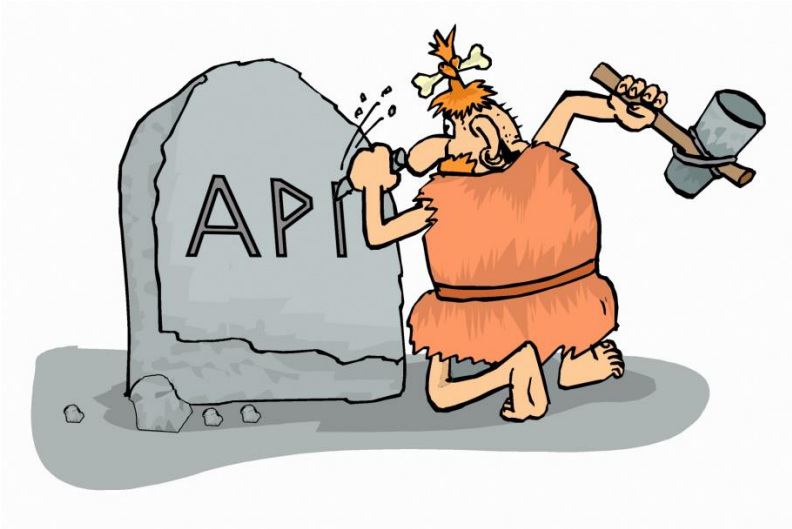
Casting APIs in Stone?



In an ideal world APIs evolve in a strictly controlled manner where everybody upgrade when a new version arrives. Unfortunately, this is seldom the case for any API implemented by multiple parties. Consider the following:

- One user group needs a new piece of information
- The security team wants to deploy a new encryption algorithm

Old Solution: We create new API version!

Well...is the new piece of information and the new encryption algorithm actually related? Probably not, effectively creating an *undesirable dependency* on a new version.

New Solution: Let each party publish an ***Authority Object*** describing their *current capabilities* like *Extensions*, *Algorithms*, *Keys*, *Service URLs*, *Trust Anchors*, etc. That is, *before* you invoke a party, you retrieve their ***Authority Object*** and adapt the call accordingly. This may seem like overhead, which is true. However, ***Authority Objects*** can usually be *cached* since such data is intended to be valid for a certain period like an hour or a day.

At a certain point in time, the API user community will surely find it worthwhile creating a brand new version of everything but even that can be advertised in an ***Authority Object***.

In Saturn, which was designed to last for *decades*, this scheme has been extensively used, not only for showing capabilities, but for establishing trust (in cryptographic sense), between the participants. The latter required *digitally signed* ***Authority Objects***.

The ***Authority Object*** concept also permits the introduction of *entirely local* or "experimental" extensions without disturbing the regular use of the API. Extensions are preferably expressed as URIs. That is, with some care, you can apply CI (Continuous Integration) even to APIs used by thousands of external parties!

Note that the actual *publisher* of an ***Authority Object*** may not necessarily be the same entity as the one being published.

A somewhat dated “defensive publication” covering this topic in more detail is available at:
https://cyberphone.github.io/doc/defensive-publications/authority-objects.pdf