DEPARTMENT OF
COMPUTER SCIENCE

JOÃO RICARDO BOGALHO BRILHA
BSc in Computer Science and Engineering

MICROBABEL

A MULTI-PROTOCOL APPROACH FOR RESILIENT AND DECENTRALIZED IOT NETWORKS

MASTER IN COMPUTER SCIENCE AND ENGINEERING
SPECIALIZATION IN DISTRIBUTED AND PARALLEL SYSTEMS

NOVA University Lisbon
Draft: *January 21, 2026*

DEPARTMENT OF
COMPUTER SCIENCE

MICROBABEL

A MULTI-PROTOCOL APPROACH FOR RESILIENT AND DECENTRALIZED IOT NETWORKS

JOÃO RICARDO BOGALHO BRILHA

BSc in Computer Science and Engineering

Adviser: João Leitão
Associate Professor, NOVA University Lisbon

MASTER IN COMPUTER SCIENCE AND ENGINEERING
SPECIALIZATION IN DISTRIBUTED AND PARALLEL SYSTEMS

NOVA University Lisbon
Draft: *January 21, 2026*

Abstract

Regardless of the language in which the dissertation is written, usually there are at least two abstracts: one abstract in the same language as the main text, and another abstract in some other language.

The abstracts' order varies with the school. If your school has specific regulations concerning the abstracts' order, the NOVAthesis LaTeX (novathesis) (LaTeX) template will respect them. Otherwise, the default rule in the novathesis template is to have in first place the abstract in *the same language as main text*, and then the abstract in *the other language*. For example, if the dissertation is written in Portuguese, the abstracts' order will be first Portuguese and then English, followed by the main text in Portuguese. If the dissertation is written in English, the abstracts' order will be first English and then Portuguese, followed by the main text in English. However, this order can be customized by adding one of the following to the file `5_packages.tex`.

```
\ntsetup{abstractorder={<LANG_1>,...,<LANG_N>}}
\ntsetup{abstractorder={<MAIN_LANG>={<LANG_1>,...,<LANG_N>}}}
```

For example, for a main document written in German with abstracts written in German, English and Italian (by this order) use:

```
\ntsetup{abstractorder={de={de,en,it}}}
```

Concerning its contents, the abstracts should not exceed one page and may answer the following questions (it is essential to adapt to the usual practices of your scientific area):

1. What is the problem?
2. Why is this problem interesting/challenging?
3. What is the proposed approach/solution/contribution?
4. What results (implications/consequences) from the solution?

Keywords: One keyword · Another keyword · Yet another keyword · One keyword more · The last keyword

Resumo

Independentemente da língua em que a dissertação está escrita, geralmente esta contém pelo menos dois resumos: um resumo na mesma língua do texto principal e outro resumo numa outra língua.

A ordem dos resumos varia de acordo com a escola. Se a sua escola tiver regulamentos específicos sobre a ordem dos resumos, o template (LaTeX) novathesis irá respeitá-los. Caso contrário, a regra padrão no template novathesis é ter em primeiro lugar o resumo *no mesmo idioma do texto principal* e depois o resumo *no outro idioma*. Por exemplo, se a dissertação for escrita em português, a ordem dos resumos será primeiro o português e depois o inglês, seguido do texto principal em português. Se a dissertação for escrita em inglês, a ordem dos resumos será primeiro em inglês e depois em português, seguida do texto principal em inglês. No entanto, esse pedido pode ser personalizado adicionando um dos seguintes ao arquivo `5_packages.tex`.

```
\abstractorder(<MAIN_LANG>):={<LANG_1>,...,<LANG_N>}
```

Por exemplo, para um documento escrito em Alemão com resumos em Alemão, Inglês e Italiano (por esta ordem), pode usar-se:

```
\ntsetup{abstractorder={de={de,en,it}}}
```

Relativamente ao seu conteúdo, os resumos não devem ultrapassar uma página e frequentemente tentam responder às seguintes questões (é imprescindível a adaptação às práticas habituais da sua área científica):

1. Qual é o problema?
2. Porque é que é um problema interessante/desafiante?
3. Qual é a proposta de abordagem/solução?
4. Quais são as consequências/resultados da solução proposta?

Palavras-chave: Primeira palavra-chave · Outra palavra-chave · Mais uma palavra-chave · A última palavra-chave

Contents

List of Figures

List of Tables

Listings

Glossary

hysteresis Use of "decision inertia" or asymmetric thresholds to prevent frequent switching in response to small or short-lived changes in system metrics *(pp. 12, 27)*

LoRa From "long range", proprietary radio communication technology *(pp. 3, 4, 10, 15, 16, 26–28)*

MQTT Lightweight publish/subscribe messaging protocol designed for constrained devices and low-bandwidth networks. Previously stood for "Message Queuing Telemetry Transport", but since 2013 it does not stand for anything in particular *(pp. 12, 18)*

RPi Raspberry Pi full-size boards *(p. 20)*

μ-Babel MicroBabel *(pp. 2–4, 6, 10, 13, 15, 18, 19, 21–27)*

Acronyms

ABE	Attribute-based encryption *(p. 23)*
AES	Advanced Encryption Standard *(pp. 11, 13, 22, 24)*
ARIMA	Autoregressive integrated moving average *(pp. 16, 17, 19)*
AS	Authentication Server *(pp. 20, 23)*
BLE	Bluetooth Low Energy *(pp. 3, 4, 7, 9–11, 15, 16, 26, 27)*
CA	Certificate authority *(pp. 19, 23)*
CoAP	Constrained Application Protocol *(pp. 12, 18)*
CR	Compression ratio *(p. 17)*
DNS	Domain Name System *(p. 1)*
DODAG	Destination-oriented directed acyclic graph *(pp. 8, 26)*
DTN	Delay-tolerant networking *(pp. 8, 10)*
EWMA	Exponentially weighted moving average *(p. 12)*
FHSS	Frequency-hopping spread spectrum *(pp. 11, 13)*
FIRE	Fast Integer REgression *(p. 17)*
GASE	Group Authentication Scheme at the Edge *(pp. 20, 23)*
GL	Group leader *(pp. 20, 24)*
HF	High frequency *(p. 10)*
HTTP	Hypertext Transfer Protocol *(pp. 8, 12, 18)*
HVAC	Heating, ventilation, and air conditioning *(p. 5)*
IoT	Internet of Things *(pp. 1–3, 5–8, 10, 16, 18–22, 25, 27)*

KDC	Key distribution center *(pp. 19, 22, 23)*
LoRaWAN	LoRa Wide Area Network *(pp. 6, 10, 12, 27)*
LPKM	Lightweight Polynomial-based Key Management *(pp. 21, 23)*
LPWAN	Low-Power Wide Area Network *(p. 10)*
MAC	Message Authentication Code *(p. 20)*
MDL	Minimum description length *(pp. 18, 19)*
novathesis	NOVAthesis LaTeX *(pp. i, ii)*
NTP	Network Time Protocol *(p. 15)*
NVIS	Near Vertical Incidence Skywave *(pp. 10–12, 27)*
P2P	Peer-to-peer *(pp. iii, 6–9, 11, 25–27)*
PDA	Personal digital assistant *(p. 22)*
PKI	Public key infrastructure *(p. 23)*
PLR	Packet loss ratio *(pp. 7, 26)*
PRNG	Pseudorandom number generator *(pp. 11, 13)*
PUF	Physically Unclonable Function *(pp. 21, 23)*
QoS	Quality of Service *(pp. 4, 6, 12)*
RF	Radio frequency *(p. 10)*
RGCS	Randomized gossip-consensus-based sync *(pp. 14, 15, 27)*
RLE	Run-length encoding *(pp. 17–19)*
RPL	Routing Protocol for Low-Power and Lossy Networks *(pp. 9, 10, 26)*
RSSI	Received signal strength indicator *(pp. 7, 11, 26)*
SDN	Software-defined networking *(pp. 12, 27)*
SSS	Shamir Secret Sharing *(p. 20)*
TCP	Transmission Control Protocol *(pp. 7, 25, 26)*
WSN	Wireless Sensor Network *(pp. 7, 10, 11, 14)*

1 Introduction

The proliferation of Internet of Things (IoT) devices has changed how we monitor and interact with physical spaces – from smart homes to industrial facilities – with the global IoT market reaching 18.5 billion connected devices in 2024, and projected to reach 39 billion by 2030[31], with industry analysts estimating that IoT technologies could unlock between $5.5 and $12.6 trillion in economic value by the same year[7].

However, current IoT systems remain fundamentally dependent on continuous Internet connectivity and centralized cloud infrastructures[43, 14]. The dominant architectural pattern across the industry involves edge devices collecting data and transmitting it to remote cloud servers for processing, storage, and other control logic. This approach delivers scalability and ease of management but creates a fundamental dependency on network availability.

This centralized model introduces several concerns beyond just connectivity: operational costs for cloud services create economic dependencies on third-party providers and often lead to vendor lock-in, data sovereignty issues arise when sensitive information must travel to and reside on external infrastructure, and system resilience becomes fundamentally tied to the availability of these remote services.

While cloud platforms can benefit IoT deployments through additional computational resources and storage capacity, systems that *depend* on constant cloud connectivity sacrifice local autonomy and introduce single points of failure. This cloud-centric model has enabled rapid IoT adoption, but introduces critical vulnerabilities across scenarios where continuous connectivity cannot be guaranteed.

Recent infrastructure failures illustrate the fragility of cloud-dependent architectures: the AWS US-EAST-1 outage in October 2025[34] disrupted services from banking to smart homes worldwide, demonstrating how a Domain Name System (DNS) configuration error in a *single region* can cascade into global disruptions[32].

Attempts have been made to address these challenges through incremental improvements, such as shifting focus to edge computing in order to reduce latency[14, 43, 28], redundant cloud regions for availability[22, 21], and hybrid architectures that combine local and remote processing[1, 9, 17].

However, these solutions remain fundamentally tied to the assumption of eventual connectivity, and often increase system complexity without eliminating the core dependency. This leads to shortcomings in key areas that demand more fundamental architectural reconsideration:

Limited suitability for hazardous environments

Remote or dangerous locations (industrial sites, disaster-prone areas) require systems that can operate reliably without constant human intervention or stable network infrastructure;

Lack of autonomous operation

Device deployment and operation en masse can be brittle, with little tolerance for individual node failures in the IoT infrastructure, which is naturally susceptible to network failures and intermittent connectivity;

Privacy and data sovereignty concerns

Cloud platforms and other third-parties are oftentimes an unavoidable middle layer between end devices and end users, raising questions about data processing and control.

fazer distinção maior entre iot e domotics? fiz no related work, devia mudar para aqui?

While these challenges affect both IoT and domotics systems alike – from industrial monitoring to residential automation – they become particularly acute in disaster response and emergency scenarios, where communication infrastructure fails precisely when needed most, rapid deployment with minimal configuration becomes essential, and autonomous operation transitions from desirable to indispensable.

During earthquakes, floods, and other emergencies, the need for real-time sensor data (structural integrity, air quality, evacuation routes) and bidirectional communication (threat alerts, user feedback) becomes critical, yet traditional infrastructure often fails first, eliminating both cloud connectivity and local network access points that deployed devices rely on.

o que achamos do rebrand MicroBabel -> μBabel?

A pragmatic issue in the context of smart homes is that without cloud connectivity, users cannot interact with their appliances, such that during a Wi-Fi outage smart lights become uncontrollable despite all hardware being physically present. While merely inconvenient domestically, this architectural dependency has critical implications in other contexts and **μ-Babel** addresses such scenarios across scales: from residential systems requiring local control, to building monitoring infrastructures that must operate during emergencies, to disaster response networks where autonomous operation becomes essential when traditional infrastructure fails.

While cloud platforms like Amazon Alexa, Google Home, and Microsoft Azure IoT Hub offer convenience for data processing and remote device control, their inherent dependence on continuous Internet connectivity introduces some critical limitations: increased latency from round-trip communication to distant servers[28, 25], reduced availability during network disruptions or cloud outages[40, 21], and poor fault-tolerance when infrastructure fails[2].

These characteristics make cloud-centric architectures fundamentally unsuitable for scenarios that require or prioritize local operation, autonomous behavior, or guaranteed responsiveness during emergencies.

The Babel Ecosystem [12] addresses these limitations by enabling devices to operate autonomously without cloud infrastructure, while still supporting cloud integration when connectivity is available and desired.

In this document we introduce μ-**Babel**, a lightweight framework targeting embedded platforms (ESP32, Raspberry Pi Pico) aimed at developing resilient, multi-protocol and decentralized IoT systems that can operate autonomously during infrastructure failures. μ-**Babel** will integrate with the broader Babel Ecosystem, which runs on hardware with greater resources such as full Raspberry Pi boards or computers, enabling a class-based architecture where resource-constrained edge devices can seamlessly interoperate with computational nodes for data aggregation, processing, and coordination.

Note on terminology: Throughout this work the term *multi-protocol* is used to refer to heterogeneity in communication technologies (Bluetooth Low Energy (BLE), LoRa, Wi-Fi, etc.) that have distinct link/physical layers, while *multi-channel* refers to frequency diversity within a single protocol (e.g., 2.4GHz Wi-Fi channels 1-13, LoRa frequency hopping). Multi-protocol operation provides technology diversity for resilience; multi-channel operation provides frequency diversity for anti-jamming and throughput.

Main Research Questions

We focus on three main research questions:

How can these systems maintain communication when traditional infrastructure fails? Traditional IoT deployments rely heavily on Wi-Fi access points, cellular towers, or other centralized infrastructure that often becomes unavailable during disasters, or is altogether unreliable in remote locations. μ-**Babel** addresses this by leveraging a multi-protocol communication approach, supporting a diverse protocol stack (BLE, LoRa, ESP-NOW, ZigBee) that can operate independently of infrastructure.

add glossary entries for esp-now, zigbee, etc?

By enabling adaptive protocol selection and peer-to-peer mesh formation, devices can establish alternative communication paths when primary channels fail.

How can device heterogeneity be leveraged to create and orchestrate these networks? IoT deployments naturally comprise devices with varying capabilities, from resource-constrained sensors, to gateway nodes with greater processing power and storage. Rather than treating this heterogeneity as a limitation, μ-**Babel** exploits it through capability-aware protocols that allow devices to negotiate roles dynamically via discovery.

Resource-rich nodes can serve as data aggregation and processing points, or bridges between a remote deployment and traditional network infrastructure (if desired), while simpler devices focus on sensing and actuation, creating a resilient multi-tier architecture.

How can we achieve (near-)zero-configuration deployment for emergency scenarios? Emergency response and hazardous environment monitoring demand systems that can be deployed rapidly without extensive configuration. μ**-Babel** provides automatic peer discovery across multiple protocols, self-organizing network formation, and decentralized coordination mechanisms that eliminate the need for pre-configured master nodes or manual network planning.

Devices will autonomously establish connectivity with each other, negotiate protocols, and begin operation upon being activated, with minimal configuration effort to enable rapid deployment even in hard-to-reach locations.

FIXED: over-promising

Expected Contributions

"main" aqui implica outras contributions secundárias, if that's the case não devia inclui-las também?

We plan to make the following contributions:

- A decentralized architecture supporting multiple communication protocols (BLE, LoRa, ZigBee, ESP-NOW) with adaptive switching based on Quality of Service (QoS) requirements, resource availability and device capabilities;
- A resource-efficient programming framework for embedded platforms that enables autonomous operation without central coordination, providing abstractions for multi-protocol communication, peer discovery, and opportunistic data forwarding;
- A proof-of-concept implementation demonstrating infrastructure- independent operation and automatic disaster-mode failover in a real-world deployment.

2

Related Work

Internet of Things (IoT) and home automation (domotics) systems share the same technological building blocks (wireless sensors, embedded devices, network communication) but diverge significantly in their operational focus and architectures:

mencionar MQTT, CoAP em detalhe somewhere?

IoT systems typically focus on **data collection and monitoring**, by streaming sensor readings to centralized platforms for analysis and processing, with control functions often being a secondary concern.

Domotics systems instead prioritize **real-time control and actuation** over physical spaces, where responsiveness and local autonomy are paramount for end user experience and privacy.

This distinction gains additional relevance when focusing on resilience requirements: while IoT deployments may tolerate delayed data aggregation and/or temporary connectivity issues in monitoring scenarios, domotics applications (such as emergency lighting control or Heating, ventilation, and air conditioning (HVAC) management) demand immediate local response regardless of network conditions.

Both domains, however, suffer from a common vulnerability when confronted with infrastructure failures during disasters: their predominantly cloud-centric architectures collapse precisely when autonomous operation becomes indispensable.

The challenges faced by disaster-resilient IoT systems span multiple dimensions:

Infrastructure failures eliminate access points that devices depend on for coordination;

Intermittent connectivity creates network partitions where subgroups of devices must operate autonomously without global state synchronization;

Resource constraints limit the computational, memory and energy budgets available for implementing sophisticated resilience mechanisms in embedded platforms.

These challenges are fundamentally architectural: existing IoT and domotics systems are designed around the assumption of stable infrastructure, treating network partitions and coordinator failures as transient anomalies rather than the norm.

Cloud-centric architectures place control decisions in remote servers, creating dependencies that cannot be satisfied when connectivity fails. Coordinator-based topologies (whether using dedicated gateways, master nodes, Software-defined networking (SDN) controllers, ...) concentrate the risk of failure in single points that can paralyze entire networks if compromised or disconnected. Even ostensibly distributed systems often rely on persistent connections (i.e., Transmission Control Protocol (TCP)) over a single medium, or heavy middleware platforms that exceed the capabilities of embedded devices and, once more, assume some form of infrastructure availability.

The fundamental point of contention is between resilience requirements (autonomy during infrastructure collapse) and resource constraints (limited computation, memory, and energy). Traditional approaches address one at the cost of the other: cloud platforms provide sophisticated coordination but fail during outages; purely local systems avoid external dependencies but struggle with inter-device coordination and protocol heterogeneity with low resource usage.

Disaster-resilient systems require architectural choices that prioritize autonomous Peer-to-peer (P2P) connectivity as the baseline mode of operation rather than having it as a secondary – even exceptional – fallback mechanism.

Naturally, this requires a fundamental rethinking of IoT systems design: how devices discover and communicate with each other using diverse protocol stacks (without depending on centralized cotnrol), how they achieve temporal synchronization and coordination without master beacons, how to optimize the usage limited available resources, and how to maintain secure operation throughout without persistent access to Certificate authorities (CAs) or Key distribution centers (KDCs).

FIXED: "suavizar" o parágrafo: mais sobre o problema menos sobre o uBabel

parágrafo moved para chap 3, WIP ainda

The following sections briefly cover related work across five areas that collectively enable autonomous operation: (Section 2.2) multi-protocol communication and adaptive selection based on Quality of Service (QoS) and device capabilities; (Section 2.1) P2P mesh networking and topology management for autonomous network formation without coordinator dependencies; (Section 2.3) decentralized synchronization mechanisms for coordinating multi-protocol communication without master nodes; (Section 2.4) lightweight data compression and energy optimization techniques to extend autonomous operation duration and device lifetime; and (Section 2.5) security and privacy mechanisms that enable systems to maintain secure communication after infrastructure failures.

For each area, we examine how existing approaches handle (or fail to handle) infrastructure failure scenarios, identify architectural assumptions that conflict with disaster-resilient requirements, and position μ-**Babel**'s contributions relative to the current state of the art.

2.1 P2P Mesh Networking and Topology Management

Infrastructure-dependent star topologies in which devices communicate via a central coordinator or gateway suffer from a similar fate to single-protocol communication: when that coordinator becomes unavailable or unreachable, the entire network loses connectivity.

This pattern pervades current IoT deployments, from Wi-Fi access point dependencies to LoRa Wide Area Network (LoRaWAN) gateway requirements, and becomes catastrophic in disaster scenarios where central coordinators are most likely to fail first.

P2P mesh architectures address this limitation by distributing coordination across all participating nodes, thus eliminating single points of failure. Nonetheless, achieving robust mesh operation requires solving three interconnected challenges: neighbor discovery and establishment of initial connectivity, topology maintenance as nodes join/leave a network, and efficient data routing through multi-hop paths when direct communication becomes impossible or inefficient.

2.1.1 Peer Discovery and Topology Maintenance

As discussed in Section 2.2.2, the Multi-Protocol IoT Gateway implementation [16] demonstrates Bluetooth Low Energy (BLE)-based neighbor discovery with Received signal strength indicator (RSSI) measurements for proximity estimation.

While that work focuses on multi-protocol integration, its topology management reveals a limitation of its tree-based approach: the system implements automatic parent reselection when coordinators fail, but the underlying tree structure imposes that nodes can only communicate through their parent-child relationships rather than arbitrary peer connections.

This restriction limits route diversity and resilience, creating dependency chains where a single intermediate node failure can disconnect entire subtrees.

A more sophisticated approach to membership management is presented in HyParView [18], in which each node maintains two distinct partial views for scalability: a small active view (size = fanout + 1) containing nodes with which symmetric links are actively maintained, and a larger passive view serving as a backup pool of potential neighbors that may be promoted to the active view if one of its nodes fails.

The active view is managed reactively, such that nodes are added during join operations and removed upon failure detection, while the passive view is maintained cyclically through periodic shuffle operations that exchange node identifiers between peers.

This hybrid strategy enables remarkable resilience, with the system recovering from 80% node failures with minimal reliability loss (maintaining 95% reliability) and from 50% failures in just 1-2 membership cycles, compared to 60+ cycles required by purely cyclic protocols like Cyclon [37].

HyParView's deterministic flooding approach, by broadcasting messages along the entire active view graph rather than probabilistic neighbor selection, enables fast failure detection since every active link is tested at each broadcast. The symmetric link requirement ensures bidirectionality: if node A can reach node B, then B can reach A, preventing the formation of one-way communication paths that complicate routing.

However, HyParView assumes TCP availability for maintaining persistent connections and using connection failures as implicit failure detectors. This dependency on full network stack functionality makes direct application to resource-constrained embedded platforms challenging, though the architectural principles of hybrid views and shuffle-based passive view maintenance remain valuable.

The heterogeneous disaster IoT architecture [26] discussed in Section 2.2.1 employs RPL for its Wireless Sensor Network (WSN) layer, demonstrating practical routing in resource-constrained disaster scenarios.

Their performance analysis reveals considerable trade-offs: convergence time scales linearly from 7 seconds for 20 nodes to 14.5 seconds for 100 nodes, while Packet loss ratio (PLR) at 10 hops reaches 80% with a 10-second sending interval but becomes acceptable at 20-second intervals.

These measurements highlight the throughput limitations imposed by tree topologies, in that their Instance 1 traffic (human data) must be restricted to 1-hop from the root node to ensure the least possible delay in its delivery, defeating the purpose of multi-hop mesh for critical communications.

The Delay-tolerant networking (DTN) component using mobile drones as data mules provides an alternative path for partitioned networks, with a maximum of 20 nodes per Destination-oriented directed acyclic graph (DODAG) to maintain acceptable convergence times during emergency situations, but this approach trades latency for eventual delivery rather than real-time mesh routing.

2.1.2 Routing in Partitioned and Intermittently Connected Networks

When network partitions prevent end-to-end paths, store-and-forward mechanisms enable eventual data delivery. The Bundle Protocol [29] addresses delay-tolerant networking through custody-based retransmission and opportunistic connectivity exploitation. While the protocol specification predates modern IoT deployments and was not designed specifically for resource-constrained devices, its core principles inform contemporary DTN approaches.

The framework presented in [23] implements elastic bandwidth utilization by dynamically adjusting transmission rates based on available connectivity, and supports scheduled, predicted, and opportunistic transmission windows, taking inspiration from the Bundle Protocol.

Data parcels are compressed, encrypted, and bundled before transmission, with a load balancer managing concurrent transfer threads to optimize bandwidth usage during brief connectivity windows.

While their Hypertext Transfer Protocol (HTTP)-based implementation targets cloud-backed IoT deployments, the core concepts of parceling data, maintaining transmission queues, and opportunistic forwarding during connectivity windows translate to P2P scenarios where aggregation nodes become neighbors in a mesh network.

2.1.3 Limitations of Centralized Coordination

The work on Resilient Edge-enabled IoT [6] addresses coordinator failures through dynamic leader election and backup mechanisms within their framework.

Their coordination model divides environments into collaboration areas, with resource-rich edge devices serving as coordinators that allocate tasks to workers under their supervision when problems arise. Workers, in turn, are *active agents* such as robots and IoT devices that reside in a particular environment, detect problems, and notify their coordinators.

When coordinators fail, the system automatically elects backups through adaptive decentralized consensus, providing "gentle degradation" during failures with restoration after recovery. Multiple coordinators operate independently, eliminating single points of failure within the coordination model itself.

This architecture depends fundamentally on edge servers running JVM-based SCAFI middleware (a Scala library) to execute the aggregate programs that specify coordination behavior, and these heavyweight infrastructure requirements – both the Java runtime environment and resource-rich edge computing nodes – conflict with embedded platform constraints and infrastructure-failure scenarios.

queremos mingle com o Babel... se calhar não entrar por aqui ou tudo bem porque não vamos usar os big-raspis para coordination?

While the aggregate computing paradigm separates concerns (sensing, actuation, communication, coordination), the implementation assumptions make it unsuitable for disaster-resilient systems where edge servers may be the infrastructure that fails. The formal guarantees of self-stabilization and compositional properties come at the cost of persistent computational infrastructure that an embedded-focused deployment cannot provide.

2.1.4 Discussion

Existing peer discovery and topology mainteance approaches demonstrate mechanisms appropriate for distributed operation but not without their limitations for resource-constrained platforms aimed at disaster scenarios.

HyParView's [18] hybrid view architecture (small active, large passive) provides remarkable resilience to node failures, but the assumption of TCP availability and persistent connections is not suitable for embedded devices using connectionless radio technology or operating under intermittent connectivity conditions.

The multi-protocol gateway's [16] BLE-based neighbor discovery works on our targeted hardware but imposes tree topologies with parent-child communication restrictions that limit route diversity and create dependency chains.

Routing Protocol for Low-Power and Lossy Networks (RPL) routing in the heterogeneous disaster architecture [26] demonstrates practical WSN operation but has significant throughput limitations: 20-second minimum send intervals at 10 hops, and restriction of critical traffic to 1-hop from root nodes.

The store-and-forward mechanism presented in the elastic bandwidth framework [23] enables partition tolerance but targets cloud-backed deployments rather than P2P mesh scenarios.

Coordination frameworks like the one in [6] provide dynamic leader election but depend on JVM-based middleware on resource-rich edge servers.

No existing work integrates connectionless neighbor discovery, hybrid view maintenance, true P2P routing (not tree-based), and partition-tolerant store-and-forward on resource-constrained embedded platforms.

μ-**Babel**'s approach (Section 3.2.2) must adapt HyParView's architectural principles to BLE/LoRa connectionless communication while supporting multi-hop routing without coordinator dependencies.

2.2 Multi-Protocol Communication and Adaptive Protocol Selection

Single-protocol (i.e. Wi-Fi only, BLE only, LoRa only, etc.) communication architectures are susceptible to a fundamental vulnerability: if their chosen medium becomes unavailable or inefficient due to interference, range limitations, or infrastructure failures, the entire system loses connectivity and most times ceases to function altogether.

This becomes especially grave in disaster scenarios where communication conditions change unpredictably due to factors such as Radio frequency (RF) interference from debris, obstacles blocking line-of-sight propagation, or damage to access points.

Existing approaches to multi-protocol IoT systems can be placed in three categories: multi-channel resilience architectures that orchestrate communication technologies for emergency scenarios; implementations that show the feasibility of protocol heterogeneity on embedded platforms; and adaptive selection frameworks that switch protocols based on runtime conditions.

2.2.1 Multi-Channel Resilience Architectures

Several disaster-focused systems explicitly address infrastructure failures through protocol diversity.

The AWCT (Always Connected Things) framework [19] orchestrates Low-Power Wide Area Network (LPWAN) on top of LoRaWAN with ad-hoc networks (Bluetooth and Wi-Fi) specifically for standby emergency communication. Their architecture adds three modules to standard IoT devices (Raspberry Pi boards, in their test case): a battery module for power management, a power interrupt handler that triggers emergency mode when the power grid fails, and an ad-hoc bridge that forwards packets between the Bluetooth, Wi-Fi and LPWAN interfaces.

The system leverages dense IoT device deployment to provide emergency coverage, demonstrating that existing infrastructure can still serve a purpose during scenarios where the main power grid suffers issues.

However, AWCT's reliance on centralized LoRaWAN gateways for Internet connectivity creates a single point of failure when those gateways become unreachable.

A more comprehensive heterogeneous approach is presented in [26], which integrates High frequency (HF) radio with Near Vertical Incidence Skywave (NVIS), satellite links, WSNs, and DTN with mobile drones for disaster monitoring. Their system uses RPL [38] with three separate instances to differentiate traffic by priority: human data (voice/text via Bluetooth) receives the highest priority, followed by drone-collected data and finally sensor data.

The NVIS backhaul provides 250 km coverage radius without line-of-sight requirements, offering a cost-effective alternative to satellite communications.

While demonstrating successful real-world validation in Antarctica and urban deployments, the architecture's core NVIS topology with centralized coordination contrasts fully distributed operational requirements. Furthermore, their WSN layer requires a minimum 20-second sending interval to maintain acceptable packet loss rates at 10 hops, which highlights throughput limitations of single-channel tree topologies.

Security-focused multi-channel approaches like MCSC-WoT [3] combine Advanced Encryption Standard (AES) encryption with dynamic channel hopping across 2.4 GHz Wi-Fi channels to defend against jamming and eavesdropping attacks. Their lightweight synchronization mechanism minimizes energy consumption while maintaining security through Frequency-hopping spread spectrum (FHSS) patterns generated via Pseudorandom number generator (PRNG). Nodes that lose synchronization can rejoin by hopping to random channels and waiting for the next synchronization signal.

However, the system depends on a master node broadcasting synchronization signals and uses an initial PRNG seed shared among all nodes, raising questions about scalability and seed distribution mechanisms – particularly how new nodes can acquire seeds if deployed to a network at a later time.

2.2.2 Protocol Heterogeneity on Embedded Platforms

The work presented in [16] shows the technical feasibility of multi-protocol operation on commodity Wi-Fi/BLE modules by integrating ESP-NOW, ZigBee, and Modbus protocols on devices from the ESP32 family to construct multi-hop, tree-based wireless networks.

Their implementation uses BLE advertising beacons for neighbor discovery with RSSI measurements (for distance estimation), computing parent selection priorities based on weighted combinations of child count, RSSI values, and hop count in the network tree of a given node. The system also supports automatic parent reselection when the current parent becomes unreachable, making it somewhat resilient to single node failures.

Testing demonstrated successful multi-hop operation up to 5 hops in office environments, but evaluation was limited to linear network topologies. Additionally, this architecture relies on a tree topology with master-slave communication that is centered around a gateway (coordinator) rather than being a P2P mesh, creating a single point of failure at the coordinator node.

2.2.3 Runtime Adaptive Protocol Selection

este é o paper do MDPI, removo?

A recent approach to protocol adaptation is presented in [44], which introduces a closed-loop control system with four integrated components: a context monitor sampling runtime metrics at 1 Hz, a decision engine using multi-criteria weighted scoring across six dimensions (message frequency, payload size, network conditions, packet loss rate, energy budget, QoS requirements), protocol adapters encapsulating protocol-specific libraries, and a learning component that adjusts thresholds using Exponentially weighted moving average (EWMA) for deployment-specific patterns.

Some key aspects of this work include hysteresis control with a threshold band to prevent oscillation between protocols, and switching cost awareness requiring benefits to exceed costs by a certain threshold before transitioning.

However, this framework assumes infrastructure availability for protocol endpoints (MQTT brokers, HTTP servers, Constrained Application Protocol (CoAP) endpoints) and was evaluated only on laptop platforms using Python libraries rather than embedded C implementations.

Nevertheless, the evaluation methodology provides valuable insight into the energy expenditure of these different approaches (further explored in Section 2.4.3), and grants useful foundations for more complex multi-protocol approaches, in particular with the multi-criteria decision framework and hysteresis control mechanism, but the actual system cannot operate when infrastructure fails.

The MINOS platform [35] exemplifies the limitations of centralized multi-protocol approaches. While providing sophisticated multi-protocol support (CORAL-SDN and Adaptable-RPL) with dynamic protocol deployment and real-time parameter tuning, the system depends fundamentally on a centralized SDN controller, MQTT broker, and web server infrastructure. The architecture's single point of failure means that when the controller becomes unreachable the entire system loses its adaptive capabilities and reverts to static operation at best, or complete failure at worst.

2.2.4 Discussion

Existing frameworks demonstrate the technical feasibility of heterogeneous channel/protocol coordination but rely on centralized control mechanisms incompatible with disaster scenarios. AWCT [19] depends on centralized LoRaWAN gateways for Internet connectivity, the heterogeneous disaster architecture [26] uses centralized NVIS coordination with minimum 20-second send intervals at 10 hops, MCSC-WoT [3] requires master nodes broadcasting synchronization signals, the multi-protocol gateway [16] uses tree topologies with gateway coordinators as single points of failure, and MINOS [35] assumes SDN controllers with persistent MQTT/web infrastructure.

While adaptive protocol selection frameworks [44] provide runtime switching strategies, they assume infrastructure availability (MQTT brokers, HTTP servers, CoAP endpoints) and have not been validated on embedded platforms.

No existing work addresses fully distributed multi-protocol coordination where nodes can autonomously select and switch protocols without persistent infrastructure. μ-**Babel**'s approach to decentralized protocol selection is detailed in Section 3.2.3, with masterless synchronization mechanisms addressed in 2.3.

se calhar "no existing work" é too much? posso não ter encontrado um mega óbvio

2.3 Decentralized Synchronization and Time Coordination

Multi-channel communication strategies, particularly those that employ coordinated channel hopping for security or efficiency reasons, require nodes to maintain synchronized clocks. Without time synchronization, devices cannot agree on when to switch channels, rendering coordinated multi-protocol operation impossible.

Traditional master-based synchronization approaches – where a designated coordinator is responsible for communicating timing signals – once more fall into the trap of having a single point of failure in that same master node: should it fail, the entire network becomes susceptible to losing accurate temporal reference, leading to the collapse of any coordination efforts.

This is a notorious challenge to overcome when designing distributed systems, and it becomes more prominent in the disaster scenarios discussed thus far, where infrastructure failures are likely to eliminate the nodes responsible for time synchronization in a given system.

Achieving robust time synchronization in such situations requires fully decentralized approaches that eliminate central coordinator dependencies while still remaining sufficiently lightweight to be executed on resource-constrained platforms.

2.3.1 Master-Based Synchronization as Counterexample

As discussed in Section 2.2.1, the MCSC-WoT framework [3] demonstrates a security-focused multi-channel hopping approach aimed at embedded platforms, but with a fundamental dependency on a master node broadcasting synchronization signals.

This approach uses a shared PRNG seed to generate channel-hopping sequences, with the master node broadcasting periodic synchronization beacons that slave nodes use to compensate for clock drift. Nodes that lose synchronization altogether can rejoin by hopping to a random channel and waiting for the next master beacon.

While the proposed goals were achieved in terms of minimizing energy expenditure and maintaining FHSS patterns, it inherits the fundamental limitation of all master-based approaches mentioned at the start of Section 2.3. If the master fails or becomes unreachable, participating nodes gradually drift out of synchronization until coordinated channel hopping is no longer possible.

The system's clock drift compensation algorithm demonstrates the feasibility of synchronization on ESP32 platforms, and their measured AES encryption performance validates that lightweight security can coexist with time synchronization on resource-constrained devices. However, the architectural dependency on a central coordinator fundamentally conflicts with infrastructure-independent operation requirements we aim to fulfill.

2.3.2 Gossip-Based Masterless Synchronization

Decentralized time synchronization eliminates coordinator dependencies by having nodes reach consensus on clock values through distributed local interactions. Two complementary approaches demonstrate the viability of gossip-based synchronization for wireless sensor networks.

mau usar WSN aqui?

The Randomized gossip-consensus-based sync (RGCS) algorithm proposed in [41] addresses time synchronization in dynamic WSNs through randomized asynchronous gossip. Each node maintains a logical clock (T) composed of rate (α) and offset (β) parameters, which together transform the node's hardware clock (τ) into synchronized logical time.

Rather than requiring fixed communication links between specific node pairs which might be fragile in dynamic topologies, their approach uses Poisson-based randomized link activation where each potential synchronization link activates with intensity λ.

The synchronization process operates through pairwise gossip exchanges: when a link activates, the triggering node sends a Sync-L beacon selecting a triggered neighbor, followed by bidirectional exchange of multivariable messages containing each node's current logical clock parameters $[\alpha, \beta, \tau]$.

The asynchronous randomized timing of these exchanges is advantageous in that collision rates drop to near zero compared to 19-23% for deterministic communication protocols, as independent Poisson intervals make simultaneous transmissions to the same receiver statistically unlikely.

The proposed RGCS employs a converge-to-max criterion rather than average-value consensus. This maximum-based approach achieves finite time convergence significantly faster than average-based protocols that require many iterations to converge under significant clock drift. Offset compensation follows suit, with nodes adjusting β parameters based on the difference between their and the neighbors' logical clocks.

Bounded communication delays are handled through a least-square estimation low-pass filter. This addresses the realistic concern of uplink and downlink delays differing, and avoids the symmetric delay assumptions made by many theoretical protocols. The filter's weighing parameter decreases over time, in order to restrain the negative effects of additive noise in stochastic approximation.

não sei se sei explicar este low-pass filter, vale a pena?

Storage complexity remains O($|N_i|$) per node per iteration (proportional only to the number of neighbors, not network size) thus ensuring scalability. The protocol simultaneously compensates both clock rate and offset, unlike approaches that handle these separately and thus require additional convergence time.

2.3.3 Coordination Through Passive View Maintenance

As discussed in Section 2.1.1, the shuffle-based passive view maintenance presented in HyParView [18] provides a complementary coordination mechanism. While primarily designed for topology management, the periodic shuffle operations in which nodes exchange lists of known peers also provide a *catalog* with potential synchronization partners beyond their immediate active neighbors. When a node receives a shuffle message containing peer descriptors (IDs and capabilities), it can evaluate these peers as candidates for time synchronization based on their advertised characteristics – for instance, prioritizing peers with Network Time Protocol (NTP) access or those maintaining more reliable clock sources.

FIXED: quantum leap? mencionar aqui GPS clock sync como possibilidade?

Nodes performing shuffle exchanges already communicate periodically; these same communication windows can opportunistically carry synchronization messages (piggybacking), reducing protocol overhead. The passive view serves as a pool of potential sync partners, so that when a node's active sync neighbors become unreachable, it can initiate sync exchanges with passive view members, providing added resilience to topology changes without requiring global network knowledge.

2.3.4 Discussion

Master-based synchronization approaches like MCSC-WoT [3] demonstrate feasibility on ESP32 platforms with measured AES encryption coexistence, but create single points of failure when master nodes become unreachable.

RGCS [41] eliminates coordinator dependencies, and provides converge-to-max synchronization with Poisson-based randomized gossip that achieves near-zero collision rates with just O($|N_i|$) storage complexity per node. However, it assumes homogeneous single-protocol networks where all nodes use the same communication medium with identical energy characteristics and range properties.

HyParView's [18] shuffle-based passive view maintenance provides complementary coordination through periodic peer exchanges that could carry piggybacked sync messages, but was not designed for time synchronization integration.

No existing work addresses decentralized time synchronization across heterogeneous multi-protocol networks where different communication technologies (BLE, LoRa, Wi-Fi) have distinct energy costs, range, and reliability properties that warrant protocol-specific gossip rates.

μ-**Babel**'s approach (Section 3.2.4) must extend RGCS to multi-protocol scenarios with per-protocol Poisson processes while guaranteeing and preventing protocol selection oscillations during sync events.

2.4 Lightweight Data Compression and Energy Efficiency

Energy constraints present one of the most fundamental limitations in battery-powered IoT deployments, particularly in hazardous deployments where replacements and recharging are impractical, and during disaster scenarios where grid power likely becomes unavailable.

While multi-protocol communication and decentralized coordination efforts provide resilience, they also introduce increased energy costs through frequent transmissions, protocol adaptation overhead and forwarding costs.

Data transmission dominates energy consumption across wireless technologies: LoRa and SIGFOX can represent up to 99.9% of power consumption for transmission, while even shorter-range technologies like BLE and IEEE 802.15.4 dedicate 85-90% of their energy budget to radio operations [33].

This reality makes data reduction techniques essential for extending operational lifetime in infrastructure-independent scenarios. However, compression itself consumes energy through additional computational work, thus creating an unavoidable tug-of-war between compression overhead and transmission benefits.

2.4.1 Prediction-Based Data Reduction

The Ambrosia protocol [33] demonstrates that lightweight prediction can achieve substantial data reductions on resource-constrained devices. The core idea is that not all sensor readings need to be transmitted if their values can be accurately predicted at the server (within the bounds of a specified error threshold), provided that both sender and receiver maintain a synchronized "prediction state".

Their approach uses window-based forecasting where the next sample is predicted by adding the average differences between recent previous samples, making it dramatically lighter than sophisticated time-series forecasting techniques like Autoregressive integrated moving average (ARIMA) [30] while still achieving comparable data reduction performance.

The protocol works by sending the first w (window size) true samples collected by the sensor node to the server, and for every sample after that comparing its true value to the one predicted locally; the true sample value is sent to the server only if the absolute difference between it and the prediction is greater than a user or application-specified error threshold δ.

The same simple prediction scheme is used on both ends of this communication, and the predicted value is the one used for further predictions – even on the sensor node that knows the true value, to ensure consistency between the two endpoints.

This design achieves up to 60% data reduction with appropriate δ configuration while maintaining sufficient accuracy for diverse applications: for error-tolerant use cases like anomaly detection, δ values will be higher to allow for reasonable fluctuations, thus enabling more significant data reductions, while error-sensitive applications requiring higher precision will naturally lean towards stricter error thresholds, but still benefitting from some amount of data reduction.

The major takeaway from this work is that the window-based prediction executed 99% faster than ARIMA forecasting in their evaluation, making it viable for streaming sensor data at high rates. The approach achieved 2× battery lifetime improvement in high-traffic scenarios and demonstrated compatibility with extremely constrained devices (livestock ear tags with Atmel 8-bit microcontrollers).

2.4.2 Lightweight Compression

While prediction-based reduction minimizes the number of transmissions, compression techniques are essential to reduce the size of the data that *must* be transmitted.

A few complementary approaches address different considerations and constraints: hybrid schemes balance accuracy with transmission costs through lossy/lossless models; lossless time-series compression exploits the natural temporal relation in continuous sensor readings; and two-tier architectures apply different techniques at both the sensor and gateway level to reduce energy costs.

Hybrid Lossy/Lossless Compression

The hybrid compression scheme presented in [10] addresses accuracy requirements by separating real-time lossy transmission from on-demand lossless reconstruction. The Fan algorithm adaptively sub-samples data maintaining bounded error ε, achieving average 7.8× and 2.1× Compression ratio (CR) for lossy and lossless compression, respectively.

The approach provides graceful degradation through battery-aware switching, in that when battery drops below a certain threshold (e.g., 20%), the system switches to lossy-only transmission instead of lossless, extending lifetime at the cost of reconstruction accuracy.

FIXED:. é bom? é mau? é adequado?

This trade-off aligns with disaster-scenario requirements where sustained operation during prolonged periods of network outages often takes higher priority over perfect data-fidelity, as approximate sensor readings over extended periods of time provide greater situational awareness than high-precision measurements from devices that quickly deplete their batteries. However, safety-critical sensors (e.g., structural integrity sensors, hazardous gas detectors) may require different thresholds (if any) or other policies to balance longevity with accuracy requirements.

Lossless Time-Series Compression

Sprintz [4] targets lossless compression for multivariate integer time-series with extreme resource constraints, achieving 2-10× compression ratios with <1KB memory footprint and 8-sample block sizes. The four-component algorithm combines forecasting (delta coding or Fast Integer REgression (FIRE) online learning), bit packing with zigzag encoding **TODO:** add citation here for zigzag or not really necessary?, Run-length encoding (RLE) for all-zero blocks, and optional Huffman coding.

Delta coding is extremely fast, and when combined with RLE becomes even more so, as it yields a run of zero errors if the data is constant, which is likely to happen for nominal sensor readings. FIRE forecasting yields better compression, but its online learning approach place it beyond the scope of this paper.

A *forescaster* is employed to predict each sample based on previous ones, and the difference between the next and the predicted sample are encoded. This difference is typically closer to zero than the next sample itself.

Any prediction errors from the previous forecasting step are zigzag encoded as a "payload", and a header is prepended with sufficient information to invert the bit packing.

If a block happens to consist only of zeros, this approach waits for a block in which there is a non-zero error, and the number of all-zero blocks are written out using RLE instead of the (empty) payload.

Finally, the bit packed representation of each block can be *entropy coded* using a Huffman coder, applied to the headers and payloads. This is done after bit packing because not only is it faster, but it also increases compression.

Decompression can achieve up to 3GB/s throughput without Huffman coding (>500MB/s with Huffman) in a single thread, enabling high-speed data retrieval on gateway-tier devices. Compression maintains >200MB/s on 8-bit data, sufficient for real-time operation even on embedded platforms.

Two-Tier Compression Architecture

The two-tier data reduction technique proposed in [27] applies compression at both sensor nodes (Tier 1) and gateways (Tier 2) to reduce energy consumption in the overall system.

Sensor nodes employ Delta encoding followed by RLE, allowing this approach to achieve 80-84% compression by exploiting the temporal correlation in sensor data. Gateways perform hierarchical clustering based on the Minimum description length (MDL) principle, transmitting *hypothesis* data sets and difference vectors instead of full data sets.

falar mais do MDL ou como é gateway side not so important?

The Delta+RLE implementation – validated through OMNeT++ simulation – proves to be lightweight enough for resource-constrained nodes, while gateway clustering reduces transmission to cloud infrastructure.

2.4.3 Receiver-Side Energy Considerations

An oft-overlooked aspect of IoT energy optimization is the asymmetry between sender and receiver energy consumption. The adaptive protocol selection framework analysis [44] revealed that receiver nodes consistently consume 15-20% more energy than senders across all evaluated protocols (MQTT, CoAP[5], HTTP).

This asymmetry shifts the energy bottleneck from endpoint sensors to intermediary nodes like gateways and has direct implications for gateway power management in disaster scenarios where battery replacement becomes impossible. The increased receiver cost stems from the added computational work required for parsing, processing, and managing incoming messages, particularly at high message rates.

For μ**-Babel**'s proposed heterogeneous architecture where sensor nodes might eventually forward data to aggregators/gateways, this finding has direct implications in that managing the power consumption of these gateways becomes as significant as sensor power optimization, since one gateway might receive substantial data from several nodes even if individually they don't perform frequent transmissions.

2.4.4 Discussion

Existing data reduction techniques demonstrate substantial energy savings on resource-constrained platforms but assume single-protocol deployments.

Ambrosia's [33] window-based prediction achieves 60% data reduction with 99% faster execution than ARIMA while maintaining accuracy within application-specific error thresholds, validated on 8-bit microcontrollers.

Sprintz [4] provides lossless time-series compression with 2-10× compression ratios using <1KB memory and 8-sample blocks, combining delta coding, zigzag encoding, RLE, and optional Huffman coding with >200MB/s compression throughput suitable for real-time embedded operation.

Hybrid lossy/lossless [10] schemes enable graceful battery-aware degradation, while two-tier architectures [27] (Delta+RLE at sensors, MDL clustering at gateways) achieve

80-84% compression exploiting temporal correlation.

None of these approaches address protocol heterogeneity by themselves, and different wireless technologies have distinct energy costs and range/accuracy trade-offs that warrant protocol-specific compression/reduction strategies.

No existing work integrates prediction-based reduction with protocol-aware adaptive thresholding and compression tuning for heterogeneous multi-protocol networks.

μ-**Babel**'s approach (Section 3.2.5) must extend lightweight prediction and compression to multi-protocol scenarios with per-protocol threshold and compression strategy selection while accounting for gateway receiver energy costs.

2.5 Security and Privacy for Resource-Constrained Emergency Communication

Disaster scenarios inherently involve sensitive data, such as location tracking for search-and-rescue operations, environmental conditions for risk assessment, among many others.

Unlike conventional IoT deployments where security infrastructure can be carefully provisioned, emergency-focused deployments must balance security guarantees against the possibility of infrastructure failure.

Traditional security approaches that assume persistent connectivity to CAs, KDCs, or even blockchain networks cannot fully function, if at all, when those very structures collapse. Embedded platforms further exacerbate this issue by ruling out computationally expensive cryptographic approaches that could more easily provide a given network with a certain degree of independence from those structures.

2.5.1 Authentication and Privacy

Authentication protocols are necessary to establish device identity and enable secure communications. In disaster scenarios, authentication mechanisms must operate without centralized coordinators while maintaining privacy guarantees that prevent device tracking and activity correlation.

Lightweight Group Authentication

The Group Authentication Scheme at the Edge (GASE) [24] demonstrates lightweight group authentication suitable for resource-constrained devices through a combination of Shamir Secret Sharing (SSS) and aggregated Message Authentication Code (MAC) usage. It operates within a three-tier cloud-edge-IoT architecture, focusing on asynchronous mass authentication of the low-end IoT nodes therein.

There are several phases to this protocol, summarily:

1. **Initialization:** The Authentication Server (AS) generates $2N$ secret-shadows, and distributes them to each IoT node (two per node). Nodes are then divided into L groups, each with a unique $(t-1)$-degree polynomial along with a random value r used to compute share tokens;

2. **Hashed-shares reveal:** Within a given time window w, $(t-1)$ nodes reveal *one* of their secret shares to the Group leader (GL);

3. **Group leader authentication:** The GL uses Lagrange interpolation to recover the polynomial secret from the revealed shares, verifies authenticity, and all remaining participants derive sessions keys from the recovered secret;

4. **Server authentication:** The GL combines all node tags (MACs derived from session key) and sends them to the edge entity, which collects all GLs tags and aggregates them into a single one via XOR operations, the AS receives this *Agg-MAC* and verifies the tag.

The protocol's computational efficiency stems from requiring only hash operations and modular arithmetic, avoiding expensive elliptic curve operations.

For key updates the AS randomly generates a new r, a new GL, and a new polynomial parameter for each group, which are published to allow nodes to compute fresh shares from their stored secret without needing a secure channel for updates or redistribution.

The main issue with this approach is that it assumes persistent availability of its centralized infrastructure components: the AS, GLs and edge entities for aggregation. If the GL becomes inactive, the authentication process cannot proceed. While RPi gateways could assume the role of an AS for initial provisioning, the GL dependency persists among sensor nodes themselves and isn't trivially solvable within the proposed architecture.

Nevertheless, the individual mechanisms are still quite valuable: the *Agg-MAC* pattern efficiently aggregates multiple authentications, the threshold approach (t-out-of-n nodes must participate for group authentication) provides fault tolerance against partial node failures, and the session key derivation combining group secrets and device-specific keys prevents node impersonation.

Privacy-Preserving Authentication

Physically Unclonable Functions (PUFs) provide a distinctive hardware fingerprint to devices by taking advantage of natural random variations present in integrated circuits, thus allowing for the derivation of pseudonyms which are a common approach to privacy in the context of IoT deployments.

Existing privacy-preserving authentication protocols based on PUFs – whether using challenge-response pairs [13, 42] or pseudonym systems [8] – assume either specialized hardware unavailable on ESP32/Raspberry Pico platforms to leverage PUFs, or centralized verification infrastructure (pseudonym servers, credential authorities). Both assumptions are incompatible with μ-**Babel**'s commodity hardware and infrastructure-less deployment model.

2.5.2 Distributed Key Management

The challenge of establishing shared cryptographic keys across resource-constrained devices without central coordination has received substantial attention, though most approaches make assumptions incompatible with disaster scenarios.

Polynomial-Based Key Management

The Lightweight Polynomial-based Key Management (LPKM) protocol [11] provides a compelling foundation for embedded key distribution. Using polynomial evaluation on 8MHz ATmega128L microcontrollers, this approach generates 128-bit group keys in 2-16 milliseconds with storage requirements of only 496-1616 bytes (depending on security parameter k).

The proposed scheme supports multiple key types within a unified framework: pairwise keys between (non-)neighboring nodes, cluster keys for local groups, and group keys for larger networks. Storage complexity is O($k + 1$) coefficients per node, independent of network or group size, ensuring scalability.

No less critical for disaster scenarios – where the possibility of malicious agents must still be taken into account – LPKM enables distributed revocation without central coordination. When a node is compromised, legitimate nodes can independently compute updated keys that exclude the revoked member, with no re-keying delay or coordinator involvement. Periodic share updating provides backward secrecy through timer-based refresh operations that require no coordination.

This approach does assume secure bootstrapping via a KDC that preloads polynomial shares into devices before deployment. For planned disaster-resilience installations (campus sensor networks, building monitoring systems), this physical preloading is acceptable, and a trusted device (laptop, Personal digital assistant (PDA), etc.) can serve as a KDC, keeping in line with our zero-config efforts.

Zero-Preloading Approaches

Alternative schemes attempt to eliminate pre-distributed keys entirely. The lightweight distributed key agreement protocol presented in [20] achieves a considerable speedup compared to Diffie-Hellman by employing hash functions and bit-wise comparisons rather than modular exponentiation.

The approach generates temporal key pairs on-demand through random secret number generation, with nodes deriving shared keys via hash-based operations and bit-wise comparisons of prefix bit-strings, thus eliminating the need for pre-distributed keys.

However, it requires an inter-sensor authentication protocol to secure the initial public key exchange, creating a circular dependency: authentication requires keys, but key establishment requires authentication. Furthermore, revocation mechanisms are only briefly mentioned without any concrete details, limiting the potential of this implementation.

This highlights a fundamental limitation in zero-config security: establishing trust without pre-shared secrets or trusted third parties is theoretically impossible. We accept this limitation and opt for provisioning during deployment.

too harsh? posso tar errado

2.5.3 Lightweight Encryption on Embedded Platforms

The MCSC-WoT framework [3] demonstrates that AES encryption can operate efficiently on ESP32 platforms while maintaining multi-channel communication. Their implementation measures encryption overhead on actual hardware and validates that symmetric cryptography remains viable on resource-constrained devices without prohibitive energy or computational costs.

This confirms that *μ*-**Babel** can employ AES-based encryption for confidentiality without posing a significant compromise to battery lifetime or real-time performance required for disaster scenarios. The coordination and synchronization challenges associated with their master-based approach have been discussed separately in Section 2.2.1.

2.5.4 Centralized Security Approaches (Incompatible with Disasters)

Several recent approaches leverage blockchain or centralized infrastructure to solve IoT security challenges, but their dependencies render them unsuitable for disaster scenarios.

The blockchain-based access control system in [36] moves Policy Decision Points onto distributed ledgers, eliminating single points of failure in access control. However, it assumes continuous connectivity to blockchain nodes and cloud storage for off-chain data, failing precisely when infrastructure collapses.

Similarly, the decentralized Public key infrastructure (PKI) approach using blockchain-based Name/Value Storage presented in [39] replaces CAs with distributed blockchain nodes. While removing central CA dependencies, it still requires Internet connectivity for NVS queries and assumes blockchain nodes remain reachable, which invalidates their usage during disasters.

Decentralized Attribute-based encryption (ABE) schemes such as the one in [15] eliminate central authorities through multi-authority cryptography but rely on bilinear pairings with computational costs far exceeding the capabilities of our targeted platforms.

2.5.5 Discussion

Existing lightweight security mechanisms demonstrate feasibility on resource-constrained platforms but assume infrastructure availability incompatible with disaster scenarios.

The minimal viable set of mechanisms to adopt comprises: (1) efficient authenti-

FIXED: identificar o minimal set of mechanisms cation primitives using hash operations and secret-sharing, (2) polynomial-based key management minimal storage complexity, (3) distributed revocation without coordinator involvement, (4) AES encryption validated on embedded platforms.

GASE [24] provides efficient group authentication using only hash operations and modular arithmetic with valuable mechanisms (MAC aggregation, threshold $(t-1)$-of-n fault tolerance, session key derivation preventing impersonation), but depends on persistent availability of centralized components (AS, glsGL, edge entities for aggregation).

LPKM [11] enables distributed key management with fast key generation on 8MHz microcontrollers, $O(k+1)$ storage complexity independent of network size, and distributed revocation without coordinator involvement, but requires secure KDC bootstrapping during pre-deployment, which is a reasonable requirement.

Privacy-preserving authentication protocols based on PUFs [13, 42] and/or pseudonym systems [8] assume either specialized hardware (unavailable on ESP32/Pico) or centralized verification infrastructure (pseudonym servers, credential authorities).

Zero-preloading key agreement approaches [20] face circular dependencies (authentication requires keys, key establishment requires authentication) without concrete revocation mechanisms.

AES encryption operates efficiently on ESP32 (validated by MCSC-WoT [3]) without prohibitive energy costs.

Blockchain-based approaches (access control [36], decentralized PKI [39]) eliminate single points of failure but require continuous connectivity to distributed ledgers; decentralized ABE schemes [15] rely on bilinear pairings exceeding embedded platform capabilities.

No existing work integrates distributed authentication, key management, and encryption for fully autonomous post-disaster operation while accepting pre-disaster physical provisioning as a pragmatic bootstrapping mechanism.

μ-**Babel**'s hybrid security model (Section 3.2.6) distinguishes pre-disaster provisioning (LPKM shares, GASE secret-shadows via KDC) from post-disaster autonomous operation (adapted threshold authentication with local GL election, gossip-based revocation, AES encryption with compress-then-encrypt).

2.6 Summary

Table 2.1 summarizes the key architectural differences between existing work and *μ*-**Babel** across the technical domains examined in this chapter.

A common pattern is present across them: existing approaches rely on centralized coordination mechanisms that become single points of failure during infrastructure collapse.

Protocol coordination depends on master nodes or SDN controllers, mesh topologies assume consistent connectivity or gateway coordinators, time synchronization requires

Table 2.1: Related work summary

Domain	**Related Work Limitations**	**μ-Babel Approach**
Protocol coordination	Master nodes (MCSC-WoT), SDN controllers (MINOS), gateway coordinators (multi-protocol gateway)	Fully distributed selection with hysteresis control
Mesh topology	Tree-based restrictions, TCP-dependent links (HyParView), centralized coordination	Connectionless BLE/LoRa discovery, hybrid views, true P2P routing
Time sync	Master beacons (MCSC-WoT), single-protocol gossip (RGCS)	Protocol-specific Poisson rates, multi-graph converge-to-max
Compression	Single-protocol optimization (Ambrosia, Sprintz), sender-focused	Protocol-aware adaptive thresholds, gateway receiver energy management
Authentication	Centralized components (GASE AS/GLs), specialized hardware (PUFs)	Two-phase: pre-disaster provisioning, post-disaster local GL election
Key management	KDC bootstrapping (LPKM), circular dependencies (zero-preloading)	Pre-deployment shares distribution, gossip-based revocation

master beacons, and authentication protocols depend on centralized servers or group leaders.

Even distributed approaches like HyParView and RGCS make assumptions (persistent TCP connections, single-protocol homogeneity) incompatible with resource-constrained multi-protocol disaster scenarios.

3

Solution Architecture

TODO: UNDER CONSTRUCTION PAST THIS POINT

3.1 Introduction

3.2 System Model

3.2.1 Como se pretende resolver cada desafio

μ**-Babel** addresses these concerns through a decentralized approach that aims to eliminate the dependency on centralized coordination, continuous connectivity and cloud infrastructure.

Rather than treating infrastructure failures as an exceptional condition requiring failover mechanisms, our architecture focuses on autonomous P2P connectivity as the baseline mode of operation, with infrastructure integration as an added benefit when available, rather than a hard dependency.

This approach means rethinking multiple aspects of traditional IoT systems design: from multi-protocol communication strategies that adapt to the available mediums and device resources without centralized control, to gossip-based synchronization mechanisms that achieve coordination through local interactions, to lightweight data compression techniques to make the most out of the limited storage and battery available in each device.

3.2.2 MicroBabel's Approach

Tree topologies and coordinator-based approaches limit resilience, and heavyweight middleware platforms or HyParView's TCP requirements exceed embedded device capabilities.

μ**-Babel** intends to address these limitations through a fully distributed P2P mesh architecture without coordinator dependencies. The system adapts HyParView's hybrid view concept (maintaining small active neighbor sets for actual communication and larger passive backup lists for failure recovery) but implements discovery and maintenance at the link layer using BLE and LoRa advertisements rather than at the transport layer via TCP connections, accommodating the connectionless nature of embedded radio/wireless communication.

FIXED: BLE/LORA mismatch with TCP

Neighbor discovery is performed through advertising beacons emitted by one or more of the supported protocols (e.g., BLE, LoRa, Wi-Fi) carrying relevant peering information (node identity, capabilities, current neighbor counts, etc.), with periodic gossip exchanges that propagate topology information beyond single-hop range.

It's immediately apparent that – if these exchanges aren't properly integrated – the same node advertising itself through different protocols could lead to duplicate neighbor entries on the receiver side. To address this, nodes possess unique IDs that are consistent across protocols, and by advertising their capabilities they simultaneously inform potential neighbors of which protocols they can expect to receive their advertisements from, which prevents unnecessary processing of packets that would only contain redundant information (e.g., by peeking at the sender's ID). Nodes supporting multiple protocols naturally serve as bridges, enabling topology formation and message forwarding between nodes that might not share the same protocol stack, this is exemplified in Figure **??**.

FIXED: addressed dupes and bridges

ADD FIGURE WITH (LORA ONLY) -> (BLE/LORA) -> (BLE ONLY)

This approach mitigates potential range limitations: while BLE typically provides 10-50m coverage depending on environment and antenna configuration, gossip-based propagation enables nodes to learn about distant neighbors through multi-hop dissemination, supporting informed routing decisions and topology adaptation.

Unlike the tree topology of Multi-Protocol Gateway or the RPL DODAG structure of heterogeneous disaster architecture, μ**-Babel** will implement true P2P mesh networking where any node can communicate with any other node through dynamically selected multi-hop paths.

Route selection considers multiple factors gathered through local observation and gossip: link quality metrics (RSSI, PLR), neighbor availability across different protocols, and current energy budgets.

Store-and-forward mechanisms inspired by the Bundle Protocol will enable operation during network partitions, but will be adapted for P2P rather than cloud-centric operation: data parcels are compressed (using techniques from Section 2.4), queued in local storage (flash, MicroSD cards) when no forward path exists, and opportunistically transmitted when connectivity windows emerge, be it through topology changes or protocol switching creating new communication opportunities.

Coordination emerges from local interactions via gossip-based information propagation, decentralized time synchronization (Section 2.3), and adaptive protocol selection (Section 2.2), rather than centralized allocation of roles.

3.2.3 MicroBabel's Approach

While existing multi-channel approaches rely on centralized decision-making for channel selection – whether through master nodes (MCSC-WoT), SDN controllers (MINOS), or infrastructure endpoints – μ**-Babel** will address these limitations through decentralized *protocol* selection, extending beyond single-medium channel hopping.

The system will employ opportunistic multi-protocol operation (BLE + Wi-Fi + LoRa + ESP-NOW) wherein devices adaptively select communication mediums based on factors like message priority, neighbor availability, energy budget, network conditions and device capabilities. This selection will be performed through local decision-making informed by information gathered through gossip protocols, without requiring SDN controllers or infrastructure coordination.

Hysteresis control mechanisms adapted from the Adaptive Protocol Selection Framework prevent oscillations in these decisions while allowing rapid response to changing conditions.

Protocol-specific compression strategies account for the different energy/bandwidth trade-offs across the proposed stack (e.g. aggressive compression for (relatively) energy-expensive LoRa, lighter compression for short-range BLE), integrating the receiver energy asymmetry observations into power management decisions.

Unlike AWCT's LoRaWAN gateway dependency, Heterogeneous IoT's NVIS centralized backhaul, MCSC-WoT's master-based synchronization, or MINOS's SDN controller requirement, μ**-Babel** operates autonomously in a P2P manner with no static coordinator dependencies.

The multi-protocol capability provides resilience through diversity rather than optimization through centralized selection: when conditions render one medium unsuitable, devices autonomously transition to alternatives without requiring coordinator intervention.

3.2.4 MicroBabel's Approach

While MCSC-WoT demonstrates master-based synchronization feasibility and RGCS provides masterless convergence, both assume homogeneous single-protocol networks.

μ**-Babel** will extend RGCS to heterogeneous multi-protocol networks through protocol-specific Poisson processes with distinct activation rates. Rather than a single λ controlling overall sync frequency, the system employs per-protocol gossip rates that take into account the energy required for each transmission, and the current network conditions:

λ_**BLE** (high frequency): provides rapid local synchronization with nearby neighbors, leveraging BLE's low energy cost for frequent exchanges

λ_**LoRa** (low frequency): maintains long-range temporal coordination, using LoRa's more expensive transmissions sparingly

λ**_Wi-Fi** / Others (medium frequency): situational use based on network density, energy budgets and other factors

Each protocol will have its own Poisson-triggered gossip loop, but all updates contribute to a single shared logical clock via the converge-to-max criterion explored above. When sync events from different protocols occur near-simultaneously, the max operation's associativity ensures correct convergence regardless of update order.

This multi-graph gossip approach provides several advantages over homogeneous sync:

Faster convergence: High-frequency BLE gossip can achieve tighter local consensus while sparse LoRa exchanges prevent drift between distant clusters. The combination converges faster than either protocol alone.

Resilience through diversity: If 2.4 GHz interference disrupts BLE/Wi-Fi synchronization, LoRa maintains loose temporal coordination across the network. Conversely, if LoRa experiences poor conditions, local BLE sync keeps nearby nodes coordinated.

Natural energy optimization: Protocol characteristics directly inform sync frequencies, so that more expensive long-range transmissions occur rarely while cheap short-range exchanges happen frequently, matching energy budgets to communication needs.

The hysteresis control mechanisms adapted from Section 2.2 will prevent oscillations in protocol selection during sync events: sync partner and protocol choices stabilize around locally optimal configurations rather than continuously switching between equally-viable options.

The approach aims to handle network partitions gracefully: nodes maintain synchronization within their reachable partition using available protocols, and when partitions merge (through mobility or topology changes), the converge-to-max criterion naturally reconciles previously independent temporal references without requiring special merge logic.

Unlike MCSC-WoT's master-based approach, this architecture will operate in a fully P2P fashion with no fixed coordinator dependencies. The multi-protocol extension goes beyond RGCS's original homogeneous network assumptions, since heterogeneous gossip graphs with protocol-specific characteristics warrant distinct sync frequencies, enabling simultaneous optimization of convergence speed, energy consumption, and range coverage.

3.2.5 MicroBabel's Approach

Extending Ambrosia's lightweight prediction approach to multi-protocol scenarios, μ-**Babel** will reduce data transmission through prediction-based filtering: nodes transmit sensor readings only when prediction error exceeds configurable thresholds (similar to Ambrosia), with protocol-aware tuning to balance accuracy against energy constraints across heterogeneous protocols.

Building on prediction-based reduction, the goal is to apply Delta+RLE compression to the readings that are transmitted, combining both techniques for added energy savings and reduced data volume over the air.

Following the two-tier architecture pattern, compression occurs at sensor nodes before transmission, with just the first sample in each window being transmitted with no compression applied.

Adaptive threshold selection will adjust δ based on both channel characteristics and node state:

estes deltas vinham do ambrosia, fica confuso agora que falo de delta encoding?

- **High δ for LoRa**: Tolerates larger prediction errors to minimize expensive long-range transmissions, sending only when predictions deviate significantly. These packets undergo aggressive Delta+RLE compression to minimize airtime costs.
- **Low δ for BLE**: Requires tighter prediction accuracy for cheap local exchanges, allowing for more frequent transmissions to maintain precision. Lighter compression (delta encoding only) is applied here for lower latency.
- **Battery-aware adaptation**: As node energy depletes, increased δ across all channels reduces transmission frequency, extending lifetime for critical messages. Compression at this point becomes more lossy to further minimize transmitted data volume.

Application-aware transmission will distinguish between data types: periodic sensor readings (temperature, humidity, light) use window-based prediction with configurable δ thresholds, while discrete events (emergency alerts, button presses, motion detection) transmit immediately without prediction. For predictable streams, accuracy-critical applications maintain low δ, while trend monitoring can accept higher values to reduce transmission frequency.

Compression strategies also vary by data type: environmental sensors can tolerate more prominent Delta+RLE compression given their high temporal correlation, while critical alerts transmit with minimal (if any) compression to reduce latency.

Gateway energy management accounts for the 15-20% receiver energy cost by implementing selective radio shutdown: gateways can disable specific protocols during low-activity periods, waking periodically to check for incoming sync beacons or when prompted via other protocols. Decompression overhead in these nodes is minimal compared to transmission costs at the sensor level.

entrar pela coisa de desligar os rádios ainda mais?

The expectation is that the combination of lightweight prediction, adaptive thresholding, and protocol-specific tuning together with Delta+RLE compression will provide substantial energy savings without requiring complex and computationally intensive compression algorithms that would themselves consume significant power.

3.2.6 MicroBabel's Approach

While zero-config approaches face circular authentication dependencies and centralized schemes fail when infrastructure collapses, μ-**Babel** will adopt a hybrid security model that distinguishes between pre-disaster deployment and post-disaster autonomous operation.

Pre-disaster deployment phase: For planned installations (building sensor networks, campus-wide monitoring), devices are provisioned during initial deployment:

- **LPKM polynomial shares:** Preloaded via KDC for distributed key management
- **GASE secret-shadows:** Two per node for threshold authentication ($(t-1)$-of-n reveal protocol)

These are reasonable assumptions for disaster-resilience scenarios, as preparation happens before any emergency operation needs to take place.

Post-disaster autonomous operation: Once a disaster occurs and infrastructure fails, the network operates autonomously using mechanisms that require no central coordination:

- **Threshold-based authentication** (adapted from GASE): Nodes form ad-hoc authentication groups where $(t-1)$-of-n members reveal secret shares (pre-distributed) to derive session keys via Lagrange interpolation. Local coordinator election replaces GASE's centralized GLs Session key derivation combines group secrets with device-specific keys to prevent impersonation, while aggregated MAC tags enable efficient multi-node verification without individual authentication overhead.

  too dense?
- **Distributed revocation** (adapted from LPKM): nodes independently compute updated keys excluding compromised peers, with revocation decisions propagated via gossip
- **Periodic share updating** (adapted from LPKM): timer-based key refresh for backward secrecy, no coordinator needed
- **Lightweight encryption**: AES encryption (demonstrated by MCSC-WoT on ESP32) with compress-then-encrypt strategies (Section 2.4) to minimize ciphertext size and transmission energy
- **Channel hopping/protocol switching for security**: multi-channel/multi-protocol operation complicates eavesdropping and uses masterless synchronization (RGCS) rather than master-based beacons

Threat model: The system will prioritize availability and resilience over complete compromise resistance. Assumptions include:

- **Honest majority**: most nodes behave correctly; adversaries cannot compromise majority simultaneously
- **Local adversary**: attackers can monitor/disrupt local regions but not entire network simultaneously
- **Physical security during deployment**: devices can be provisioned securely before disaster strikes

This threat model reflects emergency priorities where maintaining communication capability matters more than preventing all possible attacks. If an adversary with sophisticated capabilities attacks during a disaster, communication infrastructure would already be their target regardless of security mechanisms.

3.3 Progress Report and Planning

4

Conclusion

Bibliography

[1] S. M. Alamouti, F. Arjomandi, and M. Burger. "Hybrid Edge Cloud: A Pragmatic Approach for Decentralized Cloud Computing". In: *IEEE Communications Magazine* 60.9 (2022), pp. 16–29. DOI: 10.1109/MCOM.001.2200251 (cit. on p. 1).

[2] Z. Amiri et al. "Resilient and dependability management in distributed environments: A systematic and comprehensive literature review". In: *Cluster Computing* 26.2 (2023), pp. 1565–1600 (cit. on p. 3).

[3] P. Barman, R. Chowdhury, and B. Saha. "Multi-channel secure communication framework for wireless IoT (MCSC-WoT): enhancing security in Internet of Things". In: *Cluster Computing* 28.11 (2025), p. 691 (cit. on pp. 11, 13–15, 23, 24).

[4] D. Blalock, S. Madden, and J. Guttag. "Sprintz: Time Series Compression for the Internet of Things". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2.3 (2018-09). DOI: 10.1145/3264903 (cit. on pp. 18, 19).

[5] C. Bormann, A. P. Castellani, and Z. Shelby. "CoAP: An Application Protocol for Billions of Tiny Internet Nodes". In: *IEEE Internet Computing* 16.2 (2012), pp. 62–67. DOI: 10.1109/MIC.2012.29 (cit. on p. 19).

[6] R. Casadei et al. "Engineering Resilient Collaborative Edge-Enabled IoT". In: *2019 IEEE International Conference on Services Computing (SCC)*. 2019, pp. 36–45. DOI: 10.1109/SCC.2019.00019 (cit. on pp. 9, 10).

[7] M. Chui, M. Collins, and M. Patel. *The Internet of Things: Catching up to an accelerating opportunity*. McKinsey & Company. 2021-11. URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/iot-value-set-to-accelerate-through-2030-where-and-how-to-capture-it (visited on 2025-12-26) (cit. on p. 1).

[8] S. Das et al. "Securing IoT-Based Smart Healthcare Systems by Using Advanced Lightweight Privacy-Preserving Authentication Scheme". In: *IEEE Internet of Things Journal* 10.21 (2023), pp. 18486–18494. DOI: 10.1109/JIOT.2023.3283347 (cit. on pp. 21, 24).

[9] A. Dauda, O. Flauzac, and F. Nolot. “A Survey on IoT Application Architectures”. In: *Sensors* 24.16 (2024). ISSN: 1424-8220. DOI: 10.3390/s24165320 (cit. on p. 1).

[10] C. J. Deepu, C.-H. Heng, and Y. Lian. “A Hybrid Data Compression Scheme for Power Reduction in Wireless Sensors for IoT”. In: *IEEE Transactions on Biomedical Circuits and Systems* 11.2 (2017), pp. 245–254. DOI: 10.1109/TBCAS.2016.2591923 (cit. on pp. 17, 19).

[11] X. Fan and G. Gong. “LPKM: A Lightweight Polynomial-Based Key Management Protocol for Distributed Wireless Sensor Networks”. In: *Ad Hoc Networks*. Ed. by J. Zheng et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 180–195. ISBN: 978-3-642-36958-2. DOI: 10.1007/978-3-642-36958-2_13 (cit. on pp. 22, 24).

[12] P. Fouto et al. “Babel: A Framework for Developing Performant and Dependable Distributed Protocols”. In: *2022 41st International Symposium on Reliable Distributed Systems (SRDS)*. 2022, pp. 146–155. DOI: 10.1109/SRDS55811.2022.00022 (cit. on p. 3).

[13] P. Gope and B. Sikdar. “Lightweight and Privacy-Preserving Two-Factor Authentication Scheme for IoT Devices”. In: *IEEE Internet of Things Journal* 6.1 (2019), pp. 580–589. DOI: 10.1109/JIOT.2018.2846299 (cit. on pp. 21, 24).

[14] S. Hamdan, M. Ayyash, and S. Almajali. “Edge-Computing Architectures for Internet of Things Applications: A Survey”. In: *Sensors* 20.22 (2020). ISSN: 1424-8220. DOI: 10.3390/s20226441 (cit. on p. 1).

[15] J. Han et al. “Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption”. In: *IEEE Transactions on Parallel and Distributed Systems* 23.11 (2012), pp. 2150–2162. DOI: 10.1109/TPDS.2012.50 (cit. on pp. 23, 24).

[16] K. Khanchuea and R. Siripokarpirom. “A Multi-Protocol IoT Gateway and WiFi/BLE Sensor Nodes for Smart Home and Building Automation: Design and Implementation”. In: *2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*. 2019, pp. 1–6. DOI: 10.1109/ICTEmSys.2019.8695968 (cit. on pp. 7, 10, 12, 13).

[17] D. Kreković et al. “Reducing communication overhead in the IoT–edge–cloud continuum: A survey on protocols and data reduction strategies”. In: *Internet of Things* 31 (2025), p. 101553. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2025.101553 (cit. on p. 1).

[18] J. Leitao, J. Pereira, and L. Rodrigues. “HyParView: A Membership Protocol for Reliable Gossip-Based Broadcast”. In: *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN’07)*. 2007, pp. 419–429. DOI: 10.1109/DSN.2007.56 (cit. on pp. 7, 9, 15, 16).

[19] H. Li, K. Ota, and M. Dong. “Always Connected Things: Building Disaster Resilience IoT Communications”. In: *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. 2019, pp. 570–577. DOI: 10.1109/ICPADS47876.2019.00087 (cit. on pp. 11, 13).

[20] C.-C. Lin, S. Shieh, and J.-C. Lin. “Lightweight, Distributed Key Agreement Protocol for Wireless Sensor Networks”. In: *2008 Second International Conference on Secure System Integration and Reliability Improvement*. 2008, pp. 96–102. DOI: 10.1109/SSIRI.2008.30 (cit. on pp. 22, 24).

[21] P. Maciel et al. “A survey on reliability and availability modeling of edge, fog, and cloud computing”. In: *Journal of Reliable Intelligent Environments* 8.3 (2022), pp. 227–245 (cit. on pp. 1, 3).

[22] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh. “Reliability and high availability in cloud computing environments: a reference roadmap”. In: *Human-centric Computing and Information Sciences* 8.1 (2018), p. 20 (cit. on p. 1).

[23] R. Montella, M. Ruggieri, and S. Kosta. “A fast, secure, reliable, and resilient data transfer framework for pervasive IoT applications”. In: *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2018, pp. 710–715. DOI: 10.1109/INFCOMW.2018.8406884 (cit. on pp. 8, 10).

[24] M. Nakkar, R. AlTawy, and A. Youssef. “GASE: A Lightweight Group Authentication Scheme With Key Agreement for Edge Computing Applications”. In: *IEEE Internet of Things Journal* 10.1 (2023), pp. 840–854. DOI: 10.1109/JIOT.2022.3204335 (cit. on pp. 20, 24).

[25] J. Pan and J. McElhannon. “Future Edge Cloud and Edge Computing for Internet of Things Applications”. In: *IEEE Internet of Things Journal* 5.1 (2018), pp. 439–449. DOI: 10.1109/JIOT.2017.2767608 (cit. on p. 2).

[26] J. Porte et al. “Heterogeneous wireless IoT architecture for natural disaster monitorization”. In: *EURASIP Journal on Wireless Communications and Networking* 2020.1 (2020), p. 184 (cit. on pp. 8, 10, 11, 13).

[27] A. K. M. Al-Qurabat, C. Abou Jaoude, and A. K. Idrees. “Two Tier Data Reduction Technique for Reducing Data Transmission in IoT Sensors”. In: *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. 2019, pp. 168–173. DOI: 10.1109/IWCMC.2019.8766590 (cit. on p. 19).

[28] J. Ren et al. “Collaborative Cloud and Edge Computing for Latency Minimization”. In: *IEEE Transactions on Vehicular Technology* 68.5 (2019), pp. 5031–5044. DOI: 10.1109/TVT.2019.2904244 (cit. on pp. 1, 2).

[29] K. Scott and S. Burleigh. *Bundle protocol specification*. RFC 5050. RFC Editor, 2007. URL: https://www.rfc-editor.org/rfc/rfc5050.html (cit. on p. 8).

[30] R. H. Shumway and D. S. Stoffer. “ARIMA Models”. In: *Time Series Analysis and Its Applications: With R Examples*. Cham: Springer International Publishing, 2017, pp. 75–163. ISBN: 978-3-319-52452-8. DOI: 10.1007/978-3-319-52452-8_3 (cit. on p. 17).

[31] S. Sinha. “State of IoT 2025: Number of connected IoT devices growing 14% to 21.1 billion globally”. In: *IoT Analytics* (2025) (cit. on p. 1).

[32] A. Staff. *Summary of the Amazon DynamoDB Service Disruption in the Northern Virginia (US-EAST-1) Region*. 2025. URL: https://aws.amazon.com/message/101925/ (visited on 2025-12-26) (cit. on p. 1).

[33] S. Suryavansh et al. “A data-driven approach to increasing the lifetime of IoT sensor nodes”. In: *Scientific Reports* 11.1 (2021), p. 22459. DOI: 10.1155/2018/4283087 (cit. on pp. 16, 19).

[34] I. R. Team. *AWS Outage Analysis: October 20, 2025*. 2025. URL: https://www.thousandeyes.com/blog/aws-outage-analysis-october-20-2025 (visited on 2025-12-26) (cit. on p. 1).

[35] T. Theodorou et al. “A Multi-Protocol Software-Defined Networking Solution for the Internet of Things”. In: *IEEE Communications Magazine* 57.10 (2019), pp. 42–48. DOI: 10.1109/MCOM.001.1900056 (cit. on pp. 12, 13).

[36] H. T. T. Truong et al. “Towards Secure and Decentralized Sharing of IoT Data”. In: *2019 IEEE International Conference on Blockchain (Blockchain)*. 2019, pp. 176–183. DOI: 10.1109/Blockchain.2019.00031 (cit. on pp. 23, 24).

[37] S. Voulgaris, D. Gavidia, and M. Van Steen. “Cyclon: Inexpensive membership management for unstructured p2p overlays”. In: *Journal of Network and systems Management* 13.2 (2005), pp. 197–217 (cit. on p. 7).

[38] T. Winter et al. *RPL: IPv6 routing protocol for low-power and lossy networks*. RFC 6550. RFC Editor, 2012. URL: https://www.rfc-editor.org/rfc/rfc6550.html (cit. on p. 11).

[39] J. Won et al. “Decentralized Public Key Infrastructure for Internet-of-Things”. In: *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*. 2018, pp. 907–913. DOI: 10.1109/MILCOM.2018.8599710 (cit. on pp. 23, 24).

[40] L. Xing. “Reliability in Internet of Things: Current Status and Future Perspectives”. In: *IEEE Internet of Things Journal* 7.8 (2020), pp. 6704–6721. DOI: 10.1109/JIOT.2020.2993216 (cit. on p. 3).

[41] N. Xiong et al. “Randomized and Efficient Time Synchronization in Dynamic Wireless Sensor Networks: A Gossip-Consensus-Based Approach”. In: *Complexity* 2018.1 (2018), p. 4283087. DOI: 10.1155/2018/4283087 (cit. on pp. 14, 16).

[42] H. Yıldız, M. Cenk, and E. Onur. "PLGAKD: A PUF-Based Lightweight Group Authentication and Key Distribution Protocol". In: *IEEE Internet of Things Journal* 8.7 (2021), pp. 5682–5696. DOI: 10.1109/JIOT.2020.3032757 (cit. on pp. 21, 24).

[43] W. Yu et al. "A Survey on the Edge Computing for the Internet of Things". In: *IEEE Access* 6 (2018), pp. 6900–6919. DOI: 10.1109/ACCESS.2017.2778504 (cit. on p. 1).

[44] D. Żatuchin and M. Azarskov. "An Adaptive Protocol Selection Framework for Energy-Efficient IoT Communication: Dynamic Optimization Through Context-Aware Decision Making". In: *Informatics* 12.4 (2025). ISSN: 2227-9709. DOI: 10.3390/informatics12040125 (cit. on pp. 12, 13, 19).