OT DEVELOPMENT TEAM / ROMANIA

TECHNICAL WHITEPAPER • FEBRUARY 2020



SHB

ShibaToken

A Blockchain for Collective Decision-Making

The Architecture of Decentralized Choice

Version 1.5 | Technical Whitepaper

DOCUMENT NOTES

Scope and Status

We describe the intended design of the SHB protocol as of February 2020. It distinguishes proposed functions from demonstrated implementations. Network parameters, security guarantees and deployment milestones remain subject to development and testing.

The SHB token is conceived as a governance instrument. It is not proposed as a general payment system, electronic cash substitute, or guarantee of financial return. The protocol specifies a fixed supply ceiling of 100,000,000 SHB and a nominal one-minute block interval; these are design parameters, not claims of a running mainnet.

Authorship

OT Development Team, Romania. The identifier “OT” is used as a collective development pseudonym.

Document Conventions

Terms such as “proposes”, “intends”, “may” and “could” denote design intentions or possibilities rather than completed functionality. The token-weighted baseline is defined as one SHB token corresponding to one unit of voting influence, subject to the rules and eligibility criteria of the relevant community.

Not another currency. A new language for collective decisions.

FRONT MATTER

Abstract

Online communities have no shortage of discussion tools. What they frequently lack is a record that lets members check who had authority, what question was put to a vote, how the outcome was calculated and whether anyone followed through. That gap becomes more serious as an organization grows or its members stop knowing one another personally.

SHB is a proposal for such a record. The native token represents governance influence rather than payment value: one eligible SHB provides one unit of voting weight under the rules of the relevant community. The ledger records proposals, ballots, delegation and the changes those decisions authorize. Different communities may use the same network while keeping their own constitutions.

The work does not end when the votes have been counted. Members may want to trace the argument behind a decision, see whether a representative acted within a mandate or revisit a rule without rewriting its history. SHB is designed to connect those stages. Some actions can be verified by software; others will still depend on people, documents and institutions outside the network.

The following chapters explain the proposed architecture, its assumptions and the compromises that come with token-weighted governance. We describe the design as it stands in February 2020. In particular, cryptographic verification should not be confused with fairness, anonymity or evidence that a production system has already passed security testing.

Keywords

Blockchain · Decentralized Governance · Collective Decision-Making · Token-Weighted Voting · Delegation · Distributed Consensus · Community Rules

MANUSCRIPT MAP

Table of Contents

The appendices follow Chapter 20. All entries include their final starting page in this edition.

PART I — THE IDEA OF COLLECTIVE SOVEREIGNTY

01 The Unfinished Internet
02 **Beyond the Transfer of Value**
03 **The SHB Proposition**

PART II — THE NETWORK BENEATH THE VOTE

04 **One Ledger, Many Communities**
05 **Agreement Without a Central Referee**
06 **The Right to Participate**
07 **SHB: Units of Influence**

PART III — THE ART OF DECIDING

08 **The Birth of a Proposal**
09 **Counting What Matters**
10 **Delegation Without Abdication**
11 Private Choice, Verifiable Result
12 Rules That Govern the Rules

PART IV — BEYOND THE BALLOT

13 Reputation Beyond Ownership
14 Communities Within Communities
15 The Archive of Intentions
16 From Decisions to Actions
17 A Marketplace of Ideas, Not Money

PART V — BUILDING SHB

18 Making the Network Useful
19 Threats, Limits and Responsibilities
20 The Road Ahead

REFERENCE MATERIAL / APPENDICES & BIBLIOGRAPHY

A Glossary of SHB Terms
B Protocol Mathematics and Cryptographic Notation
C Bibliography and Source Concordance

PART I / THE IDEA OF COLLECTIVE SOVEREIGNTY

Chapter 1

The Unfinished Internet

Why communication alone is not enough to govern a community.

The network gave us ways to gather. It did not give us a common way to decide.

FIGURE 1.1 FROM DISCUSSION TO ACCOUNTABLE ACTION

COMMUNICATE	DELIBERATE	DECIDE	EXECUTE	REMEMBER
Exchange ideas	Compare options	Verify outcomes	Assign actions	Audit history

Figure 1.1 — The SHB proposal connects processes usually maintained by separate institutions or services.

1.1 The Internet as a Network of Communication

The internet has made distance less important to collaboration. A developer in Bucharest can review code from Helsinki, volunteers can keep a common knowledge base without meeting, and a small organization can reach thousands of readers through a browser. Yet one problem remains stubbornly familiar. Being able to talk to everyone in a group is not the same as being able to reach a decision on the group's behalf, let alone prove afterward how that decision was taken.

Communication and coordination are related, but they are not the same thing. A message may reach every member of a group while leaving unanswered the question of who has the authority to decide. A discussion may produce a clear preference without establishing whether it is representative. A poll may report an overwhelming majority without proving who voted, whether the same participant voted more than once, or which rules governed the count. Even when the result is accepted, somebody must still implement it.

Most online communities were built around a familiar sequence: someone creates a platform, others join it, and the platform's operators retain final control over permissions and records. This is often sensible. Administrators must remove spam, fix errors and maintain services. Trouble begins when a community grows beyond the scale at which informal trust can reliably answer every dispute. The participants may collectively produce the value of an organization while having little independent means to verify its decisions.

SHB begins with a simple observation: a distributed community needs more than a place to speak. It needs a dependable method of determining what a decision is, how it was reached, and what authority it carries. The proposed blockchain is intended to supply a common record for those questions without claiming that software can resolve every disagreement among people.

1.2 The Missing Infrastructure for Collective Decisions

A conventional website records votes in a database operated by its owner. Users may see a percentage on the screen, but they usually cannot inspect the complete history of the ballot or independently reproduce the calculation. The count may be entirely correct; the difficulty is that its correctness depends on confidence in a particular operator. For decisions of minor importance this may be acceptable. For a community deciding its rules, elected representatives or long-term direction, it becomes a structural weakness.

A credible decision system must answer several questions before the first ballot is cast. Who may introduce a proposal? Who is eligible to participate? At which moment is eligibility measured? How is each participant's weight calculated? Can an instruction be delegated, changed or withdrawn? What happens when a decision fails to reach quorum? When is the outcome final, and who has permission to

carry it out? These questions are constitutional rather than merely technical, yet software must express them precisely if the results are to be consistent.

Some existing tools address parts of the problem. Forums are useful for deliberation; surveys are useful for measuring sentiment; membership registers can restrict participation. Systems inspired by liquid democracy, including LiquidFeedback, have explored structured proposals and delegation. The relevant lesson is not that every community requires a blockchain. It is that collective decision-making benefits when the rules are explicit and the sequence of events is independently inspectable.

We propose to combine these functions around a shared, tamper-evident ledger. A proposal could be assigned an identifier, a defined voting period and a set of eligibility rules. A ballot could be authenticated by a digital signature and counted against an agreed snapshot of token balances. The network could retain the resulting state in a way that participating nodes can check without asking a central administrator for the official answer. Such a system would not make a poor decision good; it could make the process that produced the decision substantially easier to audit.

1.3 Why Coordination Still Depends on Trusted Intermediaries

In small groups, trust is personal. Members know who is responsible for the accounts, who chairs a meeting and who records the minutes. In larger or more dispersed groups, trust becomes institutional. The secretary maintains the register, the chairman accepts motions, the platform operator controls access, and an administrator publishes results. These offices perform useful work, but they also create points at which an error, conflict of interest or unilateral change can affect everybody else.

One might be tempted to describe decentralization as the elimination of trust. That description is incomplete. Software does not remove trust; it changes where trust is placed and what can be verified. A cryptographic signature can demonstrate that a given key authorized a ballot. It cannot, by itself, establish whether the human using that key is acting freely. A blockchain can preserve the text of a resolution. It cannot prove that the resolution is wise, lawful or socially legitimate. If a vote authorizes the repair of a school, the ledger cannot inspect the physical building and certify that the work was completed without additional evidence and responsible parties.

The appropriate objective, therefore, is narrower and more useful: reduce avoidable dependence on any one party for those operations that can be made reproducible. A participant should be able to verify the rules applied to a proposal, the state used to calculate voting power, and the arithmetic that produced a final result. A community should be able to distinguish a decision recorded by the network from a statement made on behalf of the network.

This distinction also suggests the proper role of human institutions within SHB. Moderators may still organize discussions. Elected officers may still negotiate and act. Independent experts may still provide evidence. What changes is that their authority can be defined by public rules, bounded in duration and linked to a record that other participants can examine. Decentralization, in this sense, is not the absence of responsibility. It is the distribution and verification of responsibility.

1.4 Digital Communities and the Problem of Representation

Representation has always involved a compromise between participation and practicality. Direct votes give individuals immediate control over their choices, but not everyone has the time or information required to study every question. Delegation allows a group to benefit from expertise, yet it introduces the risk that delegates become detached from those they represent. Online communities experience the same tension as traditional organizations, often at greater speed and on an international scale.

A further complication arises when influence is measured by tokens. Under the basic SHB model, an address holding one hundred tokens possesses one hundred units of voting weight, subject to the relevant community rules. This is not equivalent to one person having one vote. It is a deliberate model of stake-based influence, not an automatic guarantee of political equality. A person may control several addresses, and several people may act through one custodian. The protocol must not confuse the number of addresses with the number of human beings.

There are circumstances in which proportional influence may be appropriate. A voluntary association could use tokens to represent agreed membership shares or participation rights. A contributor network might allocate tokens according to a publicly defined distribution policy. Other communities will prefer equal voting, representative voting, or an arrangement in which certain decisions require a separate class of approval. The common ledger should not force these distinct arrangements to pretend they are identical.

SHB therefore treats one-token-one-unit as its fundamental counting mechanism, while allowing governance environments to define eligibility, thresholds and forms of delegation around it. The design must also expose uncomfortable questions. Can large holders dominate a small community? Can an attacker borrow or acquire tokens briefly to alter a vote? Can representatives accumulate too many delegated mandates? Balance snapshots, time delays, quorum rules and transparent delegation can reduce particular risks, but none can make concentrated power disappear. An honest governance protocol must make its assumptions visible.

1.5 From Participation to Enforceable Collective Rules

Most digital voting systems end with a result. A community's actual work begins there. A decision to change a rule must be translated into a new rule. A decision to appoint a representative must specify the powers attached to the appointment. A decision to approve a shared project should explain which steps follow, who is responsible and how the community will know whether its instruction has been fulfilled.

Consider an open-source software community deciding whether to change the release procedure for its project. Discussion takes place in public. Two competing proposals are submitted. Eligible members vote according to a defined method and deadline. The winning proposal establishes a review period, after which authorized maintainers may update the release policy. An ordinary online poll can report the preferred option. A governance ledger could additionally preserve the exact versions of both proposals, the eligibility snapshot, the counting rule, the final decision and the mandate granted to the maintainers.

The ability to record a mandate is not the same as the ability to execute it automatically. Some actions can be represented as deterministic state changes within software. Others require signatures, contracts, physical work or cooperation from people outside the network. SHB aims to make that boundary explicit. Where an action can be safely automated, the protocol may provide controlled execution. Where it cannot, the network may record commitments, deadlines and verifiable evidence supplied by responsible participants.

This approach also requires a distinction between immutability and inflexibility. A community should not be able to rewrite the historical outcome of a ballot. It may nevertheless need to reverse a policy, revoke an appointment or correct a mistaken instruction through a new, authorized decision. A useful governance record preserves what happened without declaring that what happened must remain correct forever. Rules ought to be amendable; their amendments ought to leave a history.

1.6 The Case for a Governance-Oriented Blockchain

The emergence of public blockchains has demonstrated that a network of independent computers can maintain a common history under explicitly defined agreement rules. Bitcoin applied this approach to electronic cash. Programmable blockchain systems have since made it possible to express broader arrangements as software. These developments do not establish that every coordination problem belongs on a chain; they establish that independently verifiable state is a practical design option where participants cannot rely on a single record keeper.

Our aim with SHB is to consider what a blockchain would look like if governance were its principal task rather than a secondary feature. Its native token would be defined by the authority it carries in collective decisions. The fundamental objects of the system would include communities, proposals, ballots, mandates and rules, alongside accounts and transactions. Success would not be measured chiefly by the volume of transfers. More relevant measures would include the integrity of vote counting, the continuity of a community's records, the cost of participation and the ability of members to verify a decision independently.

This direction creates difficult engineering requirements. The network must prevent duplicate or invalid ballots without making ordinary voting prohibitively expensive. It must distinguish the consensus that establishes blockchain state from the political process through which communities choose their policies. It must provide a predictable treatment of delegation and balance changes while a vote is open. It must acknowledge that a permanent public record can be harmful when it exposes private choices or personal information. And it must remain usable by people who have no interest in operating a node or understanding cryptographic primitives.

A specialized chain is one possible architectural response, but its justification must be earned. An independent network increases the burden of security, maintenance, validator coordination and public adoption. If existing systems can provide a requirement more safely, SHB should learn from them rather than pursue novelty for its own sake. The next chapters outline the principles against which the proposed architecture will be tested: predictable governance state, verifiable participation, configurable rules and a record durable enough to outlive the individuals who first created it.

The unfinished task of the internet is not to give every opinion equal force or to turn disagreement into unanimity. It is to give people who choose to organize together a reliable way to say what they have decided, show how they decided it and know which obligations follow. SHB is a proposal to build part of that missing infrastructure.

What the Network Must Settle

The following chapter examines the token itself: not primarily as an object to be spent, but as a measurable grant of influence within a chosen set of rules. That shift in purpose defines the economic and organizational questions the SHB architecture must answer.

References

[1] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008. https://bitcoin.org/bitcoin.pdf

[2] V. Buterin, Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform, 2014, original whitepaper.

[3] J. Behrens et al., The Principles of LiquidFeedback, and the LiquidFeedback project (initial software released 2009).

[4] L. Lamport, R. Shostak and M. Pease, The Byzantine Generals Problem, ACM Transactions on Programming Languages and Systems, 1982.

PART I / THE IDEA OF COLLECTIVE SOVEREIGNTY

Chapter 2

Beyond the Transfer of Value

What a governance token is for—and what it must never pretend to be.

The significance of a token is not determined by what it can buy. It is determined by the rules that give it meaning.

2.1 Blockchain Technology Beyond Financial Transactions

Most accounts of digital tokens begin with a transfer: one address sends a quantity, the network checks the instruction, and the ledger updates two balances. The mechanism is useful, but it does not tell us what the quantity means. It might represent money, admission to an event, membership in an association or a share of decision-making authority. The answer is found in the rules around the token, not in the arithmetic of transferring it.

The distinction matters because the language of digital currencies can obscure the more general properties that make blockchains useful. Independent participants can maintain a common state. Rules can govern which changes to that state are accepted. Digital signatures can establish that a particular key authorized an instruction. Once a result has been committed to a sufficiently secure ledger, attempts to rewrite its history become detectable. These capabilities can support financial transfers, but they can also support a procedure for making decisions.

The 2008 Bitcoin paper demonstrated one arrangement for maintaining a shared transaction history without a central bookkeeper. Subsequent work, notably the Ethereum proposal published in 2014, explored more general programmable state transitions. By February 2020, it is no longer necessary to imagine the blockchain solely as a book of balances. It is also possible to treat it as a machine for applying mutually agreed rules to a sequence of signed instructions.

There is a parallel line of development outside blockchains. Participatory software and experiments in liquid democracy have shown that proposing, amending and delegating decisions can be represented in software. These systems do not need a blockchain to function. Their relevance to SHB is different: they demonstrate that voting is a process with a structure, not simply a button marked YES or NO. A distributed ledger becomes interesting when the community wants the rules and the resulting record to be independently verifiable, including by parties who do not control the application presenting the ballot.

SHB is proposed at the intersection of these two ideas. It applies a shared-state protocol to the lifecycle of collective decisions. The token is the basic accounting unit of influence; the wider network is responsible for verifying proposals, voting rights and procedural outcomes. This is a proposed architecture, not a claim that distributing an ordinary polling application automatically makes it fairer or more secure.

2.2 Ownership, Influence and Collective Authority

Ownership is a fact about control over an asset. Influence is a fact about the weight assigned to a participant's preference. Authority is the recognized ability to make or carry out a decision on behalf of a group. These concepts frequently overlap, but they should not be confused. A person may own a token without having the authority to administer a community. A participant may be trusted to carry out a mandate without possessing many tokens. A large token holder may be influential without being representative of the people affected by a proposal.

In the SHB model, eligible token holdings establish voting weight under a stated set of rules. The weight is mechanical: a quantity can be measured and counted consistently by independent nodes. Whether that weighting is legitimate for a particular community is a separate question. A cooperative formed around common ownership may regard proportional voting as suitable for some matters. A

neighborhood association, by contrast, may require one-person-one-vote protections. SHB cannot honestly claim that a transferable token, by itself, establishes equality among human participants.

The SHB protocol design therefore distinguishes the *network rule* from the *community rule*. The network rule specifies how an authorized voting instruction is accepted and recorded. The community rule identifies the decisions to which it applies, which balances are eligible, what threshold is required and whether additional membership qualifications exist. Each vote should carry a reference to the rule set governing it, rather than relying on an administrator to explain the rules after the result is known.

This distinction also separates political authority from software execution. A chain may confirm that a proposal received enough SHB-weighted support to pass. It cannot demonstrate that its subject is ethically acceptable, that every affected person consented, or that a court will recognize its effect. Cryptography can authenticate a procedure; it cannot manufacture legitimacy. SHB must preserve room for appeal, human oversight and explicit limits on what a recorded majority may authorize.

We do not regard this limitation as a defect to be concealed. It is a design boundary. By keeping ownership, voting influence and the right to execute a decision conceptually distinct, the system can support different types of collective organization without pretending that they all share the same constitution.

2.3 A Governance Token Is Not a Payment Instrument

The SHB token is intended to answer a question of participation: *how much voting weight may this address exercise under the applicable rules?* It is not introduced as a claim on goods, a promise of redemption, a unit pegged to a national currency, or a replacement for everyday payments. In the baseline proposal, holding one eligible SHB grants one unit of weight in a token-weighted ballot. The useful act is the expression of a preference that the network can count and verify.

This does not mean the token is incapable of being transferred. Fungible digital units may move between addresses, and external parties may assign them a market price. Those possibilities do not define the protocol's stated purpose. The value of a share in a decision process is not necessarily the same as the price someone is willing to pay for the object representing that share. Indeed, trading can create serious governance problems if purchasing tokens becomes a way to purchase control over a community.

For this reason SHB should not be described as financially neutral simply because it is not designed for commerce. Distribution rules, transferable balances, validator operating costs and possible fees all have economic consequences. A public network must also discourage spam and compensate or otherwise sustain the computers that maintain it. The engineering of those resource costs belongs to the protocol design; it should not quietly convert the governance token into a payment system in everything but name.

A useful test of the concept is to imagine that no shop, exchange or merchant ever accepts SHB as payment. Would the network still have a reason to exist? If communities can use it to make decisions, allocate formal responsibilities and preserve accountable records, the answer is yes. Its central function would remain intact. This is the standard against which the project should be evaluated.

2.4 The Token as a Unit of Voting Weight

TABLE 2.1 TWO DIFFERENT PURPOSES FOR A DIGITAL TOKEN

DESIGN QUESTION	TRANSFER-ORIENTED TOKEN	SHB GOVERNANCE TOKEN
Primary purpose	Transfer or settlement of value	Measurement of voting influence
Essential question	Who controls the balance?	Who may decide, and with what weight?
Meaningful outcome	An accepted transfer	A verified procedural decision
Measure of success	Payment usefulness and reliability	Legible rules, accountable decisions

DESIGN QUESTION	TRANSFER-ORIENTED TOKEN	SHB GOVERNANCE TOKEN
Central risk	Loss, fraud, transaction failure	Concentrated influence, captured governance
Unresolved needs	Security, operating costs, availability	Security, operating costs, availability

The comparison concerns primary design objectives, not an assertion that either system can avoid security or economic trade-offs.

The baseline SHB voting model is deliberately easy to explain. Subject to eligibility rules, one SHB contributes one unit of voting weight. An address holding 100 eligible SHB may cast 100 units of influence in a particular ballot. An address holding 250 eligible SHB may cast 250. The network must count units, not impressions, identities inferred from usernames, or claims reported by a website operator.

A ballot needs a precise reference point because balances can change. We propose to determine eligibility against an agreed snapshot of the finalized ledger state, identified by a block height or another unambiguous state reference. For a proposal p and an address a, the simple form of the rule is:

$$W(a, p) = B_{\text{eligible}}(a, h_p)$$

Here W is the address's voting weight for the proposal; B_eligible is its eligible SHB balance; and h_p identifies the snapshot used by that proposal. Additional rules may exclude locked balances, impose minimum holding periods or account for delegated authority. Those are extensions of a defined rule, not adjustments to be made after the vote.

Consider a small community in which three participants have voting balances of 100, 250 and 50 SHB. The first and third vote in favor of a motion; the second votes against it. The result is 150 units in favor and 250 against. Although two of the three participants support the motion, it fails under a simple majority of token weight. The example is intentionally uncomfortable. It shows that token-weighted governance measures *assigned influence*, not the number of distinct human beings who agree.

A further example illustrates quorum. Suppose a proposal has 10,000 eligible SHB units at its snapshot, and the community requires participation of at least 20 percent. Votes totaling 2,400 units satisfy that requirement. If 1,400 units vote yes, 900 vote no and 100 abstain, the approval percentage depends on the written definition: excluding abstentions, yes represents approximately 60.87 percent of yes-plus-no weight; including abstentions, it is approximately 58.33 percent of all participating weight. The protocol must specify which convention applies. It must also specify how a tie, a missing quorum, an invalid ballot or a late instruction is treated.

Snapshotting introduces questions of its own. If tokens are moved after the snapshot, the original holder may remain eligible for that vote while the recipient acquires the current balance for later purposes. The protocol must prevent the same voting units being counted twice within a ballot. Clear rules for delegated balances and simultaneous proposals are needed before the system can safely be opened to public use.

The virtue of this model is not that it resolves every fairness dispute. It is that, once the rules have been published, independent participants can reproduce the arithmetic and identify exactly where a disagreement lies.

FIGURE 2.1 MEASURABLE INFLUENCE, VERIFIED PROCEDURE



The final tally can be reproduced from a specified state and a specified rule set. The social legitimacy of the decision remains a separate question.

2.5 Influence Without a Monetary Objective

If a token is primarily an instrument of governance, the network's success should not be measured only by transaction volume or the market quotation of its units. More relevant questions concern the decisions it enables. How many communities can operate without surrendering control of their voting records? Can a member check the outcome of a proposal without trusting its author? Is delegation easy to withdraw? Are amendments recorded accurately? Can a community establish that an authorized decision was actually followed by the promised work?

These are questions about institutional performance. Their answers are likely to depend on interface design and public confidence as much as cryptography. A perfectly verifiable ballot that ordinary participants cannot understand will produce little practical benefit. Conversely, a pleasant interface cannot compensate for an unverifiable tally or rules that an operator may change during voting. SHB will require both readable procedures and precise machine validation.

The token's distribution deserves particular care. If a small group holds nearly all eligible SHB, a formally decentralized network could still be politically concentrated. Delegation may compound that concentration. Voting snapshots can protect the integrity of a particular tally, but they cannot prevent a determined actor from accumulating influence before the snapshot. Possible responses include transparent distribution, caps for defined classes of proposals, higher approval thresholds for constitutional changes and separate membership qualifications. Each response introduces trade-offs and must be justified rather than applied as a decorative promise of fairness.

Influence should also be kept distinct from reputation. A person may write useful proposals, verify documents, maintain software or mediate disputes without receiving additional transferable voting power. Communities may want to record such contributions or award nonfinancial recognition. They may also wish to consult recognized experts while reserving formal decisions to their token holders. The architecture should allow these functions to coexist without requiring every form of trust to be purchased or converted into SHB.

SHB therefore treats governance as a utility in its own right. The proposed token makes participation measurable; the surrounding protocol makes it verifiable. Whether that measurement produces wise collective outcomes depends on the rules chosen by each community and on the continued willingness of its members to take responsibility for them.

2.6 Governance as an Independent Blockchain Utility

A governance-oriented ledger requires a vocabulary different from a conventional transfer ledger. It must understand proposals, eligibility records, ballot windows, delegated authority, quorum calculations, approvals, rejections and the status of mandates. Each event must have a defined relationship to the current network state. A node should be able to process the same ordered instructions as its peers and arrive at the same accepted result.

Take an ordinary proposal. One participant submits a signed request containing its subject, permitted voting options, the applicable community constitution and the interval during which ballots may be accepted. The protocol determines whether the proposer is authorized. When the ballot opens, the reference state for voting weight is fixed. Participants submit signed instructions, or authorized representatives act under their delegations. Once the voting period closes, the network checks the result against the quorum and approval requirements. It then records a status that can be reproduced by another participant following the same rules.

This description may sound administrative, but administration is the point. A substantial part of governance consists of agreeing in advance how exceptions will be handled and then applying those rules consistently when an exception occurs. The difficult cases are often not overwhelming majorities. They are disputed amendments, revoked delegates, late ballots, overlapping mandates, invalid signatures, network interruptions and proposals that pass by a narrow margin.

The ledger should not require every discussion message or supporting document to be stored permanently inside a block. Large files can reside in external or distributed storage while their

cryptographic digests, identifiers and relevant metadata are anchored to the chain. IPFS and ordinary content-addressed approaches provide useful components for such a design. An anchored digest can later show whether a particular document matches the version referenced in a decision; it cannot prove that the document's contents are true.

The validation network must also remain separate from the community's political judgments. Validators should enforce valid state transitions, not replace a community's vote with their own preference. Still, operational questions remain: who can become a validator, how disagreements between nodes are resolved, how availability is maintained and how upgrades are accepted. These are fundamental design decisions for a dedicated chain and are examined in later chapters. A governance-first purpose does not exempt SHB from the security requirements of a public distributed system.

2.7 Why Voting Alone Is Insufficient

Imagine that an open-source community votes to prioritize an accessibility improvement over a new visual feature. The final tally may be correct, yet three questions remain. Who is authorized to coordinate the work? How will the community know that the promised improvement has been delivered? What happens when circumstances change and the decision is no longer practical? A ballot closes one question while opening several others.

For SHB, a meaningful governance cycle begins before voting and continues afterward. Members need a place to introduce alternatives and examine evidence. They need rules for deciding who can vote and under what conditions. They may wish to delegate their influence to someone with more time or expertise, while retaining the ability to revoke that authority. Following a decision, they need a record of its terms, a named mandate where appropriate, and a means of reporting implementation. Where action occurs outside the blockchain, the record should distinguish a verified on-chain authorization from an externally supplied claim of completion.

There is also value in remembering decisions that did not pass. A rejected proposal may explain why a community adopted a different course. A close ballot may expose unresolved concerns. A minority opinion may later prove important. If the only surviving artifact is a transaction indicating success or failure, much of the community's institutional knowledge is lost. The intended SHB architecture therefore treats the history of argument, the formal result and the subsequent action as related but different records.

This wider scope suggests a practical principle: a governance protocol should record not only *what was decided*, but *under which rules*, *by which voting weight*, *for what purpose* and *with what subsequent responsibility*. Not every detail belongs on-chain, and not every report can be automatically verified. The chain should preserve the parts that must be consistent and auditable while leaving room for human deliberation, private information and decisions that require judgment.

The chapters that follow describe how SHB might supply this structure. The next chapter establishes the project's own operating principles and design boundaries before turning to nodes, consensus and the technical life of a ballot. Our ambition is not to place every aspect of community life inside software. It is to ensure that when people choose to govern something together, they can understand the rule they are using and verify the decision it produces.

A Different Accounting Unit

Our aim with SHB is simple to state and demanding to implement: give a digital community a measurable way to express collective intention without mistaking that measurement for absolute authority. A token can count influence. A protocol can make the count verifiable. The community must still decide what its influence is for.

References and Foundations

[5] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System (2008). https://bitcoin.org/bitcoin.pdf

[6] V. Buterin, Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform (2014). https://ethereum.org/whitepaper/

[7] F. Vogelsteller and V. Buterin, ERC-20: Token Standard, EIP-20 (proposed 2015).
https://eips.ethereum.org/EIPS/eip-20

[8] J. Behrens et al., LiquidFeedback project and its principles of delegable decision-making (since 2009).
https://liquidfeedback.com/en/history.html

[9] L. Goodman, Tezos: A Self-Amending Crypto-Ledger, position paper (2014).

PART I / THE IDEA OF COLLECTIVE SOVEREIGNTY

Chapter 3

The SHB Proposition

An independent protocol for decisions that belong to the people who make them.



There are already ways to issue tokens, operate smart contracts and maintain append-only records. Adding another blockchain to that list would accomplish little on its own. The case for SHB depends on a narrower proposition: collective decisions should be first-class records, with clear eligibility rules, a reproducible count and an account of what happened after the ballot. That purpose affects almost every design choice, from transaction types to the handling of disputed authority.

SHB is proposed on those terms. Its central object is not the payment but the collective decision. Its principal resource is a measurable unit of influence. Its desired output is not a transfer confirmation but a decision whose authority, counting method and subsequent status can be examined by anyone permitted to verify the relevant records. This chapter establishes that proposition before the paper turns to the technical machinery required to support it.

3.1 Introducing ShibaToken

ShibaToken, abbreviated SHB, is the proposed native token of an independent blockchain dedicated primarily to decentralized governance. The token gives a participant a measurable capacity to influence decisions taken within governance environments that recognize it. The blockchain supplies the common record and the rules through which that influence can be exercised, counted and challenged by reference to evidence rather than to the word of a central operator.

The distinction between the token and the network matters. SHB tokens are units recorded in the ledger. The SHB protocol is the system that interprets those units within proposals, ballots, delegations, community rules and the resulting states. Owning a unit and operating the infrastructure that validates the unit are separate activities. Neither possession of tokens nor operation of a validator should, on its own, permit a participant to alter the history of an accepted vote.

At its most elementary, the protocol should allow a community to publish a question, identify the rules under which the question is to be decided, establish the eligible voting weight, accept authenticated votes and publish a result that can be independently recomputed. A more developed implementation would connect that result to a mandate, a review period or a change in the community's operating rules. These features are proposed capabilities, not assertions that a functioning SHB network has already demonstrated them.

The name ShibaToken identifies the project; it does not define the project's function. The function must be described in terms of actions. An SHB holder may introduce a proposal when the relevant community permits it, cast a vote, delegate voting authority under defined conditions, support an amendment, or examine the permanent record of an outcome. The same token can be used across several independent governance spaces, but those spaces need not share an identical constitution or an identical definition of membership.

This approach does not require every organization to become a blockchain organization. A volunteer group may need only an auditable vote and a public archive. An open-source project may wish to appoint maintainers and approve technical changes. A cooperative may need more formal review and authorization steps. SHB should be broad enough to support these differences without forcing them into one administrative model.

3.2 The SHB Philosophy

Governance is often described as the power to select an outcome. In practice, it is also the process of agreeing on the question, defining who may participate, deciding how disagreements are handled, granting authority to act and remembering what the group has previously promised. An isolated vote offers a brief view of preference. A governance system must deal with the much longer relationship between a community and its decisions.

We therefore begin with a principle: the protocol should make the rules of collective action inspectable before the action occurs. A participant should not discover after a vote has closed that the quorum was calculated in an unexpected way or that a delegate's authority extended to a category of decisions never mentioned in the original mandate. When a governance rule can be expressed precisely and validated by software, it should be published in that form. When a rule depends on human interpretation, its scope and responsible interpreters should be declared rather than concealed behind technical language.

A second principle is that participation has several forms. A member who studies a proposal and votes directly participates. So does a member who delegates to a trusted specialist, proposes an alternative, tests an implementation, or records a reasoned objection. Not all these acts need to carry the same formal voting weight. SHB should nevertheless make space for them in the life of a decision. A protocol that measures only the final count risks misunderstanding the work that made a decision possible.

A third principle concerns continuity. Communities are rarely formed for one ballot. They revise objectives, appoint new representatives, encounter mistakes and learn from earlier choices. The value of a common ledger lies partly in preserving the history needed to make those revisions intelligible. The record of a rejected proposal, a minority objection or a withdrawn mandate may become more valuable over time than the brief announcement of a passing result.

These principles lead to a less romantic understanding of decentralization. SHB cannot guarantee wisdom, prevent every alliance of powerful holders or settle questions of legitimacy for every community. Its purpose is to reduce opportunities for an operator to manipulate records or procedures without detection. Decentralization, in this sense, is not the absence of authority. It is a method for defining, distributing, limiting and auditing authority.

The project is consequently not hostile to representation, expertise or leadership. It is hostile to authority that cannot be traced to a stated rule. Good governance may require people to make decisions on behalf of others. What matters is how they receive that mandate, how long it lasts, what it permits them to do and how the community can call them to account.

3.3 One Token, One Unit of Voting Influence

The basic SHB rule is deliberately simple: one eligible SHB token corresponds to one unit of voting influence. A participant holding 400 eligible tokens has four times the weight of a participant holding 100 eligible tokens in a ballot that uses the standard proportional model. The protocol counts units of assigned influence; it does not count unique human beings, infer the quality of an argument or assign moral value to a holder's preference.

This distinction is essential. One token, one unit is not the same rule as one person, one vote. A holder may control more than one address. A single address may represent a group. Tokens may be distributed unequally. Without additional identity and distribution rules, the blockchain cannot determine whether a ballot has equal representation among people. The basic model should be judged for what it offers: transparent, reproducible proportional counting of a defined resource.

For a proposal p, the voting weight assigned to an address a can be expressed as $W(a,p) = B(a,h_p)$, where B is the number of eligible SHB units associated with a at the proposal's specified reference state h_p. If a valid delegation transfers voting authority, the effective weight must also reflect delegated units received and units delegated away according to the published rules. The total influence used to decide the question must never exceed the amount of eligible weight permitted by those rules.

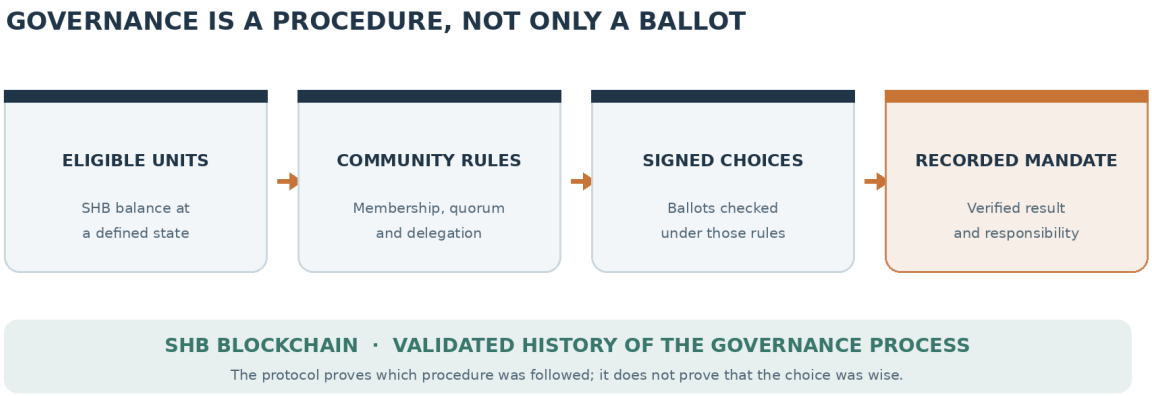
A reference state is important because token balances can change. If votes were continuously recalculated against a moving balance, a person might cast a ballot, move the tokens and allow the new holder to cast another ballot with the same underlying units. We propose to avoid this through an explicitly defined eligibility snapshot, accompanied by rules for later transfers, simultaneous votes, delegated balances and invalidated instructions. The selected mechanism must be deterministic: two honest validating nodes using the same ledger state and proposal rules should compute the same permitted voting weight.

Consider a community with 10,000 eligible voting units. A proposal receives 3,800 units in favor, 2,700 opposed and 500 abstaining; the remaining 3,000 units do not participate. The recorded participation is 7,000 units, or 70 percent. A majority of yes-versus-no votes favors the proposal, but approval under the community's constitution still depends on whether abstentions count toward the approval denominator, what quorum is required and whether the proposal needs a simple or qualified majority. Those choices cannot be left to the person announcing the result.

The straightforward model also creates difficult political questions. If a holder or an allied group controls a majority of eligible units, it may control ordinary decisions. A well-designed ledger can make that concentration visible but cannot make it disappear. Communities might elect to use special thresholds for constitutional changes, time delays for sensitive actions, transparent delegation limits or different membership conditions for particular governance spaces. Any modification of proportional voting should be explicit. A system that quietly substitutes one weighting principle for another has abandoned one of the reasons for using a verifiable ledger.

SHB should not claim that universal fairness follows automatically from token arithmetic. The project aims to make the arithmetic unambiguous and provide tools through which communities can choose, publish and defend the political rules built around it.

FIGURE 3.1 THE SHB GOVERNANCE SEQUENCE



Conceptual representation of the proposed SHB architecture; not a diagram of deployed software.

3.4 Two Kinds of Agreement: Consensus and Governance

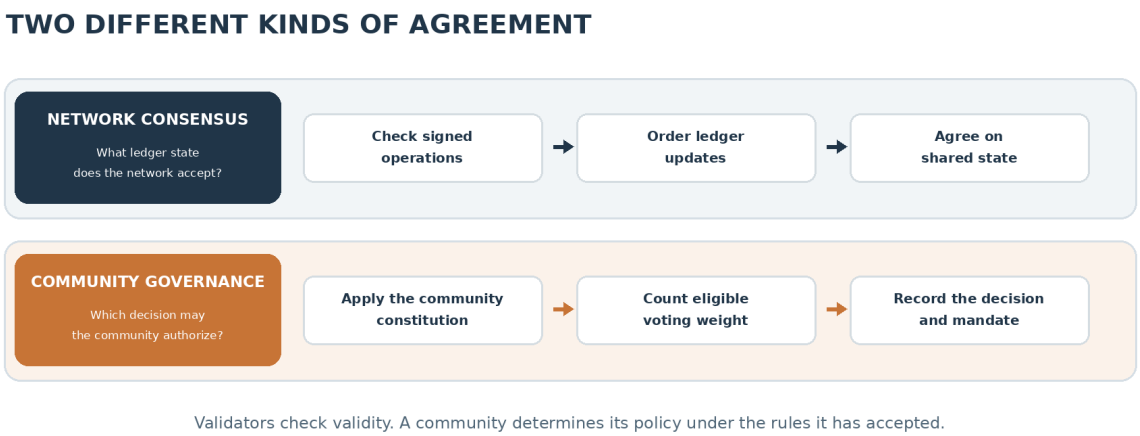
The term consensus is used in distributed computing and in public decision-making, but it refers to different problems. Network consensus is concerned with the order and validity of ledger updates. Governance concerns which authorized choices a community makes under its rules. Confusing the two would place too much political power in the hands of network operators or too much responsibility for technical safety on ordinary voters.

A validator should be able to reject a ballot bearing an invalid signature, an instruction outside its permitted voting period or an attempt to reuse voting units contrary to the protocol rules. It should not reject an otherwise valid ballot merely because the validator dislikes the voter's position. Conversely, a governance majority should not be able to declare invalid signatures valid or instruct a node to accept

contradictory transaction histories without changing the protocol itself through an authorized and technically sound process.

This separation can be stated as a sequence. The ledger first establishes an agreed state. The governance engine then applies the community's known rules to signed instructions within that state. The outcome becomes a new recorded fact, and certain outcomes may authorize later changes. The validator network guarantees only the correctness of the accepted computation under its assumptions; it does not certify that the community made the best choice.

FIGURE 3.2 NETWORK CONSENSUS AND COMMUNITY GOVERNANCE



Conceptual representation of the proposed SHB architecture; not a diagram of deployed software.

This is one reason SHB is proposed as an independent protocol rather than merely a voting page attached to a general-purpose service. Its internal data model can treat proposal identifiers, voting windows, delegated authority, constitutional versions and mandate status as first-class objects. Independent infrastructure does not mean isolation from existing technologies. Digital signatures, Merkle structures, distributed consensus designs and external content storage can all be used without treating the financial transaction as the governing metaphor for every operation.

The choice of consensus method remains a technical design question. Byzantine fault-tolerant approaches offer a relevant body of research, but validator selection, network size, performance under adverse conditions and recovery procedures must be tested before any concrete security claim is made. In particular, the assumption that a voting token is not intended for payments does not remove the need to defend the chain against malicious peers, denial of service or conflicting histories.

3.5 The SHB Network and Its Participants

There is no single SHB user. The proposal involves several roles, some of which may overlap and some of which must remain separate for security or accountability.

A token holder owns eligible SHB units at a particular ledger state. A voter uses those units to submit an authenticated choice. A proposer introduces a question under the rules of a governance space. A delegate exercises voting authority conferred by others, ideally with a stated scope and a means of revocation. A community administrator may organize information or carry out a mandate without having permission to alter an accepted vote. A developer builds the wallet, interface or application that enables those activities. A validating node verifies network rules and maintains the shared ledger.

This list is not intended to become a hierarchy of titles. Its purpose is to prevent accidental privilege. A user interface should not acquire special voting rights because it hosts the discussion. A validator should not become a constitutional court merely because it stores the blocks. A delegate should not gain ownership of tokens merely by receiving permission to exercise their voting weight. Each authorization

needs a precise meaning and a record that allows the relevant parties to determine whether it remains valid.

The community is another important unit of the design. We use the term governance space for a set of participants, rules and proposals that form a coherent decision-making environment. A governance space may establish its own proposal thresholds, ballot durations, membership criteria, delegation permissions and required majorities. One community may decide ordinary technical matters by simple majority and reserve major constitutional revisions for a higher threshold. Another may permit only verified members to initiate ballots while accepting token-weighted votes from a larger set of eligible holders.

The shared chain provides an authoritative sequence of governance records. It need not force every piece of private or lengthy material into that sequence. Supporting documents, meeting notes and technical reports may be maintained elsewhere, with cryptographic hashes anchoring the versions used in a proposal. Such a hash can establish correspondence to the recorded document; it cannot establish the truth of the statements within it. This is particularly relevant when a community acts on external evidence or physical-world events.

A practical interface must also serve members who cannot operate a node or inspect a ledger manually. Wallet software and a readable governance application should present the question, the applicable rules, the deadline, the voting weight and the status of a mandate in language people can understand. The protocol remains responsible for verifiability; the interface is responsible for making that verifiability useful.

3.6 The Principles of Transparency, Neutrality and Autonomy

SHB's architecture is guided by a small set of commitments that can be tested against particular design choices. They are more useful as constraints than as slogans.

First, a decision must have a traceable origin. It should be possible to identify the proposal that produced it, the version of the rules in force, the reference ledger state and the method used to count the submitted instructions. A statement that a community voted is insufficient when those elements are missing.

Second, procedural neutrality must be protected. The same published validity rules should apply to votes for and against a proposal. Where a community appoints moderators or reviewers, their powers must be recorded separately from the validator's mechanical duty to process valid operations. This does not prohibit moderation of abusive content; it requires such moderation not to be disguised as a technical failure of a valid ballot.

Third, governance spaces require autonomy within defined protocol limits. Different organizations will disagree about delegation, membership, quorum and the treatment of abstention. The SHB base layer should establish what records mean and how they are verified while allowing each governance space to define legitimate local procedures. Autonomy cannot extend to rewriting the shared chain or removing the security protections needed by others.

Fourth, delegation should remain accountable. Giving a trusted participant the right to vote on one's behalf is not equivalent to transferring ownership of one's tokens or surrendering every future choice. The intended model favors clearly bounded, inspectable and revocable mandates. The precise effect of revocation on ballots already submitted must be specified in advance; it cannot depend on personal interpretation after the result is known.

Fifth, history must be append-only even when policy changes. A community should be free to reverse a decision through another valid decision, but it should not be able to make the original decision disappear. This is particularly important for minority opinions and for commitments that later prove impractical. Accurate institutional memory includes mistakes.

Sixth, privacy and auditability must be considered together. A permanently public record of every individual's preference may discourage participation or enable pressure. Some communities will accept

public ballots; others may require ballot secrecy. Commit-and-reveal techniques and more advanced cryptographic approaches were already subjects of study at the time of this paper, but they offer different protections and operational costs. SHB should state exactly which privacy properties a particular ballot type provides instead of promising anonymity as a general property of the network.

Seventh, the protocol should favor comprehensible rules over obscure convenience. A cryptographically impressive feature that nobody can audit or explain may be less trustworthy in practice than a modest, well-specified procedure. Open specifications, reproducible tests and independently examined code should be treated as necessary parts of development, not merely as publicity after a launch.

These commitments do not settle every design conflict. Transparency may conflict with privacy. Local autonomy may create complicated interfaces. Delegation can improve participation while concentrating power. The obligation of the protocol is not to hide these tensions. It is to make their consequences visible before people rely on the resulting system.

3.7 The Boundaries of the Protocol

The scope of SHB is intentionally broad, but it is not unlimited. A blockchain can record that a proposal passed under a specified rule. It cannot, by that fact alone, establish that the proposal was lawful, ethical, informed or beneficial. It can authenticate a signature associated with a key. It cannot automatically prove that the key belongs to a particular human being or that its use was voluntary. It can anchor the digest of a document. It cannot determine that the document is accurate.

Nor should every act of governance be pushed into automatic execution. A change in an on-chain rule may be carried out mechanically when its conditions are satisfied. The appointment of a committee, the delivery of a public work or the adoption of an agreement outside the network may require real-world action and independent evidence. The ledger may record an authorization and later attach a report. Unless reliable verification is available, the protocol must distinguish a claim of completion from completion itself.

SHB is not designed as an alternative payment network. Token transfers may still be necessary for allocation, custody and the operation of governance rights. Transaction processing also has costs, and malicious participants can consume network resources. An initial implementation must therefore define realistic protection against spam and a means of supporting validator infrastructure. The absence of a payment objective is not a promise of costless operation or an exemption from economic design.

Neither is SHB a universal electoral standard. A token-weighted community may be appropriate for a voluntary organization whose rules are known and accepted, but different settings require different forms of legitimacy. Public elections and decisions affecting nonconsenting parties raise legal, identity, privacy and coercion questions beyond the assumptions of the baseline token model. Such use cases should not be presented as solved by deploying the SHB ledger.

At the time of this whitepaper, the protocol is a design proposal. Architecture, consensus parameters, wallet recovery, privacy methods and secure deployment require further specification, implementation and testing. The whitepaper may describe ambitious extensions without implying their immediate availability. A public test network, verifiable source code and independent security review are more meaningful evidence than a list of intended features.

3.8 A Long-Term Vision for Decentralized Coordination

Consider a distributed research community whose members live in Bucharest, Paris and Tokyo. They are jointly maintaining an open repository of educational material. They need to choose which translations to produce, appoint editors, revise quality standards and document why a policy was changed. In an ordinary online setting, their forum, shared files, membership list and voting software may each be managed by different administrators. A change in one service can break the historical chain between a discussion and the authority to act.

Under the proposed SHB model, the community establishes a governance space with a published constitution. It defines who may introduce proposals and which SHB balances are eligible. Members discuss alternatives using tools of their choosing, while the proposal record refers to the relevant documents by persistent identifiers and content digests. A ballot begins at an announced state. Votes and delegations are submitted as signed instructions. The network computes the result according to the published rules and records a mandate for the elected editors. Later, the editors attach evidence of their work, and the community can review or replace their mandate through the same procedures.

Nothing in this account requires SHB to become money. The token is useful because it establishes a countable right within an agreed decision process. The surrounding protocol is useful because no single administrator needs to be the unquestioned authority on the ballot history. The system remains imperfect: editors can act badly, voters can make poor choices, and a large token holder can exercise disproportionate influence. Yet the community can see what was authorized, by whom and through which procedure.

The longer-term ambition extends beyond a single organization. Communities may maintain different constitutions while cooperating on common initiatives. Technical standards, shared archives, delegated working groups and proposals involving several governance spaces could all be developed on top of a common ledger. Specialized applications might help participants compare alternatives, track commitments or identify unresolved questions before a vote begins. Some of these capabilities belong in the base protocol; others should remain optional software built by independent developers.

It would be a mistake to claim that every group needs this structure. Where participants already trust a small administrator and decisions carry little consequence, conventional software may be simpler and entirely adequate. SHB is most relevant where a community wishes to preserve independence from an operator, maintain a durable public record or coordinate decisions among members who do not share a single institution.

The vision is therefore not a world in which every interaction is a ballot. It is a world in which communities that choose to govern themselves can do so with clearer rules, more portable records and less uncertainty about the authority behind their decisions. The success of SHB should be judged by the quality of those capabilities, not by the number of transactions it can produce merely for display.

The Design in Brief

SHB begins with a limited, testable idea: one eligible token is one unit of influence under a known rule. The broader proposition is to connect that unit to a durable system of proposals, representation, verification and accountability. The token expresses weight; the community defines purpose; the protocol preserves the procedure. In the next chapters we turn from that relationship to the concrete network architecture needed to make it reliable.

References and Design Foundations

[10] M. Castro and B. Liskov, Practical Byzantine Fault Tolerance, Proceedings of OSDI (1999).

[11] R. C. Merkle, A Digital Signature Based on a Conventional Encryption Function, CRYPTO ’87 (1987).

[12] J. Behrens et al., LiquidFeedback: Principles and implementation of delegable decision-making (project established 2009).

PART II / THE NETWORK BENEATH THE VOTE

Chapter 4

One Ledger, Many Communities

An independent ledger with room for different rules.



Imagine two participants inspecting the same completed vote. One reads a valid result; the other finds that a delegation was counted twice. It hardly matters how polished the voting interface looks. Unless both can reconstruct the same sequence of authorized events, the record is unreliable. SHB is therefore designed around a basic requirement: independently operated nodes must interpret the same transactions, community rules and state transitions in exactly the same way.

The proposed answer is a shared state machine. Nodes receive signed operations, agree upon their order, execute the same publicly specified rules and commit to the resulting state. A community is not a separate blockchain, but a logically distinct governance environment represented inside that common state. The distinction is useful. It allows communities to choose their own procedures without each having to recruit an independent validator network; it also makes the security and governance of the shared infrastructure a responsibility that must be addressed openly.

Here we examine the proposed logical architecture as of February 2020. It is a reference design, not a report of deployed software. Particular choices of database, networking library, validator membership and binary encoding remain subject to testing. The important properties are defined now so that later implementation decisions can be evaluated against them.

4.1 The SHB Blockchain Architecture

We divide the SHB network into four cooperating layers. The peer-to-peer layer moves information between machines. The consensus layer determines the accepted sequence of blocks. The deterministic application layer interprets the operations inside those blocks. The governance layer exposes community-specific behavior to participants and applications. This division is deliberate: a validator should not be able to replace a ballot rule simply by running a different website, and a community should not have to negotiate a new consensus protocol whenever it changes its quorum.

At the network edge, wallets and application interfaces construct transactions. A transaction contains a typed instruction rather than an arbitrary instruction to send money. It may create a governance space, publish a constitutional version, register a proposal, delegate voting authority or submit a ballot. Some transaction kinds necessarily manage SHB token balances, because units of voting influence have to be assigned and transferred. Their existence does not change the network's primary purpose into payments. The operating vocabulary of the chain remains procedural: eligibility, authority, voting weight, deadlines and outcomes.

A node first checks cheap conditions, including message size, syntactic form, transaction identifier and signature. A validating node subsequently verifies each operation against the accepted application state. An operation may be correctly signed yet invalid: a proposal can refer to a nonexistent community; a ballot can arrive after the permitted closing height; a transfer can exceed an available balance. A signed instruction is proof of authorization by a key, not proof that the requested action is permissible.

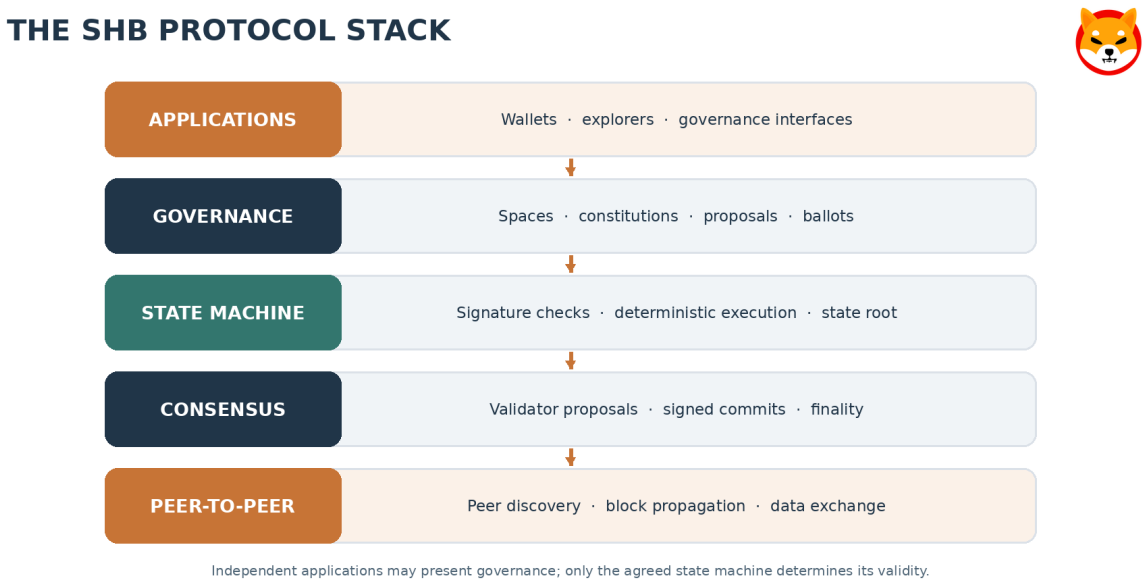
The application state, written S_h, is the complete logical state after block h. A block consists of an ordered list of transactions T_h and an authenticated header. Applying the specified state transition function F gives the next state:

```
S_h = F(S_(h-1), T_h; P_h)
```

Here P_h represents the protocol rules effective at that height. A valid implementation must make F deterministic. For a given previous state, transaction sequence and protocol version, every honest node must compute the same result, including the same errors, state changes and cryptographic state commitment. An implementation that asks an external website for a live value or uses a validator's local clock during state execution would violate this requirement. External facts may be submitted as signed evidence; the protocol must still define precisely how that evidence is treated.

The logical layers are separable but not independent in their security. If the consensus layer orders an unauthorized transaction, the application must reject its effects. If the application accepts a vote in two different ways, consensus cannot repair the disagreement. If a wallet falsely describes a proposal, the ledger may correctly record a choice the user did not intend. A coherent network design must address all three surfaces rather than treating any single layer as sufficient.

FIGURE 4.1 FOUR-LAYER SHB ARCHITECTURE AND APPLICATION BOUNDARY



Conceptual design proposed by the OT Development Team, February 2020; not a diagram of deployed software.

The proposed core should be small. We favor fixed, auditable transaction types over an unrestricted execution environment in the initial release. A general-purpose virtual machine can be valuable, but it also broadens the set of possible execution failures, resource attacks and differences between implementations. The first task of SHB is to make the essential governance state transitions reliable. Programmability beyond these operations can be explored through bounded modules or later protocol extensions.

4.2 Independent Network Operation

An independent chain must have an independent history. SHB therefore requires its own genesis specification, chain identifier, protocol version and validator rules. A genesis file is not merely a convenient starting list of accounts. It is the initial agreement about the network's identity and operating constraints. At minimum, it should describe the initial token supply and allocations, the initial validator set, consensus parameters, transaction limits and the root of the initial application state. The published genesis bytes should be reproducible, and their cryptographic digest should be available from several independent sources.

A node joining the network begins with the chain identifier and a trusted genesis or an authenticated checkpoint. It discovers peers, downloads block headers and verifies the sequence of agreed blocks. To reach the current state, it may replay transactions from genesis or verify an approved snapshot and continue from the associated height. A snapshot is an operational convenience rather than a replacement for proof. The node must have a way to authenticate the state root and the validator information on which the snapshot depends.

Under the envisaged Byzantine fault-tolerant model, validators sign consensus messages and a block becomes final only when the protocol's required supermajority commits it. The exact safety and liveness assumptions will be developed in Chapter 5. A committee with known voting power can provide prompt finality under its fault assumptions, but it introduces a problem of validator admission and rotation. For the first public test network, a small disclosed validator group may be an acceptable engineering expedient. It should never be presented as completed decentralization. The process by which validator membership expands, rotates or is removed must be public, and the initial committee must not be granted an undefined permanent privilege.

The SHB token and the consensus validator vote are different forms of weight. A person may have 10,000 SHB units and no right to sign blocks. A validator may sign blocks but have no special authority to approve a community proposal. Confusing these authorities would make validator operators the hidden legislature of every governance space. Validator selection may eventually be subject to network-level governance, but the two activities must remain logically distinct, with explicit rules for how one can modify the other.

Chain time requires similar discipline. A timestamp in a header is useful for display, ordering within a permissible window and comparison to external documents. It does not prove when an off-chain event occurred. Where the exact eligibility window matters, SHB should prefer finalized block heights and state snapshots to an unrestricted dependence on wall-clock time. The consensus algorithm still needs bounded timestamp validation and an explicit treatment of clocks that differ across machines. Retroactively placing a preferred time inside a header must not be allowed to manufacture earlier voting rights.

Network upgrades are part of normal operation. A change in the serialization format, signature algorithm or state transition logic can divide nodes that disagree about validity. The proposal therefore calls for scheduled protocol upgrades with published activation heights, version compatibility rules, code review and a stated fallback procedure. A community's amendment to its own constitution is not automatically a network upgrade; it should usually be processed as ordinary application data.

4.3 Nodes, Validators and Participants

It is useful to describe roles in terms of verified capabilities rather than imagined ranks. A full node stores the relevant blockchain data, checks blocks and computes application state. A validator is a full node authorized to participate in block consensus under the effective validator set. A relay node primarily helps peers exchange messages. A light client retains smaller proofs and trusted headers while depending on full nodes to provide requested data. A wallet creates and signs a participant's instructions. A governance interface presents proposals and outcomes to humans. One machine may perform several roles, but those roles are not interchangeable.

Each role has distinct trust and availability requirements. A light client can verify certain claims against authenticated headers without storing all votes, yet it must not claim to independently audit an entire ballot if it has retrieved only the final count. A public explorer can make a decision easy to read, but its display is not the authoritative result. A wallet can calculate the expected voting weight locally, yet a network acceptance receipt is needed to establish that its signed instruction reached the ledger. The protocol should make these distinctions visible in both documentation and application design.

The peer-to-peer protocol is expected to support authenticated peer identities, connection limits, message-type separation, version negotiation and basic anti-replay checks. Transaction propagation and consensus messages have different urgency and abuse risks. A large proposal attachment should not be able to starve the channel carrying small consensus votes. Network software should therefore bound message sizes, use admission and bandwidth controls, and separate bulky document transfer from the consensus-critical paths.

Role	Principal responsibility	Does not automatically confer
Full node	Re-execute blocks and verify state	Proposal authority
Validator	Sign consensus messages for accepted blocks	Community voting rights

Role	Principal responsibility	Does not automatically confer
Relay / archive node	Distribute or retain network records	Power to change accepted history
Light client	Verify selected data against trusted commitments	Full historical audit
SHB holder	Exercise eligible voting influence	Validator membership
Delegate	Cast votes only within a recorded mandate	Ownership of delegated tokens
Governance application	Prepare and display instructions	Authority to redefine validity rules

A careful design must also anticipate partial availability. A participant should be able to construct and sign an instruction offline and send it later through any compatible relay, subject to nonce and expiry constraints. The system should not require the continued operation of the original website through which an account was created. If a widely used interface disappears, balances, mandates and proposal histories must remain independently recoverable from the chain and its referenced content.

This is also why the network cannot be defined solely by its founding team. OT may produce the first client, establish test infrastructure and coordinate initial development. The protocol's eventual claim to independence depends on whether others can build compatible clients, operate nodes, retrieve the data and challenge an invalid interpretation without requiring permission from its authors.

4.4 Blockchain State and Transaction Structure

The core data model separates an immutable history of accepted operations from the current state produced by that history. The historical ledger answers what was submitted and accepted. The application state answers what is presently true under those operations. An explorer may show both, but a block explorer's convenient index is not itself a consensus object. If an index fails, a compliant node should be able to reconstruct authoritative state from the accepted history or a verifiable snapshot.

A proposed block header contains a protocol version, chain identifier, height, parent block hash, consensus timestamp, transaction commitment, post-execution state commitment, validator-set commitment and an optional application-receipts commitment. The protocol must specify exactly which values are known when the header is signed. In particular, a commit certificate for a block must not be hashed into that block in a way that requires the validators to sign a hash that already includes their signatures. Such a circular definition is impossible. A certificate may accompany the block or be carried in a later authenticated record under the chosen consensus format.

Header field	Meaning	Validation rule
chain_id / version	Network and interpretation domain	Must match accepted network rules
height / parent_hash	Position in canonical history	Must extend the finalized predecessor
timestamp	Bounded consensus time	Must satisfy the chosen clock policy
tx_root	Commitment to ordered transactions	Recomputed from canonical transaction bytes
state_root	Commitment to resulting application state	Recomputed by deterministic execution
validator_set_hash	Commitment to the effective signing set	Must match authorized validator schedule
receipts_root	Optional commitment to operation outcomes	Must match recorded deterministic receipts

An SHB transaction should be a typed, signed envelope. The proposed common fields are chain_id, protocol_version, sender_public_key, account_nonce, operation_type, operation_body, validity_limit and signature. A network-specific signing domain prevents a signature created for another chain or purpose from being replayed as an SHB instruction. The nonce is monotonic per signing account and prevents

repeated acceptance of an identical authorization. The validity limit prevents a forgotten instruction from being submitted after circumstances have materially changed.

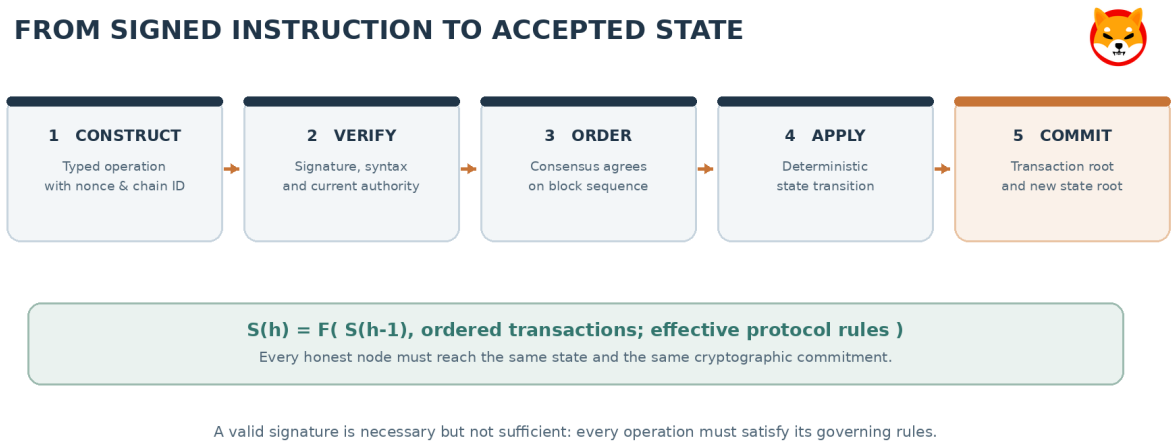
The initial transaction vocabulary can remain compact: SPACE_CREATE, SPACE_RULES_PUBLISH, PROPOSAL_REGISTER, PROPOSAL_AMEND, DELEGATION_SET, DELEGATION_REVOKE, BALLOT_CAST, BALLOT_REPLACE, MANDATE_RECORD and SHB_TRANSFER. The list is illustrative, not a finalized wire protocol. Some operations will be refined in Chapters 7–12; this chapter establishes how every such operation will be carried and interpreted.

```
sign_bytes = EncodeCanonical(chain_id, version, sender, nonce, type, body, expiry)
FORMULA: tx_id = SHA256("SHB/TX/1" || EncodeCanonical(signed_envelope))
```

The explicit domain string illustrates domain separation; the exact byte-level encoding must be frozen in a published protocol specification before a network can claim interoperability. A possible candidate available in February 2020 is CBOR, described by RFC 7049, with a narrowed, mandatory canonical profile. Merely saying that the format is CBOR is not sufficient: duplicate map keys, numeric representations and map ordering must be constrained so that two implementations hash identical bytes. A compact fixed schema would also be reasonable. The decision should be tested across independently written clients before mainnet launch.

An invalid transaction cannot partly modify state. A vote that exceeds eligible weight should leave the ballot state unchanged; a rules update that fails its authorization check should not increment the adopted constitution version. Successful transactions produce deterministic receipts identifying their outcome and affected keys, while rejected transactions must have an explicitly specified inclusion policy. Otherwise a malicious proposer could exploit ambiguity between admission rejection and an included failed operation to manipulate nonces or consume resources inconsistently.

FIGURE 4.2 DETERMINISTIC TRANSACTION PIPELINE AND STATE COMMITMENT



Conceptual design proposed by the OT Development Team, February 2020; not a diagram of deployed software.

Ordering has consequences. If a token transfer and a proposal snapshot appear in the same block, their order can change the eligible balance, unless the snapshot is explicitly fixed to the preceding finalized state. For predictability, the proposed baseline uses the state at the end of the block preceding a proposal's opening height as the voting reference. An implementation must define the precise relationship between opening height, snapshot height and block execution order; it cannot leave that relationship to user interfaces. Delegation, ballot replacement and expiry follow the same principle of explicit state references.

4.5 Communities as Separate Governance Environments

A single ledger should not imply a single electorate. SHB introduces the proposed concept of a governance space: an identified set of rules, roles, proposal records, ballot records and mandates belonging to a particular community. A space has a durable identifier, a constitution reference, an

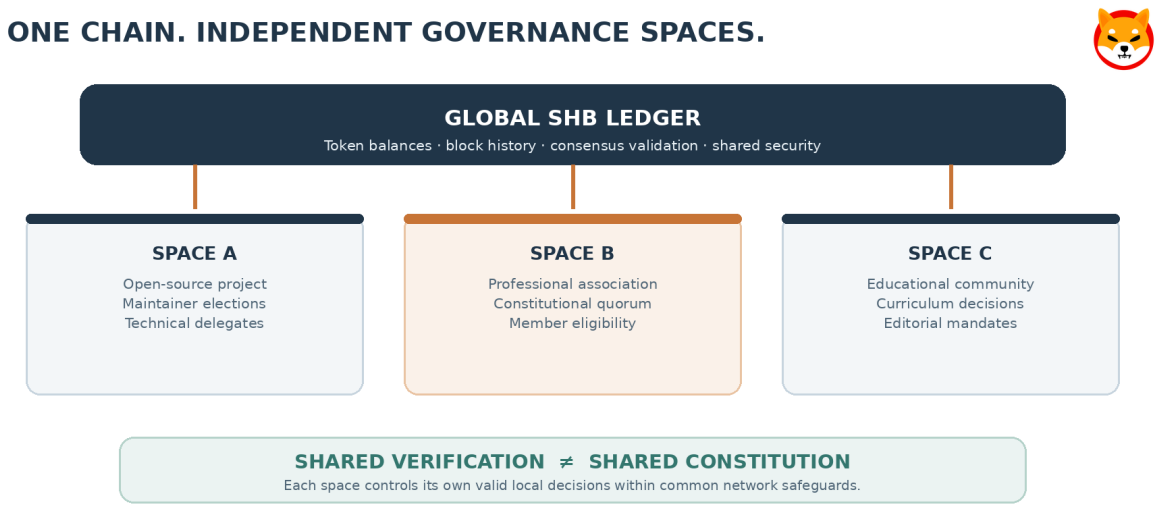
administrator or founding authorization procedure, a current rules version and an append-only history of relevant changes. It is a namespace within the common state machine, not a partition of the physical validator network.

A space may be created by a small technical association that wishes to choose maintainers, by a cooperative that requires qualified majorities for certain decisions, or by a distributed project that wants to delegate technical proposals to subject specialists. These organizations need not endorse one another's decisions. The shared chain supplies a common verification mechanism; it does not appoint itself as their political superior. The local constitution determines what a successful vote means, subject to a narrow set of network-wide invariants that protect consensus and other communities from interference.

The identifier space_id should be derived from an accepted creation operation, rather than from a human-readable name alone. Names can collide and change; identifiers must not. One possible construction is a hash of the chain identifier, creator key and creation transaction identifier. The initial constitution is committed by content digest and assigned a local version. Subsequent constitutional changes are recorded through transactions that must satisfy the amendment procedure currently in force, not whichever procedure a proposer hopes to introduce.

The application keyspace can be partitioned logically. Global records might use keys such as acct/<address>/balance or net/validators/<epoch>, while community records might use space/<space_id>/rules, space/<space_id>/proposal/<proposal_id>, and space/<space_id>/ballot/<proposal_id>/<voter>. These are conceptual paths, not a requirement to use a particular database. Their purpose is to make ownership, authorization and state commitment explicit. A valid operation against one space should not be able to rewrite another space's constitution merely by selecting a different key in a transaction payload.

FIGURE 4.3 SHARED SECURITY WITH ISOLATED COMMUNITY GOVERNANCE



Conceptual design proposed by the OT Development Team, February 2020; not a diagram of deployed software.

The relationship between global SHB balances and local voting eligibility requires particular care. A balance is a global state fact; participation is a local rule. A constitution may use the raw SHB balance at an announced snapshot, require membership in a register, limit the maximum effective weight of a participant, exclude a declared conflict of interest or specify that certain decisions are reserved to a smaller committee. Such variations must be visible and mechanically specified where possible. They do not rewrite the amount of SHB a holder owns on the global ledger.

A token can provide voting influence in two unrelated communities without being consumed by either ballot. That is not double voting: the decisions belong to different electoral contexts. The prohibition is against reusing the same eligible weight more than the applicable rules allow within a

single ballot. The protocol must therefore identify ballots by a pair such as (space_id, proposal_id), and a ballot's state must track an account's effective participation for that exact context. A community cannot borrow another's quorum merely because both refer to the same global token.

For example, suppose an address holds 250 SHB. It may be eligible to vote with a weight of 250 on a software project's maintenance policy and, on the same day, with a weight of 250 on an unrelated association's editorial calendar. Those choices have independent denominators, thresholds and deadlines. If the holder delegates its maintenance-policy vote but votes directly on the editorial calendar, the first space's delegation policy should not silently govern the second. Local mandates are contextual, revocable where allowed, and recorded under the space that issued them.

This separation makes new communities cheap to establish but not costless to operate. Every proposal consumes storage, validator computation and public attention. A design with no admission constraints could be overwhelmed by disposable spaces and enormous metadata. We therefore envisage bounded per-space payloads, rate limits, explicit document size limits and community-specific retention policies for material kept outside the chain. The rules for scarce shared resources must be common and resistant to capture; the rules for local decisions may vary widely.

4.6 Shared Infrastructure and Isolated Governance Rules

Logical isolation should be enforced through state-transition permissions, not by an interface convention. Every transaction should declare its intended space and operation. The validation function first checks network-wide requirements, then resolves the current rules for that space, then verifies that the sender possesses the particular authority required for the operation. This sequence prevents a proposal, delegation or ballot from escaping its namespace through a malformed reference.

The constitution of a space must be treated as versioned executable policy or as a precisely defined set of configuration parameters. A human-readable document may explain the policy, but the parameters that affect validation must be present in an unambiguous machine-readable record. If a constitution says that approval requires 60 percent of votes cast, the machine-readable schema must say whether abstentions belong in the denominator, whether invalid ballots are counted for quorum and whether delegated weight counts as participation. Different answers lead to different outcomes, even if every signature is genuine.

We propose a bounded governance rule set for the first implementation: eligibility source, proposal threshold, voting mode, permitted delegation types, snapshot rule, opening and closing heights, quorum expression, approval expression, amendment threshold, review interval and mandate lifetime. Each field must have a default or an explicit required value. An unsupported field must cause a clear validation failure rather than be ignored. This approach sacrifices some flexibility in exchange for identical interpretation across clients.

Rule category	Example per-space policy	Network-level boundary
Eligibility	Token snapshot plus approved member list	Cannot rewrite global token balances
Ballot	Yes / no / abstain or ranked options	Must use registered deterministic counting rule
Delegation	Allowed only for technical topics	Cannot transfer custody of SHB
Quorum	20% of eligible snapshot weight	Denominator and rounding must be defined
Amendment	Two-thirds approval plus review period	Cannot modify consensus code through local vote
Retention	Pin full discussion documents externally	Ledger commitments remain append-only

A namespace boundary should be tested with deliberately hostile input. A transaction may use another space's identifier in a nested document reference, supply a proposal identifier of the wrong length, overflow an integer field or attempt to invoke a rule version not yet active. Implementations must

reject or safely interpret each case according to a published specification. Failure to isolate state is not merely a software bug in a multi-community system. It can become an unauthorized transfer of institutional authority.

Conflicts between a space's rules and network rules should be resolved in favor of network safety, but that principle cannot be left undefined. The narrow network core should control signature validation, available transaction types, maximum resource use, token accounting, block validity and the rules by which a valid local rule update becomes active. It should not prescribe whether a literary association requires a simple majority or whether a technical committee elects a chair for six months. The more decisions can remain local without threatening shared invariants, the more credible the promise of multiple communities becomes.

The initial release should also support an explicit pause or retirement state for an inactive governance space. Pausing new proposals should not erase historical votes. Retiring a space should not permit reusing its identifier for a new community with different members. In either case, existing audit proofs must continue to resolve against historical state commitments. An archive that changes identities when an organization closes is not an archive on which members can rely.

The network must resist the temptation to use a single administrative emergency key for everything. Operational emergencies may require stopping an unsafe client release or rotating exposed validator keys. They do not justify a hidden right to change every community's ballot outcome. Where emergency authority is unavoidable during early development, its scope, threshold, expiration and use history should be declared. Reducing such authority is part of the decentralization roadmap, not an optional public-relations gesture.

4.7 Data Structures and Cryptographic Commitments

The ledger's practical credibility depends on whether a third party can verify a claim without asking the original publisher to repeat it. Cryptographic commitments allow an operator to state, in a fixed-size digest, that a particular body of transactions or state records existed in a particular form. A Merkle tree can support membership proofs: a light client checks a short path of hashes against a root anchored in a verified block header rather than downloading the entire block. The signature on the relevant consensus commitment establishes which validator set accepted the header, subject to the consensus protocol's trust assumptions.

The proposed hash construction should use an established function such as SHA-256 and separate the domains of leaf and internal-node hashing. For example, a transaction leaf may be H(0x00 || canonical_transaction) and an internal tree node H(0x01 || left_child || right_child). The tree must also define how an odd number of leaves is treated, how the empty tree is represented and how indexes are encoded. Different implementations that make different assumptions about these details will calculate different roots for the same data.

```
tx_leaf_i = H(0x00 || canonical_tx_i)
FORMULA: tx_parent = H(0x01 || left_hash || right_hash)
FORMULA: block_id = H("SHB/BLOCK/1" || canonical_header)
```

A transaction root proves membership in an ordered block when the proof includes the position and sufficient sibling data. It does not by itself prove that the transaction succeeded or that it changed a ballot tally. A receipt proof or a state proof is needed for that stronger claim. A state commitment likewise requires a tree structure whose verification rules are specified. An authenticated key-value tree, potentially a sparse Merkle tree, could support proof that a particular ballot record exists and, with the appropriate construction, that a key is absent. These are design alternatives to evaluate; no specific state-tree implementation is claimed to have been completed.

Privacy follows a different route. A hash commitment does not make a public vote private. If an account, proposal and choice are stored in readable form, the existence of a Merkle root does nothing to conceal the choice. The protocol may later support commit-and-reveal ballots or more advanced cryptographic voting methods, but their resistance to coercion, their effect on tally verification and their

operational complexity must be separately evaluated. We should not confuse cryptographic integrity with anonymity.

Large supporting documents should not be embedded indiscriminately in the blockchain. Proposals may refer to research papers, diagrams, translations and several editions of a policy document. SHB should record a digest, content type, document size, optional author signature and one or more retrieval hints. IPFS, described in work published in 2014, is a possible content-addressed retrieval mechanism. A public HTTP archive can serve the same content if its digest verifies. The ledger makes tampering detectable; it does not guarantee that an external file remains hosted, legally distributable or accessible forever. Availability requires independent replication and clear community responsibility.

There are important consequences for growth. At SHB's specified one-minute target block interval, consider an illustrative block payload of 256 KiB. The raw block payload alone approaches 0.14 TB per year before indexes, state data, redundancy or backups. That is a substantial requirement for volunteer operators. Raising capacity is not an unqualified improvement if it makes independent verification too expensive. The initial protocol should therefore cap transaction sizes, prefer compact commitments and make pruning or archival roles explicit, while retaining a path to audit the original decisions.

Light clients must understand their limits. They can verify a supplied inclusion proof against a trusted authenticated header. They cannot infer that every eligible account received a fair opportunity to vote, that a proposer disclosed all relevant evidence or that off-chain mandates were fulfilled. Technical proofs narrow the space for falsifying a record; they do not replace human scrutiny of what the record means.

4.8 Network Interoperability Considerations

An independent blockchain should not assume that every future application will run inside it. Governance applications may use ordinary web servers for discussion, Git repositories for software changes, file systems for published documents and external services for identity verification. Interoperability begins by specifying the boundary between SHB's authoritative state and systems that merely describe or consume that state.

The most basic interfaces are query and submission APIs. A client should be able to retrieve a proposal's registered identifier, the effective constitution version, a signed account's eligible voting weight at a specific reference height, the acceptance status of a ballot, and the cryptographic proof supporting a result. It should also be able to submit a correctly formed signed transaction through more than one independently operated node. APIs should distinguish pending transactions from finalized ones, and historical query responses must declare which block height they refer to.

Event streams can make the system usable without making them authoritative. An application may subscribe to proposal-created, ballot-opened, ballot-finalized or mandate-expired notifications. These events should be deterministically derived from committed transactions or state changes. If an event service is offline, a client should be able to replay the relevant blocks or query an authenticated index to reconstruct the missing information. Event delivery is a convenience; the ledger is the record of authority.

Communication with other blockchains is more demanding. A remote chain might receive a signed statement that a vote passed on SHB, but accepting that statement requires a trust model. A centralized relay is simple but places trust in its operator. A threshold of independent attestors reduces reliance on one actor but does not eliminate assumptions about the attestors. A light-client bridge may verify the originating chain's validator commitments and state proofs, yet demands careful handling of validator-set updates, finality and proof formats. None should be described as trustless merely because the messages contain hashes.

Our first interoperability goal is therefore modest and testable: publish portable proofs of SHB events, document a stable query format and let independent applications reproduce key state transitions. More ambitious cross-chain authorization should be considered only after the network's finality and proof semantics are settled. A community should not be told that an external application will

automatically honor an SHB decision when neither system has defined who may execute that decision or how conflicting instructions are resolved.

A useful example is an open-source project that adopts SHB for governance while keeping its source code on an existing development platform. A passed proposal may authorize maintainers to merge a particular change. SHB can record the proposal, eligible votes, approval condition and resulting mandate. The actual merge occurs in the external repository, where maintainers retain operational control. The connection is made verifiable through referenced commit hashes and signed implementation reports. This avoids the fiction that on-chain approval automatically changes every external system.

Compatibility is not achieved by forcing all clients to use one programming language or one hosting provider. It is achieved by documenting canonical data formats, state-transition rules, proof verification, version negotiation and known failure modes. Reference software can demonstrate these rules, but independent implementations and adversarial test cases are the stronger test of whether the specification is complete.

Design Consequences

SHB's first architectural promise is simple to state and demanding to fulfill: many communities should be able to maintain distinct rules on one verifiable ledger, without giving any community a hidden right to alter the others. Achieving that promise requires more than putting ballots into blocks. It requires deterministic transitions, authenticated histories, explicit namespace boundaries, honest treatment of validator authority and a disciplined distinction between evidence recorded on-chain and events occurring outside it. The next chapter examines the consensus problem in detail: how the validator network can agree on one history when individual machines may fail or behave dishonestly.

References and Technical Foundations

[13] E. Buchman, J. Kwon and Z. Milosevic, “The Latest Gossip on BFT Consensus,” arXiv:1807.04938 (2018).

[14] J. Benet, “IPFS — Content Addressed, Versioned, P2P File System,” draft (2014).

[15] S. Josefsson and I. Liusvaara, “Edwards-Curve Digital Signature Algorithm (EdDSA),” RFC 8032 (January 2017).

[16] C. Bormann and P. Hoffman, “Concise Binary Object Representation (CBOR),” RFC 7049 (October 2013).

[17] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” OSDI (1999).

PART II / THE NETWORK BENEATH THE VOTE

Chapter 5

Agreement Without a Central Referee

The engineering discipline behind a single accepted history.



5.1 The Distributed Consensus Problem

Take a simple conflict of timing. One member delegates voting authority; another submits a revocation or a ballot at nearly the same moment. If two nodes accept those operations in different orders, they may disagree about who was entitled to vote. That disagreement cannot be settled by showing both nodes the same messages. The messages must also have a single agreed order. Establishing that order, despite delays and faulty participants, is the task of consensus.

The SHB proposal treats this as a state-machine replication problem. Let S(h) be the accepted application state after block height h, let B(h) be the ordered operations in that block, and let F be the deterministic transition function defined by the effective protocol version. Every correctly implemented full node that begins from the same state and processes the same accepted block must calculate an identical successor. Agreement on the order of operations and agreement on how to execute them are separate requirements; neither can be sacrificed.

```
S(h) = F(S(h-1), B(h), Rules(h))
```

Distributed consensus is not the same as the community's decision-making process. A validator attesting that a ballot transaction is properly signed is not expressing support for the ballot's political content. A block can legitimately contain votes both for and against the same proposal. Consensus must determine whether an instruction is syntactically valid, authorized, correctly ordered and permitted by the prevailing application state. Governance subsequently determines the collective consequence under its own voting rules.

SHB consequently distinguishes two electorates. The first consists of validators participating in network agreement. Their consensus weight is defined by the validator configuration. The second consists of eligible SHB participants whose governance influence is determined by the relevant token snapshot and community rules. Owning more SHB must not, by itself, create consensus-signing power. Mixing these two functions would let the political outcome of a ballot alter the technical meaning of a validator's signature.

The network design must defend against more than a failed machine. A participant may send different messages to different peers, withhold a block, sign incompatible histories, imitate a legitimate node, or delay communication strategically. Such behavior is conventionally called Byzantine, after the distributed-systems problem in which some participants cannot be assumed to follow the rules. The practical work of Castro and Liskov showed that useful state-machine replication could be constructed in the presence of arbitrary faults, under carefully specified assumptions [18]. Later work on Tendermint and HotStuff offered additional approaches to the ordering problem [19, 20].

Two properties guide the proposed SHB design. Safety means that correct nodes do not finalize incompatible blocks for the same height while the fault assumption holds. Liveness means that valid instructions can eventually be finalized when enough validators are available and network delays become sufficiently well behaved. Safety is an invariant to protect even during a partition; liveness is

conditional on circumstances the network does not entirely control. The protocol must never confuse temporary failure to make progress with permission to finalize two histories.

5.2 Byzantine Fault Tolerance and the SHB Fault Model

For a simple unweighted committee of n = 3f + 1 validators, a conventional BFT construction can tolerate up to f Byzantine members. A decision generally requires evidence from at least 2f + 1 distinct validators. The familiar four-validator example makes the threshold concrete: one validator may act maliciously or go silent, while three signatures can still constitute the required quorum. Two signatures are not sufficient. A fifth signature is not available because the committee has only four members.

```
n = 3f + 1 ; q = 2f + 1  (equal-weight committee)
```

We propose a generalization to an explicitly weighted validator set V(h), whose members each have an assigned positive consensus weight w(v). Let W be total active consensus weight for a particular height. A commit certificate is acceptable only if its valid, nonduplicated signatures represent strictly more than two thirds of W. The assumed Byzantine share is strictly less than one third of W. Voting weights in this paragraph belong to consensus; they are not automatically derived from SHB token balances.

```
W = sum_{v in V(h)} w(v) ; CommitQC(B) valid iff sum_{v signs B} w(v) > 2W/3
```

The strict inequalities matter. Two independently collected groups each exceeding two thirds of the same total weight must overlap by more than one third. If Byzantine weight is below one third, the overlap necessarily includes honest signing weight. An honest validator that follows the specified lock and round-change rules must not endorse conflicting finalizations. This quorum-intersection argument is necessary but not sufficient by itself: a complete proof also requires a precise proposal, locking, unlocking and recovery protocol. Those details remain subject to formal specification and testing before mainnet release.

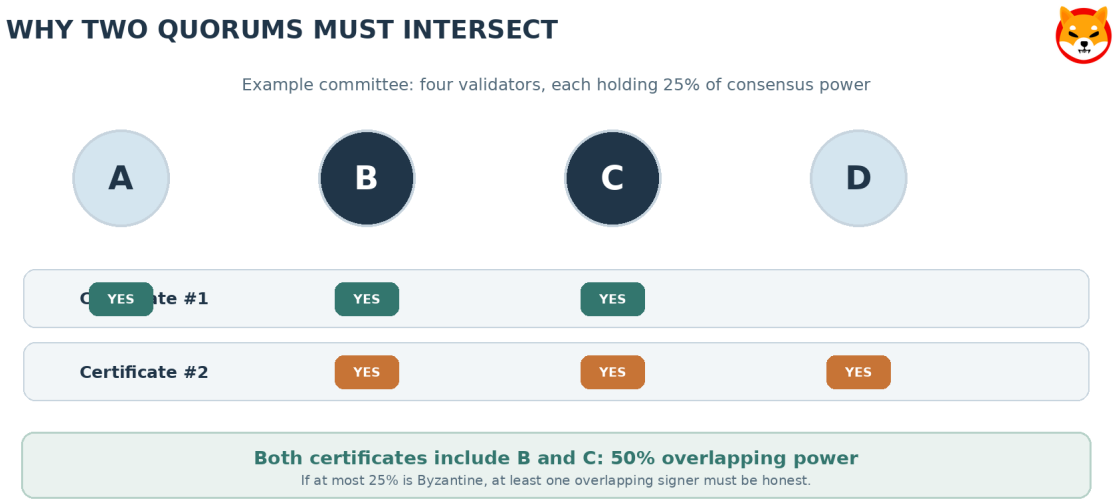
```
weight(Q1 intersect Q2) > W/3, for any Q1,Q2 with weight(Qi) > 2W/3
```

Committee model	Example	Commit threshold	Fault assumption
Equal-weight	4 validators	At least 3 signatures	At most 1 Byzantine validator
Equal-weight	7 validators	At least 5 signatures	At most 2 Byzantine validators
Weighted	Total power 100	At least 67 power	Byzantine power below 33 1/3
Temporary minority online	Total power 100, 60 online	No valid commit	Progress pauses rather than weakening threshold

The synchrony assumption is also explicit. The internet cannot promise an upper bound on message delivery at all times. The work of Dwork, Lynch and Stockmeyer introduced partial synchrony, a model in which some bounds may become effective only after an unknown stabilization time [21]. SHB is designed around a comparable practical assumption: blocks may cease to finalize during a prolonged disruption, but the network should be able to resume after sufficiently many honest validators can exchange messages within eventually reasonable delays.

An adversary who can permanently prevent messages from moving cannot be defeated by a timer. Nor can BFT consensus prevent every undesirable policy outcome. A coalition controlling most governance tokens may pass an unfair proposal while validators correctly record its passage. Conversely, a coalition controlling too much consensus weight may censor ballot transactions even if it has little governance voting influence. The two security questions must be tested independently.

FIGURE 5.1 QUORUM INTERSECTION IN A FOUR-VALIDATOR COMMITTEE



Proposed consensus design, OT Development Team, February 2020. Diagram is conceptual, not a validated deployment.

5.3 Validator Selection and Responsibilities

An independent SHB network requires a published process for establishing the validator committee. The first edition proposes a bounded committee for initial testing, because a small and visible set is easier to instrument, audit and operate than an unrestricted, economically untested validator market. This is an engineering starting point, not a claim that a founding team's handpicked nodes constitute permanent decentralization.

The genesis document should include the exact initial validator public keys, assigned consensus weights, supported protocol version, chain identifier, maximum block size and other rules required for deterministic validation. Each validator should independently operate a full node, retain its signing keys securely and publish enough operational information for the network to assess whether the committee is concentrated in one organization, country or hosting provider. A collection of machines operated by one administrator is not a decentralized committee merely because the machines have different addresses.

The initial admission policy may be governed by a transparent, explicitly limited bootstrapping process. Subsequent admission, removal or reweighting should be represented by on-chain validator-configuration operations that have been authorized under published network-level rules. The design must distinguish this network-level authority from individual community constitutions. A community's vote to elect a project moderator must not silently alter the global validator set; equally, a validator must not be able to remove a community representative simply by producing a block.

An important independence test is whether third parties can verify the admission decision without trusting an announcement from OT. The transaction that changes membership should identify the old configuration, the effective future configuration, the precise activation height and the authorization proof. Validation of that proof must be deterministic. More complicated validator-election methods may be explored later; until then, the protocol should disclose exactly who controls its committee and by which published rule.

A safe initial activation rule is that changes accepted in block h do not change the authority required to finalize that same block. A validator set scheduled at height h is interpreted from state already finalized before the block begins. A proposed delayed-activation window makes recovery, key rotation and light-client verification easier; the precise delay is a protocol parameter to be established through implementation testing. Abrupt simultaneous changes to membership and signing power should be prohibited without a documented transition procedure.

```
V_effective(h) = ValidatorSchedule(S(h-1), h)
```

The validator's task is deliberately narrow. It checks the chain identifier, protocol version, parent commitment, transaction ordering, signatures, application validity, proposed state root and correct consensus messages. It signs at most the messages allowed by its local lock state. It retains durable records of its last signed height, round and step to avoid double signing after a restart. It relays evidence of conflicting signatures where possible. It does not decide whether a member's proposal is wise, whether a minority opinion is persuasive, or whether a community deserves to exist.

The network should make it possible to operate a nonvalidating full node without asking for permission. Such a node is not an extra consensus vote, but it supplies independent verification and public accountability. By rebuilding application state from finalized blocks, independent operators can identify incorrect explorer displays, indexing errors and attempted client-side censorship. Validator membership may be limited at first; the ability to verify history should not be.

Role	Required capability	Primary responsibility	Explicit limitation
Proposer	Validator key active for the round	Assemble and announce candidate block	Cannot finalize a block alone
Validator	Authorized consensus key	Check, prevote, precommit and preserve locks	Does not determine policy merit
Full verifier	Protocol-compatible node	Replay chain and verify certificates	Does not add consensus weight
Governance participant	Eligible SHB balance or mandate	Submit and authorize governance operations	Need not be a validator
Configuration authority	On-chain authorization under network rules	Schedule validator-set modifications	May not rewrite finalized history

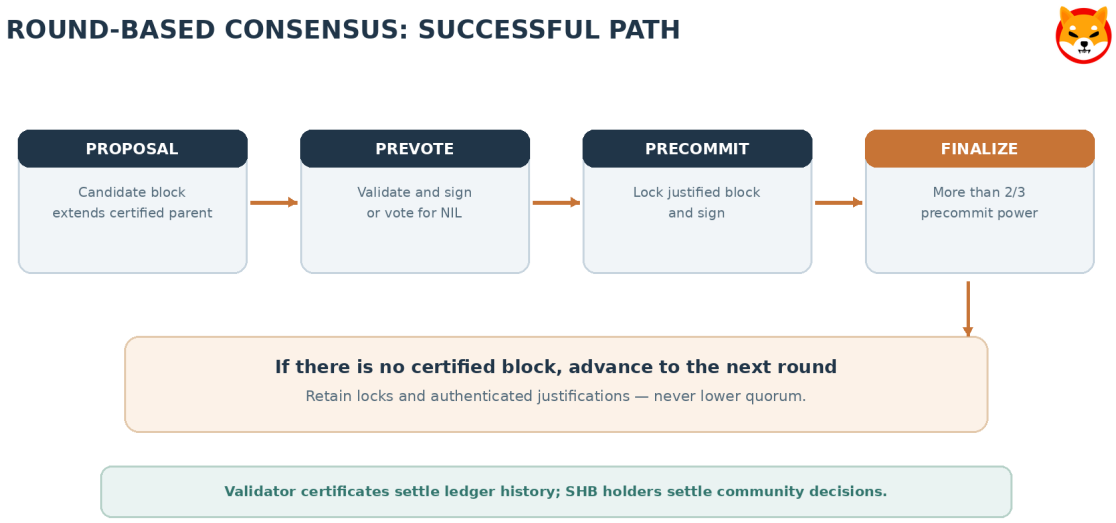
5.4 Block Production and Finality

For the initial model, SHB favors a round-based, Tendermint-inspired procedure with a proposer, a prevote stage and a precommit stage. This is a conceptual choice made from techniques publicly described before February 2020 [19]. SHB does not claim to have invented these stages, nor does this whitepaper represent an independently audited variant. The object of this section is to identify the exact obligations an implementation will have to satisfy.

At each height h the protocol begins with round r = 0. A deterministic function of the effective validator set and the round selects a proposer. The proposer collects pending valid instructions, constructs a candidate block extending the finalized parent and broadcasts that proposal. Validators verify the block, including the transition that would result from applying its operations. If a proposal is missing or invalid, a validator may prevote for NIL rather than inventing a block. A NIL vote is a consensus control message, not a governance abstention.

When a validator observes prevotes exceeding two thirds of the active power for a specific proposal in the relevant round, it may establish or update a lock on that proposal according to the defined lock rules and issue its precommit. If it cannot establish the necessary justification, it may precommit NIL. Upon receiving precommits representing strictly more than two thirds of the same set for one candidate block, a validator can finalize that block for the height, provided the block and its commit evidence are valid. If the current round cannot make progress, the validators advance to a subsequent round instead of lowering the threshold.

FIGURE 5.2 THE PROPOSED SHB CONSENSUS ROUND AND TIMEOUT PATH



Proposed consensus design, OT Development Team, February 2020. Diagram is conceptual, not a validated deployment.

The protocol's commit certificate is an authenticated collection of validator votes covering a single chain identifier, height, round, phase and block identifier. If the block identifier is a hash of the header, it cannot contain its own commit certificate; that would require signing a digest which already includes the signatures. We therefore propose that the certificate be carried alongside the finalized block or linked from a later authenticated record. The block header commits to transactions, the resulting state and the effective validator-set hash, not to a circular self-reference.

```
VoteDigest = H("SHB/CONSENSUS/1" || chain_id || height || round || phase || block_id ||
set_hash)

Finalized(h,B) := VerifyQC( PRECOMMIT, h, r, Hash(B), V_effective(h) ) = true
```

The strongest attraction of this model is deterministic finality after a valid certificate. Unlike probabilistic confirmations, a finalized block is not supposed to be displaced merely because another branch becomes longer. An honest client can verify the certificate and treat the accepted ordering as settled, conditional on the BFT security assumptions and on any applicable validator-set transition rules. This does not imply that an invalid protocol change, a flawed application or a successful threshold-level compromise is impossible. Finality expresses a defined guarantee under explicit assumptions, not a supernatural property of a timestamp.

A candidate block's consensus timestamp also has limited meaning. Validators can agree to include a time value subject to specified tolerance and monotonicity rules, but the chain cannot independently prove the real-world time at which a participant composed a statement. SHB should not market block timestamps as infallible proof of the external date of a document, photograph or manufactured object. They show when an authenticated commitment was accepted under network rules, within the limits of the clock policy.

Algorithm 5.1 — Conceptual SHB agreement round

```
input: height h, round r, effective validator set V(h), finalized parent P
proposer <- SelectProposer(V(h), h, r)
if local_validator == proposer:
    candidate <- BuildCandidate(P, pending_valid_operations)
    broadcast PROPOSAL(h, r, candidate, justification)
on valid proposal matching local safety / lock rules:
    broadcast PREVOTE(h, r, hash(candidate))
on timeout or invalid proposal:
    broadcast PREVOTE(h, r, NIL)
if power(PREVOTE(h, r, B)) > 2W/3 and lock_rules_allow(B):
    persist lock state; broadcast PRECOMMIT(h, r, B)
else when the prescribed stage ends:
    broadcast PRECOMMIT(h, r, NIL)
if VerifyQC(PRECOMMIT, h, r, B, V(h)):
    persist finality certificate; apply B; advance to h+1
else if round timeout expires:
    advance to r+1 without discarding justified locks
// Sketch only: full proposal, lock and timeout state machine is required.
```

The pseudocode is deliberately a readable sketch, not executable reference code. The actual round-change procedure cannot be reduced to "unlock after timeout." An implementation must specify the proofs that justify a later-round proposal, which earlier votes are retained, when locks may move and which messages are safe to sign. The purpose of the sketch is to show responsibilities, not to disguise open proof obligations as a completed consensus algorithm.

5.5 Validator Incentives Without a Payment-Centered Token

Consensus requires continuous costs: servers, storage, bandwidth, operating expertise, software updates and the ability to respond to incidents. Calling SHB a governance token does not remove those costs. Nor does it justify a vague promise that every validator will be rewarded somehow. The protocol needs a credible operational arrangement even if SHB is never accepted at a merchant or listed on a trading venue.

The first development network can operate through explicitly disclosed infrastructure contributions by the founding team and volunteer organizations. A public schedule of participating operators, expected resource commitments and maintenance responsibilities would make the dependency visible. A later network might use a community-approved service budget, membership contributions, externally funded infrastructure grants or limited protocol-managed allocation of resources. Each model changes the network's dependencies. None should be represented as universally sufficient before real costs have been measured.

We do not propose that a validator's right to order blocks should automatically be purchased with a large SHB balance. A stake-based validator market might be investigated, but that would require separate study of concentration, key compromise, economic penalties and the interaction between financial speculation and community votes. The most immediate governance utility of SHB is participation in decisions; turning every consensus task into a token-price problem would pull the design away from that objective.

Operational accountability can begin with measurable service records: missed signatures, independently verified equivocation, failure to upgrade when required and excessive disconnection. These records should not become a centralized blacklist maintained without appeal. Technical removal must follow preannounced rules and distinguish deliberate misconduct from ordinary failure. A validator with a dead server may reduce liveness without being dishonest. A validator that signs conflicting votes for the same height, round and phase has produced cryptographic evidence that demands a different response.

The cost model will also affect how frequently blocks are produced. Empty blocks may be useful for some forms of consistent progress, but an aggressive target interval can consume bandwidth and storage without adding value to governance. The network should measure real proposal submission rates, voting deadlines, peak ballot activity and validator latency before choosing block limits or timeouts. This whitepaper makes no mainnet throughput or confirmation-speed claim.

5.6 Fault Handling and Malicious Behavior

A BFT design assumes that malicious participants will try to make inconsistent statements. The simplest conclusive evidence is a double signature: two distinct, valid votes by the same validator, for the same chain, height, round and phase, naming incompatible block identifiers. Because both messages are signed, any independent verifier can confirm the contradiction without trusting the party that reported it.

```
Equivocation(v) := Verify(sig1,v) AND Verify(sig2,v) AND same_domain_height_round_phase
AND (block1 != block2)
```

Other attacks are less self-evident. A proposer can omit a legitimate transaction, but an omitted transaction is not necessarily proof of malicious intent: the proposer may never have received it or may have rejected it under a mistaken local state. A peer can overwhelm connections, yet heavy traffic may also result from legitimate public interest. Consequently, evidence must be classified by whether it proves misconduct, suggests an anomaly or merely identifies an operational problem. The automated penalties for each category must be different.

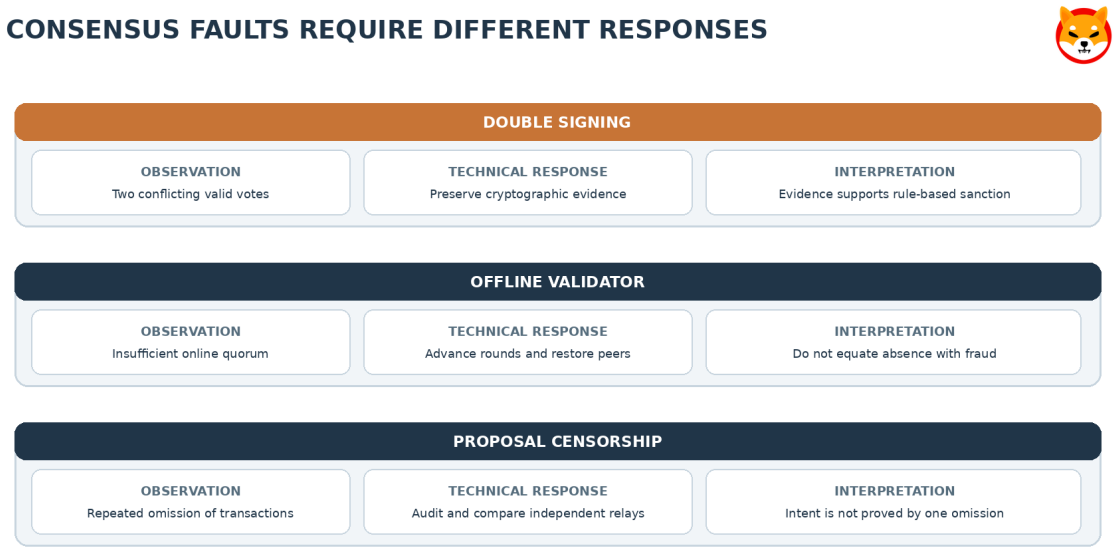
The signed-vote domain must prevent replay across networks and upgrades. A signed precommit created on one test network must never be interpreted as an SHB mainnet vote. A vote should commit to chain ID, phase, height, round, block digest and the validator-set commitment. Peer identity and transport encryption alone do not solve this problem: a cryptographically authenticated network connection is not equivalent to an authenticated consensus statement.

A dangerous case is long-range key compromise. If an attacker obtains historical keys, it may manufacture an alternative signed history aimed at newly connected clients that do not have an authenticated checkpoint or a correct chain-of-trust for validator changes. A light client must therefore verify the relevant signed validator transitions starting from a trusted anchor, or use a deliberately documented checkpoint and trust period. This issue cannot be resolved by simply counting signatures on the latest-looking block.

The protocol must also bound memory and CPU consumption. An untrusted peer should not be able to request infinite verification work by sending deeply nested evidence, extremely large vote sets or pathological proposals. Transaction admission, message-length limits, signature-verification budgets, duplicate suppression and separation between consensus traffic and archival content are practical defenses. These limits must be enforced consistently, with exceptions carefully specified to avoid creating unequal behavior among honest clients.

Threat	Observable symptom	Proposed response
Equivocation	Conflicting signed votes	Verify, preserve, propagate and process evidence
Proposal withholding	Missing proposal in assigned round	Timeout, advance round, rotate proposer
Validator outage	Insufficient timely quorum	Continue retries; never reduce 2/3 threshold
Invalid state root	Candidate disagrees with local replay	Reject proposal and report reason
Peer flooding	Large or repeated unsolicited traffic	Rate limits, bounded queues and peer controls
Governance censorship	Transactions repeatedly excluded	Public relay alternatives, audit and rule-based accountability

FIGURE 5.3 DISTINGUISHING PROVABLE BYZANTINE BEHAVIOR FROM FAILURES



Proposed consensus design, OT Development Team, February 2020. Diagram is conceptual, not a validated deployment.

For SHB, the line between auditability and punishment should remain visible. Recording evidence is a technical act. Suspending a validator can be a technical act if it follows a precommitted rule. Determining the fairness of the rule is a governance question. The network should expose all three steps rather than hiding decisions inside an administrator's control panel.

5.7 Network Availability and Recovery

No architecture can promise uninterrupted operation under unlimited failures. The appropriate goal is predictable behavior during a failure and reproducible recovery once conditions permit. A committee with less than the necessary active voting power must stop finalizing blocks instead of generating a lower-quality imitation of finality. This may delay votes, the opening of proposals or a scheduled mandate. It is preferable to issuing inconsistent governance records that participants cannot later reconcile.

If communication between validators becomes unreliable, successive round timeouts allow the committee to consider new proposers and exchange evidence of what has already been accepted. Timers may expand as repeated rounds fail, giving delayed messages a better chance to arrive after network conditions stabilize. A restarted validator must recover its finalized height and persisted last-signing state before producing another vote. Restoring a backup of the database without restoring the signing state can cause an accidental equivocation. Safe recovery procedures must treat validator keys and signing histories as one operational responsibility.

Algorithm 5.2 — Safe validator restart checklist

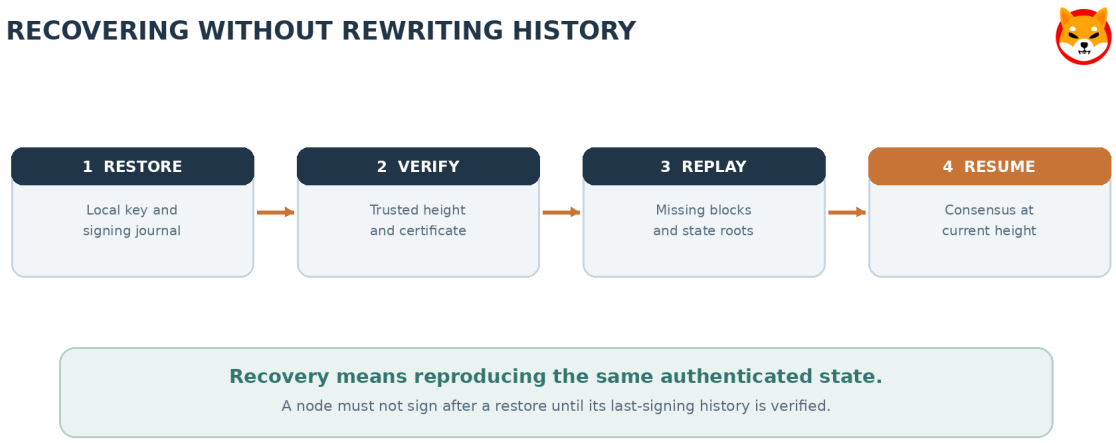
```
load chain_id, local protocol version and trusted genesis commitment
restore finalized block height and verified state root
restore durable last-signing record: (height, round, phase, digest)
verify local ledger and active validator-set commitment
fetch missing finalized blocks and validate their commit certificates
replay deterministic application transitions to the certified state root
reject any attempted signature conflicting with durable signing history
resume consensus only after state, keys and schedule agree
```

A node that has fallen far behind may synchronize by downloading authenticated blocks or by using a snapshot whose state root is committed by a trusted finalized header. A snapshot is a performance aid, not an assertion that a file obtained from a convenient server is automatically genuine. Validators should verify the header chain and any relevant validator-set updates before trusting a snapshot. A verifier who wants the strongest audit should be able to replay the entire history, subject to storage and time costs.

Membership changes deserve special care during outages. It would be unsafe to let the surviving minority declare the absent validators removed and immediately redefine a quorum in its own favor. Validator-set changes should require approval under the previously effective rules and activate through a published schedule. A catastrophic failure of the prescribed process may require a transparent social recovery or a new network, but such an event must not be described as ordinary consensus. Governance legitimacy cannot be manufactured by silently changing cryptographic thresholds.

The primary human-facing recovery promise is continuity of records. An individual who signed a ballot before an outage should later be able to verify whether that transaction reached a finalized block. If the transaction never finalized, the wallet should distinguish pending submission from accepted state and offer safe resubmission with the correct nonce. A user should never be told that a vote counted merely because a web interface received it.

FIGURE 5.4 RESTART AND RECOVERY FROM AUTHENTICATED CHAIN STATE



Proposed consensus design, OT Development Team, February 2020. Diagram is conceptual, not a validated deployment.

5.8 Consensus Performance and Scalability

Consensus performance is an empirical property of an implementation, a validator committee and its network conditions. It cannot be responsibly deduced from the name of a BFT family. Latency depends on message propagation, signature verification, transaction execution, committee size, block size, timeout policy and the number of unsuccessful rounds. Throughput depends not only on bytes per block but also on how expensive the included governance operations are to check and apply.

For a simple round-based design, a useful first approximation for the successful path is the time needed to distribute a proposal, gather prevotes and gather precommits, plus validation and scheduling overhead. It should be written as an accounting identity rather than marketed as a benchmark. The actual implementation must measure each component independently under controlled network delay and packet loss.

```
T_finality ~= T_proposal + T_prevote + T_precommit + T_validation + T_overhead
```

A straightforward gossip scheme may require message traffic that grows rapidly with the number of validators. HotStuff, published in 2019, demonstrates that leader-based designs can aim for more favorable communication complexity under partial synchrony [20]. That is a relevant alternative for later SHB evaluation, but it does not license the SHB team to claim HotStuff performance before selecting and implementing its actual protocol. Similarly, threshold signatures can compress certain certificates, but introduce their own key-management and implementation demands. The initial SHB design may prefer transparent sets of ordinary signatures for inspectability.

For governance traffic, the workload is uneven. A quiet community may publish one decision per month, while a large community may cast hundreds of thousands of ballots within hours. Bursty activity makes admission limits and predictable deadline semantics important. The protocol must specify

whether a ballot cast just before the closing time is counted only if finalized before the deadline, and how users can discover its inclusion. A timestamp issued by an unreliable website is not an acceptable substitute for a chain-verifiable rule.

A realistic evaluation program should include at least the following cases: a healthy four-validator committee, a larger committee with geographically separated nodes, one offline validator, a Byzantine proposer that equivocated, a prolonged network partition, a validator process restarted from a stale backup, a late validator-set transition, a large burst of ballot transactions, and a client verifying a historical commit from a trusted checkpoint. Each test should record safety violations (which must be zero), time to resume after faults, bytes transmitted, CPU cost, disk growth and user-visible confirmation behavior.

Measurement	Definition	Acceptance question
Safety	Conflicting finalized blocks at one height	Does this ever occur within the assumed fault bound?
Recovery time	Interval from network stabilization to next finalized block	Does progress resume without manual history edits?
Finality latency	Submission-to-certified inclusion, with percentile distribution	Is the delay usable for governance deadlines?
Network overhead	Consensus bytes transmitted per finalized block	Does the committee scale within operating resources?
Replay cost	Time to reconstruct state from authenticated history	Can independent verifiers audit the chain?
Admission fairness	Inclusion distribution for independently submitted valid transactions	Is sustained censorship detectable and challengeable?

The first mainnet release should be gated on reproducible tests, independent code review and a published threat model. Claims about transactions per second, resistance to attack and decentralization are otherwise little more than advertising. OT should disclose the initial validator concentration and demonstrate that additional independent operators can join through a documented process. The most meaningful result is not an impressive block interval. It is the ability of different operators to verify the same history without asking a central authority which copy is correct.

Consensus Requirements

The draft specification provides for a specialized governance network, but its consensus problem is familiar: independently operated computers must settle on one ordered sequence despite delay, failure and possible deception. The protocol's answer is not to ask every SHB holder to inspect every block. It is to separate the committee that verifies and certifies an accepted ledger from the communities that use the ledger to make decisions. These communities may disagree passionately while the record of their disagreement remains consistent.

The design described here favors an explicitly configured Byzantine-fault-tolerant validator committee, deterministic state execution, signed commit certificates, persistent lock discipline and cautious membership transitions. None of these elements is presented as deployed SHB software in February 2020. Their purpose in this whitepaper is to make the engineering work concrete enough to challenge, implement and test. The legitimacy of a decentralized governance system begins not with a promise that everyone will agree, but with an arrangement under which no one needs to guess what the agreed record says.

References and Technical Foundations

[18] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” 3rd OSDI, New Orleans, February 1999. https://www.usenix.org/conference/osdi-99/practical-byzantine-fault-tolerance

[19] E. Buchman, J. Kwon and Z. Milosevic, “The Latest Gossip on BFT Consensus,” arXiv:1807.04938, first published July 2018. https://arxiv.org/abs/1807.04938

[20] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta and I. Abraham, “HotStuff: BFT Consensus with Linearity and Responsiveness,” ACM PODC, July 2019.

[21] C. Dwork, N. Lynch and L. Stockmeyer, “Consensus in the Presence of Partial Synchrony,” Journal of the ACM 35(2), April 1988.

PART II / THE NETWORK BENEATH THE VOTE

Chapter 6

The Right to Participate

Cryptographic authority, wallet design and the limits of digital identity.



A careful voting system can still fail at its first step: establishing who authorized an instruction. Published thresholds and an auditable ledger mean little if a ballot can be submitted in another participant's name. Nor is an account truly under its holder's control if a platform operator keeps the signing key. SHB therefore places key ownership, signature verification and explicit authorization at the centre of participation.

SHB therefore treats participation as a sequence of distinct claims. A key proves control of an account. An account may hold SHB units. A governance space defines whether those units are eligible for a particular decision. A valid instruction expresses the account controller's authorized choice under the applicable rules. The blockchain records that instruction only after its signature, sequence, scope and timing have passed deterministic validation. These are related claims, but they should never be compressed into a vague promise that the system has verified the voter.

Two network parameters are fixed for the design described in this paper. SHB has a total supply ceiling of **100,000,000 SHB**, and its consensus schedule targets a new block every 60 seconds. The first parameter bounds the native resource used in token-weighted governance; the second establishes the nominal cadence of the ledger. They do not establish a limit of 100 million people, and one minute does not guarantee that a vote becomes final one minute after a participant presses a button. The supply ceiling does not imply that new tokens are created with each block. The rules for initial allocation and any staged circulation require their own explicit specification.

This chapter defines a proposed identity and authorization layer with that distinction in mind. It favors established signature algorithms, an inspectable transaction envelope, independent wallets and verifiable eligibility checks. It also acknowledges problems that cannot be solved merely by publishing a public key: stolen devices, forgotten secrets, identity duplication, coercion, unequal access and the disclosure of voting preferences. Those problems belong inside the design discussion rather than in a footnote added after deployment.

6.1 SHB Addresses and Cryptographic Keys

An SHB account begins with a cryptographic key pair, not a name or an email address. A private signing key remains under the account controller's custody; a public verification key may be disclosed to the network. For a candidate Ed25519 implementation, the public key is 32 octets and the signature is 64 octets, following the EdDSA construction documented in RFC 8032 in January 2017 [22]. Selecting Ed25519 for an initial implementation would provide a mature basis for digital signatures, but the choice must still be accompanied by test vectors, implementation review and well-defined encoding rules. No cryptographic primitive should be described as secure merely because it has a familiar name.

The account address is a compact reference derived from the public key and an algorithm identifier. SHB could define an address digest using SHA-256 over a fixed domain string, the signing scheme identifier and the canonical public-key bytes. A checksummed, human-readable representation would then encode the digest with a network-specific prefix. Bech32, proposed in 2017, offers an existing checksummed base32 format that may be adapted for this purpose [23]. The example prefix below is a proposal for SHB, not an issued address format or evidence of a deployed wallet.

```
address_digest = SHA256("SHB-ADDRESS-v1" || key_scheme || public_key)
```

The exact address width, error-detection rules and conversion between bytes and display strings are consensus-adjacent specifications: clients must agree on them, and the first release should publish reference vectors for both acceptable and rejected inputs. The format should distinguish a production network from a test network; otherwise a user can be led to sign or transmit an instruction on a system whose rules differ from those displayed. Different chain identifiers must also be carried in the signed transaction itself. An address prefix is helpful for human error detection, but is not sufficient replay protection.

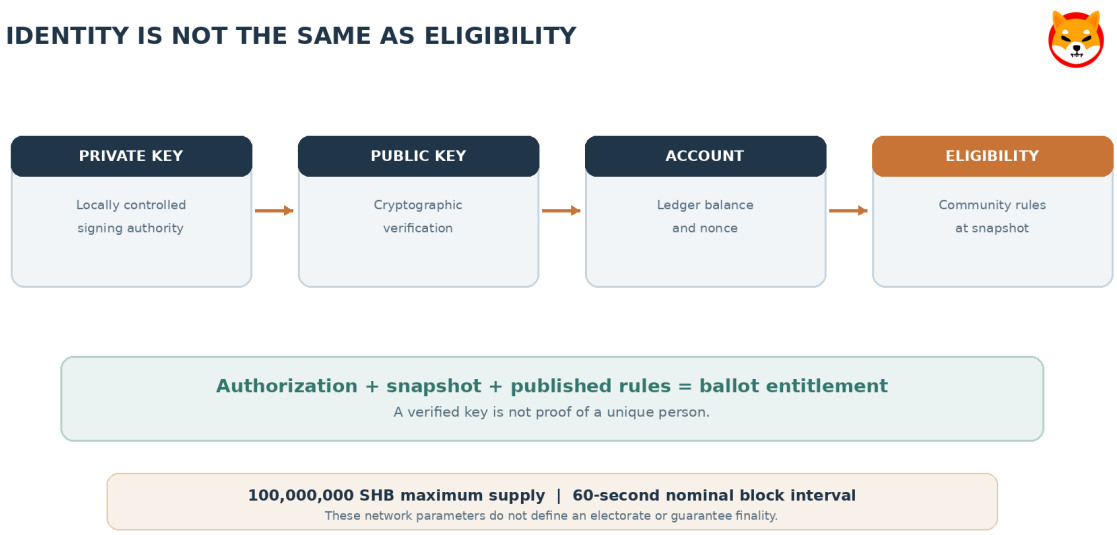
An address is a pseudonymous identifier. It does not prove that a unique individual controls it. One person may create thousands of keys without requesting permission, and one legal organization may deliberately use a shared account. That property is useful for accessibility and independent participation, yet it excludes claims of person-level equality unless a governance space introduces additional membership evidence. SHB's baseline is *one eligible token, one unit of influence*, not one person, one account or one vote.

A single key should not necessarily control every power associated with an account. The protocol may later support a distinction between an ownership authority, a daily voting authority and a recovery authority. Such separation reduces the need to expose a long-term secret during routine participation. It does not require a different currency or a second token, only clearly typed authorizations and expiration conditions. For the initial architecture, direct Ed25519 authorization remains the smallest auditable starting point; delegated or multi-key account policies are explicit extensions rather than invisible exceptions.

An ordinary balance of SHB does not confer validator authority. The validator committee described in Chapter 5 has separate consensus signing keys and a separate membership configuration. A holder with a large balance may exercise greater governance influence under a token-weighted constitution, but cannot manufacture a valid block certificate with an ordinary participant key. The protocol should publish this distinction prominently because both operations involve signatures and are otherwise easy to confuse.

Parameter	SHB design value or status	Consequence for account design
Maximum native token supply	100,000,000 SHB	Bounds aggregate token-denominated voting influence; circulation policy is separate
Nominal block interval	60 seconds	Provides expected scheduling cadence, not guaranteed wall-clock finality
Candidate account signature	Ed25519, RFC 8032 (2017)	32-octet public keys and 64-octet signatures, subject to implementation review
Address representation	Proposed checksummed network-prefixed encoding	Must be published with precise conversion rules and reference vectors
Governance unit	One eligible SHB = one unit of influence	Eligibility is ballot-specific and may depend on an authenticated snapshot

FIGURE 6.1 CRYPTOGRAPHIC IDENTITY, ACCOUNT STATE AND BALLOT ELIGIBILITY



SHB authorization and participation architecture, OT Development Team, February 2020. Proposed specification; not a completed or audited deployment.

6.2 Digital Signatures and Transaction Authorization

The digital signature is the boundary between a displayed preference and an authenticated instruction. A participant must authorize the exact operation that the network will interpret. If an interface says *vote against Proposal 17* while the signed bytes identify a different proposal, a cryptographically valid signature would not make the interaction honest. A secure wallet must therefore display the governing context before signing: the network, community, proposal, chosen option, permitted weight, expiration and any delegated authority being exercised.

Chapter 4 introduced a common typed transaction envelope: chain_id, protocol_version, sender_public_key, account_nonce, operation_type, operation_body, validity_limit and signature. Chapter 6 makes the signing rule explicit. The signature is calculated over a fixed serialization of all authorization-relevant fields except the signature itself, preceded by a domain-separation constant. The signing bytes must use one canonical encoding for integers, strings, lists and maps; allowing several equivalent encodings would invite discrepancies between wallets and validators. A field omitted from the signing payload can potentially be altered in transit without invalidating the signature.

```
m = "SHB-TX-v1" || EncodeCanonical(chain_id, protocol_version, sender_public_key,
account_nonce, operation_type, operation_body, validity_limit)

signature = Ed25519.Sign(private_key, m); accepted_only_if Ed25519.Verify(public_key, m,
signature) = true
```

The transaction identifier should be derived from a declared canonical representation of the signed envelope, including the signature where appropriate. SHB must not let one client hash JSON text while another hashes a binary format and assumes the identifiers will match. Signature and address algorithms should carry explicit version identifiers so that a later migration does not create ambiguous account identities. The design uses Ed25519 as a practical candidate rather than an unchangeable assertion about software not yet released.

The account nonce is monotonically increasing for each signing account. It prevents an old, already executed authorization from being accepted again as a new state transition. It also creates an ordering constraint for multiple instructions issued by the same key. An unsigned web request, an intercepted signed envelope or a valid transaction sent through several relays should never result in repeated voting effects. A duplicate byte sequence can be propagated many times; the ledger must interpret it at most once under the relevant nonce and ballot rules.

An expiry field adds a second boundary. A participant who signs a ballot offline may intend it to count only before a known closing height, or may wish a delegation instruction to expire if it is not

included by a stated block. The wallet should display the limit in both block height and approximate civil time, clearly identifying the latter as an estimate. With a nominal one-minute interval, 60 blocks represent approximately one hour and 1,440 blocks approximately one day under uninterrupted production. Delays, missed slots and consensus recovery can make actual elapsed time longer. Neither an ordinary computer clock nor a website countdown is the authoritative source of ledger inclusion.

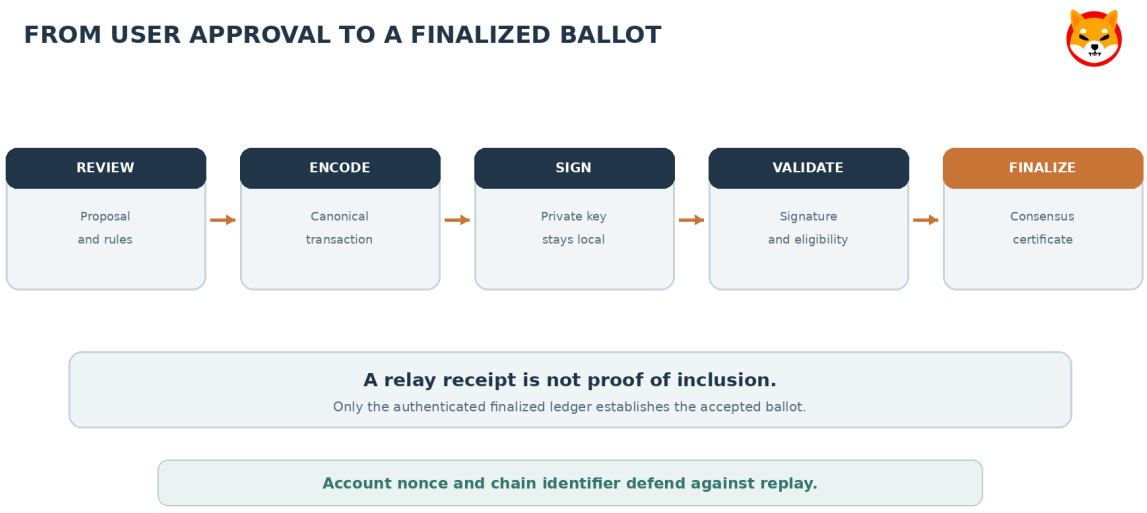
Algorithm 6.1 — Construct and authorize an SHB governance instruction

```
fetch authenticated chain_id, latest finalized height and account_nonce
fetch proposal_id, space_id and effective rules at a certified state
show human-readable operation_type, choice, authority and expiry to signer
require explicit user approval of the complete intended action
body := EncodeCanonical(space_id, proposal_id, choice, optional_authority)
msg := Domain("SHB-TX-v1") || EncodeCanonical(chain_id, version, key, nonce, type, body,
expiry)
sig := Ed25519.Sign(local_private_key, msg)
envelope := (chain_id, version, key, nonce, type, body, expiry, sig)
store only nonsecret transaction receipt and the intended action summary
submit envelope through one or more independent relay nodes
mark status as submitted; do not mark the vote counted until inclusion is finalized
```

A signature proves only that the holder of a signing secret authorized the serialized message. It cannot establish whether the signer understood a lengthy proposal, whether the signing device was compromised, or whether another person pressured the signer. A wallet's responsibility is to reduce avoidable deception through clear presentation, reproducible data retrieval, and local confirmation. A protocol's responsibility is to reject unauthorized state transitions even when a malicious interface requests them.

The application validation path therefore has two levels. A stateless layer checks canonical structure, bounds, algorithm identifiers, domain separation and signature validity. A stateful layer checks account nonce, applicable authority, balance-related eligibility, proposal rules, deadline and whether an earlier ballot has already consumed the account's permitted voting action. Every honest node must implement the same distinction. Otherwise a seemingly harmless difference in signature parsing or ballot interpretation can lead to inconsistent state roots and threaten the consensus model of Chapter 5.

FIGURE 6.2 LOCAL AUTHORIZATION AND THE ON-CHAIN CONFIRMATION SEQUENCE



SHB authorization and participation architecture, OT Development Team, February 2020. Proposed specification; not a completed or audited deployment.

6.3 Wallet Architecture

A wallet is often described as an application for holding tokens. In SHB it is better understood as an instrument for holding authorization. The wallet should reveal a user's eligible influence without asking the user to surrender the means by which that influence is exercised. A useful design would divide the

application into a key-management component, a transaction builder, a governance reader, a local policy checker and an independent submission client. The attractive web page through which a participant reads a proposal should not need possession of the participant's signing secret.

The signing component may be implemented in desktop or mobile software, or connected to a dedicated hardware signer. For a browser interface, an isolated local signing environment is preferable to transmitting a secret to a remote server. The network can remain usable even if the official website is unreachable: a participant should be able to export an unsigned instruction, review it in a separate signer and broadcast the result through another compatible node. Each route introduces interface and operational complexity, but dependence on one website would contradict SHB's reason for existing.

Human backup presents a different challenge from signature verification. BIP-39, assigned in 2013, describes mnemonic sentences that encode entropy and generate seeds for deterministic wallets [24]. An SHB wallet may adopt that established approach, provided that it states the exact word list, checksum, passphrase treatment and derivation algorithm. It must not assume that a key derivation method specified for one elliptic curve works unchanged for Ed25519. Curve-specific derivation schemes, such as hardened Ed25519 derivation based on existing published approaches, require their own interoperability tests. The phrase is not an account recovery service: anyone obtaining a working secret can usually authorize the account.

Local encryption protects wallet files against casual inspection and some forms of device theft, but it does not turn a weak password into a safe cryptographic key. The application should use a memory-hard password-based key derivation mechanism available in the relevant operating environment, authenticated encryption, secure random generation and clear backup guidance. It should avoid logging plaintext secrets or leaving temporary key material in exported files. A testing program must include malicious clipboard replacement, incorrect QR codes, browser extensions, operating-system malware and misleading approval screens.

A browser may also use stronger authentication to protect an interface session. The W3C WebAuthn Level 1 Recommendation was published in March 2019 [25]. Such authentication could reduce password-related compromise of a governance website, but it is not automatically a replacement for a protocol-valid SHB signature. A website session credential and an on-chain account authorization establish different properties. The application must not present successful login as proof that the network accepted a ballot.

The wallet needs a local view of proposal rules. Before accepting a vote request, it should query at least two independent sources when possible, verify the relevant finalized header or trusted proof, and compare the proposal identifier and constitution version with the display text. Full verification will be easier for a locally connected node; a light client may authenticate inclusion or state proofs while still relying on other parties to supply the proposal's full explanatory material. The interface should disclose which portions were cryptographically verified and which were merely retrieved from a server.

Wallet component	Responsibility	Essential failure control
Key vault	Generate, encrypt, store and use private signing material	Never export secrets to ordinary network requests
Governance reader	Display proposals, rules, snapshots and balances	Bind displayed details to authenticated identifiers
Transaction builder	Produce deterministic signed-message bytes	Canonical encoding and independent test vectors
Local policy checker	Warn about expiry, authority, duplicate votes and wrong network	Confirm the exact operation before signing
Relay client	Broadcast signed envelopes	Support independent endpoints and idempotent resubmission
Receipt verifier	Verify inclusion and finality information	Distinguish submission, acceptance and finalized execution

The minimum viable wallet should resist a more prosaic error: a participant may unintentionally vote on the wrong proposal. Large decentralized organizations can publish several motions with similar titles and different constitutional consequences. A wallet should identify each ballot by both its governance space and proposal identifier, and show the active rules version. User interface decisions here are part of the security architecture. A signature cannot correct an ambiguity that the human never saw.

6.4 Voting Eligibility

SHB's supply is bounded at 100,000,000 tokens, but eligibility is not equivalent to total supply. Some tokens may be held by accounts that are not members of a particular governance space; others may be excluded by a published rule, subject to a vesting restriction or delegated to an authorized representative. A ballot's eligible denominator must be computed from an agreed ledger state and from the rules effective for that ballot. The figure of one hundred million is therefore a network parameter, not a claim that every proposal has one hundred million eligible voting units.

For proposal p within governance space s, the proposed baseline is a finalized reference height $h_ref(p)$ selected at the proposal's authorized opening procedure. Let $B(a,h)$ be the SHB balance of account a at finalized height h. Let $E(s,p,a,h)$ be an eligibility function that evaluates to zero when the account is ineligible and to one when it satisfies the public criteria. The direct, pre-delegation entitlement is then $W0(a,p) = B(a,h_ref(p)) \times E(s,p,a,h_ref(p))$. If an account is validly delegated an additional voting mandate, a separate rule determines the amount it may cast on behalf of others. That mandate must not alter ownership of the token balance.

```
W0(a,p) = B(a,h_ref(p)) * E(s,p,a,h_ref(p));   0 <= Sum_a W0(a,p) <= 100,000,000 SHB
```

The snapshot solves a familiar double-counting problem. Suppose an account owns 250 SHB at the designated reference height, votes, and later transfers those tokens. Under snapshot voting, the buyer does not acquire a second right to vote on the already-open proposal merely because the balance has moved. The buyer may be eligible for a future proposal whose reference height falls after the transfer. The rights attached to a snapshot and the rights attached to current ownership must be distinguished in client interfaces, especially when ownership transfers remain technically possible during an open ballot.

Delegation introduces a second accounting problem. If A holds 100 eligible units, delegates those 100 to B, and B votes with the mandate, A must not cast a separate direct vote with the same units under that proposal. The effective voting budget for each ballot can be modeled as a set of nonoverlapping claims on snapshot entitlements. Revocation, overriding a delegate, or changing a ballot is permissible only when the space's constitution defines an ordering rule that allows the ledger to settle which instruction remains effective. A delegate receives authority to cast eligible weight, not an increase in the number of SHB tokens issued by the chain.

Algorithm 6.2 — Validate a token-weighted SHB ballot

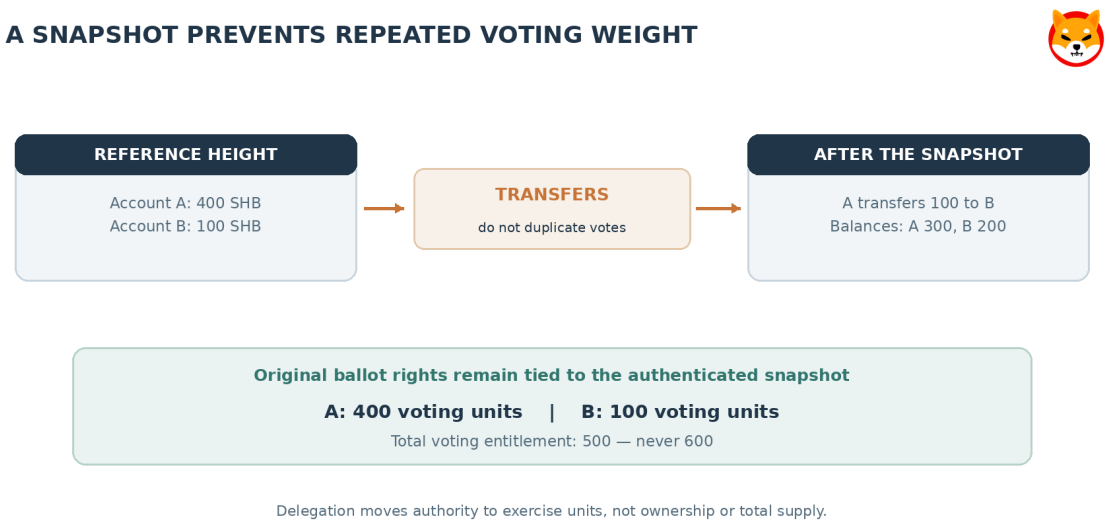
```
assert transaction signature, account nonce and network domain are valid
locate finalized proposal snapshot and active space constitution
assert ballot inclusion height is within authorized voting window
assert account or delegate has the required voting permission
resolve nonoverlapping snapshot entitlements after valid delegations
compute weight := authorized, unconsumed entitlement for this proposal
reject if weight <= 0 or if instruction conflicts with the replacement policy
update proposal-specific ballot record atomically
issue receipt containing accepted weight, proposal_id and finalized inclusion reference
```

SHB should define voting windows primarily in authenticated block heights because all honest nodes can agree on a finalized height. At the target interval of one minute, an interval of 2,880 blocks is nominally two days. But the real duration may differ if consensus pauses, and proposals should not be opened and closed using an untrusted browser clock. Communities that need real-world calendar deadlines require a separately documented mechanism relating consensus timestamps, allowable drift, finality and a policy for prolonged outages. The protocol must not hide the distinction between expected time and guaranteed time.

A community may also impose conditions unrelated to token ownership: membership in a research group, possession of a credential, participation before a cut-off date or a particular delegation mandate. These conditions must be explicitly published rather than introduced after ballots have been accepted. Where private membership lists are used, the network should commit to an eligibility set or a verifiable authorization process so that later auditors can determine which rules applied without exposing unnecessary personal data.

Case	Baseline treatment at proposal snapshot	What must not happen
A holds 400 SHB and is eligible	Voting entitlement begins at 400 units	A second account cannot spend the same snapshot entitlement
A transfers 100 SHB after snapshot	Proposal entitlement remains as defined at snapshot	A recipient cannot gain a second vote for that same ballot
A delegates 300 units to B	B may use 300 units if mandate is valid	A and B cannot both cast those 300 units
Membership credential has expired	Eligibility function rejects or limits according to rule	Wallet cannot silently substitute current balance for eligibility
Proposal closes at a defined height	Only transactions accepted under the window rule count	Website submission after deadline cannot override chain state

FIGURE 6.3 A FIXED-HEIGHT SNAPSHOT PRESERVES THE VOTING DENOMINATOR



SHB authorization and participation architecture, OT Development Team, February 2020. Proposed specification; not a completed or audited deployment.

6.5 Membership Proofs

An open blockchain is not the same as an open electorate. Some governance spaces may welcome any token holder. Others may need a verifiable membership boundary. An open-source development community might grant proposal sponsorship to known contributors while allowing a larger population of holders to vote. A professional association might require evidence of a current membership certificate. Neither arrangement can be inferred reliably from an account's token balance alone. SHB's identity layer should allow a space to declare the credentials it recognizes and the procedure by which their status is checked.

A credential is a signed assertion by an issuer about a subject. The W3C Verifiable Credentials Data Model 1.0 became a Recommendation in November 2019 [26]. Its existence provides a relevant conceptual foundation for machine-verifiable membership assertions as of the date of this paper. SHB does not require every participant to obtain a global legal identity or place personal documents on the

ledger. The design presented here instead lets each governance space name the issuers it trusts, the claims that confer eligibility, and the expiration or revocation checks that must succeed.

A typical credential could state that a particular public key belongs to an active member of an organization for a specified period. A validator or governance execution engine would not need to examine the member's home address, date of birth or identity document merely to check eligibility. It could verify a cryptographic issuer signature over a minimal claim and confirm that the credential is current. However, even a minimal credential can be linkable across organizations. The design should prefer disclosure of the least information necessary for the decision, and should not pretend that public cryptographic verification automatically preserves privacy.

Membership credentials introduce trust in issuers, not just mathematics. If a community accepts credentials from a single secretary, that secretary can influence who is eligible to vote. The blockchain can make the accepted credential and the rules governing it visible, but it cannot prove that an issuer behaved fairly when deciding whom to approve. Governance spaces should therefore be able to document issuer selection, appeals, expiration periods, revocation authority and, where suitable, acceptance of multiple independent issuers.

A credential may also establish a role rather than a person. A software maintainer can possess permission to propose a version change; a project treasurer can be authorized to submit an execution report; a temporary committee can be allowed to certify that an offline meeting occurred. These rights should be scoped to particular actions and should not automatically award additional token-weighted voting power. The SHB protocol needs to keep the dimensions of *membership*, *authority* and *weight* distinct so that one kind of trust does not silently become another.

A final limitation is resistance to duplicate identities. A credential issued by a trusted body can provide evidence relevant to personhood or organizational membership, but a public key alone cannot prevent one individual from creating more than one account. Any proposed one-person-one-vote experiment would need a separate issuance, uniqueness and privacy model, including a way to challenge errors. SHB's native proportional voting model does not make that promise. Honest disclosure of the boundary protects the integrity of both the token model and any later identity experiments.

6.6 Pseudonymity and Participant Privacy

Participation on a public ledger leaves evidence. A participant may use a name unrelated to a civil identity, yet the sequence of signed operations can reveal interests, affiliations, recurring collaborators and voting preferences. A wallet can also leak information through the servers it contacts, the time at which it requests a proposal, its Internet address or the correlation between accounts. The absence of a full legal name is not the presence of anonymity. SHB should describe the default as pseudonymous authorization, not private voting.

An ordinary public ballot makes the voter's chosen option visible to anyone inspecting the recorded operation. That is a legitimate design for some small communities, especially where delegates are expected to explain their votes. It may be unsuitable for professional associations, sensitive internal elections or decisions where coercion is plausible. The protocol should permit communities to select a public or privacy-enhanced ballot type only when the chosen method has been independently specified and verified. A cryptographic hash recorded with a ballot is not, by itself, a guarantee that an observer cannot infer the vote.

Commit-and-reveal voting is an established candidate for reducing early disclosure. In a commitment phase, a participant publishes a digest derived from the choice and a secret random value. During a later reveal phase, the participant discloses the choice and randomness so that the commitment can be checked. This can prevent other participants from learning the choice before reveal, assuming adequate entropy and domain separation. It does not permanently conceal the revealed ballot, solve vote buying, prevent coercion or guarantee that every committed vote is revealed. Late or missing reveals require published treatment.

The protocol may investigate zero-knowledge proofs for proving limited statements, such as possession of a recognized credential or membership in a committed set. Suitable proof systems and accumulators existed in research before February 2020, but deploying them responsibly would require a concrete circuit, trusted-setup analysis where applicable, verifier cost estimates and audits. The whitepaper should not claim that SHB already supports anonymous, coercion-resistant elections merely because such research exists. A privacy feature becomes part of a protocol only after its assumptions and failure cases are documented.

Some metadata can be limited without changing the consensus layer. Wallets may connect through user-selected gateways, avoid transmitting identifying analytics, minimize repeated address lookups and permit local verification. Governance spaces can keep sensitive attachments off-chain and commit only to their hashes, with explicit rules about who can retrieve the originals. None of these controls can retract data already copied from a public ledger. Privacy and irreversibility are connected: the permanent archive that strengthens accountability also increases the cost of careless publication.

Risk	What the public ledger can reveal	Proposed design response
Address linkage	Repeated ballots and transfers by one key	Clear privacy warnings, optional account separation
Network observation	Timing and relay endpoint metadata	Independent relays; minimize unnecessary client telemetry
Credential correlation	Reuse of membership identifiers	Scoped credentials and minimized claims
Premature ballot disclosure	Public choice before closing	Optional commit-and-reveal after formal review
Signer coercion	Forced signature or disclosed choice	Do not advertise coercion resistance without a dedicated protocol
Permanent personal data	Sensitive documents embedded in blocks	On-chain digests and controlled external storage where appropriate

The result should be a policy of precision rather than secrecy by assertion. SHB can promise that a properly formed digital signature is verifiable. It cannot promise that a signature was voluntary. SHB can record the rule under which a ballot was accepted. It cannot promise that all social consequences of disclosure have been eliminated. Each governance space should publish a meaningful privacy statement alongside its voting rules, including what data is public, what remains off-chain, what can be linked and what cannot later be deleted.

6.7 Lost Credentials and Account Recovery

A decentralized governance protocol cannot depend on a help-desk operator having undisclosed power to replace anyone's private key. Yet a system in which one lost device permanently removes a member's political voice is unlikely to serve broad communities well. The design must balance durable control against recoverability. It should make clear what an account can recover, who may authorize recovery, and which aspects of a past ballot are immutable once finalized.

The simplest case is local restoration from a correctly generated backup secret. The participant imports the seed or an encrypted key file into a compatible wallet, derives or restores the same signing key and continues using the existing account. This recovers possession of a secret, not a record maintained by the SHB team. The wallet should instruct users to store backups offline and test the recovery process before depending on the account. A screenshot placed in ordinary cloud storage is not a substitute for a secure backup plan.

A more advanced account may designate independent recovery authorities before anything goes wrong. For example, an account could require two of three previously registered guardian keys to approve replacement of a compromised voting key after a published delay. The guardians need not gain access to the balance or receive the right to vote. Their authority should be limited to proposing or certifying a key change, and every recovery request should be recorded in the ledger. Any multi-key

scheme introduces collusion and availability risks, so the policy cannot be presented as universally safer than a single carefully protected key.

Recovery is particularly delicate when a vote is already underway. If a participant loses access after a snapshot has fixed the eligible weight, the community must know whether a replacement key may use the original account's remaining voting entitlement. A defensible design is to attach the recovery authorization to the account identity rather than minting a fresh snapshot entitlement. The recovered controller may act only through the account's existing ballot state and must not duplicate a previously counted vote. The applicable constitution may also impose a contest period for sensitive recoveries.

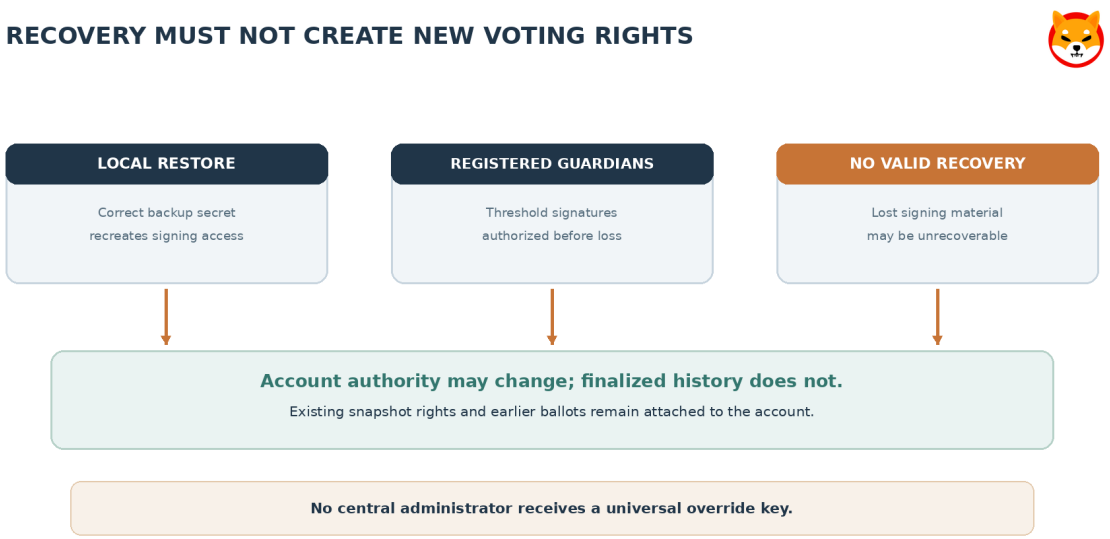
Algorithm 6.3 — Evaluate a registered account-key recovery

```
retrieve finalized account policy and original account identifier
verify guardian signatures against the registered recovery policy
verify that no guardian authorization is reused outside its intended scope
enforce the published delay, challenge window and nonce requirements
create replacement-key transition without changing token supply
preserve account balance, snapshot entitlements and prior ballot records
retire compromised signing authority at the prescribed activation height
issue a final receipt that identifies the effective new verification key
```

If no recovery mechanism was registered and all signing secrets are lost, the protocol may have no safe method to restore the account. That is not a software defect that can be repaired by giving project administrators emergency access to every wallet. An institution may compensate through its own membership procedures for future governance decisions, but it should not rewrite a finalized historical ballot or fabricate a signature for the lost key. These boundaries should be explained before users receive tokens or rely on their voting rights.

Recovery also changes the threat model of the wallet. A malicious guardian coalition, stolen recovery instructions or a social-engineering attack on the account owner may be more dangerous than an ordinary device theft. The first SHB release should therefore avoid presenting complex guardian arrangements as a default unless their implementation and user experience have been carefully tested. Basic secret backup, visible key rotation and an explicit opt-in recovery policy are more defensible milestones than silently placing participants under the control of a central recovery officer.

FIGURE 6.4 RECOVERY CHANGES SIGNING AUTHORITY WITHOUT REWRITING A BALLOT



SHB authorization and participation architecture, OT Development Team, February 2020. Proposed specification; not a completed or audited deployment.

6.8 Accessibility and Nontechnical Participation

A protocol can be cryptographically correct and socially unusable. If participation requires understanding command-line tools, operating a full validator or manually calculating a proposal identifier,

the practical electorate will be much smaller than the eligible one. That gap matters because the design measures influence with a fixed resource. A technical barrier that disproportionately excludes smaller holders may strengthen the concentration of control even when every ballot is counted accurately. The wallet and governance interface are therefore not optional marketing accessories; they are necessary parts of the participation model.

A reasonable first interface should let a participant inspect an active proposal, read the relevant rules, see eligible voting weight, choose an option, review the exact authorization and submit it without needing to understand the underlying serialization. Each significant action should disclose its current ledger status. The useful states are *prepared*, *signed*, *submitted*, *included*, *finalized*, *rejected* and *expired*. The wallet must not collapse *submitted* and *finalized* into a single green check mark. A relay may accept an envelope that the validator network never includes.

At the nominal one-minute block cadence, the network may produce one block per minute, 60 per hour and 1,440 per day. Actual finalized inclusion still depends on load, relay availability, message propagation, validator quorum and transaction validity. An interface could indicate an expected window based on recent observed behavior, but it should not guarantee a one-minute outcome. A ballot submitted close to its closing height requires especially clear feedback. The participant needs to know whether the decision record includes the vote, not whether the browser sent a request.

The system should permit independent submission. A community's preferred portal may disappear, become compromised or refuse service to an unpopular participant. Signed instructions can be delivered through another compatible relay if they remain valid. Likewise, explorers and clients should use verifiable proofs against accepted block headers instead of relying exclusively on the original operator's displayed result. Where a participant lacks the resources to verify full chain history, the software should state the narrower trust assumptions of a light-client proof.

Human accessibility deserves concrete requirements: clear typography, screen-reader-compatible controls, translated proposal summaries, keyboard access, explicit date and height information, readable error messages and protection against accidental repeated submissions. Low-bandwidth clients should be able to obtain compact proposal metadata and inclusion receipts without downloading large attachments. Proposals should include content hashes so that alternative interfaces can display the same authoritative documents. Accessibility testing with actual community members is more informative than assuming a design is understandable because its developers find it convenient.

Network abuse remains a difficult practical question. A chain producing blocks at a one-minute target cannot accept unlimited arbitrary content from anonymous clients. SHB will require bounded transaction sizes, relay admission policies, resource accounting and anti-spam measures. Those controls should not silently impose a dependence on using SHB as a retail payment currency. The protocol can study per-account limits, bounded proposal metadata, community sponsorship for expensive actions or distinct resource quotas, subject to fairness and denial-of-service analysis. The exact admission mechanism remains an open design task and should be tested before deployment.

Ultimately, participation should survive the failure of a particular application. A participant's account, voting history, valid delegation, proposal eligibility and accepted result are properties of the authenticated ledger state, not privileges lent by the operator of a website. This does not remove the need for approachable software; it sets a standard by which that software can be evaluated. An interface has served its purpose when it helps participants exercise independently verifiable rights, not when it makes them dependent on the interface for those rights to exist.

Participation Requirements

The right to participate in SHB is not defined by possession of a username. It is defined by a chain of verifiable conditions: a valid authority, a declared governance space, an effective set of rules, an authenticated record of eligible influence and a correctly formed instruction accepted within the relevant window. Every link in that chain must be visible enough to audit and narrow enough to implement consistently. Digital signatures provide a strong answer to the question of authorization; they do not

replace membership policy, privacy, human understanding or safeguards against concentrated influence.

The network is specified around a fixed ceiling of 100,000,000 SHB and a nominal one-minute block interval. Those constants describe the size of the governance resource and the intended rhythm of the ledger, respectively. They are not promises that every token represents a different human, that a vote will finalize in exactly one minute or that new tokens must be awarded for producing blocks. With that foundation in place, the next chapter can address supply accounting, initial distribution, transfer restrictions and the choices required to make token-weighted governance credible in practice.

References and Technical Foundations

[22] S. Josefsson and I. Liusvaara, “Edwards-Curve Digital Signature Algorithm (EdDSA),” RFC 8032, January 2017. https://www.rfc-editor.org/rfc/rfc8032

[23] P. Wuille and G. Maxwell, “Base32 address format for native v0-16 witness outputs,” BIP 173, assigned March 2017. https://bips.dev/173/

[24] M. Palatinus, P. Rusnak, A. Voisine and S. Bowe, “Mnemonic code for generating deterministic keys,” BIP 39, assigned September 2013. https://bips.dev/39/

[25] W3C, “Web Authentication: An API for accessing Public Key Credentials Level 1,” W3C Recommendation, March 2019. https://www.w3.org/TR/webauthn-1/

[26] W3C, “Verifiable Credentials Data Model 1.0,” W3C Recommendation, November 2019. https://www.w3.org/TR/vc-data-model-1.0/

PART II / THE NETWORK BENEATH THE VOTE

Chapter 7

SHB: Units of Influence

A finite measure of collective authority, not a promise of monetary return.



For SHB, a token is principally a voting unit. Its holder may support a proposal, oppose a rule change, delegate influence or take part in appointing a representative, provided the relevant community allows it. Nothing in those activities requires the token to act as the price of a product or service. This chapter describes how a finite supply of voting units is recorded, distributed and counted, and where that model creates difficult questions of fairness and concentration.

The design begins with two explicit network parameters. **The total SHB supply is fixed at 100,000,000 tokens**, and validators target **one block every minute**. The first parameter makes voting weight scarce and accountable. The second provides a predictable operating rhythm for recording token movements, proposals and ballots. Neither implies that validators manufacture tokens while producing blocks. We envisage no inflationary mining subsidy. The full allocation model and unlock policies set out below are a *draft for review*, not evidence that a genesis network, public allocation or distribution event has occurred.

A fixed supply does not, by itself, imply fairness. An account holding five million tokens may exercise a thousand times the voting weight of one holding five thousand. Such a rule is simple to verify, but it gives the distribution of tokens constitutional importance. The protocol must disclose who controls the initial supply, which units may vote, how authority may be delegated, and what happens when a large holder seeks to dominate a decision. It must also avoid a second mistake: treating a token ledger as a substitute for the social consent that gives a constitution legitimacy.

This chapter sets out the supply invariant, an illustrative genesis allocation, transfer and vesting semantics, ballot snapshots, delegation accounting, concentration safeguards and possible resource policies. Values marked **fixed** are SHB design commitments. Values marked **proposed** are initial engineering choices open to review before genesis. Their different status is a deliberate part of the specification.

7.1 The Native SHB Token

SHB is the native accounting unit of a dedicated governance blockchain. Its primary function is to express voting weight under the rules of a governance space. It is not merely an external token tracked by a website: token balances and governance instructions belong to the authenticated state of the same network, allowing a validating node to reproduce eligibility and ballot outcomes without querying a private administrator's database. The ledger has to keep token operations and governance operations distinguishable, even though both may appear in the same block.

The protocol shall enforce a fixed nominal supply of **100,000,000 SHB**. As an initial engineering proposal, all 100 million tokens are assigned in the genesis state to explicitly identified accounts and governance vaults. No subsequent operation is permitted to mint additional SHB, and ordinary consensus production does not create rewards. Tokens held in a vault have not vanished; they remain recorded balances subject to published conditions. For consistency with an exactly fixed outstanding supply, the proposed first release also omits a destructive burn operation. Permanent retirement, if ever introduced, would require a separate constitutional discussion and would not be a way to increase the maximum supply.

```
S(h) = sum_{a in Accounts} B_h(a) = 100,000,000 SHB, for every finalized height h >= 0
```

Here B_h(a) is the authenticated balance of account a after finalized block h. Locked and escrowed balances are included in the sum; a locked token is not a missing token. A transfer changes two balances but not their sum. A vesting operation changes the availability of an existing balance but not the amount of SHB outstanding. Validators must reject any transaction or state transition that makes a balance negative, overflows the specified integer representation, or violates the supply invariant.

The decimal precision requires a consensus-wide decision before the genesis file is signed. Six decimal places are proposed as a practical compromise between wallet usability and precise proportional voting. Under that proposal, one SHB equals 1,000,000 base units, and the genesis allocation contains exactly 100,000,000,000,000 base units. Consensus code counts integers, never floating-point numbers. The display name “SHB” remains the human-facing denomination; fractions are simply smaller portions of the same voting unit.

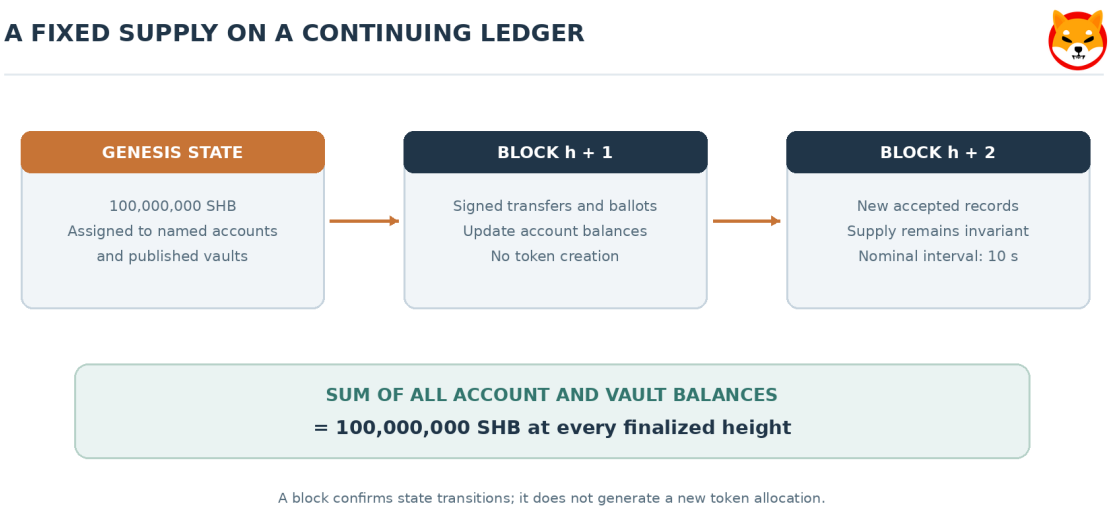
```
U_total = 100,000,000 * 10^d; proposed d = 6; therefore U_total = 100,000,000,000,000
base units
```

Fractional balances may express fractional voting weight. A wallet with 1.250000 SHB has 1.25 standard voting units, not two votes obtained by rounding. The rules must define whether a ballot accepts such precision directly or uses the integer base-unit total internally and formats it afterward. The latter approach is preferred. Software that silently rounds token weights for the sake of an attractive interface could change a close result.

Attribute	Protocol position	Status
Native asset	SHB / ShibaToken	Fixed identity
Total supply	Exactly 100,000,000 SHB	Fixed
Nominal block interval	60 seconds	Fixed target, not a finality promise
Additional block issuance	None	Proposed consensus rule consistent with fixed supply
Base-unit precision	Six decimal places	Proposed before genesis
Fundamental voting weight	One eligible SHB = one voting unit	Fixed baseline
Primary purpose	Governance and coordination	Fixed design objective
Transfer policy	Signed, ledger-validated movements, subject to explicit locks	Proposed implementation

A native token cannot guarantee an equal voice for every individual. An account is not a person, and a token balance does not prove identity. SHB therefore describes its default mechanism accurately as **token-weighted governance**. Other communities may use identity qualifications, committee structures or advisory polls, but those mechanisms must not be mistaken for properties arising automatically from ownership.

FIGURE 7.1 FIXED SUPPLY AND THE LEDGER STATE TRANSITION



The supply invariant applies to every account, including locked and protocol-controlled balances.

7.2 Token Supply and Distribution Principles

The genesis allocation is more than an economic announcement. It determines the earliest potential electorate, the balance between public participation and protocol stewardship, and the extent of influence that can remain with the founding organization. The safest way to discuss it before launch is to expose the assumptions and their consequences. A table of percentages cannot demonstrate broad distribution; it can only describe the accounts from which distribution is intended to occur.

We propose five disjoint genesis allocations totaling exactly one hundred million SHB. **The following percentages are an illustrative working allocation, not a ratified decision by the project or a representation of token sales or distribution that have already taken place.** The final genesis manifest must contain actual addresses, exact integer balances, vesting conditions and the signatures authorizing those entries. It should be published before any production validator accepts the genesis block.

Allocation	Tokens	Share	Proposed restriction and purpose
Community participation	50,000,000 SHB	50%	Publicly documented eligibility, staged distribution, initial broad-access tranche
Governance ecosystem	20,000,000 SHB	20%	Community tooling, educational material, open-source integrations, governed grants
Core protocol development	15,000,000 SHB	15%	Transparent founding allocation; delayed and gradual vesting
Validator and network operations	10,000,000 SHB	10%	Finite service grants; no newly minted block rewards
Security and contingency reserve	5,000,000 SHB	5%	Audits, emergency network maintenance and publicly approved releases
Total	**100,000,000 SHB**	**100%**	**Fully accounted for at genesis**

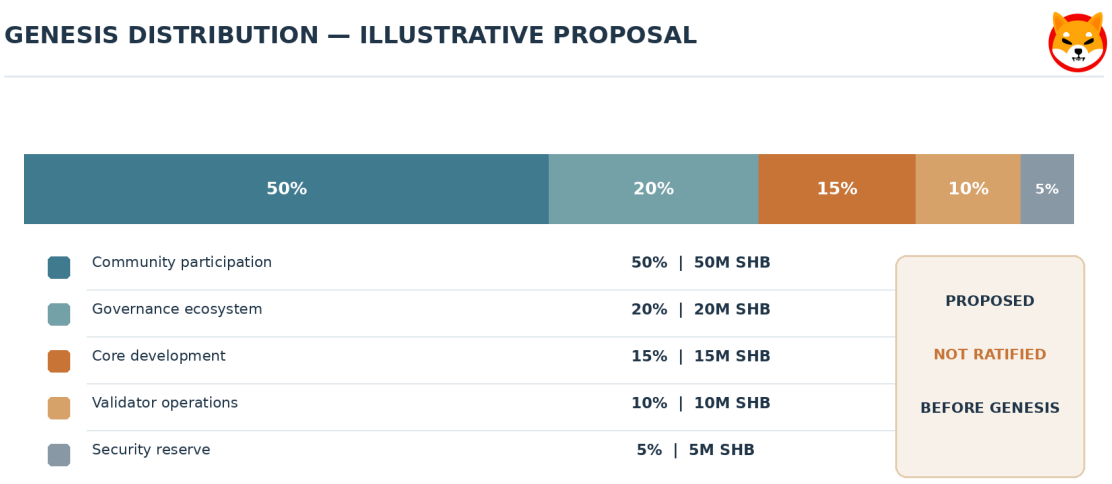
This table makes a useful distinction between *assigned*, *unlocked*, *distributed* and *voting-eligible*. A token may be assigned to the community participation allocation without yet belonging to an individual participant. A founding account may have a visible balance but be unable to transfer or vote with it before the applicable vesting condition is met. A security reserve may be accounted for on-chain without being an active participant in everyday community decisions. None of these categories creates or destroys a token; they describe the status of existing units.

A possible release policy would allow up to ten million SHB from the community allocation to be available for an initial, widely accessible participation phase, with the remaining forty million distributed through published procedures over a longer period. The ecosystem allocation would be released against independently reviewable contributions rather than a single discretionary transfer. The core-development allocation could carry a twelve-month initial lock followed by thirty-six months of gradual vesting. Network-operation support could be budgeted across sixty months, while the security reserve would remain locked pending a qualified governance decision. These durations are **planning examples**, not consensus constants or a promise of distribution to any specific party.

We deliberately do not prescribe a public-sale price, exchange listing, projected market capitalization or investment return. Those parameters are unnecessary to explain why a token-holder may participate in a vote. If SHB becomes transferable between independent parties, a secondary market could emerge; the protocol cannot guarantee that it will not. But the whitepaper should not confuse a possible market for voting units with the purpose for which those units were designed.

The proposed community allocation is substantial, yet percentages alone do not prevent concentration. A single participant can acquire tokens from many recipients; a coordinated group can use multiple addresses; and grants may favor well-connected applicants unless their selection process is visible. Accordingly, publication should include the address-level genesis manifest, the beneficiaries and terms of founding grants where disclosure is lawful, the approval rules for vaults, and a periodic report on holdings still under project-controlled custody. Claims of decentralization should be tested against actual distributions rather than inferred from the number of wallet addresses.

FIGURE 7.2 PROPOSED FIVE-PART GENESIS ALLOCATION; SUBJECT TO RATIFICATION



Illustrative genesis allocation; the percentages remain subject to review before network launch.

7.3 Balance-Based Voting Power

The default rule is intentionally direct: **one eligible SHB equals one unit of voting weight**. In an ordinary token-weighted ballot, ten SHB carry ten times the influence of one SHB. The relation is linear, so an independent validator can reproduce the result by reading authenticated token balances at a defined state height and applying the published eligibility predicate. No online polling provider must be trusted to report the weights.

A proposal p identifies its governance space g, an eligibility reference height r(p), a permitted set of accounts, a ballot method and any published exclusions. Let B_r(a) denote an account's token balance at that snapshot. Let E_g,p(a) be 1 if account a is eligible and 0 otherwise. Let L_g,p(a) represent the portion of the account balance that is ineligible due to explicit voting restrictions, such as an unvested genesis lock. The initial entitlement is then defined in integer base units:

```
W_0(a,p) = E_g,p(a) * max(0, B_r(a) - L_g,p(a));  0 <= sum_a W_0(a,p) <= U_total
```

No account can contribute more than its authenticated eligible balance to a single ballot. The inequality applies to **each proposal separately**. It does not imply that tokens become consumed merely by participating in one governance space; the same token may express a preference in two independent communities when each constitution permits it. Those decisions are separate ballots with separate records. What must be prevented is counting the same unit twice **inside the same ballot**, including through delegation.

The snapshot is chosen according to a rule published before the ballot opens. For example, the constitution may use the last finalized block before a scheduled voting window. A validator can verify that r(p) has been finalized, that the selected header matches the accepted chain, and that an authenticated state proof or replayed state gives the claimed balances. The reference height is a precise ledger coordinate; it is not simply “about one minute before voting.” A network targeting one-minute blocks can still encounter congestion, leader changes and pauses.

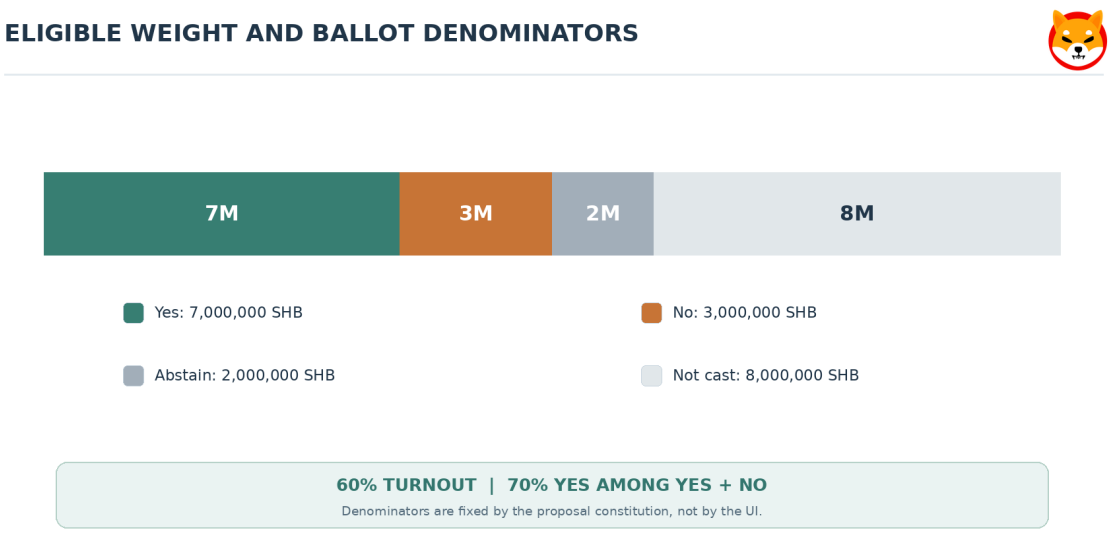
Consider a hypothetical community with twenty million eligible SHB at snapshot. During a vote, seven million units support the proposal, three million oppose it, two million record abstention and eight million do not participate. If the constitution defines turnout as yes + no + abstain, turnout is twelve million units, or 60% of eligible weight. If it defines approval over only yes and no, the approval ratio is 7 / (7 + 3), or 70%. If passage requires at least 50% turnout and 60% approval, the proposal passes. An alternative constitution could count abstentions in the approval denominator and reach a different result. That is a constitutional choice, not an arithmetic error.

```
turnout(p) = (Y_p + N_p + A_p) / W_eligible(p);  approval(p) = Y_p / (Y_p + N_p), when Y_p + N_p > 0
```

Voting state	SHB weight	Treatment under this example
Yes	7,000,000	Counted in turnout and approval numerator
No	3,000,000	Counted in turnout and approval denominator
Abstain	2,000,000	Counted in turnout, excluded from approval ratio
Not participating	8,000,000	Included only in eligible denominator
Eligible total	20,000,000	Reference denominator fixed at snapshot

This example is deliberately separate from the network's one hundred million supply. The governance space is entitled to a subset of units; treasury vaults, other communities' holdings and accounts outside the specified membership set may be excluded under its rules. Reporting a proposal's 70% approval without also reporting turnout and the electorate would give an incomplete picture of the decision.

FIGURE 7.3 A FIXED ELECTORATE AND THE MEANING OF TURNOUT



Only eligible balances at the reference height contribute to the ballot denominator.

7.4 Voting Snapshots and the Reference State

A governance token can move between accounts. If a vote were measured from live balances at the moment each ballot was submitted, a participant could vote, transfer the same units to an associate, and have them voted a second time. The ledger might faithfully record both transactions while the ballot counted more influence than the community had authorized. This is not an exotic cryptographic failure; it is a faulty definition of eligibility. An explicit, immutable reference state solves the basic accounting problem.

Every voting proposal should therefore commit to an authenticated **reference height** before voting begins. The reference height fixes the candidate balances, membership permissions and voting restrictions for the lifetime of that proposal. Later transfers affect later proposals but do not rewrite the reference state of a ballot already open. The voting engine maintains an entitlement record indexed by (governance_space, proposal_id, source_account). Once the source account has exercised its entitlement, a replay or second instruction cannot add more weight to the tally.

Algorithm 7.1 — Construct immutable proposal voting entitlements

```
require finalized reference block H_ref and authenticated state_root(H_ref)
require proposal declares governance_space, ballot_rule, start and close bounds
for each eligible source account a at H_ref:
    balance_units = authenticated_balance(a, H_ref)
    excluded_units = restricted_units(a, proposal, H_ref)
    entitlement[a] = max(0, balance_units - excluded_units)
commit entitlement root and total_eligible_units to the proposal record
reject any ballot whose claimed source weight exceeds unused entitlement
reject a second effective ballot for the same source unless recast is explicitly enabled
never recalculate entitlement from a later wallet balance
```

A production implementation need not enumerate every possible account in a separate giant snapshot table. It can derive balances from the historical state root and create per-account claims on demand, provided it proves that the aggregate eligible denominator and exclusion rules are correct. The design must make that choice explicit. Storing a large entitlement vector may improve simplicity but increase state requirements; reading a historical Merkleized balance map can reduce duplication but requires reliable retained state or proofs. In either case, two honest full nodes must obtain the same entitlement for the same proposal and source account.

The time at which eligibility is captured must be resistant to last-minute manipulation. A proposal creator should not be permitted to select an arbitrary old height after seeing which balances would favor an outcome. A community constitution could require a deterministic reference height tied to publication and a minimum notice period. It may also require a waiting interval before a recently transferred token

becomes eligible in sensitive elections. Such a rule reduces opportunistic transfers but also disadvantages legitimate new members; it should be disclosed as a policy trade-off.

A one-minute target block schedule gives a useful planning unit. At uninterrupted target cadence, one block corresponds to roughly one minute, 1,440 to one day, and approximately 525,600 to a 365-day year. These are **nominal estimates**, not guaranteed calendar durations. A governance rule that intends an actual fourteen-day review period should not silently convert that period to 20,160 blocks and assume network timing cannot drift. Where human deadlines matter, SHB should define a bounded consensus timestamp rule and clarify what happens when the chain pauses, rather than claiming that block height alone measures civil time perfectly.

```
estimated_blocks(T_seconds) = T_seconds / 60; estimated_blocks(1 day) = 1,440;
estimated_blocks(365 days) = 525,600
```

A snapshot's immutability also serves accountability. A reader inspecting a controversial result months later should be able to recover the same reference height, the same eligibility rule, the same delegation mapping and the same decisions. The reference state is not an internal cache to be discarded when convenient; it is part of the evidence explaining why the outcome was accepted.

7.5 Delegated Voting Balances

A holder need not follow every proposal personally. SHB anticipates a model in which an account may authorize another account to exercise some or all of its eligible voting weight in a specified governance space. Such delegation transfers **the authority to cast a ballot**, not title to the underlying SHB. The original balance remains in the delegator's account and does not appear again in the delegate's spendable token balance. The delegate's voting interface must distinguish its own entitlements from those exercised on behalf of others.

A delegation record requires at least the principal account, delegate account, governance-space identifier, permitted scope, authorized amount, activation condition, expiration or revocation rule and signed authorization. A constitutional rule may allow different delegates for technology policy and community membership decisions. It may also forbid chains of delegation or allow a bounded depth. The initial implementation should favor direct delegation with no recursive forwarding: it is easier to audit and cannot create unresolved cycles. Multi-hop delegation can be specified later with a deterministic cycle-handling rule.

For a proposal p, define D(a,b,p) as the number of voting base units that source account a delegates to b. The amounts delegated from a must not exceed the source's eligible entitlement. A valid tally may aggregate voting instructions from many source accounts at one delegate address, but the source ownership of each weight must remain recoverable from the authenticated record.

```
for each source a:  sum_b D(a,b,p) <= W_0(a,p);  sum_a W_effective(a,p) = sum_a W_0(a,p)
```

The conservation equation is true only when every source unit is assigned to at most one effective decision path and uncast weight remains accounted for as unexercised. Delegation must not create extra tokens or extra entitlements. If a holder delegates 600 SHB out of a 1,000 SHB eligible balance, the holder retains authority over 400 SHB for that proposal. The delegate may cast at most 600 on that holder's behalf. Neither may cast the full 1,000 independently.

Revocation should be easy but not ambiguous. The baseline proposal is that a revocation finalized before a ballot's reference height changes the mapping used in that ballot. A later revocation takes effect for future ballots, unless the current ballot's published rules explicitly support a safe mid-election override or recast. If late changes are permitted, the implementation must ensure that a previously recorded delegated vote is removed or replaced atomically rather than counted alongside a direct vote. That complexity should be acknowledged, not hidden behind the word “liquid”.

Algorithm 7.2 — Validate and apply a delegated ballot

```
load proposal snapshot and its finalized delegation map
verify delegate signature, chain identifier, ballot scope and nonce
retrieve each source entitlement assigned to the delegate for this proposal
exclude source units already exercised under the published ballot rule
reject if any source authorization is expired, invalid or oversubscribed
apply chosen option to each included source's unused voting units exactly once
record delegate signer, source attribution and aggregate cast weight
assert total counted source units cannot exceed total eligible proposal units
```

Delegation has an important social consequence. A holder may prefer an accountable specialist to a personal decision on an unfamiliar technical topic. A well-designed system can make that preference visible, reversible and limited in scope. It cannot guarantee that the specialist is honest, that an address corresponds to one individual or that a representative has not made private agreements. Those are governance risks requiring transparent records, available alternatives and effective recall procedures.

7.6 Transfers, Vesting and Governance Commitments

Transfers preserve supply by debiting one account and crediting another in a single deterministic state transition. A signed transfer specifies the sending authority, destination, amount in base units, account nonce and validity limits. Validators reject transfers larger than the available balance and must apply token locks before checking spendability. The destination account should be correctly encoded for the same network; wallet software should not rely on visual similarity between addresses.

```
B'_(sender) = B_sender - x; B'_(recipient) = B_recipient + x; sum_a B'_a = sum_a B_a
```

Vesting is not a second issuance schedule. In the proposed genesis arrangement, the vault already contains its assigned tokens. A vesting rule changes how much the vault may release to an authorized recipient. A linear schedule can be defined by a start time t0 and end time t1, with an optional cliff. For consistency across nodes, all calculations use integer base units and a published consensus interpretation of time. The vesting function should ensure that rounding residues are made available at the final unlock event and that no amount can exceed the vault's original allocation.

```
unlocked_v(t) = floor(A_v * clamp((t - t0)/(t1 - t0), 0, 1)); before a defined cliff,
unlocked_v(t) = 0
```

The preceding expression describes a prospective schedule, not a commitment that the SHB network will run on a particular commercial timetable. The one-minute block target should inform operational estimates but should not be used as proof that a vesting period has elapsed in real-world time. A clock-aware consensus rule with bounded validation, or a height-based policy explicitly labeled as approximate, must be specified before a vesting condition can be enforced safely.

A locked amount can have different voting implications. We propose that the **unvested genesis allocations held by development, validator-support and contingency vaults are non-voting in ordinary community ballots**. This prevents administrative reserves from quietly overwhelming an early community whose freely distributed balance remains small. After release, recipients may become eligible according to the relevant community's membership and reference-height rules. A community may create a separate ballot specifically authorizing a treasury release, in which voting rights belong to qualified members rather than to the vault whose funds are at issue.

Balance or restriction	Ownership treatment	Baseline voting treatment
Ordinary unlocked SHB	Controlled by account signing authority	Eligible if governance-space rules permit
Unvested founding allocation	Held by genesis vault with published schedule	Excluded from ordinary governance ballots
Conditional ecosystem grant	Vault balance pending verified release	Excluded until transferred and eligible
Voluntary participation lock	Retained ownership; transfers temporarily restricted	May remain eligible if ballot rules expressly allow

Balance or restriction	Ownership treatment	Baseline voting treatment
Delegated balance	Never transferred to the delegate	Source entitlement exercised once through delegate
Security reserve	Vault-controlled under published approvals	Non-voting in ordinary ballots until release

It matters because a promise of “community control” can be undermined by hidden reserve votes. A transparent ledger should permit observers to inspect the token balance of every vault and to distinguish the voting treatment applied to its units. The implementation must not assume that the public can infer these policies from address labels or marketing materials; they must be part of the consensus-verifiable eligibility rules.

7.7 Concentration of Influence and Its Consequences

Token-weighted governance makes the distribution of holdings a political fact. It avoids the need to rely on a central ballot counter, but it does not prevent a wealthy or coordinated participant from accumulating influence. The risk is not confined to one holder owning a majority of all SHB. A holder may control a majority of the *eligible* supply in a small governance space, or a much smaller percentage may suffice when turnout is low. Delegated authority can create further concentration even when underlying ownership remains dispersed.

One measure is the share of total supply controlled by the largest observed addresses. Another is the share of eligible voting units controlled by the largest identifiable beneficiary or delegate. Address-level numbers are an incomplete proxy for human or institutional control: one person may use several addresses, and one address may serve a collective custody arrangement. Reports should describe their data source and uncertainty rather than promise to identify unseen beneficial owners. A Gini coefficient or a Herfindahl-style concentration index may assist comparison across time, but such indicators cannot themselves demonstrate that governance is legitimate.

```
concentration_H(p) = sum_{i=1..n} (W_i(p) / W_eligible(p))^2;  use only with a declared
grouping method
```

The default linear voting rule should not be silently replaced with an arbitrary account cap. If an address with ten thousand SHB may vote with only one thousand units, the holder can distribute tokens across ten addresses and restore the original weight unless the protocol can reliably identify linked accounts. The same weakness affects quadratic voting when its anti-concentration promise relies on one person possessing exactly one account. Such variants may be investigated for communities with suitable identity systems, but they are not robust universal defaults for a pseudonymous public ledger.

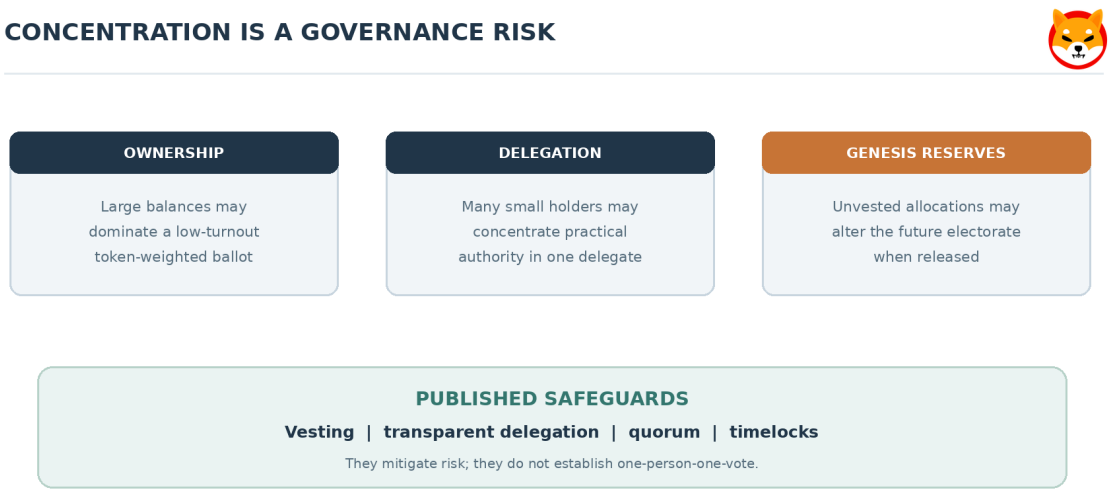
Practical safeguards therefore begin with institutional design. Vesting prevents founding allocations from entering the electorate all at once. Long public review periods make sudden changes harder to conceal. High-impact amendments may require stronger quorum and supermajority conditions. Timelocks allow participants to review implementation plans before they become effective. Delegation dashboards can show when too much authority accumulates with a small number of representatives. Independent observers should be able to reproduce these figures from the chain.

None of these safeguards is free. A very high quorum may allow a minority to block necessary maintenance by withholding participation. A long timelock may frustrate emergency incident response. A broad identity requirement may exclude participants who cannot obtain acceptable credentials. SHB should therefore support clearly declared constitutional choices, with different thresholds for ordinary measures, protocol changes, and emergency interventions. Those controls affect whether a proposal may pass; they do not create a misleading claim that 100 million voting units have become one hundred million distinct voters.

The founding team has a special responsibility here. A 15% proposed allocation for core development would be substantial even if initially locked. The release schedule, beneficiary disclosures, voting restrictions and amendment thresholds should be determined and published before those tokens

become active. If the initial distribution fails to create a broad electorate, the whitepaper must not call the result decentralized merely because validators are operated on separate machines.

FIGURE 7.4 OWNERSHIP, DELEGATION AND VAULT CONCENTRATION RISKS



Safeguards reduce the risk of concentrated influence; they do not eliminate it.

7.8 Token Economics Without Dependence on Payment Adoption

A governance network incurs real costs. Validators need equipment, connectivity and operational expertise. Developers must maintain client software and review security defects. Community participants need interfaces they can trust, and auditors need access to specifications and state proofs. It would be misleading to suggest that an absence of payment functionality makes these costs disappear. SHB's design question is whether the costs can be supported without making commercial transfer volume the central reason the network exists.

The proposed ten-million-SHB validator and network operations allocation is a **finite genesis reserve**, not an ongoing block subsidy. A validator supporting the network may receive tokens only through an authorized distribution from that reserve, subject to measurable service requirements and transparent approval. No validator obtains newly created SHB merely because another one-minute period has passed. A reward policy based on signed performance records could be examined, but must avoid allowing validators to certify their own compensation without independent review. Running a node should remain possible for parties whose primary motive is the health of a community they serve.

```
issuance_per_block = 0 SHB;  planned_blocks_per_day = 86,400 / 60 = 1,440;
annual_nominal_blocks = 31,536,000 / 60 = 525,600
```

Network operations also require protection against spam. An entirely unmetered ledger could be overwhelmed by repeated trivial transactions, including proposals that never attract meaningful participation. The first protocol release should therefore specify message-size limits, minimum transaction validity checks, local peer admission rules, per-account proposal restrictions and bounded resources for governance operations. Requiring a refundable SHB deposit for certain proposal types is an option: the units would be locked while a proposal is processed and returned or handled under transparent rules, without being spent as a purchase price. Deposits create access barriers, so a responsible constitution must provide a path for legitimate smaller participants and publish how any forfeiture would be governed.

A second option is modest computational proof-of-work for selected untrusted network messages, a concept available well before this paper's date. Such a measure could raise the cost of flooding relay nodes but would disadvantage low-powered devices and would not replace ordinary validation or peer resource controls. Neither proposal is presented here as a solved, audited anti-spam mechanism. The

initial engineering effort should favor predictable, measurable bounds and a clear procedure for revising them without arbitrary censorship.

Token circulation, validator funding and governance authority should appear as distinct lines in public reports. A transfer between participants changes who may influence later decisions but does not create revenue for the protocol. A release from a genesis reserve makes previously locked units available but does not expand total supply. A treasury grant may finance a useful application but must not be counted as evidence of increasing voting turnout. The distinction between accounting balances and effective participation is essential to judging whether the network is fulfilling its purpose.

For example, a healthy early SHB community might be characterized by a high proportion of eligible units participating in significant votes, a diversified set of independent delegates, a clear record of proposal implementation and a small number of unresolved disputes. None of these measures requires the token to become a widely used retail payment instrument. Conversely, substantial external trading activity could coexist with an inactive electorate and would not demonstrate that the SHB governance model works.

A useful operational report would therefore publish, for each relevant period: total outstanding SHB; genesis reserves still locked; freely transferable balances; active voting-eligible units; proportion represented by the largest reported accounts and delegates; number of governance proposals; verified turnout; validator availability; and ledger finality performance. A one-minute nominal block target is an engineering goal to measure against observed time-to-finality, not a substitute for that measurement. Reporting should distinguish average block interval from tail latencies during network disturbances.

Metric	Proposed report	Reason it matters
Fixed outstanding supply	Exactly 100,000,000 SHB	Detects unauthorized creation or destruction
Genesis reserve balance	Per category, with unlock conditions	Shows pending concentration of influence
Circulating and eligible balances	Separately, by declared definition	Avoids misleading turnout denominators
Delegate concentration	Weighted by source entitlement	Exposes representative capture risks
Community turnout	Eligible-weight denominator per ballot	Measures practical participation
Validator availability	Signed uptime and incident evidence	Tests ledger reliability
Finalization latency	Median and high-percentile observations	Tests the one-minute operating ambition

The chapter's central conclusion is that a finite supply and a fast ledger are only useful when linked to an intelligible governance process. One hundred million SHB create a bounded measure of potential influence. One-minute blocks provide a proposed cadence for preserving instructions. The protocol's legitimacy, however, must come from understandable rules, credible participation and results that can be independently verified. The next chapter will describe how an idea enters that process as a formal proposal, how its terms become binding on the ballot system, and how the community can examine what it is being asked to decide.

References and Technical Foundations

[27] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” October 2008. https://bitcoin.org/bitcoin.pdf

[28] F. Vogelsteller and V. Buterin, “ERC-20 Token Standard,” created November 19, 2015. https://eips.ethereum.org/EIPS/eip-20

[29] E. Buchman, J. Kwon and Z. Milosevic, “The Latest Gossip on BFT Consensus,” arXiv:1807.04938, July 2018. https://arxiv.org/abs/1807.04938

[30] C. Dwork and M. Naor, “Pricing via Processing or Combatting Junk Mail,” Advances in Cryptology — CRYPTO ’92, 1992.

PART III / THE ART OF DECIDING

Chapter 8

The Birth of a Proposal

A verifiable question, an accountable process and a durable record.



A proposal often starts in an untidy way. Someone raises a problem, several drafts circulate, and other members suggest changes before anyone agrees on the exact question. Across time zones, it becomes surprisingly easy to vote on different versions of what was supposed to be the same idea. An SHB proposal must therefore have a clear identity: who submitted it, which text is authoritative, when the voting period begins and what, precisely, an approval would authorize.

SHB treats these questions as part of the protocol rather than leaving them entirely to a website administrator. A proposal is not simply a page containing a question. It is a typed, signed and versioned governance object, bound to a community's constitutional rules and followed through a defined series of states. Its human-readable rationale may live outside the chain, but its identity, permitted actions, eligibility conditions, decisive deadlines and final outcome must be verifiable against the shared ledger. The task of the protocol is not to decide whether an idea is wise. It is to establish which idea was actually submitted and which rules applied when people considered it.

The network parameters remain unchanged: SHB has a fixed total of **100,000,000 tokens** and a **one-minute nominal block interval**. The creation of a proposal does not issue tokens, and every governance operation remains subject to the same authenticated ledger as ordinary SHB balances. A one-minute target gives the community a useful scheduling unit, but it is not a guarantee that each block will arrive exactly on time. In this chapter, critical transitions are therefore expressed primarily in finalized block heights, with calendar-time descriptions presented as estimates for participants.

This chapter sets out a proposed life cycle for ideas entering the SHB governance engine. It covers the proposal object, authorship, decision scope, sponsorship, amendments, evidence, scheduling, and the transition from accepted ballot to accountable action. The structures and algorithms below are a design for future implementation and review. They do not describe running production software in February 2020.

8.1 Proposals as Structured Blockchain Objects

A proposal must be interpretable independently of the interface through which it was created. If one website describes the ballot as a constitutional amendment while another presents it as an ordinary policy suggestion, the network cannot rely on either description. Validators need a canonical object with enough information to determine the applicable rule set. At the same time, storing lengthy essays, drawings or debate transcripts in every block would place an unreasonable burden on the machines maintaining the ledger. The proposed solution separates **consensus-critical metadata** from **referenced supporting material**.

The on-chain record identifies the governance space, the author, the proposal type, the constitutional version, the actions being requested, the amendment lineage, the governing thresholds and the applicable sequence of heights. Long-form explanations and attachments are addressed by cryptographic digest. A validator checks a digest against bytes supplied when those bytes are available; it does not infer the contents of an inaccessible document. The protocol should use canonical serialization for all signed fields, with version numbers for the encoding and each operation type. Human-readable text should never be allowed to redefine a machine-readable action after the author has signed it.

A practical distinction is required between an **intent** and an **action**. An intent proposal expresses a direction: for example, whether a software community should prioritize multilingual documentation. Its outcome may create a mandate for a working group without instructing the chain to change its own state. An executable proposal identifies a specific, permitted state transition, such as appointing a representative for a fixed term or changing an explicitly mutable parameter inside a governance space. These forms should not share a vague execution flag. They need distinct action schemas, review procedures and limits of authority.

Field	Purpose	Candidate validation requirement
`space_id`	Identifies the community and its electorate	Existing, active governance space
`proposal_nonce`	Distinguishes submissions by one author	Strictly increasing per signing account
`author_key`	Identifies the signing authority	Valid account signature and permitted role
`kind` / `scope`	Classifies the requested decision	Allowed by the space constitution
`constitution_hash`	Freezes the applicable rule version	References an authenticated constitution
`payload_hash`	Commits to explanation and attachments	Fixed-length digest and declared encoding
`action_set`	States the proposed on-chain effects	Canonically encoded, type checked, bounded
`parent_id` / `revision`	Records proposal ancestry	Valid parent, monotonic revision sequence
`heights`	Defines review, voting and execution boundaries	Ordered, constitutional, nonexpired
`sponsor_policy`	Defines required support before ballot	Policy exists in referenced constitution

The initial proposal identifier can be defined without a circular dependency on the transaction hash. The author signs a canonical *submission body* including the chain identifier, space, nonce, rule reference and proposed content hash. Its identifier is the hash of that immutable body under a proposal-specific domain separator. A network transaction then carries the body and signature. The resulting transaction identifier is separate: the same proposal object may be relayed by many peers without becoming multiple proposals. A rejected transaction has no authoritative proposal record merely because somebody computed a plausible-looking identifier locally.

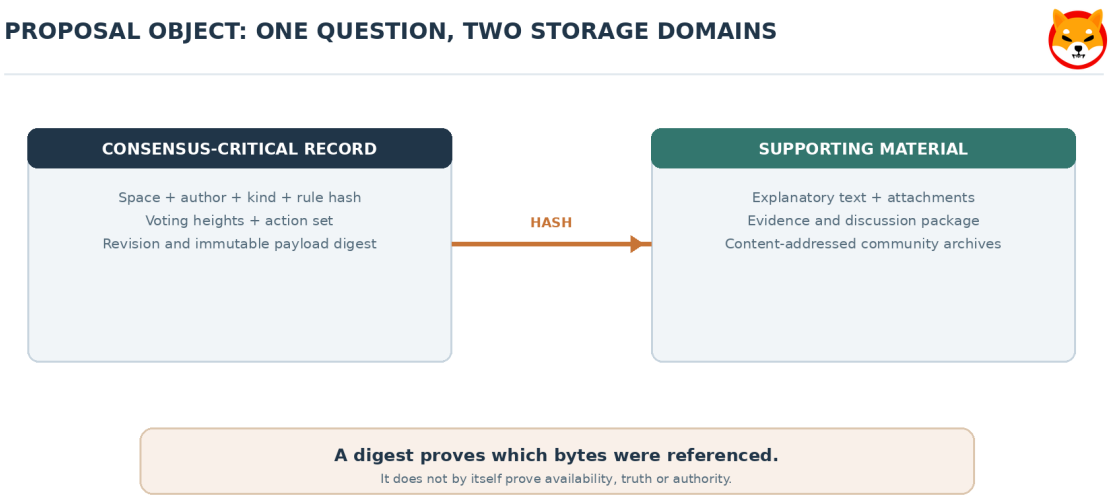
```
proposal_id = SHA256("SHB-PROPOSAL-v1" || chain_id || CanonicalEncode(submission_body))
payload_hash = SHA256("SHB-PAYLOAD-v1" || canonical_payload_bytes)
```

A future node must be able to reconstruct the status of a proposal by replaying authenticated transitions. It should not depend upon a database column called approved stored by a particular block explorer. The canonical status is derived from accepted operations and rule conditions: **Draft**, **Submitted**, **Under Review**, **Sponsored**, **Scheduled**, **Voting**, **Tallied**, **Rejected**, **Approved**, **Execution Pending**, **Executed**, **Expired**, or **Superseded**, as permitted by the selected proposal type. Some of these states are derived rather than requiring a separate transaction. A draft held on a participant's computer, for example, has no network authority until a valid submission is committed.

The same separation applies to comments and supporting documents. Off-chain availability is useful but not guaranteed by a hash alone. An archival node or independent community mirror may preserve files addressed through an IPFS-style content reference, a method discussed in research published in 2014. The on-chain record should include a digest, content type, byte length and perhaps a list of intended mirrors. If the source disappears, the network can still prove that a particular digest was

adopted, but it cannot produce the missing content from the digest. This is a limitation to disclose rather than disguise.

FIGURE 8.1 CONSENSUS-BOUND PROPOSAL METADATA AND REFERENCED SUPPORTING CONTENT



A proposal is checked against its community rules before it enters the voting process.

8.2 Proposal Authorship and the Responsibility of the Signer

The proposer has an identifiable cryptographic role, not an automatic claim to moral authority. The submitting signature proves that a key approved the exact bytes placed before the network. It does not prove that the signer is a unique person, holds a professional qualification, or has lawful authority to bind an organization outside SHB. Those additional claims require the membership and credential rules introduced in Chapter 6. A community may permit any eligible SHB holder to propose, or may reserve certain categories for an elected committee. Both are constitutional arrangements that must be inspectable before a submission is accepted.

For the ordinary public-governance case, the initial authorization check can be deliberately narrow: the space exists; the author's signing key is valid; the account nonce is fresh; the author satisfies the proposal-creation rule; the payload and action types are allowed; and the proposal is within the declared resource and scheduling limits. The check must run against a finalized state reference. If an account transfers tokens during the propagation delay between submitting and finalization, the constitution must specify whether proposal eligibility depends on the last finalized state before inclusion or on a separately declared eligibility snapshot. The first implementation should choose one rule and publish test vectors rather than leave the matter to individual validators.

A proposal may identify co-authors, editors and supporters. These are useful social distinctions but must not create ambiguous authorization. One principal submitting key should remain responsible for the canonical object; alternative multi-signature authorship could later require a precisely ordered set of signatures over the same digest. An off-chain author list is informative only unless the named contributors attest to it. Responsibility cannot be proven by an interface simply printing a person's name next to an account.

The author should be able to withdraw an uncommitted draft freely. Once a proposal has become an accepted governance object, withdrawal is a different operation: it should leave an auditable record and may require conditions imposed by the constitution. An author who has collected enough sponsorship to trigger a collective ballot should not necessarily have the unilateral right to erase the question. A sensible default allows withdrawal during review, preserves the submission and sponsorship history, and prevents deletion after the ballot is frozen. The proposal may then be cancelled only by a separately authorized process.

The protocol should not encourage flooding by pretending that every signed idea deserves permanent storage at no cost. A constitution may limit the number of open proposals per account, require a minimum account age measured in finalized blocks, impose a review period or request a refundable security bond in existing SHB. Such a bond is a temporary transfer into a governed vault, not a payment to buy democratic standing and not newly minted supply. In the default treatment proposed here, bonded tokens are unavailable as ballot weight while they remain escrowed, unless the space's rules explicitly define an alternative without double counting.

```
active_submissions(a,g,h) <= submission_limit(g, constitution_version_h)
```

A bond should not be the only path to authorship. Smaller participants need a collective sponsorship or delegated-submission route, and adverse handling of a bond must be reviewable. A single platform operator must not be able to confiscate an account's SHB merely by labeling a suggestion unpopular. Resource governance is legitimate when bounded by known rules; it becomes censorship when discretionary and unaccountable.

8.3 Categories, Scopes and Decision Authority

A governance space should distinguish proposals according to the authority they ask to exercise. **Advisory proposals** record a preference without commanding action. **Operational proposals** direct a working group or revise a project schedule. **Appointment proposals** assign a mandate to a representative. **Resource proposals** may authorize distribution from a community-controlled vault, subject to the separate accounting rules defined in the relevant constitution. **Constitutional proposals** modify who may propose, vote, delegate or amend a rule. **Protocol proposals** concern network-wide software and validator parameters and must not be confused with the internal policies of one community.

This classification determines more than the color of a label. It changes who may initiate the proposal, the minimum review time, the voting threshold, the possibility of amendments, the required implementation evidence and the effect of a favorable ballot. Constitutional changes should generally demand higher scrutiny than ordinary proposals because they alter the rules applied to future decisions. A vote changing a community's document-retention policy should not accidentally acquire the authority to modify consensus signatures or the total SHB supply.

The global token supply is fixed at **100,000,000 SHB**. This chapter's proposal engine must not admit an action type that mints additional tokens, even if a local community records unanimous support for such a request. A community constitution is subordinate to network invariants. Likewise, the **one-minute nominal block interval** is a protocol design parameter, not an ordinary local preference that a single governance space may change. Any change to a globally fixed consensus setting would require an explicitly specified protocol-upgrade process and would fall outside the local proposal permissions described here.

The protocol should use an **authority matrix**: for each proposal type, a bounded list of state keys and action families that the corresponding space may affect. A local community can elect its own moderator, update its own voting thresholds subject to constitutional constraints, or create a subcommittee. It cannot cast a vote on behalf of a neighboring community or instruct every validator to ignore the signature verification rule. Permission checks must be performed at validation time, not assumed from the fact that an action was shown in a user's dashboard.

Proposal type	Ordinary authority	Elevated consideration
Advisory	Publish a verified collective preference	Not automatically executable
Operational	Assign work and report on milestones	Must identify accountable implementers
Appointment	Create or end a scoped mandate	Requires term, duties and revocation rules
Resource authorization	Release existing vault units	Must preserve supply and audit recipients

Proposal type	Ordinary authority	Elevated consideration
Constitutional	Change eligible community rules	Stronger quorum, review and delay
Protocol upgrade	Change shared network behavior	Network-wide procedure, version coordination

A proposer must declare the expected effect of approval. The required field could be effect_class, taking one of three values: **record-only**, **mandate**, or **machine-executable**. In the first case, the only guaranteed chain effect is an authenticated result. In the second, approval creates a verifiable instruction for designated people, whose subsequent real-world performance cannot be proven merely by code. In the third, approval may authorize a deterministic operation against SHB state, but only after the execution engine confirms its conditions and capabilities. No vote can itself make an external institution obey a decision.

The distinction is especially important when several communities share one ledger. Shared history does not create shared jurisdiction. A university club may administer its own membership votes, while an open-source software association uses SHB to approve a release plan. They can inspect one another's records without obtaining the power to change one another's rules. Chapter 4 defines separate governance spaces; the proposal object is where their separation becomes enforceable.

8.4 Sponsorship, Thresholds and Entry into Review

Not every suggestion needs an immediate ballot. Requiring a formal vote for each minor question would consume network capacity and participant attention, while an absolute power to reject proposals at the entrance would invite gatekeeping. The design accordingly provides for an optional sponsorship stage that measures whether an idea has enough declared support to enter scheduled review. Sponsorship is not approval of the proposal's substance; it is a request that the community consider it.

The constitution may define sponsorship as a minimum number of distinct qualified accounts, a minimum amount of eligible SHB voting weight, an authorized committee endorsement or a combination of these. Token-weighted sponsorship follows the same basic distinction as voting: addresses are not persons, and multiple addresses can be controlled by one actor. A threshold based on unique accounts is particularly vulnerable to artificial splitting unless an independent membership mechanism exists. Consequently, the initial public-token configuration should favor verifiable eligible weight, with optional membership qualifications where a community can justify and administer them.

One SHB should not create unlimited sponsorship weight within the same proposal. Let the proposal's sponsorship reference state identify eligible balances. Signed sponsorship instructions designate some or all of those balances as support for placing the question on the agenda. Multiple sponsorship instructions from the same account must be aggregated under a unique authorization identifier, and any delegation must be checked against the same entitlement graph used for the proposal type. The protocol may permit replacement of an earlier sponsorship choice, but not an additive series that repeatedly counts the same source units.

```
sponsor_weight(p) = sum_a min(sponsored_units(a,p), eligible_units(a,p)); qualify(p) iff
sponsor_weight(p) >= T_s(p)
```

Eligibility for the future *ballot* is deliberately a separate calculation. A sponsor can support holding a vote yet later oppose the final proposal. The ballot snapshot occurs at its independently defined reference height. A governance interface should make this difference visible: “Support bringing this question to a vote” is not equivalent to “Vote yes.” The sponsorship stage may even attract people who strongly disagree with the idea but consider it important enough to decide formally.

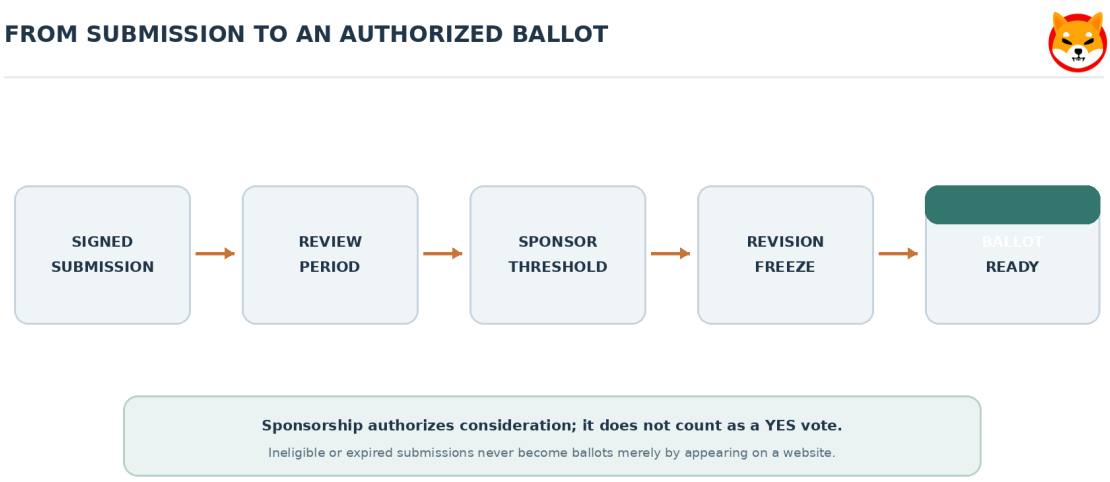
Algorithm 8.1 — Verify a proposal submission and admit it for review

```
function SubmitProposal(tx, finalized_state):
    require VerifySignature(tx.author_key, tx.signing_bytes, tx.signature)
    require FreshNonce(tx.author_key, tx.account_nonce)
    require tx.chain_id == finalized_state.chain_id
    g = LoadSpace(tx.space_id)
    require g.active and AuthorMayPropose(g, tx.author_key, finalized_state)
    policy = LoadConstitution(g, tx.constitution_hash)
    require policy.is_active_for_submission
    require AllowedKindScopeAction(policy, tx.kind, tx.scope, tx.action_set)
    require ValidCanonicalEncoding(tx.body) and BoundedResources(tx.body)
    require ValidHeightOrder(tx.heights, policy, finalized_state.height)
    id = SHA256("SHB-PROPOSAL-v1" || tx.chain_id || CanonicalEncode(tx.body))
    require not Exists(id) and WithinActiveProposalLimit(tx.author_key, g)
    if policy.requires_bond: LockExistingSHB(tx.author, policy.bond_amount)
    Persist(id, tx.body, status="Under Review", rule_hash=policy.hash)
    EmitReceipt(id, finalized_state.height)
    return id
```

The pseudocode describes application-level validity, not an independent block-consensus algorithm. A validating node applies the same rules to accepted transactions in the same order. Checks involving a bond must operate atomically with proposal creation; otherwise an implementation might lock funds for a submission that subsequently fails validation. Expiry, storage bounds, rate limits and duplicate handling require precise error codes so that a wallet can distinguish a delayed proposal from a rejected one.

Consider a hypothetical technical cooperative whose local electorate consists of 3,000,000 eligible SHB out of the network's 100,000,000 fixed total. Its constitution might require 25,000 eligible units of sponsorship for a new operational proposal, together with a minimum 48-hour review stage. An author holding 18,000 eligible SHB and two supporters holding 9,000 and 4,000 respectively could qualify a submission with 31,000 units of support. That figure means the question has earned a ballot, not that 31,000 people support it, and not that the proposal is destined to pass.

FIGURE 8.2 ADMISSION, SPONSORSHIP AND FREEZING A BALLOT FOR LATER VOTING



The execution record identifies which authorization was accepted and when it became effective.

8.5 Amendments, Competing Proposals and Version Integrity

An idea rarely reaches its best form at the moment of submission. Participants may discover an overlooked cost, improve a technical definition or propose a narrower scope that attracts broader support. A governance protocol should make room for this work without allowing the ballot itself to change while people are voting. The most damaging form of amendment is an invisible one: a participant votes for one promise, and an administrator edits the text under the same proposal number before the result is counted.

The SHB architecture provides for an explicit **revision lineage**. During a declared review period, an authorized author may submit a new revision that points to the previously accepted revision and includes a new payload digest. The parent is never overwritten. The accepted chain then contains both descriptions, the time each was accepted, the account that authorized the change and, where the constitution requires it, the renewed sponsorship that followed a material amendment. The current revision is derived from the accepted lineage and the declared cutoff rules, not from the last file uploaded to a web server.

```
revision_id(r) = SHA256("SHB-REVISION-v1" || proposal_id || parent_revision_id ||
CanonicalEncode(revision_body))
```

At the **ballot freeze**, the network commits one exact revision and one exact rule version as the question voters will answer. After that point no textual or executable field belonging to the chosen ballot object may change. A proposed improvement during an active vote must become a separate alternative proposal or wait for a new voting period. This rule can feel inconvenient when a useful correction appears late, but it preserves the meaning of signatures already cast. A typo that has no legal or computational effect can still be published as a separately signed clarification; it must not silently replace the hashed material on which participants relied.

Amendments should be categorized. An explanatory amendment can correct background material while leaving options and actions unchanged; a substantive amendment changes objectives, action parameters, beneficiaries, funding or the qualified electorate. A substantive change must invalidate the previous sponsorship count if those sponsors have not explicitly authorized the revised content. The constitution may set a review interval after each substantive revision, to prevent an author from replacing an agreed text just before the voting window opens.

Two competing proposals may be mutually exclusive. For example, a cooperative might consider maintaining an existing open-source library, rewriting it or discontinuing support. Three separate yes/no ballots could all pass under careless rules, yielding incompatible mandates. The proposal engine should permit an explicit **conflict group** with mutually exclusive execution rights. The grouping is recorded before ballots open. A result that would execute a state-changing action already committed under an incompatible mandate must be rejected or sent through a declared resolution procedure. This does not imply every community must use ranked-choice voting; Chapter 9 will examine ballot methods separately. It does mean the network cannot pretend that conflicting state changes are automatically compatible.

A more general improvement is an **alternative bundle**, in which the community votes among several signed versions of a single action plan. Each alternative must have a stable digest and an identical disclosed scope of authority. Once the bundle is frozen, neither the author nor a committee may add a more attractive option after some ballots have been cast. The protocol should allow third-party observers to reconstruct the available option set from a finalized block and verify that every accepted ballot references that same set.

A proposal can be superseded without being erased. The record should say what replaced it, by whose authority, and whether any previously authorized action has already been executed. Replacement of a future appointment mandate is different from reversal of an action already taken in the physical world. A cryptographic record can preserve the sequence of instructions; it cannot make irreversible events vanish. That distinction will matter when the execution layer is specified.

Algorithm 8.2 — Amend and freeze a proposal revision

```
function AcceptRevision(proposal_id, candidate, state):
    p = LoadProposal(proposal_id)
    require p.status in {"Under Review", "Sponsored"}
    require state.height < p.revision_cutoff_height
    require AuthorizedEditor(candidate.signer, p, state)
    require candidate.parent_revision_id == p.current_revision_id
    require VerifySignature(candidate.signer, candidate.bytes, candidate.sig)
    require ValidRevisionPayload(candidate, p.rule_hash)
    rid = HashRevision(candidate, proposal_id)
    AppendRevision(p, rid, candidate.content_digest)
    if MaterialChange(candidate): ResetSponsorshipAsRequired(p)
    return rid
function FreezeBallot(proposal_id, state):
    p = LoadProposal(proposal_id)
    require state.height >= p.freeze_height and SponsorshipSatisfied(p)
    require ValidScheduledOptionSet(p) and RuleVersionAvailable(p)
    p.ballot_hash = HashBallot(p.current_revision_id, p.rule_hash,
                               p.options, p.snapshot_rule, p.schedule)
    p.status = "Scheduled"
    return p.ballot_hash
```

The two procedures share a subtle requirement: a revision accepted in the same block as a freeze operation must be ordered deterministically. The protocol cannot allow one validator to freeze the earlier revision while another uses the later one. The simplest rule is to process transactions in canonical block order and evaluate the revision_cutoff_height consistently. For safety, the proposed first release should reject amendment operations at or after the freeze height, even when the block also contains the freeze transition.

8.6 Discussion, Evidence and the Limits of On-Chain Truth

Good governance needs informed disagreement. A verifiable ballot proves how eligible units were counted, but it does not prove that participants understood the proposal, received balanced evidence or had an opportunity to object. SHB therefore separates a **deliberation record** from the **binding proposal record**. The binding record contains the question and its rules. The deliberation record may include arguments, documents, endorsements, objections, expert comments and links to independent analyses. Both should be inspectable, but they require different controls.

A practical first release would store only compact commitments to substantial documents on-chain. Files themselves could be distributed by community archives or content-addressed storage. The system must specify whether a proposal can proceed when a mandatory document is unavailable. A reference to a missing file cannot be treated as proof that the file was made available. A community might require copies from at least two independent archives before the review clock can begin, but such an availability test would still be operational evidence rather than an absolute guarantee of future retrieval.

Every submitted document should include a declared content type, size, digest and relationship to a particular proposal revision. Content-addressed distribution makes accidental or deliberate substitution detectable: if the bytes change, their digest changes. It does not establish the truth of the bytes. An apparently authoritative economic forecast may still rest on unrealistic assumptions. A signed expert statement proves that a key attested to the statement; it does not guarantee the expert is correct. SHB should preserve these distinctions prominently whenever it claims verifiability.

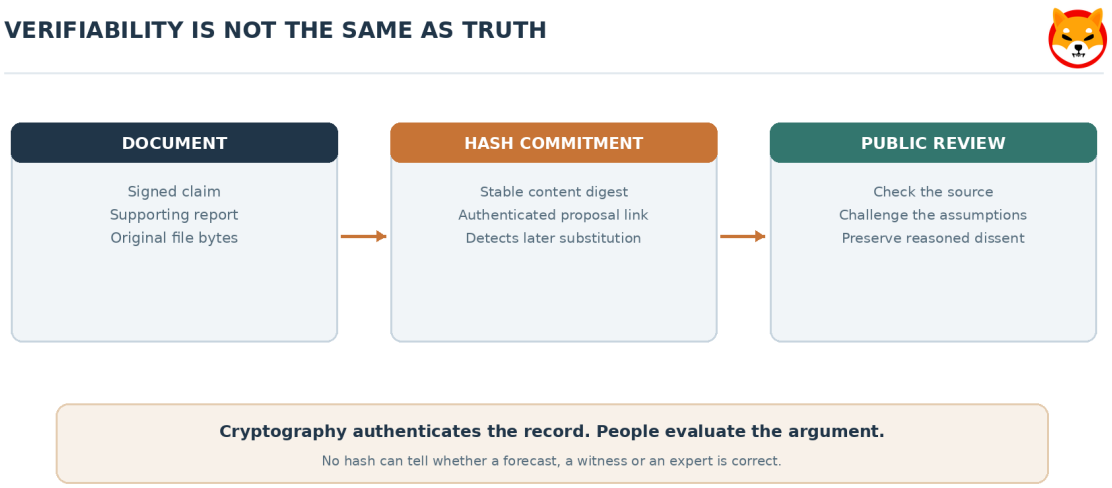
A deliberation space may permit comments without forcing each sentence into a permanent consensus record. A community can choose to commit to periodic discussion digests or signed editorial summaries. That choice reduces chain load but places trust in the archiving process. If participants want an independently reconstructable discussion, they must preserve the underlying signed messages and the rules governing moderation. No public ledger can reconstruct a deleted off-chain post that nobody retained. A mirror protocol and exportable discussion packages are therefore useful complements to on-chain commitments.

The community should also record **dissenting opinions**. A failed proposal may contain ideas worth revisiting, and an approved proposal may have raised substantial objections later shown to be correct. A

dissent record need not carry a special voting privilege. It may simply reference the final ballot, list supporting evidence, and remain accessible in the same archive. That is part of institutional memory: not merely the conclusion, but the known disagreements that accompanied it.

A credible review process should publish the scope of moderation. The community may remove illegal material from a particular interface, and archival services may be required to comply with applicable law. The blockchain should not imply that all forms of content can remain universally accessible forever. It can preserve a cryptographic commitment and an audit trail of moderation decisions while allowing interfaces to restrict display of harmful or unlawful materials. The relationship between permanent commitments and personal data protection deserves further review before public deployment.

FIGURE 8.3 DOCUMENT COMMITMENTS AND THE INDEPENDENT EVALUATION OF CLAIMS



Document commitments verify the submitted bytes, not the truth of the underlying claims.

8.7 Scheduling, Eligibility Snapshots and Ballot Readiness

Governance scheduling is a protocol operation, not a browser countdown. Each proposal has a defined series of boundaries: submission, review cutoff, revision freeze, eligibility snapshot, opening height, closing height, tally and any delayed execution. The ordering is part of the constitutional interpretation of a decision. In particular, an account must know which eligible balance will be used before it casts a ballot, and an author must not be permitted to move the closing height after observing an unfavorable trend unless the constitution explicitly provides a narrowly constrained emergency procedure.

SHB's nominal block cadence is **60 seconds**. The arithmetic is straightforward: a nominal hour contains 60 target blocks; a day contains 1,440; a 48-hour review interval contains 2,880; and a five-day ballot window contains 7,200. These are planning values, not guarantees about elapsed civil time. If validators cannot agree on a block for a period, network height may remain unchanged while clocks continue to advance. The governance engine should describe such an interruption accurately instead of pretending a height-based deadline was reached in wall-clock time.

```
target_blocks(duration_seconds) = duration_seconds / 60;  target_blocks(5 days) = 7,200
```

For a first implementation, a **height-based schedule** is easier for independent nodes to reproduce than a deadline dependent on local computer clocks. The user interface may display an approximate calendar time based on recent block production, with a warning that finalization can be delayed. A legally sensitive community might instead require a hybrid policy that combines height boundaries with a bounded, consensus-validated timestamp. That is a separate specification and must be analyzed for clock manipulation, censorship, liveness and conflicting deadlines before use.

The ballot eligibility snapshot should be fixed by the proposal rules, not chosen opportunistically after seeing who has accumulated tokens. A possible policy takes the last finalized state before the opening height. Under that policy, if a proposal opens at height 2,002,880, its reference state is height 2,002,879. Transfers finalized after the reference state do not alter the voting weight for that ballot. Chapter 7 describes why this prevents a single transferable balance from being repeatedly used to produce extra voting weight during one poll. The same rule must apply to escrowed, vesting-locked or delegated balances, using the applicable eligibility graph.

```
snapshot_height(p) = open_height(p) - 1, under the stated default policy
```

The proposal shall contain the hash of the governance constitution in effect for that ballot. A community that changes its general rules during the review period must not silently update an already frozen proposal. The protocol may require the proposal to restart under the new constitution or expressly allow it to continue under its pinned rule version. The selected behavior belongs in the constitutional transition policy. It must be possible to tell exactly which version governed a historical result even after newer rules have replaced it.

The readiness checks should therefore cover at least six properties: the proposal has a valid frozen revision; any sponsorship requirement is met; the ballot options are unambiguous; the snapshot rule and participant eligibility are defined; the voting thresholds can be computed without division by zero or missing inputs; and the requested execution actions fit the authority of the issuing space. A user interface that hides a failed readiness check should not be able to persuade validators to accept the ballot anyway.

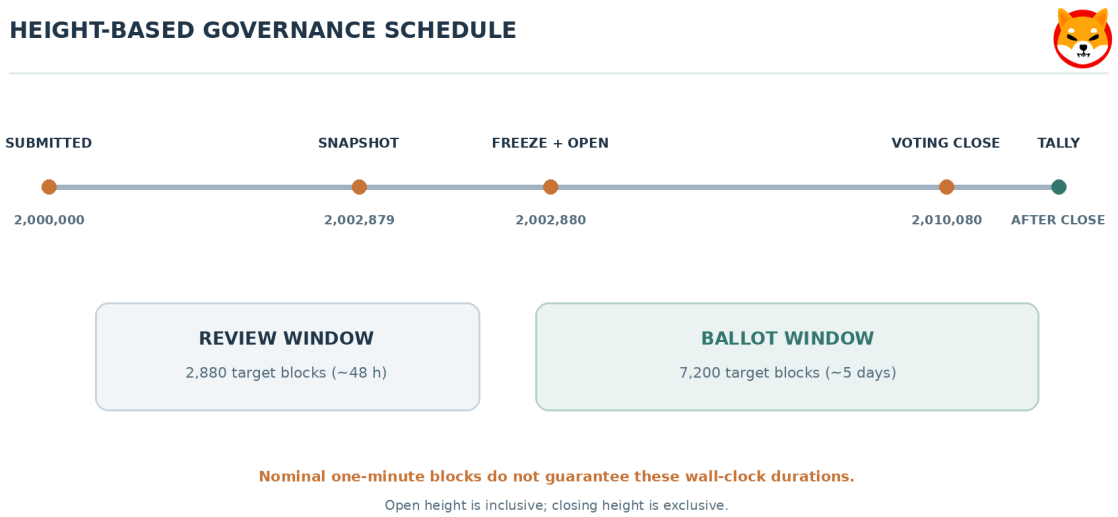
Consider the cooperative from Section 8.4, with 3,000,000 eligible SHB. Suppose its operational constitution sets a two-day review and a five-day vote, with a required 20% turnout and a 60% approval threshold among decisive yes and no votes. If the submission is finalized at height 2,000,000 and the next stages occur without delays, a simplified schedule might freeze the revision after 2,880 blocks and open the ballot at height 2,002,880. The resulting five-day window is 7,200 heights, so the closing boundary would be 2,010,080. A more conservative constitution could add a separate interval after freeze, shifting the opening and closing heights accordingly. The example is illustrative and does not imply that SHB genesis has occurred or that height 2,000,000 was ever reached in production.

Schedule element	Illustrative value	Governance meaning
Submission height	2,000,000	Authenticated registration of the draft
Review/freeze interval	2,880 target blocks	Roughly 48 hours at 60 s target
Opening height	2,002,880	Voting operations become eligible
Snapshot height	2,002,879	Fixed voting entitlement reference
Voting duration	7,200 target blocks	Nominal five-day ballot window
Closing boundary	2,010,080	New ballots rejected at or after boundary
Approval / turnout	60% / 20%	Locally specified thresholds

The finality rule must specify whether a transaction in the block whose height equals the closing boundary is too late. A clear proposed convention is the half-open interval **`open_height <= included_height < close_height`**. Under this convention a ballot in the closing-height block is invalid, regardless of when the participant first clicked “submit.” Wallets should warn that a transaction accepted by a relay but not finalized before the boundary may fail to count. The ledger cannot promise to include all signed ballots in an overloaded or disrupted network.

```
BallotInWindow(p,h) = 1 iff open_height(p) <= h < close_height(p); otherwise 0
```

FIGURE 8.4 ILLUSTRATIVE HEIGHT-BASED SCHEDULING UNDER A ONE-MINUTE BLOCK TARGET



The schedule uses nominal one-minute blocks. Network interruptions may extend elapsed time.

8.8 From Submission to Final Execution Status

A proposal reaches its most important moment after the vote: the distinction between being approved and being carried out. A signed result is evidence that the declared electorate followed the voting procedure. It is not evidence that a bridge was built, a manual was translated or a representative performed the duties assigned. SHB should preserve the complete transition from question to tally to mandate without claiming that one blockchain transaction settles every practical matter.

For an ordinary yes/no ballot, the tally operation aggregates accepted signed votes from the frozen snapshot, applies the constitution's turnout and approval rules and produces a deterministic result record. The record includes the exact ballot hash, total eligible weight, weights for yes, no and abstain, number of accepted voting instructions, result status and the governing thresholds. An external observer should be able to reproduce the result from ledger data. A published dashboard percentage, without the denominator and rule version, is insufficient.

```
approved(p) iff (Y + N + A)/W_eligible >= q_min AND Y/(Y + N) >= a_min, provided Y + N >
0
```

Using the cooperative example, 3,000,000 SHB are eligible. If 510,000 units support the proposal, 190,000 oppose it and 80,000 abstain, turnout is (510,000 + 190,000 + 80,000) / 3,000,000 = 26%. Approval among decisive votes is 510,000 / (510,000 + 190,000) = 72.857...%. Both the hypothetical 20% turnout and 60% approval conditions are met. The result may be recorded as approved. It must not be described as representing 72.857% of all SHB in existence or 72.857% of the people in the community. The actual denominator is the weight of the yes/no votes cast in that space.

A constitution may add veto periods, supermajorities, identity-based qualifications or special procedures for irreversible operations. Those requirements have to be included in the frozen ballot's rule commitment. They cannot be invented by the author after a favorable vote. For a proposal that was approved but cannot yet be executed, the result should move to **Execution Pending**, not **Executed**. A mandatory delay before executing a constitutional amendment is an example. Another is an appointment decision awaiting acceptance by the proposed representative.

Algorithm 8.3 — Close a ballot and authorize a permitted action

```
function FinalizeProposal(proposal_id, state):
    p = LoadProposal(proposal_id)
    require state.height >= p.close_height and p.status == "Voting"
    ballots = ReadAcceptedBallots(p.ballot_hash, p.snapshot_height)
    tally = RecalculateDeterministically(ballots, p.rule_hash)
    require NoDoubleCount(tally, p.entitlement_snapshot)
    result = EvaluateThresholds(tally, LoadRule(p.rule_hash))
    StoreFinalTally(proposal_id, tally, result)
    if result != APPROVED: p.status = "Rejected"; return result
    p.status = "Execution Pending"
    p.execute_after = max(state.height, p.minimum_execution_height)
    return APPROVED
function ExecuteAuthorizedAction(proposal_id, state):
    p = LoadProposal(proposal_id)
    require p.status == "Execution Pending"
    require state.height >= p.execute_after
    require ActionStillPermitted(p.action_set, p.rule_hash, state)
    ApplyActionsAtomically(p.action_set)
    p.status = "Executed"
    StoreExecutionReceipt(proposal_id, state.height)
```

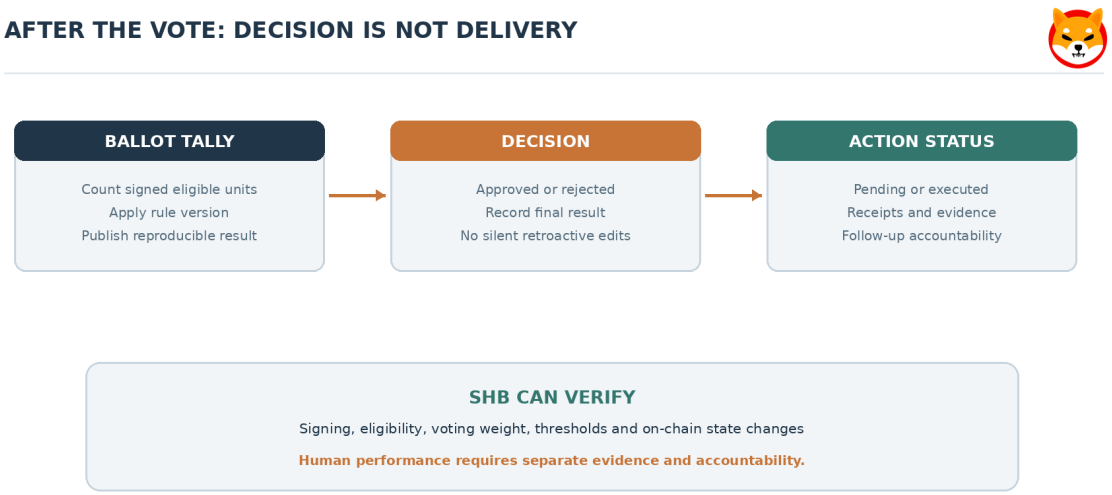
The second procedure is intentionally restricted to actions whose meaning and effects can be verified in protocol state. An external mandate should receive an execution report, not a false on-chain claim that the off-chain work has already occurred. If a working group is asked to publish an accessibility guide, the chain may record the appointment, deadline, publication digest and subsequent community review. It cannot independently verify that the guide is accurate or that every intended reader can use it.

State transitions must be monotonic where finality requires it. An accepted historical tally remains part of the ledger even if a later, separate proposal reverses the original policy. Reversal produces a new decision linked to the earlier one. A final result must not be edited in place to create the appearance that an unpopular decision never happened. Similarly, if an executable action fails due to a changed precondition, the protocol should preserve a failure receipt and require a new authorization rather than silently applying a materially different operation.

Proposal expiry, insolvency of a project treasury, loss of an implementation key and network downtime all raise different problems. The protocol should define failure categories, allow bounded retries for deterministic operations and retain evidence of every failed attempt. For human mandates, a community may specify an escalation or recall process. The execution engine is not a general authority over people; it is a record of what they collectively authorized, what the software could enforce and what later evidence they supplied.

An implementation report could include a signed report digest, the identity of the reporting representative, links to independently retained artifacts, deadlines, any objections, and an optional follow-up vote assessing performance. Such reporting is a proposal for the SHB governance layer, not a claim that the chain possesses a reliable oracle for physical events. The reliability of that last step depends on evidence and social institutions in addition to cryptography.

FIGURE 8.5 DECISION FINALITY AND THE SEPARATE PROOF OF IMPLEMENTATION



A decision is distinct from its implementation; each has a separate verifiable record.

The distinction between a proposal, a vote and an accomplished action is the organizing principle of this chapter. A proposal is an exact question, signed under known rules. A ballot is a finite interval in which eligible influence is applied to that question. A result is a reproducible calculation from accepted instructions. An execution record is evidence of what the network or a responsible participant did afterward. When these stages are conflated, a community is left trusting its interface and hoping the rest is true. When they are separated and linked by authenticated records, disagreements become easier to identify and investigate.

The next chapter turns from the construction of a question to the mathematics of the answer. It examines the counting methods that different communities may select, the treatment of abstentions, multiple options and qualified majorities, and the safeguards needed to make a tally intelligible to anyone willing to reproduce it.

References and Technical Foundations

[31] J. Benet, “IPFS — Content Addressed, Versioned, P2P File System,” arXiv:1407.3561, July 2014. https://arxiv.org/abs/1407.3561

[32] S. Josefsson and I. Liusvaara, “Edwards-Curve Digital Signature Algorithm (EdDSA),” RFC 8032, January 2017. https://www.rfc-editor.org/rfc/rfc8032

[33] B. Laurie, A. Langley and E. Kasper, “Certificate Transparency,” RFC 6962, June 2013. https://www.rfc-editor.org/rfc/rfc6962

[34] E. Buchman, J. Kwon and Z. Milosevic, “The Latest Gossip on BFT Consensus,” arXiv:1807.04938, July 2018. https://arxiv.org/abs/1807.04938

PART III / THE ART OF DECIDING

Chapter 9

Counting What Matters

Transparent arithmetic for credible collective decisions.



9.1 Direct Token-Weighted Voting

A ballot has little value unless different honest nodes can count it and arrive at the same answer. For SHB, the default unit of voting influence is the native token: **one eligible SHB represents one unit of voting weight**. Fractional SHB holdings contribute fractional weight, counted by the protocol in integer base units rather than floating-point arithmetic. The network has a fixed total supply of **100,000,000 SHB**, but a particular governance space may admit only a subset of those holdings. Global supply must never be mistaken for the electorate of every proposal.

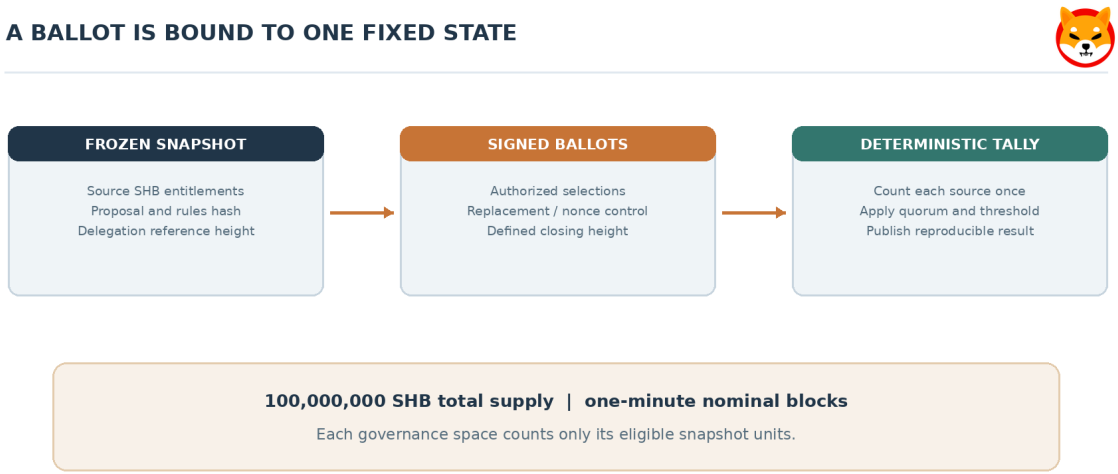
Each ballot refers to one immutable proposal revision, one constitutional rule version and one finalized eligibility snapshot. The snapshot gives each source account a maximum voting entitlement. Authorized delegates may cast a ballot on behalf of those source units, but delegation neither creates new SHB nor permits the same unit to be counted twice. A participant who transfers tokens after the snapshot does not rewrite the electorate for a ballot already in progress. This is a deliberate trade-off between immediate transferability and a stable, auditable vote.

A valid voting instruction contains the proposal identifier, the chosen method, the encoded selection, the authorizing signature or delegated authority, and a nonce or replacement counter. A replacement ballot, if permitted by the constitution, supersedes an earlier ballot for the same eligible source units; it must not add a second contribution. Multiple instructions may represent different non-overlapping source units, but the tally must account for their origins. The signature establishes authorization, not whether the voter was free from external pressure.

```
W_eligible(p) = sum over all eligible sources a of entitlement(a,p);  W_cast(p) <=
W_eligible(p)
```

The SHB blockchain is designed to target a **one-minute block interval**. A ballot opening at height h0 and closing at height h1 is therefore governed by finalized block heights, not an interface clock. Target block time allows a community to estimate its schedule; network interruptions can extend its real-world duration. The rule used by the tally must be published before the first voting instruction is accepted.

FIGURE 9.1 SNAPSHOT, BALLOT AUTHORIZATION AND DETERMINISTIC RESULT



Conceptual SHB protocol diagram. OT Development Team, February 2020. This is a design proposal, not a statement of implemented functionality.

9.2 Approval, Rejection and Abstention

The simplest ballot offers **Yes**, **No** and **Abstain**. These outcomes answer different questions. A yes vote supports the proposed action; a no vote opposes it; an abstention can affirm participation without declaring a preference between approval and rejection. An absence of a ballot is not automatically an abstention. It may represent disinterest, lost access, an unavailable participant or a deliberate refusal to participate.

We propose two independent tests for an ordinary binding ballot. First, a turnout quorum measures whether sufficient eligible weight took part. Second, an approval rule measures support among decisive votes. Unless a community explicitly adopts another constitution, abstentions contribute to turnout but not to the Yes-versus-No approval denominator. This distinction prevents a valid abstention from being silently transformed into a negative vote.

```
T = Y + N + A;  turnout = T / W_eligible;  approval = Y / (Y + N), if Y + N > 0
```

Consider a space with **12,000,000 eligible SHB**. A proposal receives 4,800,000 Yes, 2,400,000 No and 800,000 Abstain; 4,000,000 do not vote. Turnout equals 8,000,000 / 12,000,000, or **66.67%**. Approval among decisive votes equals 4,800,000 / 7,200,000, also **66.67%**. A constitution requiring 50% turnout and at least 60% approval would approve it. If the constitution instead defined approval as Yes divided by all participating weight, the value would be 60%. The chain must publish which test it applied rather than display an unexplained percentage.

Measure	Eligible / counted SHB	Result
Eligible snapshot	12,000,000	100.00% reference
Yes / No / Abstain	4,800,000 / 2,400,000 / 800,000	8,000,000 participation
Turnout	8,000,000	66.67%
Decisive approval	4,800,000 / 7,200,000	66.67%

9.3 Multiple-Option Ballots

A binary ballot is inadequate when the choice involves several legitimate approaches. We have therefore included a **single-choice ballot**, where each eligible unit backs exactly one of several options, and an **approval ballot**, where each voter may approve multiple options. The two methods must not share a tally rule by accident. Under single-choice voting, the total weight cast across alternatives

cannot exceed the weight of the participating electorate. Under approval voting, one source's weight may appear in several option totals; turnout must still count that source only once.

A community considering three software designs may prefer the option receiving the most support, require a majority, or schedule a runoff when no option crosses the declared threshold. A plurality winner is not necessarily supported by most participants. The proposal must identify whether “none of the above” or the current policy is an explicit alternative. Where ties are possible, a deterministic procedure—such as a new ballot or a previously specified neutral tie-break rule—must be applied. A validator must never choose the winner by sorting option labels in a user interface.

```
Single-choice: sum over options i of weight_i = W_cast;  Approval: 0 <= weight_i <=
W_cast for each i, while W_cast counts unique eligible sources
```

9.4 Ranked Preferences

Some decisions are better represented by an ordered preference than by a single mark. Ranked-choice ballots allow a participant to express that option A is preferred to B, while B is preferable to C. Methods for aggregating such rankings were studied long before the SHB proposal. Two plausible configurations for a future SHB governance module are instant-runoff voting and pairwise comparison; they are different rules and may produce different results from the same ballots. Neither should be described as universally fair.

In a weighted instant-runoff example, assume **10,000 voting SHB** are represented in three preference groups: 4,000 prefer A > B > C; 3,500 prefer B > C > A; and 2,500 prefer C > B > A. No option initially has more than half of the 10,000. C has the fewest first preferences and is eliminated. Its 2,500 units transfer to their next valid preference, B. The final round is **B = 6,000** and **A = 4,000**. These are units of voting influence, not numbers of distinct people.

The ranking format must specify the treatment of incomplete rankings, identical rankings, exhausted ballots and exact ties. For complex committees, a pairwise comparison approach such as the Schulze method may be considered, but it requires explicit specification and independent test cases before adoption. We propose to deploy direct voting first and treat alternative tally methods as separately versioned governance modules.

9.5 Conditional and Weighted Voting

Token weighting and conditional decisions answer different questions. Token weighting determines *whose instruction carries how much influence*; a condition determines *when an approved instruction may take effect*. For example, a community may vote to authorize a technical upgrade only after independent review or to appoint a representative for a defined term. The ballot can approve such a condition, but implementation must still verify its satisfaction through the execution mechanism described in later chapters.

The baseline voting weight is proportional to eligible SHB. We do not propose hidden multipliers for reputation, account age or validator status. A governance space may experimentally require membership credentials or a special majority for a particular decision class, yet any altered weighting scheme must be declared in its constitution before the eligibility snapshot. Neither operators nor application developers should be able to adjust a completed ballot by applying a new coefficient afterward.

9.6 Quorum and Participation

The quorum rule protects against a small participating fraction deciding matters for an entire community. It can also be abused: an unrealistically high quorum makes inaction the default and gives abstaining opponents a veto. The practical answer is not one global turnout threshold for all communities, but published, bounded rules proportionate to the decision. Routine working-group choices may use moderate thresholds; constitutional revisions may require a stronger test.

Quorum uses the **eligible weight for the specific proposal**, not the global 100 million SHB supply, unless the ballot genuinely concerns every eligible network holder. Turnout is calculated from distinct eligible source units. Invalid signatures, votes received after the closing height and superseded ballot versions contribute nothing. A quorum measurement must not inflate because an account signs a hundred replacement transactions or because one delegate holds a mandate from several accounts and casts duplicate instructions.

9.7 Voting Thresholds and Supermajorities

Approval thresholds are best expressed as ratios of integers to avoid rounding ambiguity. If a constitution requires at least two-thirds approval, an integer check such as **3Y >= 2(Y + N)** is reproducible for any positive decisive total. A simple majority may instead require **Y > N**, which rejects a tie. The constitution must say whether equality at the threshold passes and whether a zero-decisive-vote ballot fails; We propose that it does fail.

```
Pass(p) = [T / W_eligible >= q] AND [Y + N > 0] AND [Y / (Y + N) >= s]
```

A proposal changing an ordinary meeting schedule need not require the same support as one rewriting a community constitution. More stringent rules may apply to appointment recalls, financial resource authorizations and changes to voting eligibility. Each proposal must lock its threshold definition with its versioned rules so a passing majority cannot retroactively lower the barrier it just failed to satisfy.

9.8 Vote Verification and Result Finality

The tally is a deterministic computation over the finalized voting record and the frozen snapshot. Before declaring an outcome, the protocol checks the proposal revision, eligible source units, delegated authority, signature validity, ballot window, replacement sequence and method-specific encoding. The result is accompanied by its total eligible weight, unique participation weight, option totals, rule version and closing block reference. A full node can reproduce the calculation without trusting the interface that first displayed it.

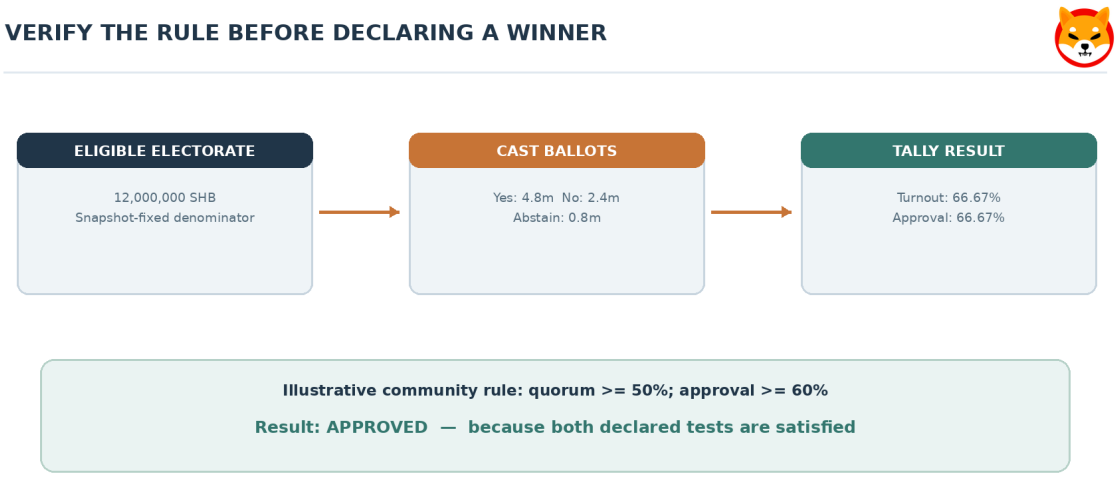
Algorithm 9.1 — Deterministic tally of a binary SHB ballot

```
function TallyBinary(proposal, finalized_state):
    rules = LoadFrozenRules(proposal)
    snapshot = LoadEligibleSourceEntitlements(proposal.snapshot_height)
    Y = 0; N = 0; A = 0
    for source in snapshot in canonical_source_order:
        ballot = LatestValidAuthorizedBallot(source, proposal, finalized_state)
        if ballot is absent: continue
        weight = EffectiveUncountedEntitlement(source, proposal)
        if ballot.choice == YES: Y += weight
        if ballot.choice == NO: N += weight
        if ballot.choice == ABSTAIN: A += weight
    require Y + N + A <= SumEligible(snapshot)
    quorum_ok = RatioAtLeast(Y + N + A, SumEligible(snapshot), rules.quorum)
    support_ok = (Y + N > 0) and RatioAtLeast(Y, Y + N, rules.support)
    return CertifiedResult(quorum_ok and support_ok, Y, N, A, rules.hash)
```

Delegated voting requires the LatestValidAuthorizedBallot procedure to resolve source-unit authority according to the frozen delegation rules, including revocation and cycles. That resolution is detailed in Chapter 10. The present algorithm deliberately counts *source entitlements*, never the same delegated weight once at the source and again at its representative.

The consensus validators finalize a block that records the outcome; they do not select a preferred option. A later political reversal must be expressed as a new decision, not as an alteration to an already certified tally. A cryptographically verifiable result does not by itself provide a secret ballot, prevent coercion or prove that every eligible participant could access the network. Those limitations remain visible in the SHB security model.

FIGURE 9.2 SEPARATE QUORUM AND APPROVAL MEASUREMENTS IN A WEIGHTED BALLOT



Conceptual SHB protocol diagram. OT Development Team, February 2020. This is a design proposal, not a statement of implemented functionality.

9.9 A Practical Standard for Every Ballot

The protocol can support several voting methods without making every early deployment complicated. The initial recommended profile is a signed, token-weighted Yes/No/Abstain ballot, bound to a single proposal revision and finalized eligibility reference. Multi-option and ranked-choice ballots can be introduced as optional, separately tested methods. Each method has to publish its input encoding, tally algorithm, threshold conditions, exceptional cases and verification examples.

No interface should announce that “the community has spoken” while concealing which community was eligible, how much of its weight took part and which rule produced the result. SHB's contribution is not a claim to have invented democracy. It is an effort to make the arithmetic, authority and record of a collective choice inspectable by the people governed by it. Counting well is not the end of that responsibility, but it is where the responsibility becomes measurable.

References and Technical Foundations

[35] M. Schulze, “A new monotonic, clone-independent, reversal symmetric, and Condorcet-consistent single-winner election method,” Social Choice and Welfare 36, 267–303, 2011. https://doi.org/10.1007/s00355-010-0475-4

[36] B. Adida, “Helios: Web-based Open-Audit Voting,” Proceedings of the 17th USENIX Security Symposium, 2008. https://www.usenix.org/legacy/event/sec08/tech/full_papers/adida/adida.pdf

[37] M. R. Clarkson, S. Chong and A. C. Myers, “Civitas: Toward a Secure Voting System,” IEEE Symposium on Security and Privacy, 2008. https://doi.org/10.1109/SP.2008.32

PART III / THE ART OF DECIDING

Chapter 10

Delegation Without Abdication

Representation should save time, not dissolve authority.



10.1 The Principles of Liquid Democracy

Delegation is a practical answer to a familiar problem: few members have the time to study every issue their community faces. A participant might vote personally on constitutional questions while relying on someone with technical expertise for software decisions. The arrangement only works if the limits are clear and the mandate can be revoked. LiquidFeedback, first developed in 2009, is an important reference here; SHB builds on the broader liquid-democracy idea with its own accounting of delegated token weight.

Delegation is especially useful in a network whose purpose extends beyond occasional elections. A software community may vote on technical standards, membership questions, public communications, and organizational appointments in the same month. Few participants are equally informed about every matter. The SHB design therefore treats representation as a configurable assignment of **voting authority**, never as an assignment of token ownership. The holder's SHB balance does not move when voting authority is delegated.

The global supply remains **100,000,000 SHB**. That supply is neither multiplied by delegations nor automatically eligible in each community. A delegate can speak for many token units, but every unit must remain attributable to one eligible source at a defined reference height. One wallet address does not necessarily represent one person; the system cannot infer personal identity or independent judgment from an address alone.

10.2 Delegating Voting Authority

The proposed delegation instruction is a signed object with source, delegate, governance_space, topic_scope, amount, activation_height, expiry_height, nonce, and chain_id. A partial assignment is permitted. For instance, a holder with 1,000 eligible SHB could delegate 600 to an engineering representative and retain 400 for a direct vote. The delegate's wallet can display the represented 600, but cannot spend or transfer those tokens. The instruction is rejected if its source lacks sufficient uncommitted entitlement, its scope is invalid, or its signature does not verify.

Delegation records are consensus transactions. A wallet should show the destination address, authority granted, affected topics, activation rule, and expiration before requesting a signature. The chain's nominal **one-minute block target** provides a scheduling unit, not a guarantee that an instruction will be finalized one minute after it is submitted. Ballots use finalized state, as defined in Chapters 7–9.

```
For each source a and proposal p: direct_weight(a,p) + sum over delegates b of
assigned_weight(a,b,p) <= eligible_weight(a,p)
```

10.3 Topic-Specific Representatives

A blanket representative may be convenient but creates a single point of influence. SHB instead proposes explicit scopes such as ENGINEERING, MEMBERSHIP, OPERATIONS, or CONSTITUTIONAL. These labels are community-defined identifiers rather than universal political categories. Each proposal carries a frozen topic classification, and each delegation applies only where that classification matches its declared scope.

If a proposal concerns both engineering and membership, the community constitution must specify which classification controls or require separate ballots. The proposer must not change the classification after the eligibility snapshot in order to attract a more favorable delegate electorate. Overlapping assignments are rejected unless a published priority rule resolves them unambiguously. More specific scope may supersede a general authorization, but no eligible unit may reach two delegates in the same ballot.

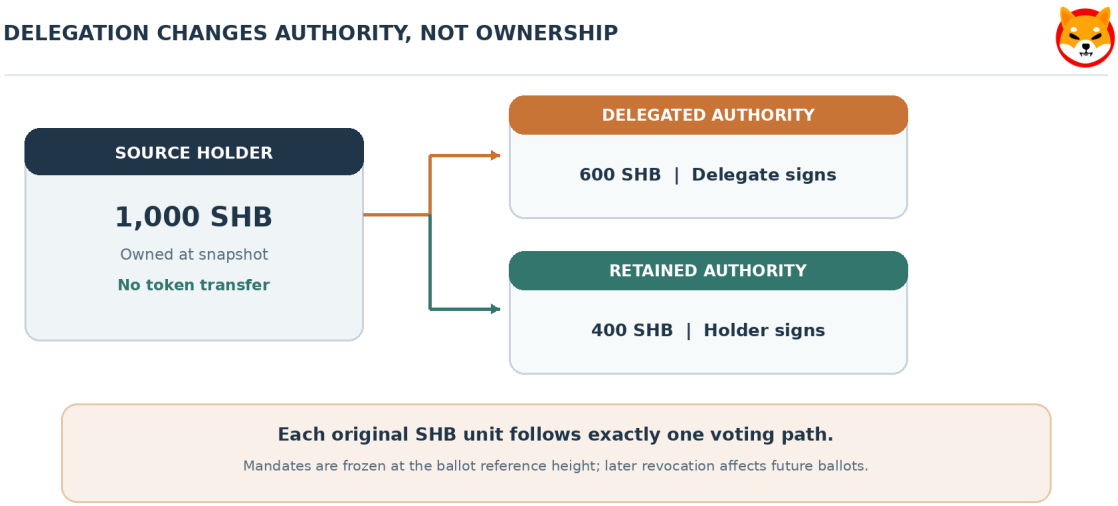
A simple priority order for the first implementation is: **proposal-specific instruction; topic-specific assignment; community-wide assignment; unassigned direct authority**. Each level operates only over source units not already assigned by a higher level. Priority rules must be fixed before the vote opens and included in the hash of the applicable community constitution.

10.4 Revocable Mandates

Withdrawal is essential to credible representation, but it must not make a ballot's electorate change unpredictably. The proposed baseline freezes the effective delegation map at the same finalized snapshot height used to determine voting eligibility. A revocation finalized before that height changes the upcoming ballot; one finalized afterward affects future ballots. It does not retroactively erase a delegate's valid vote already cast under the frozen mapping.

This restriction is not an argument against participant control. Rather, it makes the meaning of a vote verifiable. A future module may offer a controlled direct override during an open ballot, but would need to atomically replace the delegated contribution of exactly the affected source units. Until that mechanism has been formally specified and tested, the simpler snapshot rule is safer. Revocation, expiry, and key rotation remain immediately visible as transactions even when they cannot alter an already frozen election.

FIGURE 10.1 ONE SOURCE BALANCE, TWO AUTHORIZED VOTING PATHS



Illustrative SHB delegation model. OT Development Team, February 2020. The illustration describes proposed behavior, not deployed functionality.

10.5 Delegation Depth and Circular Delegation

Multi-hop delegation allows Alice to authorize Bob, and Bob to forward authority to Carol. This can widen participation, but introduces cycles and difficult questions about liability. What happens when Carol delegates back to Alice? Does Bob's personal preference automatically control the units originally assigned by Alice? How does a validator know whether a path of ten addresses is valid?

The proposed **initial SHB profile permits one delegation hop only**: original holders delegate directly to a final ballot signer, and a representative cannot forward the weight entrusted to them. An account may independently delegate its *own* holdings, but not those received from another account. A

later transitive module could use an acyclic directed graph, a strict depth limit, and deterministic treatment of cycles (for example, rendering the affected source weight uncast). Its rules would require publication and review before activation.

This distinction protects the arithmetic. Delegation does not generate new entitlements, and the same original unit cannot be counted repeatedly as a representation chain grows. Cycles must never be resolved according to network arrival order or a validator's private preference.

10.6 Participation Through Expertise

Delegation can reward careful work rather than visibility alone. A community might maintain public profiles describing a representative's reviewed proposals, technical contributions, attendance record, and declared areas of expertise. Credentials or reputation claims should be independently attributable to their issuer and should not silently multiply token voting weight. A badge for competence is not a second source of SHB.

The network should also make room for small holders. A participant with 25 SHB is entitled to retain or delegate those 25 units even when a large delegate controls thousands. An expert can provide a non-binding assessment without receiving any voting authority. By separating advice, authority, and ownership, SHB leaves room for legitimate expertise without automatically appointing experts as rulers.

10.7 Public Accountability of Representatives

Where voting is public, each completed tally should allow observers to verify the aggregate weight cast by a delegate and the source units represented, subject to the privacy design adopted by the space. A useful report includes the delegate's declared scope, active mandate range, how many source accounts are represented, total represented weight, participation history, and votes recorded. The protocol must not claim to reveal private motives or prove that a representative consulted every holder.

Consider a fictional technical community with **2,000,000 SHB eligible** for a proposal. A representative receives valid assignments covering 110,000 SHB from 52 source accounts; the representative's own eligible balance is 10,000 SHB. If the representative casts a Yes ballot for all authorized units, the recorded Yes contribution is **120,000 SHB**, not 230,000 SHB. The source-attributed tally must account for 110,000 delegated units plus 10,000 personally held units exactly once.

Account or authority	Eligible source weight	Authority to cast
Delegate's own holdings	10,000 SHB	Delegate's own signature
52 participating source accounts	110,000 SHB	Delegate under frozen mandates
Total represented in Yes	**120,000 SHB**	**One counted path per source unit**

Transparent delegate totals help identify concentration, but cannot guarantee independence. A delegate may control undisclosed wallets, and multiple named delegates may coordinate. Community constitutions can require disclosures, time-limited mandates, consultation periods, or separate approval thresholds for high-impact decisions. These are governance safeguards, not cryptographic proofs of independence.

10.8 Preventing Delegation Monopolies

The central danger is not delegation itself but the accumulation of authority that cannot be examined or withdrawn. We therefore propose visible concentration statistics, explicit expiry, bounded scopes, predictable snapshots, and simple revocation. For ordinary community decisions, a dashboard could show the fraction of eligible voting weight entrusted to the largest representative. Because this is a token-weighted system, it should display **weight concentration** as well as counts of delegating addresses; one address is not necessarily one human being.

```
delegate_share(d,p) = represented_eligible_weight(d,p) / total_eligible_weight(p), for
total_eligible_weight(p) > 0
```

The following pseudocode sketches a deterministic one-hop resolver. Reserve is an internal accounting operation, not a transfer of SHB. It must operate on source entitlements frozen at the proposal snapshot.

Algorithm 10.1 — Resolve one-hop delegation for a proposal

```
function ResolveMandates(proposal, frozen_state):
    sources = EligibleSourceBalances(proposal, frozen_state)
    mandates = ValidSignedMandates(frozen_state, proposal.space)
    resolved = EmptyAuthorityLedger()
    for source in SortByCanonicalAddress(sources):
        remaining = sources[source]
        for scope in [PROPOSAL, TOPIC, SPACE]:
            for m in CanonicalMandates(mandates, source, scope):
                require m.delegate != source and m.depth == 1
                assigned = min(m.amount, remaining)
                Reserve(resolved, source, m.delegate, assigned)
                remaining = remaining - assigned
        Reserve(resolved, source, source, remaining)
    require SumSourceWeight(resolved) == Sum(sources)
    return resolved
```

A production implementation would also reject ambiguous overlapping mandates, verify signature domain separation and nonces, and determine replacement semantics before entering the loop. It would allow only one effective vote per source unit and proposal, including where a signer submits a later replacement ballot. Honest validators must calculate the same authority map from the same finalized state.

The protocol should not promise that every collective decision becomes more democratic merely because its voting instructions are cryptographically signed. Delegation offers convenience, specialization, and continuity; it also makes influence easier to concentrate. A responsible system records both sides. The right to appoint a representative matters only when the right to remain independent remains real.

References and Technical Foundations

[38] LiquidFeedback Project, “The History of LiquidFeedback,” describing development beginning in September 2009 and releases in October–November 2009. https://liquidfeedback.com/en/history.html

[39] B. Ford, “A Liquid Perspective on Democratic Choice,” first draft November 2018. https://bford.info/pub/soc/liquid/

[40] A. Kahng, S. Mackenzie and A. D. Procaccia, “Liquid Democracy: An Algorithmic Perspective,” AAAI 2018. https://doi.org/10.1609/aaai.v32i1.11468

PART III / THE ART OF DECIDING

Chapter 11

Private Choice, Verifiable Result

Verification must not be confused with permanent ballot secrecy.

11.1 Public Ledgers and Private Decisions

A public ledger and a private ballot pull in different directions. For a blockchain to be useful, independent observers must be able to verify its records. For a sensitive vote to be trustworthy, participants may need to keep their choices from employers, colleagues or other members. A digital signature establishes who authorized a transaction, but it does not hide the choice. Likewise, a correct tally says nothing about coercion. SHB must address these questions separately rather than treating ledger transparency as a complete privacy solution.

SHB therefore distinguishes **ballot integrity**, **temporary choice concealment**, **voter anonymity** and **coercion resistance**. They are separate guarantees. The initial public-ballot profile prioritizes auditability. A proposed optional commit–reveal profile conceals a choice during a defined commitment period, then makes it visible at disclosure. It is not an anonymous ballot system. Communities must declare their ballot profile before a proposal enters voting, so that no participant mistakes delayed disclosure for permanent privacy.

11.2 Cryptographic Commitments

A commitment binds a participant to a choice without initially revealing it. Let `p` identify a proposal, `a` identify a frozen source voting entitlement, `v` be a canonical choice, and `r` a cryptographically random 256-bit secret generated locally. We propose a domain-separated SHA-256 commitment constructed from an unambiguous binary serialization:

```
C = SHA256(Encode("SHB-BALLOT-v1", chain_id, space_id, proposal_id, source_id,
ballot_version, choice, r))
```

The signer submits `COMMIT(p, source_id, ballot_version, C)` with an authorized digital signature. The transaction contains no plaintext `choice` or `r`. Because the options in a ballot are usually few, hashing the choice without a high-entropy secret would permit trivial guessing. Reusing `r`, backing it up to an untrusted service, or exposing it in wallet telemetry defeats the intended concealment. Wallets must generate new randomness for every commitment, retain it securely until disclosure, and make the risk of losing it clear to the user.

Canonical encoding, cryptographic domain separation, and exact signature rules are consensus requirements: two honest implementations must compute the same commitment bytes. The hash does not certify that a ballot choice is sensible, that the participant is human, or that the voter acted independently.

11.3 Two Windows: Commit and Reveal

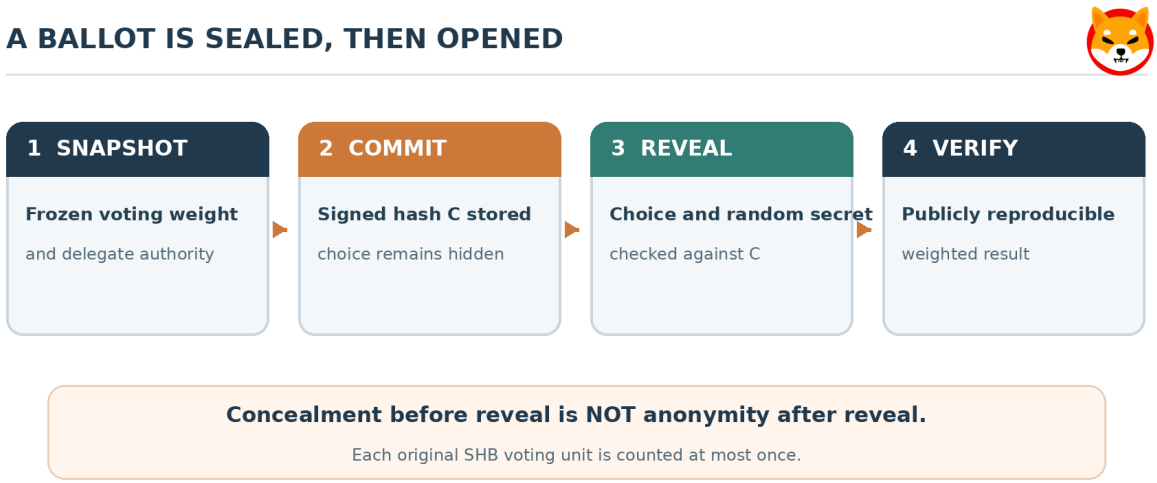
A confidential-until-reveal proposal has three fixed stages: an eligibility snapshot, a commitment window and a disclosure window. A snapshot at finalized height `H_s` freezes SHB voting weight and any one-hop delegation established under Chapter 10. Valid commitments are included only between the declared opening and closing heights. After the commitment window closes, no new vote can be introduced; previously committed signers may reveal their choices until the disclosure deadline. The result is calculated after finality of that deadline.

For illustration, a community may set **1,440 commitment blocks and 720 reveal blocks**. At SHB's nominal **one-minute block interval**, these correspond to approximately 24 and 12 hours, respectively.

Block heights, not a guarantee of wall-clock elapsed time, determine eligibility. The **100,000,000 SHB fixed total supply** is unaffected by ballot creation, commitments or reveals.

A valid `REVEAL` discloses the selected choice and secret `r`; a node recomputes `C` and requires an exact match with the accepted commitment for that source and ballot version. An unrevealed commitment has no effective Yes, No or Abstain choice. SHB would report its weight separately as **committed but not revealed**, rather than silently interpreting it as approval. For the proposed baseline, only successfully revealed eligible weight contributes to the participation calculation; a community choosing another rule must state it before voting begins.

FIGURE 11.1 THE COMMIT–REVEAL VOTING SEQUENCE



Illustrative SHB ballot profile. A disclosed choice is public; commitment alone does not provide permanent anonymity. OT Development Team, February 2020.

11.4 Ballot Confidentiality and Its Limits

Commit–reveal offers a narrow protection: during the first window, an observer seeing the commitment should not be able to infer its choice from the hash alone, assuming high-entropy randomness and no information leaked elsewhere. After disclosure, the ballot is public and may remain associated with its signing address, source entitlement or delegate. Transaction timing, network observation and publicly known wallet ownership can also disclose relationships before reveal. A large, distinctive delegated weight can make a ballot recognizable even without a name attached.

Some established cryptographic approaches, including mix networks and homomorphic tallying, provide starting points for stronger secrecy. They are not part of the proposed SHB baseline. Their use would require a precise threat model, proof verification rules, procedures for compromised trustees, and independent security analysis. A reference to zero-knowledge proofs alone does not establish ballot anonymity or resistance to coercion.

11.5 Verifying the Result Without Trusting a Counter

The purpose of a public audit is to let an observer reproduce the protocol's conclusion using finalized data. Each accepted commitment must be included in the ledger; each effective reveal must match an earlier commitment; the voting authority must trace to a frozen eligible source; and the same voting units must not appear twice. Anyone running a full node can reconstruct the accepted set and calculate the resulting weighted tally. Merkle inclusion proofs can help a lighter client check that specific records were included, although inclusion alone does not prove that the entire election was conducted fairly.

Three questions should be asked separately: **Was my instruction recorded? Was the recorded instruction counted under the published rule? Does the announced outcome match the verifiable**

record? Commit–reveal can help answer these questions but cannot conceal the choice after publication. Stronger end-to-end verifiable schemes exist in voting research; adapting them to SHB would be a separate, carefully tested protocol extension rather than a marketing promise.

11.6 Duplicate Votes and Frozen Authority

A source unit of SHB must have one effective ballot path per proposal. The finalized snapshot fixes eligible balances and the effective one-hop delegation map. The authorized signer may be the holder or a delegate, but two signatures claiming the same unit do not create two votes. Each commitment is indexed by `(proposal_id, source_entitlement_id)`; a defined replacement rule may accept a newer authorized version before the commitment deadline while invalidating the previous version. Once disclosure begins, the accepted commitment cannot be replaced.

```
Sum(effective_revealed_weight_by_source) <= Sum(frozen_eligible_source_weight)
```

The pseudocode below specifies the essential checks. `AuthorizedAtSnapshot` resolves the original source entitlement and delegate authority from finalized state, rather than accepting a self-declared wallet balance. Canonical tie-breaking, signature domain separation and replacement ordering are protocol details that must be specified and tested before deployment.

Algorithm 11.1 — Verify a revealed ballot

```
function VerifyReveal(proposal, source, version, choice, secret, signer, state):
    require state.height in proposal.reveal_window
    require AuthorizedAtSnapshot(signer, source, proposal.snapshot)
    commit = AcceptedCommitment(proposal.id, source)
    require commit != NONE and commit.version == version
    require not AlreadyRevealed(proposal.id, source)
    digest = SHA256(Encode("SHB-BALLOT-v1", state.chain_id,
                      proposal.space_id, proposal.id, source,
                      version, choice, secret))
    require digest == commit.digest
    MarkRevealed(proposal.id, source, choice)
    CountExactlyOnce(source, FrozenWeight(source, proposal.snapshot))
    return ACCEPT
```

11.7 Coercion, Vote Buying and Non-Disclosure

A permanent public receipt can make it easier for another person to demand proof of a vote. Commit–reveal does not eliminate that threat; its disclosure stage may make proof particularly easy. It also permits strategic withholding: a voter who dislikes the apparent direction of a contest can decline to disclose, potentially affecting turnout or quorum. SHB must display both disclosed participation and undisclosed commitments so that observers can identify this difference. No threshold rule should be rewritten after a vote merely to accommodate a disappointing disclosure rate.

Secret ballots in high-stakes elections require safeguards beyond software: independent observers, resistance to coercion, accessible recovery processes and clearly defined remedies for compromised devices or credentials. SHB makes no claim to solve these problems through a hash function. Its intended first applications are voluntary community decisions whose organizers can select a transparent or delayed-disclosure profile with an understanding of the trade-offs.

11.8 The Practical Privacy Standard

Privacy claims should be attached to specific mechanisms, not to the word *blockchain*. The proposed initial SHB design supports publicly signed ballots and, following testing, an optional sealed-then-revealed ballot. Both use the same frozen source weights, single-hop delegation limits and deterministic result verification. Neither mode makes one address equal one human being. Neither promises that a voter cannot be identified. And neither converts SHB into a payment instrument: the token remains a unit of governance influence, within a fixed network supply of 100 million.

Ballot property	Public signed ballot	Proposed commit–reveal ballot
Verifiable voting authority	Yes, from the snapshot	Yes, from the snapshot

Ballot property	Public signed ballot	Proposed commit–reveal ballot
Choice visible during first window	Yes	No, if secret is protected
Choice visible after finalization	Yes	Yes, after successful reveal
Permanent voter anonymity	Not provided	Not provided
Coercion resistance	Not provided	Not provided
Independent deterministic tally	Yes	Yes, for valid reveals

A mature community should be able to say not only which result won, but also which privacy properties its process actually delivered. SHB's responsibility is to make that statement testable, even when the answer is less ambitious than the community might wish.

References and Technical Foundations

[41] D. Chaum, “Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms,” Communications of the ACM, vol. 24, no. 2, 1981. https://doi.org/10.1145/358549.358563

[42] B. Adida, “Helios: Web-based Open-Audit Voting,” 17th USENIX Security Symposium, 2008. https://www.usenix.org/legacy/event/sec08/tech/full_papers/adida/adida.pdf

[43] J. Benaloh et al., “End-to-End Verifiability,” arXiv:1504.03778, 2015. https://arxiv.org/abs/1504.03778

[44] B. Laurie, A. Langley and E. Kasper, “Certificate Transparency,” RFC 6962, June 2013. https://www.rfc-editor.org/rfc/rfc6962

PART III / THE ART OF DECIDING

Chapter 12

Rules That Govern the Rules

A constitution is an agreement about who may change agreements.

12.1 Community Constitutions

The rules of a community are already there, even when nobody has written them down. Someone decides which proposals are admissible, who is allowed to vote and what counts as a majority. Someone else may be responsible for acting on the result. In an association these arrangements are usually found in bylaws or established practice. SHB allows the parts that can be stated precisely to be recorded and checked as a community constitution.

The proposed SHB **governance space** has a constitution record containing a `space_id`, a version number, an activation height, a canonical content hash and a structured set of rules. Typical fields include membership conditions, ballot types, eligible voting sources, sponsorship thresholds, quorum requirements, approval thresholds, delegation limits, discussion periods, amendment procedures and execution permissions. A text explanation may be retained using distributed storage, while the rule fields required by validators must be explicit and deterministic. A hash of a PDF does not, by itself, turn prose into executable law.

SHB would treat a constitution as a *versioned agreement*, not a document that an administrator can silently overwrite. Any update becomes a new record linked to its predecessor. The history remains available even when a community chooses a different model of governance. This makes a modest but important promise: a participant should be able to determine which rules applied when a particular decision was made.

12.2 Configurable Rules and the Scope of Authority

Communities need different rules: an engineering group may use simple majority voting, while a professional association may require verified membership and longer review. SHB permits those differences on one blockchain.

That flexibility must have boundaries. **Local constitutions govern local decisions. They do not override global consensus rules.** No community ballot may mint SHB beyond the fixed **100,000,000-token supply**, alter the network's nominal **one-minute block target**, confiscate another space's assets or grant itself validator authority. Such matters belong to network-level protocol specifications and their separately defined upgrade process. A decision concerning the direction of a community must not be confused with an instruction to change every node's understanding of the ledger.

To keep execution reproducible, each proposal references a `constitution_version` and `constitution_hash`. Those fields are frozen before its eligibility snapshot. A subsequent constitutional amendment does not revise the ballot rules, voting weights or validity of an already opened proposal. Only future proposals use the new version once it has become active. An explicitly approved migration may move unstarted drafts to a newer constitution; completed votes are never retroactively recalculated.

12.3 Amending the Constitution

A constitution must be amendable but not casually rewritten. The draft specification provides for stronger admission and voting requirements for amendments. Each candidate identifies changed provisions, supplies complete replacement rules, lists revoked permissions and publishes its hash before voting.

For an initial illustrative profile, constitutional amendments require **30% participation of eligible voting weight**, a **two-thirds approval threshold among Yes and No weight**, and a **one-day**

activation delay after finalization. Abstention counts toward participation but not toward the Yes/No approval fraction. These percentages are proposed community defaults, not universal SHB consensus constants. The actual constitution must disclose its thresholds, and amendment attempts are judged according to the constitution *already in force*, not the one they seek to introduce.

```
Q = (Y + N + A) / E ;   approval = Y / (Y + N),   provided E > 0 and (Y + N) > 0
```

Here `E` is eligible SHB voting weight at the snapshot, `Y` is the effective Yes weight, `N` the No weight and `A` the explicit Abstain weight. Under the illustrative amendment profile, acceptance requires `Q >= 0.30` and `3Y >= 2(Y + N)`. Equality counts as meeting the two-thirds threshold. Empty decisive ballots fail even if every eligible source abstains. All values are integer token units; consensus code should use exact integer comparisons rather than floating-point percentages.

Consider a community with `E = 2,000,000 SHB`. Its amendment receives `Y = 410,000`, `N = 160,000` and `A = 90,000`. Participation is `660,000 / 2,000,000 = 33%`. The Yes fraction among decisive ballots is about `71.93%`. Both illustrative conditions are satisfied. The proposal is **accepted for delayed activation**, not immediately substituted into an active vote. A second rule check must ensure the proposed changes remain within the authority of the governance space.

12.4 Protected Provisions and Constitutional Boundaries

Some clauses should be harder to remove than they are to create. We envisage optional protected provisions for the amendment procedure itself, for the rights of a minority to submit an objection, and for restrictions on emergency authority. A constitution may require a higher threshold for those provisions or a second ratification vote after a review interval. This is preferable to having an ordinary majority remove every safeguard in a single transaction.

Signatures cannot prove fairness, but nodes can enforce concrete constraints: minimum approval thresholds, nonzero review delays, and boundaries inherited from parent governance spaces. An invalid candidate must fail deterministically.

The most difficult case is an amendment that changes the amendment rules themselves. For that reason, SHB's proposed baseline evaluates a candidate using the existing rules and requires a delayed activation height. Once active, the new procedure governs only later amendment attempts. A community may choose stronger entrenchment, but no software rule can make a constitution immune to all social disagreement, coordinated abandonment or a network-level software fork.

12.5 Emergency Procedures Without Permanent Emergency Power

A decentralized system may still face compromised keys, malicious proposals or the discovery of a serious implementation defect. A community needs a way to slow down harmful actions without granting a small committee unlimited control. The proposed SHB emergency function is therefore narrow: an authorized multi-signature group may temporarily **pause the execution of specified governance actions** within its own space, publish a reason and trigger mandatory review. It may not create voting weight, change ballots already finalized or rewrite ledger history.

Emergency roles must be named in the existing constitution, constrained by a maximum pause duration and accompanied by a public record of the signers and the affected action identifiers. Extensions require a fresh vote or a separate, stricter authorization path. Expiration is automatic at a declared finalized block height. If the system cannot reach finality during a network outage, no rule should pretend the pause expired according to an unreliable wall clock; finalized height remains the reference.

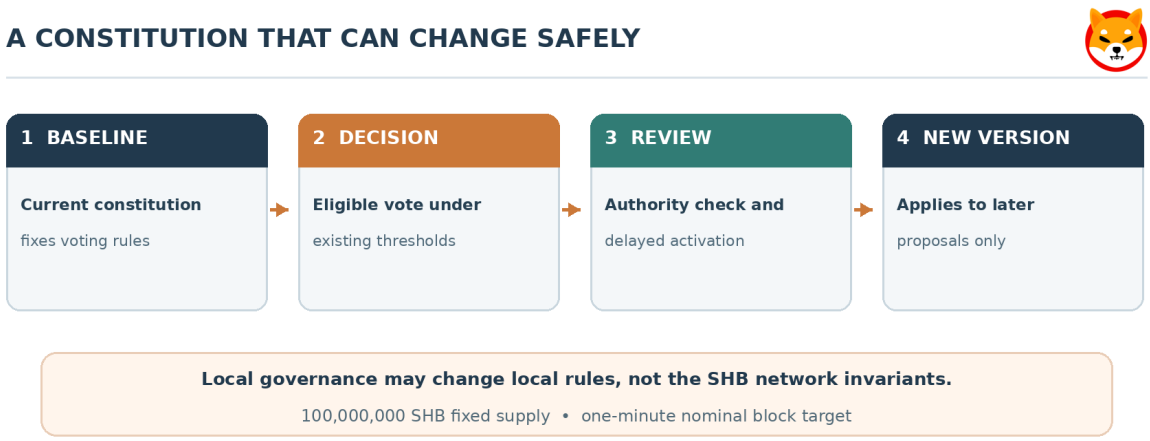
Pauses can also be abused to block valid outcomes. The design accordingly provides for public reasons, appeals and limits against repeated pauses that circumvent the permitted duration. These safeguards require testing.

12.6 Delayed Execution and Constitutional Versioning

The period between approving an amendment and activating it is a safety feature, not dead time. Wallet developers can inspect the change, participants can read its consequences and validators can verify that the candidate uses valid rule fields. A timelock also provides time to detect an incorrect configuration or a conflict with protected provisions.

At the nominal SHB target of one minute per block, a delay of `1,440` blocks corresponds to approximately one day. The exact activation condition is **finalized chain height**, not a promise of twenty-four hours of uninterrupted operation. An accepted amendment records the proposed version hash and an `activation_height`; when the chain reaches that height, each validating node applies the same transition. Pending proposals remain bound to their earlier version.

FIGURE 12.1 CONSTITUTIONAL AMENDMENT LIFECYCLE



A proposed SHB governance-space constitution changes only after a verifiable vote, authority checks and a declared activation height. OT Development Team, February 2020.

Algorithm 12.1 — Validate and stage a constitutional amendment

```
function StageAmendment(space, proposal, finalized_state):
    old = ConstitutionAt(proposal.snapshot_height, space)
    require proposal.kind == CONSTITUTION_AMENDMENT
    require proposal.constitution_hash == Hash(old)
    require Finalized(proposal.tally_height)
    E, Y, N, A = FrozenEffectiveTally(proposal)
    require E > 0 and (Y + N) > 0
    require 10 * (Y + N + A) >= 3 * E
    require 3 * Y >= 2 * (Y + N)
    candidate = DecodeCanonicalRules(proposal.payload)
    require WithinSpaceAuthority(candidate, space)
    require ProtectedClausesSatisfied(old, candidate)
    require NoRetroactiveBallotChanges(candidate)
    h = max(finalized_state.height + 1440, proposal.min_activation_height)
    return StageVersion(space, Hash(candidate), h)
```

The code illustrates a specific two-thirds, 30%-quorum profile rather than hardwiring those values into every governance space. A production implementation would read threshold parameters from the old constitution, enforce validity ranges and reject arithmetic overflow. Staging is not activation; activation still requires the specified finalized height and a valid, uniquely identified candidate version.

12.7 Disputes, Appeals and the Evidence Record

No voting protocol can determine whether a witness is truthful, whether a delegated representative acted ethically or whether a decision was wise. Such disputes need human judgment. SHB can make the process inspectable: it may hold the evidence hashes, deadlines, appointed reviewers, signed findings and the resulting authorized instructions. The documents themselves may be stored off-chain, provided that integrity and availability assumptions are stated.

Appeals may go to elected reviewers, independent delegates or an external process accepted by the community. Their powers must be defined in advance: they may delay execution or order a new ballot but never alter an immutable tally silently. Each remedy is recorded as a new event. For external facts, SHB records *who attested to them*, not a fictitious proof of truth.

Decision class	Typical authority	Safeguard
Ordinary proposal	Eligible SHB voters	Published quorum and majority
Constitutional amendment	Eligible SHB voters under the existing constitution	Higher threshold and activation delay
Temporary execution pause	Pre-authorized, limited multi-signature role	Expiration, reasons and review
Local dispute remedy	Constitutionally appointed reviewers	Narrow remedies and public record
Network protocol upgrade	Separate network-level process	Validator/client compatibility and explicit adoption

12.8 Code, Legitimacy and the Right to Leave

A constitution is only useful if those expected to obey it can understand its consequences. SHB's proposed design therefore distinguishes a *machine-checkable procedure* from a *legitimate collective decision*. A ledger can establish that a valid majority approved a rule under a particular constitution. It cannot establish that the eligible group was appropriately defined, that all participants had adequate information or that dissenters were treated fairly.

Members should be able to export a community record and move to a different governance space. This matters when token holdings concentrate authority. Cryptographic finality preserves disagreement; it cannot force agreement.

The central design test is straightforward. When a community changes its rules, can an independent observer identify the prior constitution, the electorate, the accepted ballot, the delay and the exact moment the new rules became effective? If SHB can answer those questions reliably, it will have achieved something more practical than a slogan about decentralization: it will have given a community a verifiable memory of how it governs itself.

References and Technical Foundations

[45] L. M. Goodman, “Tezos — a Self-Amending Crypto-Ledger,” white paper, September 2014. https://tezos.com/whitepaper.pdf

[46] S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,” RFC 2119, March 1997. https://www.rfc-editor.org/rfc/rfc2119

[47] E. Barker, “Recommendation for Key Management, Part 1: General,” NIST SP 800-57 Part 1, Revision 4, January 2016. https://doi.org/10.6028/NIST.SP.800-57pt1r4

[48] E. Ostrom, Governing the Commons: The Evolution of Institutions for Collective Action, Cambridge University Press, 1990.

PART IV / BEYOND THE BALLOT

Chapter 13

Reputation Beyond Ownership

Contribution can be recognized without turning credibility into a second currency.

13.1 Participation Records

A token balance answers one question very precisely: how much SHB is controlled by an eligible address at a particular block height? It does not tell us whether the holder has read a proposal, maintained community software, fulfilled a promise or helped others understand a difficult decision. A governance system that treats these facts as equivalent will mistake ownership for contribution. We have therefore included a second, deliberately separate set of records: evidence of participation and community recognition. These records should inform deliberation and certain eligibility rules without silently changing the baseline of **one eligible SHB, one unit of voting influence**.

The simplest participation record is an event already verified by the ledger: a valid proposal, a ballot commitment, a disclosed ballot, an accepted amendment or an authorized delegation. A governance space can index these events by address and proposal identifier without issuing any new asset. A record proves that a signed protocol action occurred; it does not establish sincerity, competence or good faith. Voting often may indicate engagement, but frequent voting is not necessarily better judgment. SHB should publish observable activity as activity, rather than presenting it as a universal measure of character.

The protocol is designed for a fixed supply of **100,000,000 SHB** and a nominal block interval of **60 seconds**. Neither parameter is affected by participation accounting. A record may be added at a finalized block, but it does not create or destroy tokens. Where governance rules use activity during a period, the period should be expressed in finalized block heights, with the understanding that chain time may deviate from nominal wall-clock time during outages.

13.2 Contribution-Based Recognition

Many valuable contributions happen outside the ledger. A programmer repairs a security defect; a translator prepares a governance proposal for a second language; a mediator resolves a disagreement; a reviewer detects an incorrect assumption in a technical specification. Recording every detail permanently on-chain would be expensive and often undesirable. The SHB architecture provides for **contribution attestations**: short signed statements that identify the contributor, the category of work, the relevant governance space, the issuer and a hash of supporting evidence.

An attestation is not a reward token and cannot be spent. The proposal is to treat it as a record of what a named issuer claims to have verified. Its credibility comes from the issuer's recognized authority and the community's review process, not from the existence of a signature alone. A signed false statement remains false. Communities should require a documented method for challenging an attestation, recording its withdrawal and correcting mistaken attribution.

The first implementation need not calculate a single numerical reputation score. A participant might be recognized as a documentation contributor, a successful maintainer or a completed-task reviewer, with the supporting records available for inspection when appropriate. Category-specific recognition gives communities more useful information than one score that invites comparison between unrelated forms of work.

13.3 Community Credentials

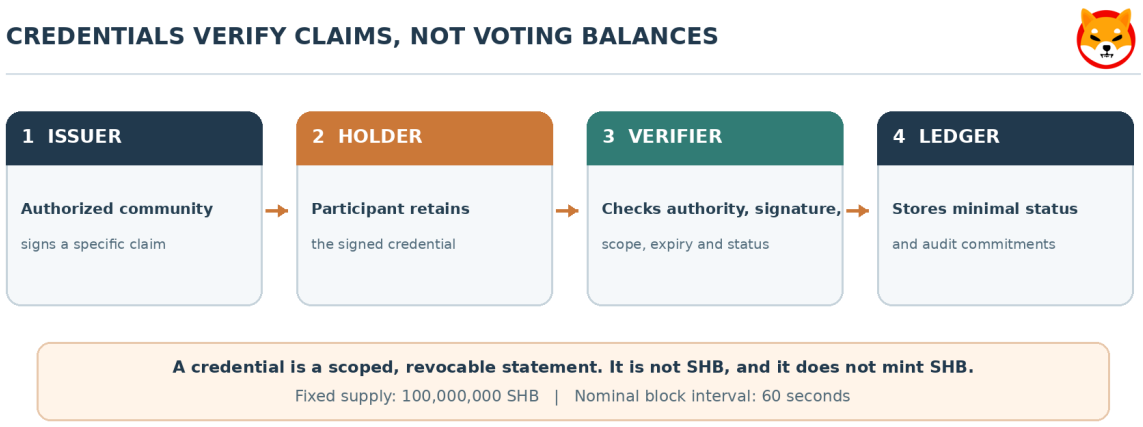
A **community credential** is a signed assertion about an address or member within a specific governance space. Examples include membership, completion of a training requirement, election to a

temporary office, authorization to review code or confirmation that a contributor completed an agreed task. In the SHB design, a credential contains a `credential_id`, `space_id`, `subject_key`, `issuer_key`, `claim_type`, `evidence_hash`, `issued_height`, optional `expiry_height`, and a revocation reference. Its canonical serialized content is signed by the issuer.

The credential can be held off-chain by its subject. The ledger need only retain minimal issuer authorizations, validity or revocation commitments and, where necessary, a hash anchoring the issued claim. This reduces disclosure of personal information and avoids treating a public blockchain as a permanent personnel file. An independent verifier checks that the issuer was authorized when the credential was issued, that the signature is valid, that the credential concerns the required governance space and that it has not expired or been revoked at the relevant finalized height.

The **W3C Verifiable Credentials Data Model 1.0**, published in November 2019, provides a contemporary reference for interoperable digital claims. SHB may draw on that model without assuming that every wallet already supports it. Decentralized identifier proposals were also under active development by February 2020, but should not be described as a completed global standard. The initial SHB implementation can rely on ordinary public keys and explicit issuer registries.

FIGURE 13.1 THE SHB CREDENTIAL VERIFICATION PATH



A signed credential is checked against authorized issuers, valid time intervals and revocation commitments. The design introduces no additional token supply or automatic voting multiplier. OT Development Team, February 2020.

13.4 Expertise and Verifiable Competence

Expertise is useful when a community faces a technical question, but expertise should not be confused with unlimited authority. We propose **subject-specific qualification records**, such as security-review eligibility, translation verification or responsibility for maintaining a software component. A governance space may define qualifications for *performing a task* or *endorsing an assessment*. Such qualifications should not automatically grant extra SHB voting weight.

A practical example is a proposal to change validator software. Every eligible token holder retains the governance rights specified in the constitution. Separately, the community may require two signed technical reviews from qualified maintainers before the proposal advances to a ballot. Reviewers attest to the scope of their examination and the version of the code they inspected; they do not certify that the software is free of defects. Their reports may disagree, and both records should remain accessible.

Qualifications should be time-limited and subject to renewal where knowledge changes rapidly. The community must specify whether eligibility is based on demonstrated work, examination, peer review or an external credential. No fully reliable on-chain test of expertise exists. SHB can enforce the announced process and preserve the signed evidence, but it cannot infer competence from a blockchain address.

13.5 Reputation Independent of Token Balances

The recommended baseline keeps three variables separate: token voting weight, verified eligibility for specialized functions, and optional reputation indicators. Let `B(a,h)` denote an address's eligible SHB balance at snapshot height `h`; let `C(a,s,h)` denote valid credentials for address `a` in governance space `s`; and let `R(a,s,h)` denote an optional, space-defined reputation signal. None is substituted for another without a publicly approved constitutional rule.

```
VotingWeight(a, p) = EligibleSHB(a, SnapshotHeight(p))

TaskEligible(a, role, s, h) = VerifyCredentials(C(a,s,h), role, Rules(s,h))
```

One possible experimental indicator sums independently reviewed contributions within a declared time window. Communities might assign small weights to categories and gradually discount older observations, but the resulting value must be explicitly labeled a *local indicator*, never an objective measure of trustworthiness. It should not determine access to basic voting rights. In particular, wealth must not buy reputation through a mechanical conversion of SHB balances, and reputation records must not expand the fixed token supply.

Consider Alice and Bob, who each hold 500 eligible SHB at the snapshot. Alice has reviewed three proposals; Bob has helped repair a serious bug. Each still holds 500 units of baseline voting weight. If the community requires verified maintenance experience to join a technical review panel, Bob may qualify because of an attestation, not because his vote is worth more. The distinction preserves transparency while acknowledging that different responsibilities require different evidence.

13.6 Conflicts of Interest, Appeals and Revocation

Records of recognition can become instruments of favoritism. A committee might attest to its own members' achievements, refuse to recognize outsiders or revoke credentials from dissenters. To make these abuses visible, The draft specification provides for explicit issuer lists, signed decision records, declared conflicts of interest and appeal procedures defined in each governance-space constitution.

Algorithm 13.1 — Verify a scoped community credential

```
function CredentialEligible(credential, role, space, snapshot_height):
    require credential.space_id == space.id
    require credential.claim_type in RolesAllowedFor(role, space)
    require VerifySignature(credential.issuer_key, credential.payload_hash,
credential.signature)
    require IssuerAuthorizedAt(credential.issuer_key, credential.issued_height, space)
    require credential.issued_height <= snapshot_height
    if credential.expiry_height exists:
        require snapshot_height < credential.expiry_height
    require not RevokedAt(credential.credential_id, snapshot_height, space)
    require SubjectControlsKey(credential.subject_key)
    return true
```

This simplified algorithm assumes deterministic encodings and an auditable issuer registry. A complete implementation must define key rotation, compromised issuer keys, historical revocation semantics, duplicate credentials and the precise relation between a credential's issuance height and its first usable block. In practice, `SubjectControlsKey` requires a challenge-response signature rather than a bare assertion. Where sanctions are disputed, revocation should be recorded as a new signed event, with the prior record retained for audit.

A participant should have a way to challenge inaccurate records and to present a reply. An appeal does not erase the historic fact that an issuer made a statement; it can alter the statement's current validity and append a correction. The process must also distinguish *withdrawal of authorization* from *proof of misconduct*. The former may be routine and time-limited; the latter is a serious accusation that software cannot adjudicate on its own.

13.7 The Dangers of Permanent Reputation

Reputation systems are especially vulnerable to the fact that blockchains remember. A permanent public record of every mistake could discourage participation, expose private relationships and make

past disputes impossible to leave behind. An immutable timestamp may prove that a claim was published; it cannot justify publishing the claim forever. The baseline SHB design therefore favors **minimal on-chain references**, off-chain evidence, controlled disclosure, expiry and contestable recognition.

Mechanism	Intended use	Boundary or safeguard
Participation event	Verifiable protocol activity	Not treated as moral reputation
Signed contribution attestation	Recognition of completed work	Issuer accountability and appeal
Scoped competence credential	Eligibility for specialized tasks	Expiration, review and revocation
Optional local reputation indicator	Helping members evaluate contribution	Never a universal score or automatic vote multiplier
Credential status commitment	Check current or historical validity	Minimal public information; off-chain evidence

No credential design can completely prevent collusion, fake identities or transfer of control over a private key. A non-transferable claim can still be misused if somebody transfers the key that controls it. Communities requiring one-person-one-role participation must assess their own identification processes and the privacy cost of using them. SHB does not claim that a signature proves an individual is unique.

European communities must also consider the General Data Protection Regulation, applicable since May 2018. The principles of purpose limitation, data minimization and rights relating to personal information are relevant even if the underlying ledger is technically immutable. SHB should avoid personal data in consensus records wherever possible and leave the actual legal obligations to be assessed for each deployment.

This chapter is intended not to create a permanent ranking of people. It is to let a community distinguish between **what an address owns, what a participant has done, and what a trusted issuer is prepared to verify**. These are different kinds of evidence. A governance protocol earns credibility by keeping them different.

References and Technical Foundations

[49] W3C, Verifiable Credentials Data Model 1.0, Recommendation, 19 November 2019. https://www.w3.org/TR/2019/REC-vc-data-model-20191119/

[50] D. Reed et al., Decentralized Identifiers (DIDs) v1.0, W3C Working Draft, 2019. Referenced only as work in progress as of February 2020.

[51] European Union, Regulation (EU) 2016/679, General Data Protection Regulation, applicable from 25 May 2018. https://eur-lex.europa.eu/eli/reg/2016/679/oj

PART IV / BEYOND THE BALLOT

Chapter 14

Communities Within Communities

A shared record does not require a shared constitution.

14.1 Governance Spaces

As communities grow, they divide work among smaller groups. A software project forms a security group; an association elects regional representatives; a cooperative creates a committee to review new members. These groups need authority to decide some matters without being able to decide everything. SHB calls the unit in which this authority is defined a **governance space**. Each space has a permanent identifier, a constitution or rule reference, an authorized membership or electorate definition, and its own proposal and ballot records. The shared blockchain orders these records; it does not require every community to adopt the same constitution.

A space identifier should be derived from a finalized creation transaction and the network identifier, so that two organizations cannot accidentally receive the same identifier. Its canonical state includes `space_id`, `parent_space_id` if applicable, `constitution_hash`, `constitution_version`, `status`, and a list of assigned powers. Large documents may remain outside the chain; the ledger records hashes and the minimum information needed for deterministic validation. A creation transaction must specify whether the space is independent or derives its mandate from an existing space.

A space is not a new blockchain and does not create a new SHB currency. The network retains its **fixed 100,000,000 SHB supply** and **nominal one-minute block interval**. A participant may take part in several spaces, but votes are counted separately under each space's eligibility rules and snapshot height. SHB held at one address cannot be counted twice *within the same proposal*, even when that address belongs to multiple communities.

14.2 Subcommittees and Working Groups

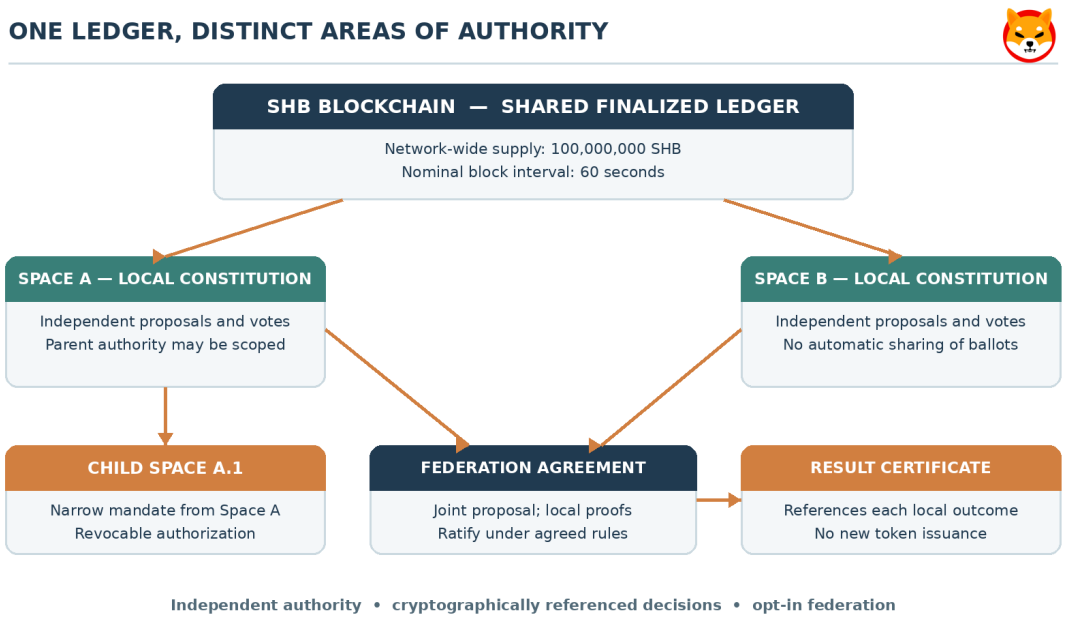
The most useful delegation is often limited. A parent community may authorize a technical committee to approve documentation updates while reserving validator-policy changes for a full token-holder vote. The committee is a **child space** whose powers are explicitly narrower than the parent's. Its creation record names the parent, the permitted proposal categories, any expiry height, the method for selecting its members, and the procedure for recalling the mandate. A child may create another child only if the parent constitution specifically permits further delegation.

The decisive rule is authority containment. An instruction is valid only if its type lies inside the child's authorized scope *and* does not violate the current parent constitution or network-wide protocol rules. A subcommittee may attach evidence, draft a recommendation, or certify completion of a task. It cannot unilaterally alter the global token supply, change the network's target block interval, or rewrite another community's records. Revocation closes the child's authority to submit new binding actions; historic votes remain verifiable.

Authority should have an auditable origin. If a committee claims the right to approve a decision, any independent node should be able to follow the finalized chain of grants back to a valid parent authorization. Administrators can disagree about the wisdom of a grant without disagreeing about whether it exists.

```
ValidAuthority(c, a, h) = GrantedByParent(c, a, h) AND WithinScope(c, a) AND NOT
Revoked(c, h)
```

FIGURE 14.1 COMMUNITY SPACES AND FEDERATION



Different governance spaces share block ordering while retaining separate constitutions. Subspaces exercise scoped authority; federation agreements reference local decisions rather than overriding them. OT Development Team, February 2020.

14.3 Independent Decision-Making Structures

Independence is more than a separate page in a user interface. It means that one space can define its own quorum, ballot periods, qualified-majority rules, proposal eligibility and representative mandates without silently changing those of its neighbors. A proposal submitted to a space is evaluated against a particular, finalized constitution version. The version used to admit the proposal remains identifiable even if the community later amends its rules. Chapter 12 describes how constitutional changes should be staged and activated.

For example, an educational community might permit any member with a valid credential to introduce a curriculum proposal, then count voting power by eligible SHB balances. An open-source project might permit proposals from code maintainers but reserve the final vote to its SHB holders. These are different procedures built on the same ledger. A credential can grant the right to introduce or review a proposal without automatically minting tokens or multiplying votes.

For reproducibility, every ballot record must identify its `space_id`, `proposal_id`, `snapshot_height` and `constitution_hash`. The result is computed from the eligible voting weight for that particular space and snapshot. The existence of another governance space does not change those numbers. If two spaces include the same token holder, that person may cast a valid vote in each separately constituted election; this is participation in two decisions, not duplication of a ballot.

14.4 Federated Communities

Federation is a way for independent communities to cooperate without surrendering all their authority to a common administrator. The baseline SHB design proposes a **federation agreement**: a versioned record identifying participating spaces, shared objectives, authorized coordinators, the categories of joint decisions and an exit procedure. It is not an extra sovereign layer above the participants. Each member retains its own constitution and can limit the powers it commits to the agreement.

A federation may maintain a shared document library, coordinate a technical standard, exchange signed endorsements or schedule parallel votes. It may also appoint a temporary coordinator to publish notices and gather proofs. The coordinator's role is procedural: collecting verifiable records, not

replacing local votes. In a truly joint decision, each member space must authorize the action through its own rules unless it has previously granted a bounded mandate for that exact category.

The ledger can store the federation agreement hash and each member's signed accession record. An exit should not retroactively invalidate decisions already finalized under a lawful mandate, but it can prevent new commitments after the specified exit height. This distinction reduces the risk that a departing member makes completed shared work impossible to audit.

14.5 Inter-Community Proposals

A joint proposal should not be approved merely because a large community can outvote a small one in a combined pool. For the first SHB federation model, each participating space produces an independently verified local result. The joint proposal becomes accepted only when the agreed ratification rule over those local results is satisfied. A straightforward rule is unanimous approval among participating spaces; a federation may instead adopt a qualified count of spaces, provided that rule was accepted in advance.

```
JointApproved(P) = AND [ LocalApproved(s, P) for every mandatory member space s ]
```

Consider a proposed shared code repository. Space A has 1,200,000 eligible SHB at its snapshot; 450,000 vote yes, 250,000 vote no and 20,000 abstain. Space B has 800,000 eligible SHB; 250,000 vote yes, 125,000 vote no and 25,000 abstain. With a 30% participation quorum and 60% approval threshold calculated over decisive votes, A records 60% participation and 64.29% approval; B records 50% participation and 66.67% approval. Both pass locally. Under the unanimous-space rule, the joint proposal passes. The same numeric votes would not authorize a third community that never joined the agreement.

Participating space	Eligible SHB	Turnout incl. abstentions	Yes / decisive votes	Local outcome
A — Developer collective	1,200,000	720,000 (60%)	450,000 / 700,000 (64.29%)	Approved
B — Education community	800,000	400,000 (50%)	250,000 / 375,000 (66.67%)	Approved

The example describes a proposed governance profile, not universal SHB voting thresholds. No SHB is transferred by recording a federation result, and a participant's vote in one space does not automatically count as a vote in another. A federation requiring stronger independence may instead give each community one ratification decision regardless of its token supply, while retaining token-weighted ballots inside each member space.

14.6 Shared Rules and Local Autonomy

The protocol should check **where** a rule applies before asking whether it has been satisfied. Network rules govern the validity and ordering of transactions. Space constitutions govern decisions inside an identified space. Federation agreements govern only the commitments that their members authorized. A local constitution may be stricter than the protocol, but it may not make an otherwise invalid network transaction valid.

To prevent ambiguity, joint proposals should record the hashes of the federation agreement and of each participating local proposal. A joint result certificate then carries each member's finalized decision reference and the agreement version under which those decisions were interpreted. Nodes verify the references, not a coordinator's assertion that “everyone agreed.” If any mandatory proof is missing or still inside an appeal or timelock period, the joint action remains pending.

Algorithm 14.1 — Verify a federated decision

```
function VerifyFederatedDecision(agreement, joint_proposal, finalized_state):
    require agreement.active_at(joint_proposal.creation_height)
    require joint_proposal.type in agreement.authorized_categories
    require joint_proposal.agreement_hash == Hash(agreement.canonical_bytes)
    approvals = 0
    for space_id in agreement.member_spaces:
        proof = joint_proposal.local_result_for(space_id)
        require VerifyFinalizedResult(proof, finalized_state)
        require proof.space_id == space_id
        require proof.joint_proposal_hash == joint_proposal.hash
        require proof.rules_hash == RulesAt(space_id, proof.snapshot_height).hash
        if proof.approved and not proof.pending_appeal:
            approvals += 1
        else if space_id in agreement.mandatory_spaces:
            return PENDING_OR_REJECTED
    require approvals >= agreement.required_space_approvals
    return RATIFIED
```

A production implementation must specify how expired agreements, appeals, abstaining member spaces and incompatible deadlines are handled. `VerifyFinalizedResult` must use the same deterministic state transition rules as the rest of SHB; it must not trust a server-generated screenshot of a ballot result. Time windows should be expressed in finalized heights, not solely in wall-clock timestamps. At a nominal one minute per block, 1,440 blocks correspond to about one day, but network interruptions can extend the elapsed time.

14.7 Collective Coordination Across Organizations

The practical value of federation lies in ordinary work. Several independent projects may agree on a shared technical specification. Regional associations may nominate delegates to a temporary board. Research groups may establish a common ethics review procedure while keeping separate membership policies. SHB can record the mandate, the votes that created it, and the evidence that it was fulfilled. It cannot itself guarantee that a real-world committee will perform its work or that an outside institution will recognize a blockchain vote as legally binding.

This distinction should guide early implementation. An initial test network can support parent-child space registration, scoped permissions, separate ballot tallies and signed federation agreements. Joint execution can follow only after the ratification rules, appeal handling and authority checks have undergone testing. Complex cross-chain voting should remain outside the baseline design: importing a statement from an unrelated blockchain is not the same as verifying that statement under SHB's consensus rules.

Federation should therefore remain an **opt-in arrangement between autonomous communities**. Its success will not be measured by how many groups submit to a central council, but by whether they can cooperate, disagree, revise their mandates and leave with a clear record of what they authorized. A distributed ledger cannot eliminate political disagreement. It can make the boundaries of agreement harder to misrepresent.

References and Technical Foundations

[52] E. Ostrom, Governing the Commons: The Evolution of Institutions for Collective Action, Cambridge University Press, 1990.

[53] W3C, ActivityPub, W3C Recommendation, 23 January 2018. A precedent for federated communication, not a blockchain voting protocol. https://www.w3.org/TR/2018/REC-activitypub-20180123/

[54] J. Benet, IPFS — Content Addressed, Versioned, P2P File System, 2014. https://arxiv.org/abs/1407.3561

[55] W3C, Verifiable Credentials Data Model 1.0, W3C Recommendation, 19 November 2019. https://www.w3.org/TR/2019/REC-vc-data-model-20191119/

PART IV / BEYOND THE BALLOT

Chapter 15

The Archive of Intentions

An institution should be able to explain not only what it decided, but how it arrived there.

15.1 The Value of Institutional Memory

An organization can lose the reasoning behind a decision long before it loses the decision itself. A vote may settle a question for an afternoon, yet months later the people involved can disagree about the exact wording, the electorate, or the conditions attached to their approval. A new committee inherits the result but not the discussion that made it reasonable. We envisage an **archive of intentions**: an ordered record that connects a proposal to the rules under which it was considered, the evidence presented, the votes cast, and the action eventually authorized. The record is not intended to replace human judgment. It should make the path of that judgment inspectable.

The ledger preserves a small, verifiable core. Each event receives a network identifier, a governance-space identifier, an event type, a reference to its predecessor where appropriate, and a cryptographic commitment to its content. A finalized block establishes the event's position in the canonical chain. The same framework can describe a successful proposal, a rejected amendment, an abandoned discussion, or a mandate later found to have been poorly executed. **Rejection and dissent are first-class historical facts**, not debris to be removed after the winner is announced.

SHB's baseline design is intentionally modest. Validators agree on valid records and their order; they do not decide whether the attached argument is persuasive or whether a submitted photograph is genuine. An archive can demonstrate that particular bytes were committed to the network by a particular height. It cannot, by itself, demonstrate that those bytes describe the outside world truthfully. That distinction is essential to any credible record of collective decisions.

15.2 Permanent Proposal Histories

A proposal should have one stable identity and as many explicitly linked versions as its rules permit. Its initial record includes `proposal_id`, `space_id`, `author_address`, `constitution_hash`, `created_height` and a commitment to the submitted text. Each accepted amendment refers to both the proposal and the prior version. Once a voting snapshot has been established, a new amendment cannot silently replace the text under consideration. It must either follow the constitution's permitted procedure or become a separate proposal.

An archival event can be represented by the following conceptual record. The precise field encodings, signature domain and hash function are protocol parameters to be fixed and tested before mainnet deployment.

Field	Meaning
event_id, space_id, proposal_id	Stable identifiers and context
event_type, previous_event_id	Action and authenticated lineage
content_hash, hash_algorithm, schema_version	Commitment and interpretation rules
signer, signature, finalized_height	Attribution and ordering
storage_reference (optional)	Location hint for an external document

The `storage_reference` is deliberately not part of the proof of authenticity. A web address may disappear, and an IPFS peer may stop retaining a file. The proof concerns the committed content hash; availability requires a separate storage policy. A history viewer should distinguish **verified record**,

retrievable attachment and **unverified interpretation** rather than presenting all three as equally certain.

15.3 Cryptographic Document Certification

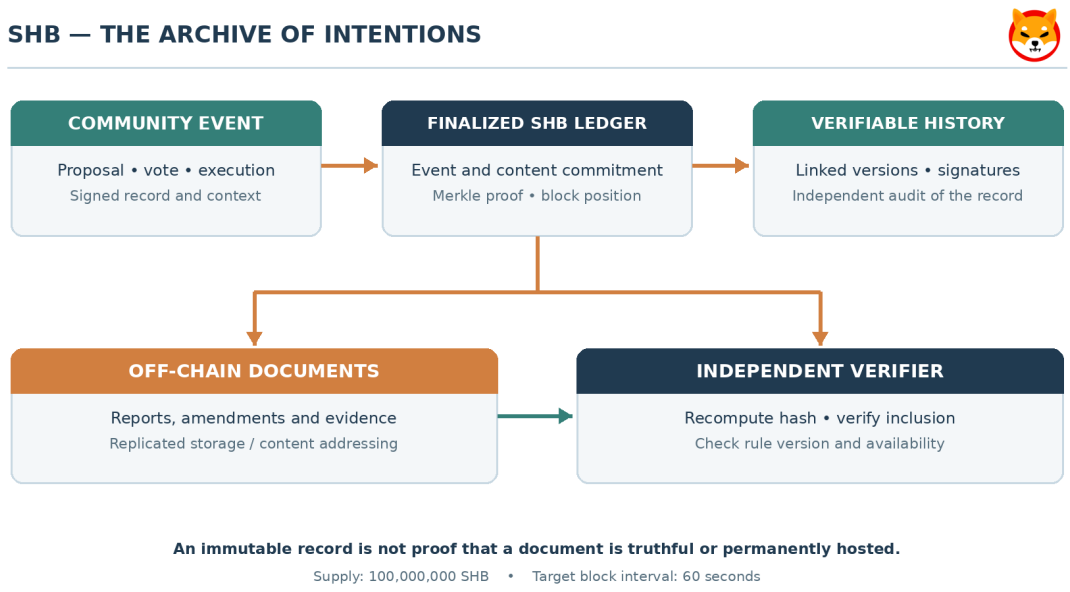
Large reports, meeting notes, design drawings and signed mandates do not belong inside every validator's state database. The SHB architecture provides for storing their bytes in ordinary replicated repositories or a content-addressed system such as IPFS, then recording a digest and sufficient metadata on-chain. To verify a retrieved file, a client applies the declared hashing algorithm to the exact byte sequence and compares the result with the finalized commitment. A changed comma, image or embedded signature changes the digest with overwhelming probability under a secure hash function.

```
document_commitment = H("SHB-DOC-V1" || length || canonical_document_bytes)
```

The prefix separates document commitments from other hash domains, and the length must have an unambiguous encoding. For structured records, canonical serialization is mandatory: the same logical object must not yield two acceptable byte encodings merely because fields were reordered. The first protocol release should specify accepted encodings and the exact algorithms instead of leaving this decision to wallet software.

A Merkle inclusion proof can link an archival event to a finalized block's transaction root without requiring a lightweight client to download every transaction. This is a *proof of inclusion*, not a certificate that the document was authored on the date claimed in its contents. With SHB's nominal one-minute block interval, finalized height is usually a more reproducible ordering reference than a wall-clock date. No network can guarantee that each block will be produced exactly one minute after the last one.

FIGURE 15.1 THE VERIFIABLE DECISION ARCHIVE



Signed community events are ordered on the SHB ledger; supporting documents remain in separately operated storage. An independent client can verify the ledger commitments but must separately assess content and availability. OT Development Team, February 2020.

15.4 Decision Timelines

The archive becomes useful when readers can reconstruct a complete timeline from independent records. We propose a timeline view that places each event under its finalized height: submission, sponsorship, accepted amendments, electorate snapshot, ballot opening, vote commitments where applicable, vote revelation, tally certificate, appeal, timelock expiry and execution report. The interface may add human-readable dates, but the underlying verification uses ordered blocks and explicit event references.

For example, a cooperative submits a safety-policy revision at height 2,000,000. The proposal is frozen at 2,001,440; voting runs for 4,320 blocks and ends at 2,005,760. At a nominal one minute per block these are approximately one day of preparation and three days of voting. Network pauses may extend those periods in calendar time. An inspector later retrieves the adopted policy, checks its content hash, confirms the electorate snapshot and recomputes the final result. The exercise does not require asking the original committee to remember which PDF was on the table.

A disputed result should remain in the timeline as a *disputed result*. The archive appends the challenge, the authority that considered it and the final disposition. It should never rewrite an old event to make subsequent developments look inevitable. Historical corrections are new, linked records carrying their own provenance.

15.5 Recorded Minority Opinions

An accurate institutional record must leave room for people who lost the vote. We therefore propose optional **minority statements** linked to a closed proposal. A statement may explain the grounds of dissent, identify a disputed assumption or request that a promised safeguard be reviewed later. It does not reopen the ballot, increase the author's voting weight or confer an automatic veto. A community constitution can define the submission period, length limits and whether a statement requires a minimum share of the decisive vote.

The archive must not force disclosure of a secret ballot. In the commit–reveal profile described in Chapter 11, an optional statement may be attributable to an address only when its author chooses to sign it publicly. A private ballot and a public explanation remain separate actions. Publishing individual vote choices, identities or personal allegations without an appropriate legal basis would conflict with the privacy goals of the system.

Minority statements serve a practical purpose: decisions are made under uncertainty. An objection that seemed unconvincing during the vote may later identify the cause of a failure. Preserving it alongside the majority rationale gives subsequent reviewers a fuller record than a single green approval indicator.

15.6 Governance Transparency and Independent Audits

A public audit should be possible without privileged access to the organization's administrators. A verifier receives the relevant finalized block headers, archival event records, Merkle inclusion paths and any documents the community makes available. It checks block finality under the SHB validator rules, verifies the event's signatures and version links, then recomputes the document commitments. The verifier can report missing attachments or contradictory claims without rewriting the chain.

Algorithm 15.1 — Verify a proposal archive

```
function VerifyProposalArchive(proposal_id, proof_bundle, trusted_finality):
    require VerifyFinality(proof_bundle.headers, trusted_finality)
    events = SortByFinalizedPosition(proof_bundle.events)
    require events[0].proposal_id == proposal_id
    previous = NONE
    for event in events:
        require event.proposal_id == proposal_id
        require VerifyMerkleInclusion(event, proof_bundle.headers)
        require VerifyAuthorSignature(event)
        require event.previous_event_id == previous or PermittedBranch(event)
        if event.document_bytes is available:
            require DocumentCommitment(event.document_bytes) == event.content_hash
        previous = event.event_id
    return VERIFIED_WITH_AVAILABILITY_REPORT
```

In practice the verifier must also enforce the relevant space constitution, event schema, allowed branch structure and changes in cryptographic algorithms. A complete audit cannot be obtained by checking hashes alone: eligibility, vote counting and authority to execute still depend on the rules preserved in earlier chapters. An archive is most valuable when each of those checks can be performed by software written independently of the original client.

15.7 Distributed Storage, Retention and the Limits of Permanence

No storage scheme is permanent merely because its index is on a blockchain. A community should designate multiple document custodians, export encrypted backups where appropriate, regularly test retrieval and publish availability status. IPFS can locate content by its identifier but does not compel anyone to keep hosting it. The network should continue to verify historical commitments even if an attachment is no longer available; the interface must say clearly when verification of the underlying bytes is impossible.

Privacy is the more difficult boundary. A hash of a sensitive document may still reveal information through guessing attacks, particularly when the document has few possible values. The first SHB archive should therefore keep names, identity documents, private deliberations and other unnecessary personal data **off-chain**. Encryption, access control, retention periods and the possibility of removing off-chain copies must be designed into the hosting layer. By February 2020, Regulation (EU) 2016/679 already imposes data-protection obligations, including rights of rectification and erasure in applicable circumstances; calling a record decentralized does not suspend those obligations.

Cryptographic longevity also requires maintenance. Algorithms may become unsuitable, keys may be compromised and archival formats may become unreadable. A future migration can append a new commitment that points back to the old one, preserving a verifiable chain of custody rather than pretending that no change occurred. SHB should distinguish an immutable **event history** from the mutable availability and lawful retention of its associated content.

The archive consumes network capacity but creates no new SHB. Token supply remains fixed at **100,000,000 SHB**, and the nominal block interval remains **60 seconds**. To keep validation practical, large files are referenced, not reproduced in every block. The proposed archive is thus neither a universal vault nor a machine for declaring truth. It is an accountable record of what a community claimed, considered, chose and subsequently did.

References and Technical Foundations

[56] B. Laurie, A. Langley and E. Kasper, Certificate Transparency, RFC 6962, June 2013. https://www.rfc-editor.org/rfc/rfc6962

[57] J. Benet, IPFS — Content Addressed, Versioned, P2P File System, 2014. https://arxiv.org/abs/1407.3561

[58] C. Adams et al., Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP), RFC 3161, August 2001. https://www.rfc-editor.org/rfc/rfc3161

[59] European Parliament and Council, Regulation (EU) 2016/679, 27 April 2016, notably Articles 16 and 17. https://eur-lex.europa.eu/eli/reg/2016/679/oj

PART IV / BEYOND THE BALLOT

Chapter 16

From Decisions to Actions

An approved proposal is an instruction to act, not evidence that the action has been carried out.

16.1 Voting as the Beginning, Not the Conclusion

A successful ballot tells the community what has been approved. It does not, by itself, repair a road, publish a report or organize an engineering team. Many collective systems become less convincing at precisely this point: an approved resolution exists, yet the work it authorized has no common timetable, responsible party or reliable record of completion. The draft specification provides for an **execution layer** connecting the accepted result to a sequence of limited, observable obligations. Its purpose is to make the difference between *decision*, *authority* and *performance* visible.

The execution layer begins only with a finalized tally and a proposal whose payload was fixed before voting. A successful result creates a mandate record; it does not grant the author unlimited powers. Every mandate belongs to a governance space, inherits the constitution used for its vote and identifies the exact actions that may follow. Execution events appear in the same historical ledger as the proposal, but the protocol distinguishes actions that can be performed deterministically by nodes from work that requires a person or an external organization.

SHB is a governance-oriented blockchain, not an automatic enforcement system for the outside world. The network can establish which instruction was authorized, whether a signer held the correct role and which evidence was submitted afterward. It cannot force a contractor to arrive or guarantee the truth of a completion report. Those boundaries should be stated before any automation is promised.

16.2 Execution Instructions and the Mandate Object

The proposed **mandate object** binds the winning proposal to a controlled action plan. It carries mandate_id, space_id, proposal_id, tally_certificate_hash, constitution_hash, action_root, authorized_roles, activation_height, expiry_height and a state code. A Merkle root may commit to several approved actions without copying their full supporting documents into every block. The precise binary representation and hash algorithms are protocol parameters to be specified before implementation.

Mandate class	Permitted result	Boundary
STATE_CHANGE	Update a permitted governance-space setting	Must pass the active scope and rule checks
ROLE_ASSIGNMENT	Record or revoke a defined local responsibility	Cannot confer network-validator powers
DOCUMENT_PUBLICATION	Commit to an approved text or report hash	Does not guarantee hosting or factual accuracy
EXTERNAL_TASK	Assign a human or organizational obligation	Records a mandate, not physical execution
FOLLOW_UP_VOTE	Schedule a later community decision	Creates no advance approval of that decision

An instruction must identify its expected output and its authorized actor. The instruction must not be interpreted as a blank cheque. Where the proposal omits essential parameters, SHB should reject automatic execution rather than let a delegate improvise an action that was never voted upon. A mandate may describe several steps, but no step can expand the authority originally approved.

16.3 Conditional Actions and Deterministic Execution

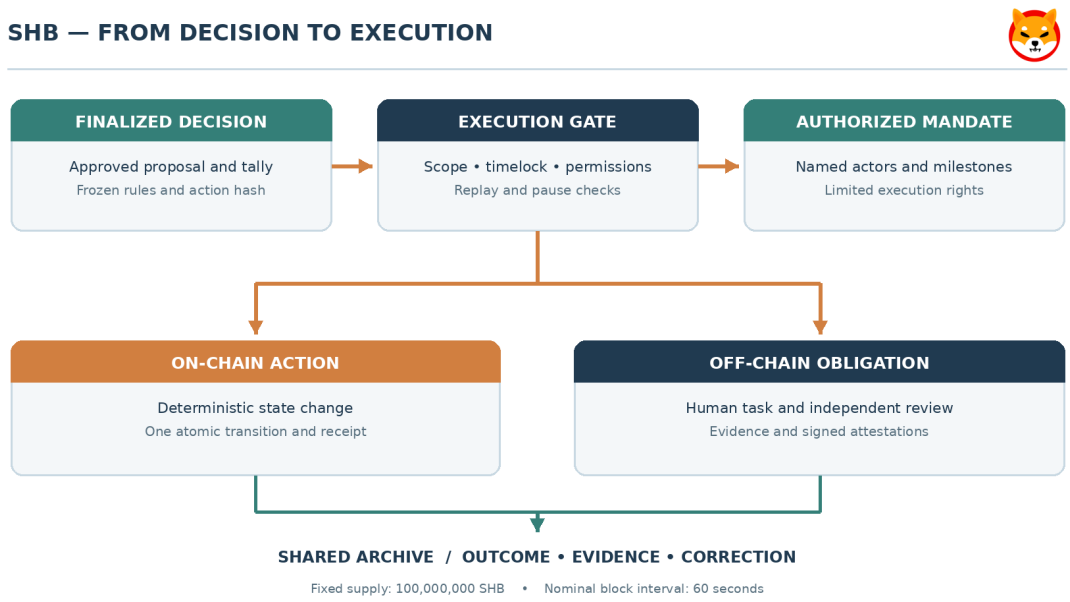
Some decisions are suitable for automatic execution. A governance space may schedule publication of a new constitution version, grant an internal role for a fixed term or activate a local rule after a review delay. These operations can be expressed as deterministic state transitions that every validating node reproduces from the same finalized chain. They must be bounded in execution cost, free from dependence on unverified live web data and protected against replay or repeated application.

A proposed action is executable only when its vote has passed, the relevant delay has elapsed, no authorized emergency pause is active, its constitutional scope is valid and its declared preconditions are satisfied. A unique action identifier prevents the same approved operation from being applied twice. External information may be entered through a signed attestation, but validators can verify the attestation's form and authority, not its real-world truth.

```
executable(a,h) = finalized(a) AND approved(a) AND h >= activation_height(a) AND h < expiry_height(a) AND permitted(a) AND NOT consumed(a) AND NOT paused(a)
```

The expression is a specification shorthand; actual consensus code must define each predicate and the handling of absent or expired fields. The complete state change should be atomic: either it passes every check and commits once, or it leaves the previous state unchanged. Complex unrestricted programs are not required for the first SHB release. A small catalogue of audited native actions would reduce both implementation risk and the possibility that a poorly constructed mandate could threaten the ledger.

FIGURE 16.1 THE SHB MANDATE EXECUTION PIPELINE



A finalized decision enters a bounded authorization gate. Deterministic state changes may execute on-chain; human obligations remain subject to review and evidence. Both outcomes are linked to the permanent archive. OT Development Team, February 2020.

16.4 Milestones and the Difference Between Progress and Proof

A project can be approved long before it is completed. The design accordingly provides for **milestone records** containing a responsible actor, a deliverable description, an expected block-height window and a reference to supporting evidence. Milestones may be PENDING, SUBMITTED, ACCEPTED, DISPUTED, OVERDUE or CANCELLED. Acceptance is not the same as an assertion by the person responsible; it requires the review procedure designated in the mandate or constitution.

Consider a cooperative that approves a public safety manual at finalized height 2,500,000. Its mandate activates after 1,440 blocks, at height 2,501,440, nominally one day later. A first draft is due within 43,200 additional blocks, at height 2,544,640, nominally thirty days after activation. The cooperative receives the draft, verifies its hash, reviews its content and records acceptance or

objections. Actual calendar deadlines must also account for network stalls: SHB's **one-minute target block interval** is not a guaranteed wall-clock service.

A completed digital step may be proved by a new on-chain state or by a published document commitment. A completed physical step needs independent inspection, trusted signers or some other agreed external verification process. SHB can preserve who made the claim and who accepted it; it cannot derive physical reality from a hash.

16.5 Responsible Representatives and Limited Mandates

Responsibility should be assignable without surrendering the community's control. A mandate may name a delegate, a working group or a multisignature committee, with different permissions for submitting work, approving milestones and escalating disputes. Chapter 10's delegation mechanism concerns voting influence; an **execution appointment is a distinct permission**. Receiving delegated SHB votes must never silently grant access to edit a constitution, publish official records or act on behalf of an organization.

Each assignment should specify its scope, duration, signing threshold and revocation process. For consequential actions, a community might require two of three named reviewers to accept a report, while allowing one project coordinator to upload a draft. The separation of duties reduces the risk that the same person can authorize, perform and self-certify an action. A representative removed from office retains an auditable historical record, but their authorization to perform future actions ends at the relevant finalized transition.

16.6 Evidence, Receipts and Verifiable Completion

An **execution receipt** links an individual action to the mandate, the signer, the outcome and the evidence hash. Its core includes action_id, mandate_id, executed_height, result_code, output_commitment and attestation_signatures. Each receipt should be included in the archival sequence described in Chapter 15. A receipt for an automatic state change can be checked by replaying the accepted transition. A receipt for an external task must declare the level and source of verification, rather than borrowing the certainty of blockchain finality.

Algorithm 16.1 — Apply an authorized on-chain action

```
function ExecuteMandateAction(action_id, finalized_height, state):
    a = LoadApprovedAction(action_id, state)
    require FinalizedTally(a.proposal_id, state)
    require ApprovedUnderFrozenRules(a.proposal_id, state)
    require finalized_height >= a.activation_height
    require finalized_height < a.expiry_height
    require not IsPaused(a.mandate_id, state)
    require not IsConsumed(action_id, state)
    require WithinGovernanceScope(a, state)
    require PreconditionsHold(a, state)
    next = ApplyBoundedNativeTransition(a, state)
    next.consumed_actions.add(action_id)
    next.receipts.add(MakeExecutionReceipt(a, next))
    return AtomicCommit(next)
```

This pseudocode describes a proposed state transition, not a completed SHB implementation. It deliberately excludes direct execution of unverified external work. The receipt and consumed marker must be committed in the same atomic transition so that retries after network interruption cannot repeat the operation.

16.7 Monitoring Commitments Without Inventing Authority

A community should be able to ask what happened after the vote without relying on private correspondence. An SHB monitor may calculate the portion of milestones accepted by their deadlines, identify mandates with missing reports and create public review notices. A simple completion measure can help organize the work, but it must not erase the difference between verified and self-reported outcomes.

```
completion_ratio = accepted_milestones / defined_milestones, where defined_milestones > 0
```

A milestone that has been submitted but not reviewed is not yet accepted. The ratio is descriptive, not a universal score for the competence or honesty of a representative. Monitoring tools may recommend escalation, a new vote or a public inquiry, but none should possess an independent right to invent penalties. Any sanction, removal or replacement must be authorized by the applicable constitution and recorded as a separate decision.

16.8 Reversibility, Correction and the Limits of Automation

A confirmed block is not reversed merely because a community regrets a decision. The appropriate remedy is usually **a new, authorized counter-action**: revoke a mandate, restore a previous local setting, suspend further steps or approve a replacement policy. Restoration may be possible for an internal governance state; it may be impossible for a document already copied by outsiders, a contract signed under applicable law or work already performed in the physical world. A useful execution layer distinguishes reversible actions from irreversible consequences before the vote opens.

The first SHB execution profile should prioritize narrowly defined native actions, explicit time delays, role separation, idempotent receipts, independent review and auditable corrections. The security concerns of programmable smart contracts were already well documented by February 2020 [60–61]; they argue for conservative design rather than unrestricted automatic powers. Every extension would require testing and independent scrutiny.

Nothing in this chapter changes the two base network parameters: the total supply is fixed at **100,000,000 SHB**, with no block-reward issuance, and the nominal block production interval is **60 seconds**. A community can authorize useful work without making SHB a payment token. The ledger provides continuity between a decision and its consequences; people remain responsible for turning the decision into reality.

References and Technical Foundations

[60] L. Luu et al., Making Smart Contracts Smarter, Proceedings of ACM CCS, 2016, pp. 254–269. https://doi.org/10.1145/2976749.2978309

[61] N. Atzei, M. Bartoletti and T. Cimoli, A Survey of Attacks on Ethereum Smart Contracts (SoK), Principles of Security and Trust, 2017, pp. 164–186. https://doi.org/10.1007/978-3-662-54455-6_8

PART IV / BEYOND THE BALLOT

Chapter 17

A Marketplace of Ideas, Not Money

The value of a proposal often begins before anyone is ready to vote on it.

17.1 Collaborative Idea Development

Useful ideas usually go through several revisions before a vote. A member identifies a problem, another brings evidence, and a third discovers that the suggested solution has already been tried elsewhere. Ordinary discussion software can support this exchange, but it rarely preserves a reliable connection between the first idea, its revisions, the objections raised and the proposal ultimately put to a vote. We envisage an **idea layer** that keeps those relationships visible without treating every unfinished thought as a binding governance transaction.

An idea is not a proposal and an endorsement is not a vote. The distinction matters. An author may register an idea reference with a space_id, an idea_id, a title, a short summary, an author signature, a document commitment and a parent reference, if one exists. The full working text, attachments and discussion should ordinarily remain outside the consensus state. The ledger may preserve selected signed milestones or their hashes; a community interface can display the richer conversation. This design limits the burden on validators while allowing readers to check whether a later document matches a previously recorded version.

A simple lifecycle is DRAFT, OPEN_FOR_REVIEW, REFINED, PROPOSAL_CANDIDATE and ARCHIVED. These are organization labels, not proof that an idea is correct or popular. Contributors may propose alternatives, merge compatible suggestions or keep competing approaches separate. When a group chooses to introduce a formal governance proposal, it creates a new proposal object under Chapter 8 rather than silently converting informal reactions into votes. The community's constitution continues to determine who may submit that formal object.

17.2 Public Proposal Libraries

Many organizations repeatedly discuss the same questions because their earlier reasoning is scattered across messages, private files and inactive websites. The SHB architecture provides for **public idea and proposal libraries** in which materials can be searched by topic, community, document version, decision outcome and subsequent implementation. An entry should distinguish the original idea from its derived proposals and from actual decisions. A reader must be able to discover that an attractive proposal was considered and rejected, not merely that it exists in a catalogue.

A minimal library record may contain record_id, space_id, subject_tags, source_commitment, author_reference, parent_record_ids, revision, status and license_reference. Tags and search rankings can be maintained by community applications; they are not consensus-critical and must not change vote tallies. Document commitments, signed attributions and finalized status transitions are stronger claims because independent clients can verify them against the ledger. A public library must also disclose when referenced material is unavailable: **a verifiable hash does not preserve the file it identifies**.

Reuse needs attribution and permission. Publication on a blockchain does not transfer copyright or grant an automatic license to reproduce a design. Community authors may choose open licenses, but the chosen terms must be displayed and preserved alongside the cited version. Governance should never be confused with ownership of someone else's work.

17.3 Argument and Evidence Mapping

A long discussion often rewards the loudest participant rather than the clearest reasoning. We propose an optional **argument map**: a set of individually attributable claims connected by typed

relationships such as SUPPORTS, CHALLENGES, DEPENDS_ON and SUPERSEDES. A claim may point to a document commitment, a signed observation or a public source. Changes to a claim create a new version so that a later reader can reconstruct the evidence available when a decision was made.

The graph is a description of the debate, not a machine for establishing truth. Two participants may reasonably disagree about what a report proves; a cryptographic signature establishes who submitted an interpretation, not whether the interpretation is sound. Interfaces should separate *the existence of evidence*, *the reliability of its source* and *the conclusion someone draws from it*. Communities may authorize reviewers to classify sources, but a reviewer must not be able to remove a valid contrary argument from the historical archive.

Evidence label	What an SHB client may verify	What remains a human judgment
Signed statement	Signer and committed bytes	Whether the statement is accurate
Document reference	Hash match and ledger position	Whether the document is complete
Recorded objection	Origin, revision and submission order	Whether the objection is persuasive
Independent review	Reviewer's credential and signed report	Whether the reviewer is impartial

A useful additional feature is a **contradiction register**. When two incompatible claims influence the same proposal, their relationship can be recorded as unresolved rather than hidden in a thread. The register would encourage a proposal author to specify which uncertainty matters to the decision. It should never delay a valid vote automatically unless the community's published rules explicitly require further review.

17.4 Nonfinancial Recognition of Contributors

People contribute useful work for reasons other than payment: expertise, curiosity, duty, shared identity or the satisfaction of solving a difficult problem. The draft specification provides for signed recognition records for contributions such as translating a proposal, detecting a flaw in an implementation plan, preparing an independent review or documenting an earlier experiment. These records can appear as **nontransferable acknowledgements**, linked to evidence and capable of correction or revocation under a defined procedure.

Recognition is deliberately separate from the SHB balance. No badge creates new SHB, increases the global supply or automatically multiplies a participant's token-weighted vote. A participant's base influence remains defined by eligible SHB at the applicable snapshot. A community may require verified qualifications for a technical review role, but this is a permission to perform a specific task, not ownership of additional votes. Chapter 13 sets out the broader safeguards for reputation and credentials.

Recognition should remain optional. For some contributors, anonymity is a legitimate protection. A person must not have to disclose an employer, home address or personal history simply to improve a proposal. A public credential may attest to a completed contribution while disclosing only the minimum information that a particular role requires. Interfaces should also avoid turning the number of badges into a crude ranking of people: prolific activity and sound judgment are not the same thing.

17.5 Community Challenges

An organization may know that something is wrong without knowing which solution to adopt. We envisage **community challenges**: bounded calls for ideas with an explicit problem statement, admission rules, evaluation criteria and a closing height. Examples include finding a better document-retention policy, improving accessibility to a public service, or selecting a safer testing procedure for an open-source component. A challenge can accept several candidate solutions without requiring the community to vote immediately.

The challenge organizer may appoint reviewers, publish a comparison matrix and invite an open comment period. Reviews must be attributable and archived with their stated criteria. A shortlist is only a recommendation. Any binding adoption still requires a formal proposal and the relevant SHB vote. This

separation prevents a small review committee from acquiring governance authority simply by controlling the discussion interface.

A proposed challenge record contains challenge_id, space_id, question_hash, entry_deadline_height, criteria_hash, reviewer_policy_hash and result_reference. At a nominal block interval of **60 seconds**, a review period of 4,320 blocks corresponds to approximately three days, subject to delays if consensus stalls. The challenge mechanism requires neither prize payments nor token issuance. If an organization chooses to offer some external reward, that decision and its legal consequences remain outside the SHB protocol's primary purpose.

17.6 Preference Discovery Before a Formal Ballot

A vote is often too late to discover that participants understood the question differently. We have therefore included an optional **preference-discovery stage** in which members can endorse multiple ideas, indicate uncertainty, request revisions or mark alternatives as worth comparing. These signals are advisory. They do not constitute a governance ballot, and an interface must label them accordingly.

A community can choose whether discovery signals are public, pseudonymous or visible only in aggregated form. Simple counts are especially vulnerable to multiple-address participation: an SHB address is not proof of a distinct person. The default discovery interface should therefore avoid presenting raw endorsement counts as evidence of democratic legitimacy. If token-weighted sampling is used, the weighting and snapshot height must be stated; it remains different from the constitutional vote, even if the same SHB balances are consulted.

One useful output is a **consensus-and-difference summary**: which options attract broad support, which objections remain unresolved, and which questions have insufficient information. Such summaries can be constructed through ordinary tabulation and documented statistical techniques available in February 2020. They should expose their input rules and allow independent review. Any automated clustering of opinions is an aid to reading many responses, not a substitute for community choice or a source of binding authority.

17.7 Collective Forecasting Experiments

Communities routinely make decisions about uncertain futures. They expect a policy to improve participation, a software change to reduce errors or a working group to deliver on time. The SHB architecture provides for a small **forecast register** that lets participants attach an explicit probability and a future resolution rule to a verifiable question. For example: “Will the test application meet its published performance target by a designated review height?” A forecast entry should identify its question, forecast probability, author reference, submission height and the version of the criteria under which the outcome will be assessed.

A forecast is not a vote. A large SHB holder does not become a more accurate forecaster by owning more tokens. Where a binary event is resolved as o = 1 for occurrence and o = 0 otherwise, a simple accuracy measure is the Brier score, known since 1950 [64]:

```
Brier(p,o) = (p - o)^2 ; 0 <= p <= 1 ; o in {0,1}
```

A forecast of p = 0.60 followed by the event occurring has a score of 0.16; smaller scores indicate better accuracy for this individual binary question. A credible comparison between forecasters requires multiple questions, known resolution criteria and attention to selection effects. The protocol should not equate one successful guess with expertise, nor should forecasts silently modify SHB voting power.

The outcome of a real-world question often depends on evidence outside the chain. SHB can verify who submitted a forecast and which outcome report was signed, but it cannot independently determine whether a bridge was repaired or a business target was achieved. An appointed resolver, multisignature review or formal community vote may be required. Unresolved or disputed questions must remain visibly unresolved rather than be assigned convenient outcomes.

17.8 Open Knowledge as a Protocol Resource

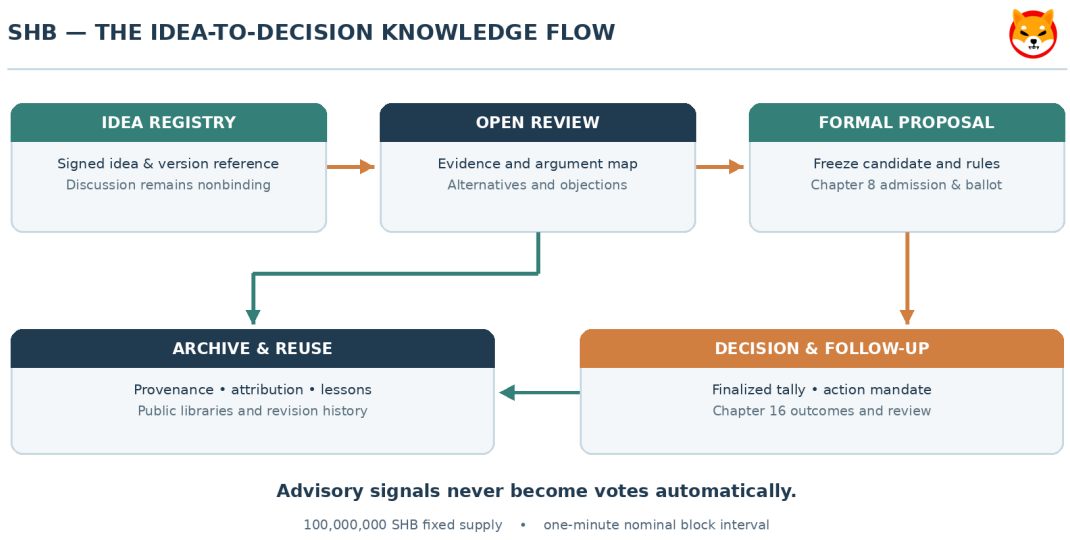
The purpose of this layer is to make a community's reasoning useful beyond one ballot. Ideas can be improved, challenges can be repeated, evidence can be re-examined and mistakes can be documented without requiring the entire process to begin again. A small cooperative and a large distributed software project may use different governance constitutions, yet both can benefit from versioned questions, transparent review and reusable knowledge.

Algorithm 17.1 — Promote an idea to a formal proposal

```
function PrepareProposalCandidate(idea_id, space_id, state):
    idea = LoadIdeaReference(idea_id)
    require idea.space_id == space_id
    require VerifyAuthorAndRevisionChain(idea)
    require AllDeclaredEvidenceHashesWellFormed(idea)
    require SourceLicenseAndAttributionRecorded(idea)
    candidate = FreezeCandidateDocument(idea.current_revision)
    candidate.parent_idea_id = idea_id
    candidate.evidence_root = CommitEvidenceSet(idea.evidence)
    candidate.status = "PROPOSAL_CANDIDATE"
    return SubmitThroughChapter8Admission(candidate, state)
```

This illustrative algorithm is an application-layer design, not a claim of an existing SHB deployment. The final admission call still checks the constitution, sponsorship requirements, eligibility, proposal scope and other rules described in Chapter 8. An idea becomes binding only through the established governance process.

FIGURE 17.1 THE IDEA-TO-DECISION KNOWLEDGE FLOW



Signed ideas can be reviewed, developed and admitted as formal proposals. Nonbinding discussion and recognition remain separate from voting, while the decision and its reasoning are preserved for later reuse. OT Development Team, February 2020.

To avoid making every conversation a permanent transaction, SHB's initial architecture should commit only selected records and version hashes to its shared ledger. Supporting discussions and attachments can be retained off-chain by independently operated custodians, subject to their licensing, privacy and availability obligations. The fixed supply of **100,000,000 SHB** is unaffected by endorsements, recognition or forecasts; the network continues to target one block every 60 seconds. Nothing in this idea layer constitutes a payment or a monetary claim.

The most valuable outcome is not a larger number of proposals. It is a community that remembers why it chose one course over another, which alternatives it declined, and what it learned afterward. An archive of decisions is useful; an archive of the thinking behind them can be more useful still.

References and Technical Foundations

[62] W3C Provenance Working Group, PROV-O: The PROV Ontology, W3C Recommendation, 30 April 2013. https://www.w3.org/TR/2013/REC-prov-o-20130430/

[63] P. Rossini and J. Stromer-Galley, Citizen Deliberation Online, in Oxford Handbook of Electoral Persuasion, published online 1 October 2019. https://doi.org/10.1093/oxfordhb/9780190860806.013.14

[64] G. W. Brier, Verification of Forecasts Expressed in Terms of Probability, Monthly Weather Review, 78(1), 1950, pp. 1–3. https://doi.org/10.1175/1520-0493(1950)078(0001:VOFEIT)2.0.CO;2

PART V / BUILDING SHB

Chapter 18

Making the Network Useful

An independently verifiable network should be usable without central permission.

18.1 Core Protocol Modules

We divide the proposed SHB software into a small collection of components with sharply defined responsibilities. The **consensus layer** orders blocks and establishes finality; the **state-transition engine** validates transactions and calculates the resulting state; the **governance engine** interprets constitutions, eligibility snapshots, ballots, delegations and mandates. An application can display or search this information, but it must not be able to change a valid tally by changing its own database. This separation is a practical safeguard against an otherwise familiar failure: one attractive interface becoming the only authority that users can consult.

The initial implementation should favor deterministic operations over unrestricted program execution. Every validator must obtain the same result from the same ordered inputs, including when a transaction is malformed. A canonical transaction envelope contains chain_id, protocol_version, transaction_type, sender, nonce, payload, and signature. Signing bytes are derived from an unambiguous binary serialization that includes the chain identifier and transaction domain. User-facing JSON is suitable for client communication; it should not become the definition of consensus-critical serialization, because alternate parsers may handle equivalent-looking documents differently. The version number identifies decoding and validation rules, not permission for a node to invent its own rules.

The ledger starts with a fixed supply of **100,000,000 SHB** allocated in genesis state. Transactions may move eligible units or alter governance records under published rules, but ordinary block production does not mint SHB. The intended **one-minute block interval** is a nominal cadence, not a promise of continuous finality through partitions. Chapter 5's BFT safety requirements continue to govern whether a block may be committed.

18.2 The Validator Software

A validator node needs networking, a durable state store, block and vote verification, a consensus service, a transaction admission queue and an operator interface. It should keep signing material separate from ordinary administrative access. The signing process must reject an attempt to endorse conflicting blocks or votes for a consensus position already signed; a crash-recovery journal can reduce accidental double-signing, although it does not remove the need for operational discipline. Chain data, validator configuration and private keys should be backed up under different procedures.

A full node verifies committed block signatures, state transitions and governance invariants without producing consensus signatures. An observer may serve queries or maintain a searchable index while leaving the canonical state untouched. This role separation allows a community to run its own reader infrastructure even if it does not operate a validator. The validator set and its admission mechanism remain governed by the network configuration and the phased decentralization process described elsewhere in this paper; no universally permissionless validator-selection method is claimed as implemented here.

For a normal block, a node should check the prior block commitment, validator-set version, ordered transaction list and deterministic state root before accepting the result. Failed transactions must have specified effects: a rejected instruction must never partially mutate a ballot or move SHB balances. Operational metrics should distinguish a proposed block, a received block and a **finalized** block. Treating these as interchangeable would mislead wallets and voters.

18.3 Wallet and Governance Interfaces

The first SHB wallet should be primarily a **participation wallet**. It displays an address, eligible token balance, community memberships, active proposals, delegated voting weight, pending commitments and finalized ballots. A voting action must show the proposal identifier, constitution version, snapshot height, available choices, effective weight and closing height before it requests a signature. The signer should approve the exact meaning of the message, not a cryptic digest whose contents only a server understands.

Private keys should be generated and retained on the user's device whenever practical. The application may submit a signed transaction through several independent public endpoints, but no endpoint should need the private key. Hardware-backed storage and offline signing can be supported through an open signing format; their absence from an early wallet should be disclosed. A wallet must also distinguish an unconfirmed submission from a finalized result, and explain that changing SHB balances after the snapshot does not retroactively alter a completed ballot.

A nontechnical interface can offer clear language, accessibility controls and a receipt containing a transaction identifier and block height. It should also permit independent verification of that receipt through a separately operated node. The objective is not to make users read cryptographic proofs before every vote. It is to ensure that they *can* verify the record without depending on the same interface that collected their choice.

18.4 Application Programming Interfaces

We propose a read-oriented public API and a narrowly scoped transaction-submission API. A conventional JSON-RPC 2.0 interface is one feasible choice available in February 2020 [65]. For ordinary clients, the API may expose methods such as getBlock, getAccountState, getGovernanceSpace, getProposal, getVotingSnapshot, getFinalizedTally and broadcastSignedTransaction. Method names shown here are illustrative rather than a finalized wire specification. Every response describing ledger state should carry a block height or finalized state reference, so that two clients can tell whether they are examining the same state.

Client operation	Minimum verifiable response	Trust boundary
Query a proposal	proposal_id, version, state height, commitment	Indexing and display are not consensus
Query voting power	Snapshot height, eligible units, delegation rule	Weight comes from finalized state
Send an instruction	Transaction identifier and admission status	Acceptance into a queue is not finality
Verify a decision	Finalized height, tally, supporting proof reference	Local recomputation remains possible

Public RPC endpoints require request limits, authentication where appropriate, transport encryption and protection against expensive queries. These operational controls should not silently change the chain's governance eligibility rules. For example, a gateway may reject excessive requests from one client while a valid signed ballot remains admissible through another node. No external developer should be required to license a private API to verify a community's decisions.

18.5 Smart Contract Considerations

General-purpose smart contracts can express sophisticated workflows, but they also introduce larger attack surfaces, resource accounting problems and difficult upgrade questions. SHB does not require unrestricted smart contracts for its first practical release. A **native governance module**, with explicit transaction types and deterministic validation, is the preferred initial path for proposal creation, voting, delegation, constitution changes and execution mandates. Narrowly defined modules are easier to test against the exact meaning of a vote.

Extensions may later support user-authored conditional actions or limited execution scripts, but any such feature would need a defined instruction set, bounded execution cost, reproducible results, versioned interfaces and independent security review. A decision that depends on a physical-world event cannot be made true by a contract alone. If an organization votes to complete a building inspection, SHB can record the authorized mandate and an inspector's signed report; it cannot independently inspect the building.

18.6 Developer Tools

The development kit should include a reference full node, a command-line wallet, a transaction encoder, a signature verifier, a local multi-validator test environment and sample clients. A deterministic fixture library would allow two independent implementations to process the same sequence of transactions and compare their resulting state roots. Test vectors should cover wrong-chain signatures, duplicate nonces, expired proposals, conflicting delegations, malformed commitments, aborted vote revelations, invalid snapshots and emergency rule changes.

A straightforward test is not merely whether a transaction is accepted. It is whether every honest node *rejects the same invalid transaction for the same protocol reason*. The following application-facing procedure illustrates the separation between local signing, network admission and verified finality:

Algorithm 18.1 — Submit and independently verify a governance instruction

```
function SubmitAndVerify(ballot, wallet, rpcA, rpcB):
    require wallet.hasValidatedProposal(ballot.proposal_id)
    context = rpcA.getFinalizedProposalContext(ballot.proposal_id)
    require context.chain_id == wallet.expected_chain_id
    require context.snapshot_height == ballot.snapshot_height
    payload = EncodeCanonicalGovernanceInstruction(ballot, context)
    signed = wallet.signLocally(payload)
    tx_id = Hash(signed)
    rpcA.broadcastSignedTransaction(signed)
    receipt = rpcB.waitForFinalizedReceipt(tx_id)
    require VerifyFinalityProof(receipt.finality_proof)
    require VerifyReceiptAgainstState(receipt, tx_id)
    return receipt
```

The procedure intentionally uses an independent query route for verification, although independence of operators cannot be inferred from different website addresses alone. The functions are conceptual API boundaries, not a claim of available SHB software. Open documentation and reproducible builds should make external security review possible.

18.7 Governance Application Templates

A standard wallet cannot anticipate every organization's constitution. We therefore propose reusable templates for associations, open-source projects, professional working groups and cooperatives. A template should create a **new governance space**, not a new currency. Each space defines eligibility, proposal admission, available ballot types, delegation permissions, quorum, execution authority and review periods. Templates must clearly identify which settings can be changed locally and which are invariants of the shared chain.

A template is not a legal charter and cannot replace one where a jurisdiction requires formal corporate procedures. Its value is operational consistency. A modest association may use simple approval ballots and named secretaries; a distributed software group may adopt delegated voting and public technical reviews. Both can verify their records against the same SHB ledger without imposing the same substantive rules on one another.

18.8 Network Resource Limits

A nominal one-minute interval implies **1,440 block opportunities per 24 hours** when the network advances on schedule. It does not establish throughput by itself. Let T_b denote the number of accepted governance instructions in a finalized block and Δ the observed average time between finalized blocks. An illustrative application throughput measure is:

```
throughput_observed = sum(T_b) / sum(Delta_seconds_between_finalized_blocks)
```

A hypothetical block carrying 100 valid instructions every 60 seconds yields approximately 1.67 instructions per second; this is an arithmetic example, not a benchmark. Processing time, validator communication, evidence sizes, signature checks, disk operations and the distribution of active communities will determine practical limits. Large documents belong in external storage with commitments recorded on chain, not in every consensus block. Pruning and snapshot synchronization may reduce storage costs, but archival nodes must remain able to validate historical proofs under documented rules.

Spam resistance remains necessary even without a payment thesis. Initial controls may include bounded transaction sizes, per-account sequence checks, proposal admission thresholds and local peer rate limits. Any consensus-level admission quota must be public, deterministic and assessed for its effect on smaller participants. A design that protects the chain by making ordinary voters unable to submit ballots would fail its central purpose.

18.9 Operational Sustainability

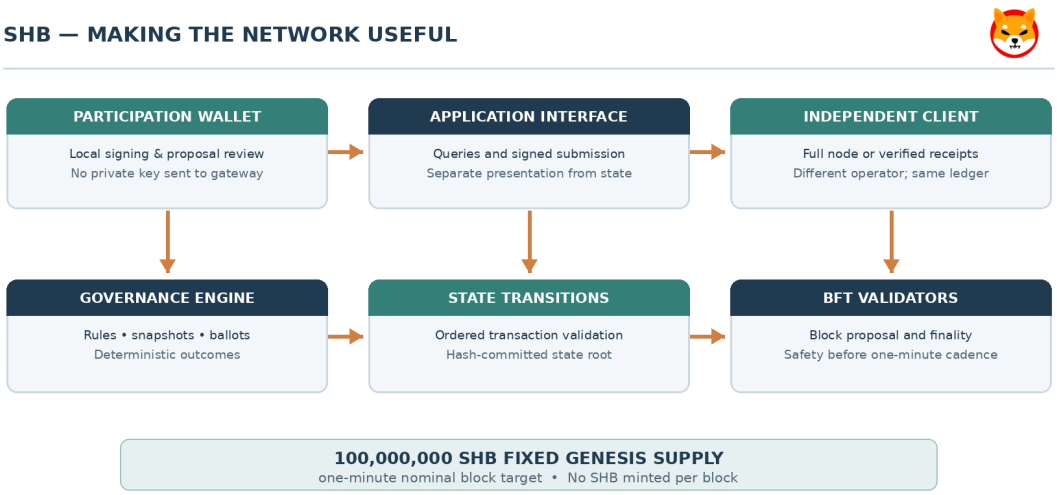
Independent validators must pay for servers, network access, backups, monitoring and security work. A fixed-supply governance token does not make these costs disappear. The draft specification provides for a sustainable operating model based on separately disclosed funding arrangements, contributor support, community infrastructure commitments or other arrangements consistent with applicable law. No continual block reward is assumed, and creating new SHB for operational expenses is incompatible with the 100,000,000-token cap.

The initial team should publish node requirements, update procedures, supported software versions, incident reports and a clear schedule for removing unnecessary operational privileges. Source-code transparency alone does not guarantee decentralization: if one entity controls the validator keys, software releases and most public gateways, users remain dependent on that entity. The operating model should therefore be evaluated by who can independently validate, maintain and exit the network, not only by how many nodes appear in a dashboard.

18.10 Potential Adoption Scenarios

A student association might use SHB for constitutional amendments and elections of project representatives. An open-source team might vote on priorities and record acceptance criteria for a release. A federation of local organizations might approve a shared policy only after each participating community reaches its own required threshold. None of these arrangements requires SHB to serve as a merchant payment instrument. Each needs trustworthy records of who could participate, which proposal was considered and what the group decided.

FIGURE 18.1 THE SHB IMPLEMENTATION AND VERIFICATION STACK



The SHB client stack separates user signing, public interfaces, deterministic governance state and validator finality. Every element shown is a proposed implementation component, not evidence of a deployed network. OT Development Team, February 2020.

The proposed first release should therefore remain narrow: a verifiable ledger, a safe participation wallet, reliable proposal and ballot workflows, independent node access and clear instructions for operating a governance space. Additional capabilities can be introduced only after the core rules survive testing under disagreement, malfunction and hostile behavior. The goal is a network that communities can use, examine and challenge without asking the development team to be their permanent referee.

References and Technical Foundations

[65] JSON-RPC Working Group, JSON-RPC 2.0 Specification, originating 26 March 2010, updated 4 January 2013. https://www.jsonrpc.org/specification

[66] S. Josefsson and I. Liusvaara, Edwards-Curve Digital Signature Algorithm (EdDSA), RFC 8032, January 2017. https://www.rfc-editor.org/rfc/rfc8032

[67] T. Bray, The JavaScript Object Notation (JSON) Data Interchange Format, RFC 8259, December 2017. https://www.rfc-editor.org/rfc/rfc8259

PART V / BUILDING SHB

Chapter 19

Threats, Limits and Responsibilities

Decentralization changes where trust is placed. It does not remove the need for it.

19.1 Token Concentration and Governance Capture

The rule that one eligible SHB carries one unit of voting weight is easy to audit. It is not a claim of political equality between people. A single person may control many addresses, and a large holder may win a ballot in which few others participate. We should be explicit about that limitation. SHB can verify how much influence was exercised under a published rule; it cannot establish that every human being had an equal share of it.

Consider a governance space with 3,000,000 eligible SHB, a 20% participation quorum and a requirement that 60% of decisive votes approve a measure. An organized holder with 900,000 SHB cannot command a majority of the eligible supply, yet could approve a proposal if only 300,000 additional units cast opposing votes. Participation would be 1,200,000 units, above quorum, and support would represent 75% of decisive votes. Both numbers are correct; neither proves broad community consent. Interfaces should disclose eligible supply, turnout, participation concentration and the rule actually applied, rather than displaying only the winning percentage.

Community constitutions may employ longer review periods, higher thresholds for irreversible decisions, timelocks, topic-specific eligibility or independent oversight. Such protections must be established before the vote. An emergency amendment invented after an unfavorable result would itself be governance capture.

19.2 Sybil Attacks and Artificial Participation

A Sybil attack occurs when one actor presents multiple identities as though they represented independent participants [68]. SHB must distinguish three different objects: *addresses*, *voting units* and *people*. In a simple token-weighted ballot, splitting 1,000 SHB among ten wallets does not create additional weight if the same eligible units are counted once at the snapshot. It can, however, distort visible participation statistics and exploit mechanisms that grant privileges per account.

Community membership rules and reputation schemes have a different exposure. If an attacker can create many qualifying credentials, it may obtain repeated proposal rights or dominate a membership-based committee. The design accordingly provides for issuer verification, credential expiry, revocation records, eligibility checks and explicit limits for any per-identity benefit. These measures reduce particular risks but do not solve universal proof of personhood. For the first release, eligibility methods should be identified by the community that uses them, together with their trust assumptions.

19.3 Validator Collusion and Consensus Failures

The validator layer decides the accepted order of transactions, not the political content of a community's decisions. In the proposed BFT design, finalization requires a certificate meeting the configured quorum, normally more than two-thirds of voting power under the fault assumption described in Chapter 5. Safety depends on honest voting power and consistent rules; it does not follow merely from the existence of several servers.

Validators may attempt to exclude a ballot, sign conflicting messages, withhold blocks or cooperate in a network partition. A wallet should distinguish **submitted**, **included** and **finalized** instructions. Nodes should preserve signed evidence of equivocation, monitor missing finalized blocks and allow

users to submit valid instructions through more than one operator. Evidence of misconduct does not automatically restore availability. If the fault assumptions cease to hold, the defensible outcome may be to stop finalization rather than manufacture an apparently valid history.

A nominal block interval of **60 seconds** is a performance target, not a security override. No participant should be promised that a decision will complete at an exact wall-clock time during an outage.

19.4 Vote Manipulation, Censorship and Coercion

A signed ballot proves control over the signing key; it does not establish that the voter understood the proposal or acted freely. The main threats are misleading interfaces, phishing, selective presentation of evidence, voter intimidation, transaction censorship, premature disclosure of intended choices and confusion about delegation. The protocol can verify a signature and a final tally, but it cannot judge whether an argument was fair.

Each ballot should bind its signature to chain_id, governance_space_id, proposal_id, proposal_version, snapshot_height and the selected option. Wallets should display these fields intelligibly. A vote for one text must not be silently reused for an amended text. Independent endpoints, locally verifiable receipts and archived proposal versions provide some protection against interface manipulation.

Commit–reveal voting, discussed in Chapter 11, can conceal a choice before revelation. It does not in itself provide permanent ballot secrecy, resistance to coercion or protection against a voter revealing a commitment voluntarily. Any stronger privacy mechanism would require a separate security design and external assessment.

19.5 Cryptography, Keys and Software Defects

A sound mathematical primitive does not make an unsafe wallet secure. Private keys may be stolen through malware, copied from backups or exposed by a malicious signing interface. Cryptographic implementations may fail through weak randomness, incorrect signature validation or ambiguous message encoding. State machines may disagree on integer overflow, invalid inputs or the order in which checks are applied. Such defects can turn a visible governance process into a contested ledger.

We envisage canonical serialization, chain-specific signature domains, explicit integer ranges, deterministic transaction validation and reproducible test vectors. A validator must apply every check before changing state; a rejected transaction must not leave half a delegation or half a ballot in the ledger. Independent code review, fuzz testing and conflicting-implementation tests are required before the software can be called production-ready. This document presents design requirements, **not** an assertion that a security audit has occurred.

19.6 Privacy, Public Records and Data Retention

An immutable history is useful for accountability and dangerous when it carries unnecessary personal information. Governance activity may expose address balances, patterns of association, voting preferences and relationships between delegates. Hashing a document does not necessarily anonymize it: if the underlying information is identifiable, the hash or the surrounding metadata may remain personal data.

The network should record minimal on-chain information: commitments, identifiers, authorization proofs and final decision records. Names, detailed membership files, supporting evidence and other sensitive materials should remain off-chain under separately governed access and retention policies. Deleting an off-chain copy cannot guarantee erasure of every replica, and an on-chain commitment cannot generally be removed. These limits must be explained before deployment.

For communities operating in the European Union, the General Data Protection Regulation, applicable since May 2018, already imposes obligations concerning lawful processing, data minimization and individual rights [70]. SHB is not automatically compliant because it is decentralized. Responsibility for collecting and publishing data must be examined for each deployment.

19.7 Network Spam and Denial of Service

SHB is not intended to operate as a payment network, but submission and storage still consume scarce resources. An attacker can flood nodes with invalid signatures, repeated proposals, expensive queries or oversized attachments. With a nominal block every minute, unbounded work can overwhelm validation before any governance rule is evaluated.

The proposed defenses separate **peer-level** controls from **consensus-level** rules. Public gateways can use request limits and connection controls; nodes can reject malformed transactions early. The protocol can impose maximum payload sizes, per-account nonces, proposal admission requirements and bounded computation. All consensus-level restrictions must be deterministic and published. Spam controls must also be tested against legitimate bursts of ballots near closing height, so that protection does not become selective exclusion.

19.8 Software Upgrades and Emergency Powers

Upgrades may repair vulnerabilities but also introduce new forms of central control. A protocol revision should include a public specification, compatibility tests, a declared activation height, an independently verifiable release and a defined procedure if part of the validator set refuses to upgrade. A community constitution may govern its own local rules; it cannot silently redefine the shared ledger's validation rules.

The fixed supply of **100,000,000 SHB** is a protocol invariant. Ordinary blocks and community votes must not mint additional tokens. The intended **one-minute** block cadence is likewise a network-level parameter, rather than something an individual governance space may amend. Changing a foundational network rule would require an openly specified network upgrade and the corresponding consensus process; a published whitepaper cannot make an incompatible client accept it automatically.

Emergency authority should be narrow, time-limited and publicly logged. A temporary pause that protects funds or governance records is preferable to an unreviewable administrator key that can change a ballot or confiscate influence. The team should publish which privileges, if any, remain during development and the conditions for their removal.

19.9 Legal, Organizational and Ethical Responsibilities

SHB can certify that a particular rule was followed by a network. It cannot determine whether a decision is lawful, whether an association had authority to make it or whether a court will recognize it. Employment matters, regulated elections, corporate approvals and public-sector procedures may require identity checks, notices, procedural safeguards or records beyond the protocol. A digitally approved instruction does not override legislation or contractual obligations.

Organizations using SHB should define who is responsible for personal data, erroneous public statements, off-chain implementation and dispute handling. They should also disclose how token distribution may affect governance power. A governance token is not automatically outside financial, consumer or other regulation merely because its stated purpose is non-payment. Legal characterization depends on actual rights, distribution and use. The OT Development Team proposes infrastructure; individual communities remain responsible for using it appropriately.

19.10 The Limits of Technical Decentralization

Decentralization is not measured only by the number of public keys in a validator list. It also depends on who develops the software, controls initial allocations, operates public gateways, publishes information and can organize users in a dispute. Source availability is valuable, but it does not by itself remove control over funding, infrastructure or governance culture [69].

The first SHB release should therefore report measurable conditions: validator concentration, reachable independent full nodes, public-client alternatives, turnout distribution, unrevealed commitments, unresolved security findings and time required to recover from tested failures. The

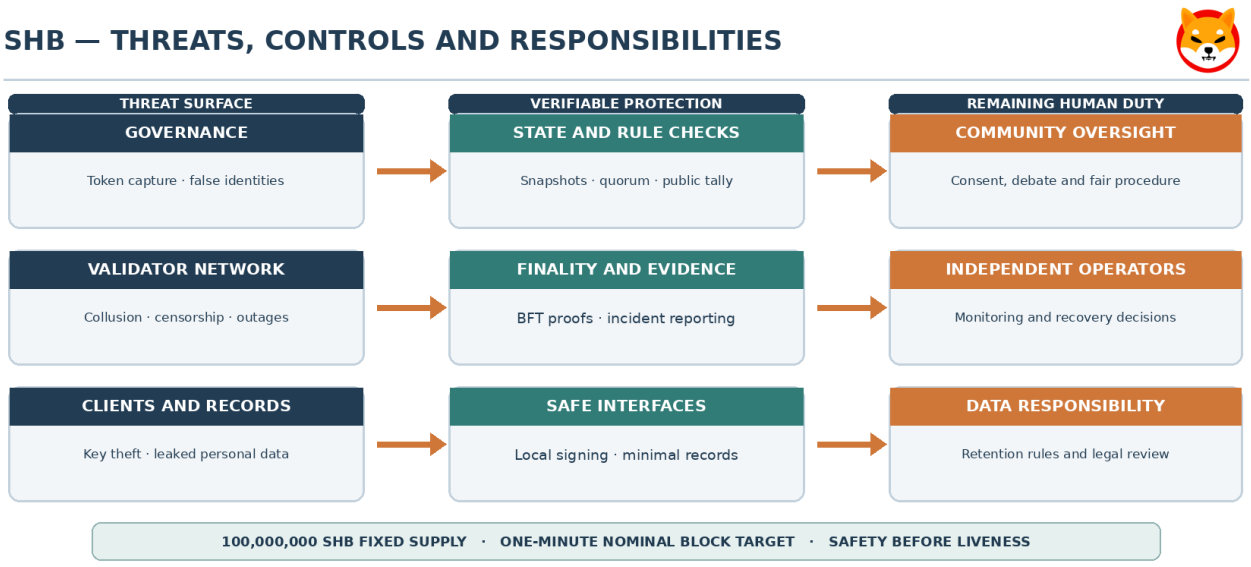
following review procedure is a proposed operational discipline, not a claim that it has already been deployed.

Algorithm 19.1 — Pre-activation security gate

```
function ReviewProtocolChange(release, activation_height):
    require release.hasPublicSpecification()
    require release.hasDeterministicTestVectors()
    require ValidateSupplyInvariant(release, 100000000)
    require ValidateGovernanceIsolation(release)
    require release.hasIndependentReviewReport()
    require activation_height > CurrentFinalizedHeight() + ReviewDelay
    publish(Hash(release), activation_height, OpenRisks)
    if CriticalUnresolvedRisk(OpenRisks):
        return DO_NOT_ACTIVATE
    return READY_FOR_CONSENSUS_APPROVAL
```

Risk	Principal control	Residual limitation
Concentrated token holdings	Public turnout data, stricter constitutional thresholds	Economic influence remains unequal
Sybil identities	Snapshot accounting, issuer checks for membership rules	Real-world personhood may remain unverified
Colluding validators	BFT finality proofs, equivocation evidence, independent nodes	Safety assumption can fail if adversarial power is excessive
Exposed voting preferences	Data minimization, optional commit–reveal	Revelation and metadata can identify participants
Faulty upgrades	Public specifications, test vectors, delayed activation	Bugs and coordination failures remain possible

FIGURE 19.1 THE SHB SECURITY AND RESPONSIBILITY MODEL



Protocol checks reduce observable risks; operational and legal responsibility remains with identifiable actors. Conceptual architecture, February 2020.

A secure governance protocol should make failures detectable, recovery processes understandable and authority accountable. It cannot make all participants honest or all choices wise. SHB's responsibility is to define what can be verified, publish where trust is still required and avoid presenting cryptography as a substitute for legitimate collective judgment.

References and Technical Foundations

[68] J. R. Douceur, *The Sybil Attack*, International Workshop on Peer-to-Peer Systems, 2002. https://www.microsoft.com/en-us/research/publication/the-sybil-attack/

[69] D. Yaga, P. Mell, N. Roby and K. Scarfone, *Blockchain Technology Overview*, NISTIR 8202, October 2018. https://doi.org/10.6028/NIST.IR.8202

[70] European Parliament and Council, *Regulation (EU) 2016/679 (General Data Protection Regulation)*, 27 April 2016, applicable from 25 May 2018. https://eur-lex.europa.eu/eli/reg/2016/679/oj

PART V / BUILDING SHB

Chapter 20

The Road Ahead

A sequence of evidence, not a promise of dates.

20.1 The Initial Development Phase

The first milestone is not a token launch. It is an executable specification that independent developers can implement and test. Every accepted SHB instruction needs a fixed encoding, a clear authorization rule and a predictable effect on the ledger. Two implementations processing the same finalized blocks should arrive at identical state commitments. Until that test passes, expanding the feature list would only make the underlying problems harder to find.

The founding parameters are simple and deliberately limited: **a fixed total supply of 100,000,000 SHB**, allocated at genesis under a publicly verifiable schedule, and a **nominal one-minute block interval** for the proposed BFT network. Neither parameter authorizes per-block token creation. The illustrative allocation discussed in Chapter 7 remains a proposed distribution to be reviewed before genesis, not evidence that any tokens have already been distributed. The finalized genesis manifest must list every initial allocation, any restriction attached to it and the total supply proof.

Initial engineering should prioritize the state-transition specification, transaction serialization, account nonces, signing domains, validator message formats, voting snapshots, ballot finality and deterministic failure handling. The reference implementation should ship with a command-line tool capable of creating a governance space, submitting a proposal, casting a vote and independently checking the result. A cleanly implemented small protocol is more useful than a broad list of untested promises.

20.2 Prototype Network

The prototype is intended to run under controlled conditions with a small number of validators. It must exercise the complete path from a signed instruction to a finalized receipt, including invalid signatures, duplicate submissions, unavailable validators and conflicting transactions. Tests should cover at least three participant roles: a token holder casting a direct vote, a delegate acting under a revocable mandate and an administrator introducing a rule change within a governance space.

The prototype must also prove that application voting cannot bypass consensus, and that consensus operators cannot silently rewrite the recorded result. For a proposal with a fixed snapshot height, the same eligible token units must produce the same tally regardless of the gateway used to inspect it. Governance documents may be stored outside the chain, but their content commitments must remain verifiable against the finalized record.

The earliest release is an engineering demonstration, not an invitation to rely on SHB for binding decisions. Test balances should be clearly labelled as valueless test-state units, with no implication that they represent transferable claims to future mainnet SHB.

20.3 Public Testnet and Independent Operation

A public testnet should follow only after deterministic local tests are repeatable. Its purpose is to expose assumptions the original team may have missed. Independent operators should be able to install a full node, synchronize from a documented genesis, verify a finalized state, submit a signed transaction and observe recovery after temporary disconnection. Public bug reports and reproducible failure cases should be accepted without requiring private access to the development team.

The one-minute interval is a target for normal network operation. Test reports must disclose observed finalization latency, missed block opportunities, recovery time, message overhead and state

growth. A testnet that produces one block every minute with only one active operator has not established the behavior of a decentralized network. Results should include latency and failure measurements under geographically distributed validators and deliberately degraded network conditions.

For the first public version, supported functions should remain bounded: token accounting, governance spaces, proposals, snapshot-weighted ballots, direct delegation, finality proofs and limited execution mandates. More elaborate forms of reputation, federation or confidential voting may be developed as separate experimental modules. They should not be activated in the reference client merely because they are described elsewhere in this paper.

20.4 Independent Code Review

Before a mainnet launch is considered, the executable protocol should receive external review of its cryptographic usage, validator safety assumptions, state machine and wallet key handling. Reviewers should be able to reproduce builds, inspect test vectors and trace critical requirements to the source code. At minimum, testing must include malformed data, integer boundaries, adversarial network scheduling, contradictory validator messages, replayed votes, inconsistent snapshots and the transition between validator configurations.

An audit is not a certificate of permanent safety. Public findings should be classified by severity, remediation status and affected versions. Serious unresolved issues should prevent mainnet activation. The team should publish a reproducible security test plan and a procedure for responsible vulnerability reporting. Tests should also verify that a software upgrade cannot quietly change the **100,000,000 SHB** supply invariant or substitute an unauthorized genesis allocation.

20.5 Governance Testing

Governance testing requires more than generating a correct percentage. Participants must understand the question being voted upon, the eligible electorate, the relevant snapshot and the consequences of approval. A demonstration community should complete ordinary proposals, constitutional amendments, rejected ballots, abstentions, delegated votes, expired mandates and contested procedural actions. Each result must be independently reconstructable from public proofs.

Special attention should be given to governance capture and weak turnout. The network can prove that a configured quorum was reached; it cannot prove that a decision expressed broad social consent. Test reports should therefore state the distribution of voting influence and simulate concentrated holdings, low participation and delegates controlling substantial balances. Community rule changes must be judged under the rules active at the start of the relevant voting procedure, with the prescribed activation delay applied afterward.

No pilot should assume that on-chain approval automatically binds an external association, company or public authority. Any pilot claiming legal effect requires a separate organizational and jurisdiction-specific arrangement.

20.6 Mainnet Readiness and Launch Conditions

Mainnet should be activated because measurable conditions have been met, not because a calendar date has arrived. The release gate should require: a frozen and reviewed genesis manifest; verified accounting of all **100,000,000 SHB**; independent validators and reachable full nodes; safe wallet signing; deterministic proposal and voting behavior; recovery procedures; documentation of resource limits; and a published incident-response process. Any discretionary powers retained by OT must be explicit, narrow and temporary.

```
total_genesis_SHB = sum(initial_allocations) = 100,000,000; block_reward_SHB = 0
```

The intended block target remains **60 seconds**, corresponding to approximately **1,440 nominal block opportunities per day** if progress is uninterrupted. The network must not compromise safety merely to meet that cadence. A mainnet release should clearly disclose its practical throughput, finality

measurements, validator assumptions and limitations. No statement in this roadmap is a claim that such measurements have already been obtained.

Stage	Deliverable	Evidence required before advancing
Specification	Versioned state rules and genesis format	Deterministic test vectors and supply invariant
Local prototype	Multi-validator development network	Reproducible proposal-to-finality tests
Public testnet	Independently operated test network	Fault, latency, synchronization and governance reports
Security review	External code and protocol assessment	Published findings and remediation of critical issues
Mainnet candidate	Signed release and reviewed genesis	Independent replay, wallet safety and incident procedures
Mainnet	Public, verifiable SHB network	Transparent operational reporting and governance review

The table defines a proposed sequence rather than guaranteed delivery dates. Planning estimates may be published as development progresses, but failing a release gate should postpone the next stage rather than weaken the gate.

20.7 Gradual Decentralization

At the beginning, OT may need to operate development infrastructure and coordinate compatible software releases. This practical role must not be confused with permanent ownership of community decisions. Decentralization should be assessed by independent validation capacity, the diversity of validator operators, alternative client access, transparent software governance and the ability of participants to verify a final state without a favored website.

The proposed handover is incremental: publish node documentation, expand the set of independent operators, remove unnecessary release-key privileges, require documented approval procedures for protocol changes and report operational concentration. Responsibility must remain identifiable throughout the process. Replacing a known operator with an opaque committee does not automatically improve accountability.

Algorithm 20.1 — Mainnet release gate (conceptual)

```
function AssessMainnetReadiness(release, genesis, evidence):
    require Sum(genesis.initial_allocations) == 100000000_SHB
    require release.per_block_issuance == 0
    require release.nominal_block_seconds == 10
    require evidence.deterministic_replay_passed
    require evidence.vote_snapshot_tests_passed
    require evidence.finality_and_recovery_tests_passed
    require evidence.external_review_completed
    require not evidence.has_unresolved_critical_findings
    require evidence.independent_operators_verified
    require evidence.wallet_and_incident_procedures_published
    return APPROVE_RELEASE_CANDIDATE
```

This algorithm represents a review checklist, not an executable mainnet guarantee. Approval would authorize consideration of a release candidate; it would not replace operational observation or independent judgment.

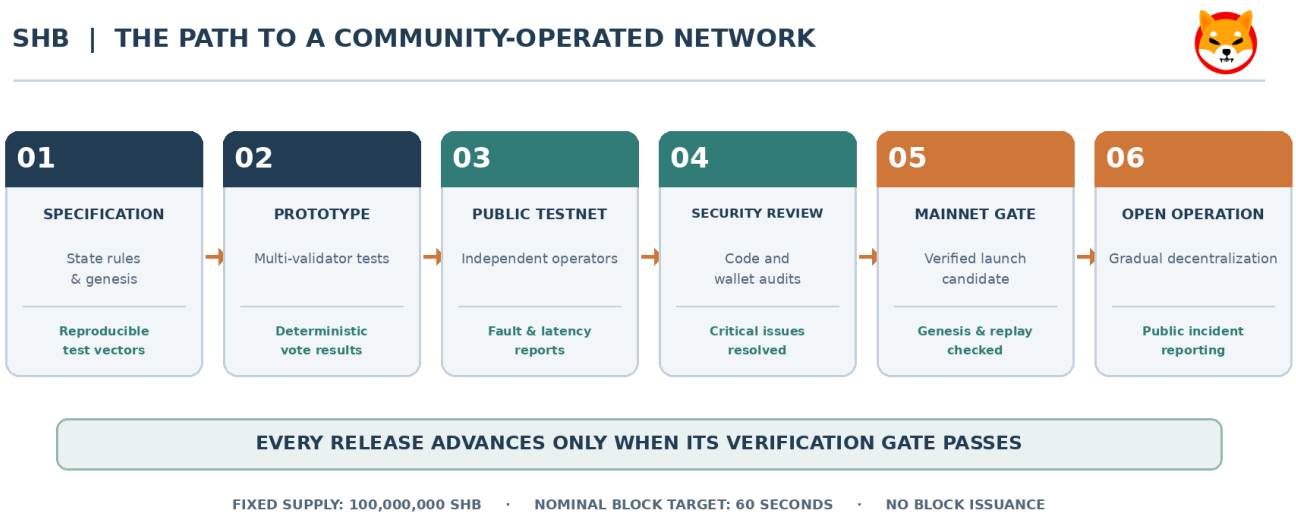
20.8 The Future of SHB

A mature SHB network would allow communities to adopt their own constitutions, create shared decision records and establish cooperation without depending on a single administrator to announce the winner. Its usefulness would be visible in mundane events: a proposal whose text cannot change after voting begins; a delegate whose authority can be revoked; an archived objection that survives the loss of a website; and an execution report tied to a decision that anyone can verify.

Future extensions may include broader voting methods, richer credential systems, federation between governance spaces, selective disclosure and improved interfaces for participants unfamiliar

with cryptographic tools. Each extension would need to preserve the distinction between **verifiable procedure** and **human legitimacy**. A mathematically correct tally can still embody a poor decision, and software cannot replace discussion, law or judgment.

FIGURE 20.1 FROM SPECIFICATION TO COMMUNITY-OPERATED NETWORK



Each stage is separated by an evidence-based release gate. All dates and phases are proposed from the perspective of February 2020.

SHB is proposed as a durable instrument for collective agency rather than a new way to spend money. The engineering program should be judged by whether its rules remain comprehensible under disagreement, its proofs remain checkable without privileged access and its communities can eventually proceed without the founding team deciding on their behalf. That is the intended destination. The roadmap is the work required to reach it.

References and Technical Foundations

[71] K. Scarfone, M. Souppaya, A. Cody and A. Orebaugh, *Technical Guide to Information Security Testing and Assessment*, NIST SP 800-115, September 2008. https://doi.org/10.6028/NIST.SP.800-115

[72] E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.3*, RFC 8446, August 2018. https://www.rfc-editor.org/rfc/rfc8446

REFERENCE MATERIAL / CONTENTS

Glossary, Technical Annexes & Bibliography

Reference information for readers, developers and independent reviewers.

The following material is part of the SHB technical whitepaper. It consolidates the vocabulary, mathematical rules and research foundations used across Chapters 1–20. All references are limited to material available by February 2020.

Appendix A — Glossary of SHB Terms

A set of 57 defined terms covering governance, validators, voting, identities, cryptographic proofs and network operation.

Appendix B — Protocol Mathematics and Cryptographic Notation

The fixed supply of 100,000,000 SHB, nominal one-minute block interval, voting calculations, BFT thresholds and validation invariants are stated in one place.

Appendix C — Bibliography and Source Concordance

A consolidated set of 45 technical and research references, together with a concordance for the 72 source numbers used throughout the manuscript.

The appendices describe the proposed protocol and its research basis; they do not establish that the software was deployed, audited or operational in February 2020.

APPENDIX A / GLOSSARY

Appendix A

Glossary of SHB Terms

A common vocabulary for the components, rights and limits of SHB.

This glossary defines the terms used throughout the SHB whitepaper. Definitions describe the February 2020 design proposal; they do not assert that the associated feature has been implemented, deployed or independently audited.

TERM	PROTOCOL MEANING
Account	An identifiable record whose balance, nonce and permissions are maintained in the SHB state.
Account nonce	A monotonically increasing value used to order signed instructions and reject replayed transactions.
Address	A protocol-defined identifier associated with a public key or account record.
Amendment	A versioned change to the text or parameters of a proposal before its ballot is frozen.
Ballot	A signed or cryptographically committed expression of an eligible participant's preference.
Ballot commitment	A hash-based commitment that binds a voter to a choice without revealing that choice immediately.
Block	An ordered group of valid transactions plus a header linking it to the preceding chain state.
Block height	An integer index locating a block in the finalized chain history.
Block interval	The nominal one-minute production cadence proposed for SHB; not a guarantee under faults.
Byzantine fault	Arbitrary or malicious behavior by a network participant, including inconsistent messages.
Byzantine fault tolerance (BFT)	Agreement methods that can maintain safety under a bounded number of Byzantine validators.
Commit–reveal	A two-stage ballot mechanism: publish a commitment first, then reveal choice and secret salt.
Community constitution	A versioned collection of local rules governing proposals, voting and institutional authority.
Consensus	The validator process that agrees on one ordered and finalized transaction history.
Delegation	Permission for a representative to exercise another holder's eligible voting weight within a specified scope.
Delegation cycle	A chain of delegated authority that ultimately returns to its originator; excluded from the initial direct model.
Deterministic execution	A rule under which honest nodes obtain identical results from identical prior state and inputs.
Digital signature	A cryptographic proof that an instruction was authorized by the controller of a signing key.
Electorate	The total set of accounts and voting weight eligible under a community's frozen rules.
Eligibility snapshot	A fixed state reference used to measure participating accounts and voting weights for a ballot.
Execution mandate	An approved and bounded instruction that records who may act, how and until when.
Finality	The condition under which an agreed block cannot be replaced without violating the assumed fault model.
Finality certificate	Verifiable evidence that a sufficient validator quorum committed to a particular block.
Federation	A cooperative arrangement between separate governance spaces without merging their constitutions.
Genesis block	The network's first block, establishing chain identity, the validator configuration and initial allocations.

TERM	PROTOCOL MEANING
Governance capture	Disproportionate control of decisions by a small holder group or colluding participants.
Governance space	An isolated community-specific set of roles, rules, proposals, ballots and mandates on SHB.
Hash function	A deterministic function that maps arbitrary data into a fixed-length digest.
Identity credential	An attestation of a property, such as membership or expertise, governed by an issuer and verification rules.
IPFS	A content-addressed peer-to-peer file system proposed in 2014, suitable for optional off-chain documents.
Key pair	A cryptographically linked private key and public key used to authorize and verify instructions.
Liquid democracy	A governance arrangement combining direct participation with revocable representation.
Mainnet	The intended production SHB chain, distinct from a disposable test network.
Mandate	A bounded authorization created after a decision and subject to recorded conditions.
Merkle root	A compact hash commitment to the leaves of a tree, allowing inclusion proofs.
Multi-signature	An authorization rule requiring signatures from multiple designated keys.
Off-chain evidence	Supporting documents or observations stored outside the ledger, optionally anchored by hashes.
On-chain record	State or event data committed through the SHB consensus process.
Peer	A network node that exchanges protocol data with other nodes.
Proposal	A uniquely identified, versioned question submitted for a governance decision.
Proposal sponsorship	A configured condition through which eligible supporters authorize a proposal to advance.
Pseudonymity	Participation through identifiers that do not, by themselves, disclose real-world identity.
Quorum	A minimum participation requirement measured against the relevant eligible electorate.
Replay attack	Reuse of a previously signed instruction in a context where it should no longer be valid.
Reputation record	An optional history of contributions and assessments, separate from SHB balance.
SHB	The proposed token that represents a basic unit of governance influence.
Snapshot height	The finalized block height at which eligibility and voting weights are determined.
State root	A cryptographic digest committing to a specified version of the complete protocol state.
Sybil attack	An attempt to gain influence by creating multiple participant identities.
Timelock	A mandatory delay between authorization and execution or activation.
Token supply	The entire SHB quantity represented in the ledger; fixed by proposal at 100,000,000 SHB.
Validator	A network participant that verifies, proposes and votes on blocks under the consensus rules.
Validator set	The active group authorized to contribute to block finality at a given height.
Vesting	A predeclared schedule restricting when allocated tokens become transferable or otherwise usable.
Vote weight	The number of eligible SHB units attributed to an account or valid delegate for one ballot.
Wallet	Software or hardware used to manage keys and create authorized SHB transactions.
Zero-knowledge proof	A cryptographic argument that can establish a statement without disclosing the associated witness; an optional research direction, not an initial SHB guarantee.

APPENDIX B / TECHNICAL ANNEX

Appendix B

Protocol Mathematics and Cryptographic Notation

The quantitative rules that should remain stable across implementations.

Notation is provided to make assumptions auditable, not to prescribe one programming language. Unless explicitly indicated otherwise, examples are illustrative and do not represent actual mainnet state.

B.1 Constants and symbols

SYMBOL	MEANING	INTERPRETATION
S_max	100,000,000 SHB	Fixed total token supply
T_block	60 seconds	Nominal block interval; actual latency may vary
h	Nonnegative integer	Finalized block height
Σ_h	Protocol state	Validated network state after block h
T_h	Ordered transactions	Instructions finalized in block h
F	State-transition function	Deterministic rules executed by full nodes
g	Governance space identifier	A community's local decision environment
p	Proposal identifier	Unique ballot/proposal reference
h_s	Snapshot height	Finalized height used to calculate voting weight
E_{g,p}	Eligible voting units	Total eligible SHB weight for proposal p in space g
w_{i,p}	Eligible weight of account i	Weight held at h_s after authorized constraints
Y, N, A	Vote totals	Yes, no and abstain respectively
Q_min	Quorum threshold	Minimum participation ratio fixed by the constitution
α	Approval threshold	Minimum yes-to-decisive-vote ratio
n, f	Validator counts	n validators, at most f Byzantine under the model
H(x)	Cryptographic hash	Commitment to bytes x under a specified algorithm
‖	Byte concatenation	Unambiguous, length-delimited encoding is required
σ	Digital signature	Signature verified against the authorizing key

B.2 Network supply and block production

SHB has one global supply invariant. No validator reward, governance action, community constitution or scheduled block is permitted to mint new SHB.

```
Σ balances(h) = S_max = 100,000,000 SHB     and     block_reward(h) = 0
```

Balances here include all genesis allocations, locked or vested allocations and any protocol-managed accounts; locking does not make token units disappear. Transfers alter ownership, never global supply.

```
nominal_blocks_per_day = 86,400 / 60 = 1,440
```

This is a cadence estimate, not a finality-time guarantee. During a partition or validator outage the protocol may advance more slowly, or stop safely.

B.3 Voting weights, participation and approval

For proposal p in governance space g, calculate each eligible weight from one immutable snapshot. A token must never be counted twice because it was transferred, pledged or delegated after that snapshot.

```
E(g,p) = Σ(i ∈ eligible(g,p,h_s)) w(i,p)    with    0 ≤ E(g,p) ≤ 100,000,000
turnout = (Y + N + A) / E(g,p)    ;    approval = Y / (Y + N)
```

For a decision to pass, the ballot must satisfy its constitutionally defined turnout threshold and approval threshold. Abstentions count toward participation only if the governance rule says so, and they do not enter the decisive yes/no denominator in this illustrative convention.

ILLUSTRATIVE BALLOT	SHB WEIGHT
Eligible voting weight	2,000,000
Yes	420,000
No	180,000
Abstain	100,000
Turnout	700,000 / 2,000,000 = 35%
Approval of decisive votes	420,000 / 600,000 = 70%
Example ballot rules	25% quorum and two-thirds approval
Illustrative result	Passes both conditions

The example is not a protocol-wide voting rule. Each community's valid constitution may establish a different turnout denominator and decision threshold before the proposal opens.

B.4 Delegation and conservation of voting influence

The initial direct-delegation model assigns voting power to either the owner or one authorized representative for a given scope and snapshot. The same SHB units cannot support both ballots.

```
Σ(direct_weight_i + delegated_weight_i) ≤ E(g,p)
```

When an owner revokes delegation during an active ballot, the protocol must follow the ballot's frozen revocation and snapshot rules. Retroactive recomputation of already finalized outcomes is prohibited.

B.5 BFT threshold and deterministic state

For a conventional equal-weight BFT validator set, the model admits at most f Byzantine validators among n ≥ 3f + 1. A quorum of 2f + 1 ensures any two quorums share at least f + 1 validators under the same fixed configuration.

```
n ≥ 3f + 1    ;    q = 2f + 1    ;    |Q₁ ∩ Q₂| ≥ f + 1
Σ_(h+1) = F(Σ_h, T_(h+1))
```

Weighted validator sets require their own quorum proofs and safe membership-change procedure. These expressions are a model for review, not a substitute for a complete, audited consensus implementation.

B.6 Proposals, commitments and domain separation

All signatures and hashes require a documented canonical byte encoding. The chain identifier, transaction type and proposal reference are included where appropriate to prevent cross-chain and cross-purpose replay.

```
proposal_id = H(chain_id || space_id || author || nonce || body_hash || rules_version)

ballot_commitment = H(chain_id || proposal_id || voter || choice || random_salt)
```

A commitment hides a choice until reveal, provided the salt has adequate entropy. Once the reveal is public, it does not provide enduring ballot secrecy or resistance to coercion. The ledger also cannot prove whether submitted supporting evidence is factually correct.

B.7 Minimal verification checklist

The following conceptual checklist is intended to guide prototype tests. It is not executable source code and makes no claim of formal verification.

```
function CheckGovernanceProposal(proposal, snapshot, ballots):
    require proposal.chain_id == local_chain_id
    require proposal.rules_version == frozen_rules_version
    require finalized(snapshot.height)
    require all(signature_valid(b) and eligible(b, snapshot) for b in ballots)
    require no_duplicate_units(ballots, snapshot)
    require sum_weight(ballots) <= eligible_weight(snapshot)
    return evaluate_thresholds(ballots, proposal.rules_version)
```

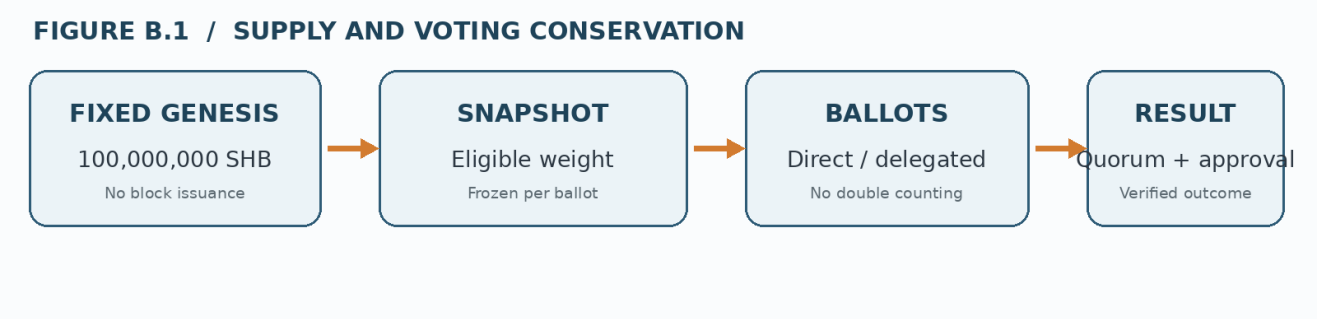


Figure B.1 — The chain-wide token supply is constant; community-specific eligibility and ballots redistribute voting influence but never create SHB.

APPENDIX C / BIBLIOGRAPHY

Appendix C

Bibliography and Source Concordance

A consolidated bibliography limited to material available by February 2020.

The preceding chapters use sequential source labels [1]–[72]. Several works appear more than once because the chapters were drafted separately. The consolidated bibliography below identifies each distinct work only once as [R01]–[R45], followed by a concordance that maps every chapter citation to its canonical record. A date below refers to the cited version or its original publication; online landing pages may later have been updated.

The bibliography distinguishes published standards from research papers, working drafts and community proposals. In particular, decentralized identifiers were represented by a W3C Working Draft in November 2019 and must not be described as a completed W3C Recommendation in this February 2020 document.

Distributed ledgers

[R01] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. https://bitcoin.org/bitcoin.pdf

[R02] V. Buterin. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. 2014. https://ethereum.org/whitepaper/

[R04] L. Lamport, R. Shostak and M. Pease. The Byzantine Generals Problem. ACM TOPLAS 4(3), 1982. https://lamport.azurewebsites.net/pubs/byz.pdf

[R06] L. M. Goodman. Tezos: A Self-Amending Crypto-Ledger. White paper, 2014. https://tezos.com/whitepaper.pdf

[R07] M. Castro and B. Liskov. Practical Byzantine Fault Tolerance. OSDI, 1999. https://www.usenix.org/conference/osdi-99/practical-byzantine-fault-tolerance

[R09] E. Buchman, J. Kwon and Z. Milosevic. The Latest Gossip on BFT Consensus. arXiv:1807.04938, 2018. https://arxiv.org/abs/1807.04938

[R10] J. Benet. IPFS — Content Addressed, Versioned, P2P File System. arXiv:1407.3561, 2014. https://arxiv.org/abs/1407.3561

[R13] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta and I. Abraham. HotStuff: BFT Consensus with Linearity and Responsiveness. PODC, 2019. https://doi.org/10.1145/3293611.3331591

[R14] C. Dwork, N. Lynch and L. Stockmeyer. Consensus in the Presence of Partial Synchrony. JACM 35(2), 1988. https://groups.csail.mit.edu/tds/papers/Lynch/jacm88.pdf

[R43] D. Yaga, P. Mell, N. Roby and K. Scarfone. Blockchain Technology Overview. NISTIR 8202, October 2018. https://doi.org/10.6028/NIST.IR.8202

Governance and voting

[R03] J. Behrens et al. LiquidFeedback project and principles of delegable decision-making; initial releases in 2009. https://liquidfeedback.com/en/history.html

[R21] M. Schulze. A New Monotonic, Clone-Independent, Reversal Symmetric, and Condorcet-Consistent Single-Winner Election Method. Social Choice and Welfare 36, 2011. https://doi.org/10.1007/s00355-010-0475-4

[R22] B. Adida. Helios: Web-based Open-Audit Voting. USENIX Security Symposium, 2008. https://www.usenix.org/legacy/event/sec08/tech/full_papers/adida/adida.pdf

[R23] M. R. Clarkson, S. Chong and A. C. Myers. Civitas: Toward a Secure Voting System. IEEE Security and Privacy, 2008. https://doi.org/10.1109/SP.2008.32

[R24] B. Ford. A Liquid Perspective on Democratic Choice. Draft, November 2018. https://bford.info/pub/soc/liquid/

[R25] A. Kahng, S. Mackenzie and A. D. Procaccia. Liquid Democracy: An Algorithmic Perspective. AAAI, 2018. https://doi.org/10.1609/aaai.v32i1.11468

[R27] J. Benaloh et al. End-to-End Verifiability. arXiv:1504.03778, 2015. https://arxiv.org/abs/1504.03778

[R38] P. Rossini and J. Stromer-Galley. Citizen Deliberation Online. Oxford Handbook of Electoral Persuasion, published online 2019. https://doi.org/10.1093/oxfordhb/9780190860806.013.14

[R39] G. W. Brier. Verification of Forecasts Expressed in Terms of Probability. Monthly Weather Review 78(1), 1950. https://doi.org/10.1175/1520-0493(1950)078(0001:VOFEIT)2.0.CO;2

Standards and cryptography

[R05] F. Vogelsteller and V. Buterin. ERC-20 Token Standard, EIP-20; proposal created in 2015. https://eips.ethereum.org/EIPS/eip-20

[R08] R. C. Merkle. A Digital Signature Based on a Conventional Encryption Function. CRYPTO 1987.

[R11] S. Josefsson and I. Liusvaara. Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032, January 2017. https://www.rfc-editor.org/rfc/rfc8032

[R12] C. Bormann and P. Hoffman. Concise Binary Object Representation (CBOR). RFC 7049, 2013. https://www.rfc-editor.org/rfc/rfc7049

[R15] P. Wuille and G. Maxwell. Base32 Address Format for Native v0-16 Witness Outputs. BIP 173, 2017. https://bips.dev/173/

[R16] M. Palatinus, P. Rusnak, A. Voisine and S. Bowe. Mnemonic Code for Generating Deterministic Keys. BIP 39, 2013. https://bips.dev/39/

[R17] W3C. Web Authentication: An API for Accessing Public Key Credentials Level 1. Recommendation, March 2019. https://www.w3.org/TR/webauthn-1/

[R18] W3C. Verifiable Credentials Data Model 1.0. Recommendation, 19 November 2019. https://www.w3.org/TR/2019/REC-vc-data-model-20191119/

[R20] B. Laurie, A. Langley and E. Kasper. Certificate Transparency. RFC 6962, 2013. https://www.rfc-editor.org/rfc/rfc6962

[R26] D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM 24(2), 1981. https://doi.org/10.1145/358549.358563

[R28] S. Bradner. Key Words for Use in RFCs to Indicate Requirement Levels. RFC 2119, 1997. https://www.rfc-editor.org/rfc/rfc2119

[R29] E. Barker. Recommendation for Key Management, Part 1: General. NIST SP 800-57 Part 1, Rev. 4, January 2016. https://doi.org/10.6028/NIST.SP.800-57pt1r4

[R31] W3C. Decentralized Identifiers (DIDs) v1.0: Core Data Model and Syntaxes. First Public Working Draft, 7 November 2019 (not a Recommendation). https://www.w3.org/TR/2019/WD-did-core-20191107/

[R33] W3C. ActivityPub. Recommendation, 23 January 2018. https://www.w3.org/TR/2018/REC-activitypub-20180123/

[R34] C. Adams et al. Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). RFC 3161, 2001. https://www.rfc-editor.org/rfc/rfc3161

[R37] W3C. PROV-O: The PROV Ontology. Recommendation, 30 April 2013. https://www.w3.org/TR/2013/REC-prov-o-20130430/

[R40] JSON-RPC Working Group. JSON-RPC 2.0 Specification. 2010; updated 2013. https://www.jsonrpc.org/specification

[R41] T. Bray. The JavaScript Object Notation (JSON) Data Interchange Format. RFC 8259, 2017. https://www.rfc-editor.org/rfc/rfc8259

[R45] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018. https://www.rfc-editor.org/rfc/rfc8446

Security, law and institutions

[R19] C. Dwork and M. Naor. Pricing via Processing or Combatting Junk Mail. CRYPTO 1992.

[R30] E. Ostrom. Governing the Commons: The Evolution of Institutions for Collective Action. Cambridge University Press, 1990.

[R32] European Parliament and Council. Regulation (EU) 2016/679 (GDPR), adopted 2016; applicable from 25 May 2018. https://eur-lex.europa.eu/eli/reg/2016/679/oj

[R35] L. Luu et al. Making Smart Contracts Smarter. ACM CCS, 2016. https://doi.org/10.1145/2976749.2978309

[R36] N. Atzei, M. Bartoletti and T. Cimoli. A Survey of Attacks on Ethereum Smart Contracts (SoK). POST, 2017. https://doi.org/10.1007/978-3-662-54455-6_8

[R42] J. R. Douceur. The Sybil Attack. International Workshop on Peer-to-Peer Systems, 2002. https://www.microsoft.com/en-us/research/publication/the-sybil-attack/

[R44] K. Scarfone, M. Souppaya, A. Cody and A. Orebaugh. Technical Guide to Information Security Testing and Assessment. NIST SP 800-115, 2008. https://doi.org/10.6028/NIST.SP.800-115

C.1 Concordance with chapter citation numbers

The left-hand identifiers refer to the original chapter references [1]–[72]. The right-hand identifiers point to the deduplicated records above. The sequence of chapter citations is unchanged.

MASTER ID	CHAPTER REFS	MASTER ID	CHAPTER REFS
R01	[1], [5], [27]	R24	[39]
R02	[2], [6]	R25	[40]
R03	[3], [8], [12], [38]	R26	[41]
R04	[4]	R27	[43]
R05	[7], [28]	R28	[46]
R06	[9], [45]	R29	[47]
R07	[10], [17], [18]	R30	[48], [52]
R08	[11]	R31	[50]
R09	[13], [19], [29], [34]	R32	[51], [59], [70]
R10	[14], [31], [54], [57]	R33	[53]
R11	[15], [22], [32], [66]	R34	[58]
R12	[16]	R35	[60]
R13	[20]	R36	[61]
R14	[21]	R37	[62]
R15	[23]	R38	[63]
R16	[24]	R39	[64]
R17	[25]	R40	[65]
R18	[26], [49], [55]	R41	[67]
R19	[30]	R42	[68]
R20	[33], [44], [56]	R43	[69]
R21	[35]	R44	[71]
R22	[36], [42]	R45	[72]
R23	[37]		

C.2 Editorial cut-off and limitations

The cutoff for cited technical material is February 2020. Historical publications are cited for the concepts they establish; they should not be read as endorsements of SHB. The design distinguishes protocol aspirations from tested software and recognizes that a cryptographic proof of a recorded vote is not proof of voter independence, equal representation, factual truth or legal validity.

For implementation, each cited algorithm, RFC and working draft would require an exact version, security review and compatibility assessment. SHB has not claimed security certification, regulatory authorization or operational deployment in this edition.

END OF WHITEPAPER / OT DEVELOPMENT TEAM / ROMANIA / FEBRUARY 2020