



LIGHT · FAST · TRUST

TrustNote

The TrustME-PoW Consensus Scheme:

Decentralized, Network Partition Tolerance, Movable

TrustNote Institute of Technology

April 2018

TrustNote

Disclaimer

TrustNote Institute of Technology and Research & Development section hereby declare that, this package is under MIT open source software license and this software distributed without any warranty. TrustNote Institute of Technology declares that we are **NOT** responsible for direct, indirect, incidental, or consequential damages resulting from any defect, error, or failure to perform. This package is **experimental** and a **work-in-progress**, use at your own risk. The contents of this report are in implementation phase, thus TrustNote can update (add/remove packages) any time without informing the users. Finally, we declare that, TrustNote White paper and all other technical reports related to TrustNote **only** can be accessed from:

△ <https://github.com/trustnote/document>

△ <https://trustnote.org/>

We do not guarantee the faulty or misleading data available in documents downloaded from any other website rather than two official websites introduced above.

Contact Us

Business Enquires: foundation@trustnote.org

Technical Support: community@trustnote.org

Copyright

© 2018 TrustNote Institute of Technology. All rights reserved.

Contents

Glossary.....	1
1. Introduction	1
1.1 Node Taxonomy and Topology.....	2
1.2 TrustME-PoW Scheme Overview	2
1.3 Report Organization.....	5
2. Super Node	5
2.1 Motivation	5
2.2 Methodology	5
2.3 Deposit Mechanism	6
3. Main Chain	8
3.1 Basic Concept.....	8
3.2 Best Parent and Main Chain Determination.....	9
3.3 Main Chain Index.....	10
3.4 Main Chain Stabilization Algorithm	11
4. TrustME-PoW Consensus	13
4.1 Motivation	13
4.2 How to Select Attestors	13
4.3 PoW Unit.....	14
4.4 Consensus Round Switching	16
4.5 Equihash Difficulty Calculation	18
4.6 TrustME unit	18
4.7 Attestation Reward	20
5. Switching from Witnesses to TrustME-PoW	21
5.1 Overview	21
5.2 Procedure	21

Glossary

- ▲ **Node:** Refers to any active user, installed TrustNote client (any devices such as phone, pc, IoT, etc.) and having a valid wallet address.
- ▲ **Unit:** Refers to a data structure which contains many messages generated by the nodes including: Transactions messages, text messages and etc.
- ▲ **Full Node:** Refers to Cloud Host Server/Workstation, and PC, which maintaining synchronization and verification of ledger data.
- ▲ **Super Node:** Refers to Mining Systems, Cloud Host Server/Workstation, and PC, which generates a deposit contract and paying the deposit, and running the TrustME-PoW mining program.
- ▲ **Parent Unit:** Refers to units generated at an earlier time and Child Units can reference them.
- ▲ **Child Unit:** Refers to units generated at a later time and referencing one or more parent unit.
- ▲ **MC:** A single chain along Child-Parent links within the DAG which is determined by applying the Parent Selection Algorithm recursively.
- ▲ **MCI:** Main Chain Index.
- ▲ **Attestor:** Refers to a Super node, which participates in a round of consensus and successfully obtains Attestation power.
- ▲ **PoW Unit:** Refers to unit containing Equihash solution.
- ▲ **TrustME unit:** Refers to unit, used to determine the MC and its first message is a TrustME message.
- ▲ **Micro-Node:** Refers to client running on Microcontrollers and Smart Cards.
- ▲ **Light Node:** Refers to client running on Smartphone and Tablet PC.
- ▲ **Fine-grained PoW consensus:** Each Super Node independently starts PoW, and there is no direct bound between nodes.
- ▲ **Coarse-grained PoW consensus:** The Super Node periodically starts PoW. Each round selects a certain number of Super Nodes as Attestors. These Attestors only belong to that specific round. Once that round finished, the Attestors automatically lose their Attestation power.
- ▲ **Silent-locking time:** Refers to the period of time when, a super node doesn't send any attestation unit.

1. Introduction

TrustNote is a minable public DAG-ledger with an innovative, two-tier consensus mechanism designed to be "lightweight, efficient and trustworthy". Such two-tier consensus mechanisms can improve transaction throughput and reduce transaction confirmation delay, which effectively solving the problem of "Excessive Bifurcation" and "Double Spending". TrustME-PoW enables support for high concurrency transactions, fast transaction confirmation, and decentralized transaction unit strict sequencing mechanism. Even more, it also provides an important capability for TrustNote to support advanced declarative smart contracts and Micro-Nodes.

This report explains fundamental protocols governing the TrustNote platform, such as, super nodes management protocol, Main chain selection and best parent selection, Main chain stabilization protocol, and etc. After introducing all the prerequisites, TrustME-PoW consensus mechanism will be fully explained in detail.

1.1 Node Taxonomy and Topology

The TrustNote network supports four types of nodes: Super-nodes, Full-nodes, Light-nodes, and Micro-nodes; a comparison between these four types of nodes presented in the table below.

Table 1-1 Comparison of four types of node

	Super Node	Full Node	Light Node	Micro-Node
ledger	full ledger	full ledger	light ledger	N/A
transaction	✓	✓	✓	commissioned
DAG consensus	✓	✓	indirect	×
TrustME-PoW	✓	×	×	×
TrustME-BA	✓	×	×	×
Hosting Micro-Node	✓	×	×	×
deployment	+ Mining Systems + Cloud Host + Server/Workstation + PC	+ Cloud Host + Server/Workstation + PC	+ Smartphone + Tablet PC	+ MCU + Smart Card

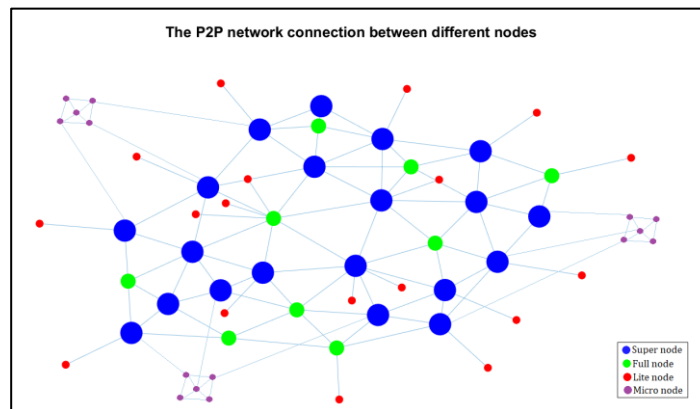


Figure 1-1 P2P network Connection diagram between nodes

1.2 TrustME-PoW Scheme Overview

- △ Decentralized Attestor selection among super nodes.
- △ Users who want to become Super Nodes are required to create the deposit contract and pay the deposit. Full Nodes can become Super Nodes and join TrustME-PoW by paying **500,000** MegaNote to their deposit contracts which would be created by themselves. Once the silent-locking time be over, the deposited TTT will be returned to the super node specified address. Super nodes won't lose their deposit under any circumstances. Any malicious unit or suspicious behavior would only extends the silent-locking time.
- △ Super Node Clients are enjoying the premium package.

- ▲ Super Node Clients will be running multi-threaded platform (Full ledger thread, Super Node thread, Attestation thread, etc.). see figure 1-3
- ▲ Full ledger thread is used to synchronize and verify the generated units.
- ▲ Super Node thread let the Super Node clients to run Equihash. Once the super node thread finds the Equihash solution and after a specific procedure, they will receive Attestation power.
- ▲ Equihash solutions will be filtered by Difficulty filter. Also, Difficulty Adjustment performed by modified version of Digishield v3/v4 algorithm.
- ▲ The Main Chain selection algorithm, best parent selection and MC stabilization are triggered by the generation of each unit, to provide a high throughput for transactions confirmation while it preventing from creation of any forks and double spending.
- ▲ The PoW units and other units will be stabilized after the MC is determined and stabilized.
- ▲ The Attestors selection protocol is triggered once the PoW units are stabilized.
- ▲ Attestation thread will be started once Super Node client can establish a stable unit (containing Equihash solution) on the MC of DAG-ledger and its result would be evaluated as high performer.
- ▲ Attestation rewards are provided for Super Node clients who are having an active Attestation thread in a consensus round. Calculation of the rewards are according to the share of issued stable TrustME units located on MC.

TrustME-PoW Protocol Sketch

Full Ledger Thread

On demand from the user to generate a new unit:

- (a) The best parent algorithm, determines the best parents of the new unit.
 - a. Selects some childless units by random and finds the best parents by comparing their Attestation level, the higher Attestation level will be selected as the best parent.
 - b. If the candidate best parent remained from (a) are more than one, compares their unit level.
 - c. If the candidate best parent remained from (b) are more than one, compares their unit hash.
- (b) The unit will be generated and it will be sent to the DAG-ledger of the user who generated this message.
- (c) The new unit will be gossiped to the peers DAG-ledger.
- (d) The unit validation will be triggered by arrival of the new unit.
- (e) The childless units list and best parent units list will be updated.
- (f) The MC will be determined based on the best parents and Attestation units.
- (g) The units lie on MC will be stabilized and the normal units will be stabilized once the MC units are stable
- (h) The double spend and forks are prevented based on the MCI.

On arrival of the new units

- (a) Verify whether the unit hash is wrong or right.
- (b) Verify that the header fee is correct.
- (c) Verify whether message array is empty or not.
- (d) Check duplicated units.
- (e) Validate message hash tree.
- (f) Validate that the parent nodes are reasonable.
- (g) Validate Authors and verify the signatures.
- (h) Check if the unit is a TrustME unit:
 - a. If it's a TrustME unit, determine and update the MC.

On stabilization of a unit:

- (a) Validate and verify the message/s.
- (b) Verify the UTXO (if the Node has enough TTT or not).
- (c) If the unit fails at any steps the unit header will be stored on the DAG-ledger but the message will be.
- (d) Check if the unit is a PoW unit:
 - ▲ If it's a TrustME unit, update the Attestor list.

On demand from user:

- (a) Create and gossip a deposit unit.
- (b) Wait the deposit unit to become stable.
- (c) Send or wait for another node/s to send enough TTT to the deposit address.
- (d) If there are enough TTT in the deposit address starts up the super node thread.

For each transaction about deposit:

- (a) If the Super node generated any invalid unit.
 - Check silent-locking time
 - payout_address
 - reward_receiver_address

- (b) If the super node, didn't generate any TrustME unit for a certain time.
- (c) If any of the above fails, the silent-locking time will be reset.

For each PoW unit created:

- (a) Check the PoW solution generated by other Super Nodes.
- (b) Wait for twenty PoW solution.
- (c) Determine the Attestors' priority in next consensus round.

Super Node Thread

At the beginning of each consensus round:

- (a) Calculate the Public seed based on:
$$Seed_i = \text{blake2}_{256}(Seed_{i-1} || \text{Coinbase}_{i-2} || \text{First Stable MC unit}_{i-1})$$
- (b) Generate the node specific seed using public key.
$$\text{Public_Seed}^i = \text{blake2}_{256}(\text{Key}_{\text{public}} || \text{Seed}_i)$$
- (c) Calculate the [difficulty](#).
- (d) Run Equihash and get the solution ([TR-2018-01](#)).
- (e) Generate and gossip [PoW Unit](#).
- (f) Super Node starts the procedure from the beginning again to obtain the opportunity to take part in more consensus rounds.

When the PoW Unit becomes stable:

- (a) Check if the Super node will be on top **eighteen** list:
 - ▲ If on the list, it will receive the Attestation power.
 - ▲ Determine its own priority.
 - ▲ The Attestor thread will be triggered.

Attestor Thread

At any time:

- (a) Generates the [TrustME unit](#) according to its priority and should starts sending messages on its turn within five seconds.
- (b) Check if there are enough Attestors selected for the next round.
- (c) If there is not goes to (a).
- (d) If there are enough Attestors for next consensus round, it will be off the list.

When the next consensus round Attestors is determined:

- (a) Calculate the [Attestation rewards](#) for Attestors of previous consensus round.
- (b) Search among the Attestors of current consensus round with the priority equal to one.
- (c) The first Attestor of current consensus round sends out the first TrustME unit containing the [Coinbase message](#).

If all the normal units are stable:

- (a) Don't generate new TrustME unit until new normal units arrive.

Figure 1-2 TrustME-PoW Protocol Sketch

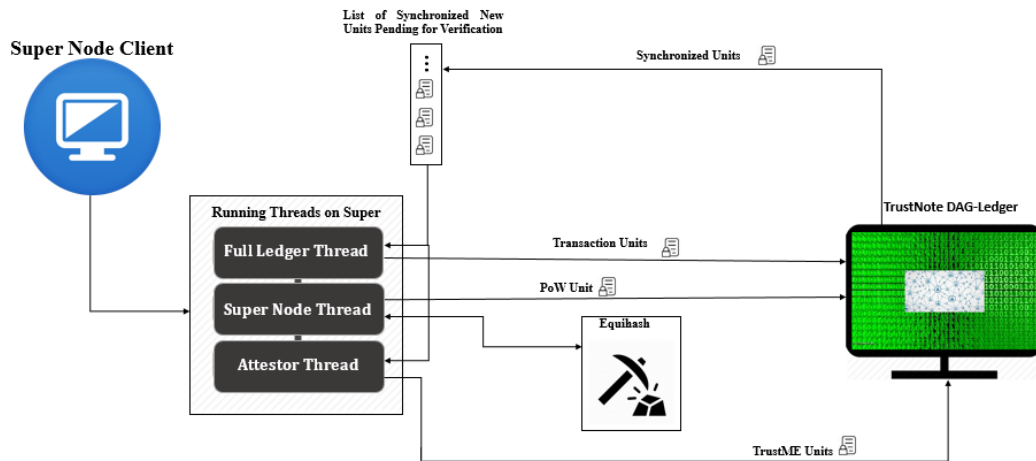


Figure 1-3 Illustration of the underlying mechanism of TrustME-PoW

1.3 Report Organization

The remainder of this technical report is organized as follows. In chapter 2, the node taxonomy, network topology and importance of Super nodes will be explained. Also, more detail about how a user can become a super node and the procedure the user should be through explained. Then, we will explain why the user should pay the deposit to become a Super Node and the procedure the user would be through to get it back. Chapter 3 is dedicated to Main Chain selection and stabilization algorithm, and the best parent protocol and its importance in MC stabilization and determination will be explained. In chapter 4 the innovative TrustME-PoW consensus with all detail will be explained. Even more, the reason why we have Attestors, what Attestors are doing, how Attestors benefit from helping the maintenance of the DAG-ledger, the switching procedure between rounds and so on are carefully explained. The procedure of calculation of the rewards and the procedure of sending rewards to Attestors also explained in this chapter. Finally, in chapter 5 the required steps that TrustNote should be taken to upgrade the network from Witnesses mode to TrustME-PoW mode fully explained.

2. Super Node

2.1 Motivation

Why do we need super node?

- △ Select Attestor in a decentralized way.
- △ Verification and analysis of the new units.
- △ Increasing the total computing power of the whole network.
- △ The more super nodes, the higher security of network.
- △ Super nodes and adoption of a Coarse-grained network algorithm make the whole network, partition tolerant.

2.2 Methodology

Super nodes playing a key role in verification of the transactions, selection of parent units,

transaction units sequencing, and micro-node protocols. In order to encourage nodes with high computing storage and network resources to become Super nodes, TrustNote has a total bonus of **500,000,000 MN**. The total amount of bonuses is equal to the total circulation in the previous period. These bonuses prized during 30 years to those Super Nodes who will help in maintenance of the TrustNote DAG-ledger. The Attestation Reward Policy allocates 6.79% of Total Attestation Rewards for the first year, after which the yearly allocated Attestation Rewards decays year by year according to Figure 2-1, of these, 90% of Attestation Rewards are allocated to the Super Nodes who provide valid Attestation Units, 10% of Attestation Rewards are allocated to TrustNote Foundation to support community operations, project incubation, and rewards to contributors etc.

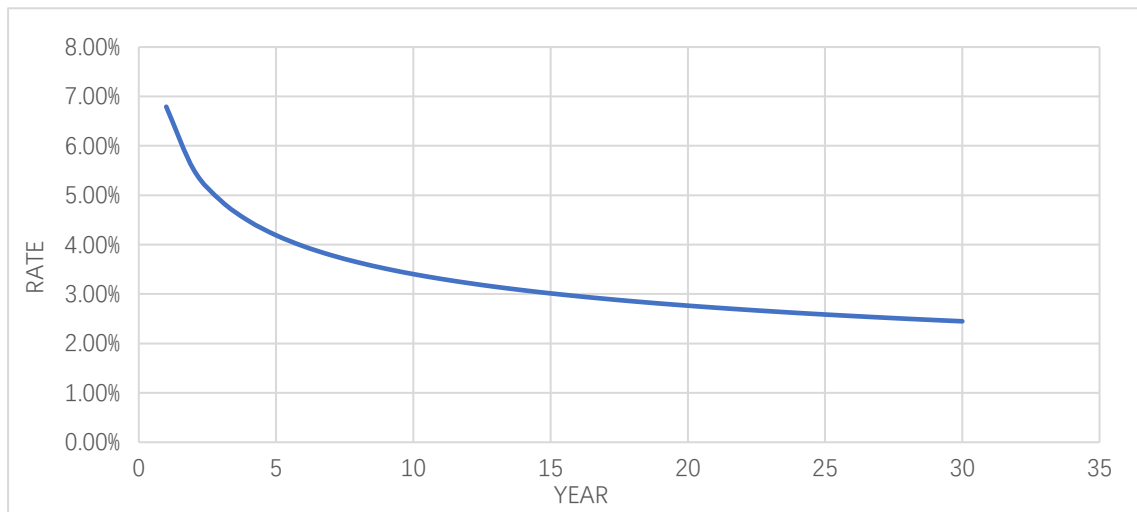


Figure 2-1 Attenuation Chart of Attestation Rewards

Full nodes can become Super node, while light nodes and micro-nodes cannot become Super nodes. To become a Super node, first it is required to download and install the Super node client program, then create the deposit contract and pay it, also it needs to assign the address that the deposit will goes to it later. After the Super Node obtains the Attestation power, it must synchronize and verify the rapid growth of the DAG-ledger data, while participating in the competition for the next round of Attestation at the same time. Even more, it will become the agent of the micro node in the future. Therefore, Super nodes need to have high computing performance. Super Node resource consumption:

- △ Synchronous DAG-ledger consumes network, memory, and hard disk IO resources.
- △ Verification of ledger consumes CPU, memory, and hard disk IO resources.
- △ PoW consumes RAM and CPU/GPU resources.
- △ Database operations consume hard disk IO, CPU, and RAM resources.

2.3 Deposit Mechanism

The Super node can generate normal units and TrustME units. The address of the generated TrustME unit must be the same as the one used to generate the deposit contract. There is only one Super node address for each deposit contract. If a Super node wants to have more than

one competing Attestors' address (such as a mining pool), it needs to generate multiple deposit contracts and pay them. By considering the safety issues of the Super Node itself and security of the network, the deposit required for the Super Node assigned as **500,000 MN**.

The main purpose of the deposit is to guarantee the trustworthy of the super nodes, if any super node tries to violate the rules the deposit value will be reduced; also, it reduces the malicious node's attack on the consensus mechanism. When there is a sufficient number of TTT in the deposit contract address issued by a certain node, the node can immediately participate in the TrustME consensus to obtain Attestation power in certain rounds, send Attestation units and then get its Attestation rewards. In order to prevent the loss of the Super nodes' deposit and the witch attack due to hacking, the Super nodes deposit has a silent locking time (SLT) when the Super nodes' deposit is frozen in a smart contract. The Super node deposit silent locking time is 17,280 consensus rounds. If the Super node has not sent any TrustME unit or wrong unit within the lock time, all the TTT in the deposit contract can be transferred to the specific address assigned by that Super node in the smart contract, otherwise the silent-locking time reset.

Super nodes are always online and vulnerable to attack. Therefore, a shorter consensus round duration should be set. If it is set to 5 minutes, the probability of the Attestor node being invalid or being hacked is reduced significantly. In order to improve the security of the Super Node funds, the deposit contract sets the deposit payable address and the Attestation bonus receive address. Super Node should set different addresses as the deposit payable address, payout address, and the Attestation bonus. Even more, Super Nodes are required to use the cold wallet to set the Attestation bonus receive address to increase the security of the Super node deposit and Attestation bonus.

Deposit Contract Message

```
messages: [{
  app: 'Deposit',
  payload_location: 'inline',
  payload_hash: 'hash of payload',
  payload: {
    silent-locking time: '17280',
    payout_address: 'Wallet Address of Node',
    reward_receiver_address: 'Wallet Address of Node'
  }
}]
```

3. Main Chain

In this section it is explained that:

- △ What the Main Chain is and introducing some basic concepts about it.
- △ How the main chain is determined and what the best parent is.
- △ How we determine the best parent and what the procedure for main chain stabilization is and more.

TrustNote does not have a block limitation. Any node can generate a new unit (e.g. a transaction) at any time and broadcast it to other nodes. The unit can select one or more previous units as parent units, each new unit validates its own parent units and adds the parent units' hash value to its own header. In this manner, the parent-child relationship between them can be expanded into a directed acyclic graph. DAG is a finite directed graph with a topological ordering (a sequence of the vertices such that every edge is directed from earlier to later in the sequence). The Acyclic property of the graph means there is no way to start at any unit U and follow a sequence of vertices that finally loops back to U again.

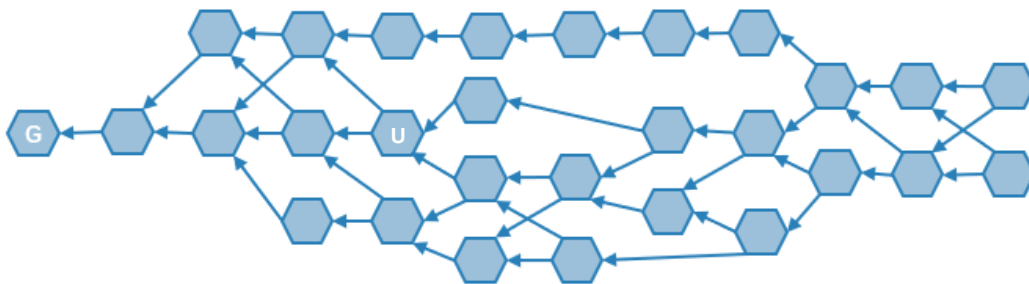


Figure 3-1 TrustNote simplified DAG-ledger

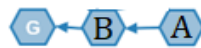
Assume an attacker wants to modify a unit. The attacker will be successful if it would have conspiracy with all other nodes to gossip the new ledger which containing the modified unit; which with the existence of honest nodes it's impossible. Even more, assuming that the digital signature is unforgeable, the greater the number of nodes on the entire network, the more difficult it is for an attacker to tamper a unit. Consequently, it's about to impossible for an attacker to do so.

3.1 Basic Concept

- △ **Unit Inclusion:** If "A" verifies "B", the unit "A" header contains the hash of the unit "B", so we say the unit "A" includes the unit "B". This inclusion relationship between the nodes is represented by a directed arrow between edges "A" and "B" on the DAG. If unit "A" includes unit "B" and the minimum path between the two units is "1", then it is said that unit "A" includes unit "B" directly; if unit "A" includes unit "B" and the minimum path between the two units is greater than "1", then unit "A" includes the unit "B" indirectly. Examples of unit inclusions are presented in figure 3-2 (a) and (b) below.
- △ **Parent Units and Child Units:** If unit "A" directly includes unit "B", we say unit "A" is the child of unit "B", and unit "B" is a parent of unit "A".

- ▲ **Genesis Unit:** The genesis unit has no parent and is the first unit in TrustNote DAG ledger which indicated with the letter "G" in figure 3-2.
- ▲ **Childless Unit:** The units without child are called childless units. Unit "A" in figure 3-2 is an example of a childless unit.
- ▲ **Sequential Units:** Any two desired units are defined as sequential units if they would have a directed path connecting those units, otherwise they are called non-sequential units. All the units sent by an Attestors from a certain address must be sequential units, otherwise they will be eliminated from the DAG.

a)



b)



Figure 3-2 Examples of inclusion relationship between units: a) Unit "A" verifies and contains unit "B" directly Unit "B" is the parent of unit "A". b) Unit "A" contains unit "B" indirectly and unit "A" is called a childless unit.

- ▲ **Unit Level:** The unit level is defined as the longest path length from that unit to Genesis unit. For instance, the Unit level of unit "A" in figure 3-2 (b) is 3
- ▲ **Attestation Level:** To determine the Attestation Level for any unit labelled as starting unit, follow the path along [best parent chain](#), until finding more than half of all Attestors' Attestation unit along the path. Then calculating the unit level of the stop unit, this value is the Attestation Level of the starting unit. The genesis unit is created by all initial Attestors, so it's the best parent naturally.

3.2 Best Parent and Main Chain Determination

Each unit (except Genesis unit) in DAG ledger has one or more parent units. According to a consistent algorithm, we can select the best parent units among the available parent units. The best parent unit selection algorithm is showed as below:

1. Start from a unit.
2. Selecting the parent unit with the highest Attestation level as its best parent unit.
3. If there are multiple candidate units, the unit with the lowest unit level will be selected as the best parent unit;
4. If there are still multiple candidate units, the unit with the smallest unit hash is the best parent.

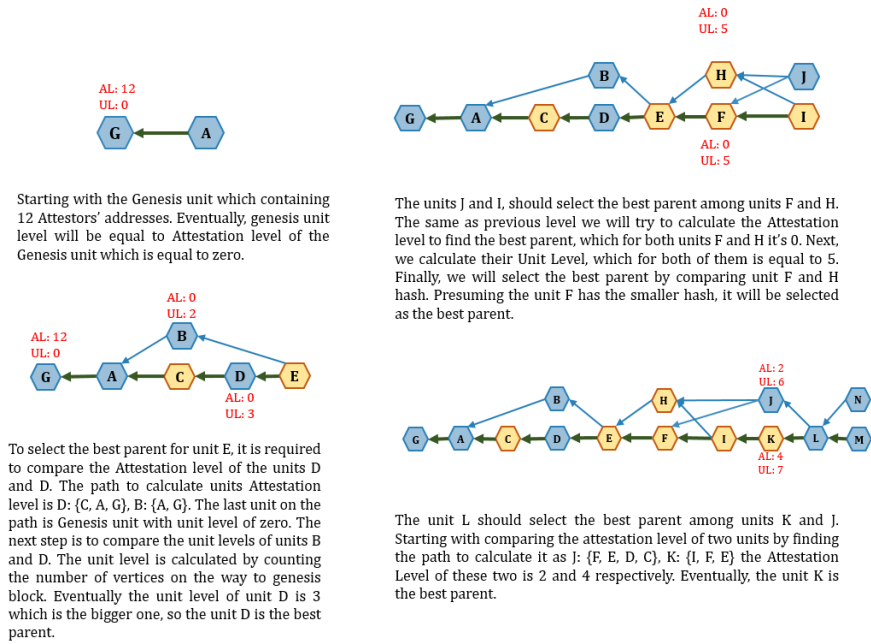


Figure 3-3 the best parent selection explained in four slides.

In this way, you can get a path starting from any Childless Unit, connecting the selected best parent units and finally reaching the Genesis Unit. This path is called the “Main Chain”. As many childless units exist, there might be many “Main Chains”, but the one which starts from the Childless Unit with the highest Attestation Level finally will be selected as **Current Main Chain**.

3.3 Main Chain Index

According to the Best Parent Unit Selection algorithm, all main chains started from childless units will merge at a point. The part from the merging point to the Genesis Unit are completely same. If any new childless units added to DAG ledger cannot change the merging point, the part of main chain from the merging point to the Genesis Unit can be called stable. All units included by the stable main chain unit directly or indirectly will be stable too. The last stable main chain unit is called the Last Stable Unit. All stable main chain units have an index, called Main Chain Index (MCI).

- △ The MCI of the Genesis unit is 0, which increases sequentially along the stable main chain.
- △ The MCI of a normal unit is equal to the minimum MCI of the stable main chain units which include that unit.

According to MCI, we can obtain the total order of all stable units and can use this definitive total order to prevent double spending. This process is called “unit final confirmation”. With the addition of new units, the stable main chain on DAG ledger will continue to grow and all related units will be confirmed.

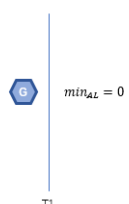
3.4 Main Chain Stabilization Algorithm

The main chain stabilization algorithm uses the Current Main Chain as a reference to determine whether the last stable unit can be transferred to the next unit of the Current Main Chain. According to the child unit number of the last stable unit, this algorithm can be divided into two cases:

Case 1: The last stable unit has only one child unit

Assuming that UL is the unit level of the unique child. By tracing back along the Current Main Chain from the child until more than half of the Attestors of the current consensus round's Attestation units are met, the minimum Attestation level of these Current Main Chain units is denoted as min_{AL} . If $min_{AL} > UL$, the last stable unit can be transferred to the unique child. This procedure (simplified as much as possible) is presented in the slides below:

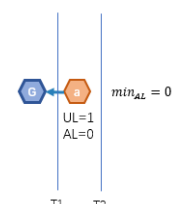
TrustME-PoW @ T1



T1

- Assume **three** Attestors taking part in each consensus round.
- Genesis Unit containing the address of all three Attestors of the first round and it's an stable unit inherently.

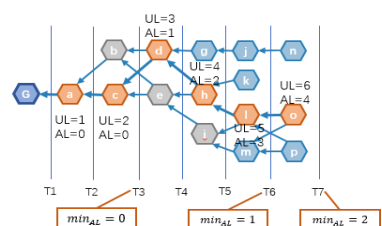
TrustME-PoW @ T2



T1 T2

- The **Orange** unit is an Attestation Unit, generated by one of the Attestors in the first consensus round.
- Genesis Unit has only one child and considers the condition in **Case 1** to judge whether "a" should be stable or not.
- Considering **Case 1** for unit **a** @ **T2**: with $min_{AL} = 0$, $UL = 1$, unit "a" is unstable at this stage.

TrustME-PoW @ T7



T1 T2 T3 T4 T5 T6 T7

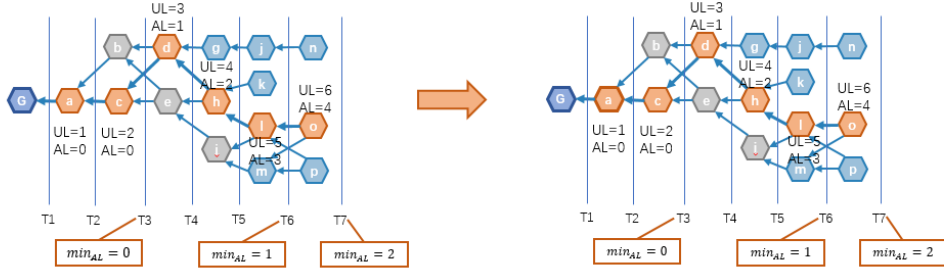
- Considering **Case 1** for unit **a** @ **T3-T5**: $min_{AL} = 0$, $UL = 1$ indicates unit "a" is still unstable.
- Considering **Case 1** for unit **a** @ **T6**: $min_{AL} = 1$, $UL = 1$ indicates unit "a" is still unstable.
- Considering **Case 1** for unit **a** @ **T7**: $min_{AL} = 2$, $UL = 1$, indicating unit "a" becomes stable @ T7 and its MCI is 1.

Case 2: The last stable unit has multiple child units

If these child units that are not on the current main chain, are not the best parent of the other units; they have no chance to compete for be a main chain unit, so we should turn back to use Case 1. Otherwise, we can get a complete alt-branch according to that best parent chain. The maximum unit level of all alt-branch units is called max_{UL} . If $max_{UL} < min_{AL}$, we can push the last stable unit to next unit along the current main chain. Even if less than half of the other

Attestors all support the alt-branch, the Attestation level of any unit on the alt-branch will not exceed max_{UL} . Continue presenting the previous example to reach this condition as below.

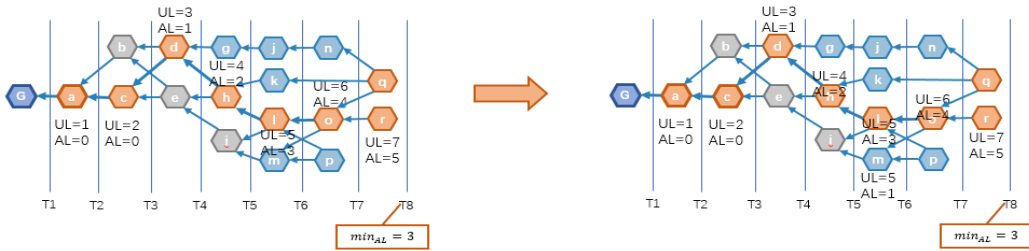
TrustME-PoW @ T7



- Considering **Case 1** for unit **a** @ T3-T5: $min_{AL} = 0$, $UL = 1$ indicates unit "a" is still unstable.
- Considering **Case 1** for unit **a** @ T6: $min_{AL} = 1$, $UL = 1$ indicates unit "a" is still unstable.
- Considering **Case 1** for unit **a** @ T7: $min_{AL} = 2$, $UL = 1$, indicating unit "a" becomes stable @ T7 and its MCI is 1.

- Unit "a" has two child units **b** and **c**. Units **d**, **e**, and **f**, regarding unit "c" as their best parent. unit "b" is not a best child of unit "a" and it has no chance to becomes a Main Chain Unit. Therefore, considering **Case 1** to check whether unit "c" is stable or not.
- Considering **Case 1** for unit **c** @ T7: $min_{AL} = 2$, $UL = 2$, indicating unit "c" is unstable at this stage.

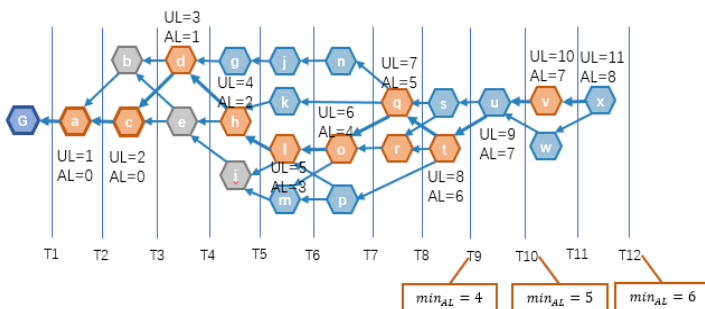
TrustME-PoW @ T8



- Considering **Case 1** for unit **c** @ T8: $min_{AL} = 3$, $UL = 2$, indicating that unit "c" becomes stable and its MCI is 2.

- Unit "c" has 3 child units **d**, **e** and **f**. Units **d** and **e** are both best parent of their children; considering "Case 2" to determine which one should be selected as stable unit. Units **d**, **h**, **l**, **o** and **r** are forming the Current Main Chain, and units **e**, **i** and **m** forming an alt-branch according to the **best child selection** algorithm.
- Considering **Case 2** for alt-branch units **e**, **i** and **m** @ T8, $max_{UL} = 5$ and $min_{AL} = 3$, results in infeasibility in determination of stable unit at this stage.

TrustME-PoW @ T12





- The alt-branch is still constituted from units **e**, **i** and **m**, as none of the units selecting "m" as best parent.
- Considering **Case 2** for alt-branch units **e**, **i** and **m** @ T12: $max_{UL} = 5$ and $min_{AL} = 6$ indicating that Current Main Chain completely defeated alt-branch, and consequently unit "d" becomes stable at T12.

4. TrustME-PoW Consensus

TrustNote adopts a two-tier consensus mechanism comprising “base consensus” and “TrustME consensus”. The base consensus, also known as “DAG consensus”, requires new units sent by nodes to verify previous units and reference them. The TrustME consensus is used to select a main chain on DAG ledger and give out the total order of all stable units based on the main chain. Such two-tier consensus mechanisms can improve transaction throughput and reduce transaction confirmation delay, thus effectively solving the problem of Excessive Bifurcation and double-spending.

4.1 Motivation

TrustME-PoW designed in two different types during the conceptual design phase:

-  Fine-grained
-  Coarse-grained

Both methods studied carefully, but in fine-grained PoW schemes, there is not any bound between Super Nodes necessarily. In case of network-partitioning, as long as there are Super nodes in each partition, MC growth and stability will still be promoted, which will lead to successful confirmation of a double spent unit. To resolve this issue some attributes suggested for strengthening the bounds between the Super nodes. Eventually, it observed that nodes' bounding that achieves a fine-grained consensus is equivalent to transforming the fine-grained consensus into a coarse-grained consensus. Therefore, in final design Coarse-grained PoW selected.

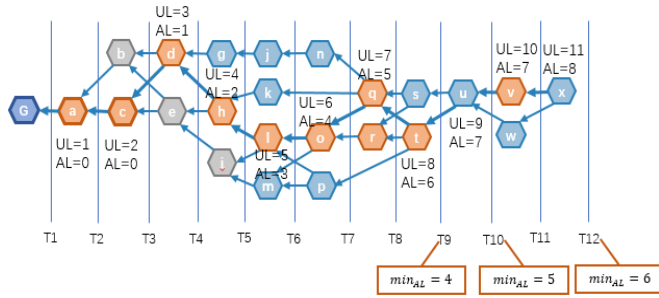
In addition, when the network is divided, networks with more than $\left(\frac{n}{2}\right)$ Attestors will continue to expand, and the main chain will continue to grow and stabilize over time. If in the partitioned network, the Attestors number is less than or equal to $\left(\frac{n}{2}\right)$, all the units in the network will not become stable and a new round of TrustME-PoW consensus cannot be completed until the network get more than $\left(\frac{n}{2}\right)$ Attestors.

4.2 How to Select Attestors

TrustME-PoW conducted in separate rounds. Each round lasts **until the next round Attestors list** will be completed. **Eighteen** Attestors will be selected in **each** round and **two** other Attestors maintained by **TrustNote** will be added to the consensus round, which results in **twenty** Attestors in each consensus round.

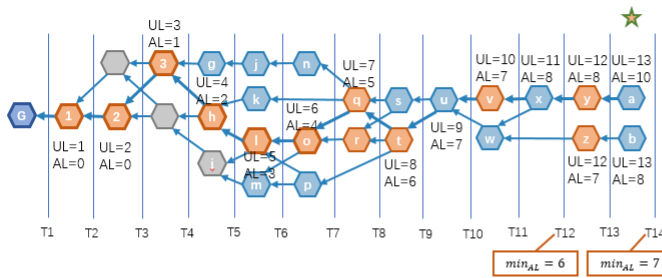
Continuing the example in [section 3.4](#) of this report, we will demonstrate more detail about the working mechanism of consensus round. Assuming three units in gray presented in figures below are PoW units each for one of the Attestors to take part in the next consensus round.

TrustME-PoW @ T12



- Unit “d” has two child units g and h. Units g, j and n are forming an alt-branch, which results in considering “Case 2”.
- Considering Case 2 for alt-branch units g, j and n @ T12: $max_{UL} = 6$ and $min_{AL} = 6$, results in infeasibility in determination of stable unit at this stage.

TrustME-PoW @ T14



- The alt-branch is still constituted from units g, j and n, because no unit selects unit “n” as best parent.
- Considering Case 2 for alt-branch units g, j and n @ T14: $max_{UL} = 6$ and $min_{AL} = 7$, indicates Current Main Chain completely defeated alt-branch, and eventually unit “h” becomes stable at this stage.
- Unit “h” has two child units k and l, but only unit “l” is the best child, which results in considering “Case 1” to check whether unit “l” is stable or not.
- Considering Case 1 for unit l @ T14: $min_{AL} = 7$, $UL = 5$, indicating unit “l” becomes stable at this stage.
- Then, Considering Case 1 for unit o @ T14: $min_{AL} = 7$, $UL = 6$, indicating unit “o” becomes stable at this stage.
- Then, Considering Case 1 for unit q @ T14: $min_{AL} = 7$, $UL = 7$, indicating unit “q” is an unstable unit.
- @ T14, three Attestors for next consensus round have been selected.

Once unit “l” becomes stable, at T14 last PoW unit, unit “i” for one of the Attestors interested in taking part in next consensus round becomes stable and next consensus round starts immediately. The partial MCI order of each PoW unit becomes stable determines the priority of the Attestors in next consensus round.

4.3 PoW Unit

For i^{th} consensus round, the super node thread calculates $pubSeed_{i-1}, Coinbase_{i-2}$ and $pubSeed_i = \text{blake2}_{256}(Seed_{i-1} || Coinbase_{i-2} || \text{hash}(FirstStableUnit_{i-1}))$. Finally, it uses the public seed to calculate the Equihash calculation input for one super node n as $Seed_{i,n} = \text{blake2}_{256}(pubKey_n || pubSeed_i)$, and then calculates Equihash difficulty factor “d”. The difficulty factor “d” is designed and adjusted for each consensus round based on desired time of about **five** minutes to find the solution of Equihash. It means this parameter is a threshold and the result of solving Equihash must have a certain number of leading zeroes to be accepted. This parameter will be adjusted with respect to desired time of generating a PoW unit and it will be applied to the system smoothly.

Super nodes are running the Equihash Algorithm with the calculated parameters to find Equihash solutions as quick as possible. Once they find a right solution, they will generate a PoW unit containing the solution and publishes it to the DAG ledger. The first eighteen addresses owning a stable PoW unit will obtain the Attestation power and to be an Attestor in the i^{th} consensus round. If there are multiple Super nodes submitting many PoW units at the same time, the priorities are determined according to the MCI of the PoW units. If the MCI is the same, the smaller the unit hash is, the higher the priority it has. In each round consensus, 18 different Super nodes corresponding to the first 18 stable PoW units will be given the Attestation Power.

PoW Unit

```
{
  unit: {
    version: 'Protocol Version',
    alt: 'Alternative Currency',
    messages: [{
      app: 'payment',
      payload_location: 'inline',
      payload_hash: 'Hash of Payload of this Transaction Message',
      payload: {
        outputs: [{
          { address: 'Destination Wallet Address', amount: 9989459 },
          { address: ' Destination Wallet Address', amount: 10000 }],
        inputs: [{
          unit: 'Wallet Address of Origin of the Message',
          message_index: 0,
          output_index: 0
        }]
      }
    ]},
  messages:[{
    app: ' PoW-Equihash ',
    payload_location: ' inline ',
    payload_hash: ' hash of payload ',
    payload: {
      round: ' round number ',
      seed: ' string of seed ',
      difficulty: '',
      solution: '',
      attestor_address: ' Wallet Address of Node '
    }
  ]}
  authors: [{
```

```

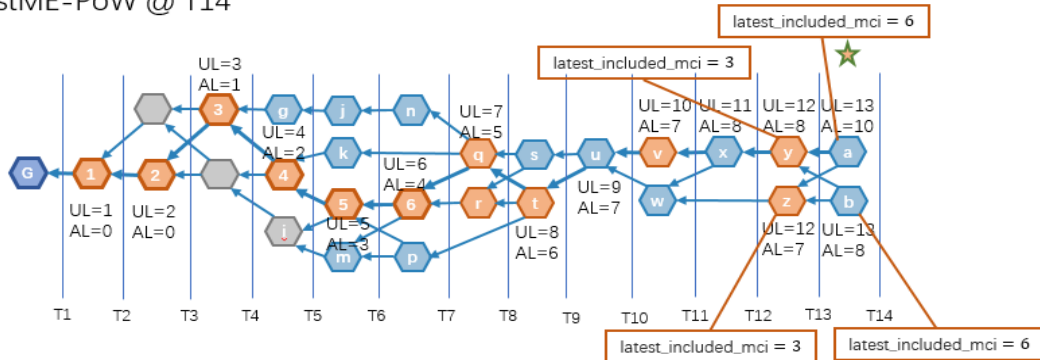
    address: 'Attestor Address',
    authenticifiers: { r: 'Attestor Signature' }
  },
  parent_units: ['FZvSHUDsqG+TDJizrqzkNBrQUv7PDE4EwWdm68FNMuU='],
  last_ball: 'CFmFF4hnl4eTMv4AR/45gpVZW8ty3uXEohur5oHdls=',
  last_ball_unit: 'WzqSo49GLwjNE52tj2kw32/gxnuvmCypkikdVyLnHrQ=',
  headers_commission: 60% of Total PoW unit Size,
  payload_commission: 40% of Total PoW unit Size,
  unit: 'Hash of this PoW unit'
}
}

```

4.4 Consensus Round Switching

The fact that consensus round switching happens is deterministic, but there are still some questions might be tickling your mind such as when the right time is to switch to next consensus rounds, or How to determine the validity of an Attestation unit and so on. The slides below (continued from [section 3.4](#) and [section 4.2](#)) presenting additional details to answer all these questions.

TrustME-PoW @ T14



- The biggest MCI in 1st consensus round attester list is 0.
- The biggest PoW unit MCI in attester list of 2nd consensus round is 5.
- The attester list of 3rd consensus round is empty @ T14.

TrustME-PoW Consensus Round Switching

- When is the right time to switch to next consensus rounds?
Once all attestors of next consensus round have been selected, switch to the next round immediately.
- All full ledger nodes maintain Attestor List of every consensus round.

	priority	attestor	PoW unit MCI	PoW unit		priority	attestor	PoW unit MCI	PoW unit
i^{th} round	1	xxxxxxxx	a	xxxxxxxx	$(i+1)^{\text{th}}$ round	1	xxxxxxxx	c	xxxxxxxx
	2	xxxxxxxx	b	xxxxxxxx		2	xxxxxxxx	d	xxxxxxxx

	n	xxxxxxxx	x	xxxxxxxx		n	xxxxxxxx	y	xxxxxxxx

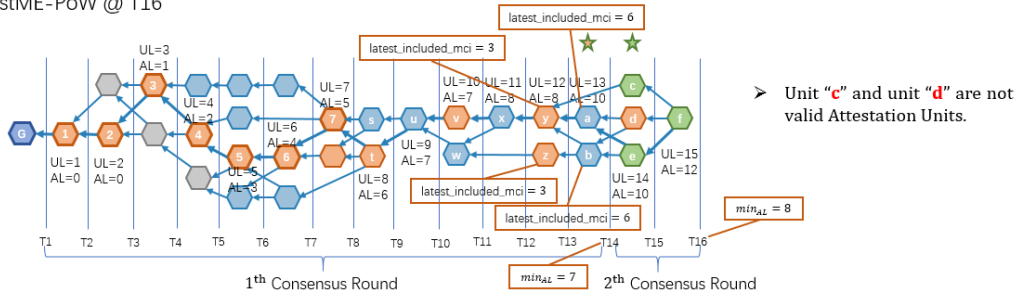
- How to determine the validity of an Attestation unit?
➤ **latest_included_mci** of a unit is equal to the MCI value of the latest Main Chain unit it included.
➤ Validity Checking Algorithm:

```

for each attestation unit in  $i^{\text{th}}$  round:
    if latest_included_mci of all included units < x, then
        return False;
    if (the  $(i+1)^{\text{th}}$  attestor list is full) && (the latest_included_mci of any included unit >= y), then
        return False;
    return True;

```

TrustME-PoW @ T16



```

for each attestation unit in 1st round:
    if latest_included_mci of all included units < 0, then
        return False;
    if (the 2th attestor list is full) && (the latest_included_mci of any included unit >= 5), then
        return False;
    return True;

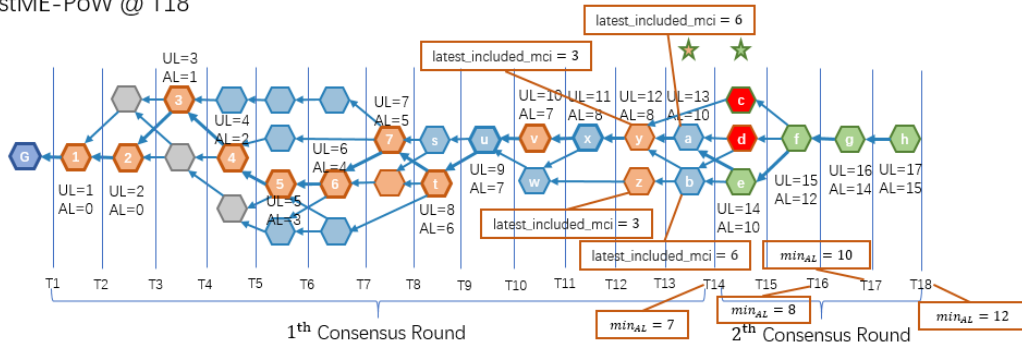
```

```

for each attestation unit in 2th round:
    if latest_included_mci of all included units < 5, then
        return False;
    if (the 3th attestor list is full) && (the latest_included_mci of any included unit >= ?), then
        return False;
    return True;

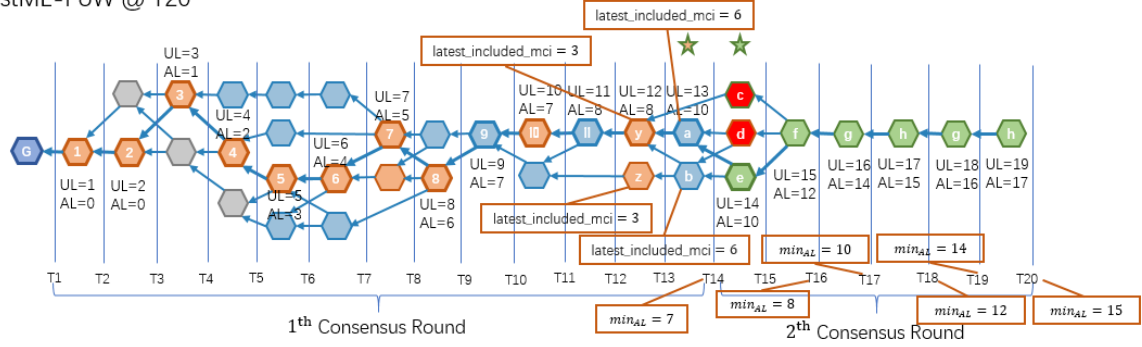
```

TrustME-PoW @ T18



- Considering **Case 1** for unit **t** @ **T18**: $min_{AL} = 12$, $UL = 8$, indicating that "t" becomes stable.
- Then, considering **Case 1** for unit **u** @ **T18**: $min_{AL} = 12$, $UL = 9$, indicating that unit "u" becomes stable.
- Then, considering **Case 2** for alt-branch resulted from units **w** and **z** @ **T18**: $max_{UL} = 11$ and $min_{AL} = 12$, indicating Current Main Chain completely defeated alt-branch, and eventually unit "v" becomes stable.
- Then, considering **Case 1** for unit **x** @ **T18**: $min_{AL} = 12$, $UL = 11$, indicating that unit "x" becomes stable.
- Then, Considering **Case 1** for unit **y** @ **T18**: $min_{AL} = 12$, $UL = 12$, indicating unit "y" is an unstable unit.

TrustME-PoW @ T20



- Considering **Case 1** for unit **y** @ **T20**: $min_{AL} = 15$, $UL = 12$, indicating that “y” becomes stable.
- Then, considering **Case 1** for unit **a** @ **T20**: $min_{AL} = 15$, $UL = 13$, indicating that unit “a” becomes stable.
- Then, considering **Case 1** for unit **e** @ **T20**: $min_{AL} = 15$, $UL = 14$, indicating that unit “e” becomes stable.
- Then, Considering **Case 1** for unit **f** @ **T20**: $min_{AL} = 15$, $UL = 15$, indicating unit “f” is an unstable unit.
- Unit **e** is the first Main Chain Attestation Unit of 2nd Consensus Round. When unit **e** becomes stable, **PoW of 3rd Consensus Round can start.**

4.5 Equihash Difficulty Calculation

Difficulty Calculation

Super Node thread

- INPUTS:
 - Target round time: $T = 5 \text{ min}$
 - New consensus round number: i
 - OUTPUT: Difficulty in the next round D_i
- (a) Calculate the number of previous rounds to be considered in the calculation of difficulty: $PB = \text{floor}(f(T))$
- (b) Calculate the Sum of Average PoW round time:
- $SumAvgPoW = \sum_{i=N+1}^i \min[(T \times C_1), (AvgPoW[j] - AvgPoW[j - 1])]$
- (c) Calculate the Gross Average PoW Time: $GAPW = f(N, T, SumAvgPoW)$
- (d) Calculating the difficulty for next round: $D_i = f(T, GAPW, SumAvgPoW)$

4.6 TrustME unit

An Attestor can send multiple TrustME units within its consensus round. The TrustME units will be selected as the MC unit with a large probability. Each Attestor in the consensus round sending TrustME units according to the priority order. To reduce the delay in confirmation of the transactions, if any of the Attestors don't send the TrustME message during a certain time the next two Attestors can send the TrustME unit.

TrustME unit Sending Mechanism

Attestation threads

At any Consensus round:

- (a) Top twenty Attestors, aware of their priority joining the consensus round.
- (b) First Attestor should send its first TrustME unit within 5 seconds, otherwise it will be replaced with the next **eighteen** Attestors on the list.

- (c) First Attestor on the list send the first TrustME unit.
- (d) The first TrustME unit contains the Coinbase list of rewards for previous consensus rounds which will be sent by the first Attestor.
- (e) All the Attestors should send their units with respect to priority list.
- (f) The consensus round finishes and (a) will be performed again.

⚠ If TrustNote Attestors or any of other eighteen Attestors in the consensus round wouldn't be available for any reason the next priority Attestor in current consensus round will immediately start sending TrustME units.

TrustME Unit

```
{
  unit: {
    version: 'Protocol Version',
    alt: 'Alternative Currency',
    messages: [{
      app: 'payment',
      payload_location: 'inline',
      payload_hash: 'Hash of Payload of this Transaction Message',
      payload: {
        outputs: [{
          { address: 'Destination Wallet Address', amount: 9989459 },
          { address: ' Destination Wallet Address', amount: 10000 }],
        inputs: [{
          unit: 'Wallet Address of Origin of the Message',
          message_index: 0,
          output_index: 0
        }]
      }
    }],
    messages:[{
      app: 'TrustME',
      payload_location: 'Inline',
      payload_hash: 'Hash of Payload',
      payload: {
        round: 'Round Number',
        PoW_solution: 'The PoW Unit Hash',
        priority: 'Priority of Attestor',
        coinbase of (i - 2)th round: [
          {address: '...', amount: 21 MN},
          {address: '...', amount: 19 MN},
          ...
        ]
      }
    ]
  }
}
```

```

    }
  }}
  authors: [{
    address: 'Attestor Address',
    authenticifiers: {r: 'Attestor Signature' }
  }],
  parent_units:['FZvSHUDsqG+TDJizrqzkNBrQUv7PDE4EwWdm68FNMuU='],
  last_ball: 'CFmFF4hnwI4eTMv4AR/45gpVZW8ty3uXEohur5oHdls=',
  last_ball_unit:'WzqSo49GLwjNE52tj2kw32/gxnuvmCypkidkVyLnHrQ=',
  headers_commission: 200,
  payload_commission: 200,
  unit: 'Hash of this PoW unit'
}
}

```

4.7 Attestation Reward

If the TrustME unit sent by the Attestor of a consensus round is a stable unit on MC, the Attestor will receive Attestation reward and the larger the proportion of the TrustME units it sent, the more Attestation reward it would receive. In the absence of stability in the main chain, it is impossible to determine which TrustME units are on the main chain and it is not fair to calculate the Attestation bonus. Therefore, this consensus mechanism does not provide Coinbase directly (like Bitcoin) in the unit. Only after all the TrustME units belonging to the same round on the main chain are stable, TrustNote Attestors calculate how many Attestation bonuses can be obtained for each Attestor in that specific round; the reason for this is to eliminate the cost of generation of this unit for Attestors. After all the TrustME units of a consensus round are stable, the number of TrustME units at each address in the statistical chain shall be calculated according to the proportion of the Attestation bonuses. Even more, if any Attestors has doubt in calculated share of Attestation reward, they can generate the Attestation share list of that specific round by themselves and reference the TrustNote unit to make sure.

Coinbase calculation

full ledger thread

- INPUTS:
 - List of Attestors: $\{A_1, \dots, A_{20}\}$
 - Round number: i
 - Reward of consensus round i : R_T^i
- OUTPUT: list of the share of each Attestor from the Attestation rewards round i $\{P_{A_1}^i, \dots, P_{A_{20}}^i\}$
 - (a) Calculate total number of Stable units on MC in round i : P_T^i
 - (b) Calculate the number of stable TrustME units on MC generated by each Attestor $\{A_1, \dots, A_{20}\}$: $\{MCU_{A_1}^i, \dots, MCU_{A_{20}}^i\}$

(c) Calculate the share of each Attestor: $P_{A_j}^i = \frac{MCU_{A_j}^i}{P_T^i} \times R_T^i$ ($j = 1, \dots, 20$)

5. Switching from Witnesses to TrustME-PoW

5.1 Overview

At a certain time, the i^{th} round's Attestors are on duty.

Round Number	Attestor	TrustME unit
$i - 1$	invalid	being stable
i	sending TrustME units until all units in round $(i - 1)$ has been stable and there are enough Attestors for round $(i + 1)$	not stable
$i + 1$	waiting until the first TrustME unit of round (i) to become stable, and making consensus with the seed of round (i) , the Coinbase of round $(i - 1)$ and First Stable MC unit (i)	N/A

⚠ At the junction, there might be cases where the i^{th} TrustME unit is sent after starting the $(i + 1)^{\text{th}}$ round of TrustME. At this time, the TrustME unit of the i^{th} round will be deemed as illegal.

5.2 Procedure

This section gives out the list of steps about how to convert to the TrustME-PoW consensus mechanism.

Step	Description
1	Wait for a MC unit with a specific MCI value (e.g. 1,000,000) to become stable.
2	The R_1 consensus round ¹ is initiated. The R_1 public seed is an initial random number and then the respective seeds of the R_1 round super node are calculated. The initial difficulty factor is 1. The super node using difficulty and node specific seed to calculate the Equihash solution. The super node proceeding to take part in the R_1 consensus round should issue one PoW unit.
3	Wait for Witnesses to make first eighteen PoW units to become stable and select all

¹ First consensus round after switching to TrustME-PoW.

Step	Description
	twenty Attestors for R_1 round.
4	Once the list of Attestors are determined, the witness consensus mechanism is deactivated, and the R_1 Attestors start sending the TrustME units.
5	Wait for the first TrustME unit of R_1 to become stable on MC.
6	The super nodes proceeding to attend the R_2 consensus round, calculates the R_2 public seed based on the public seed of the R_1 and the genesis unit. The super node using difficulty and node specific seed to calculate the Equihash solution.
7	The super node proceeding to take part in the R_2 consensus round should issue one PoW unit.
8	Wait for Attestor of the R_1 to make first eighteen PoW units to become stable and determine the top eighteen Attestors for R_2 .
9	The R_1 Attestors end the mission, and the R_2 Attestors start sending the TrustME units.
10	Wait for the first R_2 TrustME units to become stable on MC.
11	The super nodes proceeding to attend the R_3 consensus round, calculates the R_3 public seed based on the public seed of the R_2 , the $Coinbase_{i-2}$ equal to Coinbase of round R_1 and the $FirstStableUnit_{i-1}$. The super node using difficulty and node specific seed to calculate the Equihash solution. The super node proceeding to take part in the R_3 consensus round should issue one PoW unit.
12	Wait for Attestor of the R_2 to make first eighteen PoW units to become stable and determine the top eighteen Attestors for R_3 .
13	The R_2 Attestors end the mission, and the R_3 Attestors start sending the TrustME units. The first TrustME unit contains the Coinbase result for the Attestors of R_1 .
14	Wait for the first R_3 TrustME units to become stable on MC.
15	The super nodes proceeding to attend the R_4 consensus round.

Eventually, all the consensus rounds after R_3 following the same procedure as R_3 .