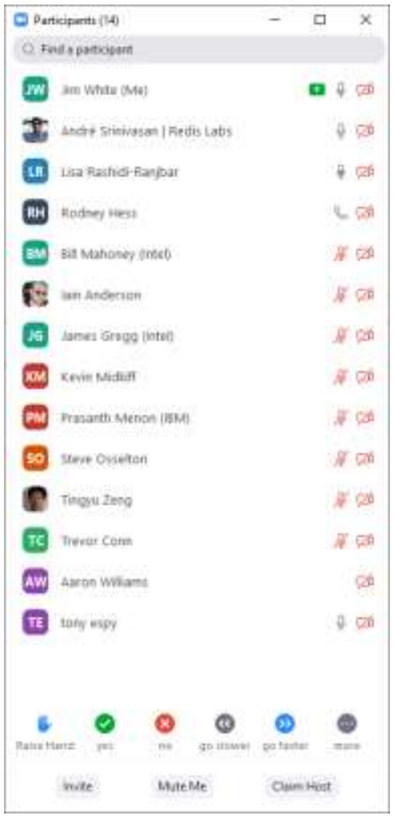
Architect’s meeting, May 18, 10am PDT

Attendance:



Participants (14)

Find a participant

- Jim White (Me)
- Andre Srinivasan | Redis Labs
- Lisa Rashidi-Ranjbar
- Rodney Hess
- Bill Mahoney (Intel)
- Iain Anderson
- James Gregg (Intel)
- Kevin Midkiff
- Prasanth Menon (IBM)
- Steve Osselton
- Tingyu Zeng
- Trevor Conn
- Aaron Williams
- tony espy

Some attendees may have joined after the call started when this record was captured.

- Explore process and procedures for vetting 3rd party packages. This initially came out of the security working group as a need to vet 3rd party packages for security issues (see #1947). The community now has a larger interest in exploring it from a license perspective, size/performance impact, better alternative, as well as a security vetting. James Gregg has done research to explore options. James will lead the initial discussion (see https://github.com/jamesrgregg/1947-explore).
    - Listen to Security WG recording (Password: 6U#=Q%?^) for context and more information.
    - Implementation considerations – how to trip the report
    - 3 – 4 places to apply (use cases/requirements)
        - Existing code (skeletons in closet)
            - Covered by Snyk
        - Code in holding (analysis before we accept code from holding)
            - Require developer that wants to move use Snyk to provide a report
        - PR with new dependency – how do we vet
            - For just Go or for other languages (C, Javascript)?
            - How can we determine when a new package has been brought in?
            - Is there tools/bots/etc. to find when go.mod has an addition?
            - Action - James to research
        - Possibly look at runtime dependencies (like Mongo, Redis, etc.)

        - Covered by Synk
  - Specific requirements (what are we scanning)
    - License compatibility (within our pipeline – this doesn't happen today – James to research)
      - Covered by advanced Synk report
    - Security CVE issues
      - Covered by Synk
    - Security maturity of a project (are they doing static analysis of their code, software bill of materials, vulnerability process) – may be tough to do in automated way
      - Manual – paper study?
      - To automate this, we'd need place to run python script
    - Is it an active project (number of developers, releases, recent commits) – but be careful as to not exclude simple package
      - Manual – paper study?
  - Non-functional requirements
    - Collection of metrics automated, but review manual
    - Don't want to introduce too heavy of lift (not Fossology)
    - Large expense for tooling
    - Require a lot of support from LF
    - Don't want change to code base (change go.mod)
    - Something that report accurately
  - Open question – when does it apply
    - Frequency
    - Timing
- Explore how we can inform people of architectural decisions (especially the small ones vs ones done via ADR). Can we tag items in the project boards? If so, which ones and how? (comes from the Hanoi planning meetings)
  - Requested to be top of list for next meeting
- How should we apply semantic versioning to modules? When do we update the minor and major versions of modules? (comes from the Hanoi planning meetings)
- PR Template for conventional commits is now in place for all repositories for all PRs but without TSC approval. It doesn't appear to be affecting any problem. We need to finalize the shape of this and officially approve the template by the TSC.
- Extract of Device Service requirements to ADR legacy - what are all the pieces that need to be moved there?
- Per the Hanoi planning conference - we need to better define "bound checking" so that a design (and eventual implementation) can be brought forth to meet the requirements
  - Currently considering limiting the number of operations that can be performed on a service (like a device service) over a period of time or setting the max request size (that lends to DoS attacks)
  - Can the solution be more globally applied?
- Design metadata about the "gateway" or host platform (identity, location, …)