Eurotoken

Robbert Koning
dept. name of organization (of Aff.)
Delft University of Technology
Delft, The Netherlands
R.M.Koning@student.tudelft.nl

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

In recent years, the European Central Bank (ECB) has increased its efforts in exploring the possibility of realising its own Central Bank Digital Currency (CDBC), the 'digital Euro'. The ECB has published various reports and resources that outline the desirability, necessity even, of such a project (i.e. [1], [2]). Most resources mention the decline of cash usage and corresponding rise of digital payments as a prominent reason for a digital Euro. According to reports published by De Nederlandsche Bank (DNB), the national bank of the Netherlands, the share of cash payments dropped from 56% in 2010 to 21% in 2020 [3] [4]. Cash is currently the only publicly accessible form of sovereign money [2].

Digital payments are made using services provided by private and/or foreign (non-European) actors. The money involved in these transactions is a liability of the respective actor and not a claim on a European central bank. A report published by ECB discusses a potential 'currency substitution'; a scenario where a new form of money that is not regulated by ECB becomes a viable medium of exchange and store of value. This currency substitution could reduce the effectiveness of ECB's monetary policy, harm market competition, and finally even threaten the European Union's strategic independence [1]. The private and/or foreign actors that are largely responsible for the fear of currency substitution are large corporations, big tech, and foreign central banks [2] [5]. In order to compete with these parties and make its CBDC attractive for mass adoption, the ECB has enumerated many requirements and wishes for its CBDC. First and foremost it is necessary for the value of digital Euros to be anchored to physical Euros. Moreover, the ECB wishes for its CBDC to enjoy beneficial cash-like features, such as being protective of citizens' privacy, being spendable in an offline setting, and being able to be remunerated at varying interest rates. For a full specification of the ECB's requirements and wishes for its CBDC, we refer the reader to the report [1].

Some of the demands and wishes mentioned by the ECB are difficult to realise individually and perhaps not even unifiable. The aforementioned report outlines multiple scenarios and analyses [1]. This research focuses on a scenario that attempts to closely resemble cash usage; physical Euros are mimicked by digital units of fixed, indivisible value ('tokens') and emphasis is placed on researching their spendability in an offline setting. Due to technical limitations however, some design choices were made that do not fall in line with the anonymity and decentralisation of cash usage. In particular the choice for a centralised validation process instead of peer-to-peer stands out. Please refer to Section ? For further elaboration. The transaction system that inspired this research was introduced in a work by Blokzijl and modified for this scenario [6]. This research contributes (1) an improvement in transaction throughput and scalability compared to Blokzijl's system (2) a performance analysis of various bottlenecks in the system to highlight its weaknesses and empirical upper performance bounds and (3) a fully rewritten and software-tested reference implementation.

II. Problem Description

The ECB desires a system with the benefits of both digital money and cash. To the best of our knowledge no system exists that combines these properties. Blokzijl's system and the system highlighted in this research face similar difficulties [6]. This is due to overlapping design decisions. We consider this acceptable within the limited scope of this and Blokzijl's research.

An important difference is that Blokzijl's system is balance-based and this scenario is token-based. A token-based system requires generation of tokens and a modified transaction protocol. The token generation process is described in Section ? and the transaction protocol in Section ?. A major implication of a token-based system is that multiple tokens need to be sent per transaction, comparable to how cash payments often require multiple notes and coins.

However, the challenges presented to both systems are more similar than they are different. Both systems require their currencies to be anchored to the price of the Euro. Blokzijl opted for a system where an exchange guarantees that 1 unit of their currency can at all times be bought or sold for 1 Euro. This is a commonly used practice to keep the value of an asset stable compared to another asset. It requires every unit of currency to be collateralized by 1 Euro. We saw no major limitations with regard to this approach in the scope of this

research, and decided to not further concern ourselves with exchanging currency.

Another challenge that both systems face pertains to the ECB's desire for its CBDC to be spendable without a network connection. In both decentralised and offline transaction systems it is non-trivial to verify whether parties still own the funds they want to spend and have not spent them before. We assume the reader to be familiar with this so-called 'double spending problem' [7]. To mitigate the impact of the double spending problem, Blokzijl's system leans on one or more validators to verify balances. These validators are trusted parties in the network and thus balance validation is not peer-to-peer but a centralised process. We opted for a validation system comparable to Blokzijl's; providing near-immediate finality and scalability at the cost of having to trust the network's central nodes. Different from Blokzijl however, validating nodes must keep track of individual tokens rather than account balances.

If validators are unreachable, Blokzijl's system allows transactions to be made in a peer-to-peer fashion, deferring finality until the proper validator is available again. In this period during which the system is offline, double spending can occur and can only be detected afterwards during the validation process. Though not ideal, it is in line with the design principles of Trustchain, the framework upon which Blokzijl's implementation was built, which also guarantees fraud detection but not prevention [8]. This research adheres to the same principles with regard to double spending.

From measurements it became apparent that Blokzijl's system's transaction throughput was not high enough to facilitate the needs of the Eurozone. Transactions were measured to be around ?. VISA is capable of processing 24000 transactions per second (self-proclaimed, [9]) and Alipay 544000 [10]. It is worth noting that the scale of these systems is massively larger than the evaluation done by Blokzijl, which results in skewed measurements. Section III takes care to fairly compare Blokzijl's system with this work.

III. Evaluation

In an attempt to improve Blokzijl's proof-of-concept implementation, we found that it was unsuited for major expansion and opted to rewrite our own reference implementation in Kotlin. To help identify bottlenecks and to have a proper frame of reference, we also performed a brief performance analysis of low-level functionality.

A. Setup

All results discussed in this section were obtained with the same setup, unless mentioned otherwise. Experiments were performed on a Lenovo Thinkpad with an Intel i5 CPU operating at 2.11 GHz and 8 GB of DDR4 RAM [BRON]. All experiments were performed 5 times.

B. UDP Throughput

In Figure 1, the setup's maximum attainable UDP throughput is shown for different configurations. The labels are structured as *locality*|*protocol*|*type*|*language*.

The *locality* column is either *IPC* for Inter-Process Communication (i.e. measurements performed on a single machine) or *LAN* for Local Area Network.

In all but one case, *protocol* is UDP. The *Pipe* entry refers to measurements obtained with a script that uses *named pipes* to transfer data on a single thread. It was included to give an estimated upper bound to how fast IPC could be with the current setup.

The *type* column is either *Reuse* or *Recreate* (where applicable). *Reuse* indicates that a UDP packet was constructed once, before measurements started, and reused. *Recreate* indicates that every sent packet was generated individually, causing more overhead due to additional required CPU usage and memory allocation. For *Pipe*, this label is not applicable.

Finally, the *language* column refers to the programming language that was used to implement the measurement script.

The results in Figure 1 are based on the elapsed time of receiving 100 megabytes of arbitrary data. In the *LAN* case this data was sent over a cable supporting 1 gigabit per second. Because the actual sent data was not important, lost UDP packets were not acknowledged and instead the sending process kept sending until 100 megabytes were received. The 'whiskers' in the plot entries denote the interquartile range of the results.

From Figure 1 it is evident that UDP throughput is reduced when data is sent over LAN. In that case the throughput drops to around 125 megabytes, which corresponds to 1 gigabit. From these results, we consider it highly unlikely that UDP throughput will become the system's main bottleneck.

C. Cryptography

We measured the performance of various cryptographic operations to ascertain the performance bounds of our setup. Figure 2 shows the achieved results for different operations. The labels are structured as *Operation*|*Type*|*Language*. The *Operation* is either *Sign* for signing data or *Verify* for verifying signed data, both using Ed25519 [BRON]. Both operations were performed using a Kotlin port of Libsodium [BRON] and made use of CPU parallelism. The *Type* is either *Packet* or *Bulk*; *Packet* referring to signing or verifying individual data packets of 1492 bytes, and *Bulk* referring to signing or verifying 100 megabytes of data. The parameters chosen for these operations were identical to those used in *Kotlin-ipv8*; private keys were set to be 512 bits; corresponding public keys to 256 bits; signatures to 512 bits. Note that the Eurotoken protocol disallows signing multiple tokens at once, thus *Bulk* processing tokens is not representative.

Figure 2 shows that data verification can be done at a rate of roughly 90 megabytes per second. Assuming 157 bytes to be verified per token, this limits the processing rate of our setup to roughly 630000 token transactions per second. In reality, this rate will be much lower due to various other operations that are necessary to process and validate a token.

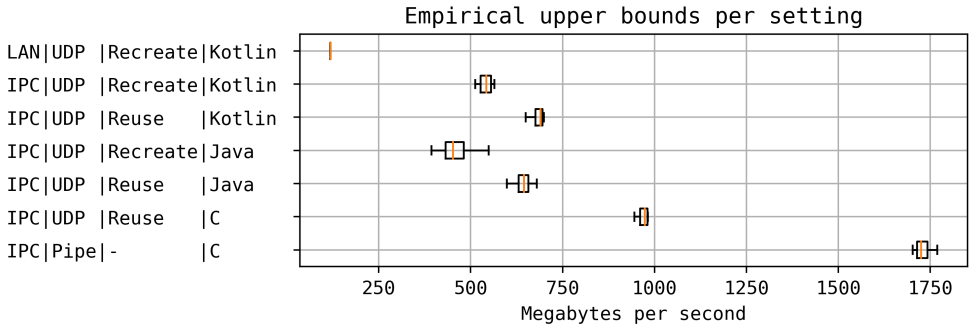

Fig. 1. Empirical maximum throughput for UDP in megabytes per second.

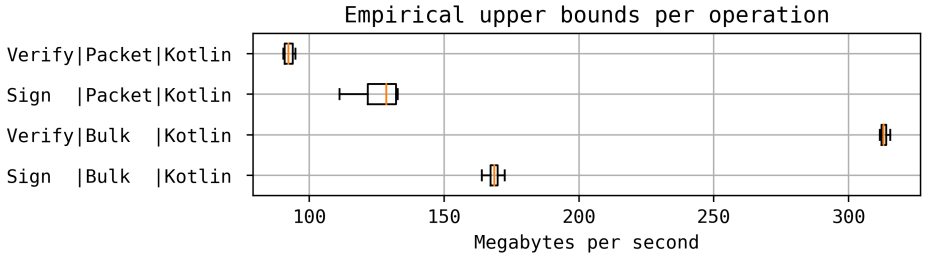

Fig. 2. Empirical maximum throughput for various cryptographic operations in megabytes per second.

References

[1] ECB, "Report on a digital euro," ECB, Tech. Rep., 2020. [Online]. Available: https://www.ecb.europa.eu/pub/pdf/other/Report_on_a_digital_euro~4d7268b458.en.pdf

[2] F. Panetta, "Public money for the digital era: towards a digital euro," 2022, keynote speech by Fabio Panetta, Member of the Executive Board of the ECB. [Online]. Available: https://www.ecb.europa.eu/press/key/date/2022/html/ecb.sp220516~454821f0e3.en.html

[3] N. Jonker, L. Hernandez, R. de Vree, and P. Zwaan, "From cash to cards: how debit card payments overtook cash in the netherlands," DNB, Tech. Rep., 2018. [Online]. Available: https://www.dnb.nl/media/kx1akmnb/201802_nr_1_-2018-_from_cash_to_cards_how_debit_card_payments_overtook_cash_in_the_netherlands.pdf

[4] N. Jonker and P. Zwaan, "Betalen aan de kassa 2020," DNB, Tech. Rep., 2020. [Online]. Available: https://www.dnb.nl/media/e34bo5zu/betalen_kassa_2020.pdf

[5] BIS, "G7 working group on stablecoins," BIS, Tech. Rep., 2019. [Online]. Available: https://www.bis.org/cpmi/publ/d187.pdf

[6] R. Blokzijl, "Eurotoken," Master's thesis, Delft University of Technology, 2021. [Online]. Available: http://resolver.tudelft.nl/uuid:132faae8-6883-454f-a8ce-94735340dce9

[7] U. Chohan, "The double spending problem and cryptocurrencies," *SSRN*, 2021.

[8] P. Otte, M. de Vos, and J. Pouwelse, "Trustchain: A sybil-resistant scalable blockchain," *Future Generation Computer Systems: the international journal of grid computing: theory, methods and applications*, vol. 107, pp. 770–780, 2020.

[9] VISA, "Visa acceptance for retailers." [Online]. Available: https://usa.visa.com/run-your-business/small-business-tools/retail.html

[10] J. Zhang, "How alibaba powered billions of transactions on singles' day with 'zero downtime'," South China Morning Post, 2019. [Online]. Available: https://www.scmp.com/tech/e-commerce/article/3038539/how-alibaba-powered-billions-transactions-singles-day-zero-downtime?module=perpetual_scroll_0&pgtype=article&campaign=3038539

Acknowledgment

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove the template text from your paper may result in your paper not being published.