Deniz Aytemiz

HyperService: Interoperability and Programmability Across Heterogeneous Blockchains

As blockchain becomes more dominant in current technology, its scope of usage is also being extended. Especially with the introduction of smart contracts, blockchain is transformed from append only ledgers to programmable state machines. This opens the door for novel technological architectures along with establishing new technical challenges too. In this paper the problem of interoperability and programmability of blockchain is examined and a prototype called 'HyperService' is introduced as a provider of interoperability and programmability. Interoperability problem refers to being able to make state transitions between different blockchain networks. This is currently done in the scope of token exchange. Interoperability is especially important to achieve as Web 3.0 is becoming the next evolution of the World Wide Web. Programmability on the other hand is again a very important aspect especially with the increasing use and build of decentralized applications (dApps). Being able to achieve programmability and interoperability, one can develop and execute cross-chain decentralized applications in disjoint blockchain networks.

HyperService has four main components in its architecture. Firstly there are lightweight dApp clients which can develop or execute dApps. Secondly, there are Verifiable Execution Systems (VES) which translates the high-level programming language used by dApp clients to blockchain executable transactions. Their communication leverages UIP cryptography for secure transfer. The UIP protocol introduces two parts which are Network Status Blockchain (NSB) and Insurance Smart Contracts (ISCs). NSB provides the unified view of dApps execution status where ISCs examine the correction and security of executions in trust-free manner. UIP's correctness mostly relies on the performance of NSB. In HyperService technology, the cross-chain dApps are created and executed by Unified State Model (USM) which provides an extensible and blockchain independent 'unified' model for dApps. This model is independent of blockchain because it abstracts blockchain parameters such as consensus mechanism or programming language as *objects* with state variables and methods. The developers of dApps are programming by specifying the operations over those objects and the order of execution of those operations. HSL is developed for programming dApps and is used under USM. USM provides a 'unified ' visualization layer for programmers without getting involved in the heterogeneity of contracts. All the blockchain models considered in this work have public ledgers such that external pirates are able to examine it.

There are many technical challenges mentioned in this paper on both providing interoperability and programmability for blockchain along with the specific challenges in designing the HyperService. Firstly there is an ambiguity on the model of cross-chain dApps meaning there should be a virtualization that unifies or provides a standard for different blockchain interactions and operations as if all execution being placed in the same state machine. This is solved by the introduction of the Universal State Model (USM). Another issue is to ensure the security and correctness of transactions between dApp clients and blockchain infrastructure since there is no trusted authority responsible for this process. Designers leverage

UIP cryptographic protocol to ensure security between clients and VESes. UIP is executable in any blockchain and verifies the correctness of execution only if all stakeholders agree on it. Otherwise it detects and holds accountable the misbehaving parties. UIP itself is not 100% secure itself and in the threat model a possibility of an adversary exploiting the protocol and compromising is considered. However as long as one protocol participant is not compromised, UIP's security is guaranteed. One other challenge with UIP again is the lack of accountability between off-chain transactions between client and VESes protocols. To surpass this problem protocol of UIP enforces proof of actions in which Protocols of both client and VESes stake their execution steps on NSB. The atomic cross-chain swaps are restricted with token exchange currently and the operations needed for this are too simple to realize a dApp. Authors developing HyperService for this task enables simple dApps although it is indicated that more complex operations cannot be performed by their high level programming language.

The paper is informative and significant in terms of bringing a new architecture that enables programming and secure state transactions between disjoint blockchain environments. I especially found the design of 'mostly' secure unified layer for programmers from different blockchain networks interesting and insightful. Though HSL does not support complex functions at the first prototype it has other advantages such as being simpler for the developers with respect to other languages used in cross-chain dApps programming. The measurement results of the experiments made with the prototype also signals low latency though it must be emphasized that clients are lightweight nodes in the architecture. A problem mentioned is that HyperService is not fully dApp atomic since if a dApp terminates prematurely UIP cannot revert the process and update its state accordingly. The authors propose a stateless contract system where states are loaded before the execution.

I have learned about dApps and how they interact with blockchain structure, especially how USM and HSL works. I learned the terms atomic swapping and byzantine corruption model. More importantly I have gained significant insight on how different blockchains can securely interact and how a unifying or standardized protocol can be applied to them. I also understand the reasons why a platform such as HyperService or more advanced ones in future that introduces interoperability and programmability is crucial in the current technological setting.