Note for convenience: pre-generated identities are in the /demo/identities folder

Defining the journey

<u>What is the journey?</u>

- Let's look at the steps of the journey:
  - Family purchases insurance coverage
  - Individual from family seeks medical treatment and hospital issues invoice to individual
  - Individual goes to insurance company, makes a claim with the invoice
  - Insurance company checks authenticity, probably with hospital and with other insurers to check for double claims
  - Insurer reimburses what is due to the individual in coverage from the registered policy
- While authenticity is only used once in the above description and in the brief, we can identify many implicit assurances and assumptions that have to be verified along the way in the journey:

  - **Insurer:**
    - Is the family already covered by us?
    - Is the family real?
    - Does the individual belong in the family?
    - Is the individual alive?
    - Can this family be covered by us?

    Family/Individual:
    - Is the insurer real?
    - Is my privacy and data being protected from everyone?

  - **Hospital:**
    - Is the individual I am treating who he/she claims he is?

    Family/Individual:
    - Is the hospital real?
    - Am I covered by insurance if I choose this medical option?
    - Is my privacy and data being protected (specifically, from the insurer)?

  - **Insurer:**
    - Is the family covered by us?
    - Does the individual belong in the family?
    - Does the individual qualify for cover?

    Family/Individual:
    - Am I insured for this medical procedure?
    - Is my medical data being protected (what if the insurer uses this to revise my premiums in response to perceived risk?)

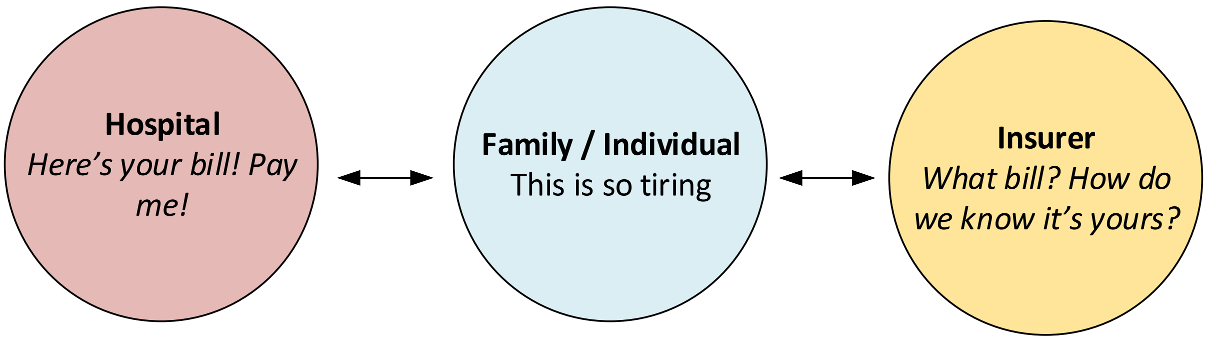
  - **Hospital:**
    - Is the insurer real?
    - Should we trust the insurer with patient data?
    - Will not sharing data affect our standing amongst other insurers?

Insurer:
  - Is the hospital real?
  - Is the hospital sure that they treated the person who is claiming?

- **Insurer:**
  - Are we paying the right party?
  - Has the invoice amount been tampered with?

  Family/Individual:
  - What is the status of my claim?
  - How much am I to be reimbursed?

- These questions of verification mostly stem from inherent self-interests in the well-being of each respective entity.
- The closely coupled and often opposing nature of each entities' interest leads to inherent conflicts in the process
- This inevitably results in a potentially long and arduous journey to verify and ensure that no one's interest is breached.
- The journey can thus be described as a journey of **identification and trust**.


Hospital
Here's your bill! Pay me!
Family / Individual
This is so tiring
Insurer
What bill? How do we know it's yours?

How to digitize?

- Guiding principles/assumptions (from own painful experiences) - To succeed, digitization must:
  - Provide a tool for the process, not a hinderance
  - Leave an established process alone. Revamping an existing, widely practiced and accepted process can lead to abandonment of a digitized solution
  - Bring real value to every user in the journey, whether perceived or real
  - Be universal in applicability, without shoehorning any user into the solution
  - Factor in multi-party interests beyond simply delivering value (e.g. sovereign interests, optics, etc)
- Keeping this in mind, a possible key value proposition appears to be **facilitating identity trust**.
- One way to do this is to have **self-contained trust verification in every two-way party process**

Who benefits and how from digitizing identity trust?

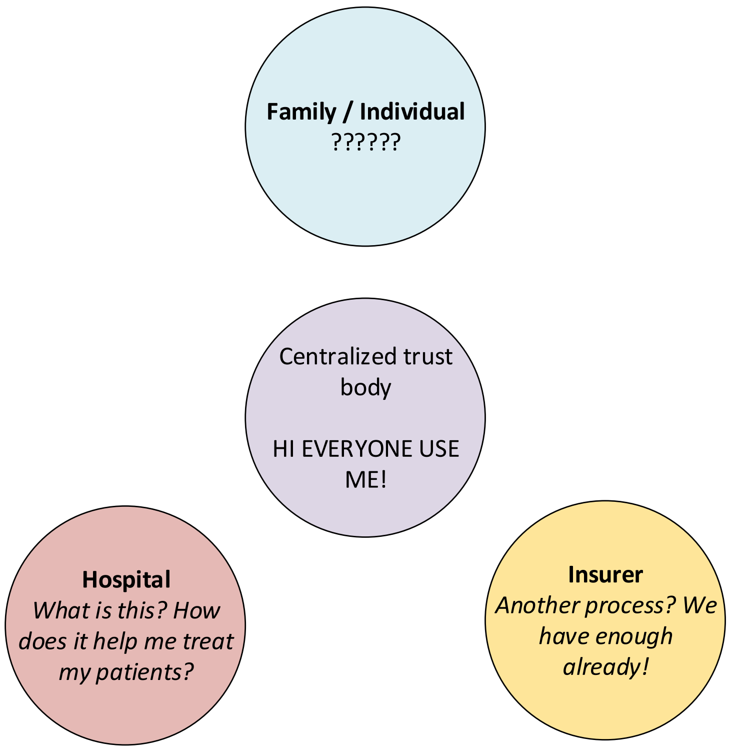
- A trust solution would streamline the process and reduce or answer most of the verification questions highlighted above quickly and relatively painlessly (e.g. no digging through multiple physical paper records to find if an invoice has been processed by another insurance company before)

- Families can be described as the target beneficiaries as they would have a reduced time-to-claim delay
- Nevertheless, there are benefits for the insurer and the hospital as entities too
- These benefits would primarily be associated with cost savings in reduction in the amount of liaison and verification work that has to be done in a claim (KYC savings)
- We can thus define a few groups of users of such a trust network:
    - The family, and by extension an individual in the family
    - The hospital
    - The insurer

Solutioning for a trust framework

<u>Initial analysis</u>

- Healthcare / insurance sector (as with many other sectors) is extremely fragmented with multiple entities delivering non-standardized services
- Introducing a central system of trust (a centralized notary) or a body coordinating the trust might be a solution
- Introducing such a central trust body, however, brings about other issues:
    - Is this central system of trust dedicated to serving the healthcare / insurance sector solely? Won't that be rather specific / expensive / overkill if participants in the industry are the only ones to have pay for it?
    - Why should I trust the central entity? I still need to hold my own data to verify the truth! This leads to every fragmented entity holding some data, increasing the chance of a breach or compromise
    - Why would they want to sign up for such a centralized notary anyway? Afterall, optimizing their own due-diligence process today might give them a competitive edge over others in the industry
    - What will be stored in the central notary? Will it contain all records? What happens if it gets hacked or compromised?
    - Does this mean we don't insure anyone or treat anyone if the system goes down?
    - Worst of all, why should anyone trust the centralized notary? What areas of jurisdiction does it cover?


Family / Individual
??????
Centralized trust body
HI EVERYONE USE ME!
Hospital
What is this? How does it help me treat my patients?
Insurer
Another process? We have enough already!

- We need to therefore seek a generic solution that delivers a lean layer which focuses on addressing and delivering identity trust that is independent on data and that doesn't depend on data
- This would in theory demote the notary from being a notary to merely a 'peer'. For example, instead of verifying or validating, it can simply state that it has observed an entity with a key an act as a secondary source of truth.
- This has the added advantage of adding non-industry specific trust to be established that can be reused for other industries, thereby
    - keeping costs low
    - improving uptake and acceptance via network effects
    - having no key data stored, i.e. fulfilling the goal of trust and data decoupling, thereby preserving data privacy
    - permitting reduced coupling, which can translate to international portability between jurisdictions, i.e. each to their own in interpreting, using and applying the facility
- Also, as mentioned above, we need all information to be self-contained in each process, such that no additional dependencies from three parties on trust are required
- This leads on towards the idea of generic, distributed and verifiable identities, against which data can be associated in the confines of each individual entity
- Natural persons should by default be only known by their public keys and that should be the only data that is stored to minimize identity exposure, thus protecting privacy to some extent

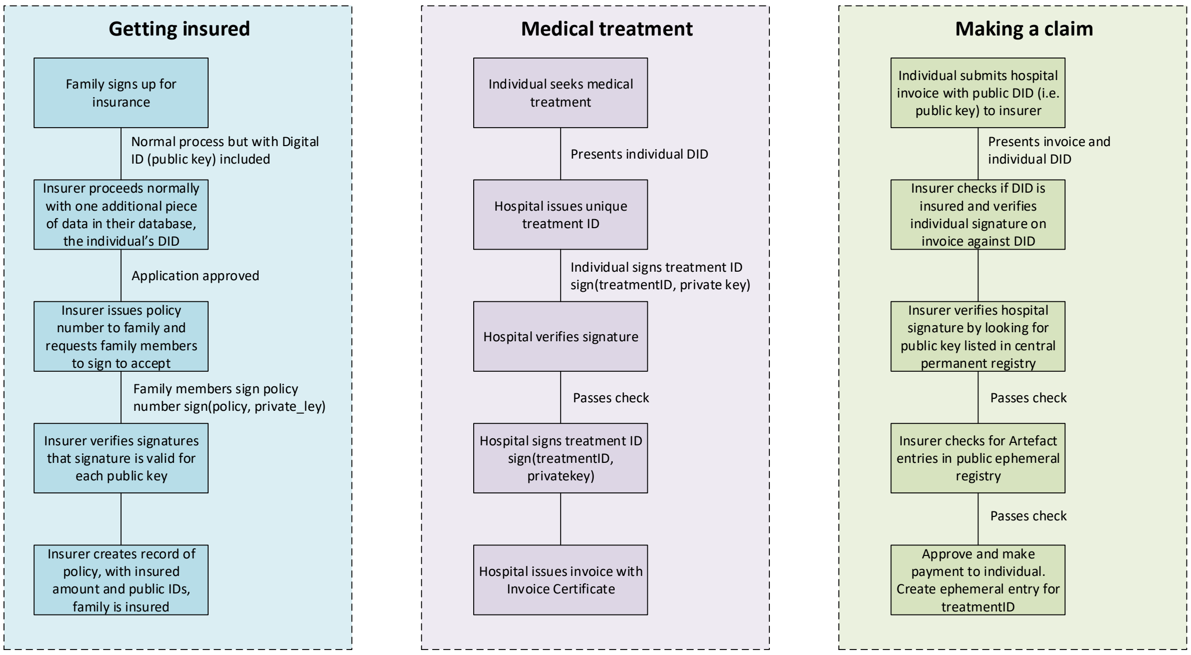
Distributed identities

- Use well established PKI style techniques to verify identity!
    - Industry accepted, well developed standards, e.g. RSA
    - Reasonable performance
    - Good track record

- Permits each individual or entity to verify an identity independently through signatures, with minimal access to a central network
- No technical integration is required, but automation can be applied. This might help uptake.
- Central network will comprise a registry for registration by all seeking public visibility and enhanced standing.
- This would contain public keys of entities along with optional information the entity wishes to reveal or make public. For instance, an individual would want others to know they exist or are registered in the system, but not anything else, whereas an insurer or hospital would conceivably want to include other information like the corporate registration, address, license and so on. Interesting thought: license numbers could also be a signature from an issuing body relying on the same PKI infrastructure.

Mechanics

- How to apply?
  - Unique identities can be assigned to each entity in the form of private/public key pairs. The public key forms their own Digital ID (DID). This term is used interchangeably with public key below.
  - Each entity can signature various artefacts (one-time identifiers in this case) which can be used as thumbprints to track and verify that a piece of data is indeed associated with them
- Proposed user story:


Getting insured
Family signs up for insurance
Normal process but with Digital ID (public key) included
Insurer proceeds normally with one additional piece of data in their database, the individual's DID
Application approved
Insurer issues policy number to family and requests family members to sign to accept
Family members sign policy number sign(policy, private_ley)
Insurer verifies signatures that signature is valid for each public key
Insurer creates record of policy, with insured amount and public IDs, family is insured
Medical treatment
Individual seeks medical treatment
Presents individual DID
Hospital issues unique treatment ID
Individual signs treatment ID sign(treatmentID, private key)
Hospital verifies signature
Passes check
Hospital signs treatment ID sign(treatmentID, privatekey)
Hospital issues invoice with Invoice Certificate
Making a claim
Individual submits hospital invoice with public DID (i.e. public key) to insurer
Presents invoice and individual DID
Insurer checks if DID is insured and verifies individual signature on invoice against DID
Insurer verifies hospital signature by looking for public key listed in central permanent registry
Passes check
Insurer checks for Artefact entries in public ephemeral registry
Passes check
Approve and make payment to individual. Create ephemeral entry for treatmentID

Eliminates direct KYC liaison and potential unnecessary information exchange between insurer and hospital

Solution

<u>Logic flow</u>

Hospital Invoice Workflow

Step	Purpose

Generate invoice hash that includes invoice number, amount, patient DID and a nonce	Ensure that there is a method of detecting if the invoice has been tampered with
Get the patient to sign the invoice hash with their private key	Acts as acknowledgement that patient was treated
Verifies patient signature against patient public DID	Ensures that the patient is who they claim they are, i.e. they are in ownership of the private key corresponding to their DID
Sign invoice hash	Provides means to verify that hospital was indeed one who issued invoice
Generate certificate object	Data of two signatures and hash to prove to insurance provider that the invoice is authentic

Insurer Invoice Verification Workflow

Step	Purpose
Obtain DID from insured customer	Make sure customer is insured by insurer
Do public registry search for issuing hospital / clinic	Ensure that invoice issuer is 'reputable'. Record down their public key (i.e. DID)
Reconstruct invoice hash and match against certificate	Verify that the invoice was not altered
Verify signature of individual on invoice hash using individual DID	Ensure that individual was indeed the one who accepted the invoice hash from the issuer
Verify signature of issuer on hash using individual DID	Ensure that issuer was indeed the one who signed the invoice hash
Check if invoice has already been registered	Prevent potential double claims from other insurers (?)
Invoice should be valid	

<u>Scope pruning in lieu of recommended timebox constraints</u>

- We will assume that all users have 'digitized' to some extent prior and are emailing each other the invoice. This would be required to copy/paste the large chunks of text.
- As a follow up to this, we shall be sticking with pure text. In a prototype, the certificates could be a mnemonic or even a QR code. It is not practical to key in a 1000+ character string in an operational or clinical setting.
- We assume that each app has an IT person to set it up
- We will ignore authentication requirements for the applications AND databases
- No caching
- We will focus on a proof-of-concept verification of the document (including its uniqueness)
- We assume that there is some means to securely add public entries into the public registry exists (i.e. the OTP faucet has controlled access)
- We assume that the length of time that the individual has to claim is limited to 30 days.
- We assume that no individual (natural person) wants their public keys published and there is no such registry that should exist (i.e. a publicly accessible list of all citizens, for example, of a country)
- We will assume that this is an ideal world and no DID / public key revocation is needed.
- We will not attempt to implement dynamically generated identities for every interaction with each different entity. This would be a better solution to preserve privacy of the

individual. Blind signatures are a possible solution to this. Interesting question: Do I want to let everyone in my family know about the insurance claim?

- Error handling will be limited to bare essentials.

<u>Technology Stack</u>

- Node.js (Express.js + Nunjucks (templating)) + MongoDB backends + YAML configuration
- Bootstrap + jQuery (couldn't justify to myself to use a full react app for a single page)
- RSA-PSS public private keys with SHA hashing for identity verification
- HTTP microservice for registry implementing simple identity lookup API
- Docker for containerization
- Docker swam for resilience

<u>Component Details and Run Instructions</u>

- Identity generator
  - Generates ID files for consumption by all other components
  - Located at /standalone/individual/ generate.html
- Individual App
  - Simple offline page demonstrating generation of identity, loading of saved identity and signing of one-time artefact ID's
  - Signing performed on client side
  - Located at /standalone/individual/ sign.html
- Hospital app
  - Simple web application demonstrating generation of data to be printed on the invoice. Data comprises of:
    - Treatment ID (or more generically, the artefact ID)
    - Amount
    - Individual signature of treatment ID
    - Hospital signature of treatment ID
  - Signing is performed on server side, since we might have multiple payment points, demonstrating client/server interoperability
  - Features integration with registry to register self as an entity
  - Root and readme located at /applications/hospital
- Insurer app
  - Simple static page demonstrating verification of individual signature on insuring sign-up
  - Simple static page demonstrating validation of document using information from:
    - Insurance sign-up process (individual's public key)
    - Hospital invoice (certificate which includes Invoice ID, Amount, Individual Sign, Hospital Sign)
    - Hospital public key from registry search
  - Root and readme located at /applications/insurer
- Registry
  - Express.js implementing stateless HTTP API microservice
  - Simple API for entity registration with an OTP
  - Simple page to search for entities
  - Simple page to add new ephemeral data
  - Root and readme located at /applications/registry

Potential security issues

- There is no authentication on the registry, and specifically, adding new entities and the ephemeral store.
- The databases do not have authentication configured.
- Supplementary verification and a proper mechanism for identity generation is assumed to be external to the process. For e.g. verifying that an individual is part of a family is the role of the insurer.
- The identity file is not encrypted.
- There is no mechanic to ensure that a user does not control multiple keys. This could mean that the individual can own multiple insurance policies which would affect underwriting risk at the insurer. The only way to fix this would be to ensure that a user can only register and control one private key.
- Theft of private key. For e.g. if patient phone is lost, somebody else can simply claim the private key is his/hers. Potential mitigation – build data that is dependent on a physical feature of the individual (biometrics) into the public / private key pair. E.g. could be a multi-signature arrangement where one key is derived from PCA of the thumbprint.
- As above, we are assuming that the one in possession of the private key is the individual.
- Are we looking at the right identity vs is the identity real?
- Compromising the insurer's database can compromise the whole process, given that a false injection of a false public key for an individual can result in a fraudulent claim. This can be fixed by having a lookup or database of natural persons' public keys to check if they exist and are alive. However, this would be equivalent to publishing all public keys, as mentioned above, in some sort of a registry, which might lead on to privacy concerns.
- In the case of blind signatures, a plaintext message can be recoverable if RSA is used to blindly sign anything – cut and choose
- Limits not implemented for ephemeral store
- The registry for issued ID's is not implemented with real immutability and a hack or breach of procedure could potentially alter critical registry data in it. (mongodb is a mutable database) Provable (true) immutability of registry (use blockchain style database)