nahmii

Commercially-Focused Ethereum Scaling

Mark Briscombe, mark@hubii.com
Jens Ivar Jørdre, jensivar@hubii.com
Morten Fjeldstad, morten@hubii.com
John Derbyshire, john@hubii.com
Jacobo Toll-Messia, jacobo@hubii.com

August 14, 2019

Contents

1 Preface

Blockchain has been widely heralded as the next great innovation for businesses, with a forecasted value-add of over $3 trillion USD by 2030 (Gartner). It is therefore somewhat surprising that, as of mid-2019, there are currently no major commercial applications built on the technology. This disconnect between promise and delivery is explained by blockchain's inability to scale to meet commercial needs; in terms of throughput, neither Bitcoin or Ethereum can process more than 15 transactions per second. Before today, businesses have been forced to choose between security and performance; that changes with the first commercially-viable scaling solution for blockchains: **nahmii**.

The origins of the nahmii project date back to 2017, when Norwegian startup hubii AS decided to move their existing content aggregation business to the blockchain. Choosing to build on Ethereum, hubii understood that a scaling solution would be required as Ethereum alone could not meet our performance needs. After careful evaluation of the solutions in development at the time, it became clear that no proposed Ethereum scaling solution would deliver what hubii required: a protocol capable of handling millions of transactions per second, with predictable low fees, minimal latency and instant transaction finality.

In early-2018, hubii began work on the nahmii project (then known as 'striim'); an Ethereum-based scaling solution capable of handling commercial-scale blockchain applications. Hubii chose to build on Ethereum in order to leverage the technical excellence of the platform, specifically the ability to utilise smart contracts. This approach provides maximum flexibility for the future of the protocol, including the possibility of porting nahmii to both Bitcoin (via RSK) and Libra. Additionally, nahmii will also serve as a bridge providing interoperability between blockchains.

Development of the protocol moved swiftly throughout mid-2018, with the original version of nahmii's white paper published in June 2018. At the same time, nahmii's smart contracts were deployed to the Ropsten test network. Following extensive testing, nahmii was deployed to the main Ethereum network in four stages starting in December 2018. The protocol has been fully live since March 2019 and includes the ability to make deposits, send payments, settle accounts and withdraw. Despite starting long after our competitors, we have succeeded in delivering the first ever commercially-viable scaling solution for blockchain.

Now, following significant further development of the protocol, we have chosen to update nahmii's white paper to reflect the progress made in the last year. As with the original document, the purpose of this white paper is to offer an approachable insight into the nahmii protocol. Our aim is to use non-technical language where possible, although some basic knowledge of blockchains and Ethereum is required to understand everything. Similarly, the examples set out in this paper - notably relating to fraud detection - do not cover all eventualities.

All of nahmii's smart contracts are public, as our many of hubii's GitHub repositories relating to the protocol and its development tools. Given the public nature of this code, we consider the nahmii protocol itself to represent its own technical documentation.

2 Partner Words

“ The nahmii system addresses a real-world use case: incentivization of producing information goods. For example, the Bugmark project addresses market failures in software incentivization by trading futures contracts on the status of issues in a software issue tracker. This enables not only developers, but also code reviewers, testers, and managers, to overcome market failures that have historically resulted in software being produced at a quality level below that desired by either developers or users.

In order for futures contracts to fulfill their promise as a new form of software incentivization, we require

- low transaction costs, especially across jurisdictions (futures contracts are a way to trade information goods, not compensate labor, so can have extremely low transaction costs if the platform supports it)
- ease of deposit and withdrawal, even of small amounts (such as would be generated by bug triage, code reviews, or other low-overhead ’meta’ tasks)
- quick results for typical requests, in order to facilitate what might be multiple trades per issue for large numbers of related issues.

Where both conventional payment platforms and existing cryptocurrencies can satisfy some of our requirements, it appears that nahmii can address all of them and help facilitate a new type of market for software quality incentivization. ”

Don Marti, Strategist for Mozilla and advisory board member for Incentives Research

3 Introduction to nahmii

3.1 What is nahmii?

nahmii is a second layer scaling solution for the Ethereum blockchain. Second layer solutions are designed to handle much greater transaction volumes than the main Ethereum network; they do this by moving the majority of the processing off-chain. These off-chain transactions are then enforced by the security constructions on the Ethereum main chain, which acts as the arbiter of all disputes.

All transactions using nahmii will be processed initially by hubii, who will act as the Operator of the protocol. Importantly, the security of nahmii does not rely on users trusting hubii; the system has been designed to be trustless.

3.1.1 driips

Today, the nahmii protocol can be used to make off-chain payments in Ether and a range of supported ERC-20 tokens. Soon, nahmii will also be able to process trades (atomic swaps) between supported currencies. We use the term ‘driip’ to refer to any off-chain transaction within nahmii that can result in a change of state on the Ethereum main chain. Please note that driips are not limited to payments and trades; we anticipate that nahmii will support many different driip types in the future.

3.1.2 nahmii Tokens (NII)

The nahmii protocol will be tokenized, with almost 100% of transactions fees paid to token holders (the remainder goes into security bonds). There are a total of 120 billion nahmii tokens, known as NII, which are held in time-locked contracts. These contracts release 1 billion tokens per month, which are then distributed according to the allocations set out below. 50% of NII tokens will be airdropped to HBT token holders through the nahmii protocol directly.

Token holders are rewarded for playing an important role in the security of nahmii. Only NII tokens are accepted for staking into nahmii’s Data Availability Oracle (discussed later in this paper). Token holders are expected to monitor, validate and actively protect the protocol in this way.

3.1.3 hubii Tokens (HBT)

HBT is the native token of the hubii ecosystem, which was created in September 2017. Its function remains unchanged by the introduction of nahmii. Note that 50% of NII tokens will be airdropped to HBT holders on a proportional basis each month, this is discussed in further detail below.

3.2 Foundation Model

nahmii will be governed in accordance with a foundation model, whereby members are responsible for the efficient and reasonable management of the protocol. All members, hubii included, will be equal partners with the same associated rights, privileges and responsibilities. Members will be geographically distributed and leading companies across diverse industries, ensuring that the Foundation itself will be decentralised in nature.

Members are required to play an active role in monitoring, validating and protecting the protocol. Their role also includes a commitment to building nahmii-based solutions and the provision of nodes to assist protocol operation. The Foundation might be required to ensure the responsible divestment of hubii's large holding at an appropriate point in time. This divestment is part of a broader plan to guarantee a plurality of nahmii token holders, thus making the protocol more resistant to a 51% attack.

4 nahmii Fundamentals

4.1 State Channels

nahmii can be thought of as a multi-party state channel. A state channel is the name given to one general purpose off-chain scaling solution for Ethereum; it is essentially the exchanging of signed messages between users off-chain, which allows for eventual on-chain settling of the final state at a later point.

This is best explained by way of an example: Alice wants to pay Bob 1 token per tweet for 1,000 tweets. Using Ethereum, she could theoretically send 1,000 on-chain transactions to Bob of 1 token each time he tweeted. However, if everyone did this, the Ethereum network would rapidly become congested, transaction fees would rise and Alice's total cost to send those tokens would be punitively high. We can improve this situation by using state channels. In the simple case whereby Alice and Bob both trust each other, Alice could send an email containing a signed Ethereum transaction for 1 token after tweet 1 and a second email with a 2 token transaction after tweet 2. This would go on until she sent a 1,000 token signed transaction after tweet 1,000, at which point Bob could then send the transaction from the last email to the network and receive 1,000 tokens. This example is less relevant for nahmii as Alice and Bob trust each other; here, Alice could simply send the payment in advance of the tweets, trusting that Bob would deliver on his side of the bargain. Regardless, this example serves to highlight some of the major issues which would need to be solved in the case of Alice and Bob mistakenly trusting one another, they include:

- Bob sending all of the transactions he received from Alice to the network. In this case, he can potentially steal $1{,}000 + 999 + 998 + \ldots + 3 + 2 + 1 = 500{,}500$ tokens from Alice, assuming her balance was high enough
- Alice waiting until Bob has sent 1,000 tweets before moving her tokens to another address. When Bob tries to claim his 1,000 tokens by sending the final signed transaction to the network, there are no funds in Alice's account to pay him. Alice is effectively getting 1,000 tweets for free

State channel constructions become progressively more complex as additional security provisions are added to protect users. Similarly, the move from unidirectional payments to bidirectional payments and, ultimately, more general state transitions increases the complexity of the system significantly.

4.2 Benefits of nahmii

There are a number of benefits to off-chain constructions or so-called 'second layer scaling solutions', described below. Understandably, there is a huge amount of excitement and anticipation surrounding these kinds of projects.

4.2.1 Security

Whilst nahmii is an off-chain solution, it is architected such that it maintains security which can theoretically approach that of the Ethereum base layer. All attempts to defraud the protocol should lead to failure and punishment. The security model of nahmii is discussed in detail later in this document.

The protocol has a clear route to becoming fully trustless, which preserves nahmii's current ability to support commercial use cases.

4.2.2 Transactional Throughput

Performance and throughput has been a major priority in the design and implementation of the nahmii backend. Best practices allowing massively scalable/web-scale deployments have been followed and compromises in performance have only been made where we feel security would otherwise be jeopardised. The resulting backend will conservatively be able to process 15 driips per second per user (address), *with no practical restriction on the number of potential addresses*. For driips that require serial processing across addresses (such as trades, which require the maintenance of an order book) we will be more limited, but equally as performant as centralised exchanges. nahmii can therefore accommodate millions of users, handling millions of driips per second as required.

driips Per Second Per User

We prefer to specify the transactional throughput of nahmii in terms of driips per second per connection. We believe this is the ultimate metric for assessing a protocol, particularly when considering the capability of nahmii to handle trading, microtransactions or devices connected via the Internet of Things. As an example, consider microtransactions related to content: a user cannot have a positive experience of browsing or viewing content if there is a noticeable delay of seconds between actions. Users or companies who require more throughput than 15 transactions per second can simply make additional connections to the protocol.

Base Layer Scaling

Scaling of the Ethereum network base layer is complementary to constructions such as nahmii. Ethereum transactions are needed for deposits, disputes and withdrawals for the nahmii protocol. Therefore, as Ethereum scales, the on-ramps and off-ramps for nahmii gain capacity with the associated benefits of quicker and cheaper transactions.

4.2.3 Transactional Volume and Gas

In ordinary operation, nahmii will only require transactions to be submitted to the Ethereum network for deposits and withdrawals from the system. Once a user has deposited into nahmii that user may make essentially infinite driips within the platform, only needing to send further transactions to the Ethereum network to either top up their account balance or make withdrawals. Despite the potential additional gas overhead for depositing and withdrawing on-chain, nahmii users will benefit from significant cost savings after just a few payments or trades.

The on-chain dispute mechanisms within nahmii also incur additional transaction volume and gas costs. This is a small price to pay for the associated security benefits that these mechanisms provide; nahmii is designed to be a robust system where attackers are dissuaded by maximally significant penalties for failure and similarly high chances of detection. As such, any dispute overhead is minimal for the protocol, relative to other security options.

4.2.4 Finality

Transaction finality is an essential component of any economic system. In an ideal scenario, the process of finalising transactions would be limited only by latency. In blockchain-based systems however, transaction finality is generally probabilistic; a Bitcoin transaction is considered final after 6 block confirmations, which are usually completed in around 60 minutes. For Ethereum the equivalent might be considered to be 12 block confirmations or just under 3 minutes.

The finality of driips within nahmii is *immediate*, i.e. transactions are final as soon as a signed execution receipt is published by the Operator.

Finality is critical for an exchange which seeks to have tight spreads on currency pairs. Arbitrageurs need assurance that both sides of their trades are executed across whichever two exchanges they are using. If an exchange provides rapid guarantees that a trade has taken place and cannot be reversed, arbitrageurs are exposed to significantly reduced risks. The lack of these guarantees goes some way to explaining the significantly wider spreads on decentralised cryptocurrency exchanges compared with traditional forex markets. These risks are not fully mitigated by using a centralised exchange, which provides a receipt of trade execution, as execution guarantees are contingent upon successfully withdrawing from the exchange due to counterparty risk. The nahmii protocol is designed to avoid this counterparty risk.

Latency

For nahmii, driip finality is determined by latency in the system. nahmii's architecture ensures that a user's connection latency will have no impact on the execution of an order and its driip finality once a signed order has been registered with the system. The main source of any delays in processing would therefore be the security and fraud detection checks that are performed as the order is registered and subsequently executed. It is important to ensure that the user's experience is optimised for instant feedback in any products built upon nahmii. This requires a robust backend architecture.

Of particular importance for users of an exchange is transparent order processing and management of their orders. As such, the user interface must give near real-time feedback about order placement and cancellation. Our off-chain order book must add minimal latency, as discussed earlier. We expect that 95% or more of all placed orders will eventually be cancelled by traders using nahmii; latency adds risk to traders, in particular arbitrageurs.

Storage and Bandwidth Requirements

There may be a concern about the possible storage requirements of nahmii once it achieves a high throughput. However, it is possible to trim settled driips from the live driip database. Once driips have fully settled they are effectively checkpointed.

The bandwidth requirements of nahmii, notably its data publishing element, will be in keeping with similarly scalable web applications. The architecture can also leverage enterprise grade cloud infrastructure if needed.

The architecture of nahmii ensures effective decentralisation, as opposed to requiring a set of decentralised nodes which are relied upon to maintain consensus about data. This approach is far better suited to real world use cases.

4.2.5 driip Volume Limits

The nahmii architecture places no arbitrary limits upon the amount of tokens in a driip. As such, payments and trades will not normally be limited in size.

From Micropayments to Picotransactions

Micropayments have long been hailed as the solution to the problem of content monetisation, however the underlying technology has failed to facilitate this vision. Payments made in fiat currencies are limited either by the minimum transaction size or the requirement for a third-party to aggregate smaller amounts. In contrast, true micropayments are made using tiny fractions of a dollar, with minimal transaction fees. The nahmii protocol is able to handle extremely small driips such as *pico*transactions, depending on the token. Due to our highly efficient architecture, nahmii can provide the ideal platform for genuine micropayments: very small transactions with very low fees, delivered trustlessly to content providers.

Fees are the determining factor for minimum driip amounts. For a fee of 0.1% and a typical ERC20 token with 15 decimal places, this minimum driip size is 1×10^{-12} tokens, or a picotoken. For Ether which has 18 decimal places the minimum driip size is 1×10^{-15} tokens or a femtotoken.

The ability to have value flow in this way opens up numerous new business models, many of which hubii will be taking advantage of immediately by leveraging our existing content business. Aside from content monetisation, genuine micropayments and the countless financial applications, this also means that nahmii can function as a protocol upon which IoT (Internet of Things) systems can be developed. Note, the minimum driip size may be modified in the future to ensure the protocol is resistant to abuse.

4.2.6 Transaction Fees

There will be fees for transacting any type of driip on nahmii. These fees are essential for the security of the platform and for building an ecosystem of products based on nahmii. The fee structure has been designed so that the costs of using nahmii should be competitive with, if not considerably lower than, the fee for performing the equivalent transaction on the Ethereum base network. Our architecture is able to benefit from extremely cheap computation and should therefore be substantially more efficient in processing transactions.

Transaction fees within nahmii are also consistent and known in advance. Unlike Ethereum's variable gas price, which determines the fee paid to the network, nahmii transactions have a fixed cost. Even better, transactions within nahmii incur a fee which is paid in the currency of that transaction. Transactions on the Ethereum network require the user to hold ETH, which is paid to the miners in exchange for processing the transaction. This is a significant UX issue. With nahmii, transaction fees are paid in the currency of the transaction: the fee for sending HBT within nahmii is paid in HBT; on Ethereum the user requires ETH. It is self-evidently better for the user to pay the transaction fee in the same currency as their payment.

4.2.7 Account Flexibility

Ordinarily, state channel constructions can introduce undesirable properties, such as restrictions on top-ups, movement of arbitrary token denominations or partial withdrawals. nahmii will have no such restrictions. Users may top-up with ease, transfer or trade any arbitrary amount of funds and make partial withdrawals as needed. Users are not required to close their nahmii account to settle their balance and withdraw.

4.2.8 Hot, Warm and Cold Wallets

With clever adaptation of the user interface, users will have the option to store funds in 'hot', 'warm' and 'cold' wallets within nahmii:

- A cold wallet is the most secure method of storing funds within nahmii and is the recommended solution for users with large balances. Such a wallet would usually be controlled by a hardware device, such as those from Trezor or Ledger
- Warm wallets require a password, code or similar passphrase in order to sign driips, with user's key pairs stored securely in an encrypted format on the device they are using. This model offers strong security, however it carries a higher risk than the cold wallet option
- A hot wallet can be used within the nahmii ecosystem and remain 'unlocked' upon entry to our products. Typically this would not be considered secure, however it can be acceptable for small sums of money (such as a micropayments wallet)

Due to nahmii's low latency, instant finality and ease of use, we can enable these features in a user friendly way which is not currently possible on the Ethereum network.

4.2.9 Upgrades

One of the key architectural principles behind nahmii is that the development of the protocol should keep pace with the needs of commercial use cases. This has been a particular problem for the blockchain ecosystem as a whole, leading to many competing 'blockchains' of dubious quality and purpose. It is our strong belief that any governance of a blockchain should be introduced at the second layer. The base layer, in nahmii's case Ethereum, should remain untouched as a bastion for the key principles of blockchain technology: decentralisation, immutability and permissionless innovation.

The nahmii protocol has already proven itself to well suited to responsive development. Smart contracts were deployed in a modular fashion over four months, with new contracts supporting trades coming in mid-2019. Similarly, we made significant progress towards establishing the nahmii Foundation with the first members expected to be announced in Q3 2019. Users will therefore benefit from the partial decentralisation of nahmii now, even before the protocol is fully decentralised.

hubii and the nahmii Foundation will ultimately be responsible for ensuring the security of nahmii during any future upgrades. In general, nahmii has been designed to be easily upgradable; however, users may have to opt-in to upgrades in a trustless fashion.

5 nahmii Architecture

5.1 Security Construction

The nahmii security architecture divides into three levels, each containing a set of nodes. This separation of concerns is best understood as:

1. **Operator**

   The nahmii protocol will be operated initially by hubii, who will provide the first point of validation on the platform. This arrangement is subject to the governance and approval of the nahmii Foundation. Much of nahmii's security construction is designed to ensure the Operator is unable to commit fraud, even if compromised.

2. **User Monitoring**

   Any user of the Ethereum network can submit fraud challenges to the smart contract, with the reward for a successful challenge being the fraudulent user's balance in the currency of the fraud. These challenges are explained in more detail in the 'Continuous Fraud Challenge' and 'Settlement Challenge' sections.

3. **Data Availability Validation**

   The fraud challenges detailed above require users to submit proof in the form of driip receipts. These receipts are published by the Operator and nahmii token holders are responsible for validating that this data is available at all times. The 'Data Availability Validation' section provides more detail on these points.

The security provisions within the nahmii protocol are designed to protect against three possible fraudulent attacks, characterised in terms of data availability. First, users are protected against the possibility of a compromised protocol Operator through the 'Continuous Fraud Challenge' mechanism. Second, users are protected against illegitimate driip rollbacks through the 'Settlement Challenge' mechanism. Each of these challenges requires transaction data (in the form of receipts) to be both available and accurate, hence the need for a third security provision for when data is not available. The fully decentralised 'Data Availability Oracle' is designed to continually test data availability; this is the third security provision.

This section sets out the three security provisions in detail, explaining the rationale behind each and how they work together to ensure the safe operation of the nahmii protocol.

5.1.1 Operator

The Operator of the nahmii protocol will be hubii, with the option to decentralise this processing later with the support of the nahmii Foundation. Additionally, it is theoretically possible for other entities to run their own nahmii-based systems. As an example, an online gaming company may wish to be the Operator for an in-game item distribution system powered by nahmii.

The security constructions within nahmii have been designed to protect user's funds in the event of a rogue or compromised Operator. In the event that the protocol has been compromised, nahmii will simply close down gracefully and within minutes restart under a different set of suitably air-gapped smart contracts. Users can then opt-in to this migration, if they are happy to use nahmii again.

The security of nahmii is further enhanced by the foundation model of governance, which makes the possibility of a malevolent Operator gaining overall control substantially less likely.

5.1.2 User Monitoring

Continuous Fraud Challenge

In order for nahmii to approve a potential driip, it must first be signed by both the user initiating the driip and the Operator using their respective private keys. The integrity of the protocol depends on the Operator only signing valid driips, hence the requirement for a security check to ensure that this is the case. This 'Continuous Fraud Challenge' is therefore designed to protect users from the possibility of a compromised or rogue Operator. Note, below we give one simple example of fraud, but the nahmii protocol must be able to detect all forms of fraudulent driips.

The twin sign off process is best illustrated by way of a simple example, a request by Alice (A) to make a payment of 100 tokens to Bob (B). First, Alice initiates the payment request and signs it with her private key to verify the transaction. Next, the Operator (O) checks Alice's balance to ensure that the appropriate funds are present. Provided that Alice has a sufficient balance, the Operator signs the driip using their own private key. After performing this second check, the Operator decrements Alice's balance by 100 tokens and increases Bob's balance by the same amount. Finally, the Operator publishes the driip details on a publicly accessible website in the form of a standardised receipt. The process can therefore be represented as:

1. A initiates a payment request to send 100 tokens to B
2. A signs the transaction using her private key

3. O checks that A has sufficient balance for the payment (A does)
4. O signs the driip using their private key
5. O decrements A's balance by 100 tokens
6. O increases B's balance by 100 tokens
7. O publishes the driip receipt on a publicly accessible website

In this example, we have presumed that both Alice and the Operator hold valid private keys and that the Operator has not been compromised. If Operator has been compromised, this raises the prospect of invalid driips being signed. Once again, this is best illustrated by an example:

1. A initiates a payment request to send 100 tokens to B
2. A signs the transaction using her private key
3. O checks that A has sufficient balance for the payment (A does)
4. O signs the driip using their private key
5. O decrements A's balance by 100 tokens
6. **O increases B's balance by** 200 **tokens**
7. O publishes the driip details to a publicly accessible website

Clearly something has gone wrong; Bob's balance should increase by 100 tokens, not 200. The Operator here has executed A's valid request incorrectly and signed an invalid transaction. This transaction is invalid because the payment amounts do not match. The only explanation for the Operator signing invalid transactions is that the Operator has been compromised; the protocol needs to be halted. Importantly, Bob can never withdraw this falsely inflated balance. If Bob attempts to settle their account and withdraw, he will need to provide the driip receipt as evidence. This will be challenged as fraudulent and Bob's withdrawal will never be approved, regardless of whether the Operator has signed the transaction. What this means is that Bob, as with any other user should check a driip receipt is valid before they accept payment; this is trivial to implement using wallet software.

The key to identifying fraudulent driip of this type lies in step 7, where the Operator publishes all driip data to a publicly accessible website. Based on this information, users of the Ethereum network can challenge the driip by calling the appropriate function of the smart contract. All of the necessary information is publicly available to prove the fraud. Once a successful proof is submitted to the smart contract, the nahmii protocol is halted and the user who raised the challenge receives a reward from the security bond. We refer to this state as 'exit mode' and nahmii can recover from being halted in this way by redeploying new contracts once Operator control has been reestablished. Note that user's funds are never at risk here; a compromised Operator cannot make fraudulent withdrawals from the Client Fund.

An example of a successful challenge by Carol, C, is:

1. A initiates a payment request to send 100 tokens to B
2. A signs the transaction using her private key
3. O checks that A has sufficient balance for the payment (A does)
4. O signs the driip using their private key
5. O decrements A's balance by 100 tokens
6. **O increases B's balance by 200 tokens**
7. O publishes the driip details to a publicly accessible website
8. C raises a challenge against the driip by calling the smart contract
9. C provides evidence of fraud using the data from step 7
10. Smart contract confirms fraud and puts nahmii into exit mode
11. C is awarded a share of the security bond as a reward

This challenge is not free, as calling the smart contract incurs a gas cost. This mitigates against an effective DoS (Denial of Service) attack against nahmii, as to do so the attacker must DoS attack the entire Ethereum network. If many challenges are raised in a short period of time, the cost of calling the smart contract would rise quickly to the point where a sustained attack would be uneconomical. Importantly, there is no cost for the Operator to permit these fraud challenges.

This feature eliminates a further vulnerability whereby an attacker could repeatedly spam the Ethereum network with fraudulent settlement requests. Each of these settlements must be then challenged to secure the protocol, placing a significant financial burden on the Operator. Further protection against this type of attack comes from nahmii's minimum balance requirement, which ensures that successful challenges against a fraudulent settlement always receive an appropriate reward..

We anticipate that many nodes will be monitoring the published transaction data for anomalies, including those nodes run by the nahmii Foundation partners. Please refer to the earlier 'Foundation Model' section of this document for more information regarding these points.

Settlement Challenge

The second security provision within nahmii is designed to ensure that driips are settled such that a user's balance is brought up to date. The nahmii protocol functions by moving the majority of transactions off-chain, away from the slower Ethereum base layer. Eventually a user will need to reconcile their off-chain activity with their on-chain state; this process is called 'settlement'.

Secure settlements are critical to ensuring that users are unable to perform effective personal rollbacks, which would otherwise undermine trust in the system. Please note that the description provided here is something of a simplification, as some flexibility has been added to the settlement process without compromising security. We provide one simple example of an illegitimate driip settlement below; however, there are a number of more complex possibilities not covered here that nahmii is capable of handling.

The need for a 'Settlement Challenge' is best understood in terms of the nahmii withdrawal process. nahmii requires that users 'settle' their account before withdrawing and the settlement process may include an intent to withdraw only a certain portion of a user's available funds.

Once a user has initiated the settlement process, the smart contract starts the 'dispute period' timer during which the request can be challenged. It is at this point that any nahmii user can challenge the request through the 'Settlement Challenge'. As with the explanation of the 'Continuous Fraud Challenge', this process is best understood by way of an example:

nahmii user Alice (A) has completed ten driips on the exchange and is yet to settle her account. The ninth driip shows that Alice had a balance of 100 HBT tokens and Alice requests to settle her account to this wallet nonce. She also requests to withdraw 50 HBT tokens of her balance once her account is settled. Alice makes the appropriate request by calling the smart contract and providing the details of driip nine. The smart contract then begins the dispute period, during which an 'Settlement Challenge' is possible. A successful withdrawal request without a successful challenge therefore looks like this:

1. A begins the nahmii withdrawal process by requesting to settle her account
2. A calls the smart contract and provides both the details of the driip she wishes to settle up to and the balance she wishes to withdraw
3. The smart contract checks that the request is valid and begins the dispute period
4. No successful 'Settlement Challenge' is made during the dispute period
5. A's request is approved and the appropriate funds are moved to her withdrawable balance
6. A may now withdraw funds from this balance at any point

The alternative scenario is one in which A's withdrawal request is successfully challenged by C. In our example, A wanted to settle her account up to driip nine of ten at which point her balance was 100 HBT. If A's tenth driip was a payment sending 75 HBT tokens to B, her available balance *taking all ten driips into consideration* is 25 HBT. A's request to withdraw 50 HBT tokens is therefore fraudulent, as she does not have these funds available in her account (A's actual balance is 25 HBT, not 50 HBT). C may challenge this fraudulent request by providing the appropriate evidence to the smart contract as proof, namely the details of driip ten. A successful 'Settlement Challenge' rewards the challenger with the fraudulent user's balance in that currency. This is detailed below:

1. A begins the nahmii withdrawal process by requesting to settle her account
2. A calls the smart contract and provides both the details of the driip she wishes to settle to and the balance she wishes to withdraw
3. The smart contract checks that the request is valid and begins the dispute period
4. C raises a 'Settlement Challenge' by calling the appropriate smart contract function
5. C provides evidence of A's fraudulent request, namely the later driip showing the discrepancy in A's available balance
6. The 'Settlement Challenge' is successful and C receives A's balance in that currency (25 HBT) as a reward

Importantly, A's payment of 75 HBT to B is still honoured; B is not punished for A's mistake. If A had been allowed to make a payment of 75 HBT, then her balance would have been 0 HBT. If we allowed a payment to zero an account then there would be no reward for the ‘Settlement Challenge’, therefore we require a minimum balance to perform payments in nahmii. For more information please see the ’Minimum Balance’ section. This ensures finality is preserved in nahmii.

This ‘driip’ Settlement Challenge (or DSC) process is one of two ways that users can settle their account and withdraw from nahmii. Deposits into nahmii do not count as driips, so it is possible that a user can have a balance in nahmii without a subsequent driip receipt. While the user could make a transaction, thus generating a receipt, prior to starting a standard DSC, this is suboptimal from a UX perspective. Users without an appropriate driip receipt can therefore start a ‘null’ Settlement Challenge (or NSC) if their last interaction with nahmii was in the form of a deposit.

There are two reasons why a user would start a NSC over a DSC. First, the user has only ever deposited into nahmii with no other transactions. Second, the user has made previous transactions within nahmii (thus generating receipts) but their last action was to deposit into the protocol. In both cases, the user has no driip receipt which shows their updated balance inclusive of the latest deposit. An NSC is therefore required.

The NSC process follows the same workflow as the DSC example outlined above. If Alice deposits 100 HBT then starts a NSC to withdraw her full balance, this settlement request could be challenged by any subsequent outgoing payment driip (as this would take her balance below the deposited amount). We will not punish Alice if she has also made further deposits, taking her balance back above the deposited amount before a successful challenge was made.

Minimum Balance

Users are required to maintain a minimum balance in order to use the nahmii protocol, but this can be implemented Operator-side without adding any risk to users. In the unlikely event that the Operator chose to set the minimum balance requirement at an excessively high level, users would still be able to exit and withdraw from nahmii.

The minimum balance requirement is designed to mitigate against the ‘nothing at stake’ problem, whereby users can effectively test the security of the protocol for free. If users can still transact with a nominal balance, the deterrent for attacking nahmii (namely, losing that balance) is insufficiently small to outweigh the potential gains from a successful attack. Similarly, the reward for challenging a fraudulent driip settlement (namely, claiming that balance) is an insufficiently great incentive for other users of the system who might otherwise raise a ‘Settlement Challenge’. Raising a ‘Settlement Challenge’ is not free, it incurs a gas cost; the system therefore requires that the cost to raise a ‘Settlement Challenge’ should always be lower than the potential reward, as otherwise there is no incentive for a rational actor to do so.

We recognise that the minimum balance requirement represents a minor inconvenience for users of nahmii; users can't take their balance lower than a small threshold value when making a payment, but they can always withdraw their full balance. In this way, the small balance acts as a security bond posted by the user. It is expected that a value of approximately $1 per currency would be sufficient for this bond, though this can be adjusted over time. The user experience cost of imposing this requirement is more than outweighed by the associated security benefits.

The minimum balance requirement highlights one of the fundamental security principles underpinning nahmii's architecture: any attempt to defraud nahmii must always be maximally risky for the attacker and accompanied by a sufficiently great cost if they fail. In this way, we can design a system which makes prolonged attacks unsustainably unaffordable and opportunistic attacks unattractive.

It is important to note that we will only implement a minimum balance for payments; for trades it is assumed that when a user empties their balance in one currency in exchange for another currency, they maintain the same value. The other currency involved in the trade then effectively becomes the bond.

5.1.3 Data Availability Validation

The Data Availability Problem

The fraud and settlement challenges set out in this section rest on the principle of data availability, which requires that the Operator publish accurate and complete driip receipts at all times. This data is crucially important for ensuring confidence in nahmii and the Operator. Without the relevant data, users cannot challenge fraudulent driips as there is no evidence to send to the smart contract. Similarly, the off-chain elements of nahmii cannot be verified without the driip receipts; users cannot therefore be confident that their driips have been processed correctly by the Operator without access to the appropriate data.

nahmii users cannot simply presume that the Operator is trustworthy, they must be able to inspect the published data to be sure. As such, nahmii requires a decentralised method by which the smart contract can check that data is available. Furthermore, this must be secured against external manipulation.

The challenge described above is known as the 'data availability problem'. If users of a protocol like nahmii need the Operator to publish driip data to check whether the Operator is trustworthy, how can users ensure that the Operator is publishing the appropriate data? Our solution is the 'Data Availability Oracle', a fully decentralised mechanism by which distributed nahmii token holders are incentivised to monitor data availability.

Data Availability Oracle

The 'Data Availability Oracle' is designed to protect nahmii users from a potentially compromised Operator who is withholding data. As discussed, the 'Continuous Fraud Challenge' and 'Settlement Challenge' processes rely on users submitting proof of a fraudulent driip. This proof is taken from data published by the Operator, without which the challenges cannot function. In the unlikely event that a compromised Operator chooses to publish fraudulent driip data, this attempted fraud is easily identifiable through the driip signatures, sums and values. Far more likely is the alternative scenario, whereby a compromised Operator refuses to publish the relevant data (either by selectively excluding certain driips or simply withholding all data). In this case, the proof of the attempted fraud is the *absence* of data rather than the presence of tangible evidence. This requires a different kind of solution.

The Oracle is best understood as a function of the account settlement process within nahmii. Before a user can withdraw funds, they must first request to settle their account to a particular driip. The first step in the withdrawal process is therefore to check whether the user's driips up to the driip in question are all valid, this requires the user to call the smart contract and start the dispute timer. During this period, other nahmii users may challenge the request by providing evidence in the form of a transaction receipt. If the request is proven to be fraudulent, the user loses their balance in the currency of the settlement. If the dispute period ends without a successful challenge, the user can then effectively reactivate the smart contract (which has been dormant during the dispute period). As the settlement request has not been proven fraudulent, the smart contract performs one final check before allowing a withdrawal: *is data available?* This is done by querying the Oracle.

In order for the Oracle to function, it must return a binary response when the smart contract checks whether data is available (yes/no, true/false etc.). The Oracle is a game theory-based distributed intelligence tool, which was first discussed in The Hubii Network white paper. It operates on the principle of a small reward for being a good actor and a severe punishment for being a bad actor. The purpose of the Oracle is to continually test several statements relating to data availability during a settlement dispute period. These statements have only two truth conditions, true or false (or equivalents, i.e. no indeterminacy is possible), where the combination of these statements provides a similarly clear answer to the question 'is data available?'. The outputs of the Oracle should also account for the possibility of a legitimate, temporary problem with data availability which is fixed later. If the Oracle subsequently returns the response indicating that data is available again, the nahmii settlement process is effectively reactivated.

Questions around data availability must necessarily include a temporal component, as data availability can change over time. This relates to the staking mechanism at the heart of the nahmii Oracle, whereby nahmii token holders stake their tokens against 'true' or 'false' for each of the data availability statements. It is trivial to see why the monitoring market must include a temporal component. Without this time restriction, two users may stake their tokens on opposite outcomes and both be correct. Consider the simple statement 'data is available', this can be both true at time t_1 and false at time t_2. It is therefore essential that the crucial monitoring questions are formulated correctly. In order for the status of a question to change between 'true' and 'false', then the staking of users must achieve a variety of criteria. It is insufficient for this system to be based on a simple honest majority assumption.

Users are incentivised to participate in this monitoring market by the promise of payment for being correct, with the optional introduction of additional incentives for staking early in the process if required. This is known as the 'Data Availability Bond'. This bond is accrued from nahmii transaction fees and a portion is available as a reward for staking correctly. By only awarding a fraction of the bond as a reward for identifying when data availability changes, we ensure that there is always a reward for reversing any status change. Tokens of users who staked in opposition are seized and provide an additional reward; this is an essential punishment for being a bad actor.

The Oracle will function entirely on-chain as a true decentralised process, with no possibility of centralised interference. Therefore, the Oracle must be optimised over time and is still in testing. It must be able to quickly and accurately resolve whether data is available, yet be Sybil and 51% attack resistant. This is a non-trivial requirement.

As part of the foundation model, discussed earlier in this paper, nahmii Foundation members will be responsible for ensuring a plurality of token holders, minimising the risk of attack on the Oracle. Similarly, key Foundation members will be required to host replicate driip data. This will help mitigate some data availability false alarms. Whilst the Oracle is in the testing phase, Foundation members can also be given the ability to 'vote' to pause a settlement if they become aware of missing data. This results in nahmii being immediately decentralised and protected by reputable third-parties.

5.2 Liquidity of Funds

When a user deposits into nahmii, their funds will ordinarily fund the liquidity pool, known as the 'Client Fund'. Upon settlement, withdrawals will be made from the same pool. The nahmii protocol has been designed so that users can only access the appropriate amount of funds in the Client Fund; as such, nahmii is non-custodial and all funds are fully backed by user's deposits.

5.3 driip Ordering

The finality of driips on the nahmii protocol is determined by publication of data. In addition we utilise the concept of checkpointing; each time a new driip is confirmed through the settlement process, all transactions in the history of that driip are considered checkpointed. The full ancestry of a driip may be a complex branching structure and contain many thousands of previous driips and wallets. With checkpointing, validators and users only need to be concerned about the recent ancestry back to any previous checkpoints in the family tree

In the event that Operator fraud is detected, users will be able to settle their account up to the 'last known good' driip. Any driips dependent upon a fraudulent transaction are tainted and must never be settled; to do so would risk compounding the prior fraud.

nahmii will use either Merkle trees or accumulators to ensure sufficient information is encoded within a driip receipt, such that a driip's ancestry can be fully reconstructed trivially and checked by validators. Proofs-of-inclusion and proofs-of-exclusion can be used to ensure that no driips with a fraudulent driip in their ancestry can be settled. We are currently testing both Merkle trees and accumulators to determine which approach is most optimal; there are differences in computation for validators, but we must also consider the proofs and the associated gas usage for on-chain submission of those proofs.

5.4 Temporary Rollback - No Longer Necessary

In the first implementation of nahmii, we temporarily used a protocol-wide global nonce to determine the 'last known good' point: If the last known good driip is at global nonce 100 and the driip at global nonce 200 was shown to be fraudulent, we could not yet say whether driips 101 to 199 were valid or not. At this stage, the smart contract would only allow users to begin the settlement process for driips up to global nonce 100. Any driips beyond that point would be rejected.

In an earlier version of this white paper, we proposed an incentivised on-chain game where users could increase the checkpoint nonce. The idea was that users would eventually push the last known good settlement up to the one previous to where the Operator committed fraud. This global nonce was a simple temporary measure to ensure ordering of transactions; it did impact protocol performance and so this has already been removed. The change also effectively removes the need for a discussion around 'Temporary rollbacks' and this section of the white paper is now redundant.

6 Tokens and Airdriip

nahmii is a tokenised protocol. There are a total of 120 billion nahmii (NII) tokens, which are held in time-locked contracts and released at the rate of 1 billion tokens per month over 10 years. The majority of tokens will be distributed through regular airdrops, which are known as *airdriips* following nahmii's naming convention.

All transactions within the nahmii protocol incur a transaction fee, with fees accruing to token holders and nahmii's community of protocol facilitators. The overwhelming majority of fees will go to NII token holders, with a small percentage being used to fund various security bonds and nahmii's Data Availability Oracle.

The nahmii project was not mentioned in hubii's original white paper; however, development of the protocol has been funded exclusively by hubii and we therefore recognise our existing community with the NII token allocation. As they are released from the time-locked contract, NII tokens will be distributed accordingly:

- 50% proportionally to HBT token holders (including any HBT on deposit within nahmii)
- 20% to be sold or airdriipped as seen fit, with oversight by the Foundation[1]
- 20% to be held and controlled by the nahmii Foundation
- 10% to the key partners that developed nahmii

As stated, effectively all transaction fees will go to the token holders and facilitators of the protocol. It is essential that a significant percentage of nahmii fees are shared with token holders. In a similar fashion to many projects in this space, the security of the protocol is strongly related to the value of the token itself. As such, the token value acts as a bond for participants to ensure the correct operation of the security mechanisms. It is therefore critical to note that the funds that accrue to token holders is not a passive income; token holders must monitor, validate and secure the protocol. This is an incentive mechanism for participation, just as Bitcoin miners receive a mining subsidy and transaction fees.

The majority of the remainder of the generated fees contribute additionally to the 'Data Availability Bond' described in this paper. This bond, which increases over time, provides specific incentives for data availability validation and staking. This represents an additional reward for further active participation in the protocol for token holders, as only NII will be accepted for staking on data availability questions. Again, overall protocol value contributes directly to the difficulty of a 51% attack. As this value grows, a 51% attack becomes more expensive and the risk for an attacker increases.

[1]Note, for the first 8 months this was airdriipped to Ethereum holders that registered for the airdriip

Additionally there will be a minor security bond. This bond incentivise users to identify a number of Operator-only attacks, where there are no colluding user's funds to be seized. This fund might eventually be capped upon reaching a certain value.

The exact division of nahmii's transaction fees between token holders, data availability bond and the other minor security bond will be optimised over time. This long term optimisation will most likely be a Foundation decision.

6.1 Balance-Blocks

In order to calculate the appropriate airdriip share for each address holding HBT (and ETH whilst that portion was allocated), we introduce the concept of *balance-blocks*. Balance-blocks are designed to measure both the number of tokens held at an address and how that balance has changed over time, where balance is measured in tokens and time in blocks. More formally, the balance-block is defined as the integral under the balance versus block height chart for a given address across a specified period. One balance-block is therefore equivalent to holding one token for the period of one block.

The balance-block concept is sensitive to how a user's balance changes over time, ensuring that all token holders are treated fairly during the airdriip. This approach compares favourably with the traditional method of simply recording address balances at a fixed point in time and allocating tokens accordingly. In the traditional case, there is a strong incentive for a user to only hold HBT tokens around the time of the airdriip; a user who transfers 100 HBT into their wallet one day before the airdriip assessment will receive the same share as another user who has held the same number of tokens for the entire month. This could cause undesirable liquidity squeezes on HBT, which would detract from its main function as a currency. Under the balance-block model, the second user would receive a much greater of share of the airdriip relative to the first. This additional share is proportional to the duration and magnitude of their holdings, thirty times more in this case, as they would have held 100 HBT for thirty days compared to the 100 HBT held for one day by the first user.

Airdriipped NII will be distributed to users in accordance with their balance-block holdings over the qualifying period. While this method of distribution will serve to minimise monthly liquidity squeezes on HBT trading due to the periodic nature of the airdriip, we have also chosen to utilise balance-blocks as we strongly believe that this form of distribution is the fairest and safest possible way to organise an airdriip. The nahmii airdriip uses the nahmii protocol to deliver tokens directly to a user's off-chain wallet. Unlike with most airdrops, which must deliver a minimum value of tokens in order to be viable, nahmii's flexible fee structure means that no minimum holding is required to participate in the airdriip. This feature highlights a key benefit of using nahmii: micropayments are now viable, as the fee for sending these payments is proportional to their value.

Note that the distribution of tokens during the nahmii airdriip cannot be a trustless process due to the limitations of the Ethereum network. While the airdrop is currently processed by hubii directly, this task may be handled by the nahmii Foundation in the future.

6.2 Transaction Fees

All forms of driips will generate transaction fees within the nahmii protocol. Unlike their equivalent on the Ethereum network or in some other scaling solutions, fees within nahmii are designed to be predictable and transparent. Importantly, transaction fees within nahmii are paid in the currency of the transaction. This compares favourably with Ethereum and many other scaling protocols, which require a second currency to pay for fees. Our implementation of native currency fees is essential for commercial applications.

Fee levels within nahmii are not fixed and can be adjusted to meet the needs of the market. Fees will be initially set by hubii, however the nahmii Foundation will ultimately be responsible for future fee decisions.

6.2.1 Payment driip Fees

Payment fees are accrued trustlessly on a percentage basis, with discounts based on individual volume of each payment. The discount mapping can be set per currency, but there are also default amounts. Fees will be extremely competitive with Ethereum transactions, even at high individual payment volume. There may be a minimum fee requirement added to mitigate spam. Fees are paid in the currency of the payment and are currently set at a flat rate of 0.1%; however, nahmii fees will soon change to include the discounts detailed above.

6.2.2 Trade driip Fees

Exchange fees are based on a user's rolling 30 day volume, decreasing as transactional volumes rise. 30 day volume figures will be calculated in a trusted fashion and will most likely be calculated in USD equivalent initially. This will only be a minor trust issue as the fraud checks will not allow driips with fees that exceed the lower or upper boundaries. In the worst case, the Operator can only apply discounts incorrectly. If it is observed that the Operator is not consistently applying the right volume-based discounts then users can always safely exit and the Operator's reputation will be damaged. Fees are paid in the currency that the user is trading; each trade will generate fees in two currencies.

6.2.3 Trustless Generation and Claims

Fees in nahmii are incurred upon settlement, rather than at the time of the driip taking place. Every subsequent driip a user makes tracks the fees that they owe. Upon settlement, this fee is transferred automatically to a transaction fee fund. Funds will accrue periodically and token holders will be able to trustlessly claim their share after each period is closed by the Operator.

A token holder's claim on their share of nahmii's transaction fees will be determined by the NII balance-blocks that they have accrued in the previous qualifying period. For more information on balance-blocks, please see the 'Balance-Blocks' section. To make this claim process trustless, a minor modification to the nahmii ERC20 token was required in order to keep track of balance-blocks during transfers.

If NII are deposited to an address where the user does not have control of their private keys, there is no guarantee that the user will be able to claim their share of the transaction fees. This would ultimately be at the discretion of the third-party service that the user has deposited their NII into. Token holders are therefore strongly incentivised to keep their tokens in addresses that they control at all times, which also ensures that those tokens are always available for staking into the nahmii Data Availability Oracle. This includes tokens which have been airdropped to users within nahmii; users must withdraw their tokens from the protocol in order to activate them for accrual purposes. NII tokens within nahmii will not be eligible for accrual of transaction fees.

It should be noted that whilst tokens are staked into the Oracle itself, a token holder's share of the transaction fees will continue to be trustlessly accrued and can be claimed back from the Oracle contract later. This ensures that there is no disincentive to stake into the Data Availability Oracle.

7 hubii core

hubii core is the first product built upon nahmii. It should be considered the reference implementation and is open source. We expect and actively encourage many user interfaces to be created to interact with nahmii. Users will find not only a method of making payments and trading using nahmii, but also a growing set of features to interact in general with Ethereum. hubii core already includes hardware wallet integration for maximal security.

At the time of writing, hubii core is live on the public mainnet and has support for nahmii's deposit, payment, settlement and withdrawal functions. In addition, hubii core supports multiple wallets, two languages and both mainnet and Ropsten networks.

8 The Future

8.1 Immediate Withdrawal

Ordinarily, users will have to complete a successful settlement process prior to withdrawal. This process includes a dispute period, the duration of which will be optimised for the safety of users' funds. However, we can enable the possibility of immediate withdrawal. This option requires that other nahmii users can review the settlement request, before offering immediate liquidity to the settling user in exchange for a fee. This fee will be market driven and agreed by the two parties in advance of the withdrawal. The reviewing user, providing liquidity, will receive the settled funds after the dispute period instead of the original settling user. As such, the reviewer will accept the risk that the settlement process will not be successful. This risk should be very low, provided that the reviewing user has first checked that the relevant data was both available and correct. The fee for this service charged by the reviewing user should therefore approach the settling user's perception of the time value of their funds. This feature will be added as soon as it is sufficiently tested.

8.2 Derivatives

The implementation of trustless derivatives is one example of additional forms of driips. nahmii will natively handle payments and trades, but other forms of driips will become available over time. Margin trading is well understood by the wider cryptocurrency community and this is likely to be the next driip that is added to nahmii once trades are live. Although it is not particularly challenging to implement this function within nahmii, there are potential regulatory compliance issues which must be addressed before this form of trading is publicly released.

In addition to the benefits for traders, this feature is very important for hubii itself; trustless margin trading is our proposed method to provide the option to remove the exchange rate risk of hubiits (HBT) for users of our platform. Given sufficient liquidity, it is possible to hedge against price fluctuations and therefore mitigate this issue. This was discussed in our previous content platform white paper.

8.3 Fiat-Backed Tokens

The simplest way to integrate fiat in nahmii is by using fiat-backed ERC20 tokens. We are already exploring this possibility with various partners and we hope to add this soon. Inevitably, a fiat-backed token will be a trusted construction as fiat in itself is inherently trusted. It is therefore critical that we work with leading industry partners who are regulated by e-money or banking licenses. In many ways, there is no functional difference between a fiat-backed ERC20 token and an account balance in an e-money service. However, there are regulations and restrictions which must be adhered to.

8.3.1 Know Your Customer/Anti-Money Laundering

In order to comply with regulations, particularly when users are interacting with fiat currencies, it will at some point be necessary to identify certain users. nahmii has been constructed to have this functionality natively without impacting those users who choose not to use fiat.

8.4 Cross-Chain Interoperability

nahmii is designed for Ethereum and ERC20 tokens. We are working with partners on cross-chain implementations and it is possible to implement this trustlessly. Initially, we expect the most immediate use case is a cross-chain atomic swap. Let's imagine Alice has Token A on Blockchain A and Bob has Token B on Blockchain B. Assuming Alice and Bob want to simply swap their tokens, they sign their respective matching orders and the operator produces a cross-chain trade receipt. This receipt can be redeemed by Alice for her Token B on Blockchain B and similarly Bob can redeem his receipt for Token A on Blockchain A. Alice now has Token B and Bob has Token A, as expected.

Notably, we already have plans to port nahmii to a number of other blockchains. These include Bitcoin, via RSK, and Libra. Interestingly, nahmii will then act as a common interoperability layer between these blockchains.

8.5 Privacy

Transactional privacy has long been a primary concern of the blockchain community. While most blockchains offer pseudo-anonymity, there has always been an interest in moving to absolute privacy. We have undertaken research into nahmii driips with similarly high levels of privacy, which can maintain the same security guarantees. Implementations have been demonstrated on the main Ethereum network, which use a large amount of gas and so have significant cost per transaction. We are exploring an integration with a partner, in order to mesh their privacy transactions with nahmii's scalability yet allow for compliance with regulations.

8.6 Data Availability Redundancy

Data availability is a critical element of the nahmii protocol and it is necessary that we avoid 'data-withholding' false alarms. One way to maximise redundancy and minimise reliance on a single API is to use Distributed Hash Tables (DHTs). This is a topic of import for the Foundation.

8.7 Multisignature Wallets

An essential tool for security over funds within the Ethereum ecosystem has been multisignature wallets. nahmii will be multisignature compatible and this feature will be added to the platform at the earliest possible date.

8.8 Bond Limits

nahmii has a number of minor security bonds which are funded by a small share of nahmii's transaction fees. Currently a fixed percentage of nahmii's fees is diverted to increase these bonds; however, there may be a sensible limit applied to how much these bonds can grow. Under the assumption that the Operator is a good actor these bonds will continue to grow and should never need to be paid out, as such they might be viewed as burned funds. As nahmii achieves mainstream success, this ever-increasing bond size may become needlessly large. We may therefore make provisions for limiting this fund if required; it is likely to be a Foundation decision.

8.9 Other Token Standards

Adding other forms of tokens, such as ERC721 and ERC1155, is relatively simple and we have plans to add support for these tokens in the near future.

8.10 Patent

Certain elements of the nahmii protocol are patent pending. The decision to seek patent protection is driven by the need to keep the nahmii protocol both open and democratic on a perpetual basis. There are two principles behind this decision: in the first instance, that the patent application will ensure that no vested interest can interfere with or prevent the protocol from being deployed and used; second, by vesting the patent in the hands of the Foundation we are demonstrating our faith in the nahmii community to manage its availability and build upon it for the future.

When granted, this patent will be given over to the Foundation in perpetuity, conditional on the Foundation adhering to certain key principles. It will then be up to the members of the Foundation to determine a strategy for the nahmii patent. It may be decided that the best strategy is to give the patent away; however, this does not undermine the logic of applying for the patent in the first instance as to do so gives the Foundation the choice of how to proceed.

9 Appendix - Comparison Between nahmii, Raiden and Plasma

There are countless scaling solutions proposed across the blockchain ecosystem. For Ethereum, the two most prominent constructions until now have been Raiden and Plasma. There are many alternatives, which are in most cases effectively equivalent to these two constructions, but with a different name. A high level comparison between nahmii, Raiden and Plasma is provided here. It is important to note that this was based on hubii's understanding at a particular time (Q2 2019), it may not always reflect the current situation as these protocols continue to develop.

9.1 Raiden

Raiden is a payment-focused project, closely related to the Lightning Network from Bitcoin. It is a system designed around payment channels between individuals. The current status is that these payment channels are unidirectional and many-to-one on mainnet. However, testnet demonstrates many-to-many and bidirectional payments, mainly for small payments. Once implemented in full, payments can be routed through a network of nodes, making multi-hop transactions feasible and allowing any user to pay another user of the network, who is connected through a chain of nodes.

A high level comparison between Raiden and nahmii:

+ Raiden is posited as a fully decentralised system. However, its design can certainly create some centralising forces the extent to which remains somewhat unknown

= Raiden payments can offer somewhat improved privacy over the Ethereum base layer

= Raiden is highly scalable

- Atomic swaps, or trades on Raiden are coming in the future, but with no timeframe

- Raiden is usually low latency, similar to nahmii. Routing can introduce latency problems though

- The system is best suited and targeted for small transactions due to its architecture, but it is difficult to quantify the scale until the system is established

- There is a high capital inefficiency inherent within Raiden's design

- Fees will be determined by the nodes through which payments are routed and therefore might be unpredictable

- Raiden's full release timeframe is unclear

In our opinion Raiden and similar state-channel constructions can be useful for many use cases and we welcome further development.

9.2 Plasma

Plasma consists of a blockchain type construction or 'externalised multiparty channels'. It is a form of side-chain, known as a child-chain. Regular commitments are made to its parent: the root Ethereum chain. Plasma chains are similar to a blockchain, in that they work by bundling transactions into blocks which are submitted to their parents for later evidence. These transactions are merkleised and so only the merkle root of the block needs to be submitted to the root chain. There are many other Plasma *flavours*, which ostensibly solve some problems, at the expense of introducing more issues.

A high level comparison between Plasma and nahmii:

= Plasma can potentially have natively increased privacy, though not in most implementations

- Plasma will always have severe latency; more than Ethereum itself. This is a terminal problem for many applications. Of particular concern is the ability to ever build a liquid exchange on a protocol with high latency

- Plasma will have slower finality when compared to the root Ethereum chain and this prevents it from being applicable to many use cases, particularly payments. 'Fast finality' proposals remain to be proven and would not result in instant finality

- Plasma, in most cases, has an unclear route to decentralisation

- Plasma has an unclear route to trustlessness

- There exist a number of unresolved security issues for Plasma constructions

- Plasma has an unclear release timeframe

- Fees on Plasma will be unpredictable

- Due to it's high latency and periodic (potentially computationally intensive) commitments to parent chains, Plasma has scaling limitations overall and at an individual account level. An individual account might only be able to perform 1 transaction every 7.5s at best

- A Plasma construction will always be seriously impacted by any network congestion on the root Ethereum chain, either by fees rising, increased latency or reduced throughput

In our opinion there may never be a successful Plasma or child-chain type construction, despite the community popularity and interest. There are simply too many fundamental flaws to make Plasma commercially viable. Furthermore, development progress has in many cases stalled; at best, development efforts are fractured.