BIS Innovation Hub

Project FuSSE
(Fully Scalable Settlement Engine)

Exploring the scalable, flexible and secure design of modern settlement engines

January 2026

IDB

banco central Chile

BANK OF CANADA
BANQUE DU CANADA

Publication date: January 2026.

Contents

Acronyms and abbreviations

AE	Advanced economy
AI	Artificial intelligence
AKS	Azure Kubernetes Service
API	Application programming interface
CPMI	BIS Committee on Payments and Market Infrastructures
CPU	Central processing unit
DLT	Distributed ledger technology
EMDE	Emerging market and developing economy
FMI	Financial market infrastructure
FuSSE	Fully Scalable Settlement Engine
HSM	Hardware security module
IDB	Inter-American Development Bank
IoT	Internet of Things
IPS	Instant payment system
JSON	JavaScript Object Notation
MVP	Minimum Viable Product
NIST	National Institute of Standards and Technology
NP	Nondeterministic polynomial
PFMI	Principles for financial market infrastructures
PKI	Public key infrastructure
PoC	Proof of concept

PQC	Post-quantum cryptography
RAM	Random-access memory
RTGS	Real-Time Gross Settlement
TARGET	Trans-European Automated Real-time Gross settlement Express Transfer
TIPS	TARGET Instant Payment Settlement
TPS	Transactions per second
UPI	Unified Payments Interface

Executive summary

Financial market infrastructures (FMIs) are the backbone of the financial system and must remain secure, resilient and adaptable as technologies and markets evolve. The continued global rise of digital payments – accelerated by rapid innovation in areas such as the Internet of Things (IoT), artificial intelligence (AI)-driven commerce and expanding fintech participation – places increasing demands on existing infrastructures. These developments promise efficiency and inclusion but also introduce greater complexity, scalability challenges and new forms of operational and cyber risk.

To meet these challenges, the next generation of FMIs will need to embody three reinforcing design qualities: flexibility, to adapt to innovation and regulatory change; scalability, to accommodate sustained growth and stress conditions; and security, to ensure quantum readiness and cryptographic agility[1] in an evolving threat environment.

Project FuSSE (Fully Scalable Settlement Engine) explored these issues through a proof of concept (PoC) that examined how a modular, microservices-based[2] architecture could support the design of flexible, scalable and secure settlement systems. The project showed that microservices could enable systems to process high transaction volumes efficiently, achieving 10,000 transactions per second (TPS)[3] without linear increases in computing power. This architecture allows for individual services, including those handling cryptographic operations, to scale independently, improving performance and resilience. It could also facilitate cryptographic agility, enabling adaptation to emerging post-quantum cryptography (PQC) standards without large system redesigns.

At the same time, the project highlighted trade-offs. Microservices architectures introduce new layers of operational complexity, require careful orchestration and expand the potential attack surface. PQC algorithms add computational overhead, which could multiply across service boundaries, but these impacts could be mitigated through targeted scaling and load management.

The project also underscored that operational agility – the ability of institutions to adapt governance, certification and incident-response frameworks in parallel with technological change – is as important as cryptographic agility for maintaining trust and continuity.

---

[1] Cryptographic agility means designing a system in such a way so encryption and signature methods can be swapped or upgraded more easily (for example, if an algorithm becomes unsafe), without needing to rebuild the whole system.

[2] Microservices are independent services (ie building blocks), each responsible for a specific function and able to operate, be updated, and be scaled separately from the others.

[3] FMI grade level performance typically also means measuring delays and how quickly transactions become final during heavy load. These results are from simulations and will vary by deployment and configuration.

For some advanced economies already running high-capacity instant payment systems (IPS), Project FuSSE showed approaches to modularising the settlement core - breaking it into distinct components - and embedding quantum-resilient cryptography and how this could enable cryptographic and operational agility.

For some smaller or emerging-market jurisdictions that may be operating legacy architectures and considering upgrading their systems or planning to introduce real-time settlement, the project demonstrates a modular, open source approach that could scale without requiring one-for-one (ie linear) increases in infrastructure investment.

Although Project FuSSE is a PoC, its findings provide practical insights for central banks and FMI operators. Its purpose is to explore design principles – not to prescribe operational models or policy choices – and as such it could be deployed by interested parties for experimentation and learning.

The results should be interpreted as technical findings illustrating architectural feasibility under controlled test conditions, rather than a performance benchmark or implementation guide. The project does not deliver a minimum viable product (MVP) or production-ready components of a system, nor does it meet operational, security or regulatory requirements under the principles for financial market infrastructures (PFMI). The findings of the project could help system designers consider important aspects of modern approaches to system scaling and quantum agility, both essential to the future of FMIs.[4]

Project FuSSE was made possible in partnership with and through resource contributions from the Inter-American Development Bank (IDB), the Central Bank of Chile and the Bank of Canada.

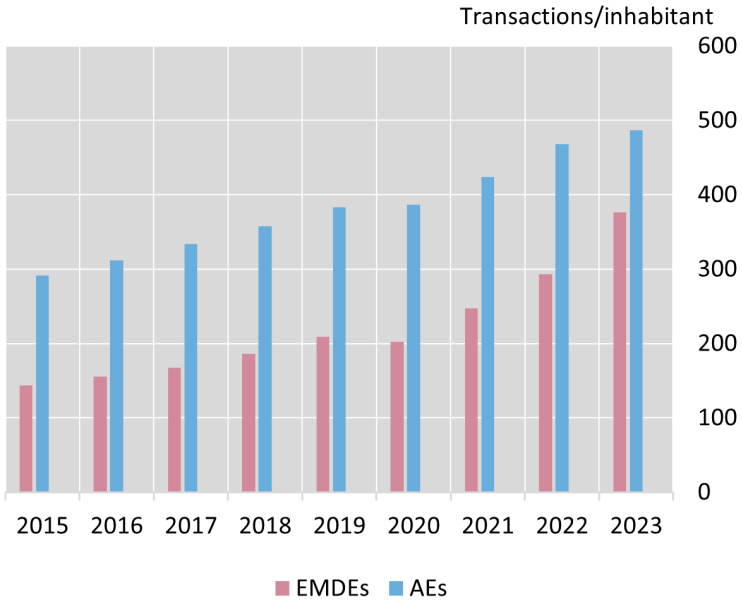
[4] The project does not: (i) specify or test a particular payment scheme, cost model or governance arrangement; (ii) recommend any specific deployment model (the use of cloud infrastructure was illustrative and deployment-agnostic); (iii) select or endorse a particular PQC algorithm or migration path; or (iv) provide full functional coverage of a real time gross settlement (RTGS) or IPS (including but not limited to liquidity management, gridlock resolution, participant access, clearing or netting variations).

1. Introduction

Financial market infrastructures (FMIs) are the backbone of the financial system and as such must remain secure, resilient and adaptable over many years.

Across the world, the volume of digital payments continues to rise each year, with especially strong growth in emerging market economies (see Graph 1). In Southeast Asia, digital retail payment volumes have expanded rapidly for over a decade, while in Latin America adoption accelerated sharply during the Covid-19 pandemic. Looking ahead, transaction volumes are expected to grow even more rapidly,[5] driven not only by these trends but also by new developments such as Internet of Things (IoT)-based payments and emerging forms of artificial intelligence (AI)-driven, agentic commerce.

Cashless payment volumes — Graph 1



Note: Number of cashless transactions (credit transfers, direct debits, card and e-money payments and cheques) per capita for emerging markets and developing economies (EMDEs) and advanced economies (AEs).

Source: CPMI, *Red Book statistics*.

While these innovations promise greater efficiency and convenience, they also create scalability challenges for traditionally designed FMIs. Furthermore, the breadth and

[5] See CapGemini Research Institute (2025). Non-cash transactions stood at approximately 1.4 trillion in 2023 and are forecasted to more than double to 3.5 trillion in 2029.

diversity of participants in FMIs is likely to grow as new kinds of institutions, such as fintechs, gain access. This expansion increases both the availability of and demand for new products and services, many built on emerging technologies.

While this fosters innovation, it also adds complexity and heightens the challenges of designing infrastructures with sufficient flexibility.

Increases in both volume and participation could grow harder for traditionally designed FMI systems to manage.[6] The architecture and design of next-generation FMIs could consider at least three reinforcing qualities:

- **flexibility,** to evolve structurally and adapt to new technologies, regulations and market innovations while maintaining stability;
- **scalability,** to handle long-term growth, high volumes and sudden surges in activity (without sufficient capacity, stress or growth could cause operational failures with systemic impact);[7] and
- **security,** in the form of quantum readiness and both operational and cryptographic agility, to safeguard trust in the face of a complex and evolving cyber threat landscape.

Together, these qualities enable the architecture and design of stable, resilient systems that could be capable of supporting the dynamic needs of the future.

While traditional measures of payment system performance emphasise cost, speed and reliability, these describe outcomes as observed by users at a given point in time. By contrast, Project FuSSE focuses on the architectural qualities that sustain those outcomes as technologies, market conditions and risks evolve.

Flexibility, scalability and security function as enablers of these traditional objectives: flexibility supports long-term cost efficiency and adaptability to policy or technological change; scalability maintains high throughput and operational reliability as volumes expand,[8] and security, including quantum readiness, safeguards trust and continuity.

Project FuSSE therefore does not assume that cost, speed and reliability are already achieved, but rather seeks to explore how they could be maintained and improved over successive iterations of system design.

---

6 See Darbha et al (2025), ECB (2023) and CPMI (2022).

7 See CPSS-IOSCO (2012).

8 Scalability also maintains costs proportional to uptake. A scalable system reduces the risks of building unused capacity, avoiding high capital cost.

2. Project FuSSE

2.1 Overview

Project FuSSE (Fully Scalable Settlement Engine) explores how settlement engines could be designed from the outset with three reinforcing qualities: (i) the flexibility to evolve as standards mature, (ii) the scalability to handle exponential growth without proportional infrastructure costs, and (iii) the security to remain resilient in a post-quantum threat environment.

While several instant payment systems (IPS) have successfully adopted microservices architectures, for example the Unified Payments Interface (UPI) in India, the TARGET Instant Payment Settlement (TIPS) in Europe and Pix in Brazil, optimising for scalability and resilience, Project FuSSE places equal emphasis on forward-looking security requirements, particularly the transition to PQC.

Project FuSSE differs from distributed ledger technology (DLT)-based settlement approaches. While DLT systems offer benefits such as atomic settlement and cryptographic verification across multiple participants, they often face throughput constraints due to consensus mechanisms and synchronisation requirements. Project FuSSE shows that other approaches could achieve comparable or greater transaction throughput, while incorporating quantum-resistant cryptography, when using cloud-native design patterns that enable independent scaling of security-intensive operations.[9]

2.2 Technical approach

The PoC was designed as a settlement engine architecture rather than as a full payment scheme. It simulates the core functions that underpin real-time settlement in central bank money, consistent with RTGS principles, while remaining agnostic to the payment layer or use case that connects to it. The architecture could therefore support next-generation retail IPS, wholesale interbank settlement or other FMI components requiring high throughput and deterministic finality.

The system was deployed in a cloud-based environment to demonstrate elastic scaling and resilience using widely available open source tools (eg Kubernetes, Kafka, Redis), illustrating that cloud-native principles, containerisation, stateless processing and horizontal scaling could be applied in either private, hybrid or on-premise infrastructures.[10]

---

[9] References to alternative settlement designs are intended to provide context rather than a head-to-head evaluation. Any comparative assessment would need to align trust assumptions, finality semantics and threat models before drawing conclusions on scalability or security.

[10] It should be noted that this does not imply that FMIs should operate in the public cloud.

The architecture introduced several novel features under the themes of flexibility and scalability. FuSSE employs a microservices-based[11] design in which each functional component operates as a stateless service that could be deployed, updated or replaced independently. This modularity enables continuous system evolution without requiring full-system redeployment. The architecture supports horizontal scalability, making it possible to increase overall capacity by running additional service instances rather than upgrading hardware, achieving sub-linear growth in computing resources as transaction volumes rise.

The system also implements a decentralised communication pattern, in which transaction messages carry their own routing instructions ("routing slips"). Decentralised routing improves modularity but requires strong controls for route integrity, versioning and auditability in order to ensure predictable processing and supervisory transparency.

Testing validated that such an architecture could handle large and rapid increases in transaction volumes while maintaining fast settlement and enabling new functions to be applied with minimal disruption.

The project also tested the application of PQC[12] to explore the potential of a microservices approach in supporting cryptographic agility, as well as to understand the impact of emerging security standards on scalability and flexibility.

2.3 Monolithic vs microservices architectures

Traditional financial systems often rely on monolithic architectures, in which application logic, data access and interfaces are bundled into a single deployable unit. While this simplifies deployment and ensures consistency, it limits scalability and flexibility as systems grow.[13] Even minor updates require redeployment of the entire system, creating brittleness, slowing innovation and exposing single points of failure. Over time, such architectures become bottlenecks, particularly in financial infrastructures facing high transaction volumes, frequent regulatory change and evolving cyber risks.[14]

In contrast, microservices architectures break functions and software into small, independent pieces (or services) that are each designed for a specific function and that communicate via lightweight protocols. Each service could be scaled, updated

---

[11] A microservices architecture is a way of designing software in which applications are broken down into small, independent services that each handle a specific function. These services communicate through Application programming interface (APIs), making systems more scalable, more flexible and easier to maintain.

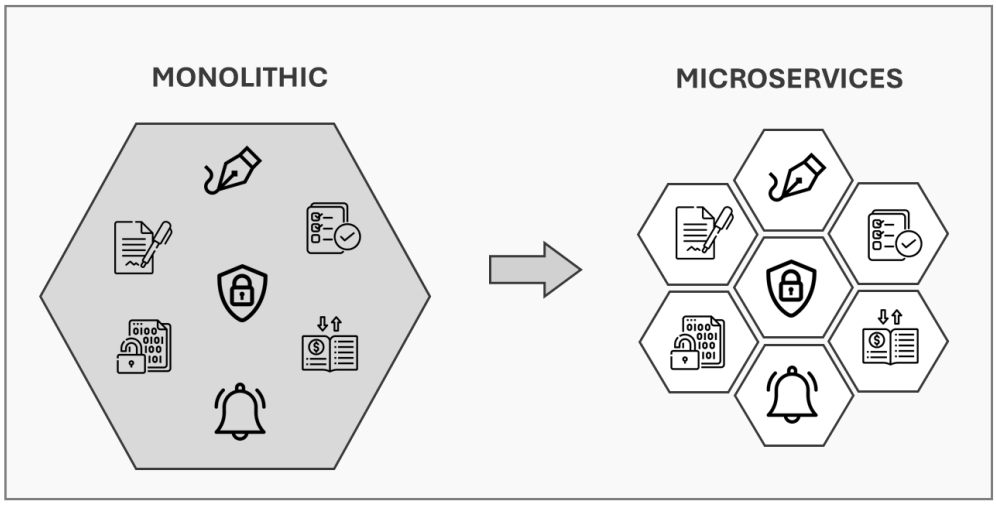
[12] PQC is important because most of today's cryptography relies on mathematical problems that are extremely hard for classical computers to solve but that could be broken by future quantum computers. This creates a risk of the "harvest now, decrypt later" strategy, in which criminals store sensitive information, such as financial data, with the intention of unlocking it once quantum technologies mature.

[13] Other drawbacks of monolithic architectures may include high complexity and coupling, "all-or-nothing" deployment, technology lock-in and potential reliability risks, while other advantages may include easier testing and debugging.

[14] See Tapia et al (2020).

and operated independently, enabling greater flexibility and faster adaptation to new technologies or regulatory requirements. This modularity also allows for incremental adoption of enhanced security standards, including PQC.

Monolithic vs microservices architecture Graph 2



Note: Monolithic (left-hand side) vs microservices-based (right-hand side) architectures.

However, the approach adds complexity: orchestration,[15] data consistency and inter-service security[16] must be carefully managed, and the attack surface expands as the number of interactions grows. In fintech and payments, cloud-native and event-driven microservices are increasingly seen as a way to balance scalability, security and flexibility in a composable framework.[17,18]

Instant payment systems (IPS) and microservices architectures

Several payment systems demonstrate how microservices architectures could support high scalability, flexibility and resilience in real-time retail payments.

In India, UPI was launched in 2016 with a microservices architecture. UPI has grown rapidly and now settles around 18 billion payments every month across more than

[15] Orchestration manages and synchronises automated tasks across multiple systems, combining them into comprehensive workflows so that individual tasks operate together to achieve a specific goal or process.

[16] Inter-service security in microservices ensures that communication between services is authenticated, authorised and encrypted to prevent unauthorised access and data breaches.

[17] See Challa (2021).

[18] See Battula (2025).

675 banks.[19] Its modular architecture has enabled the system to scale to higher transaction volumes and to integrate new services and participants more easily.

In Europe, the TIPS system, operated by the Eurosystem, similarly leverages modular and distributed design features to support very high throughput. TIPS can process up to 2,000 payments per second and has demonstrated the capacity to complete and finalise more than 43 million transactions in a single day.[20]

In Brazil, the Pix system was launched by the Central Bank of Brazil in 2020. Built to accommodate massive participation, Pix now has almost 900 million account aliases registered by 160 million individuals and more than 15 million businesses. In aggregate, users initiate some 6.6 billion transactions every month.[21]

These examples highlight how microservices could provide the architectural foundation for rapid scaling, continuous innovation and resilience in the face of rising digital payment volumes. They illustrate the benefits of modular design for the future development of FMIs.

It is also worth noting that payment schemes and the FMIs that settle their transactions (which may include a production settlement engine) could have different governance arrangements.

For example, in Brazil, the Pix scheme is owned by the central bank, which also operates SPI, its payment system. TIPS follows a similar approach. In India, UPI is run jointly by private banks and the central bank through the National Payments Corporation of India (NPCI), an organisation created specifically for this purpose.[22]

2.4 Architecture and components

The Project FuSSE PoC showed how a microservices-based architecture could be used for a settlement engine. In this design, all transaction-related tasks are performed by a set of independent microservices that scale horizontally to increase overall system capacity.

Each service is responsible for a discrete function, such as decrypting, processing, validating or settling transactions, before issuing an encrypted confirmation message to both the sender and the receiver. A supporting set of technologies manages these services and enables secure communication between them.

The service is stateless and therefore receives all the information it requires to complete its task directly within the transaction message itself, eliminating the need to reference historical records. This not only enables rapid processing but also allows multiple instances of the same service to operate simultaneously, thereby supporting

[19] See NPCI (2025).

[20] See Bank of Italy (2020).

[21] See Central Bank of Brazil (2025).

[22] See Frost et al (2024).

horizontal scaling. Capacity could thus be increased by running more service instances, rather than by upgrading the underlying hardware.

To coordinate processing, Project FuSSE tested a decentralised communication pattern (known as a routing slip pattern)[23] in which transaction messages carry their own "itinerary" - akin to a dynamic list of the steps required for completion (Graph 2).

Each service reads the embedded routing slip, performs its task, prunes the completed step and forwards the message to the next service. This removes the reliance on a centralised orchestrator, improving resilience and efficiency. It also means that the full processing path is visible within each message, which enhances traceability and auditability.[24]

This communication pattern provides multiple benefits. By decoupling orchestration logic from services, the system allows for new steps to be added or existing steps to be modified without redesigning a central controller, improving modular flexibility.

Common processing units, such as for signature verification, could be reused across different transaction flows, while the stateless design improves fault tolerance by allowing retries or rerouting if a component becomes unavailable. Versioned routing ensures that each transaction follows a stable process even if the system is updated, while process-agnostic services could be reused across multiple flows. Together, these features simplify maintenance, support diverse deployment models and enhance adaptability to evolving business requirements.

Additionally, the communication among microservices follows an event-driven design pattern,[25] in which events executed by one microservice are subscribed to by others. This minimises hard dependencies and makes it easier to evolve or replace components.

In practice, each transaction event is accompanied by a routing slip detailing its required processing chain. The framework automatically updates the routing slip before forwarding messages, abstracting this logic away from individual service developers.

The FuSSE architecture therefore combines efficiency with modularity, while maintaining transparency for auditing and control.

---

[23] More information can be found in sources like G Hohpe and B Woolf, *Enterprise integration patterns: Designing, building, and deploying messaging solutions*, Addison-Wesley, October 2003.

[24] Decentralised orchestration can improve modularity, but it introduces governance and control requirements around route integrity, versioning and auditability. In production environments, routing metadata and permitted processing paths would need strong integrity protection and change controls to maintain predictable processing and supervisory transparency.

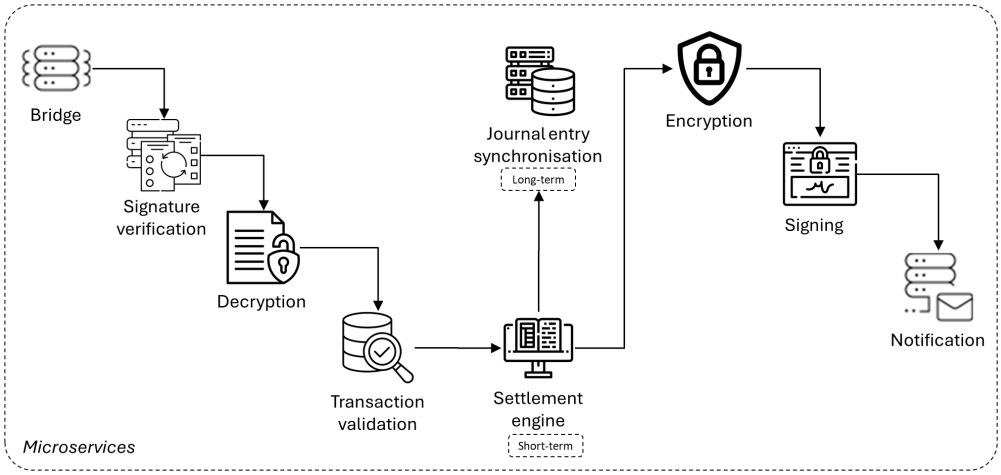
[25] For additional information, see Christudas (2019). An event-driven design reduces tight connections between services by letting them communicate in response to an event instead of through direct calls, making it easier to update or replace parts without affecting others. A direct call from service A to service B depends on service B being available and responding in an expected way.

Additional information on the architecture and open source components and underlying technologies employed by Project FuSSE can be found in Appendix A.

2.5 Microservices design and workflow

The Project FuSSE PoC architecture uses many different microservices[26] to process, decrypt, manage and settle transactions as well as to encrypt and sign notifications to senders and receivers (Graph 3).

FuSSE architecture — Graph 3



The architecture included more microservices than shown above (eg those for monitoring the system), but these were not included for simplicity.

Together, these services illustrate how a microservices-based FMI design could support scalability, flexibility and enhanced security while offering a practical pathway for integrating quantum-ready cryptography into next-generation settlement systems. The end-to-end settlement process tested in the project can be simplified into nine steps, as shown in Graph 3.

[26] The PoC implemented the following microservices:
- bridge, the entry point for participant messages;
- signature verification, which validates the authenticity of digital signatures against sender public keys;
- encryption and decryption, quantum-ready mechanisms to protect confidentiality;
- signing, which secures messages to guarantee integrity and authenticity;
- signature key pair generation, which provides secure keys for cryptographic functions;
- transaction validation, which performs business checks such as funds availability;
- settlement engine, which executes ledger updates, acknowledgments and multi-ledger support;
- journal entry synchronisation, which persistently stores transaction outcomes;
- notification, which communicates settlement updates to participants;
- audit trail and logging, which records status messages for monitoring and client queries; and
- profile management, which administers digital identities and participant information.

- First, a transaction message (associated with a payment, for example) is received into the bridge microservice.
- The signature verification microservice then checks that the message signature is from a participant.
- If it is, the message is decrypted by the decryption microservice.
- The transaction validation microservice then checks if the message format is correct, if the sender's account has met predefined conditions (eg in the context of a payment, if the sender has enough money in their account) and that the receiver's account information is correct.
- Now that the settlement message is received, verified, processed, decrypted and validated, it can be settled. The settlement engine microservice manages this. To ensure speed, this microservice updates the sender and receiver balances in fast short-term memory.
- To ensure completeness, the journal entry synchronisation microservice updates the long-term ledger with the full payment details in persistent storage.
- Finally, following settlement, the sender and receiver are notified (by the notification microservice) with a message that is encrypted and signed by two separate microservices, the encryption and signing microservices, respectively (see Section A.4 for information on the decryption and encryption standards employed by the PoC).

2.6 Cryptography

Cyber resilience in FMIs is a critical determinant of the resilience of the wider financial system and economy.[27] While cyber resilience encompasses multiple dimensions, one essential element is the protection of participant privacy (confidentiality) and transaction integrity through the encryption of payment messages transmitted to and from the system.

Equally important are strong authentication mechanisms, to ensure that only authorised parties can access or initiate transactions, and non-repudiation controls that prevent participants from denying the validity of their actions within the system.

Traditional cryptography relies on mathematical problems that are computationally difficult to solve using classical computers. This protects access to systems and data by making decryption practically impossible without the appropriate keys. Quantum computing represents a transformative advance in computational capabilities, posing a disruptive challenge: algorithms such as Shor's algorithm[28] could make these problems solvable in practice, undermining the foundations of current cryptographic methods that underpin global financial systems. This risk is heightened by the "harvest

[27] See CPMI-IOSCO (2016).

[28] Shor's algorithm is a quantum algorithm theoretically capable of calculating prime factors of large numbers. This algorithm poses a threat to asymmetric cryptography (such as the encryption algorithm RSA), which relies for its security on the difficulty of efficiently factoring very large numbers (BIS (2023a)).

now, decrypt later" strategy, in which adversaries capture encrypted data today with the expectation of decrypting them once quantum computing resources become available.[29]

To address this risk and strengthen resilience, the Project FuSSE cryptography module implements a hybrid architecture, enabling quantum-ready algorithms to run alongside traditional ones. This design was chosen with two key objectives:

1. **cryptographic agility**, the ability to evolve and replace cryptographic algorithms as new standards emerge without redesigning the entire system; and
2. **scalability** – since cryptographic operations are resource-intensive, microservices could be scaled independently to meet performance demands, minimising the impact on the rest of the settlement engine.

As mentioned, Project FuSSE integrates PQC microservices with traditional technology in its architecture. The PQC microservices are:

- **signature verification,** which validates the digital signatures on each payment message; and
- **signing,** which securely signs messages to ensure the authenticity and integrity of outgoing data.

The microservices that are implemented with traditional cryptographic technology are:[30]

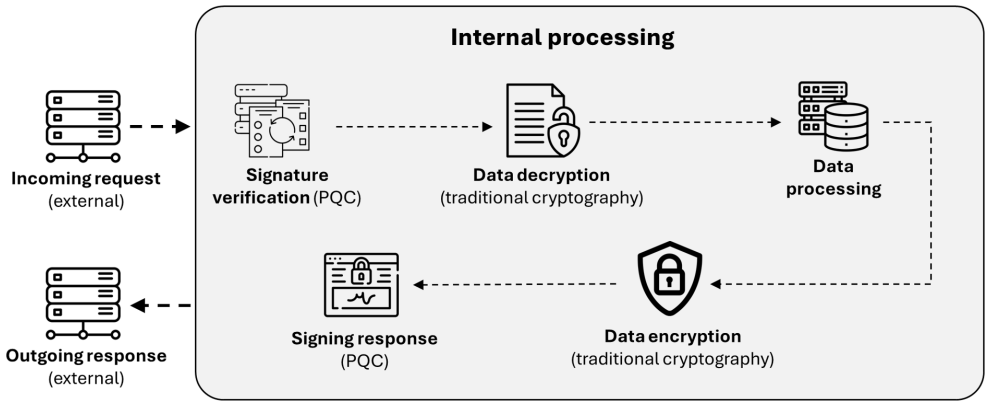
- **decryption,** which decrypts incoming messages to ensure they can be processed securely; and
- **encryption,** which encrypts settlement messages sent to participants.

Graph 4 highlights which microservices are PQC-enabled and which are implemented with traditional cryptography. Additional information can be found in Section A.4.

[29] See BIS (2023a).

[30] See BIS (2023a). It is argued that it is possible to make an AES, a specific traditional algorithm for encryption employed by FuSSE, quantum-resistant by increasing the size of the key it uses.

Cryptographic microservices Graph 4



Note: There are two PQC-enabled microservices and two microservices that employ traditional cryptography. The third microservice for the key pair generator is not included in this diagram.

2.7 Monitoring

A custom monitoring tool was developed for the PoC to identify bottlenecks and issues across each microservice as part of the end-to-end settlement process. This tool included "system views" for performance; "operator views" for information on microservice instances, balances and transactions; and a "business view" for balances and transactions. Additional information can be found in Section A.5.

3. Results and discussion

3.1 Test results

The PoC achieved its specific target goal of settling 10,000 TPS[31] as a means to demonstrate the achievement of its broader objectives of flexibility, scalability and security discussed in the next subsections. This was achieved across three stages (2,500, 5,000 and then 10,000 TPS). In each stage the solution was further optimised with computing power added as required.

Increasing throughput fourfold (from 2,500 to 10,000 TPS) required between two and a half and less than four times more resources (ie servers, memory, central processing units (CPUs), etc).

The project showed that it was possible to increase the number of settled TPS without an equivalent increase in computing power, even with more resources added for the more computationally intensive cryptographic microservices.

Infrastructure cost[32] did not scale linearly with performance.

This shows one of the benefits of the flexibility afforded by a microservices architecture. Additional information on the testing, the results achieved and the infrastructure used in this PoC can be found in Appendix B.

3.2 Discussion

3.2.1 Scalability and flexibility

The microservices-based architecture adopted in the PoC demonstrated both scalability, by effectively handling increasing loads through independent service scaling, and flexibility, achieved through a modular service design that enables independent updates, rapid feature extension and seamless integration of new components without affecting the overall system. Testing showed that by tuning parameters at both the infrastructure and microservice levels, the PoC could sustain transaction volumes that are typically higher than those of other IPS currently in operation, without requiring a proportional, linear increase in computing power. This efficiency highlights how computing resources could be allocated in a flexible way, with additional computing capacity directed specifically to more computationally intensive cryptographic microservices as required, without over-provisioning the entire system.

---

[31] See NPCI (2025) and Central Bank of Brazil (2025). Two widely used IPS systems are UPI (India) and Pix (Brazil). The former processes around 7,500 TPS and the latter around 2,500 TPS.

[32] Testing was conducted in the cloud-based Azure Kubernetes Service (AKS) provided by Microsoft.

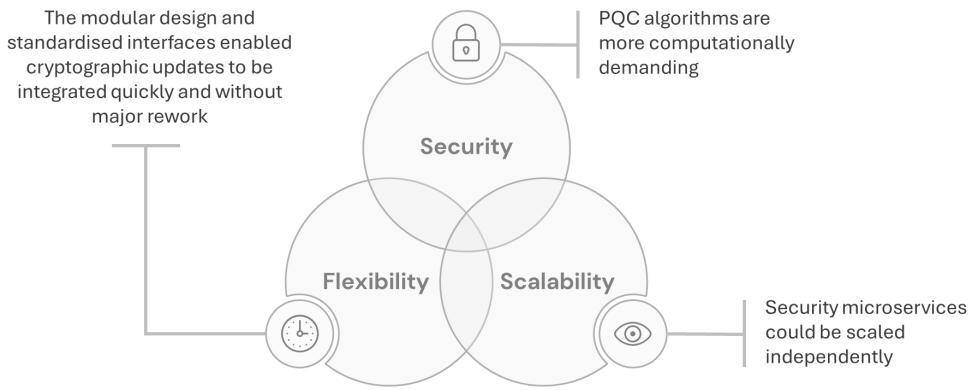
The architecture's modularity also proved valuable when adapting to design requirements. For example, to comply with the project's open source approach, one of the underlying database components had to be replaced and another was downgraded.

Given the isolation provided by a microservices design approach, such a change only affected the specific microservice that relied on that component, while the rest of the system continued to function without disruption. This showed how architectural modularity not only supports scalability but also simplifies system evolution and maintenance.

3.2.2 The interplay between architecture, security, scalability and flexibility

As described in Section 2.4 and in the first Project FuSSE release report,[33] the project tested PQC components to explore how a settlement engine could be designed in a quantum-resilient manner while maintaining scalability and flexibility (see Graph 5).

Security, scalability and flexibility — Graph 5



PQC algorithms are generally more computationally demanding, with larger keys and signatures that introduce additional latency and throughput challenges. In this context, the adoption of a microservices architecture proved to be advantageous for several reasons.

First, it allowed specific, security-related microservices, such as those responsible for signing, signature verification and encryption, to be scaled independently, limiting their performance impact on the wider system. This isolation made it possible to manage heavier cryptographic workloads while maintaining operational stability.

Second, the architecture provided cryptographic agility: because PQC standards are still evolving, a modular design enables individual components to be modified,

[33] See BIS (2024).

updated or replaced as new algorithms mature, without requiring major re-engineering of the entire system.

At the same time, the project underscored that the interplay between architecture and security introduces inherent trade-offs. In a distributed microservices environment, PQC overheads could multiply at every service boundary, increasing resource demands and potentially affecting system latency.

Nevertheless, modular design and PQC could reinforce one another if systems are engineered to anticipate the costs of transmitting and processing larger cryptographic payloads.

Scalability could help offset these impacts through the addition of compute nodes or dynamic load management, but choices must be made to balance performance, resilience and security.

The modular design and standardised interfaces employed in Project FuSSE further enabled cryptographic updates to be integrated quickly and without major rework.

In parallel, the use of technologies with broad community support ensured regular maintenance and upgrades.[34] As a result, when both the public key infrastructure (PKI) component and the cryptographic library released updates, integration and interoperability of these were easily and consistently achieved across the system.

Project FuSSE showed how settlement systems can remain secure, resilient and adaptable by decoupling cryptographic services from the core settlement logic, but it also showed that security, scalability and flexibility can at times pull in different directions.

Flexibility and crypto-agility could be embedded in the foundation of settlement systems, but this introduces operational complexity and performance trade-offs.

Looking forward, key design questions for next-generation FMIs could include how to balance flexibility with complexity, how to scale without compromising resilience and how to prepare for quantum-era risks without undermining today's reliability and trust.

As PQC standards and operational frameworks mature, coordinated progress across technology, regulation and governance will be essential to ensure that next-generation infrastructures remain both secure and adaptable.

3.2.3 Cryptographic agility and operational agility

The project highlighted the importance of cryptographic agility and how a microservices architecture could support this, particularly as PQC standards continue

[34] However, this reliance on community-driven ecosystems also introduces potential drawbacks, as development priorities and design decisions made by the community may not always be fully aligned with the specific requirements or long-term objectives of the user institution.

to evolve. This capability becomes essential to ensuring long-term security and trust. However, the project also underscored that cryptographic agility alone is not sufficient. It must be accompanied by operational agility, the institutional capacity to adapt governance, risk management and operational processes in line with evolving security requirements. This includes updating key management practices, certification frameworks, incident response mechanisms and oversight procedures in a coordinated and timely manner.[35]

3.2.4 Broader technical considerations

The Project FuSSE PoC explores a limited set of functionalities, and other considerations exist that were not in scope of the PoC.

The maturity and interoperability of the PQC ecosystem and integration with existing PKI and hardware security modules (HSMs) are important considerations.

Systems must also meet operational and resilience standards under the Principles for financial market infrastructures (PFMI), covering availability, fraud prevention and incident management.

Further technical aspects include liquidity and gridlock resolution, access and participation requirements, procedures for participant defaults, enhanced analytics and observability, cryptographic asset management, and incident response and recovery procedures.

[35] Operationalising PQC will depend on ecosystem maturity and integration with existing controls, including PKI profiles, key lifecycle procedures, interoperability testing and, where required, HSM support. Quantifying end-to-end impacts (eg payload size, bandwidth and CPU overhead across service boundaries) would support migration planning.

4. Conclusion

Project FuSSE showed that a settlement engine could be developed using a modular, microservices-based architecture that supports scalability, flexibility and quantum-resistant security. The PoC showed that such an approach enables the adoption of PQC measures while maintaining performance and operational stability. In particular, the architecture's modularity allows services responsible for cryptographic operations, such as encryption, decryption and signature verification, to be scaled horizontally and independently on demand, ensuring low latency and consistent throughput under high loads. This isolation of specific business and technical functionalities, such as cryptography, allows resource-intensive functions to be scaled without affecting other components, while also simplifying upgrades, maintenance and system evolution.

At the same time, the project underscored that these architectural benefits come with trade-offs. Microservices architectures introduce added complexity, requiring careful orchestration, monitoring and management of inter-service dependencies, while also broadening the potential attack surface. Project FuSSE further revealed that post-quantum algorithms, particularly digital signatures, could create additional computational and bandwidth overhead, yet these impacts could be effectively managed through scalable design and targeted resource allocation. Beyond cryptographic agility, the ability to adapt cryptographic algorithms as standards evolve, the project highlighted the need for operational agility, ensuring that governance, certification and incident-response processes evolve in step with technological change.

This PoC demonstrates architectural feasibility and provides design insights for central banks considering next-generation systems. However, any production implementation would require significant additional work, including operational resilience frameworks, security hardening, regulatory compliance validation and extensive testing under real-world conditions. The findings should be viewed as a foundation for further development rather than a deployment-ready solution.

References

Bank of Italy (2020): *The role of TIPS for the future payments landscape* https://www.bis.org/review/r201130c.pdf.

Bank for International Settlements (BIS) (2023a): *Project Leap: Quantum-proofing the financial system*, June, www.bis.org/publ/othp67.pdf.

——— (2023b): *Project Tourbillon: Exploring privacy, security and scalability for CBDCs – Final report*, November, www.bis.org/publ/othp80.pdf.

——— (2024): *A strategic framework for quantum-cryptography in FuSSE*, April, www.bis.org/publ/bisih_fusse.pdf.

Battula, S T (2025): "A comprehensive framework for evaluating the scalability and security of fintech web applications in a cloud-native environment", *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol 11, no 2, March, pp 1940–50.

CapGemini Research Institute (2025): *World Payments Report 2026 – The (not-so) silent takeover: Winning back merchants means playing differently*, September, www.capgemini.com/insights/research-library/world-payments-report/.

Central Bank of Brazil (2025): *Pix statistics*, www.bcb.gov.br/en/financialstability/pixstatistics.

Challa, K (2021): "Cloud native architecture for scalable fintech applications with real time payments", *International Journal of Engineering and Computer Science*, vol 10, no 12, December, pp 25501–15.

Christudas, B (2019): *Practical microservices architectural patterns: Event-based Java microservices with Spring Boot and Spring Cloud*, Apress, June.

Committee on Payments and Market Infrastructures (CPMI) (2014): *Cyber resilience in financial market infrastructures*, November, www.bis.org/cpmi/publ/d122.pdf.

——— (2022): *Improving access to payment systems for cross-border payments: best practices for self-assessments*, May, www.bis.org/cpmi/publ/d202.pdf.

Committee on Payments and Market Infrastructures and the Board of the International Organization of Securities Commissions (CPMI-IOSCO) (2016): *Guidance on cyber resilience for financial market infrastructures*, June, www.bis.org/cpmi/publ/d146.pdf.

Committee on Payment and Settlement Systems and the Technical Committee of the International Organization of Securities Commissions (CPSS-IOSCO) (2012): *Principles for financial market infrastructures*, April, www.bis.org/cpmi/publ/d101a.pdf.

Darbha, S, R Arora, C Minwalla and D Shah (2025): "A retail CBDC design for basic payments: Feasibility study", *Bank of Canada Staff Discussion Paper*, no 2025-9, June, doi.org/10.34989/sdp-2025-9.

European Central Bank (ECB) (2023): *A big future for small payments? Micropayments and their impact on the payment ecosystem*, August, www.ecb.europa.eu/pub/pdf/other/ecb.micropaymentsimpactonnpaymentsecosystem202308~bb92cda8ce.en.pdf.

Frost, J, P Koo Wilkens, A Kosse, V Shreeti and C Velásquez (2024): "Fast payments: design and adoption", *BIS Quarterly Review*, March, pp 31–44, www.bis.org/publ/qtrpdf/r_qt2403c.pdf.

National Institute of Standards and Technology (NIST) (2024): "NIST releases first 3 finalized post-quantum encryption standards", 13 August, www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards.

National Payments Corporation of India (NPCI) (2025): *Product statistics – UPI,* www.npci.org.in/what-we-do/upi/product-statistics.

Panetta, Fabio (2020): "The TARGET Instant Payment Settlement (TIPS) system", speech, 27 November, https://www.ecb.europa.eu/press/key/date/2020/html/ecb.sp201127~a781c4e0fc.en.html.

Tapia, F, M Mora, W Fuertes, H Aules, E Flores and T Toulkeridis (2020): "From monolithic systems to microservices: a comparative study of performance", *Applied Sciences*, vol 10, no 17, August.

Appendix A: Underlying technologies

A.1 Resource management

Managing multiple microservices and the increasing computational power required by scaling them is a challenge. When transaction volumes increase and microservices multiply to handle them, underlying technology is required to distribute, monitor and optimise the sources of this computational power (eg the best mix of servers). For this task, the PoC uses **Kubernetes.**[36] Kubernetes allows the overall capacity of the system to elastically grow or shrink by managing how the requirements of multiple microservices are met.

A.2 Communication between microservices

Communication between microservices is a second challenge. Microservices are part of a process, but they operate independently. Payment messages flowing between microservices in a linear queue could create bottlenecks. The PoC uses **Apache Kafka** as an intermediary for all transaction messages as they are worked on by the different microservices.[37] This event-driven way of communicating allows each microservice to pick up messages that need attention and then publish them once the task is complete (ready for the next microservice). This allows many microservices to work simultaneously without delay or duplication.

Kafka was selected due to its strengths in high-throughput messaging (millions of messages are handled per second), horizontal scalability (brokers and consumers scale independently), durability and fault tolerance (messages are persisted to disk and replicated), support for decoupled microservices in event-driven designs, and replay ability and auditability (events are retained for a configurable time window). Kafka acts as the central nervous system of the Project FuSSE architecture, enabling coordination and consistency while maintaining flexibility and decoupling for scalable and secure settlement systems.

A.3 Persistence

Settled transactions must also be recorded. The PoC uses **Apache Cassandra** as a longer-term database where every transaction is stored. Recording every transaction on a long-term replicable "golden copy" is a slow process.[38] Therefore, **Redis**[39]

---

[36] Kubernetes is an open source platform that automates the deployment, scaling and operation of containerised applications across clusters of machines.

[37] Kafka is a distributed event-streaming platform used to publish, store and process high volumes of messages between systems in real time.

[38] See CPMI (2014). A golden copy is an accurate picture of settled transactions in a payment system that can be used to recover in the event of a disaster.

[39] Redis is an in-memory data store commonly used as a fast cache or database for low-latency reads/writes.

supplements Cassandra[40] as a short-term, "in-memory" database for recording the balances held by senders and receivers.

Multiple synchronised instances of Redis can run in parallel just like a microservice to avoid any bottlenecks. The two databases complement one another. Cassandra acts as a large-scale filing system with lots of information and the potential for queries, whereas Redis can be updated very quickly with limited and simple information for balance queries by microservices.

In summary, Project FuSSE employs a dual-database architecture combining Redis and Cassandra to meet requirements for both real-time responsiveness and long-term durability:

- **Redis (in-memory/hot path):** Used for latency-sensitive data like account balances and recent transactions, offering sub-millisecond response times. It supports features like time-to-live (TTL)[41] commands, pub/sub and streams, and achieves horizontal scalability via Redis Cluster with high availability through Redis Sentinel. The in-memory ledger quickly updates sender and receiver balances in Redis. To speed up balance updates, "connection pooling" is used to reuse existing connections between the in-memory ledger microservice and Redis instances.
- **Cassandra (persistent/cold path):** Used for long-term, durable storage of historical data and system state. Its write-optimised engine ensures consistent performance under high ingestion loads, following an eventual consistency model. Cassandra acts as a large-scale filing system for full transaction details and potential queries, while Redis handles quick balance queries.

A.4 Cryptography

Many of the existing encryption standards used in financial and payment systems are vulnerable to future quantum computers.[42] The three cryptographic microservices utilise quantum-safe CRYSTALS (Cryptographic Suite for Algebraic Lattices) cryptography algorithms.[43] Specifically, the digital signature scheme implemented follows the post-quantum standard set by the National Institute for Standards and Technology (NIST),[44] namely Dilithium (ML-DSA (FIPS 204)).

This "lattice-based" cryptography is considered promising against future quantum computers because it relies on nondeterministic polynomial (NP)-hard

---

40 Cassandra is a distributed NoSQL database designed for high availability and fast writes at scale across multiple servers and locations.

41 Time-to-live (TTL) commands are used to determine the remaining time to live of a key that has an expiration set.

42 See BIS (2023b).

43 CRYSTALS is a family of PQC algorithms based on lattice math, designed to resist attacks from both classical and quantum computers (eg Kyber for key establishment and Dilithium for digital signatures).

44 See NIST (2024).

"shortest-vector" and "closest-vector" problems, which are exponentially difficult to solve efficiently even with known quantum algorithms. Due to the computational intensity of these quantum-safe standards, the number of signature microservices required to avoid bottlenecks during testing was five times that of any other service.

More information on the cryptographic aspects of Project FuSSE can be found in its first release report.[45]

A.5 Monitoring

To test the PoC, a monitoring tool was required. Many interfaces already exist to check system performance. However, in testing Project FuSSE's architecture, it was necessary to identify any bottlenecks or issues that might require better calibration or additional microservice instances. In other words, any monitoring tool had to show the performance across each microservice as part of the end-to-end settlement process.

A custom monitoring tool was built for the PoC. The tool contained "system views" to show performance and "operator views" and "business views" with information on balances and transactions. Prometheus[46] collects the metrics exposed by each microservice and Grafana,[47] querying Prometheus, creates visualisations of those metrics. Both Grafana and Prometheus are open source software. In accordance with the rest of the PoC's architecture, the custom monitoring tool was scalable and flexible, allowing for additional microservices and views to be easily added.

[45] See BIS (2024).

[46] Prometheus is an open source monitoring system that collects time series metrics (often by scraping endpoints) and supports querying and alerting.

[47] Grafana is a dashboarding and visualisation tool that turns metrics and logs into charts and operational views.

Appendix B: Testing and results

B.1 Environment

The settlement messages were representative, containing fictitious data, and were used to test the PoC using software that simulated multiple senders (eg 10,000 different accounts) sending messages to the system.

Each settlement message was in JSON file format[48] (which is efficient for data transmission), containing the sender account, the receiver account and the value of the transaction. Apache JMeter[49] was used to create multiple senders sending messages to the 10,000 different accounts used in testing.

B.2 Results

The PoC was tested across three phases, first reaching 2,500 TPS before reaching 5,000 and 10,000 TPS (Table 1).

FuSSE testing results — Table 1

Phase	TPS	M/services	Partitions	Pods	CPUs	RAM	Worker nodes[50]
1	2,500	2	12	20	14	12GB	2
2	5,000	3	24	32	21	18GB	3
3	10,000	5	75	54	35	30GB	5

In the first phase, the development team assigned each microservice a Kubernetes "pod" and increased them to handle additional messages.[51]

At 20 pods, the PoC processed 2,500 transaction messages every second end-to-end, using two worker nodes with 14 CPUs and 12 GB of random-access memory (RAM).

In the second phase, a goal of 5,000 TPS was set. To achieve this, the development team added more computing power, but also made three key changes to the PoC:

[48] JSON is a lightweight text format for representing structured data (key-value pairs and lists) that is widely used for data exchange between systems.

[49] Apache JMeter is a load-testing tool that simulates many users or clients to generate traffic and measure system performance.

[50] Worker nodes are the Kubernetes compute nodes that provide CPU and memory resources for running application pods.

[51] Kubernetes allows the overall capacity of the system to elastically grow or shrink by storing microservices in multiple pods that can be assigned computing resources from multiple sources and still be coordinated.

(i) Kafka was optimised for every microservice; (ii) Redis connection pooling was used; and (iii) additional microservice instances for decrypting transactions were deployed.

- To enable microservices to move messages faster, the Kafka connections between each one had to be individually configured. Kafka connects microservices by picking up messages that need attention and then publishing them once a task is complete. However, each microservice has a different task and speed at which that task is completed, so Kafka's connections to each microservice had to be adjusted to optimise how many messages were picked up and then available for next time.
- To speed up the balance updates from settled transactions, the connection between the in-memory ledger microservice and Redis was optimised. Where multiple instances of the microservice and Redis existed, connection pooling was employed to reuse existing connections instead of creating new ones. This sped up balance updates.[52]
- To avoid a bottleneck with decrypting messages, more decrypt microservices were added. Due to the quantum-safe encryption standards employed, the number of decrypting microservices required was five times that of any other service.

The additional microservices required more Kubernetes pods and computing resources. The number of pods increased to 32 and the cluster was scaled to three worker nodes, with 21 CPUs and 18 GB of RAM allocated to the workloads.

From this base, the third phase aimed at settling 10,000 TPS. No changes to the configuration were made in this phase. Instead, additional computing power was added to understand the cost of doubling performance. For this phase, premium virtual machines replaced the standard-issue ones used previously. This increased the CPUs and RAM available, as well as the "networking interfaces" available and the performance of the disc on which Cassandra recorded transactions.

The results successfully showed that the PoC could scale horizontally. Computing power did not need to be doubled in order to double performance (ie from 5,000 to 10,000 TPS).

For the final phase of testing, 35 CPUs with 30 GB of RAM were allocated to the workloads across five worker nodes. These results were also achieved continuously when 100,000 transactions were pushed to the system. Even in this scenario, 10,000 TPS were consistently settled with no issues.

[52] Cassandra did not require any equivalent optimisation.

Appendix C: Applicability

This appendix outlines the contexts in which FuSSE-type designs may be useful and the considerations that determine their suitability.

C.1 Applicability across FMI contexts

The potential relevance of the type of architecture explored in Project FuSSE varies according to the maturity and structure of national payment and settlement systems.

In emerging or smaller jurisdictions, FMIs often rely on legacy batch systems that have limited capacity to scale and may face resource constraints that make comprehensive system replacement difficult. In such settings, the modular characteristics of Project FuSSE could support incremental modernisation. Individual components could be introduced gradually, allowing for functions such as transaction validation or message routing to be modernised without needing to replace the entire infrastructure.

For mid-sized economies that operate conventional RTGS systems but have not yet introduced instant payment capabilities, the same design principles could provide a bridge to more flexible real-time settlement. A lighter-weight settlement layer built using microservices could connect to the existing RTGS, extending its functionality while maintaining established governance and operational arrangements.

In advanced economies with well developed IPS, current infrastructures are generally robust and high-capacity, but often remain complex to modify or upgrade. In such cases, the concepts explored in Project FuSSE may be most relevant at the component level, for example in introducing independent services for message handling or signature verification or in testing cryptographic agility in anticipation of post-quantum standards.

C.2 Settlement model and deployment environment

The Project FuSSE PoC simulated settlement in central bank money within an RTGS-type framework. It did not implement a specific payment scheme, but showed a potential core engine that could underpin various payment or FMI applications.

The PoC was deployed in a cloud-based environment to explore elasticity and resilience. This configuration served the purposes of the project and does not imply any policy preference for cloud deployment. The same architectural principles could be applied in private or hybrid infrastructures operated under existing oversight and resilience frameworks.

C.3 Scalability and resource considerations

The system's ability to process up to 10,000 TPS illustrates horizontal scaling potential rather than a performance target.

In smaller systems, the same architecture could operate efficiently at lower volumes; in larger or growing systems, capacity could be increased by running additional service instances. These results indicate that modular scaling could help align computing resources with transaction demand without major redesign.

C.4 Implementation factors

Institutions evaluating a Project FuSSE-type design may consider:

- **strategic objective:** whether the aim is to enhance capacity, modernise system architecture or strengthen cryptographic resilience;
- **integration approach:** whether to build new functionality alongside existing systems or to refactor selected components;
- **operational capability:** readiness to manage containerised and event-driven infrastructures securely; and
- **regulatory alignment:** ensuring the approach remains consistent with the PFMI, particularly on operational resilience and governance.

C.5 Summary

Project FuSSE represents a technical exploration of settlement engine design rather than a prescriptive model. Its architectural concepts, modularity, horizontal scalability and cryptographic agility may be adapted to different institutional settings according to scale, maturity and policy priorities.

The applicability of such approaches depends on each operator's objectives, technological environment and capacity to manage more distributed architectures within existing oversight frameworks.

Appendix D: Security learnings from the cryptographic microservices

Project FuSSE demonstrates how cryptographic functions can be modularised into independently scalable microservices (eg signing and signature verification) to support high throughput while maintaining cryptographic agility as post-quantum cryptography (PQC) standards evolve. In FMI contexts, such services may be expected to support two related but distinct security objectives: (i) authentication and integrity (ensuring that only authorised participants can originate messages and that messages are not altered) and (ii) non-repudiation (ensuring that participants cannot credibly deny having authorised actions, including in third-party dispute resolution).

A related security assessment of the cryptography microservices used in the PoC highlights that achieving these objectives, particularly non-repudiation, depends not only on algorithm choice but also on end-to-end protocol and evidence design. The learnings below illustrate considerations that may become important if adopting such architectural approaches.

This appendix summarises learnings from an assessment of the cryptographic microservices and message workflow used in the PoC.[53]

D.1 Learnings from workflow and encryption/signature design

What is signed matters for non-repudiation. In the illustrative PoC flow, messages are verified before decryption, implying that the signed object may be ciphertext. For use cases requiring non-repudiation and third-party verifiability, institutions may wish to consider ensuring that signed content remains readable and unambiguous to an independent verifier (eg by signing a canonical plaintext representation, or a well-defined commitment that can be verified without disclosure of secret keys).

Evidence retention is a functional requirement. Authentication-only designs may verify signatures and then discard them. Where non-repudiation is a requirement, durable evidence is typically needed, including signed messages, signatures, signer identities and certificates (often including the chain), retained in a protected archive that supports future verification. The PoC does not specify a signature-evidence archiving approach, highlighting a future design consideration, where evidence preservation could be treated as part of the settlement/audit design (eg embedded in ledger records or protected audit logs).

Time anchoring strengthens long-horizon assurance. Signer-generated timestamps can be disputed in scenarios such as key compromise or certificate revocation. Where evidentiary assurance must remain robust over long horizons,

[53] Many of these learnings could also relate to PFMI expectations on governance and operational risk management (including cyber resilience), but are discussed here as design learnings rather than as indicators of PFMI compliance.

institutions may wish to consider third-party timestamping of message hashes to strengthen verifiability without disclosing message content.

Non-repudiation depends on identity governance and certification policy. Non-repudiation is influenced by the trust model for identity binding (eg the independence and assurance of certification authority operations) and by certificate-policy details relevant to dispute contexts. This reinforces the report's broader conclusion that operational agility, governance, certification and oversight processes, must evolve in parallel with cryptographic agility.

Signatures do not secure the protocol on their own. Digital signatures are designed to be verifiable many times, therefore there is a risk signed messages can still be replayed. In asynchronous, high-throughput microservice environments, institutions may wish to consider protocol-level protections such as unique transaction identifiers recorded in the ledger and, where sequencing matters, sequence numbers or message-chaining mechanisms to detect replay, omission, or re-ordering.

PQC readiness involves standards and migration readiness, not only algorithm selection. As PQC standards mature, scaled implementations may need to consider standards-compliant algorithms (eg ML-DSA) where interoperability and institutional acceptance are required, and may explore hybrid approaches as a risk-management option for long-lived evidence.

D.2 Standard controls that remain essential in microservice-based settlement designs

Many security requirements are well established (eg secure service-to-service transport, authorisation controls, robust secrets management, and hardened key storage, potentially including HSM integration). In Project FuSSE type architectures, these controls would particularly important because microservices and messaging middleware increase the number of components, interfaces and operational dependencies that must be secured consistently.

D.3 Key considerations

Taken together, these learnings reinforce a key message from the report, that scaling and cryptographic agility introduce additional design choices and trade-offs that must be managed deliberately.

Institutions may wish to treat evidence design (what is signed, what is archived, how time is anchored and how identity is governed) and protocol integrity (replay and sequencing protections) as core requirements alongside throughput, modularity and PQC agility.

Contributors

Bank for International Settlements

Alonso Carrillo – Adviser

Baltazar Rodriguez - Adviser

Beju Shah – Head, Toronto Innovation Centre

Daniel Tavares de Castro – Adviser

Darko Micic – Cloud architect

Eleni Siskou – Legal Counsel

Henry Holden - Adviser

Holly Luo - Adviser

Jose Luis Lopes - Adviser

Juan Jose Lopez - Adviser

Karmela Holtgreve - Deputy Head, BIS Innovation Hub

Keerthi Nelaturu - Adviser

Miguel Diaz – Deputy Head, BIS Innovation Hub

Victor Rayado Perez – Adviser

Inter-American Development Bank

The Government of the Grand Duchy of Luxembourg (Financing)

Ana María Zárate Moreno – Senior Specialist, Financial Markets

Anderson Caputo Silva – Division Chief, Connectivity, Markets and Finance

Diego Herrera Falla – Principal Specialist, Financial Markets

Henrique Chitman – Senior Consultant, Financial Markets

Central Bank of Chile

Carolina Contreras Rodríguez - Software Engineer, Financial Technology Hub

Enrique González Vásquez - Senior Project Manager, Financial Technology Hub

German González Morris - Senior Project Architect, Financial Technology Hub

Jaime Pradenas Baeza – Head, Financial Technology Hub

Juan Ignacio Zucal - Senior Project Manager, Financial Technology Hub

Rafael Troncoso Aguirre - Senior Project Manager, Financial Technology Hub

Bank of Canada

Dinesh Shah - Senior Payments Technical Advisor, Banking and Payments Department

Francisco Rivadeneyra - Senior Research Director, Banking and Payments Department

Umar Faruqui - Director, Payments Policy and Strategy, Banking and Payments Department