Time is a very important tool to coordinate people’s activity, or to build distributed systems. In fact, clock-driven algorithm can significantly simplify fault-tolerant distributed systems (e.g. L. Lamport, Using Time Instead of Timeout). Indeed, several blockchains explicitly rely on Time to coordinate participants (e.g. V.Buterin, Network Adjusted Timestamps).

The downside is that clocks can fail too. For example, on-board clocks of most computers are not very stable, losing or gaining up to 100 ppm (i.e. up to 8 seconds a day - link), most important factors being temperature changes and aging. Thus, distributed processes’ clocks will inevitably diverge and a clock synchronization protocol is required.

BFT system designers relying on an assumption that clocks are synchronized should validate that the resulting system is really resilient. Since additional faults can be induced via such clock synchronization protocols, e.g. it can become an attacker’s target.

We consider a particular case, when distributed clocks are synchronized to a common time standard, e.g. UTC, which is highly desired in many practical systems. This can introduce a dependence on a Time Service, which is necessarily centralized or relatively centralized. Thus, it can be an interesting opportunity for an attackers, since it can induces multiple correlated or coordinated faults.

The goal of the post is to propose a model, which can be use to analyze security properties of protocols relying on a public Time Service.

From a more generic perspective, the model can be extended to protocols accessing a generic (Public) Service, which can also become an implicit centralized dependency.

Time Providers

There can be several ways to synchronize clocks with UTC:

- an atomic clock, which is synchronized to UTC initially only
- GNSS receiver
- Radio Wave clock
- Power Grid clock
- NTP serivce

Relatively cheap atomic clocks are available, though their cost is still prohibitive for most users. However, the option is ideal, in some sense, since one can reasonably assume an adversary cannot control such a clock, while it’s stable enough to keep time without a synchronization protocol.

GNSS, Radio Wave and Power Grid Clocks are also attractive, since it’s possible that there are multiple correlated faults, but one can assume they are localized in some geographical area, since it will be extremely costly to suppress radio-waves world-wide. These options are not expensive, however, they can still be prohibitive, since many blockchain nodes are hosted (for example, 2/3 of Ethereum nodes at the moment of writing are hosted - link).

So, the options are attractive for relatively wealthy node operators, e.g. in permissioned or consortium networks. But in open permissionless systems, the main option is to use NTP service, i.e. using time servers, which itself synchronized to stratum 0 Time Server, either directly or through an intermediate Time Server. Thus, a malicious adversary has a much wider range of possible attacks.

Other Public Services

Time is not the only Public Good, so the model can be adapted to other settings too. For example, nodes can use a set of root nodes to bootstrap their operation. Or they can use a common DHT service.

Another example is that nodes can be hosted via the same hosting provider, so it exposes them to the similar correlated faults.

The whole Internet infrastructure can be abstracted as a Service Provider too.

The general idea is to model correlated faults that have some specific structure. Time Service is used as a motivating example.

Anonymous Access

One particular interesting case is that we can realistically assume a GNSS Time Provider cannot constitute a two-faced clock, e.g. providing different readings to two users at the same moment of time (within reasonable limits, ignoring precision time measurements, for example).

To be able to do so, one has to distinguish different kind of users, since Public Services have to provide a qualitative service to public. While GNSS infrastructure in general assumes different categories (and different spatial locations) of users, in our case, a casual GNSS receiver is indistinguishable from many others.

There can be the case that a particular GNSS receiver is faulty and thus other GNSS receivers will result in different readings. However, that can be treated as regular node fault, since such faults should be rare and uncorrelated.

System Model

The proposed model is a variation of a traditional model, popular in distributed process literature: there is a set of distributed processes (or nodes), which are connected by channels (links). Various failure models of nodes and channels can be assumed.

We can model a Time Provider as a special kind of distributed process (node), which does not participate directly in the main protocol (e.g. Byzantine consensus). However, it is accessed as a server - to distinguish it from a client, which is another special kind of a distributed process, used in some models.

In the Time context, we can also call the Time Providers as reference clocks. In a general case, we can call them (Public) Service Providers.

The regular distributed processes (nodes) access Service Providers via special kind of links.

Limiting Adversary Power

The special kinds of processes and links form a hybrid model, where adversarial power limits can be specified independently for different kinds of parties (processes or links).

For example, in a simple GPS model, the GPS service can be described as a single instance of Service Provider, which cannot fail, however, links connecting regular nodes to it can fail by omission.

A more complex model would be to have several GPS Service instances, each connected to multiple nodes, so that GPS services can fail by crashing, however, the failures are independent or a small upper bound can be specified (an adversary can choose no more than t

GPS services to fail). This allows to introduce correlated faults (attached multiple processes will suffer from the fault).

Another model can be to mix several Time Providers, e.g. GPS and Radio Clocks (or GPS+GLONASS+Galileo+BeiDou), so that Time Providers can fail, but do that independently.

As noted above, we can restrict Time Service or link faults to be non-Byzantine only, e.g. crash or omission.

NTP Time Servers can be modeled as a special kind of nodes too. In the case, we can assume a more powerful adversary, for example, able to make any $t < n$

NTP Time Servers behave as two-faced clocks. As vast majority of node operators are likely to be using Linux distros (e.g. Ethereum node distribution by OS), which is configured to use NTP Pool by default, there is a risk of NTP pool Sybil-like attack. We can model the possibility of the attack by specifying high percentage of NTP time servers that can be controlled by an adversary.

The more powerful adversary is justified, as there exist NTP attacks (e.g.one, two) or NTP pool attacks (e.g.link)

As there are many Public NTP servers which can be randomly chosen by validators to set up their NTP daemons, a probabilistic model may be required in general case to model adversarial power limits.

(Bounded) Rational Validator

It's known that popular “honest majority”/“dishonest minority” models are not realistic (e.g.link), since they ignore economical (dis)incentives, which can be significant, in certain cases. Node administrators can be “lazy” too.

An alternative is to model validators as rational actors.

In the context of Time Providers, setting up a reliable time source can require both money and skills. As pointed out, many nodes are expected to be hosted, which effectively means NTP service will be used as a Time Provider.

And the easiest and cheapest option is to use default NTP configuration, which means using NTP pool in most Linux distros (link).

Thus, hosted node deployment exposes a validator to a risk of an NTP attack, which questions whether the “rational majority” is a realistic assumption. A bounded rational model may be required.

Conclusion

The main question that such a model should answer is how security properties of a BFT protocol are affected, if it relies on public Time Providers (or Public Services).

In general, if a protocol relies on a centralized or a relatively centralized service, then an attack could be possible on the latter, leading to compromising of security properties of the former protocol. We will discuss this in more details in the follow-up posts.