ENHANCING THE SECURITY AND PRIVACY OF SELF-SOVEREIGN IDENTITIES ON HYPERLEDGER INDY BLOCKCHAIN

Co-authored by Manas Pratim Bhattacharya

Pavol Zavarsky

Sergey Butakov

Project report

Submitted to the Faculty of Graduate Studies,

Concordia University of Edmonton

in Partial Fulfillment of the

Requirements for the

Final Research Project for the Degree

MASTER OF INFORMATION SYSTEMS SECURITY MANAGEMENT

Concordia University of Edmonton

FACULTY OF GRADUATE STUDIES

Edmonton, Alberta

April 2020

ENHANCING THE SECURITY AND PRIVACY OF SELF-SOVEREIGN IDENTITIES ON HYPERLEDGER INDY BLOCKCHAIN

Manas Pratim Bhattacharya

Approved:

Pavol Zavarsky [Original Approval on File]

Pavol Zavarsky Date: April 14, 2020

Primary Supervisor

Edgar Schmidt [Original Approval on File]

Edgar Schmidt, DSocSci Date: April 27, 2020

Dean, Faculty of Graduate Studies

Enhancing the Security and Privacy of Self-Sovereign Identities on Hyperledger Indy Blockchain

Manas Pratim Bhattacharya
Department of Information Systems Security and Assurance Management
Concordia University of Edmonton
Edmonton, Canada
mbhattac@student.concordia.ab.ca

Dr. Pavol Zavarsky
Department of Information Systems Security and Assurance Management
Concordia University of Edmonton
Edmonton, Canada
pavol.zavarsky@concordia.ab.ca

Dr. Sergey Butakov
Department of Information Systems Security and Assurance Management
Concordia University of Edmonton
Edmonton, Canada
sergey.butakov@concordia.ab.ca

Abstract— **Self-sovereign identities provide user autonomy and immutability to individual identities and full control to their identity owners. The immutability and control are possible by implementing identities in a decentralized manner on blockchains that are specially designed for identity operations such as Hyperledger Indy. As with any type of identity, self-sovereign identities too deal with Personally Identifiable Information (PII) of the identity holders and comes with the usual risks of privacy and security. This study examined certain scenarios of personal data disclosure via credential exchanges between such identities and risks of man-in-the-middle attacks in the blockchain based identity system Hyperledger Indy. On the basis of the findings, the paper proposes the following enhancements: 1) A novel attribute sensitivity score model for self-sovereign identity agents to ascertain the sensitivity of attributes shared in credential exchanges 2) A method of mitigating man-in-the-middle attacks between peer self-sovereign identities and 3) A novel way of determining the reputation of a credential issuer based on the number of issued credentials in a window period, which is then utilized to calculate an overall confidence level score for the issuer.**

Keywords—self-sovereign identity, blockchain, Hyperledger Indy, data disclosure, credential exchange, man-in-the-middle attack, attribute sensitivity, reputation, confidence level

I. Introduction

Self-sovereign identity is an emerging field of digital identities where identities are implemented in a decentralized manner using distributed ledger technology or blockchain. Self-sovereign identity redefines digital identities by providing the identity holder autonomy and control of his or her identity even when dealing with multiple authorities. This contrasts with traditional identity management where identities are managed and controlled by central authorities [1].

A noteworthy protection mechanism of self-sovereign identities is controlled and selective disclosure of data. An identity holder has full control on what data to disclose to the verifier and can selectively aggregate data from multiple credentials into a proof presentation [2]. The holder can also prove to a verifier the knowledge of an attribute without revealing the attribute itself using zero-knowledge proofs [3].

Another important feature in a self-sovereign identity system are unique peer relationships. This enables one identity holder to form a relationship with another using unique decentralized identifiers (DIDs) [4] and keys, that are resolvable only by the parties in the relationship and no one else. A unique ID for every relationship is an implementation of the privacy-by-design concept that keeps relationships isolated from one another to ensure that they remain private in context and data [5].

This study explored self-sovereign identities with respect to security and privacy alongside its credential verification process on Hyperledger Indy, an open-source permissioned distributed ledger infrastructure for self-sovereign identities. It then went on to make relevant contributions towards enhancing the privacy and security as follows. By proposing a model of determining attribute sensitivity by a score, a technical solution is made available for Hyperledger identity agents to arbitrate what attributes or combination thereof is sensitive to share. For a peer relationship between two unknown peers, this study provides a method of detecting and mitigating man-in-the-middle attacks by self-signing credential attributes which assures the receiving party that the creator and deliverer of the message is the same party. The study further proposes another novel model for computing quantitative values for the reputation of and the confidence level in a credential issuer based on its rising or falling trend of the number of credentials issued using an exponential weighted moving average method with appropriate normalization. Credentials can be offered by any identity owner known to the ledger based on a schema, but its reliability depends on the reputation of the issuer. The afore-mentioned model computes a quantitative value for confidence level in an issuer at a point of time, for agents to make informed decisions on future data exchanges with a peer who has obtained credentials from such an issuer.

The remainder of the paper is arranged as follows: Section II describes the credential verification process in Indy. Section III proposes a novel attribute sensitivity score model that helps agents decide on risk-free exchange of identity attributes. Section IV describes a method of mitigating a man-in-the-middle attack between peer DIDs. Section V proposes a novel quantitative model for measuring reputation and confidence level for identity credential issuers. Section VI discusses and analyzes the propositions. Section VII provides a conclusion.

II. Credential Verification Process

A typical credential verification flow applicable to Indy is shown in Fig. 1. To obtain a credential, the receiver accepts a credential offer from the issuer on an established connection (relationship). The credential, containing the raw and encoded values for each attribute with appropriate cryptographic CL (Camenisch-Lysyanskaya) signatures [3], is issued to the identity holder who stores it in his/her digital wallet. Self-sovereign identities are manifested as DIDs (Decentralized Identifiers) [4] in a blockchain. The identities themselves interact with each other and the ledger via software entities called agents.

978-1-7281-5628-6/20/$31.00 ©2020 IEEE

To verify any credential, a verifier makes a proof request to the holder (prover) requesting certain attributes and predicates. Some of these are mandatorily required to be verifiable (e.g. social security number, age etc.) while some can be self-asserted by the identity holder (e.g. name, nickname or phone number). The verification consists of verifying whether the keys of the signatures match with that of the entity signing it. It is worth noting that, in verification process, the onus is on the verifier whether to trust an issuer of a credential, although Sovrin does propose a 'web of trust' for trusting attributes [6].

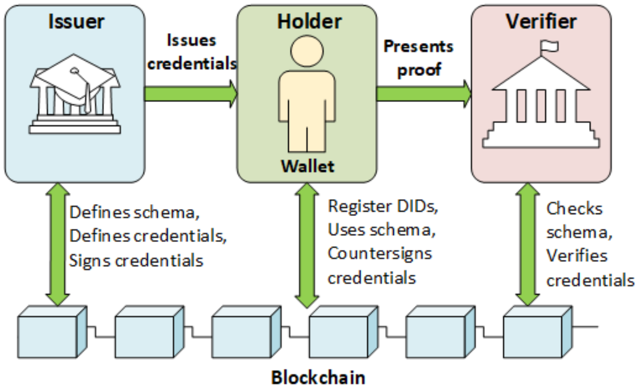
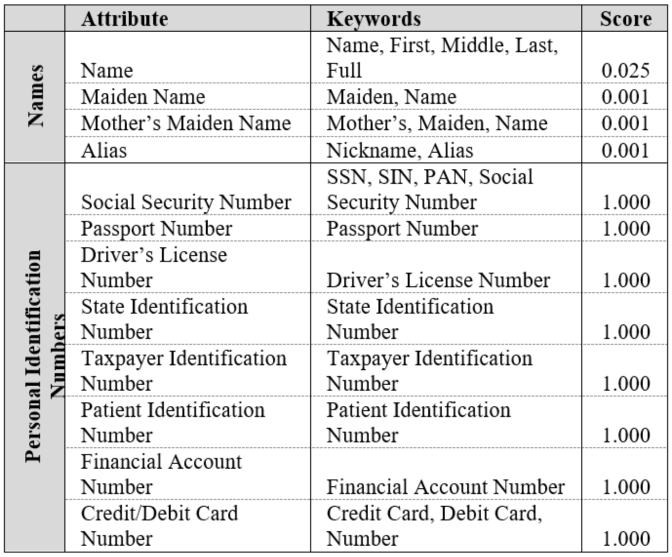


Fig. 1 Core actors and information flow in the credential verification process [2]

III. Attribute Sensitivity Score Model

Certain privacy concerns in self-sovereign identities centers around disclosure of information that can happen via proof exchanges [7]. To be able to discern between proof requests that requests sensitive data versus ordinary disclosable data by an agent, a model is proposed here that provides a novel way for agents to arrive at a decision on sharing attributes for proof requests between communicating DIDs by first assigning sensitivity scores to identity attributes and then feeding facts to an expert system to determine how sensitive the requested attributes are. If the system is unable to arrive at a decision, it can pass the obligation to the identity holder for further action.

Using the definitions of Personally Identifiable Information (PII) in [8] and extending upon the PII examples from [9], this model first groups around 86 attributes/attribute types into nine categories: 1) Names 2) Personal Identification Numbers 3) Address 4) Asset 5) Phone Numbers 6) Personal Characteristics 7) Student Related Information 8) Personally owned property and 9) Linkable Information. For each attribute/type a list of keywords are defined which can possibly appear in a JSON proof request for that particular attribute or type. A lookup table consisting of name-value pairs is then constructed to manually assign scores on a scale of 0.0-1.0 to each attribute depending on the information sensitivity it actually carries individually. A score of 1.0 indicates highly sensitive data. A Boolean value for the score is not suitable at this point of time since some attributes may not be sensitive on their own but may become hence after if combined with other attributes with varying a degree. A proof request for attributes in JSON is processed using a Python program along with this lookup table and a rules engine to provide a cumulative score to a request based on a combination one or more attributes. The program will flag the agent on the sensitivity using the total score. A snapshot of the categorization performed to create the lookup table is shown in Fig. 2.



	Attribute	Keywords	Score
Names	Name	Name, First, Middle, Last, Full	0.025
	Maiden Name	Maiden, Name	0.001
	Mother's Maiden Name	Mother's, Maiden, Name	0.001
	Alias	Nickname, Alias	0.001
Personal Identification Numbers	Social Security Number	SSN, SIN, PAN, Social Security Number	1.000
	Passport Number	Passport Number	1.000
	Driver's License Number	Driver's License Number	1.000
	State Identification Number	State Identification Number	1.000
	Taxpayer Identification Number	Taxpayer Identification Number	1.000
	Patient Identification Number	Patient Identification Number	1.000
	Financial Account Number	Financial Account Number	1.000
	Credit/Debit Card Number	Credit Card, Debit Card, Number	1.000

Fig 2. Partial snapshot of the categorization of the attributes with manually assigned scores

The cumulative score calculation process by this model follows a rules-based approach [10]. This approach, as well as the manual score assignment process, relies on certain pre-defined rules representing a generally accepted standard or an expert estimation on the sensitivity of the data in question. This model relies on intentional rules corresponding to conditions related to attribute names and keywords. An example pseudo-code assigning a score to student related combination of attribute keywords can have the following expression to calculate it:

```
if attributeName contains 'Student' ∨ 'ID' ∨
'Number' ∨ 'Truancy' ∨ 'Suspension' ∨ 'Migrant'
∨ 'Homeless' ∨ 'Status' then total_score = sum
(score (student id number) +score (habitual
truancy) + score (suspension)+ score (migrant
status) + score (homeless status))
```

The cumulative total score is then normalized to get a value between 0.0 and 1.0, with 1.0 being the most sensitive data to share. The combination score is decided by feeding the attributes into a rules engine created using PyKnow which is a Python library for creating expert systems [11]. The schematic of this proposed model of expert system is shown in Fig. 3.

The Inference Engine comprises of pre-defined sets of rules that leverages the PyKnow library to process single attributes as well as a combination of attributes to make an inference on the sensitivity of the data in a proof request. Attributes requested from an agent are captured and populated as "facts" using the interface program and fed to the engine. The engine arrives at a decision using its knowledge base and rule base. Once the system decides the sensitivity of the data and passes the same to the Hyperledger agent, the agent can make an informed decision, with additional programming logic, on whether to proceed with the proof presentation for the given proof request or not. In cases where the rules engine is unable to arrive at a decision, the decision is passed onto the agent to prompt for user action.

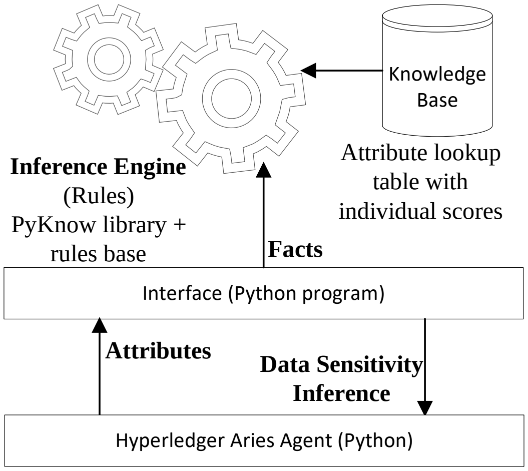
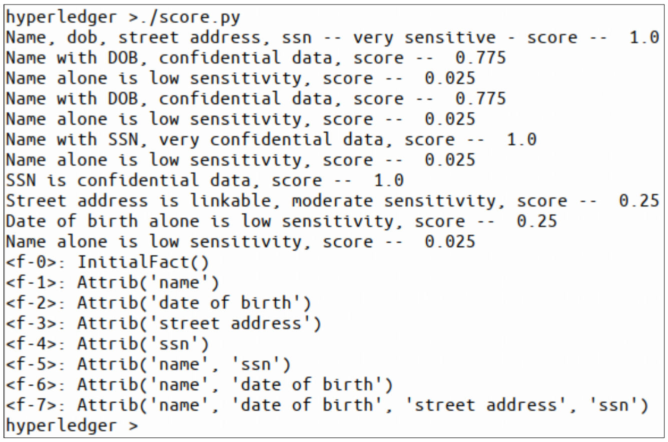

Knowledge Base
Attribute lookup table with individual scores
Inference Engine
(Rules)
PyKnow library + rules base
Facts
Interface (Python program)
Attributes
Data Sensitivity Inference
Hyperledger Aries Agent (Python)

Fig. 3 Schematic of the proposed expert system interfaced with a Hyperledger Agent

Attributes are assessed by the engine individually or in combination with other attributes as the case may be when they are input to the engine. A sample output of the engine for individual and combinational assessment of the four attributes ‘name’, ‘date of birth’, ‘street address’ and ‘ssn’ is given in Fig 4. It is to be noted that name or date of birth on its own has little value, but when combined together the information becomes sensitive. A ‘street address’ has some linkable value while an ‘ssn’ (Social Security Number) is confidential information by itself. A cumulative total score is then calculated out of the combination which is provided to the agent for it to arrive at a decision whether to proceed with the data exchange or not.



```
hyperledger >./score.py
Name, dob, street address, ssn -- very sensitive - score --  1.0
Name with DOB, confidential data, score --  0.775
Name alone is low sensitivity, score --  0.025
Name with DOB, confidential data, score --  0.775
Name alone is low sensitivity, score --  0.025
Name with SSN, very confidential data, score --  1.0
Name alone is low sensitivity, score --  0.025
SSN is confidential data, score --  1.0
Street address is linkable, moderate sensitivity, score --  0.25
Date of birth alone is low sensitivity, score --  0.25
Name alone is low sensitivity, score --  0.025
<f-0>: InitialFact()
<f-1>: Attrib('name')
<f-2>: Attrib('date of birth')
<f-3>: Attrib('street address')
<f-4>: Attrib('ssn')
<f-5>: Attrib('name', 'ssn')
<f-6>: Attrib('name', 'date of birth')
<f-7>: Attrib('name', 'date of birth', 'street address', 'ssn')
hyperledger >
```

Fig.4. Inference engine at work after populating with 7 facts. The rules fire whenever there is a match, even if partial. The facts (f-1 to f-7) show what attributes were fed to the engine

The next section describes the trust of first use vulnerability of interacting self-sovereign identities on the Hyperledger Indy platform and a method of mitigating this vulnerability.

IV. A Method of Mitigating Man-in-the-Middle Attacks on Communicating Peer DIDs

Unless peers are certain about the authenticity of their peer DID connection, it is essential to verify either parties once a new connection is established. A recommended way of doing this is using verifiable credentials [5]. However, if the attacker is able to proxy both a request and its response (the proof presentation) between parties, then this mechanism fails. So, an extra proof is needed which can confirm that the entity which constructed the presentation also delivered it. To mitigate the risk, claims are signed using the signing key of the key-pair with which the peer-DID relationship is formed, so that every claim can be verified by the receiver to have come from the actual sender. A signature mismatch allows the receiver to detect that the claim did not originate from the actual sender and can terminate the peer connection to prevent unauthorized disclosure of data. The sequence diagram of such an arrangement is shown in Fig. 5.

In an experimental setup comprising of a 4-node Hyperledger Indy network, two Hyperledger Aries Cloud agents (Python coded) were setup to communicate with each other to exchange credential attributes. The sender was setup to sign the attributes using the Sodium cryptographic library [12] using Python wrapper functions [13]. The credential attributes in JSON format are converted to a string and then signed using the signing key of the sender. This signing key corresponded to the same public key that was used to form the peer connection with the receiver agent (Fig. 5). The signature is then extracted and converted to base58 format and attached as a key value pair in the credential exchange payload. The receiver would use the public key of the sender’s peer DID connection to verify the signature.

The addition of the signing function causes a marginal overhead as observed in the performance tests conducted thereafter. In progressive iterations of credential exchanges, the average time taken per credential exchange with 4 signed

	A	B
My Peer DID	V8qL4qYGi7BVGr6GAKwDT1	FA86evVKWRqYNhjF53fqM3
Their Peer DID	FA86evVKWRqYNhjF53fqM3	V8qL4qYGi7BVGr6GAKwDT1
Public DID	5Va2M6eyKsNeYftihcAsyD	QHXRWWNbEVL59yAcSoVikq

P_{kA}, S_{kA} = key-pair associated with Peer DID A (V8qL4qYGi7BV…)
P_{kB}, S_{kB} = key-pair associated with Peer DID B (FA86evVKWRqY…)

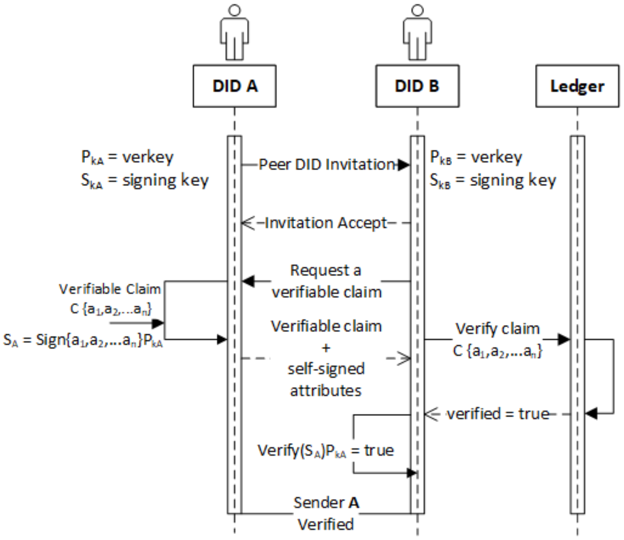


Fig. 5 Sequence diagram showing the signing of attributes using Peer DID keys by one party as a way to authenticate that the message was generated and delivered by this very party

attributes had a median value of 0.78 seconds compared to a median value of 0.725 seconds for attributes exchanged without signing. Table I gives a summary of the test results. The next section proposes a quantitative reputation and confidence level measurement model for credential issuers.

TABLE I. VARIATION OF TIME TAKEN PER CREDENTIAL EXCHANGE WITH AND WITHOUT SIGNED ATTRIBUTES

No. of credentials exchanged	**Avg. Time/credential (sec)**		**Throughput (credentials/sec)**	
	without signing	**with signing**	**without signing**	**with signing**
50	0.64	0.69	1.56	1.44
100	0.79	0.84	1.26	1.19
150	0.73	0.76	1.36	1.32
200	0.75	0.86	1.34	1.16
250	0.72	0.78	1.38	1.28
300	0.73	0.78	1.37	1.28
350	0.78	0.81	1.28	1.23
400	0.72	0.78	1.39	1.29
450	0.72	0.86	1.38	1.17
500	0.65	0.72	1.54	1.38
Median	0.725s	0.78s	1.375/s	1.28/s

V. A QUANTITATIVE MODEL FOR REPUTATION AND CONFIDENCE LEVEL MEASUREMENT FOR IDENTITY ISSUERS

This section proposes a novel way of quantitative confidence level measurement for issuers. A self-sovereign identity system like Hyperledger Indy which leverages verifiable credentials, has three primary actors - the issuer, the verifier and the identity holder [2]. As discussed in section VI, exchanging verifiable credentials during the establishment of a peer DID connection between two unknown peers is essential to confirm the authenticity of the peers. The peers can have a certain confidence in the peer relationship if it can be determined 1) what the reputation of the issuer who issued the exchanged credentials is and 2) the trust score of each peer identity holder. Together they allow us to determine a quantitative confidence level in the relationship from the perspective of each peer. A higher confidence level will allow the peers to freely do future transactions without security or privacy concerns. A lower confidence will enable the peers to exchange information cautiously, or even break the peer relationship.

In [14], Gruner et. al. proposes a quantifiable trust model for blockchain identities that uses directed graphs to determine the trust flow between nodes and calculates quantitative trust values for claims by the number of attestations it has received. It further derives trust values for identities from trust scores of claims issued to identities.

Assuming the peer identities have a certain initial trust score, the quantitative trust score for each identity, based on the trust of its claims, is given by T_i, where $0 \leq T_i \leq 1$ and maximum trust is represented by a value of 1 [14].

To calculate a quantitative value of the reputation of the issuer issuing the credential, some of the requirements were put forth in the reputation framework described in [15]. However not all of these can be used here directly. The below requirements are proposed in this model based on their relevance and applicability to self-sovereign identities use case. These are:

1) An identity newly onboarded as an issuer is assigned an initial reputation score R_0

Issuers are onboarded by the trust and governance framework of the blockchain network as trust anchors after due-diligence and signing of the trust agreement as part of the onboarding process. Networks like Sovrin, which is based on Hyperledger Indy, has a Governance Framework for this purpose [16]. The assignment of the initial score R_0 may be based on various aspects of the credential issuing organization such as 1) Products 2) Services 3) Governance model 4) Certifications 5) Innovation 6) Performance 7) Financial status/market capital 8) Endorsements 9) Compliance etc.

2) The calculated value of reputation shall lie within the closed interval of [1,100]

The reputation value of an issuer R_t at time t shall have a lower and upper bound of 1 and 100 respectively, with a value of 100 representing maximum reputation. This is unlike the framework in [15] which has values bounded between 0 and 1. This is because the ultimate aim is to calculate a value for the confidence level which is a product of T_i [14] and R_t and since the confidence in the identity is directly proportional to the trust in that identity, R_t must be scaled to a factor greater than 1 to reflect a proportional increase in confidence value. It can, however, it can have fractional values within this range.

3) The reputation value of an issuer must be configurable

The initial reputation score R_0 is assigned to an issuer needs to be configurable to a lower or higher value depending on the changes in the status-quo of the issuer during the course of time. The range of this value is [1,100]. The Stewards of the network alone must be permitted to do this. Factors both external and internal to the ledger must be considered in making changes to this score. The external factors shall, but not be limited to the ones described in 2) above and must be done at periodic intervals determined by the governance framework. The proposed internal factor to be considered in this model is the number of credentials issued by that particular issuer in a given interval of time.

A random dataset was generated for the number of credentials issued per day (limited between 0 and 10,000) for a period of 100 days. Its moving average was calculated over a 30-day window. This, however, places equal weightage to all observations, which is not appropriate for the needs here.

Calculation approaches for such measurements in [15] and [18] center around the weighted average method and the exponential weighted average methods respectively. The goal of this model is to assign a reputation score based on the existing reputation ascribed to the issuer and a current reputation factor based on the number of credentials issued in a window period so as to depict rising or falling trend in reputation. Exponential Weighted Moving Average (EWMA) as applied in [15] is more appropriate in this case as well since the data can be highly fluctuating for the given criterion (credentials issued). Hence, it is pertinent to put more weightage to more recent samples than distant samples in the past. One reason for this can be, for example, an issuer who had been very active in the past but has had little activity recently may be on the path of decline since no identity holder is requesting credentials from this issuer recently. In such as case its reputation factor must be weighed against his most recent activity window using EWMA.

The EWMA reputation factor calculated on day ***t*** can be formulated from [17] as:

$$\text{EWMA}_t = \lambda Y_t + (1-\lambda)\,\text{EWMA}_{t-1} \qquad \text{for } t=1,2\ldots n\ldots\ldots(1)$$

where

- Y_t = sample taken at time t
- λ = weighting factor
- n = Total samples taken, including EWMA_0
- EWMA_0 = mean of historical data

The weighting factor λ is selected such that a value closer to 1.0 gives more weightage to recent samples [17]. The formula in (1) is used to calculate the average number of credentials issued on a particular day using a window of *n* prior days from that day. A simple average of the result gives the average number of credentials issued in that period and is used as a factor to calculate the reputation. The starting value of EMWA_0 is determined from a simple mean of historical data or an equivalent assumption made. Practically, a reputation score can be calculated after a certain period after initial onboarding.

In [18], a reference is made of trust and reputation systems like EigenTrust and PeerTrust and how the final value of trust or reputation is computed using a weighted average of all factors combined.

Since the final reputation value in question here is bounded between 1 and 100 and is also affected by the initial reputation score assigned to the issuer at onboarding, the reputation of an issuer at time *t* can be defined by:

$$R_t = R_0 + \alpha * F_t \qquad \ldots (2)$$

where,

- R_t is the reputation at time t
- R_0 is the initial reputation assigned to the issuer
- F_t is the specific factor considered here which is equal to the exponential weighted average of number of credentials issued at time t
- α is the weighting factor

Using min-max normalization [19], the normalized value of F_t in the range [a, b] is given by:

$$F_t = C_{t\,normalized} = (\text{b - a})\,\frac{C_t - \min(C)}{\max(C) - \min(C)} + \text{a} \quad \ldots(3)$$

where,

- C_t is the exponential moving average of the number of credentials issued in time t
- C is a vector representing all the values of C_t in the given window.
- Here range [a, b] is taken as [-1, 1], since it gives appropriate positive and negative values for high (+) and low (-) figures to finally depict rise/fall

So, from (3), equation (2) can be re-written as

$$R_t = R_0 + \alpha * C_{t\,normalized} \qquad \ldots (4)$$

Now, the value of the weighting factor α is selected such that it appropriately represents one of the group of factors in measuring the reputation as mentioned in requirement (1) above. If an equal weightage is placed on the 9 external factors in requirement (1), the addition of the "number of credentials issued" as a factor will be 10th and hence each will have a weight of 0.1. Under this assumption, α has a value of 0.1. So, equation (2) can now be written as:

$$R_t = R_0 + 0.1 * C_{t\,normalized} \qquad \ldots (5)$$

So, the confidence level in the issuer *i* at time *t* can now be computed by the below product:

$$CL_{i,t} = T_i \,.\, R_t \qquad \ldots (6)$$

where,

T_i = Trust score of the issuer, as discussed earlier [14]
R_t = Reputation of the issuer as per (5)

Using the same dataset and a 30- day window with an initial trust of 50% (or T_i = 50), initial reputation of R_0 = 50, λ = 0.8 and EWMA_0 = 4000 (assumed historical data), Table II is

TABLE II. COMPUTED SCORES OF REPUTATION AND CONFIDENCE LEVEL ON AN ISSUER IN A WINDOW OF 30 DAYS

Day	**Y_t**	**EWMA_t**	**C_t**	**R_t**	**$CL_{i,t}$**
0		4000			
1	5056	4844.80	0.020032	50.0020032	25.0010016
2	5341	5241.76	0.12046	50.012046	25.006023
3	3334	3715.55	-0.26566	49.973434	24.986717
4	9969	8718.31	1	50.1	25.05
5	4708	5510.06	0.188338	50.0188338	25.0094169
6	5741	5694.81	0.235078	50.0235078	25.0117539
7	7136	6847.76	0.526765	50.0526765	25.02633825
8	1244	2364.75	-0.607402	49.9392598	24.9696299
9	1752	1874.55	-0.731418	49.9268582	24.9634291
10	9391	7887.71	0.789865	50.0789865	25.03949325
11	1277	2599.14	-0.548103	49.9451897	24.97259485
12	2116	2212.63	-0.645887	49.9354113	24.96770565
13	1641	1755.33	-0.76158	49.923842	24.961921
14	2407	2276.67	-0.629685	49.9370315	24.96851575
15	447	812.93	-1	49.9	24.95
16	4246	3559.39	-0.305167	49.9694833	24.98474165
17	6924	6251.08	0.37581	50.037581	25.0187905
18	5351	5531.02	0.19364	50.019364	25.009682
19	6107	5991.80	0.310214	50.0310214	25.0155107
20	218	1372.76	-0.858367	49.9141633	24.95708165
21	3142	2788.15	-0.500285	49.9499715	24.97498575
22	5429	4900.83	0.034207	50.0034207	25.00171035
23	3631	3884.97	-0.222798	49.9777202	24.9888601
24	4872	4674.59	-0.02303	49.997697	24.9988485
25	4293	4369.32	-0.100261	49.9899739	24.99498695
26	1336	1942.66	-0.714187	49.9285813	24.96429065
27	3797	3426.13	-0.338881	49.9661119	24.98305595
28	9817	8538.83	0.954593	50.0954593	25.04772965
29	1177	2649.37	-0.535395	49.9464605	24.97323025
30	2858	2816.27	-0.49317	49.950683	24.9753415

generated which shows the reputation score and confidence level on an issuer at a given time *t* during a window period of 30 days.

Table II shows that even though the issuer issues a very high number of credentials on a particular day (row 10), its reputation and confidence level score increase only by a small fraction. Similarly, on a very low day (row 20), the scores decrease only marginally. So, the method computes a score that doesn't skew the initial score much. Rather, it changes on a daily basis in the measured window but places more weightage on recent transactions than the past ones. Also, for significant variations in transactions within the window, the reputation increases or decreases only marginally which reduces unnecessary fluctuations and guarantees a certain amount of stability of the score based on its weighting factor. So, this model is able to provide a positive or negative trend in a particular criterion, in this case, credentials issuance to calculate a quantitative confidence level in that issuer for peer agents to decide whether to proceed with a transaction.

VI. Analysis and Discussion

a) Proposed Attribute Sensitivity Score Model

The process of controlling and selectively disclosing data rests on the identity holder willing to share the data that is being requested. Two aspects where recommendations can be made here are: (1) technological controls to prevent leakage of data and (2) trust between the two communicating parties. The layer that performs the interaction between two parties is the agent layer. Agents can be configured to auto-accept requests for information, which is risky from privacy perspectives. So, agents must not automatically accept an invitation or proof request. A proof request must prompt the identity holder necessary warnings on the risks of sharing that data along with the fundamental question whether the identity holder is willing to trust the requester. A technical solution that helps agents make an instinctive decision is by utilizing the proposed attribute sensitivity score model.

The proposed model of assigning a sensitivity score to attributes to make a decision before exchanging credentials or any other data is based on a knowledge base and a rules engine. It covers the most common PII and the corresponding rules in the model mostly handle the most common combinations but not all. Also, certain permutations and combinations of attributes can be challenging to evaluate and hence the inference engine may have to fall back on the human user for further action.

b) Method of mitigating man-in-the-middle attacks

The man-in-the-middle attack scenario described here is hard and unlikely to be carried out practically. Even then, an appropriate way of detecting and mitigation it is by self-signing claim attributes with keys derived from the peer DID relationship before exchanging credentials. The slight performance overhead can be minimized in production, since the test utilized demo agents coded in Python. The addition of self-signed claims is a possibility in the future roadmap of Indy [5], however the likelihood of the risk will give it priority. Nevertheless, it is an important inclusion since the platform also targets security by design.

c) Proposed Quantitative Model of Issuer Reputation and Confidence Level

Building trusts between two unknown private parties can be challenging so is essential to build the trust in stages. The proposed model is a novel way of computing a quantitative value of reputation for an issuer considering an initial reputation score and adds up to it from the reputation measured from the credentials issuance activity. The initial endorsement can be performed during the onboarding of the issuer by the governance framework the network is mapped to. The score is useful to calculate the overall reputation and confidence level at a point in time. The quantitative values show a rising or falling trend akin to stock prices and their volatility. And this quantitative trend affects only a fraction of the initial reputation, a score is modifiable only by a steward and hence requires human judgment at the onset. The selection of a value for λ is dependent on how recent or far into the past samples the model should base its calculations on. A value of λ closer to 1 indicates more recent samples. The instantaneous values do not affect the existing reputation/confidence level, nonetheless, it is useful for trend analysis and its implementation on a blockchain is feasible provided appropriate methods are implemented to capture the statistics. The proposed model, however, targets just one aspect of identity interaction which is issuance of credentials. As discussed in section V, the reputation of an issuer can be attributed to a number of other factors - both internal and external to the ledger. Other internal factors that can be modeled for similar measurements can be number of unique credentials issued, credential acceptance rate etc.

VII. Conclusion

This study contributed towards the understanding of certain aspects of security and privacy of self-sovereign identities on Hyperledger Indy and by proposing technical enhancements for the same. It examined disclosure of data during negotiation of proofs and whether there is a possibility of sensitive data leakage by disclosing attributes which are normally meant to be private. It also analyzed the risk of a man-in-the-middle attack that might possibly takeover a peer DID connection during initial setup. The paper's main contributions are the proposition of a novel attribute sensitivity scoring model that enables agents to ascertain if attributes within credentials are risky to be shared with regards to personal data within it. With regards to man-in-the middle attacks between peer DIDs, the study proposed a method to self-sign attributes using the private key of the peer DID of the sender so that it can be guaranteed that the party which generated the message as the one who delivered it. Lastly, the study also proposed a novel quantitative model for computing reputation scores for issuers which when combined with the trust score of the identity allows peer DIDs to get a quantitative confidence level value for an issuer which helps dispel security and privacy concerns when communicating with an unknown peer who has presented verifiable credentials issued by that issuer. Future research can be in areas of refining upon the proposed models and on developing best practices that focusses on building trust between DIDs, how much and what minimum data needs to be shared between parties for completion of a task and on prevention of aggregation of private data not just by attackers but even by legitimate parties.

REFERENCES

[1] Christopher Allen, “The Path to Self-Sovereign Identity”, Apr 2016. [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html. [Accessed: 3 Apr 2020]

[2] Manu Sporny, Dave Longley and David Chadwick “Verifiable Credentials Data Model 1.0”, 19 Nov 2019. [Online]. Available: https://www.w3.org/TR/verifiable-claims-data-model/. [Accessed: 3 Apr 2020]

[3] Mike Lodder and Brent Zundel, “Anonymous Credential Protocol”, Jan 2019. [Online]. Available: https://github.com/hyperledger/indy-hipe/tree/master/text/0109-anoncreds-protocol. [Accessed: 3 Apr 2020]

[4] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen, Ryan Grant and Markus Sabadello “Decentralized Identifiers (DIDs) v1.0”, 3 Apr 2020. [Online]. Available: https://w3c-ccg.github.io/did-spec/. [Accessed: 3 Apr 2020]

[5] Oskar Deventer, Christian Lundkvist, Márton Csernai, Kyle Den Hartog, Markus Sabadello, Sam Curren et.al. “Peer DID Method Specification”, 4 Jan 2020. [Online]. Available: https://openssi.github.io/peer-did-method-spec/index.html. [Accessed: 3 Apr 2020]

[6] Paul Dunphy and Fabien A.P. Petitcolas, “A First Look at Identity Management Schemes on the Blockchain”, 2018. IEEE Security & Privacy, 16(4), pp 20 - 29

[7] “Trust of first use question”. [Forum post]. [Online]. Available: https://lists.hyperledger.org/g/indy/topic/34534377. [Accessed: 3 Apr 2020]

[8] Erika McCallister, Tim Grance and Karen Scarfone, “Guide to Protecting the Confidentiality of Personally Identifiable Information (PII)”, April 2010, NIST Special Publication 800-122. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf. [Accessed: 3 Apr 2020]

[9] Wisconsin Department of Public Instruction, “Personally Identifiable Information Examples”. [Online]. Available: https://dpi.wi.gov/sites/default/files/imce/wisedash/pdf/PII%20list%20of%20Examples.pdf. [Accessed: 3 Apr 2020]

[10] Cédric Mouza, Elisabeth Métais, Nadira Lammari, Jacky Akoka, Tatiana Aubonnet, Isabelle Comyn-Wattiau, Hammou Fadili and Samira Si-Saïd Cherfi, “Towards an Automatic Detection of Sensitive Information in a Database”, 2010 Second International Conference on Advances in Databases, Knowledge, and Data Applications, pp 247 – 252

[11] Roberto Abdelkader Martínez Pérez, “PyKnow: Expert Systems for Python”. [Online]. Available: https://github.com/buguroo/pyknow. [Accessed: 3 Apr 2020]

[12] Libsodium documentation. [Online]. Available: https://libsodium.gitbook.io/doc/. [Accessed: 3 Apr 2020]

[13] Levien van Zon, “Bindings and examples for using low-level libsodium functionality in Python”, Dec 2017. [Online]. Available: https://github.com/lvzon/libsodium-python-examples/. [Accessed: 3 Apr 2020]

[14] Andreas Grüner, Alexander Mühle, Tatiana Gayvoronskaya and Christoph Meinel, “A Quantifiable Trust Model for Blockchain-Based Identity Management”, 2018 IEEE International Conference on Internet of Things (iThings), pp 1475 – 1482

[15] Tara Salman, Raj Jain and Lav Gupta, “A Reputation Management Framework for Knowledge-Based and Probabilistic Blockchains”, 2019 IEEE International Conference on Blockchain, pp 520 – 527

[16] Sovrin Foundation, “The Sovrin Governance Framewok”. [Online]. Available: https://sovrin.org/library/sovrin-governance-framework/. [Accessed: 3 Apr 2020]

[17] NIST, “EWMA Control Charts”, Engineering Statistics Handbook. [Online]. Available: https://www.itl.nist.gov/div898/handbook/pmc/section3/pmc324.htm. [Accessed: 3 Apr 2020]

[18] Gong Shang-Fu and Zhu Jian-Lei, “A Survey of Reputation and Trust Mechanism in Peer-to-Peer Network”, 2012 International Conference on Industrial Control and Electronics Engineering, pp 116 - 119

[19] Sebastian Raschka, “About Feature Scaling and Normalization - and the effect of standardization for machine learning algorithms”, Jul 2014. [Online]. Available: https://sebastianraschka.com/Articles/2014_about_feature_scaling.html. [Accessed: 3 Apr 2020]