Public-Permissioned blockchains as Common-Pool Resources

Jesus Ruiz

Public-Permissioned blockchains as Common-Pool Resources

Jesus Ruiz - hesus.ruiz@gmail.com - Version 0.2, 04-02-2020

Abstract

For many, using together the words **Public** *and* **Permissioned** *when referring to a blockchain network, as in Public-Permissioned Blockchain, is a misnomer. Until now the words Public and Permissioned have been used as opposites, where Permissioned is seen as synonymous with Private. However, a new kind of blockchain network is emerging, being pioneered by Alastria.* **Public-Permissioned** *blockchain networks bring a lot of value to the digitization of the productive economy of a country or a region like the European Union, complementing - but not replacing - the other types of networks, namely Public-Permissionless and Private-Permissioned (aka Private Consortiums).*

In this context Blockchain networks are socio-technical systems that, given their inherent difficulties to scale in different dimensions, can be considered as a kind of good with some scarce resources which require a proper governance model to avoid resource abuse and depletion.

Two classical models to manage scarce resources are the **state** *(government controlled) and the* **market** *(managed as private goods subject to offer and demand).*

We explore in this document a third way to manage that type of resorces, the **Common-Pool Resources (CPR)** *model.*

1. Introduction to Public-Permissioned blockchain networks

A **Public-Permissioned** blockchain network is a new type of network filling the gap between the Public-Permissionless networks (like Bitcoin or Ethereum) and the Private Consortium networks. See Figure 1 for a representation of this term, which is based on the taxonomy being specified in the governance group of ISO TC 307.

We use in this document the word **blockchain** in a generic sense, including a DLT.

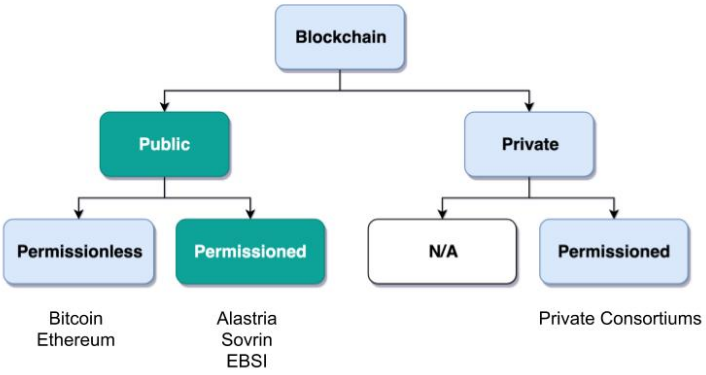

Figure 1. Blockchain taxonomy

A Public-Permissioned blockchain network **combines the permissioning from private consortiums with a decentralized governance model**, trying to achieve the best properties of both models. This is done in order to obtain features required for the implementation of many important use cases that can not t in any of the other two models.

It is important to understand that the above taxonomy does not apply to software (eg. Geth, Besu, Fabric, Corda, etc.) but to an actual operating network implemented with that software, and more specifically to its governance model. The taxonomy is very simple and so it can not include many nuances affecting different aspects related to the same concepts in different real-world implementations. However, this simplicity is very useful to discuss the main properties of the model, at least when comparing it to Public-Permissionless and Consortium networks.

The main aspects considered in the taxonomy are **permissioning** and **governance model**.

Permissioning

The permissioning feature allows us to bring some benefits from the Consortium arena:

- **Technical benefits**: we can use consensus algorithms better fitted for permissioned blockchains and which provide better performance and are more energy-efficient.
- **Compliance benefits**: participating nodes have well known real-world identities, facilitating compliance with regulations like GDPR or AML, especially if their location is restricted to a region with a common regulatory system.
- **Operational benefits**: easier to manage and implement crisis management.
- **Economic benefits**: the network does not require a cryptocurrency embedded in the consensus algorithm in order to incentivize miners. This makes the operation of the network very similar to the operation of any other infrastructure. The transaction costs for the participants can be made strictly proportional to the cost of operating the infrastructure, so they are very stable and do not depend on speculation in the market of an embedded cryptocurrency.

Decentralized governance model

On the other hand, having a **decentralized and transparent governance model** is critical for improving the level of trust and confidence in the network, especially for external users that do not participate directly (e.g. citizens and businesses who do not run a node), but instead access services indirectly via some other entity.

The concept of combining permissioning with a decentralized governance model is is represented in Figure 2.

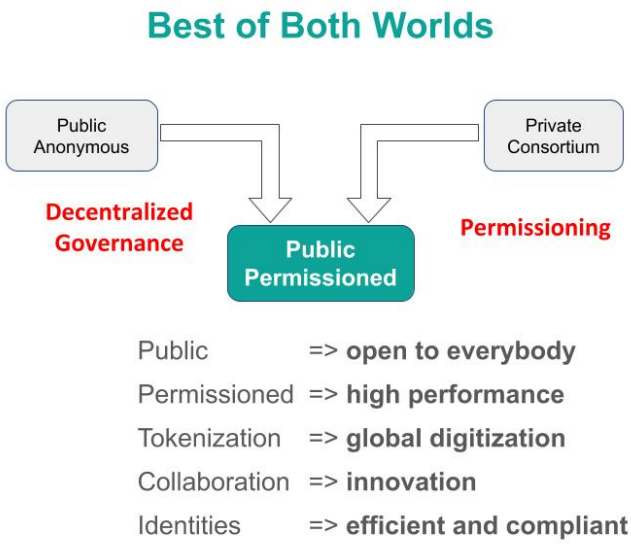

Figure 2. Best of both worlds

1.1. Public as in Public Services

The word **Public** has in this context a different meaning than the one used (improperly) until now.

A Public-Permissioned network is public in much the same way as most essential public services of a country like public health, public education or public roads. Using the analogy of those public services, we could say that those services are **permissioned** in the sense that citizens and entities must identify themselves. However, there are no artificial barriers of entry for citizens in order to access public health or public education. In the case of public roads, the criteria are more stringent but, in any case, they are objective and transparent: any car using the public roads has to display very clearly the license plate, which is sort of the identity of the car. Furthermore, anyone can drive a car in the public roads, provided she has a valid driver's license.

There are public goods that do not require "permissioning" in order to use them, but when those goods are scarce and subject to depletion if abused, then permissioning is required in order to ensure inclusion, fair access and usage, and sustainability of the resource.

1.2. Permissioning and Decentralized Governance Model

It can be argued that requiring the network to be permissioned reduces decentralization and increases the level of trust required by participants in the network.

Indeed, permissioning was initially applied in the blockchain space to create Private Consortium blockchain networks, where permissioning is used not only to verify the identities of participants, but also to create **barriers of entry** to external participants like for example to avoid competitors from entering the Consortium. Additionally, because the number of participants is low and typically from the same sector (eg. banks with banks), implementing the same use-case(eg. logistics) and normally highly regulated, the governance model of the Consortium is centralized. This is normally not a problem in those consortiums because the objective normally is to increase efficiency via a "virtual" shared database without requiring that the database be operated by a central entity.

This explains why many people associate permissioning with centralized and non-inclusive governance models in the blockchain arena.

In order to merit the name of Public-Permissioned, a special Governance Model is required in order to ensure inclusion, fair access and usage and sustainability of the network, and at the same time ensure that it is not controlled by a single entity or a cartel of entities. More concretely, this blockchain network could be considered as a new type of infrastructure with the following principles [Navarro2018]:

Non-discriminatory and open access

> Access is non-discriminatory even if it is not free, because pricing is determined using transparent mechanisms, typically cost-oriented. Access is open because everybody has the right to join and use the infrastructure according to the access rules.

Open participation

> Everybody has the right to join the community to participate in the construction, operation, provision and governance of the infrastructure. The network should be inclusive, open to participation of any entity independent of its size or sector of activity.

Such a governance model is critical in providing the required level of trust and confidence in the network from all participants. Running a network that is permissioned and at the same time public (in the sense of inclusive) and sustainable, presents many challenges that have to be addressed explicitly and proactively and are specific to this type of network and which do not appear in either Public-Permissionless or Consortium networks.

There are several specific instances of the governance model that can achieve these objectives, but in this document we focus on a specific governance model which arises from considering the blockchain network as a Common-Pool Resource (CPR) [Ostrom1990].

1.3. The Position of Alastria in the Trust Continuum

Even though in practice Public-Permissionless networks are more centralized than what they are normally assumed to be, from a theoretical point of view Alastria (and in general Public-Permissioned networks) can be positioned in the so-called *Trust Continuum* as depicted in the following figures:

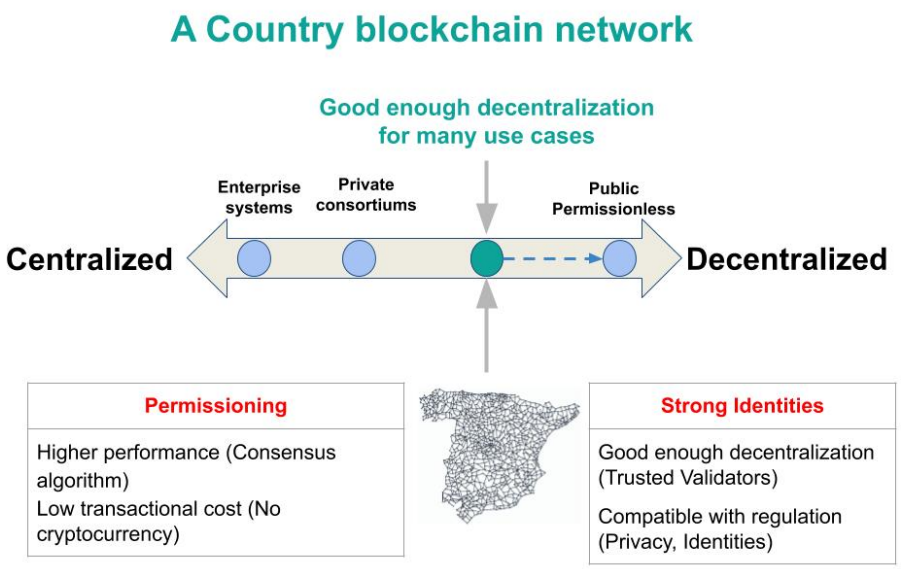

Figure 3. A country blockchain network

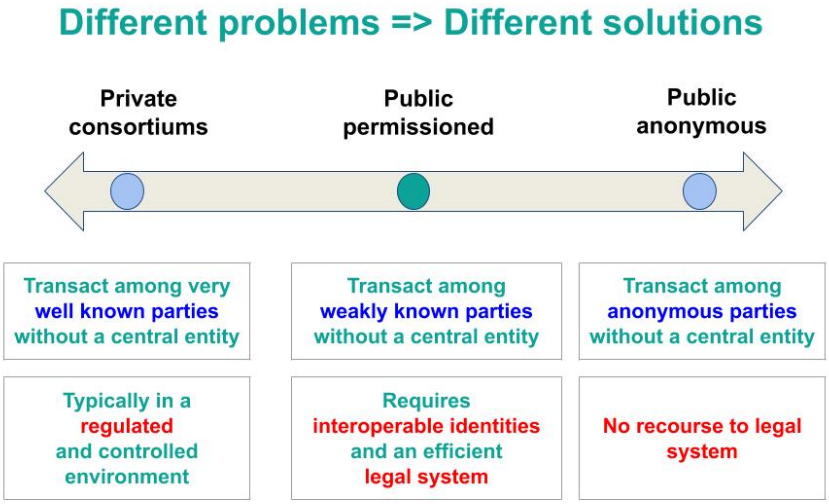

Figure 4. One Size Does Not Fit All

2. The blockchain as a Common-Pool Resource (CPR)

As exemplified in the Blockchain trilemma [Buterin2014], blockchain networks can be considered as a technical resource that can not be scaled easily. If we consider for example the throughput (number of transactions per unit of time that the network can process globally for all users), we can see that with a given blockchain technology this resource does not scale easily. This is in contrast with other infrastructures like the Internet backbone, where the bandwidth can be scaled by adding communication lines in parallel. Or in traditional applications, adding more machines or bigger ones can scale the number of transactions per second.

In this sense, a Public-Permissioned blockchain network can be considered as a communal resource like the ones described by Elinor Ostrom, Nobel Prize in Economics 2009 [Ostrom1990], where the resource to be managed is the **transactional capability** of the network, making sure at the same

time that the network is **safe** and **always available**. Ostrom's studies focused on how communities manage to successfully govern communal resources by revisiting Hardin's influential article on *The tragedy of the commons* [Hardin1968].

This governance model is different from the two standard ways of managing private goods or public goods and is the most efficient for goods that have the property of subtractability, like private goods, but they share the difficulty of exclusion with public goods. This concept is represented in Figure 5.

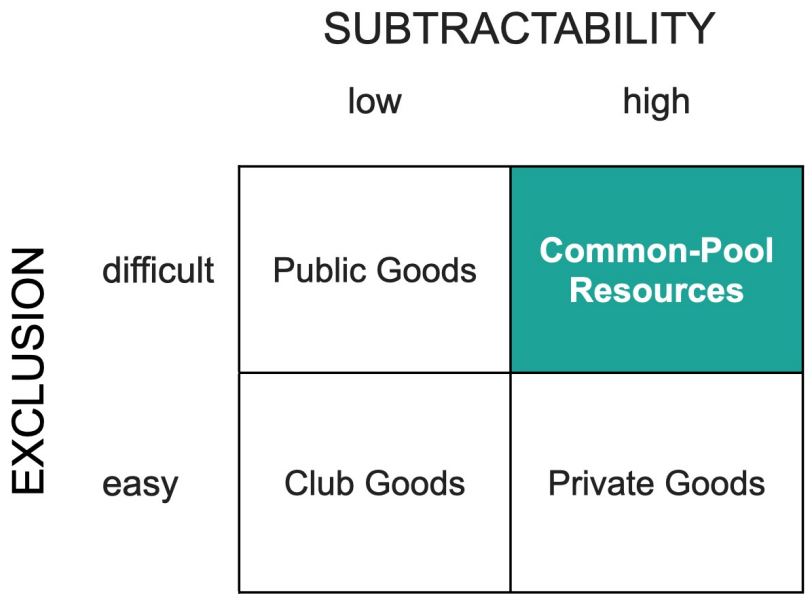

Figure 5. Common-Pool Resources

2.1. From natural resources to socio-technical infrastructures

Until now, the CPR model has been applied almost exclusively to natural resources, as fisheries, forests or irrigation systems. And in most cases, there are fundamental limitations in the size or scale of those resources in order to be able to apply effectively the CPR model (essentially, having to do with the required flow of information and trust. These limitations in geographic distribution of resources appear also when applied to *clasical* technical infrastructures (that is, non-blockchain ones), because it is very difficult to achieve the required level of trust among participants that is required for the successful implementation of the CPR model and rules. However, a unique property of a blockchain network with respect to all other Common-Pool Resources (natural resources or classical technical infrastructures) is the ability to encode some governance rules using the programmable nature of the blockchain, making the enforcement of the rules not only transparent but also automatic and immutable (actually, the rules can be modified with the consensus of the community; the word immutable is used to indicate that nobody can unilaterally modify them). This is what we call **"on-chain governance"**.

The literature has a small number of documents describing governance models and their automated implementation based on the blockchain [DavidsonEtAl2016], but in general they are at

the application (dApp) level, and they **assume the existence** of a blockchain network with the appropriate characteristics [RozasEtAl2018]. In this document we are instead interested on the governance model required for the management and operation of a blockchain network infrastructure which is Public-Permissioned, according to the definition above. That is, we focus on the **governance of the blockchain** instead of the **governance based on the blockchain**. And specifically, on the on-chain governance of the blockchain network infrastructure in contrast to the off-chain governance processes, even though we have to consider the whole governance process in order to derive the properties of the blockchain infrastructure.

2.2. The eight principles for managing a Commons

Before entering into the specifics of blockchain, let's summarize the eight principles for efficiently managing Common-Pool Resources, as described in [Ostrom1990]:

	Principle	Description
1.a	**User boundaries**	Clear boundaries between legitimate users and nonusers must be clearly defined.
1.b	**Resource boundaries**	Clear boundaries are present that define a resource system and separate it from the larger surrounding environment.
2.a	**Congruence with local conditions**	Appropriation and provision rules are congruent with local social and environmental conditions.
2.b	**Appropriation and provision**	The benefits obtained by users from a common-pool resource (CPR), as determined by appropriation rules, are proportional to the amount of inputs required in the form of labor, material, or money, as determined by provision rules.
3	**Collective-choice arrangements**	Most individuals affected by the operational rules can participate in modifying the operational rules.
4.a	**Monitoring users**	Monitors who are accountable to the users monitor the appropriation and provision levels of the users.
4.b	**Monitoring the resource**	Monitors who are accountable to the users monitor the condition of the resource.
5	**Graduated sanctions**	Appropriators who violate operational rules are likely to be assessed graduated sanctions (depending on the seriousness and the context of the offense) by other appropriators, by officials accountable to the appropriators, or by both.
6	**Conflict-resolution mechanisms**	Appropriators and their officials have rapid access to low-cost local arenas to resolve conflicts among appropriators or between appropriators and officials.

	Principle	Description
7	**Minimal recognition of rights to organize**	The rights of appropriators to devise their own institutions are not challenged by external governmental authorities.
8	**Nested enterprises**	When the system is very complex, appropriation, provision, monitoring, enforcement, conflict resolution, and governance activities are organized in multiple layers of nested enterprises.

2.3. The CPR principles applied to a blockchain

When applying Ostrom's CPR principles to a Public-Permissioned blockchain network, we see that there is a potential to automate the execution and enforcement of some of the principles in a way which is impossible for any other type of CPR nlike natural resources. A summary can be found in the following table.

	Principle	Description
1.a	**User boundaries**	Self-Sovereign Identities (associated to legal identities) both for natural and juridical persons.
1.b	**Resource boundaries**	Decentralized permissioning of nodes via Smart Contracts connected to Trusted Third Parties (TTPs) and other official Registries and Regulatory bodies in the country (eg. the Spanish Business Registry for normal businesses, or the Ministry of Education for Universities).
4.a	**Monitoring users**	Using Gas to control resource usage by accounts (self-monitoring).Need transaction origin traceability (enode that injected tx)
4.b	**Monitoring the resource**	Monitor the Consensus execution (eg. report detectable Crash and Byzantine behavior) in a transparent way
5	**Graduated sanctions**	Automated proactive and reactive management of the Consensus set via Smart Contracts complemented with off-chain sanctioning.
6	**Conflict-resolution mechanisms**	At the lowest level of the operation of the network, the same mechanisms used for monitoring and graduated sanctions are used for automated arbitration of conflicts arising among members (eg. non-compliance to the Service Level Objectives defined in the operational policies of the network).

3. Consensus algorithms in Public-Permissioned blockchains

A blockchain is an append-only, sequential, linked, data structure replicated over a peer-to-peer network, where transactions are stored and grouped to form new blocks. Participants of the network (peers) achieve distributed consensus on the validity of and the ordering of transactions.

Consensus is a set of rules and procedures that allow a blockchain system to maintain and update the distributed ledger and to ensure the trustworthiness of the records in the ledger. This trustworthiness – often referred to as safety – is the systems' reliability, authenticity, and accuracy. Consensus mechanisms are implementations by which consensus is achieved in blockchain systems. There are many alternative consensus mechanisms in use in different blockchain systems. Examples of consensus mechanisms include Proof-of-Work, Proof-of-Stake, Delegated Proof-of-Stake, Paxos, Practical Byzantine Fault Tolerance, Proof-of-Authority, Proof-of-Burn, Proof-of-Capacity, and Proof-of-Ownership.

For a Public-Permissioned network, the permissioning of nodes allows for the usage of consensus algorithms other than PoW or PoS, taking advantage of the well-known identities of the nodes executing the consensus algorithm.

As mentioned above, the consensus algorithm is a very important component of a blockchain network, affecting many aspects of the system like **scalability**, **sustainability** and even the **governance** of the technical platform:

- The **efficiency** and transaction throughput that can be achieved are much greater than those obtained in public anonymous networks.
- For some consensus algorithms **transaction finality** is deterministic which is a requirement for facilitating many legal transactions in the real-world productive economy.
- The proper governance of the nodes participating in the consensus algorithm can have a critical influence in improving the level of **trust and confidence** in the network.

In the research literature there are several surveys and detailed analyses of the different types of consensus algorithms for blockchain networks and their properties. See for example [CachinVukolic2017], [NguyenKim2018] or even for specialized fields like IoT [MackenzieEtAl2018].

In this document we will focus only on the properties of blockchain consensus algorithms which are most suitable for the type of use cases that will be initially implemented in Public-Permissioned networks.

3.1. Some terminology

In a permissioned transaction ledger, in general only a limited set of nodes participate in the execution of the consensus algorithm. In order to maintain generality and independence from specific blockchain technology, we will use the term *consensus nodes* to refer to the set of nodes that execute the consensus algorithm. It should be noted that in some implementations the consensus nodes are called *validator nodes* and in other environments *ordering nodes*.

With this terminology, we can define two main roles of participants in a permissioned blockchain network:

Consensus nodes

are responsible for the execution of the consensus algorithm

Regular nodes

perform the maintenance of a local copy of the blockchain using the blocks generated from the set of consensus nodes.

In this discussion, in order to determine the position of a transaction within the transaction ledger we use a pair **(h, i)**, where **h** is the height of the block including the transaction, and **i** is the position of the transaction within the block.

3.2. System model

Any discussion about a consensus algorithm assumes some properties of the network and the threat model to be true.

Network model

As is very common in related literature, we assume an eventually asynchronous network.

Failure model

We consider a Byzantine failure mode system, where Byzantine nodes can behave arbitrarily. In contrast, honest nodes never diverge from the protocol definition.

3.3. Consensus properties

The properties required from the consensus algorithm for Public-Permissioned are the following.

3.3.1. Proven in the field, peer-reviewed and its behaviour formally analyzed, especially with respect to resiliency

As Cachin [Cachin2017] states:

> Over the recent years countless proposals for new features in distributed ledger systems and completely new blockchain protocols have appeared. Most of them come without formal expression of their trust assumption and security model.
>
> Instead, broad agreement on trust assumptions, security models, formal reasoning methods, and protocol goals is needed. Developers, investors, and users in the industry should look towards the established scientific methodology in cryptography and security with building trustworthy systems, before they entrust financial value to new protocols.

In other words, a Public-Permissioned network should take a somewhat conservative approach to consensus algorithm selection, and accept only the most robust systems available which are also compatible with the other project objectives. More important than the maturity of the software implementation is that the underlying consensus algorithm has been peer-reviewed by the scientific community and has been formally proven to work according to specifications.

Software bugs can be detected and fixed, but algorithm bugs can be extremely hard and costly to fix, if not impossible to fix if further theoretical analysis proves that they are not correct. Formal proofs of algorithm correctness is a critical requirement of any consensus algorithm that a Public-Permissioned blockchain network uses.

3.3.2. Deterministic (strong transaction finality)

Transaction finality is an indication of whether a transaction is considered final once it has been added to the blockchain. Once confirmed, transaction finality refers to the guarantee that a past transaction can never be changed.

In blockchain systems, all transactions are considered immutable. This being said, most blockchain systems only give *probabilistic transaction finality* which states that transactions are not immediately final, but become final eventually.

For example, in PoW temporary forks and chain reorganizations are allowed during normal operation, because more than one miner can solve the cryptographic puzzle at the same time, and generate different blocks at the same time. In this case, we say that the transaction finality is probabilistic and clients will have to wait until several confirmations are submitted and confirmed before they can consider the probability of transactions being reverted is sufficiently low for the application domain being implemented.

There are other consensus algorithms, that are not probabilistic during normal operation (no Byzantine attacker), but because they are still ***longer chain wins***, an attacker with enough resources can rewrite history. These algorithms are called **Deterministic Longest-Chain Protocols** in [Shi2018]. PoA algorithms like Aura (Parity) and Clique (Ethereum) are two notable examples of these "longer chain wins" algorithms. The transaction finality assurances of these algorithms are higher that in PoW, but still they do not have full transaction finality, what we call **strong transaction finality**.

Finally, there are consensus algorithms which are designed from the ground up for strong transaction finality, like IBFT (Quorum, Besu) or BFT-SMaRt (Corda). Formal proof of strong transaction finality in IBFT can be found later in this document.

A Public-Permissioned network requires a consensus protocol with strong transaction finality.

3.3.3. Resiliency-optimal

The algorithm used should be *resiliency-optimal*, meaning that the resources required by the algorithm to achieve the stated safety guarantees are equal to the theoretical minimum requirement. In the case of an eventually synchronous network with Byzantine actors this means that the relationship of the total number of consensus nodes n with respect to the maximum tolerated byzantine ones f is:

n = 3*f + 1

This lower limit has been proven by the scientific community to be the theoretical minimum number of nodes needed to provide Byzantine resilience in eventually asynchronous networks, like the one we are assuming for a Public-Permissioned network.

3.3.4. Latency-optimal: efficiency of communication among consensus nodes

There are many factors that can affect the efficiency and performance of a consensus algorithm. However, one of the most critical factors for a resilient Byzantine consensus, especially on a wide-area network, is the total number of messages exchanged among the consensus nodes in order to achieve agreement.

It is known that deterministic resiliency-optimal Byzantine consensus protocols cannot use less than three communication steps [DuttaEtAl2005], [MartinAlvisi2006]. This means that latency-optimal protocols for BFT consensus that use 3f + 1 nodes to tolerate f Byzantine faults (e.g., PBFT [CastroLiskov2002]) require at least three communication steps for the consensus (without taking into account the messages required for the client to inject the transaction into the network, the propagation of the transaction across the network and the reception of the reply).

3.3.5. Energy efficient, computationally-optional and sustainable

Taking advantage of the permissioning of nodes and avoiding the excessive computation required for algorithms like PoW. The energy consumption required should be reasonable for the transaction workload that the network supports.

3.3.6. Transparent execution (efficient monitoring by all users)

There should be enough information about real-time consensus algorithm execution which is visible to all participantes in the network. In many implementations, if a consensus node behaves in a Byzantine way, the consensus algorithm continues working keeping the network safe, that is, the consensus algorithm is "masking" the bad behaviour. In EBSI, however, it is not enough to mask the error (or byzantine behaviour) but instead it should be reported in a way that all network participants can be aware of that specific node bad behaviour.

3.3.7. Enable accountability (responsibility) of consensus nodes

Requires unforgeable digital signatures of messages and seals during the process of consensus. Any action performed by the consensus nodes can not be denied if an ex-post analysis is performed. Auditability of the consensus algorithm execution is a required property for proper accountability of execution by each consensus node. In other to maximise throughput, many proposed BFT consensus implementations eliminate digital signing of messages exchanged across consensus nodes. Even though many still use different techniques for message authentication, they may prove extremely difficult to be audited ex-post. We believe that digital signatures where the public keys of each consensus node is well-known by the participants is the most simple and robust method to provide audibility and accountability for the proper behaviour of consensus nodes.

3.3.8. Fair

Each member of the network should be able to run its own node or set of nodes. It is also understood that the members can participate if they wish in the execution of the consensus algorithm by operating a consensus node. With PoW, the probability of a node to have the opportunity to create a block (and so decide the transactions included in it) depends on the total computational power that a given entity has (the so-called hashing power). In addition, PoW opens the possibility of several nodes collaborating and joining forces to achieve enough computational power to have a higher probability of creating blocks (mining pools). This is clearly not "fair" because the mechanism favours those with more power. As far as possible, the consensus algorithm should allow that anybody can participate in consensus, subject to some fair and transparent rules, and not only the most powerful.

4. The Consensus algorithm and On-chain Governance

The consensus algorithm is one of the most important components of a blockchain network, affecting many aspects of the system like scalability, sustainability and even to the governance of the technical platform. In addition, governance of the consensus set is critical to the level of trust that this network has for the rest of the participants (the so-called ``regular nodes'').

The current Alastria network, Red T, uses *Istanbul Byzantine Fault Tolerance* (IBFT) as consensus algorithm, which belongs to a family of PBFT consensus algorithms sharing many of the properties heavily discussed and formally proven during the last decades [CastroLiskov1999], and its properties are very well known.

When applied to the blockchain, the PBFT variants comply with the **Robustness property** [Saltini2019] when the maximum number of Byzantine validator nodes **t** follows the well-known relationship with **n**, the total number of validator nodes:

n = **3t** + **1**.

However, standard implementations of PBFT and in particular IBFT, tend to focus on masking failures. That is, they make failures transparent to the users, but they do not manage those failures in a way that proactive or reactive measures can be taken to ensure the long-term health of the network.

This is the reason why Alastria is implementing a set of tools surrounding the base IBFT consensus algorithm, which together with complementary off-chain governance processes allow the realization of the the principles of the governance of the blockchain as a Common-Pool Resource.

4.1. An example of on-chain governance of the Consensus set

The subject is too complex to be thoroughly treated in a reduced space, but the following figure describes a summary of an example of on-chain governance of the blockchain network.

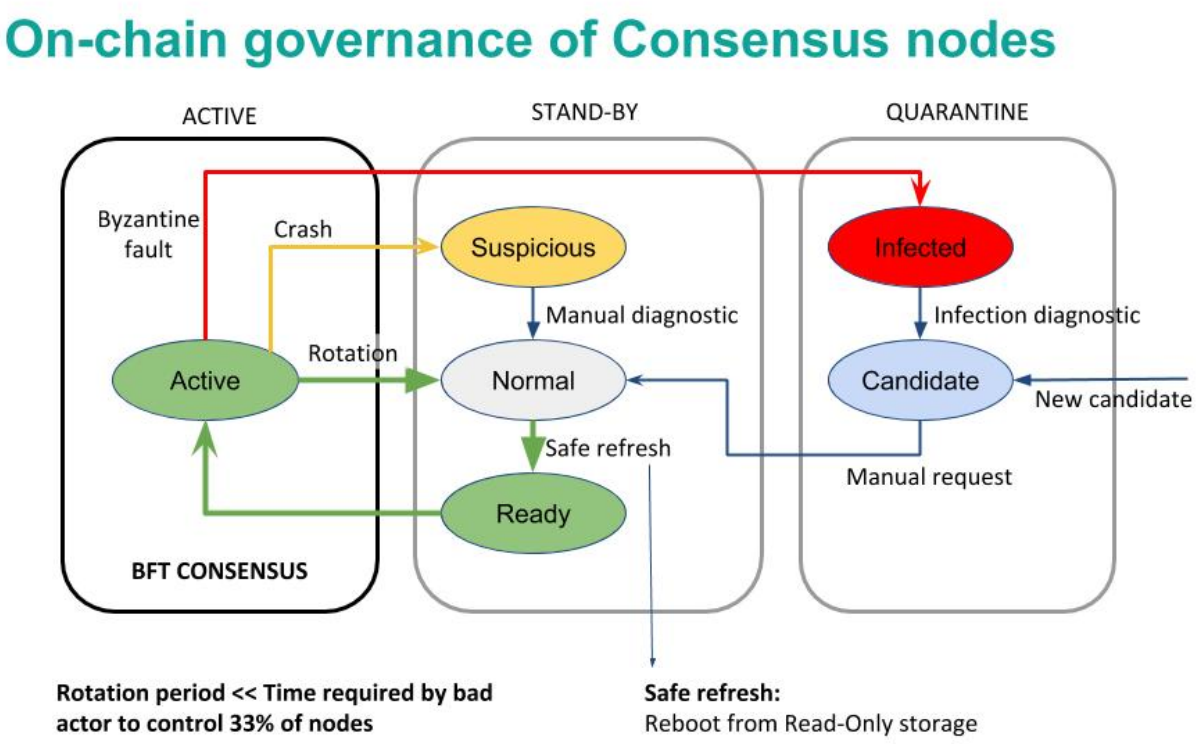

Figure 6. On-chain governance of Consensus set

The following aspects can be observed:

- This could be considered a generalization of the proactive recovery mechanism described in [CastroLiskov2002]
- The consensus nodes in the Active state (that is, executing the base IBFT algorithm) are being monitored, as per principle 4.b of the CPR governance principles
- The events signaling different types of faults are used for the reactive governance of the nodes. Even though it is not shown in the figure, in addition to the automated reaction, the events are reported in a way that any participant in the blockchain network (not just the consensus nodes). This is required to implement the high levels of transparency and collaborative monitoring that are required for the effective management of CPR resources.
- Depending on the severity of the fault detected (crash or byzantine), the system reacts automatically applying a graduated set of sanctions, as per the principle 5 of CPR governance.

For example, when the fault is byzantine, the consensus node affected is put in quarantine, effectively stopping the node from participating in the consensus execution. If the owner is willing to continue participating, a manual process (off-chain governance) is required, with sufficient explanation and justification to the other members in order to be accepted again.

References

- [Navarro2018] L. Navarro. *Network infrastructures: The commons model for local participation, governance and sustainability*. 2018
- [Ostrom1990] E. Ostrom. *Governing the Commons: The Evolution of Institutions for Collective*

Action. 1990

- [Buterin2014] V. Buterin. *On sharding blockchains*. 2014
- [Hardin1968] G. Hardin. *The Tragedy of the Commons*. 1968
- [DavidsonEtAl2016] S. Davidson, P. De Filippi etAl. *Disrupting Governance: The New Institutional Economics of Distributed Ledger Technology*. 2016
- [RozasEtAl2018] D. Rozas, A. Tenorio-Fornés etAl. *When Ostrom Meets Blockchain: Exploring the Potentials of Blockchain for Commons Governance*. 2018
- [CachinVukolic2017] C. Cachin and M. Vukolić. *Blockchain Consensus Protocols in the Wild*. 2017
- [NguyenKim2018] G. Nguyen and K. Kim. *A Survey about Consensus Algorithms Used in Blockchain*. 2018
- [MackenzieEtAl2018] B. Mackenzie, X. Bellekens etAl. *An Assessment of Blockchain Consensus Protocols for the Internet of Things*. 2018
- [Cachin2017] C. Cachin. *Blockchains and Consensus Protocols: Snake Oil Warning*. 2017
- [Shi2018] E. Shi. *Analysis of Deterministic Longest-Chain Protocols*. 2018
- [DuttaEtAl2005] P. Dutta, R. Guerraoui etAl. *Best-case complexity of asynchronous Byzantine consensus*. 2005
- [MartinAlvisi2006] J. Martin and L. Alvisi. *Fast byzantine consensus*. 2006
- [CastroLiskov2002] M. Castro and B. Liskov. *Practical Byzantine Fault Tolerance and Proactive Recovery*. 2002
- [CastroLiskov1999] M. Castro and B. Liskov. *Practical Byzantine Fault tolerance*. 1999
- [Saltini2019] R. Saltini. *Correctness Analysis of IBFT*. 2019