guardtime

APRIL 2020

# Guardtime Health
# Shortage Alerts Engine

# Technical Briefing

Classification: Public

GUARDTIME.COM

# Executive Summary

The Shortage Alerts Engine by Guardtime Health has been designed to track aggregate availability of critical supplies, such as medicines, across a network with multiple providers. The providers, as competitors on the same market, are assumed to be unwilling to share their inventory levels with each other or with any one third party.

To preserve confidentiality, each supplier breaks their input into multiple shares distributed among independently operated nodes. The nodes then use secure multi-party computation (MPC) techniques to compute the sum of all inputs without knowing any of the inputs themselves.

Internally the system operates on shares of the inputs and shares of the sum. Each input is split into secret shares in such a way that the complete set unambiguously represents the input, but any smaller subset reveals no information about it. The shares of the inputs are then combined into shares of the sum of inputs and finally the sum can be reconstructed from these derived shares.

External reporting depends on the country and the views of the supply chain stakeholders. In some markets, only alerts are emitted when the aggregate values fall below some predefined thresholds or remain below such thresholds for some predefined durations. In other markets, the aggregate values themselves may be shared more broadly (even if alerts have not been triggered) to allow services that aim to make the supply chain more efficient.

The system supports distributed auditing: the correctness of behavior of each participant can be verified separately and independently, implying both the correctness of the results of the whole computation and the privacy of input data.

The rest of this document gives an overview of the architecture and the technologies used to achieve these properties.

# General Architecture

The overall architecture of the system is shown in Figure 1. Each data source (such as a manufacturer, an importer, or a reseller) operates an MPC gateway that receives the stock level from the respective inventory records and splits it into multiple shares in such a way that the complete set unambiguously represents the input, but any smaller subset reveals no information about it.

Each share is sent to a separate MPC service node operated by an independent party. These parties may be the data sources themselves, but this is not a requirement. Crucially, each node receives only one share from each data source, and therefore can't recover any of the inputs.

Each MPC service node then computes a share of the sum from its shares of the inputs and forwards the share of the sum to the coordinator node. The coordinator node, after receiving all shares of the sum, can combine them to recover the sum itself, but will learn nothing about the individual inputs (apart from the obvious fact that none of them exceeds the sum).

Each MPC gateway is hosted and operated by the data source itself. As each gateway sends out individual shares to different service nodes over encrypted links, no external party will have the information needed to recover the confidential input.

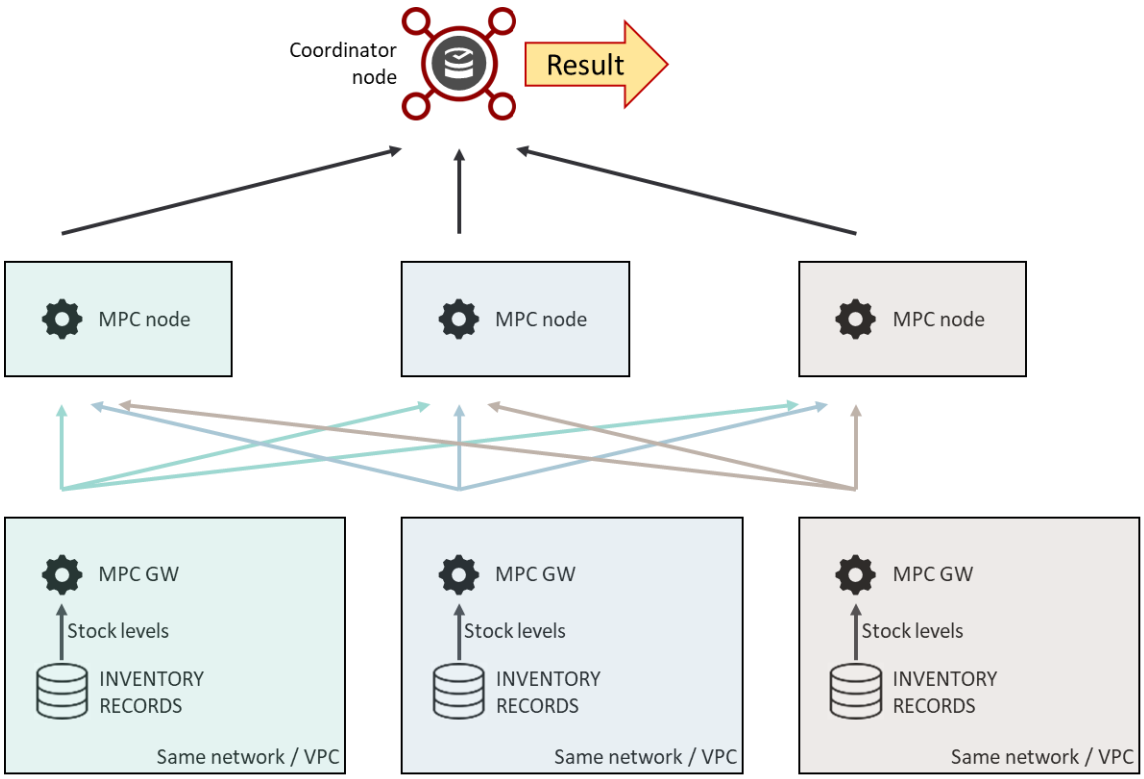


*Figure 1: General Architecture.*

# Additive Secret Sharing

The computation is based on additive secret sharing. When configured with K service nodes supporting N data sources to compute the sum $S=A_1+A_2+\ldots+A_N$, it operates as follows:

- the MPC gateway of each data source i represents their input $A_i$ as a sum of K shares $A_{i,j}$ chosen so that $A_i=A_{i,1}+A_{i,2}+\ldots+A_{i,K}$ and distributes the shares among the K nodes;
- each MPC service node j in turn receives shares from all N gateways and computes its share of the sum as $S_j=A_{1,j}+A_{2,j}+\ldots+A_{N,j}$;
- finally, the coordinator node receives the shares of the sum from the service nodes and recovers the sum itself as $S=S_1+S_2+\ldots+S_K$.

In the table in Figure 2, each row (except the last) represents one data source's input and the shares it sends out, and each column (except the first) represents the shares one service node receives and aggregates into its share of the sum. The value of S can be viewed either as the sum of the first column or the sum of the last row, and the two views will always be the same.

| | | | | | |
|---|---|---|---|---|---|
| **$A_1$** | $A_{1,1}$ | … | $A_{1,j}$ | … | $A_{1,K}$ |
| **…** | … | … | … | … | … |
| **$A_i$** | $A_{i,1}$ | … | $A_{i,j}$ | … | $A_{i,K}$ |
| **…** | … | … | … | … | … |
| **$A_N$** | $A_{N,1}$ | … | $A_{N,j}$ | … | $A_{N,K}$ |
| **S** | **$S_1$** | **…** | **$S_j$** | **…** | **$S_K$** |

*Figure 2: Multi-Party Computation with Secret Sharing.*

To improve privacy, the computations are done in so-called modular arithmetic that "wraps around" a modulus value M. Computing with time is a common example where M=12: if the time now is 10 o'clock, then in 4 hours it will be 2 o'clock, as the count resets at 12 and starts from 0 again. Note how, due to the wrap-around, the sum in this case is smaller than either of the "addends". To get correct results, M must be larger than any valid sum. The value $2^{64}$ is used in the Shortage Alerts Engine, which can therefore handle results up to 19 digits long.

In a modular arithmetic system, the first K-1 shares for each input $A_i$ can be chosen completely randomly from the range 0…M-1, and there will be a value from the same range for the final share that ensures the equation $A_i=A_{i,1}+A_{i,2}+\ldots+A_{i,K}$ will hold, most likely after some number of wrap-arounds. Interestingly, also the final value, though computed deterministically after the previous ones have been chosen, appears completely random to an external observer. Thus, the scheme ensures perfect privacy.

# Distributed Auditing

The system also supports a distributed auditing mechanism that allows the behavior of each party to be verified separately and the global correctness of the process to be deduced from these local audits. This removes the need to have one auditor that everyone must trust to look at their confidential data (if such a trusted party existed, all data sources could just send their inputs to that party and have them compute and announce the result).

The audits are supported by a bulletin board seen by all the parties (Figure 3). Before a party sends out a share, it posts a commitment of it on the board. The recipient only accepts the share if it matches the commitment. The commitments are computed using one-way hash functions, so the original data cannot be recovered from them. Yet, the commitments are binding: it is infeasible to come up with two different data sets to match the same commitment.

To remove the need to trust the bulletin board to operate correctly, two mechanisms are used. First, all updates to the board are cryptographically time-stamped, so that anything posted cannot be changed after the fact. Second, each MPC gateway and each MPC service node has a signing key and uses it to digitally sign the commitments before posting them to the board, so that no party can post commitments in the name of other parties.

These features allow the board to be used as a common reference point that links the local audits of all parties into an integrated whole and allows concluding the correctness of the global process from the consistency of all local audits, and also provides hard evidence in case some party has misbehaved.

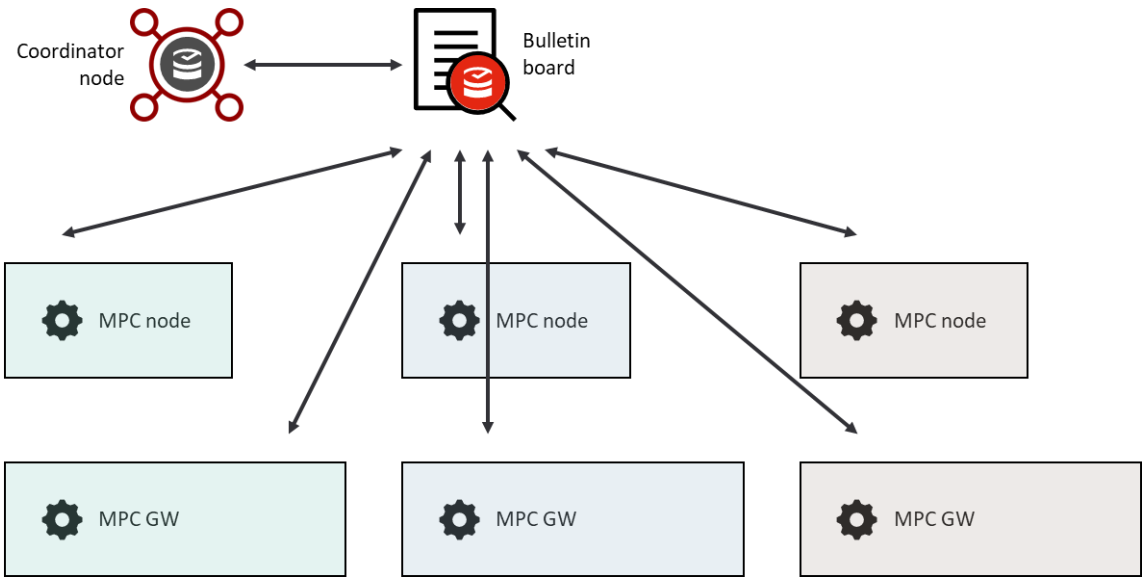


*Figure 3: Bulletin Board for Distributed Auditing.*