Release of Electronic Health Records Using Self Sovereign Identity

Problem Statement

In many healthcare systems around the world information on a single patient may be divided between many different hospitals, insurers, and doctors. At its most benign this problem can result in excessive amounts of paperwork, but at its worst a lack of information on a patient's medical history could result in improper diagnoses or treatments. This is an especially difficult problem to solve in systems like the US, which lack the level of centralization present in, for example, the UK's NHS. Self Sovereign Identity (SSI) offers an excellent way to create and store fully interoperable electronic health records (EHR).

SSI offers an elegant and efficient solution by putting patients in charge of their own medical data. Using SSI a given patient could store copies of their medical records as a collection of verifiable credentials and claims in a digital wallet, which could then be shared at the patient's discretion with any given healthcare provider. A full implementation of such a system, however, is beyond the scope of this project.

Assuming such a system has been implemented, a potentially life saving use case arises: in an emergency situation a first responder could access electronic copies of a patient's medical records. This could provide a first responder with vitally important information on a patient, such as blood type, diabetes diagnosis, medicinal allergies, etc. This project provides a basic proof of concept for this use case.

Implementation

The design of this project is fairly simple. There are four entities involved, two of which issue credentials and two of which receive them. A Doctor provides a patient, in this case Bob, with a verifiable credential representing some of his basic medical information. A state agency, in this case the Texas Department of State Health Services (DSHS) provides a First Responder with a verifiable credential representing his or her license as a Paramedic. Some time after both of these events occur, Bob gets in an accident and is attended to by the Responder. The Responder wants Bob's medical information, and so makes a connection with Bob and requests a copy of the medical record issued to Bob by his Doctor. Bob then responds with a request for proof of the Responder's license, which the Responder provides, at which point Bob provides proof of his medical record to the Responder, who then has access to the information contained therein.

An important note here is that this implementation has Bob creating the connection with the responder on the spot. Were Bob unconscious or otherwise impaired this may not be the case. To deal with such a situation it would be necessary for Bob and the First Responder to

make a connection ahead of time, for example through Bob's insurance company. I originally wanted to dive further into such a scenario, but decided not to because of time constraints. For the purposes of this demo let us assume that Bob can create the connection but cannot recall all of his medical information. The First Responder wants to ensure the information is accurate, which can be guaranteed using a Verifiable Credential.

Running the Project

The steps to run this project are as follows:

1. Open up 4 terminal windows or tabs in the folder containing the .py files. Each terminal will be used to run a separate .py file.
2. Run dshs.py
3. When prompted, run responder.py
4. Paste the connection from dshs.py into responder.py. The agency will then issue a credential to responder representing their license as a paramedic.
5. Run doctor.py
6. When prompted, run bob.py
7. Paste the connection from doctor.py into bob.py. The "Doctor" will then issue a credential to bob representing his medical record.
8. Responder.py will then prompt the user to enter a patient name. Enter "bob"
9. Paste the connection from responder.py into bob.py
10. Wait for the program to finish. At the end Responder.py should print out the proof received from bob.py, which will contain an example set of medical information.