enhance community adoption and long-term maintenance. By bridging conceptual recommendations with validated implementations, such energy modules would empower researchers and industry stakeholders to design, evaluate, and optimize protocols with tangible, evidence-based energy insights.

6.5. SECURITY

Security remains one of the most underrepresented aspects in IoT simulations, despite its prominence in real hardware implementations where encryption and authentication can consume nearly half of a device's specification. Most simulators exclude cryptographic processes due to computational cost, overlooking the critical trade-offs they impose on latency, throughput, and energy. A realistic IoT simulation framework must move beyond omission and represent the *overhead* of security mechanisms, thereby enabling accurate evaluation of secure-by-design protocols.

To achieve this, future research should not simulate full cryptographic suites, but model their measurable effects. Incorporating simplified yet calibrated overhead models will bridge the current gap between theoretical simulation and real-world IoT behaviour.

- **Processing delay:** extra time spent encoding/decoding packets.
- **Packet overhead:** increased frame size due to enabled security fields.
- **Energy cost:** additional CPU and transceiver consumption.
- **Performance trade-off:** quantified impact on reliability and scalability.

VII. CONCLUSION

This study examined the challenges surrounding IoT networking and emphasized the vital role of simulation in supporting academia, industry, and standardization bodies in shaping next-generation protocols tailored for IoT devices. While standardization efforts remain fragmented and influenced by market pressures, the need for robust and flexible simulation platforms is evident. A simulator that can provide accurate protocol modeling, maintain usability through intuitive GUIs, and enable reproducibility through standardized benchmarks is essential for guiding informed decisions and spotting optimization opportunities. Among the promising directions are improved support for protocols like IEEE 802.15.4e TSCH, LoRaWAN, and Thread, alongside realistic modeling of energy consumption, security overheads, and hardware–software interactions.

Building on this foundation, future work must go beyond identifying priorities and instead implement and validate these capabilities with empirical evidence. Contributions such as trace-calibrated simulation models, nonlinear energy and security-aware frameworks, and emulator bridges that integrate real devices will not only strengthen fidelity but also accelerate adoption across research and industry. By addressing these research gaps with open, validated, and reusable tools, the academic community can add measurable value, transforming IoT simulators into trusted instruments for both innovation and standardization.

REFERENCES

[1] A.G. Ramonet, T. Noguchi, IEEE 802.15. 4 now and then: Evolution of theLR-WPAN standard, in: 2020 22nd International Conference on AdvancedCommunication Technology, ICACT, IEEE, 2020, pp. 1198–1210.

[2] J.R. Cotrim, J.H. Kleinschmidt, LoRaWAN mesh networks: A review and classificationof multihop communication, Sensors 20 (15) (2020) http://dx.doi.org/10.3390/s20154273

[3] M. Slabicki, G. Premsankar, M. Di Francesco, Adaptive configuration of loranetworks for dense IoT deployments, in: NOMS 2018 - 2018 IEEE/IFIP NetworkOperations and Management Symposium, 2018, pp. 1–9, http://dx.doi.org/10.1109/NOMS.2018.8406255.

[4] D. Magrin, M. Centenaro, L. Vangelista, Performance evaluation of LoRanetworks in a smart city scenario, in: 2017 IEEE International Conferenceon Communications, ICC, 2017, pp. 1–7, http://dx.doi.org/10.1109/ICC.2017.7996384.

[5] L. Vangelista, A. Cattapan, Extending the lora modulation to add parallelchannels and improve the LoRaWAN network performance, in: 2021 11th IFIPInternational Conference on New Technologies,

Mobility and Security, NTMS,2021, pp. 1–5, http://dx.doi.org/10.1109/NTMS49979.2021.9432659.

[6] NXP, IEEE 802.15.4 Stack user guide, 2024, https://www.nxp.com/products/wireless-connectivity/zigbee/ieee-802-15-4-for-jn516x-7x:IEEE802.15.4,(Accessed January. 17. 2024).

[7] Texas Instruments, TI MAC software stack, 2023, https://www.ti.com/tool/TIMAC, (Accessed December. 15. 2023).

[8] Nordic Semiconductor, Product Specifications, 2024, https://infocenter.nordicsemi.com/index.jsp, (Accessed January. 17. 2024).

[9] NXP, JN-518x DataSheet, 2024, https://www.nxp.com/products/wirelessconnectivity/thread/jn5189-88-t-high-performance-and-ultra-low-power-mcusfor-zigbee-and-thread-with-built-in-nfc-option:JN5189_88_T, (Accessed January.17. 2024).

[10] Expressif, ESP32-H2 Datasheet, 2023, https://www.espressif.com/sites/default/files/documentation/esp32-h2_datasheet_en.pdf, (Accessed December. 15. 2023).

[11] Silicon Labs, EFR32MG13 Mighty Gecko Multi-Protocol Wireless SoC Family DataSheet, 2023, https://www.silabs.com/documents/public/data-sheets/efr32mg13-datasheet.pdf, (Accessed December. 15. 2023).

[12] Nsnam, Ns-3 network simulator, 2023, https://www.nsnam.org/, (AccessedDecember. 15. 2023).

[13] Nsnam, Ns-2 network simulator, 2023, https://www.isi.edu/nsnam/ns/index.html, (Accessed December. 15. 2023).

[14] Keysight, QualNet Network Simulator, 9.3.0 Programmers Guide, Sensor NetworksLibrary, 2023, https://www.keysight.com, (Accessed December. 15.2023).

[15] T. Boulis, D. Pediaditakis, Castalia 3.2 User Manual, 2024, https://github.com/boulis/Castalia, (Accessed January. 17. 2024).

[16] Riverbed Opnet Modeler, Online documentation, 2023, https://www.riverbed.com, (accessed December. 15. 2023).

[17] A.G. Ramonet, T. Noguchi, IEEE 802.15. 4 now and then: Evolution of theLR-WPAN standard, in: 2020 22nd International Conference on AdvancedCommunication Technology, ICACT, IEEE, 2020, pp. 1198–1210.

[18] Connectivity Standards Alliance, Zigbee Specification R23, 2023, https://csaiot.org/all-solutions/zigbee/, (Accessed December. 15. 2023).

[19] Thread Group, Thread Network Fundamentals V3.1, 2023, https://www.threadgroup.org/, (Accessed December. 15. 2023).

[20] IETF, Mesh Link Establishment, 2024, https://datatracker.ietf.org/doc/html/draft-ietf-6lo-mesh-link-establishment-00, (Accessed January. 04. 2024).

[21] Alberto GallegosRamonet,Tommaso Pecorella,Benedetta Picano,Kazuhiko Kinoshita, “Perspectives on IoT-oriented network simulation systems”, Computer NetworksVolume 253, November 2024, 110749, doi.org/10.1016/j.comnet.2024.110749