

Repository Statement

We wish to create a store of identities and audits for the Daml Package Manager project. The identities will be used to identify **auditors** and **publishers** in the infrastructure we are designing as a part of this proposal. They should be managed in a way that gives assurances that each identity is who they say they are. This will allow users to decide which auditors they trust statements from. The identity store shouldn't make any guarantees about the security of artefacts generated by identities, this decision will be made by the users themselves. Any assurances surrounding an identity will be backed up with evidence, and a clear policy will determine if this evidence is sufficient. This evidence will then be stored in the repository git tree. A security policy will be enforced by **gittuf**, which will log all commit activity in an append-only log. An elected council will vote on decisions, which will also be auditable via **gittuf**. This document proposes a formalisation of the processes involved.

The audits will be stored in the same repository, allowing us to restrict auditors to only those who have been registered as an identity.

Identities

An identity in the repository is exactly one GPG key. Thus an organisation authoring packages must sign these packages with their single organisation key. Likewise an auditing organisation must sign their audits with their key.

Audits

An audit should classify a particular DAML artefact as safe or unsafe. This classification should be signed and dated by an identity in the repository.

The Committee

Any notion of trust in the repository stems from the committee. No one else is given the ability to Add/Rotate/Revoke keys. The initial committee is comprised of three members:

- **Digital Asset**
- **Canton Foundation**
- **Obsidian Systems**

Each of which will be associated with a GPG key, representing the member's vote. The committee members have a responsibility to keep the keys safe: they are the source of trust for the repo. We would advise generating a primary key on an airgapped machine, then exporting a subkey to use.

Note: Votes are managed via **gittuf**, which is currently configured to accept an approval via the GitHub PR web interface by the correct GitHub account. This is very convenient for the committee, but also a potential attack vector. We could assert that a committee github account **MUST** be protected by 2FA. Is this acceptable?

Adding to the committee

The proposal outlines an initial committee of 3 members, but as the repository expands in size, we will need to think about adding more members to the committee and additionally changing the threshold of votes required for certain operations. Adding a new committee member is a much more dangerous operation than just adding keys. We will need to have tighter checks to ensure a committee member is who they say they are AND can be trusted. This could mean verifying in-person.

For this document we will assume that the committee consists of the three members described above, and we will add to the relevant processes later once we decide on secure governance for adding new members.

Repository Format

The file structure we propose is:

```
identities/
  <id>/
    meta.json
    FA31...A3/
      key.asc
      evidence/
      ...
    B491...D5/
      key.asc
      revoke.json
      evidence/
      ...
audits/
  <id-fp>-<package-id>-<date>.json
  <id-fp>-<package-id>-<date>.json.sig
  <id-fp>-<package-id>-<date>.revoke.json
  <id-fp>-<package-id>-<date>.revoke.json.sig
```

Where:

- **identities/** contains directories corresponding to each identity in the repo
 - **<id>/** is a directory containing the history of all keys for an identity. The naming convention here is that **<id>** is the **kebab-case** version of the **Name** field of the **gpg** keys.
 - * **meta.json**: Metadata about the identity
 - * **<key>/**: Directory containing information about a particular key
 - **key.asc**: The exported GPG public key. Generated via **gpg --export --armor**
 - **revoke.json**: A json file indicating that the key has been revoked and why.
 - **evidence/**: Directory containing evidence used by the committee when evaluating the key's identity
- **audits/** contains audits:
 - **<id-fp>-<package-id>-<date>.json** is an audit of **<package-id>** by **<id-fp>** on **<date>**.
Note: No claims are made by this filename, it's solely to reduce collisions. All of the significant content is contained within the file. It conforms to the specification outlined in **schemas/audit-report.json5**.
 - **<id-fp>-<package-id>-<date>.json.sig** is the detached signature of the above file.
 - **<id-fp>-<package-id>-<date>.json.revoke** expresses a revocation of the above file.
 - **<id-fp>-<package-id>-<date>.json.revoke.sig** expresses the signature of the above file.

Note: The “one-entity, one-key” relationship still stands here. The multiple keys living under **<id>/** are simply previous rotated keys. Only one will be considered active.

We decide that a key for an identity is valid by checking that it hasn't been revoked.

Common Processes

Adding an Identity

Identities are added via a GitHub PR in the repository. The PR must contain:

- `identities/<id>/meta.json`: A metadata file containing:
 - Organisation Name
- `identities/<id>/<key-fp>/key.asc`: A *single* gpg public key

The user opening the PR will then describe what option they will use to verify the identity. It is then handed over to the committee to:

- Generate `evidence/` files that attests that they have deemed the evidence appropriate

Acceptable Proof of Identity

The following could be acceptable forms of proof:

- Proof that they control a well-known domain: Publish a document on this domain containing their public keys. This is what is used by the `keyoxide` project.
- Confirmation via an organisation email that the keys contained within the PR do reflect the keys of the organisation.

Note: We're very open to suggestions here!

Each committee member must then independently verify that they accept the evidence provided, and generate:

- A `<verifier-id>.json` with a declaration containing:
 - What evidence was observed
 - That the evidence was deemed sufficient
 - The public key fingerprint of the committee member used to sign this file.
- A `<verifier-id>.sig` Which is the signature of the declaration by the key defined above.

These are then added as commits to the PR, signed using the key linked to their vote. We will then enforce via CI that only the votes by committee members that have submitted evidence are valid.

Key Removal Operations

The following outlines the various processes where keys are removed, and how this affects the audits under `audits/`:

Rotating keys

To rotate a key, an entity must open a PR with the following files:

- `identities/<id>/<old-key-fp>/revoke.json`: with a note stating that the key has been rotated
- `identities/<id>/<new-key-fp>/` created with:
 - `key.asc`, the new public key
 - `evidence/` containing:
 - * `rotation.json`. A statement that key **X** has been rotated to **Y**
 - * `rotation.old.sig`. A signature from key **X**
 - * `rotation.new.sig`. A signature from key **Y**

A PR that rotates a key, where the rotation commit is signed by the new key, can be merged with 0 approvals as long as CI verifies that `rotation.old.sig` and `rotation.new.sig` have the correct signatures.

Note: All audits that were signed by this key after it was rotated will be considered invalid.

Compromised Key

If a key has been compromised an entity must open a PR with the following files:

- `identities/<id>/<key-fp>/revoke.json` : Specifying that the reason for revocation is that the key has been compromised. A date of when it was compromised should be specified.
- `audits/*.revoke.json` : The audits authored by this key after the compromised date will be revoked.

No evidence needs to be provided for a compromised key. This action should require 0 approvals, so long as CI verifies that the revocation commit is authored by the key in question.

Note: All audits that were signed by this key after it was compromised will be considered invalid

Malicious Owner

If a key holder is malicious they should be removed from the repository, along with all of their audits. This requires a vote of 2 committee members. A PR should be opened containing:

- `identities/<id>/<key-fp>/revoke.json` specifying that the reason for the revocation is `malicious`
 - `evidence/` containing all evidence for committee members that the key holder IS malicious.
- `audits/*.revoke.json` : The audits authored by this key will be revoked.

Note: All audits authored by this key will be considered invalid

Adding an Audit

Modifying the `audit/` directory is subject to the following rules:

- Adding an audit can be done with 0 approvals, so long as the signature of the commit adding the audit is registered in the repository.
- The `audits/` directory is append-only, meaning that an audit can only be removed via adding a `.revoke.json` file.
- Every audit must conform to a later defined specification
- Each audit must also have an accompanying `.sig` file, to prove that it was authored by the stated author

Responses to Questions

- *What guarantees does the repository provide about the keys inside?* The repository only aims to assert that a key belongs to an organisation/entity, and will provide evidence towards this assertion, but won't give any guarantees about the security of the organisation. The user can then decide to use this organisation for audits.
- *Should we enforce that keys should be on CI machines?* Again, I don't think we can make any guarantees about the security of the keys, and that we can leave that decision up to the users. A user could decide that they don't want to use any packages from a company because they don't publish artefacts via CI/CD.
- *Can a committee member approve their own pull request?* Github prevents users from requesting a review from themselves; the mechanism that **gittuf** uses to attest to a PR. Additionally, even if a malicious committee member did open a PR under someone else's name, the PR would still require $n - 1$ votes from the committee.