With Google as our cloud hosting provider, we are confident that we have more than enough room to grow our infrastructure to meet the increasing demands of the Factom protocol: we can provide up to 96 vCPUs, 1434 GB of RAM and 16 persistent disks for a total of 64 TB of disk space per instance (GCP currently provides up to 128 persistent disks in beta). If we are selected, with our proposed server configuration, we are also going to support 6000 sustained random IOPS per instance upon launch, with the ability to increase this up to 40000 IOPS for reads and 30000 IOPS for writes.

At Factomatic, we boast a geographically distributed team, with experienced technical people in Netherlands, Bulgaria and the US. Server administration is one of the core competencies of the company and we have redundancy of server management expertise both in Europe (Ivan & Valentin) and in the US (Mihail & Tito). This allows us to react to unforeseen circumstances 24/7, while not putting an unrealistic and unsustainable burden on any single team member and without placing trust in any 3rd parties for managing the infrastructure of nodes we might be awarded.

Within the company, we have also thought at length about how to handle unscheduled restarts. While we realize initially there will be kinks to iron out and restarts might require manual supervision to ensure they were successful, we are firm believers in the necessity to automate this process as much as possible.

One possibility is to use a second blockchain as the signalling mechanism for a restart, since the Factom blockchain might be unusable in case multiple Federated nodes are down. This brings all the advantages of distributed ledger technology: decentralization, redundancy and security. As an example, we envision a 2-of-N Ethereum smart contract that could be used as the notification agent. Using a standard feature of Solidity and the Ethereum blockchain, smart contracts can emit one or more events as part of a transaction, which are available to listeners without them needing to spend any gas. Any such event can have associated metadata, which could be used to provide start and end timestamps for the reboot, as well as any other relevant information for the coordination e.g.:

```
FactomMainnetReboot(uint startTs, uint endTs)
```

Due to the low block time of Ethereum (around 15 seconds), such notifications can be issued in a timely manner, and in case of network congestion, using a very high gas price would not be a problem, since the restart events will (hopefully) be far apart, thus not incurring any significant costs. Authority Server operators would need to have a (reasonably) synchronized time (e.g. using ntp)

and can run a special service which is listening for `FactomMainnetReboot` events and acts accordingly if one is emitted by the smart contract. To avoid the need of hosting an Ethereum node, listening for events could be accomplished by connecting to a 3rd party node, such as the ones provided by **Infura**, or alternatively by community hosted Ethereum mainnet nodes, sponsored by the Grant Pool.

To allow for manual intervention and supervision, a service such as **PagerDuty** can be used for sending SMS/call/email or push notifications to Authority Node operators. To ensure maximum reliability the restart event can issue two time intervals: one for manual restarts and one for automated ones, and if no manual restart is done by an operator due to unavailability an automatic one could be attempted.

The above is only one example and we believe there are many other options that should be explored and experimented with on the testnet: e.g. a more traditional notification system could use multiple services sending push notifications, instead of an Ethereum smart contract. Security with such a setup will be enforced by signing of restart messages and DoS protection will be provided by the cloud hosting infrastructure for the service or a third party. To ensure maximum protection, several instances of the service could be deployed to different cloud providers as well.