This is a follow up of Time as a Public Service in Byzantine context.

Many Proof-of-stake designs assume validator clocks are roughly synchronized. Often the clock sync property is taken for granted, but, in reality, clocks should be synchronized with some protocol. Thus, such clock sync protocol becomes a target for an attacker.

Particularly, in beacon chain protocol, if a validator's clock disparity is more than one epoch, then its attestations won't be included in a chain by correct processes. Due to inactivity lick, the validator will be penalized for not participating in the protocol.

In the post, we analyze how the clock sync property affects security models of Casper FFG with inactivity lick, using beacon chain protocol as the main example.

Clock sync property and time attacks

We assume that the clock property does not hold by itself, but should be enforced by participants with some clock synchronization protocol.

In most cases, NTP protocol is used to synchronized clocks, often with a default configuration. Such time source setup is vulnerable to attacks.

However, other choices of time source setup require additional money, efforts and/or knowledge. In a public permissionless system that can become a problem for most participants.

To model that, we assume that a validator has either set up its clock correctly, so that the clock sync property holds or done it incorrectly, so that it can be manipulated by an adversary.

We also assume no internal clock sync protocol among participants. We also assume for simplicity that clocks are synchronized against common time standard, e.g. UTC. Assuming coordination during clock setup conflicts with uncoordinated rationality models and is not important for the purposes of the post.

Relative cost of time attack

Time attack is attractive because its cost can be much lower than the cost of validator deposits.

Assuming, there are 300K validators and 10K nodes, there are 30 validator per node on average. To perform 51% attack directly, an attacker should own 150K validators, which would cost 4.8M ETH (about $1.3B at the time of writing).

However, if many validating nodes use NTP with insecure setups, they can be attacked with much lower cost (see, for example, one, two, three).

For example, NTP pool currently has about 4K time servers, however, serving NTP requests does not require lot of traffic (e.g. link), so an attacker needs several thousand IPs and several servers to serve them. As the infrastructure can be reused to attack multiple nodes, the per validator cost of such attack will be negligible compare to the cost of a validator deposit (32 ETH, about $8.6K at the moment of writing) or compared to the cost of several deposits (as there will 30 validators per node, onn average, using the parameters above).

Casper FFG and Beacon Chain

Our main goal is to study beacon chain protocol, however, the same results apply to other protocols based on Casper FFG with inactivity lick, if a time attack can lead to the same consequences.

The main property of beacon chain protocol that we are using is that if a validator clock is slower than others (above some threshold), then others will ignore its attestations. So, if an attacker can slow down someone's clock then it can effectively isolate it from nodes with correct clocks. However, the isolation is one way, since the slow validator can see others messages as early ones.

So, we make additional assumptions about a generic protocol:

- it's based on Casper FFG with inactivity lick, which gradually penalize inactive participants
- the protocol assumes an upper bound on message delay, so that if it's exceeded then the sender will be deemed inactive. And, thus, inactivity lick applies.

Thus, the essence of the time attack is to partially isolate some participants from others, where partially means that fast participants can not see messages from slow ones, while the slow ones can see messages from fast participants as too early.

Security models

Honest Majority Model

and Uncoordinated Rational Majority Model

are not particularly interesting from a theoretical perspective, since, basically, an honest/rational should set up its clock correctly. However, this questions applicability of the two models.

Bribery Model

is more interesting and realistic, in that perspective.

Honest Majority Model

An honest validator should set up its clock correctly, so there is no need to treat time attacks in a special way.

In practice, however, honest majority model looks extremely unrealistic from clock set up point of view, as in public permissionless system, most validators are expected to use the default NTP setup, which is vulnerable.

Uncoordinated Rational Majority Model

A rational validator has choice: the correct clock setup option incurs additional costs while the incorrect one makes validator vulnerable to attacks.

Under uncoordinated rational majority model and assuming the correct setup cost is less than the cost of being vulnerable to time attacks, an uncolluding rational validator should set up its clock correctly.

In practice, it’s not clear if a real low-staked validator has enough incentives to behave rationality. Or from another perspective, the cost of correct clock setup can become too high (not justifying countermeasures against apparent risks). For example, if a validator uses a hosting service it could be a problem to attach GNSS receiver to its node, for example. And using some service (e.g. provided by the hoster) exposes to a possible attack via the additional dependency.

Bribery Model

We assume an adversary can bribe any validator, however, its budget restricts its power. To model clock attacks, we assume that the adversary has two bribing options:

- full validator control
- control of a validator clock, when it’s set up incorrectly

We assume that clock control option costs much less, perhaps all incorrect clocks can be controlled with an attack of a fixed cost. Basically, each incorrect clock can lower cost of a successful attack on the protocol.

We use $A$

, $B$

and $C$

letters to designate three disjoint sets of validators: fully controlled by the adversary ($A$dversarial), clock controlled by the adversary ($B$ribed) and correct validators ($C$orrect). We use N

to designated the set of all nodes, obviously $A \cup B \cup C = N$

We assume the adversary cannot fully control majority of validators. However, we illustrate how incorrect clocks can help adversary violate protocol safety or liveness more efficiently (using less budget).

Adversarial majority case

$|A| \lt \frac {|N|} 2$

, however $\frac {|N|} 2 \lt |A| + |B| \lt \frac {2 |N|} 3$

.

In the case, the adversary cannot directly control majority of validators, but it can control them indirectly with lower cost, via time attack of validators who set up their clocks incorrectly.

In the case, the adversary can hasten the clocks of validators it can control - $A \cup B$

, so that honest validator clocks appear slow. Thus, messages from $C$

will be ignored by $A \cup B$

validators, and correct validators will be losing their balances in the “adversarial” chain.

As $A \cup B$

constitute majority, but not supermajority, they can break liveness but cannot justify/finalize epochs. However, as correct validators losing their balances, at some point of time, the adversary will be able to justify and finalize epochs.

After, the adversary has eliminated correct validators, it can slow down clocks of B

so that their messages are ignored too. Depending on the relative sizes of A

and B

it may require several iterations, so that slowed down clocks constitute minority.

After eliminating B

, the adversary gains full control over the network.

Note, that the validators which are fully bribed by the adversary do not lose their balances, why correct ones do. That lowers cost of the attack too, since it avoids slashing and/or balances elimination due to inactivity lick. I.e. the A

validators can be “rented” instead of “bought out”, since their value is preserved during the attack.

Attack cost calculation

Let’s assume the following:

- full control bribing costs k

times more than clock control, $k \gg 1$

- p

is the fraction of validators which set up their clocks incorrectly, $p \lt \frac 1 2$

- it costs c

to fully bribe a validator

51% attack requires the adversary to bribe $\gt \frac {|N|} 2$

validators, which costs $\gt \frac {|N|c} 2$

.

Bribing ${|N|p}$

validators to control their clocks costs $\frac {|N| p c} k$

. Additionally, the adversary needs bribe $\gt (\frac 1 2 - p)|N|$

to be able to control majority of clocks, which costs $\gt (\frac 1 2 - p)|N|c$

. In total, the attack with eliminating C

and B

first costs $\gt |N|c(\frac 1 2 - p(1-\frac 1 k))$

.

Dividing latter by the former, the elimination

attack costs $1 - 2p(1-\frac 1 k)$

fraction of full 51% attack. In reality, k may be very big (see above), so that one can ignore $\frac 1 k$

term.

In result, if many validators have incorrect clocks (i.e. near half of them), then the cost of the attack becomes very cheap.

Adversarial minority case

$|A| + |B| \lt \frac {|N|} 2$

.

If adversary cannot control clocks of majority of validators, it can eliminate partially bribed validators B, by slowing down their clocks. After their balances become low enough, it can bribe a bit more than a third of the rest of validators and then it will be able to violate liveness (by voting differently than correct nodes).

Since $|A| + |B| + |C| = |N|$

then $|C| \gt \frac {|N|} 2$

. Liveness violation condition $|A| > \frac {|C|} 2$

, which means $|A| > \frac {|N|} 4$

and $|B| \lt \frac {|N|} 4$

. To violate liveness without eliminating B

first, the adversary need bribe $\gt \frac {|N|} 3$

validators. Thus it can reduce the cost of such attack by less than $\frac 1 {12}$

.

Conclusion

Attacks exploiting inactivity lick are not very realistic, since it will take log time for an inactive balance to become very low. Thus, the attack will be detected by administrators.

However, the purpose of the post is to illustrate that controlling clock of validators can be very efficient ingredient of a complex attack, since many validators can be isolated from the rest, if the former ones set up their clocks incorrectly (so that they are vulnerable to NTP attacks).

It’s likely that in public permissionless system, there will be many such validators, since they will often use Linux distros, which use NTP pool by default. The secure time source set up can be costly and requires certain expertise in NTP or time source setups.

It’s also unlikely that in the case of hosted deployment, time source options like GNSS receivers or Radio Wave clocks are available.