2023 IEEE/IFIP Network Operations and Management Symposium (NOMS 2023) - Shortpaper

# XRP-NDN overlay: Improving the Communication Efficiency of Consensus-Validation based Blockchains with an NDN Overlay

Lucian Trestioreanu*, Wazen M. Shbair*, Flaviene Scheidt de Cristo*, and Radu State*
* University of Luxembourg, SnT, 29, Avenue J.F Kennedy, L-1855 Luxembourg
Email:{flaviene.scheidt, wazen.shbair, lucian.trestioreanu, radu.state}@uni.lu

NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium | 978-1-6654-7716-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/NOMS56928.2023.10154402

***Abstract*—With growing adoption of Distributed Ledger Technologies, their networks must scale while maintaining efficient communication for the underlying consensus and replication mechanisms. New content distribution concepts like *Named Data Networking* create opportunities to achieve this. We present and evaluate XRP-NDN overlay, a solution to increase communication efficiency for consensus-validation blockchains like XRP Ledger. We send consensus messages over different communication models and show that the chosen model lowers the number of messages at node level to minimum, while maintaining or improving performance by leveraging overlay advantages.**

***Index Terms*—XRP, blockchain, overlay, NDN, communication**

## I. Introduction

While different aspects of Distributed Ledger Technology (DLT) benefited from increased research attention, the underlying communication schemes, often relying on flooding mechanisms due to their one-to-many and many-to-many communication needs, received somewhat less attention. The task of scaling these networks comes with its challenges, one being to maintain or improve the efficiency and resilience of underlying communication. Each blockchain type has its specifics, and per our understanding, a one-size-fits-all solution is far from possible. For the Ethereum (ETH) blockchain based on Proof of Work (PoW), Gossipsub [1] was proposed to improve its communication layer, while [2] proposed a Named Data Networking (NDN)-based design for block propagation. The community effort was mainly directed towards PoW-type DLTs, with other types like consensus-validation blockchains receiving less attention. XRP Ledger (XRPL) is such a DLT [3], [4]. The size of its consensus protocol messages is small enough (around 0.5kB) as to not be a challenge. XRPL communication needs are rather near-real-time because: i) by design it aims to do real-time settlement [5], ii) it needs close synchronization, interconnection, and fault-free operation between validators [6], and iii) by implementation, in the 3-5s between 2 ledgers multiple consensus rounds and their message exchanges are held; in comparison, on BTC median block propagation time is 6.5s and mean=12.5s [7]. It is rather the dissemination of a high number of flooded messages and their processing at each node that challenges XRPL scalability, by increasing requirements for channel bandwidth, node hardware, and costs. If unaddressed, at some point this can result in network performance degradation. The problem can be stated: How can the performance burden due to a high number of messages induced by flooding at scale, be alleviated? Different approaches can be considered, e.g. improving the dissemination protocol, or external solutions such as overlays. We focus on decreasing the number of messages while deviating them through an NDN overlay where we can leverage specific properties to achieve this goal. NDN [8] is a type of content distribution network which instead of delivering packets to a given destination (IP), it fetches the data by name, offering *content caching* to improve delivery speed and reduce congestion, and built-in *multicast*.

Our contribution is two-fold: i) to our knowledge for consensus-validation DLTs there was no prior work on the topic, and ii) we propose, implement and evaluate multiple models to find the best one for the concrete case of XRPL.

## II. Background

**XRPL** is an open-source, permissionless, decentralized blockchain appreciated for transaction (tx) throughput (1500 tx/s), speed (tx settles in 3-5s), low fees and low energy consumption. The blockchain building process consists of a Byzantine Fault Tolerant *"Consensus"* [6] and a *"Validation"* stage. Consensus-wise a node only needs to communicate with those from its Unique Node List (UNL). *UNL* [9] is the set of nodes that a node does not necessarily consider to be all honest, but trusts not to collude. *Consensus* stages are: i) *"Open"*, when new tx's are received; ii) *"Close"*, when new tx are not accepted but *consensus* advances towards ledger close; iii) *"Establish"*, where nodes agree on effective close time and current tx set by exchanging *proposals* and adding or removing tx's; iv) *"Consensus reached"*, when nodes agree on the tx set to include in ledger; v) in *"Accept" phase* nodes apply the agreed tx set in canonical order and share the result, and vi) *"end round" state* meaning the round is finished and participants move to ledger *validation*. During *Validation*, validators share results as signed messages, called *validations*, containing the calculated ledger's hash to check if they obtained identical results. Then, they compare the results and declare the ledger validated IF enough trusted validators agree. Flooding is the main dissemination protocol, which ensures robustness and simplicity at expense of efficiency. Dissemination efficiency for main flooded data types: *Transactions* (Tx), *Proposals*, and *Validations* could be optimised.

Because today's Internet is rather used as an information distribution network, **NDN** [10], [11] *fetches* data *by name*. NDN distinguishes itself as follows: i) *Data* is named by application, and *Consumers* request it by *name* - a consumer-driven process; ii) *Data* is signed by *Producers* and can be verified by *consumers*; iii) Routers record data requests *(interests)* and erase it once received. As such, smart strategies can be used for forwarding, and loops eliminated. NDN offers *content caching* to improve delivery speed and reduce congestion, a *simpler configuration* of network devices, and *data-level security*. On NDN *Producers* create data while *Consumers* are *interested* to receive or *"consume"* it. Hence, the two packet types: i) *Interests* sent by *Consumers* ask *Producers* for data; ii) *Data* created by *Producers* is sent to *Consumers* in response to *Interests*. Other building blocks are *Content Store* which stores for some time data already seen to serve it immediately in case of new requests; *Pending Interest Table* stores unfulfilled *interests*; *Forward Information Base* helps packet routing.

## III. Design and Implementation

We chose *overlays* because their specific properties can be leveraged to achieve our goal without touching the application or underlay, which offers flexibility. Applications can process less messages by shifting some overhead to overlay, opt in/out of it, or fallback to p2p as backup. NDN was chosen as overlay because of its in-network caching which can lower the number of messages, and native multicast which should soon have mechanisms to reduce message duplicates [12]. Data can be disseminated on NDN in two ways: In the native pull-based approach, *consumers* request and receive data by name. In a push approach [13] data is sent over *interests* with multicast.

Aiming to decrease XRPL nodes' load, i.e. number of messages processed, we seek to answer following questions:

**Q1** On which models can we map XRP consensus to NDN?
**Q2** How do models compare each other and with baseline?

To answer Q1 we identified several NDN-specific communication models shown in Fig. 1 and a fourth one which is an optimized multicast model, called *"piggybacking"*.

1) *"Polling"*: Each validator associates to each validation an increasing *"sequence number"*. Interested nodes send periodic interests to fetch last *"sequence number"*. If sequence unchanged, they do nothing; if sequence increased, they ask for the new validation. As the interval between ledgers on XRPL is 3-5s, we chose a 200ms polling interval to ensure we don't delay much the propagation of validations.

2) *"Announce-pull"*: Validators having created a new validation send a multicast interest to let all nodes know the sequence of their new validation. Interested nodes pull the validation with the given sequence.

3) *"Advanced-request"*: Because on XRPL *Consumers* know the identity of originating *Producers* (UNL validators), and because the interval between validations is generally 3-5s, the announcement of a new validation can be considered made before the validation is produced. The time required to forward interests to source is eliminated by proactively requesting validations in advance, and data is served as soon as created.

NDN NOMS MODELS ALL.pdf

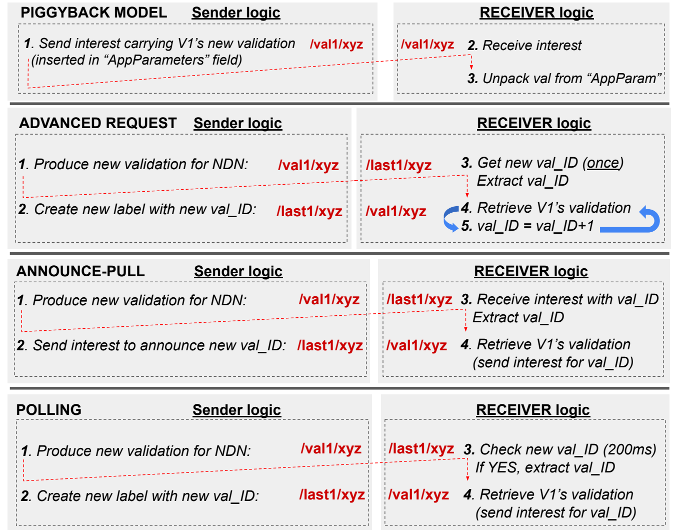



Fig. 1: Assessed NDN dissemination models

4) *"Piggyback"*: Validators encapsulate validations in *Interests* (*"appParameters"* field) and send them with multicast to all nodes. While this amounts to broadcast, it can be advantageous because: i) number of messages at XRPL application level (M1) is lower than baseline, while we also obtained some improvement for M3; ii) current work to decrease number of duplicates for NDN multicasting [12], can further improve efficiency; iii) on XRPL the multicast can be sent only to specific nodes interested to hear only messages from validators on their UNL. Compared to announce-pull it reduces the number of messages on overlay because for disseminating a validation, only the Interest is sent. It can also help latency-wise in some cases (no two way request-response): in the pull approach data caching can generally help latency only when multiple nodes request same data on same path.

To answer Q2, we define the following metrics:

**M1** *XRPL node load*: How do the models compare with regard to the number of validations in/out of a node?
**M2** *Network load*: How models compare each other and with baseline regarding the number of messages and bytes travelling the overlay to send a validation to all nodes?
**M3** *XRPL network stability*: How do models affect the inter-arrival time between validations?

We also analysed **M3** for UNL validators in production XRPL network which gave us real-life behavior information.

## IV. Experimental Evaluation

For evaluation we used a real (lab) testbed and the production network. The *baseline* was original *xrpl_v1.7* [14], which required significant effort to integrate [15] NDN. We used *NDNts* library [16], [17] for the overlay. Experiments were performed on three topologies, shown in Fig. 2, which can reveal if topology influences performance. *Star* topology is seven NDN nodes disposed in star, where the three edge nodes

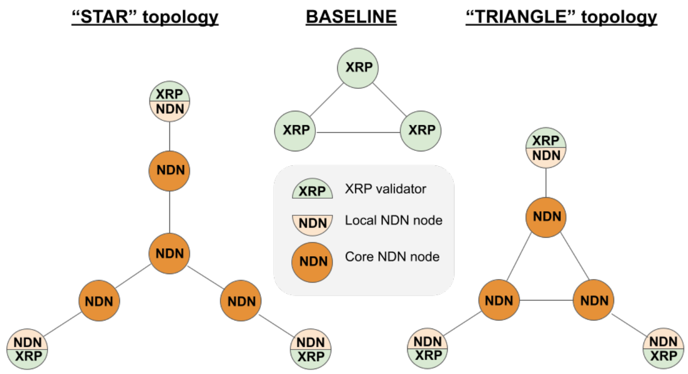


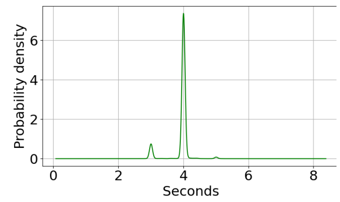

Fig. 2: The experimental topologies

are also XRP nodes. The central node is the most stressed traffic-wise so it could potentially be a bottleneck. The *triangle (tri)* topology is six NDN nodes in triangle. The three edge nodes are also XRP nodes. This one is more balanced: the three middle nodes share traffic more fairly. *Baseline* is a full mesh of three unmodified XRP nodes which is a natural choice for a fair comparison with the other topologies: at XRP logical level (message-wise) it is the closest equivalent to the others.

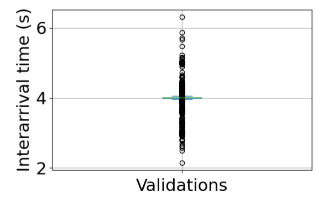
Collection of metrics: **M1**: For the unmodified XRPL, *Rippled Monitor* [18] and *Grafana* collected the number of validations and bytes in/out of a node. For the modified XRPL, we used our tool. **M2**: *Vnstat* [19] and *Tshark* [20] counted bytes/packets at machine NIC level. **M3**: We parsed XRPL logs for validation inter-arrival times. To improve figures readability we plot in *orange* the *rolling mean* ($rm$) over the previous 20 data-points (generally over 1-2 min, depending on interarrival times); in *green*, the $rm(20)$ plus 2 times the *rolling standard deviation* ($rSTD$) computed over the same 20 data-points: $rm(20) + 2 * rSTD(20)$; and in *red*, the same $rm(20)$ from which we substract 2 times the $rSTD(20)$, i.e.: $rm(20) - 2 * rSTD(20)$. **Results** obtained are discussed below:

*1) Production XRPL:* We deploy an XRP node on the live network to listen for validations from official UNL validators. We record the first unique validation from each of them, and drop duplicates. We do not collect **M1, M2** because a fair comparison is impossible for these metrics: such topology, number of nodes, and real-life internet can not be recreated in lab. Under **M3**, while generally validations were spaced at 3-5s (mean=3.92s; median=4s; quantile(0.25)=3.98s; quantile(0.75)=4.02s, as in Fig. 3c, 3a, 3b), some validators showed somewhat different behavior, not presented for space reasons.

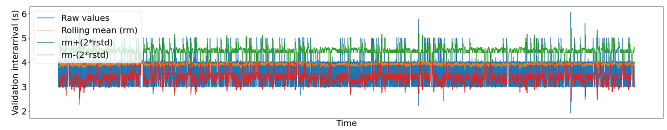
*2) "Baseline":* From one of the nodes, we record the intervals between the first arrived unique validations from other validators and drop duplicates. **M1**: We compute the ratio of validations in+out to ledgers created: on average 7.34 validations go in/out of a node to build a ledger, and 17845 in 2 hours. Regarding **M2** *Tshark* recorded 14420 packets in 10 min and *Vnstat* 58kbit/s in 5 min, while under **(M3)** interarrival times are spaced sharply at 3s with mean, median and quantiles around 3s, as in Fig. 4a, 4d.




(a) Pdf: validation interarrival time

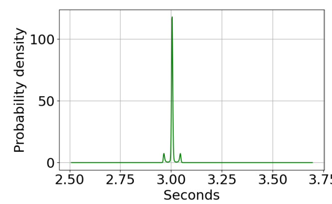



(b) Validation interarrival time

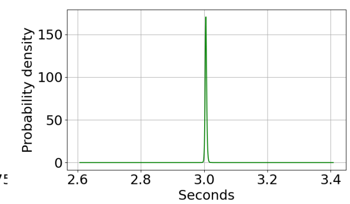



(c) Time series: validation interarrival time

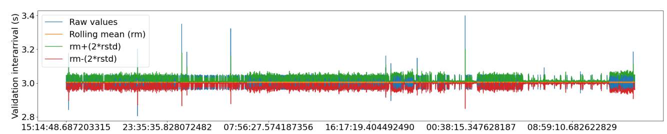
Fig. 3: Typical validation interarrival time on XRPL *livenet*



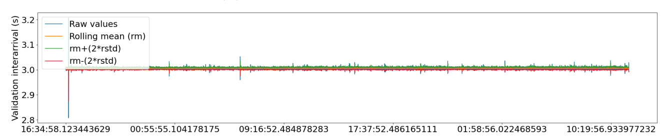

(a) Pdf: *baseline*




(b) Pdf: *piggyback* (*tri* topology)




(c) Time series: baseline




(d) Time series: piggyback model

Fig. 4: Validation interarrival time: *piggyback* versus *baseline*

*3) "Polling":* was the first model, tested on the *tri* topology, mostly to see how XRPL and NDN work together. We consider network stability (M3) eliminatory. So because it performed worse than the *baseline* and *piggybacking* under M3 (Fig. 5a), and because of the high number of overlay messages due to polling, we didn't evaluate further. However it can be improved e.g. to use *adaptive* polling intervals.

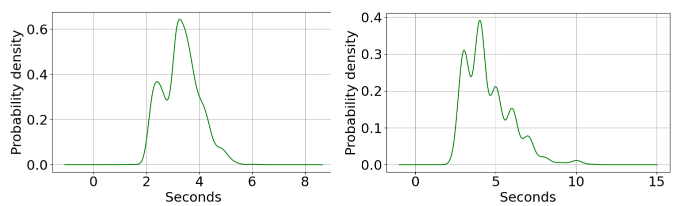
*4) "Announce-pull"* was the second model, tested on both topologies as an improvement to *"polling"*. Fig. 6 shows *tri* was better, without approaching *baseline* regarding M3 ($rm$, $rSTD$). Fig. 6, 5a show *announce-pull(star)* topology is better than *polling* under **M3**. We didn't collect M1, M2 because the model was worse than *baseline* and *piggybacking* (Table I).

*5) "Advanced-request":* **M3** for the *tri* topology (Fig. 5b) was not satisfactory so this model was not investigated further.

*6) "Piggyback":* Under **M1**, there were three validations

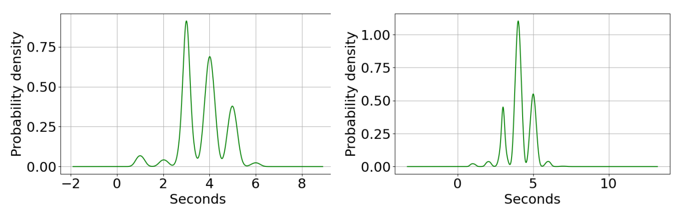
TABLE I: Experiments summary

| Model | Topo | Val inter-arrival time | | | XRP node load | NIC load | | Content Store (rates / min) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | q(0.25) | q(0.5) | q(0.75) | vals in+out/ledger | avg bitrate (5min) | pkt/10min | misses | hits | entries |
| Baseline | tri | 3.00 | 3.00 | 3.00 | 7.34 | 59kbit/s | 14420 | N/A | N/A | N/A |
| Adv-req | tri | 3.00 | 4.00 | 5.00 | not collected | 20kbit/s | 11800 | not collected | | |
| Polling | tri | 2.95 | 3.48 | 4.52 | not collected | | | | | |
| Announce | star | 3.00 | 3.86 | 4.21 | not collected | | | 170->790 (2h) | 0 | 887->1520 (2h) |
| Pull | tri | 3.86 | 4.07 | 4.84 | not collected | | | 900->1500 (2h) | 0 | 190-785 (2h) |
| Piggyback | tri | 3.00 | 3.00 | 3.00 | 3 | 80kbit/s | 13700 | 785 (flat) | 0 | 65 (flat) |




(a) Pdf: polling (tri) (b) Pdf: advanced-request (tri)

Fig. 5: Validation interarrival time: *Polling*, *Advanced-request*




(a) Pdf: *star* topology (b) Pdf: *tri* topology

Fig. 6: Validation interarrival time: *Announce-Pull model*

in+out of the XRP node per ledger which is 2.44 times better than baseline (7.34 validations). Probability distribution plots also show improved performance over baseline. For **M2**, *tshark* recorded 13713 packets in 10 min, and *vnstat* 80kbit/s over 5 min. **M3** is better than baseline (Fig. 4d,4c). Evaluation was done on the *tri* topology.

As per Table I summary, the *most suitable solution* is the encapsulation of *validations* in *Interests* disseminated with multicast (goal is to minimise the number of messages, and the ratio *piggybacking/baseline* is 3/7 under M1). The model improves over baseline as shown by comparing interarrival times, while ensuring robust dissemination and low latency.

## V. Related Work

Work to optimise protocols efficiency includes: temporarily "squelching" [21] some peers, not yet in production; Erlay [22] reduces bandwidth by 84% but increases latency by 2.6s; Perigee [23] focuses on propagation delay but not message number; *GossipSub* [1] improves communication of PoW-ETH; *Epidemic Broadcast Trees* are embedded on a gossip-based overlay in [24], while *Splitstream* [25] evenly distributes between nodes the load to forward messages.

Overlays are proposed to improve DLT messaging in [26]; *BoNDN* [13] proposes tx dissemination for Bitcoin through pushing over NDN *interests*, and subscribe-push for blocks. This is challenged in [2] for using NDN multicast: it is doubtful if in practice NDN nodes would enable multicast for given labels. XRPL has known-in-advance validators (UNL) so this is not problematic. A solution to propagate tx and blocks for PoW-ETH is proposed in [2]. In our opinion the needs of consensus-validation DLTs are fairly different from PoW-DLTs', to require separate consideration: size of XRPL consensus messages is much smaller than ETH blocks, and XRPL uses UNLs where from a consensus perspective a validator needs only receive messages from nodes in its UNL. Also on NDN data can be signed and dated by *producer*, which on XRPL is known (UNL validators), making some attacks discussed in the paper not applicable for XRP-NDN overlay. ETH P2P is used to broadcast new blocks' creation, then they are pulled on NDN. On XRPL the challenge consists of a very large number of messages - result of flooding at scale, which need to be minimised, and in this work we search and propose a paradigm suitable to XRPL. Data sync is dismissed by [2] for various reasons including security. While for XRPL validations the sync vector can be easily constructed, we agree sync was not designed for Byzantine environment, and could add unnecessary traffic hindering scalability.

## VI. Conclusions and Future Work

XRPL flooding mechanism lacked peer-reviewed research. In this paper we investigate how messaging can be optimised using NDN, a promising candidate because of its well researched and optimised caching and dissemination mechanisms. We propose multiple mapping models for message dissemination and investigate the advantages and disadvantages of each model according to the needs of consensus-validation blockchains. Similar to *validations*, *proposals* can also use *piggybacking*. For fast *tx* propagation, so that is included in the earliest possible ledger (case of high frequency trading), tx could use same model. Attacks such as poisoning may require mitigation such as in-flight verification, auditing, or node scoring. Experimentation was limited to the scenarios and topologies reported. We plan to test real-life scenarios to further assess robustness and security, and also a cost analysis.

## Acknowledgment

This work is supported by the Luxembourg National Research Fund through grant PRIDE15/10621687/SPsquared. In addition, we thankfully acknowledge the support from the RIPPLE University Blockchain Research Initiative (UBRI) for our research.

## References


[1] D. Vyzovitis, Y. Napora, D. McCormick, D. Dias, and Y. Psaras, "Gossipsub: Attack-resilient message propagation in the filecoin and eth2.0 networks," https://arxiv.org/abs/2007.02754, 07 2020.

[2] Q. T. Thai, N. Ko, S. H. Byun, and S.-M. Kim, "Design and implementation of ndn-based ethereum blockchain," *Journal of Network and Computer Applications*, vol. 200, p. 103329, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804521003143

[3] B. Chase and E. MacBrough, "Analysis of the XRP ledger consensus protocol," *CoRR*, vol. abs/1802.07242, 2018. [Online]. Available: http://arxiv.org/abs/1802.07242

[4] D. Schwartz, N. Youngs, and A. Britto, "The ripple protocol consensus algorithm," 2014.

[5] C. Ma, Y. Zhang, B. Fang, H. Zhang, Y. Jin, and D. Zhou, "Ripple+: An improved scheme of ripple consensus protocol in deployability, liveness and timing assumption," *Computer Modeling in Engineering & Sciences*, vol. 130, no. 1, pp. 463–481, 2022. [Online]. Available: http://www.techscience.com/CMES/v130n1/45706

[6] I. Amores-Sesar, C. Cachin, and J. Mićić, "Security analysis of ripple consensus," 2020. [Online]. Available: https://arxiv.org/abs/2011.14816

[7] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE P2P 2013 Proceedings*, 2013, pp. 1–10.

[8] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, and L. Zhang, "A brief introduction to named data networking," *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, pp. 1–6, 2018.

[9] P. Todd, "Ripple protocol consensus algorithm review," online, 05 2015, accessed: Jan. 2023. [Online]. Available: https://cdn3.baserank.io/reviews/9035a989-c2cd-4b5b-85a7-219abbed0900_RippleProtocolConsensusAlgorithmReview.pdf

[10] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, and L. Zhang, "A brief introduction to named data networking," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, 2018, pp. 1–6.

[11] "Named data networking," online, accessed: Jan. 2023. [Online]. Available: https://named-data.net/

[12] S. Dulal and L. Wang, "Adaptive duplicate suppression for multicasting in a multi-access ndn network," in *Proceedings of the 9th ACM Conference on Information-Centric Networking*, ser. ICN '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 156–158. [Online]. Available: https://doi.org/10.1145/3517212.3559480

[13] J. Guo, M. Wang, B. Chen, S. Yu, H. Zhang, and Y. Zhang, "Enabling blockchain applications over named data networking," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[14] XRP Ledger Foundation, "rippled," online, 02 2021, accessed: Jan. 2023. [Online]. Available: https://github.com/XRPLF/rippled/blob/develop/RELEASENOTES.md

[15] F. Scheidt, W. Shbair, L. Trestioreanu, and R. State, "Flexipipe," online, 07 2022, accessed: Jan. 2023. [Online]. Available: https://github.com/FlavScheidt/sntrippled

[16] J. Shi, "Ndnts: Named data networking libraries for the modern web - codebase," https://github.com/yoursunny/NDNts, accessed: July 2022.

[17] J. Shi, "Ndnts: Named data networking libraries for the modern web - homepage," https://yoursunny.com/p/NDNts/, accessed: July 2022.

[18] M. Bhandary and R. Zhang, "Rippledmon," online, 04 2021, accessed: Jan. 2023. [Online]. Available: https://github.com/ripple/rippledmon

[19] T. Toivola, "Vnstat," online, accessed: Jan. 2023. [Online]. Available: https://humdi.net/vnstat/

[20] The Wireshark Foundation, "tshark," online, accessed: Jan. 2023. [Online]. Available: https://www.wireshark.org/docs/man-pages/tshark.html

[21] G. Tsipenyuk and N. D. Bougalis, "Message routing optimizations, pt. 1: Proposal & validation relaying," online, 03 2021, accessed: Jan. 2023. [Online]. Available: https://xrpl.org/blog/2021/message-routing-optimizations-pt-1-proposal-validation-relaying.html

[22] G. Naumenko, G. Maxwell, P. Wuille, A. Fedorova, and I. Beschastnikh, "Erlay: Efficient transaction relay for bitcoin," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 817–831. [Online]. Available: https://doi.org/10.1145/3319535.3354237

[23] Y. Mao, S. Deb, S. B. Venkatakrishnan, S. Kannan, and K. Srinivasan, "Perigee: Efficient peer-to-peer network design for blockchains," in *Proceedings of the 39th Symposium on Principles of Distributed Computing*, ser. PODC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 428–437. [Online]. Available: https://doi.org/10.1145/3382734.3405704

[24] J. Leitao, J. Pereira, and L. Rodrigues, "Epidemic broadcast trees," in *2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007)*, 2007, pp. 301–310.

[25] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream: High-bandwidth content distribution in cooperative environments," in *Peer-to-Peer Systems II*, M. F. Kaashoek and I. Stoica, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 292–303.

[26] L. Trestioreanu, C. Nita-Rotaru, A. Malhotra, and R. State, "Spon: Enabling resilient inter-ledgers payments with an intrusion-tolerant overlay," in *2021 IEEE Conference on Communications and Network Security (CNS)*, 2021, pp. 92–100.