Toward an Interoperability Architecture for Blockchain Autonomous Systems

Thomas Hardjono, Alexander Lipton, and Alex Pentland

***Abstract*—There is considerable interest today in the use of blockchain technology to provide better visibility into shared information among a number of participants and systems arranged in a decentralized peer-to-peer topology. Several challenges in blockchain technology remain to be addressed, including the interoperability, survivability, and manageability of blockchain systems. Crucial to answering these challenges is the need to understand aspects of the Internet architecture that has made it scalable, resilient, and a commercial success as a global connectivity infrastructure. In this paper, we discuss a design philosophy for interoperable blockchain systems, using the design philosophy of the Internet architecture as the basis to identify key design principles. We recast some of the challenges faced in the design of the Internet architecture to that of the design of an interoperable blockchain architecture. We emphasize interoperability as a crucial requirement for the survivability and manageability of blockchain systems. The goal is to define an interoperable blockchain architecture, in which common components of the blockchain architecture can begin to be standardized, leading to lowering of development costs, better reusability, and higher degree of interoperability.**

***Index Terms*—Computer network management, computer security, cryptography, decentralized control, distributed computing.**

I. Introduction

THERE is considerable interest today in the use of blockchain technology to provide better visibility into shared information among a number of participants and systems arranged in a peer-to-peer (P2P) topology. However, more attention needs to be placed on challenges around the aspects of the manageability of blockchain systems, the survivability of blockchain networks, and the cybersecurity of systems and infrastructures that participate in blockchain communities. Crucial to answering these challenges is the need to understand aspects of the Internet architecture that has made it scalable, resilient, and a commercial success as a global connectivity infrastructure.

The goal of this paper is to bring to the forefront the notion of *interoperability*, *survivability*, and *manageability* for blockchain systems, using lessons learned from the three decades of the development of the Internet. Our overall goal is to develop a design philosophy for an interoperable blockchain architecture, and to identify some design principles that promote interoperability.

Currently, there is considerable interest (real and hype) in blockchain systems as a promising technology for the future infrastructure of a global value-exchange network—or what some refer to as the "Internet of value." The original blockchain idea of Haber and Stornetta [1], [2] is now a fundamental construct within most blockchain systems, starting with the Bitcoin system which first adopted the idea and deployed it in a digital currency context.

Many parallels have been made between blockchain systems and the Internet. However, many comparisons often fail to understand the fundamental goals of the Internet architecture as promoted and led by the Defense Advanced Research Projects Agency (DARPA) and thus fail to fully appreciate how these goals have shaped the Internet to achieve its success as we see it today. There was a pressing need in the Cold War period of the 1960s and 1970s to develop a new communications network architecture that did not previously exist, one that would allow communications to survive in the face of attacks. In Section II, we review and discuss these goals.

We argue in this paper that if blockchain technology seeks to be a fundamental component of the future global distributed network of commerce and value, then its architecture must also satisfy the same fundamental goals of the Internet architecture. This is the core of Section II.

In Section III, we attempt to recast some of these fundamental goals of the Internet to the current context of blockchain technology. Among others, we look for some design principles for blockchain technology that should remain true across any blockchain system implementation.

Section IV discusses the notion *blockchain gateways* as a means to provide interoperability across blockchain systems. We review how routing gateways today allow various Internet service provider (ISP) networks to interconnect and provide end-to-end connectivity. The notion of boundary or perimeter is examined, and three potential uses of blockchain gateways are then presented. This paper is then closed with some observations and conclusions.

II. Design Philosophy of the Internet

In considering the future direction for blockchain systems generally, it is useful to recall and understand goals of the

Manuscript received February 17, 2019; revised April 22, 2019; accepted May 28, 2019. Review of this manuscript was arranged by Department Editor K.-K. R. Choo. *(Corresponding author: Thomas Hardjono.)*

T. Hardjono and A. Lipton are with MIT Connection Science and Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: hardjono@mit.edu; alexlipt@mit.edu).

A. Pentland is with MIT Connection Science and Engineering and the MIT Media Lab, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: pentland@mit.edu).

Digital Object Identifier 10.1109/TEM.2019.2920154

Internet architecture as defined in the early 1970s as a project funded by the DARPA. The definition of the Internet as view in the late 1980s is the following: it is "a packet switched communications facility in which a number of distinguishable networks are connected together using packet switched communications processors called gateways, which implement a store and forward packet-forwarding algorithm" [3], [4].

A. Fundamental Goals

It is important to remember that the design of the Advanced Research Projects Agency Network (ARPANET) and the Internet favored military values (e.g., survivability, flexibility, and high performance) over commercial goals (e.g., low cost, simplicity, or consumer appeal) [5], which, in turn, has affected how the Internet has evolved, has been used. This emphasis was understandable given the Cold War backdrop to the packet-switching discourse throughout the 1960s and 1970s. The ARPANET was an early packet-switching network. It was the first network to implement the transmission control protocol (TCP)/Internet protocol (IP) suite.

The DARPA view at the time was that there are seven goals of the Internet architecture, with the first three being fundamental to the design, and the remaining four being second-level goals. The following are the fundamental goals of the Internet in the order of importance [3], [4].

1) *Survivability:* Internet communications must continue despite loss of networks or gateways. This is the most important goal of the Internet, especially if it was to be the blueprint for military packet switched communications facilities. This meant that if two entities are communicating over the Internet, and some failure causes the Internet to be temporarily disrupted and reconfigured to reconstitute the service, then the entities communicating should be able to continue without having to reestablish or reset the high-level state of their conversation. Therefore, to achieve this goal, the state information that describes the on-going conversation must be protected. But more importantly, in practice, this explicitly meant that it is acceptable to lose the state information associated with an entity if, at the same time, the entity itself is lost. This notion of state of conversation is related to the end-to-end principle discussed below.
2) *Variety of service types:* The Internet must support multiple types of communications service. What was meant by "multiple types" is that at the transport level, the Internet architecture should support different types of services distinguished by differing requirements for speed, latency, and reliability. Indeed, it was this goal that resulted in the separation into two layers of the TCP layer and the IP layer, and the use of bytes (not packets) at the TCP layer for flow control and acknowledgment.
3) *Variety of networks:* The Internet must accommodate a variety of networks. The Internet architecture must be able to incorporate and utilize a wide variety of network technologies, including military and commercial facilities.

The remaining four goals of the Internet architecture are:

4) distributed management of resources;
5) cost effectiveness;
6) ease of attaching hosts;
7) accountability in resource usage.

Over the ensuing three decades, these second-level goals have been addressed in different ways. For example, accountability in resource usage evolved from the use of rudimentary management information bases into the current sophisticated traffic management protocols and tools. Cost effectiveness was always an important aspect of the business model for consumer ISPs and corporate networks.

B. End-to-End Principle

One of the critical debates throughout the development of the Internet architecture in the 1980s was in regard to the placement of functions that dealt with reliability of message delivery (e.g., duplicate message detection, message sequencing, guaranteed message delivery, and encryption). In essence, the argument revolved around the amount of effort put into reliability measures within the data communication system and was seen as an engineering tradeoff based on performance, that is, how much low-level function (for reliability) needed to be implemented by the networks versus implementation by the applications at the endpoints.

The line of reasoning against low-level function implementation in the network became known as the *end-to-end argument* or principle. The basic argument is as follows: a lower level subsystem that supports a distributed application may be wasting its effort in providing a function that must be implemented at the application level anyway [6]. Thus, for example, for duplicate message suppression, the task must be accomplished by the application itself seeing that the application is most knowledgeable as to how to detect its own duplicate messages.

Another case in point relates to data encryption. If encryption/decryption was to be performed by the network, then the network and its data transmission systems must be trusted to securely manage the required encryption keys. Also, when data enter the network (to be encrypted there), the data will be in plaintext and, therefore, susceptible to theft and attacks. Finally, the recipient application of the encrypted messages will still need to verify the source authenticity of the message. The application will still need to perform key management. As such, the best place to perform data encryption/decryption is the application endpoints—there is no need for the communication subsystem to provide for automatic encryption of all traffic. That is, encryption is an end-to-end function.

The end-to-end principle was a fundamental design principle of the security architecture of the Internet. Among others, it influenced the direction of the subsequent security features of the Internet, including the development of the IP-security sublayer [7] and its attendant key management function [8]. Today, the entire virtual private network (VPN) subsegment of the networking industry started based on this end-to-end principle. (The global VPN market alone is forecasted to reach 70 billion dollars in the next few years.) The current day-to-day usage of the secure sockets layer [9] to protect HTTP web traffic (i.e.,

browsers) is also built on the premise that client–server data encryption is an end-to-end function performed by the browser (client) and by the HTTP server.

C. Autonomous System Paradigm

Another key concept in the development of the Internet is that of *autonomous systems* (ASs) (or *routing domains*) as a connectivity unit that provide scale-up capabilities. More specifically, the classic definition of an AS is a connected group of one or more networks (distinguishable via IP prefixes) run by one or more network operators, which has a single and clearly defined routing policy [10]. The notion of ASs provides a way to *hierarchically aggregate routing information*, such that the distribution of routing information itself becomes a manageable task. This division into domains provides independence for each domain owner/operator to employ the routing mechanisms of its choice. IP packet routing inside an AS is, therefore, referred to as *intradomain* routing, while routing between (across) ASs is referred to as *interdomain* routing. The common goal of many providers of routing services (consumer ISPs, backbone ISPs, and participating corporations) is that of supporting different types of services (in the sense of speed, latency, and reliability).

In the case of intradomain routing, the aim is to share best-route information among routers using an intradomain routing protocol (e.g., distance vector such as routing information protocol (RIP) [11] or link-state such as open shortest path first protocol (OSPF) [12]). The routing protocol of choice must address numerous issues, including possible loops and imbalances in traffic distribution. Today, routers are typically owned and operated by the legal owner of the AS (e.g., ISP or corporation). These owners then enter into *peering* agreements with each other in order to achieve end-to-end reachability of destinations across multiple hops or domains. The primary revenue model in the ISP industry revolves around different tiers of services appropriate to different groups of customers.

There are several important points regarding the AS paradigm and the positive impact this paradigm has had on the development of the Internet for the past four decades:

1) *AS paradigm leads to scale:* The AS paradigm, the connectionless routing model, and the distributed network topology of the Internet allows each unit (the AS) to solve performance issues locally. This, in turn, promotes service scale in the sense of throughput (end-to-end) and reach (the large numbers of connected endpoints). As such, it is important to see the global Internet today a connected set of "islands" of AS, stitched together through peering agreements.
2) *Domain-level control with distributed topology:* Each AS typically possesses multiple routers operating the same intradomain routing protocol. The availability of multiple routers implies availability of multiple routing paths through the domain. Despite this distributed network topology, these routers are centrally controlled (e.g., by the network administrator of the domain). The AS as a control unit provides manageability, visibility, and peering capabilities centrally administered by the owner of the domain.
3) *Each entity is uniquely identifiable in its domain:* All routers (and other devices, such as bridges and switches) in an AS are uniquely identifiable and visible to the network operator. This is a precondition of routing. The identifiability and visibility of devices in a domain is usually limited to that domain. Entities outside the domain may not even be aware of the existence individual routers in the domain.
4) *AS reachability:* ASs interact with each other through special kinds of routers—called *Gateways*—that are designed and configured for cross-domain packet routing. These operate specific kinds of protocols (such as an exterior Border Gateway Protocol [13]), which provides transfer of packets across domains. For various reasons (including privacy and security), these exterior-facing gateway protocols typically advertise only *reachability* status information regarding routers and hosts in the domain, but do not publish internal routing conditions.
5) *ASs are owned and operated by legal entities:* All routing ASs (routing domains) today are owned, operated, and controlled by known entities. ISPs provide their *autonomous system numbers* and routing prefixes to *Internet routing registries* (IRRs). IRRs can be used by ISPs to develop routing plans. An example of an IRR is the American Registry for Internet Numbers [14], which is one of several IRRs around the world.

In the next section, we remap the fundamental goals of the Internet architecture in the context blockchain systems, with the goal of identifying some fundamental requirements for blockchain interoperability.

III. Interoperable Blockchains: Toward a Design Philosophy

During the 1970s and 1980s, several local area network (LAN) systems were in development and were marketed for Enterprises (e.g., IBM SNA [15] and DECnet [16]). However, these LANs were distinct enough in their technological approaches (e.g., PHY layer protocols) that they did not interoperate with each other [5]. Today, we are seeing a very similar situation, in which multiple blockchain designs are being proposed (e.g., Bitcoin [17], Ethereum [18], Hyperldeger [19], and CORDA [20], each having different technological designs and approaches. Most share some common terminology (e.g., "transaction," "mining node," etc.), but there is little or no interoperability among these systems.

Given the history of the development of the Internet and of computer networks in general (e.g., LANs and wide area networks), it is unlikely that the world will settle on one global blockchain system operating universally. The emerging picture will most likely consist of "islands" of blockchain systems, which—like ASs that make up the Internet—must be "stitched" together in some fashion to make a coherent unity (see Fig. 1).

Following from the first fundamental goal of the Internet architecture, the lesson learned there was that *interoperability is key to survivability*. Thus, interoperability is core to the entire value proposition of blockchain technology. Interoperability across blockchain systems must be a requirement—both at the

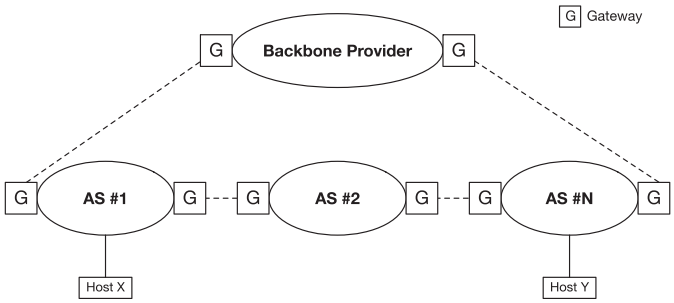

Fig. 1. ASs as a set of networks and gateways (after [4]).

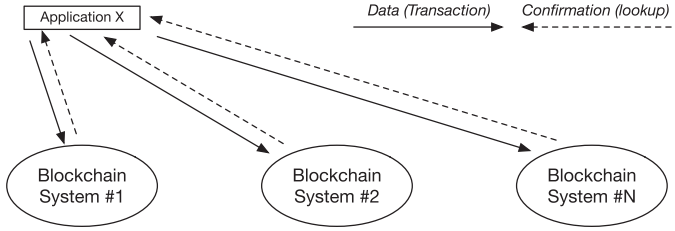

Fig. 2. Example of the reliability of a simple transaction.

mechanical level and the *value level*—if blockchain systems and technologies are to become the fundamental infrastructure of the future global commerce [21], [22].

This paper focuses primarily on the interoperability across blockchain systems at the mechanical level, as the basis to achieve a measurable degree of *technical trust* across these systems. In turn, technical trust is needed by the upper level functions to achieve interoperability at the value level, so that *legal frameworks* can be created that are able quantify risks based on the technological choices used to implement technical trust. Poorly designed blockchain systems should present a higher risk for commerce, and vice versa. Finally, *business trust* can be built upon these legal frameworks to allow business transactions to occur seamlessly across multiple blockchain systems globally.

In this section, we identify and discuss some of the challenges to blockchain interoperability, using the Internet architecture as a guide and using the fundamental goals as the basis for developing a design philosophy for interoperable blockchains.

In order to clarify the meaning on "interoperability" in the context blockchain systems, we offer the following definition of an "interoperable blockchain architecture" using the NIST definition of "blockchain" (see [23, p. 50]):

> *An interoperable blockchain architecture is a composition of distinguishable blockchain systems, each representing a unique distributed data ledger, where atomic transaction execution may span multiple heterogeneous blockchain systems, and where data recorded in one blockchain are reachable, verifiable, and referenceable by another possibly foreign transaction in a semantically compatible manner.*

In the following, we recast the aspects of survivability, variety of service types, and variety of systems in the context of blockchain systems.

A. Survivability

As mentioned previously, interoperability is key to survivability. In the Internet architecture, survivability as viewed by the DARPA [3], [4] meant that communications must continue despite loss of networks and gateways. In practical engineering terms, this meant the use of the packet-switching model as a realization of the *connectionless* routing paradigm.

In the context of blockchain systems generally, survivability should also mean continued operations in the face of various kinds of attacks. The possible types of attacks to a blockchain system have been discussed elsewhere and consist of a broad spectrum. These range from classic network-level attacks (e.g., network partitions, denial of services, distributed denial-of-service (DDOS), etc.) to more sophisticated attacks targeting the particular constructs (e.g., consensus implementation [24]–[26]), to targeting specific implementations of mining nodes (e.g., code vulnerabilities and viruses). Similar to applications on the Internet, we can also view survivability more specifically from the perspective of the application (and its user) that is transacting on the blockchain. A user's transaction should proceed as far as possible despite the blockchain being under attack.

For blockchain systems, we propose to reinterpret the term "survivability" to mean the completion (confirmation) of an application-level transaction independent of blockchain systems involved in achieving the completion of the transaction. Furthermore, the transaction may be composed of *subtransactions* and in the same sense of a message on the Internet may consist of multiple IP datagrams. Thus, in the blockchain case, an application-level transaction may consist of multiple ledger-level transactions (subtransaction), where each could be intended for (and be confirmed at) different blockchain systems (e.g., subtransaction for asset transfer in blockchain A, simultaneously with subtransaction for payments and subtransaction for taxes in blockchain B).

Here, the notion of packets routing through multiple domains being opaque to the user's communications application (e.g., e-mail applications and browsers) is now recast to the notion of *subtransactions confirmed on a spread of blockchain systems generally being opaque to the user application*. Thus, the challenge of reliability and "best effort delivery" becomes the challenge of ensuring that an application-level transaction is completed within reasonable time, possibly with the application itself being oblivious to the actual blockchains, where different ledger-level subtransactions are finally confirmed.

To illustrate the challenges of survivability as interpreted in this manner, we start with the simplest case in which an application sends a "data" transaction (signed hash value) to a blockchain for the purpose of recording it on the ledger of the blockchain (see Fig. 2). We ignore for the moment the dichotomy of permissionless and permissioned blockchains and ignore the specific logic syntax of the blockchain. Here, the application does not care *which* blockchain records the data as long as once the transaction is confirmed, later the application (and other entities) can find the transaction/block and verify that the data have been recorded immutably. Fig. 2 illustrates the scenario. The application transmits data bytes (hash) to a blockchain system No. 1 and waits for confirmation to become available on the blockchain. After waiting for some predetermined time unsuccessfully (i.e., timeout), the application transmits the same

data bytes to a different blockchain system No. 2. The application continues this process until it is able to obtain the desired confirmation.

Although the example in Fig. 2 may appear overly simplistic and inefficient and has the undesirable side effect of confirmations on multiple blockchains, it highlights a number of questions similar to those posed in the early days of the Internet architecture development:

1) *Application degree of awareness:* To what degree must an application be aware of the internal constructs of a blockchain system in order to interact with it and make use of the blockchain? Most (all of) wallet applications today must maintain configuration information regarding which blockchain system to which a key applies.
   As a point of comparison, an e-mail client application today is not aware of constructs of packets, media access control protocol data units (MPDUs), routing, and so on. It interacts with the mail server according to a high-level protocol (e.g., post office protocol 3 (POP3), Internet message access protocol (IMAP), and simple mail transfer protocol (SMTP)) and a well-defined API. The e-mail client needs only to know the destination e-mail address.
2) *Distinguishability and addressability of blockchain systems:* For an interoperable blockchain architecture, each blockchain AS must be distinguishable from a naming perspective as well as from an addressing/routing perspective. This introduces some new challenges, such as the situation where a node is permitted to participate in several blockchain systems simultaneously. From a key management perspective, there is also the question regarding multiple uses of the same public key pair across several distinct blockchain systems.
3) *Placement of reliability functions:* What is the correct notion of "reliability" in the context of interoperable blockchain systems and where should the function of reliability be placed? That is, should the function of retransmitting the same data bytes (transaction) be part of the application, part of the blockchain system, or part of a yet to be defined "middle layer"?
   As a comparison, within the TCP/IP stack, the TCP protocol has a number of flow control features that "hides" reliability issues from the higher level applications.
4) *Semantic interoperability:* If in the future there will emerge blockchain ASs with differing applications (e.g., registry of assets, currency trading, etc.), what mechanisms are needed to convey to an external system the functional goal of a blockchain and its application-specific semantics?
   As a comparison, the HTTP protocol and remote procedure call (RPC) interprocess communications both run on the TCP/IP layer. However, these represent different resource access paradigms for different types of applications.
5) *Objective benchmarks for speed and performance:* How do external entities obtain information about the current performance/throughput of a blockchain system and what measure can be used to compare across systems?

B. Variety of Service Types

The second goal of the Internet architecture was the support for different types of services, distinguished by different speeds, latency, and reliability. The bidirectional reliable data delivery model was suitable for a variety of "applications" on the Internet, but each application required different speeds and bandwidth consumptions (e.g., remote login, file transfer, etc.). This understanding led to the realization early in the design of the Internet that more than one transport service would be needed, and that the architecture must support simultaneously transports wishing to tailor reliability, delay, or bandwidth usage. This resulted in the separation of TCP (that provided reliable sequenced data stream) from the IP that provided "best effort" delivery using the common building block of the *datagram*. The User Datagram Protocol [27] was created to address the need for certain applications that wished to trade reliability for direct access to the datagram construct.

For blockchain systems, we propose to reinterpret the notion of service types from the perspective of the different needs of various applications. We distinguish three basic types of service.

1) *Immediate direct confirmation:* This refers to applications that require the fastest confirmation from a specific destination blockchain system. The confirmation of the transaction must occur at the destination blockchain. As such, speed and latency are the primary concerns for these types of applications. This is summarized in Fig. 3(a). This situation is an analog of the classic TCP-based login service, in which the user performs login to a specific computer system and needs confirmation in as minimal delay as possible (e.g., milliseconds and seconds).
   Digital currency applications (e.g., currency trading system) are a typical example of cases needing direct and immediate confirmation with low latency.
2) *Delayed mediated confirmation:* This refers to applications that are satisfied with a "temporary" confirmation produced by a *mediating* blockchain system, which will then seek to "move" the transaction to its intended destination blockchain system. This is summarized in Fig. 3(b). The application will obtain two confirmations: the first would be a temporary confirmation from the mediating blockchain system, while the final confirmation will occur at the destination blockchain system at a later time. As such, there are two latency values corresponding to the two confirmations. The understanding here is that the application deems the first latency to be acceptable from a practical perspective, and that the second latency can be of a longer period of time (e.g., minutes). This is akin to the store-and-forward method used by classic electronic mail systems.
   An example of this type of application are noncritical notarization applications, which seek to record static (unchanging) data (e.g., birth date on a birth certificate) and which do not require low-latency confirmations.
3) *Multiparty mediated confirmation:* This scenario is a multiparty variation of the single-party mediated case mentioned above. Here, two (or more) applications are seeking

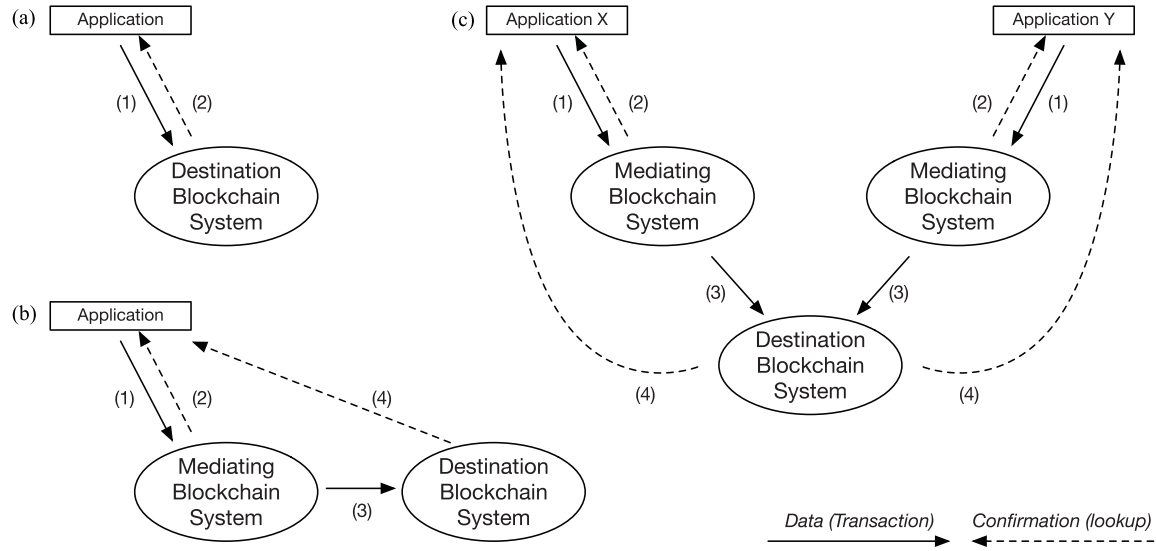

Fig. 3. (a)–(c) Service types based on different confirmation models.

to complete a common transaction at an agreed destination blockchain system, with the aid of settlement logic that executes at the destination blockchain system. Each of the applications are willing to accept a "temporary" confirmation produced by a mediating blockchain system, with the understanding that they will obtain a final confirmation from the destination blockchain system. This is summarized in Fig. 3(c).

This situation is akin to TCP-based messaging or chat servers (e.g., extensible messaging and presence protocol (XMPP)), in which two (or more) parties converge on a common server even though they each may have their own local servers.

C. Variety of Blockchain Systems

The third fundamental goal of the Internet architecture was to support a variety of networks, which included networks employing different transmission technologies at the physical layer (e.g., X.25, SNA, etc.), local networks and long-haul networks, and networks operated/owned by different legal entities. The *minimum assumption* of the Internet architecture—which is core to the success of the Internet as an interoperable system of networks—was that each network must be able to transport a *datagram* as the lowest unit common denominator. Furthermore, this was to be performed "best effort"—namely with reasonable reliability, but not perfect reliability.

For blockchain systems, we propose a reinterpretation of the minimal assumption as consisting of the following:

1) a common standardized *transaction format and syntax* that will be understood by all blockchain systems regardless of their respective technological implementation;
2) a common standardized *minimal operations set* that will be implemented all blockchain systems regardless of their technological choices.

The notion of a common transaction format is akin to the definition of the minimal IP datagram, which was first published in the 1974 milestone paper by Cerf and Kahn [4]. The operation involved in the datagram case is simple and is implicit in the datagram construct itself, namely that a set of bytes needs to be transmitted from one IP address to another. The situation is somewhat more complex in blockchain systems. Aside from the current common fields found in transactions in current systems (e.g., sender/receiver public keys, timestamp, and pointers), there is the question of *semantic meaning* of the operations intended by the op-code symbols. Some mathematical operations are clear (e.g., op codes for addition, multiplication, and hash function), but others may introduce some degree of ambiguity across systems.

Similar to the variety of technologies implementing LANs and local routing in the 1980s and 1990s, today, there are several technological aspect that differentiate one blockchain system from another.

1) *Governance model:* The term "governance" in the context of blockchain systems is typically used to refer to the combination of: a) the human-driven policies for the community of participants; b) the rules of operations that are encoded within the blockchain software and hardware fabric itself; and c) the intended application of the blockchain, which is often expressed as the "smart contracts" (stored procedures available on nodes) that are application specific.
2) *Speed of confirmation:* The speed (or "throughput") of a blockchain system refers to the confirmation speed, based on the population size of the participating nodes and other factors.
3) *Strength of consensus:* An important consideration is the size of the population of nodes (i.e., entities contributing to the consensus) at any given moment and whether this information is obtainable. Obtaining this information maybe challenging in systems, where nodes are either anonymous

or perhaps unobtainable by external entities in the case of permissioned systems.

4) *Degrees of permissionability:* Currently, the permissionless/permissioned distinction refers to the degree to which users can participate in the system [23]. Interoperability across permissioned blockchains poses additional questions with regard to how data recorded on the ledger can be referenced (referred to or "pointed to") by transactions in a foreign domain (i.e., another blockchain system).
5) *Degrees of anonymity:* There are at least two degrees of anonymity that is relevant to blockchain systems. The first pertains to the anonymity of end users (i.e., identity anonymity [28]–[30]), and the second is the anonymity of the nodes participating in processing transactions (e.g., nodes participating in a given consensus instance). Combinations are possible, such as where a permissioned system may require all consensus nodes to be strongly authenticated and identified, but allows for end users to remain permissionless (and even unidentified/unauthenticated).
6) *Cybersecurity and assurance levels of nodes:* The robustness of a blockchain system consisting of a P2P network of nodes is largely affected by the security of the nodes that make up the network. If nodes are easily compromised directly (e.g., hacks) or via indirect means (e.g., dormant viruses), the utility of the blockchain system degrades considerably [26].

IV. Gateways for Interoperability and Manageability

As mentioned previously, similar to the Internet architecture consisting of a network of ASs, the future blockchain technology may evolve to becoming a network of interconnected blockchain systems—each with differing internal consensus protocols, incentives mechanisms, permissions, and security-related constraints. Key to this interconnectivity is the notion of *blockchain gateways*. In this section, we discuss the potential use of blockchain gateways to provide interoperability and interconnectivity across different blockchain systems and service types.

Interoperability becomes a complex matter when transactions in permissionless blockchains (publicly readable ledgers) interact with permissioned (private) blockchains, where transaction entries on the ledger may reveal confidential information and, therefore, considered to be private. The use-case examples typically involve interactions between ledgers that record factual existential information about a given asset or object and ledgers that record legal ownership of that asset or object.

A. Intradomain and Interdomain Nodes

Similar to a routing AS being composed of one or more (possibly nested) routing domains, we propose viewing a blockchain system as consisting of one or more *ledger management domains*. Thus, just as routers in a routing domain operate one or more routing protocols to achieve best routes through that domain, *nodes* in a blockchain domain contribute to maintaining a shared ledger by running one or more *ledger management protocols* (e.g., consensus algorithms and membership management) to achieve stability and fast convergence (i.e., confirmation throughput) of the ledger in that domain.

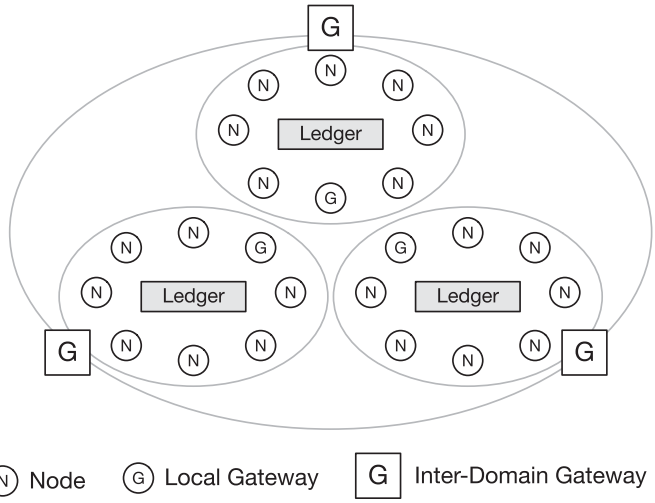

Fig. 4. Blockchain AS, domains, and gateways.

Nodes could, therefore, be classified from the perspective of ledger management as operating either intradomain or interdomain. Fig. 4 illustrates the concept, showing one blockchain AS, with three local domain blockchains, each managing a distinct ledger.

1) *Intradomain nodes:* These are nodes and other entities whose main task is maintaining ledger information and conducting transactions within one domain. Examples includes nodes that participate in consensus computations (e.g., full mining nodes in Bitcoin [17]), nodes that "orchestrate" consensus computations (e.g., Orderers and Endorsers in Hyperledger Fabric [19]), and nodes that perform validations only (e.g., Validators in Ripple [31]).
2) *Interdomain nodes:* These are nodes and other entities whose main task is dealing with cross-domain transactions involving different blockchain ASs. We refer to these nodes as *interdomain gateways*.

Although Fig. 4 shows a small number of nodes G to be designated as interdomain nodes, ideally, all nodes N in a given blockchain AS should have the capability (i.e., correct software, hardware, and trusted computing base) to become an interdomain gateway. This allows dynamic groups (subsets) of the population of nodes to become *gateway groups* that act collectively on behalf of the blockchain system as a whole [32]. In the remainder of this paper, we will denote interdomain gateways simply as "gateways."

B. Defining the Perimeter for Blockchain ASs

In the history of routing on the Internet, the emergence and evolution of the concept of ASs was driven partly by the need to manage networks. Among others, the owners and operators of networks needed to define their network physical perimeter, deploy administrative controls over the networks, and legally understand the areas of business responsibilities and liabilities. This arrangement provided the freedom on the part of the operators to design the routing topologies according to their business needs, applying different protocols, tools, and devices in each

domain. The result is that the physical perimeter of each network is clearly demarcated, without any ambiguities with regard to the legal ownership of each AS.

The situation is somewhat more complex in blockchain systems, which employ a P2P network of nodes in a geographically distributed topology, and where the participation of nodes are dynamic over time. One revolutionary aspect of the Bitcoin system [17] is its openness for any person or entity to participate in the act of mining by independently and anonymously deploying CPU cycles to compute the proof of work (consensus) algorithm. As such—and in contrast to routing domains—the perimeter of the Bitcoin network of nodes is not a physical or geographical one but rather an a computation-participatory one. This independence and anonymity meant that it is difficult or even impossible to know how many nodes (and which nodes) actually spent CPU cycles (successfully or not) in computing a given instance of the proof of work. Although unauthenticable anonymous identities maybe useful in some scenarios, in the context of the Bitcoin system. it may allow certain entities (e.g., state-sponsored actors) to amass computing power in a large mining pool and to "weaponize" that hash power at the opportune moment.

The future of blockchain ASs may evolve into an interconnected set of autonomous and independent blockchain systems, each with its own interior protocols, entities, and systems, and where each system's perimeter is defined along one or more of the following axis.

1) *Degree of identifiable and authenticable participation:* The "membership" of a node and entity in a blockchain AS may be defined as the potential for that node or entity to positively (or negatively) impact the community in a considerable way.
   In tightly permissioned blockchain systems, the organization or community that oversees the operations of the system may demand that all member nodes (i.e., legal owners of nodes) not only register their identities, but also report their participation in one measure or another (e.g., participated in a mining instance). For example, in an enterprise (single organization) privately owned blockchain system, all nodes are legally owned by the organization, and thus, the degree of participation is fully controlled by the organization.
   In a multiorganization consortium arrangement, the consortium may require all nodes belonging to consortium members to be identified beforehand (i.e., registered), but a node's actual participation in each consensus computation may be at the discretion of the member. Thus, the consortium as a collective may wish to ensure that no unknown or rogue node affects the consortium as a whole but allows each member to control their own resources.
   To this end, approaches using anonymous-verifiable identity schemes [29], [33] may be used to offer some degree of anonymity to the nodes. Similarly, methods to prove participation in computation can be devised based on schemes that use a combination of the consensus algorithm, a periodic reporting of hardware internal state (e.g., Trusted Platform Module (TPM) registers [34] and Quote protocol [35]), and secure multiparty computation techniques [36].
2) *Degree of trust and assurance:* Another factor related to perimeters and membership is the degree of *provable trust* each node can attain and can convey to other nodes and entities. The idea here is that the ability of a node to perform its tasks with high assurance (e.g., perform proof of work, safeguard its private keys, etc.) becomes input into the decision as to whether to accept the computation results of that node.
   This factor is notably important in the multiorganization consortium arrangement. The aspect of provable assurance may determine the acceptability internally of the results of the consensus computations by the member nodes. For example, in networks that deal with high-value transactions, the consortium may require its members to deploy trusted computing technologies that convey technical trust.
   In this context, it is useful to revisit some key architectural designs of the TPM from the late 1990s, which provided a basic understanding on trustworthiness [37], [38]. Reusing some of the concepts in trusted computing, a node can be considered to exhibit *technical trust* if it: a) operates unhindered and shielded from external influences or interference; b) operates for a well-defined task; and c) has the ability to report results of its computations and its internal status truthfully.
3) *Business model of the organization or community:* The business purpose of a blockchain AS may determine the degree of required identifiable and authenticable participation, as well as the minimal required trust and assurance. For example, a blockchain system for supply-chain management of components for a defense organization [39] has a different set of constraints compared to a blockchain used for supply-chain management of consumer goods [40].
   Similarly, a consortium of organizations whose goal is high-speed trading in digital assets using blockchain technology has different business purpose than a consortium of music publishers seeking greater accessibility to rights data in a global music ecosystem [41].

C. Use Case: Interdomain Transactions

To illustrate and aid discussion, we use a simple example shown in Fig. 5, in which an asset recorded in blockchain system BC1 is to be transferred to blockchain system BC2. Both blockchains BC1 and BC2 are permissioned/private blockchain systems.

In Fig. 5, User A with Application X has his or her asset ownership (e.g., land title deed) recorded on the ledger inside blockchain BC1. The local transaction identifier for this ledger entry is $Tx1_{\text{privateID}}$. User A wishes to transfer legal ownership of the asset (e.g., sell) to a different User B running Application Y. However, User B requires that the asset be "moved" to the blockchain BC2 and be authoritatively recorded on the ledger of BC2. This would allow User B to later sell the asset locally in blockchain BC2 to other users in BC2. Note that being private

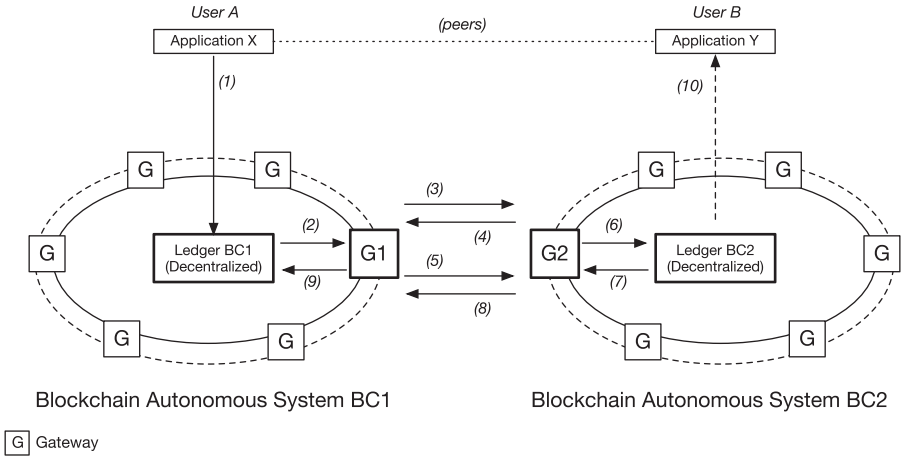

Fig. 5. Example of interdomain transaction across two blockchain systems.

blockchain systems, none of the gateways or nodes in BC1 can directly read/write to BC2, and vice versa.

The following is a high-level summary of the transaction flows between BC1 and BC2. In step 1, User A initiates the transfer to User B by submitting a (candidate) transaction to BC1, with the recipient address being the public key of User B in BC2 (e.g., $\text{pubkey}_B/\text{BC2}$). Because the destination blockchain BC2 is a foreign system, only the gateway nodes in BC1 are permitted or have the capability to process this transaction. In step 2, gateway G1 selects the candidate (unprocessed) transaction of User A in BC1 and proceeds to process the transaction. Seeing that the pending transaction is destined for $\text{pubkey}_B/\text{BC2}$, gateway G1 locates one or more gateways G2 in BC2 and proceeds perform trust negotiations with G2 (see Section IV-E).

In step 3, because BC1 is a private system, gateway G1 has to mask the private identifier value $Tx1_{\text{privateID}}$ with a new public value $Tx1_{\text{publicID}}$. G1 has to persistently maintain this table of mappings ($Tx1_{\text{privateID}}$ and $Tx1_{\text{publicID}}$). In the future, gateway G1 must provide a means to resolve the public value $Tx1_{\text{publicID}}$ back to the internal private value $Tx1_{\text{privateID}}$ should that be required (see Section IV-D). In steps 4 and 5, (multiround) gateways G1 and G2 must establish trust by executing a key establishment protocol that includes the exchange of keying parameters, hardware root of trust certificates (e.g., attestation identity key (AIK) certificates in TPM [34]), hardware status reports (e.g., Quote protocol reports [35]), and other relevant trust establishment parameters.

In step 6, after pairwise technical trust has been established between gateways G1 and G2, gateway G2 proceeds to submit a new local transaction with identifier $Tx2_{\text{privateID}}$ into the ledger of BC2 addressed pubkey_B. This local transaction references the asset (in BC1) identified by the public value $Tx1_{\text{publicID}}$. In effect, gateway G2 is "registering" this asset $Tx2_{\text{privateID}}$ as belonging to User B with public key pubkey_B. In step 7, confirmation has been achieved in the ledger of BC2. Gateway G2 needs to indirectly report this confirmed status of $Tx2_{\text{privateID}}$ to gateway G1. As such, G2 has to mask local transaction identifier $Tx2_{\text{privateID}}$ with a new public identifier $Tx2_{\text{publicID}}$. Gateway G2 must henceforth maintain a persistent mapping between $Tx2_{\text{publicID}}$ and $Tx2_{\text{privateID}}$.

In step 8, gateway G2 issues a signed assertion to G1 stating that the asset with the transaction identifier $Tx2_{\text{publicID}}$ has been confirmed on ledger BC2 and includes a hash of the private identifier $Tx2_{\text{privateID}}$ in the assertion. In step 9, upon seeing the signed assertion from G2, gateway G1 proceeds to submit a new "invalidation" transaction in BC1, essentially marking that the asset previously known as $Tx1_{\text{privateID}}$ has been moved to BC2. The invalidation transaction also includes a reference to the new home of the asset, namely $Tx2_{\text{publicID}}/\text{BC2}$. It should also include a hash of the signed assertion from G2. Finally, in step 10, User B is able to see the confirmed transaction $Tx2_{\text{privateID}}/\text{BC2}$, while User A sees that $Tx1_{\text{privateID}}/\text{BC1}$ is also being confirmed.

Note that the above use case is an abstract example only. Many variations are possible for these flows, including the incorporation of commitment protocols (e.g., two-phase commit) in steps 2–9.

D. Visibility and Referenceability of Transaction Identifiers

One key potential use of gateways in the context of blockchain interoperability is to provide some degree of control over the visibility (i.e., read access) of transaction identifiers residing on the ledger of the blockchain system.

1) *Masking of private identifiers:* In cases of private/permissioned blockchain systems, where all transaction information on the confirmed blocks on the ledger is considered confidential information (including the transaction-identifiers), a gateway may offer the possibility to support the notion of identifier "masking." As discussed in Section IV-C, a substitute transaction identifier is used for external referenceability in a persistent manner. In a sense, this is akin to the network address translation found in network address translation (NAT) devices and dual-stack IPv4/IPv6 routers.
2) *Resolution of private identifiers:* If identifier masking or translation is used, a corresponding resolution function can be implemented at gateways. Thus, in the example of Section IV-C, after the asset has been moved from BC1 to BC2, whenever one of more nodes in BC1 obtains a query regarding $Tx1_{\text{publicID}}/\text{BC1}$, the node can

forward this query to one or more of the gateways in BC1, which collectively share the mapping table. In turn, one of these gateways in BC1 can remap the query into $Tx2_{\text{publicID}}$/BC2 and redirect the query to one or more of the gateways in BC2. This resolver role is similar to the domain name system (DNS) systems, and also to the online certificate status protocol (OCSP) Responder model in public key infrastructure (PKI) [42], which can report on the status of a public key in an X.509 certificate issued by the Certificate Authority who operates the Responder service.

E. Interdomain Trust Establishment

A second potential use of gateways in the context of blockchain interoperability is to support the establishment of trust (i.e., technical trust) across blockchain ASs. We believe there is a promising role for trusted hardware to implement many of the function of the gateways. As mentioned previously, ideally, all nodes in a given blockchain AS should possess the relevant trusted hardware and software to allow them to take on the role of gateways as required.

Examples of trusted hardware include the TPM [34] with its various roots of trust for measurement, storage, and reporting. The first successful version was TPM v1.2 that supported a "one-size-fits-all" approach that primarily targeted the PC market. A second-generation TPM v2.0 expanded trusted computing features to better support vertical markets. TPMv2.0 introduced platform-specific profiles that define mandatory, optional, and excluded functionality for PC Client, Mobile, and Automotive-Thin platform categories. Platform-specific profiles allow TPM vendors flexibility in implementing TPM features that accommodates a specific market. Additionally, TPMv2.0 supports three key hierarchies, for storage, platform, and endorsement. Each hierarchy can support multiple keys and cryptographic algorithms. We believe that TPM v2.0 profiles for trusted gateways could be developed for the blockchain infrastructure market.

Another example of trusted hardware is the *Software Guard Extensions* (SGX) from Intel Corporation [35]. The SGX offers another perspective on trusted computing base, where a trusted environment exists within a user process called an *Enclave*. The SGX TCB consists of hardware isolated memory pages, CPU instructions for creating, extending, initializing, entering, exiting, and attesting the enclave, and privileged CPU modes for controlling access to enclave memory. A second-generation SGX (see [43]) added support for dynamic memory management, where enclave runtimes could dynamically increase or decrease the number of enclave pages.

There are multiple steps to establish measurable technical trust that can be input into legal frameworks in the context of peering. Some of these are as follows.

1) *Mutual verification of gateway device identities:* Prior to interacting, two gateways belonging to separate blockchain AS must mutually verify their device identities (e.g., AIK certificates in TPM).
2) *Mutual attestation of gateway device status:* As part of trust establishment, each gateway may be required to attest to its hardware and software stack, as well as the current state of some of its hardware registers (e.g., Quote protocol [34], [35]).
3) *Mutual session key establishment:* For use cases involving session keys, the gateways have the additional task of negotiating the keying parameters and establish the relevant session keys.
4) *Mutual reporting of transaction settlement:* In use cases involving one (or both) private blockchains, an additional requirement could be the signing of assertions using a gateway's device keys.

F. Peering Points for Peering Business Agreements

The third potential use of gateways in the context of blockchain interoperability is to serve as the *peering points* identified within peering agreements or contracts. In the case of various ISPs that make up the Internet, peering agreements are contracts that define various interconnection aspects (e.g., traffic bandwidth, protocols, etc.) as well as fees ("settlements") and possible penalties. For the interoperability of autonomous blockchain systems, a notion similar to peering agreements must be developed that possess features specifically for blockchain technology and the governance model used by the systems. Peering agreements should include, among others, the following.

1) *Identification of gateways chosen as peering points:* A blockchain peering agreement should require the clear identification of gateways, which are permitted to peer with other gateways. This agreement may specify the device certificates, hardware and software manifest (e.g., hash of manifest), root certificates, device status attestations, and so on.
2) *Specify the minimal trust establishment mechanisms and parameters:* A peering agreement should specify the trust negotiation and establishment protocols, the respective known parameters (e.g., size of key parameters), the key management protocols, standards compliance required, minimal assurance level required, and others.
3) *Specify warranties and liabilities:* Similar to peering agreements for ISPs and Certificate Practices Statement for certificate authorities, blockchain peering agreements should clearly identify the liabilities of parties (e.g., in monetary terms) in negative or catastrophic scenarios (e.g., gateway is compromised).

V. Conclusion

The fundamental goals underlying the Internet architecture has played a key role in determining the interoperability of various networks and service types, which together compose the Internet as we know it today. Interoperability is key to survivability. A number of design principles emerged from the evolution of internet routing in the 1970s and 1980s, which ensured the scalable operation of the Internet over the last three decades.

We believe that a similar design philosophy is needed for interoperable blockchain systems. The recognition that a blockchain system is an AS is an important starting point that allows notions such as reachability, referencing of transaction data

in ledgers, scalability, and other aspects to be understood more meaningfully—beyond the current notion of throughput ("scale"), which is often the sole measure of performance used with regards to many blockchain systems today.

Furthermore, interoperability forces a deeper rethinking into how permissioned and permissionless blockchain systems can interoperate without a third party (such as an exchange). A key aspect is the semantic interoperability at the value level and at the mechanical level. Interoperability at the mechanical level is necessary for interoperability at the value level but does not guarantee it. The mechanical level plays a crucial role in providing technological solutions that can help humans in quantifying risk through the use of a more measurable notion of technical trust. Human agreements (i.e., legal contracts) must be used at the value level to provide semantically compatible meanings to the constructs (e.g., coins and tokens) that circulate in the blockchain system.

REFERENCES

[1] S. Haber and W. Stornetta, "How to time-stamp a digital document," in *Proc. Conf. Theory Appl. Cryptography*, 1991, pp. 437–455.

[2] D. Bayer, S. Haber, and W. Stornetta, "Improving the efficiency and reliability of digital time-stamping," in *Sequences II: Methods in Communication, Security and Computer Science*, R. Capocelli, A. DeSantis, and U. Vaccaro, Eds. New York, NY, USA: Springer, 1993, pp. 329–334.

[3] D. Clark, "The design philosophy of the DARPA Internet protocols," *ACM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 106–114, Aug. 1988.

[4] V. G. Cerf and R. E. Kahn, "A protocol for packet network intercommunication," *IEEE Trans. Commun.*, vol. COM-22, no. 5, pp. 637–648, May 1974.

[5] J. Abbate, *Inventing the Internet*. Cambridge, MA, USA: MIT Press, 1999.

[6] J. Saltzer, D. Reed, and D. Clark, "End-to-end arguments in system design," *ACM Trans. Comput. Syst.*, vol. 2, no. 4, pp. 277–288, Nov. 1984.

[7] S. Kent and R. Atkinson, "Security architecture for the Internet protocol," RFC2401, Nov. 1998. [Online]. Available: http://tools.ietf.org/rfc/rfc2401.txt

[8] D. Harkins and D. Carrel, "The Internet key exchange (IKE)," RFC2409, Nov. 1998. [Online]. Available: http://tools.ietf.org/rfc/rfc2409.txt

[9] T. Dierks and C. Allen, "The TLS Protocol Version 1.0," RFC2246, Jan. 1999. [Online]. Available: http://tools.ietf.org/rfc/rfc2246.txt

[10] J. Hawkinson and T. Bates, "Guidelines for creation, selection, and registration of an autonomous system (AS)," RFC1930, Mar. 1996. [Online]. Available: http://tools.ietf.org/rfc/rfc1930.txt

[11] G. Malkin, "RIP Version 2," RFC2453, Nov. 1998. [Online]. Available: http://tools.ietf.org/rfc/rfc2453.txt

[12] J. Moy, "OSPF Version 2," RFC2328, Apr. 1998. [Online]. Available: http://tools.ietf.org/rfc/rfc2328.txt

[13] K. Lougheed and Y. Rekhter, "Border gateway protocol (BGP)," RFC1105, Jun. 1989. [Online]. Available: http://tools.ietf.org/rfc/rfc1105.txt

[14] ARIN, "American Registry for Internet Numbers—Autonomous System Numbers (asn.txt)," 2018. [Online]. Available: https://www.arin.net

[15] J. H. McFadyen, "Systems network architecture: An overview," *IBM Syst. J.*, vol. 15, no. 1, pp. 4–23, 1976.

[16] S. Wecker, "DNA: The digital network architecture," *IEEE Trans. Commun.*, vol. COM-28, no. 4, pp. 510–526, Apr. 1980.

[17] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[18] V. Buterin, "Ethereum: A next-generation cryptocurrency and decentralized application platform," Bitcoin Magazine, Jan. 2014. [Online]. Available: https://bitcoinmagazine.com/articles/ethereum-next-generation-cryptocurrency-decentralized-application-platform-1390528211/

[19] E. Androulaki *et al.*, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, 2018, Art. no. 30.

[20] R3CEV, "R3," 2018. [Online]. Available: https://www.r3.com

[21] A. Lipton and A. Pentland, "Breaking the bank," *Sci. Amer.*, vol. 318, no. 1, pp. 26–31, 2018.

[22] A. Lipton, T. Hardjono, and A. Pentland, "Digital Trade Coin (DTC): Towards a more stable digital currency," *J. Roy. Soc. Open Sci.*, Aug. 2018. [Online]. Available: https://doi.org/10.1098/rsos.180155

[23] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," NIST Draft NISTIR 8202, Jan. 2018. [Online]. Available: https://csrc.nist.gov

[24] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proc. 18th Int. Conf. Financial Cryptography Data Secur.*, Mar. 2014, pp. 436–454.

[25] A. Gervais, G. O. Karame, V. Capkun, and S. Capkun, "Is bitcoin a decentralized currency?" *IEEE Secur. Privacy*, vol. 12, no. 3, pp. 54–60, May/Jun. 2014.

[26] B. Schneier, "There's no good reason to trust blockchain technology," *Wired Mag.*, Feb. 2019. [Online]. Available: https://www.wired.com/story/theres-no-good-reason-to-trust-blockchain-technology/

[27] J. Postel, "User datagram protocol," RFC0768, Aug. 1980. [Online]. Available: http://tools.ietf.org/rfc/rfc0768.txt

[28] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–88, Feb. 1981.

[29] J. Camenisch and E. Van Herreweghen, "Design and implementation of the Idemix anonymous credential system," in *Proc. 9th ACM Conf. Comput. Commun. Secur.*, 2002, pp. 21–30.

[30] T. Hardjono and N. Smith, "Cloud-based commissioning of constrained devices using permissioned blockchains," in *Proc. 2nd ACM Int. Workshop IoT Privacy, Trust, Secur.*, 2016, pp. 29–36.

[31] D. Schwartz, N. Youngs, and A. Britto, "The ripple protocol consensus algorithm," Ripple Inc., San Francisco, CA, USA, Tech. Rep., 2014. [Online]. Available: https://arxiv.org/abs/1802.07242

[32] T. Hardjono and N. Smith, "Decentralized trusted computing base for blockchain infrastructure security," submitted for publication.

[33] T. Hardjono, D. Shrier, and A. Pentland, *Trust::Data: A New Framework for Identity and Data Sharing*. Cambridge, MA, USA: Visionary Future Press, 2017.

[34] Trusted Computing Group, "TPM Main—Specification Version 1.2," Trusted Computing Group, TCG Published Specification, Oct. 2003. [Online]. Available: http://www.trustedcomputinggroup.org/resources/tpm_main_specification

[35] F. Mckeen *et al.*, "Innovative instructions and software model for isolated execution," in *Proc. 2nd Int. Workshop Hardware Archit. Support Secur. Privacy*, Tel-Aviv, Israel, Jun. 2013, Art. no. 10. [Online]. Available: https://sites.google.com/site/haspworkshop2013/workshop-program

[36] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or a completeness theorem for protocols with honest majority," in *Proc. 19th ACM Symp. Theory Comput.*, 1987, pp. 218–229.

[37] Trusted Computing Group, "TPM Main—Part 1: Design Principles—Specification Version 1.2," Trusted Computing Group, TCG Published Specification, Oct. 2003. [Online]. Available: http://www.trustedcomputinggroup.org/resources/tpm_main_specification

[38] T. Hardjono and N. Smith, "TCG infrastructure reference architecture for interoperability (Part 1)—Specification version 1.0 Rev 1.0," Trusted Computing Group, TCG Published Specification, Jun. 2005. [Online]. Available: http://www.trustedcomputinggroup.org/resources

[39] C. Nissen, J. Gronager, R. Metzger, and H. Rishikof, "Deliver uncompromised: A strategy for supply chain security and resilience in response to the changing character of war," MITRE Corp., McLean, VA, USA, Rep., Aug. 2018.

[40] M. del Castillo, "IBM-Maersk blockchain platform adds 92 clients as part of global launch," *Forbes Mag.*, Aug. 2018.

[41] P. Panay, A. Pentland, and T. Hardjono, "Open music: Why success of the music modernization act depends on open standards," *Billboard Mag.*, Oct. 2018. [Online]. Available: https://www.billboard.com/articles/business/8482056/mma-op-ed-open-music-initiative-open-standards

[42] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet public key infrastructure online certificate status protocol—OCSP," RFC2560, Jun. 1999. [Online]. Available: http://tools.ietf.org/rfc/rfc2560.txt

[43] F. McKeen *et al.*, "Intel software guard extensions (Intel SGX) support for dynamic memory management inside an enclave," in *Proc. Workshop Hardware Archit. Support Secur. Privacy*, Seoul, South Korea, Jun. 2016, Art. no. 10. [Online]. Available: http://caslab.csl.yale.edu/workshops/hasp2016/program.html

Thomas Hardjono received the B.S. (Hons.) degree in computer science from the University of Sydney, Sydney, Australia, in 1987, and the Ph.D. degree in computer science and engineering from the University of New South Wales, Sydney, Australia, in 1991.

He is the Chief Technology Officer (CTO) of Connection Science and Technical Director of the MIT Trust-Data Consortium, Massachusetts Institute of Technology, Cambridge, MA, USA. For several years prior to this, he was the Executive Director of the MIT Kerberos Consortium. Over the past two decades, he has held various industry technical leadership roles, including a Distinguished Engineer with Bay Networks, a Principal Scientist with VeriSign PKI, and CTO roles at several start-ups. He has been at the forefront of several industry initiatives around identity, trust, and cybersecurity. He has authored several dozen technical papers, patents, and books covering cryptography, network security, identity, and blockchain security. His research interests include Internet of Things security, trusted computing, decentralized identity, personal data privacy, peer-to-peer networks, and blockchain systems.

Alexander Lipton received the M.S. (summa cum laude) and Ph.D. degrees in pure mathematics from Lomonosov Moscow State University, Moscow, Russia, in 1979 and 1982, respectively.

He is the Cofounder and Chief Technology Officer (CTO) of SilaMoney, Partner with Numeraire Financial, a Connection Science Fellow with the Massachusetts Institute of Technology, Cambridge, MA, USA, and a Visiting Professor of Financial Engineering with École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. He is a Board Advisory Board Member for several fintech startups. For 20 years, he held senior managerial positions with preeminent financial institutions, including Bank of America Merrill Lynch and Citadel Investment Group, and, in parallel, prestigious academic appointments with renowned universities, including Oxford University and New York University. He was a Full Professor of Mathematics with the University of Illinois and a Consultant with the Los Alamos National Laboratory. He has authored/edited eight books.

Prof. Lipton is the first recipient of the Quant of the Year Award.

Alex (Sandy) Pentland received the B.G.S. degree in mathematics and statistics from the University of Michigan, Ann Arbor, MI, USA, in 1976, and the Ph.D. degree in artificial intelligence from The Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1982.

He has helped create and direct MIT Media Lab, Massachusetts Institute of Technology, Cambridge, MA, USA, the Media Lab Asia, and the Center for Future Health. He is the Chair of World Economic Forum's Data Driven Development Council, the Academic Director of the Data-Pop Alliance, and a member of the Advisory Boards for Google, Nissan, Telefonica, the United Nations Secretary General, Monument Capital, and the Minerva Schools. He is among the most cited computational scientists in the world, and a pioneer in computational social science, organizational engineering, wearable computing (Google Glass), image understanding, and modern biometrics. His research has been featured in *Nature*, *Science*, and *Harvard Business Review*, as well as being the focus of TV features on BBC World, Discover, and Science channels. His most recent book is *Social Physics* (London, U.K.: Penguin Press, 2015). Over the years, he has advised more than 50 Ph.D. students.

Prof. Pentland was the recipient of the McKinsey Award from *Harvard Business Review* in 2013. In 2012, Forbes named him one of the "seven most powerful data scientists in the world," along with Google founders and the Chief Technology Officer of the United States.