



Università degli Studi di Pisa

Dipartimento di Informatica

Master Degree in Computer Science

Assessing the Trustworthiness of Social Content Through Self-Sovereign Identity

Candidate:

Calogero Turco

Supervisors:

Prof. Laura Emilia Maria Ricci

Dott. Paolo Mori

Academic Year 2023-2024

Contents

1	Introduction	4
2	Background	10
2.1	Digital Signatures	10
2.2	Blockchain	11
2.3	Bitcoin	12
2.3.1	Consensus and Proof-Of-Work	13
2.4	Ethereum and Smart Contracts	14
2.5	Self Sovereign Identity	15
2.5.1	DID	17
2.5.2	VCs the W3C Data Model, JOSE and JWT	19
2.6	Large Language Models (LLMs)	21
2.7	Transformers: GPT and BERT	22
2.8	Named Entity Recognition (NER)	24
3	Related Work	25
3.1	Propagation Characteristics	26
3.2	Widely used solutions and discussed characteristics	27
3.2.1	Study on Fake News Credibility	28
3.2.2	Fact-Checking by Mainstream Media Organizations	34
3.2.3	Sources of Fact-Checking	34
3.2.4	Propagation characteristics	35
3.3	Solutions/Ideas Similar to ours	36
3.3.1	TRUSTD	36
3.3.2	Tracing with Blockchain	36
3.3.3	W3C Credentials Community workshop: Verifiable State- ments	37
4	Veramo, IPFS, ESCO Ontology, and AI Tools	38

4.1	Veramo	38
4.1.1	Veramo DID Agent and Veramo Javascript library methods	39
4.2	IPFS	41
4.2.1	IPFS using Helia Library	41
4.3	KeyBert	42
4.4	ESCO ontology	44
4.5	Nesta's Skills Extractor Library: ojd daps skills	47
4.5.1	A critical look to the tool	49
4.5.2	Example	51
4.6	OpenAI GPT APIs	51
5	Assessing Social Content Trustability with SSI	55
5.1	Problem statement and overview	55
5.2	Key actors and proposed solution	57
5.2.1	Key Actors	58
5.3	User Stories and High-level Overview	59
5.3.1	Use case	60
5.4	Match Statement and Certificates Skills	64
5.5	Credibility Assessment and Verification System	67
5.5.1	Types of Diffusion: VP Diffusion of type "origin" and type "diffusion"	70
5.5.2	CAVS and use case	74
6	CAVS implementation	77
6.0.1	Architecture details	78
6.0.2	Trust in AI Models	79
6.0.3	Testing AI tools	80
6.1	Webapp and CAVS in detail	80
6.1.1	Certificate Issuer section	82
6.1.2	CAVS dashboard	82
6.1.3	Author section	85
6.1.4	Spreader section and VP Diffusion	88
6.1.5	VCs and QRCode	91
6.1.6	Reader section	92
7	Use Case on Real Online Social Network	96
7.1	DeSo	97
7.1.1	DeSo blockchain	97

7.1.2 DeSo Wallet and Mapping DeSo Accounts to DID . . .	98
7.2 Integration View	102
7.2.1 DeSo address to Ethereum address	103
7.2.2 Author Posting	103
7.2.3 Spreader	104
7.2.4 Reader tracing back and verifying	104
8 Experimental results and Analysis	111
8.1 Testbed Overview: Ground Truth and Skill-Based simulations	112
8.2 Dataset Generation for the Testbed	114
8.3 Ground Truth Simulation	117
8.4 Skill-Based Simulation	119
8.5 Running and Comparison	119
8.5.1 User State Evolution	119
8.5.2 Posts propagation modeled as tree	123
8.5.3 User centric statistics	127
8.5.4 Comment	128
8.6 Performance Evaluation	129
8.7 Security Analysis	134
8.8 Trust Issue mitigations with CAVS	136
8.8.1 DID:WEB and TLS/SSL Certificate Authorities	137
8.8.2 CAVS Centralized solution: known issuers list	137
8.9 Web of Trust and Alternative System: CMCS	139
8.9.1 Decentralized Web of Trust System Architecture . .	139
8.9.2 Implementation in DeSo	142
8.9.3 DID-Based Alternative for Post-Level Trust	142
9 Conclusion and future works	145
9.1 Future Works	147
Bibliography	148

Chapter 1

Introduction

Assessing the trustworthiness of information encountered in distributed environments has become increasingly difficult in today's society, particularly on Online Social Networks (OSNs) [1]. Social Media platforms enable anyone to generate and spread Social Content (e.g., articles, news, comments, and images). Furthermore, due to the vast amount of content people share daily, assessing the credibility of each of them is becoming increasingly complex [2]. Across various fields, from politics to science, false information is frequently presented as truth, causing widespread issues and making it challenging to discern fact from fiction. For instance, a study suggests that people have acted publicly as if a fake news story was true even if they individually recognize that said story is fake [3] based on the belief that others will recognize it as true. Given this, we suggest a privacy-preserving solution that allows people to judge fake news when reading it and stop it from spreading to a great public.

OSNs often allow their users to create pseudo-anonymous identities, making it difficult for readers to trust the published content since they don't know the real authors of the posts. For instance, many fake news authors are non-human bots that disguise them-self as humans behind fictional profiles. Furthermore, by hiding their identity, malicious people are more likely to spread misinformation on online platforms, given that anonymity makes them feel more protected from consequences. Ultimately, readers need the author to disclose their identity fully to obtain clear evidence of the author's expertise on the topic to be sure of

the trustworthiness of its content; otherwise, readers need to research the author's qualifications or mindlessly decide to trust the author's words. Unfortunately, even traditional methods of verifying the identity of an OSN account, such as blue check marks, may fail, having authors disguised as someone else or a company and spreading false information behind their identity [4]. It would benefit readers to have proof of authoritativeness from authors about the topic they talk about. Unfortunately, authors may wish to maintain their privacy without exposing traditional credentials.

We can see the issue of privacy, misinformation costs, risk reduction, and ease of information sharing as a trilemma that we propose, illustrated in Figure 1.1 and detailed as:

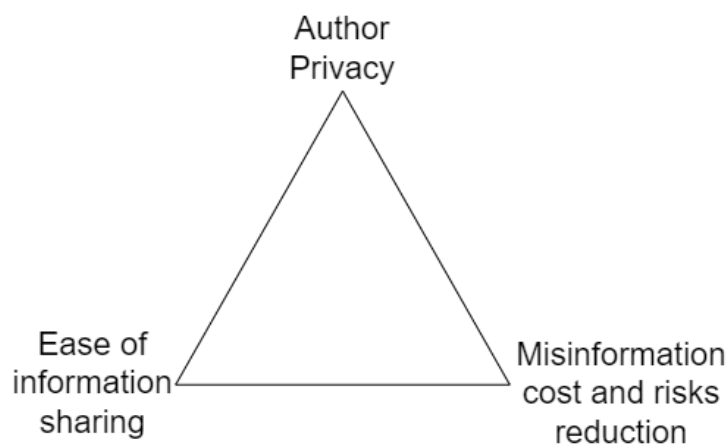


Figure 1.1: +
The triangle of OSN information sharing and the truthfulness of Social Content

Author Privacy ensures that individuals can share information, leaks, rumors, and controversial opinions without facing the consequences and risks. The trade-off is that it reduces accountability for false information: tracking, stopping, or punishing misinformation sources becomes complicated.

Ease of Information Sharing enables the rapid and easy spread of contents. The trade-off is that it often conflicts with efforts to control misinformation and privacy concerns. First, it allows both true and

false information to spread quickly, given that no effort is needed for an author to prove their expertise when they share posts anonymously. Second, if privacy is not their concern, authors may prove their expertise by sharing their real name instead of a pseudonym.

Misinformation Cost and Risk Reduction requires strict verification and monitoring systems, which may infringe on privacy and hinder the ease of information sharing. The trade-off is balancing the freedom of communication with the need to reduce the risks associated with misinformation.

Any solution that proposes to diminish misinformation risk on an OSN by obtaining an assessment of post-author competencies faces a very challenging task. The one of **not infringing the privacy of the authors**, by requiring them to provide extensive information about their identities.

Besides, various solutions exist in the literature to mitigate the spread of false information, as the more widespread solutions are AI-based or with human fact-checkers. However, these approaches have limitations, which we will discuss in this thesis. For example, they may be too slow to halt the rapid **spread of misinformation** before it **causes harm**, such as damaging a business's reputation.[5]

A solution involving the assessment of competencies for authors before allowing them to publish information may be possible, or to provide contextually said information alongside the post for readers. Nevertheless, it would require authors to disclose their identities to offer extensive information on their background, allowing readers or authorities, such as OSN moderators, to verify the credibility of the information they spread. These moderators could block the authors from posting false or harmful content. This approach mirrors traditional journalism, where information is controlled and verified by centralized sources and authorities. However, in the OSN scenario, this approach would restrict the **easily accessible, easily sharable**, and decentralized nature of the information that OSN allows.

Moreover, providing personal information alongside the social content diffusion would significantly limit users' **freedom of expression** in OSNs[6], as readers disagreeing with the opinions shared by an author may target them, for instance, to censor the information shared by ha-

passing them both on multiple online platforms, in their workplace, or even at their house [7]. For example, there were instances of persons fired from their jobs because of an opinion they shared by their name.

*This thesis proposes **Credibility Assessment and Verification System (CAVS)**, a solution we built aiming to preserve the privacy of OSN users, still allowing them to prove their authoritativeness in the topic while publishing their statements. CAVS's design aims to mitigate the issue of malicious or naive spreaders of false information online and to find a balance between Author Privacy and Misinformation risk reduction with minimal reduction to the ease of information sharing.*

This approach would support sharing correct but unpopular opinions, fostering healthy online discourse with competencies that give weight to statements while not leading the privacy of those who make them.

CAVS leverages **Self-Sovereign Identity (SSI)** [8] technology and can be a viable privacy-preserving mitigation strategy for the fake news problem. SSI technology enables the generation of digital certificates and the controlled sharing of embedded information by the individuals they refer to. The certificate themselves are bound to a Decentralized Identifier bound to the individuals, which can cryptographically prove to hold them. In particular, SSI is essentially a novel solution to the issue, satisfying the author's privacy online [9] while providing the information required to reduce misinformation spread.

In CAVS, authors who post a statement online can provide an SSI credential showing proof of their expertise. CAVS acts as an *enhanced verifier* by i) checking the validity of the credential and ii) verifying that the competence embedded in the credential is relevant to the statement's content. Artificial Intelligence (AI) services match the credential-embedded competence with the statement's content. For this, we employed the ESCO ontology to categorize and match skills and competencies.

Authors obtain from CAVS an attestation of the truthfulness of the content they created without disclosing their whole identity, such as name, surname, and affiliations, thus preserving their privacy.

We suggest that this solution would not significantly hinder the ease of information sharing. The only additional step that authors must take before spreading their social content is obtaining an attestation from the CAVS system regarding its trustworthiness. This additional step and the presentation of this attestation significantly reduce misinformation

risks, as readers will find a trustworthiness assessment produced by CAVS alongside each statement they read and may place trust on it[10]. Using VC to provide proof of authoritativeness ultimately protects the authors' privacy.

This thesis describes Self-Sovereign Identity (SSI) and machine learning models that we found fundamental as building blocks for CAVS, introducing Veramo that implements SSI and the AI tools KeyBert and Nesta Skills Extractor. Subsequently, we present the CAVS system and its integration into a real OSN, DeSo. In particular, the DeSo integration was conceived as a proof of concept to showcase CAVS on a real OSN. The thesis concludes with an experimental evaluation of CAVS as an assessment system capable of reducing the spread of fake news. Using OpenAI GPT, we generated a synthetic dataset of OSN profiles with posts to perform these simulations. Our evaluation showed that simulating 100 users, with half acting as fact-checkers using CAVS, reduced the number of posts containing fake news across 1000 epochs from around 169,000 (with no fact-checkers) to just under 9,200, slightly worse than when fact-checkers act as oracles (know for sure which news are fake and stop them), where only 8,000 posts contained false news. CAVS has shown its effectiveness while having only an overhead of 30 seconds in the post-publication process to produce its assessment and provide it to the author. The chapters follow this roadmap:

- **Chapter 2 - Background:** Introduce the Self-Sovereign Identity paradigm and the AI models used in this thesis.
- **Chapter 3 - Related Work:** In this chapter, we provide a set of definitions for the issue of fake news and discuss some techniques proposed in the literature that try to solve said issue
- **Chapter 4 - Veramo, IPFS, ESCO ontology, and AI tools:** This chapter contains an introduction to the ontology used for the skills and competencies of Social Content authors, to the AI tools for keyword and skills extraction from text and for Veramo to program SSI-based agents.
- **Chapter 5 - Assessing Social Content Trust with SSI :** In this chapter, we describe the problem of assessing the trustworthiness of Social Content. Furthermore, we describe the actors involved in its diffusion and consumption. Ultimately, we present CAVS, the

proposed system, to tackle their issues with its architecture and workflow.

- **Chapter 6 - CAVS implementation:** This chapter illustrates the CAVS system implementation as a microservice-based system that leverages dockers and exploits the tools presented before.
- **Chapter 7 - Use case on real Online Social Network** This chapter illustrates the integration of CAVS in the DeSo OSN.
- **Chapter 8 - Experimental results and Analysis:** In this chapter, we propose simulations following a SIR model of an OSN with fact checkers knowing the ground truth and comparing it with fact checkers using CAVS. Furthermore, we (will) provide a performance evaluation (time) of the tools exploited. Additionally, the chapter includes a security analysis with threats and mitigations. Moreover, we analyze an alternative mapping of the SSI concepts to solve the problem of trustworthiness, which is different from the CAVS system and propose some attack scenarios and mitigations.
- **Chapter 9 - Conclusions:** The chapter provides the conclusions of the thesis. It summarizes the takeaways from exploiting SSI to mitigate fake news in this thesis and discusses whether SSI with AI tools offers a valid solution to this societal issue. Ultimately, it gives some ideas for future improvements to CAVS, the fake news mitigation system.

Chapter 2

Background

This chapter introduces the technologies, frameworks, and AI models this thesis includes. We introduce the concept of Self-Sovereign Identity to manage user certifications with Verifiable Credentials and Decentralized Identifiers. We start the chapter by introducing cryptographic tools for exchanging authenticated messages and blockchain technology. Furthermore, we delve into AI models, introducing Large Language Models such as the notorious GPTs and BERT.

2.1 Digital Signatures

We distinguish cryptography between symmetric key cryptography and asymmetric key cryptography. The first allows sending a message through an unsafe channel in a protected way, as it is encrypted using a shared secret, a key, which the sender and the recipient exchange in a secure channel.

Asymmetric cryptography leverages, for each entity involved in a message exchange, a pair of two keys:

- Private key: used for data signature data or decryption of previously encrypted data with the pairs' public key.
- Public key: used to encrypt data or to verify signatures performed with the paired private key.

The private key will never be disclosed or exchanged. In contrast, the

public key can be available in a repository or exchanged between the sender and recipient of messages, even in an unsafe channel.

Digital signatures leverage asymmetric key cryptography. For instance, suppose that Bob wants to sign a digital certificate that is available in a repository. Bob uses their private key to sign a fixed length hash of it, called digest, which allows them to perform the signature on arbitrary length data with a time complexity upper bounded by the hash length[11, p.443]. Afterward, Alice can retrieve Bob's public key from a public repository and may prove that Bob signed it. Since the signature is created over one specific hash based on the digital certificate, any modification to the certificate would invalidate the signature. Ultimately, this ensures that a Man-in-the-Middle cannot forge the signature since it comes from Bob's private key and cannot be provided with another certificate by Alice [12].

2.2 Blockchain

Blockchain technology acts as a distributed ledger, representing an evolution of traditional bookkeeping for financial transactions like account ledgers. In practical terms, the blockchain is a data structure that functions as a list of blocks [13, p.195]. Each block points to the previous block in the chain and contains a list of records called transactions. The blockchain is a collaborative database distributed among interconnected Peer-to-peer (P2P) network nodes. It is populated with transactions sent by the peers themselves, and each sender must sign each transaction before sending it to the ledger. All other peers must validate every transaction before being stored in the blockchain. There is an election process to ensure that each peer stores an identical copy of the blockchain, i.e., the same blocks with the transactions having the same order for each peer. The peer proposing the new block is elected through a "consensus algorithm." We discuss the inner workings of blockchains through Bitcoin and Ethereum, which are the most popular ones.

2.3 Bitcoin

Bitcoin is currently the most popular blockchain in the world. The name Bitcoin refers to both the blockchain and its associated cryptocurrency, the bitcoin.

Historically, the first bitcoin block was mined on January 3, 2009. Within this block is a message that reads: *The Times 03/Jan/2009 Chancellor on brink of second bailout for banks*. Said message seems to signify a mission statement for the Bitcoin Blockchain: Bitcoin is an asset to protect wealth against monetary inflation [14]. We cannot discard the importance of Bitcoin and, in general, blockchains as cryptocurrency enablers. However, that is not the only possibility that they enable. In particular, they act as a trust anchor. Furthermore, blockchains are a distributed ledger; anyone who can read them knows their content is immutable and untameable. In particular, this thesis will focus on Blockchains as technology enablers for Self-Sovereign Identity (SSI), which we introduce in section 2.5.

Bitcoin is also a peer-to-peer (P2P) system, as the Bitcoin blockchain is stored by multiple peers, who keep an up-to-date copy of it. Data is stored and added through transactions; the network peers validate the transactions through consensus rules and store the validated transactions on the blockchain, acting as a ledger. Given that it is a P2P system, there needs to be a global consensus among peers about what transactions to insert in the distributed ledger; this consensus is reached with the Proof-of-Work algorithm. The distributed ledger is called blockchain because it is a chain of blocks, with each block having a list of transactions that exchange bitcoins [13].

The Bitcoin network has different types of nodes, each serving distinct functions. Full nodes maintain a complete copy of the blockchain, validating transactions and blocks according to Bitcoin's consensus rules. On the other hand, lightweight or SPV (Simplified Payment Verification) nodes do not download the entire blockchain. Instead, they download block headers and rely on full nodes for transaction verification. Mining nodes are specialized full nodes that participate in the mining process to add new blocks to the blockchain and receive rewards. We see the concept of mining and consensus in the following section.

2.3.1 Consensus and Proof-Of-Work

Blockchain and cryptocurrencies rely on a robust consensus protocol; in Bitcoin, there is Nakamoto consensus, which is both effective and simple. Its simplicity comes from the Proof of Work process, where a miner independently proposes a solution. It's not a cooperative or distributed effort.

Before a miner adds transactions to the Blockchain through Bitcoin mining, the memory pool of each blockchain node Memory Pool may contain conflicting entries. The memory pool is stored locally on each node in the network. Each node in a blockchain network, such as Bitcoin, maintains its version of the Mempool. This Mempool is typically kept in the node's main memory (RAM), where unconfirmed transactions are temporarily stored until they are either included in a block or discarded.

The activity of adding a block to the blockchain is called mining because the nodes solving the consensus protocol problems, called miners, receive newly minted coins as a reward for doing so. Over time, the value of minted coins decreases due to Bitcoin's finite supply. Blocks contain multiple transactions, which are more energy-efficient than having a block for one transaction. Once a miner includes a transaction on a block and wins the challenge to add said block to the blockchain, each other node removes the transactions already there from their memory pool after receiving the updated blockchain.

The consensus protocol is a competition to solve a complex problem and add a block to the blockchain by adjusting a value in the block header. The block header is a compact summary of transactions and metadata, much smaller than the full transaction data.

The table from Mastering Bitcoin in Figure 2.1 summarizes the block header format. Timestamps in a distributed system are approximations. Transaction verification relies on a Merkle Tree Root Hash stored in the header. This approach prevents tampering with individual transactions, as any change alters the Merkle Tree Root hash and, in turn, the header hash. The header hash is crucial to the consensus protocol.

Each new block includes the hash of the previous block. Changing any block's header requires recalculating the Proof of Work for all subse-

¹<http://creativecommons.org/licenses/by-sa/4.0/>

Size	Field	Description
4 bytes	Version	A version number to track software/protocol upgrades
32 bytes	Previous Block Hash	A reference to the hash of the previous (parent) block in the chain
32 bytes	Merkle Root	A hash of the root of the merkle tree of this block's transactions
4 bytes	Timestamp	The approximate creation time of this block (seconds from Unix Epoch)
4 bytes	Difficulty Target	The Proof-of-Work algorithm difficulty target for this block
4 bytes	Nonce	A counter used for the Proof-of-Work algorithm

Figure 2.1: Figure showing the structure of a Bitcoin block header. Image from Mastering Bitcoin [13], licensed under CC BY-SA 4.0 ¹.

quent blocks. The goal of Proof of Work is to compute the header hash by adjusting the nonce until it meets the required threshold. A block's Proof of Work is solved if its hash meets the difficulty target d , which is adjusted by changing the number of leading zeros. The nonce is 32-bit, allowing up to 2^{32} trials. The Proof of Work function involves adjusting the nonce (x) to satisfy the challenge (c). If the values of d , c , and x are known, the function can be easily computed. However, if the nonce x is unknown, it becomes a computationally intensive problem within the class of NP (nondeterministic polynomial time) problems. While it's difficult to solve without guessing, it's still feasible and can be verified quickly once a solution is found. Once the Proof of Work is solved, the miner sends a P2P message to inform other nodes. We acknowledge that different approaches to consensus exist today, such as Proof of Stake, which reduces the environmental impact of Proof of Work and has been adopted by the Ethereum blockchain.

2.4 Ethereum and Smart Contracts

The Ethereum blockchain powers an integrated programming language called Solidity, which is Turing complete [15]. Solidity allows the creation of so-called smart contracts.

Ethereum makes a distinction between two types of accounts:

- externally owned account: we will refer to them as Ethereum Wallet Ethereum Accounts in this thesis. They are personal accounts with a private key that persons may use to send funds by creating

transactions. The account is stored in the Ethereum Blockchain alongside its balance. They are the accounts capable of creating a Smart Contract

- **contract accounts:** those accounts are accounts for Smart Contracts, which are stored on the blockchain and are composed of storage a storage area and code. They do not have a private key to sign transactions, but they can be interacted with by having messages sent to them, which are message call that triggers the execution of code [16], which allows them to change the Smart Contract state and storage.

2.5 Self Sovereign Identity

Self-Sovereign Identity (SSI) is a paradigm shift in the concept of digital identity, which empowers individuals to own and manage their digital identity fully [8].

SSI technology allows the generation of digital certificates and the sharing of embedded information under the control of the individual they refer to. These digital certificates are called Verifiable Credentials (VCs), and the main actors in the SSI scenario are the holder, the Issuer, which signs a VC with its private key, and the Verifier, which verifies the VC signature with the Issuer's public key.

The Issuer creates claims, which are statements about identity; this term overlaps with the term credential in the SSI community; it must be noted that credentials are as stated for VC authenticated with a digital signature.

Said VCs are issued from and to subjects identified by a Decentralized Identifier (DID), which we introduce in the following section. Blockchains are pivotal in enabling SSI and DIDs, providing a secure platform for storing key pairs of DIDs. VCs are securely stored in a wallet under the holder's control, and any stakeholder can act as a verifier, checking the validity of the data by cryptographically verifying related VCs.

The core ten principles that were proposed for SSI [17] are:

1. **Existence:** A person's identity is rooted in their inherent existence and is not solely digital. SSI ensures that some aspects of this

inherent identity are public while preserving the core self.

2. **Control:** People have authority over their identities, including the ability to update, hide, or share information. Users are in charge of their identity but cannot control others' claims about them.
3. **Access:** Persons must always have full access to their identity data without any hidden elements or gatekeepers. This ease of access does not mean they may access other people's data.
4. **Transparency:** Identity systems and algorithms must be open and understandable to ensure trust and accountability.
5. **Persistence:** Identities should endure over time as long as the user desires, with the possibility for modifications or disposal, respecting the user's right to be forgotten.
6. **Portability:** Identity information should not be confined to any single third-party entity, ensuring users can retain control regardless of changes in jurisdiction or service providers.
7. **Interoperability:** Identities should work across different systems and borders, allowing global use without losing user control.
8. **Consent:** Users must agree to how their identity is used and shared, ensuring that Consent is explicit, deliberate, and informed.
9. **Minimalization:** Only the minimum necessary identity data should be disclosed for a given purpose to protect user privacy.
10. **Protection:** The identity system should prioritize the rights and freedoms of users, ensuring security and decentralization in identity authentication.

One particular issue that all SSI-based systems face is the problem of trust, as traditional digital identity Certificate Authorities (CA) provide verified information about the binding of the real identity and the digital identity of individuals, organizations, devices... In SSI, while the binding between VCs and DID is possible, establishing the binding between the DID and an identity is not prescribed. Solutions proposed are a Web of Trust between issuers or integrating CA as a root of trust for DIDs [18].

2.5.1 DID

DID are a type of Uniform Resource Identifiers (URI) [19], representing a digital identity and resolving to a so-called DID Document. In this way, the digital identity is described as verifiable as there is a key pair associated with the DID while being decentralized from a central element such as an identity provider or CA. DID represent a DID subject, and the DID controller decides which subject they represent. Usually, it corresponds to a person creating their own DID, but it may differ, as a DID subject represents a sensor.

It is fundamental for a DID to be resolved to a DID document; how it can be resolved depends on the DID method, which specifies how a DID may be created, resolved to a DID document, updated, or deactivated.

In particular, the DID document holds a public key that may be used to verify signatures with its associated private key.

There are multiple DID methods; in particular, there is the DID method WEB, which leverages web pages to host the DID document and is resolved by the use of Domain Name System [20].

While there is this alternative, mainly DID methods use a blockchain to host DID documents, so DID methods usually resolve DID to blockchains. Some DID methods have been proposed, such as DID:ETHR based on the Ethereum Blockchain or DID:sov/indy, which is based on Hyperledger Indy Blockchain.

The format of the DID scheme is divided into three parts, each separated by a colon (:), i.e.:

```
did:<did-method>:<unique-id>
```

The first part shows that the URI is a DID, the second is the DID method, and the third is the unique identifier in the domain of said DID method. Some examples of DID are:

```
did:ethr:0x8e23d12445a5e2a5d2f7e041a38e5b0bfc5d21b6
did:sov:WRfXPg8dantKVubE3HX8pw
did:web:example.com:users:alice
did:web:w3c-ccg.github.io:user:alice
did:key:z6Mkf9q6tH9jzA6Mtvf1EDdrC5ExVZX8Pp3u5f55hvJW8C4rz
```

The Proof Of Work mentioned in section 2.3.1 allows verifiable information on the blockchain, which is crucial for verifying DID, as the public

key associated with the DID is written on the blockchain.

DID:ETHR

The DID method DID:ETHR leverages Ethereum Smart Contracts, as discussed in subsection 2.4, to associate each Ethereum Wallet Address with a unique DID. This method uses the corresponding Ethereum key-pair for cryptographic operations related to the DID.

The smart contract used is the Ethereum DID Registry. Said contract allows Ethereum addresses to present signing information about themselves with no prior registration [21].

The Ethereum DID Registry considers all Ethereum accounts identified by an address to be account identifiers without the need to be registered by the owner. Note that the Ethereum account address is derived from a public account key. The associated private key is the one that the DID controller uses for signature.

Using the ETHR DID Resolver, it is possible to use any Ethereum Wallet address as a fully self-managed DID. Secp256k1 public keys are the public keys from which the addresses are derived. An Ethereum address can be encoded as a DID by prepending on ethr as i

`did:ethr:ethereum_account_address`

the DID Resolver is capable of said DID to a DID Document by looking at the events related to the Ethereum address based on the Ethereum DID Registry[22].

DID:KEY

The DID:KEY method is a DID that does not need a registry such as a Blockchain to store its DID Document. DID:KEY does not need the setup of publishing the DID Document somewhere and is the most suitable for subjects that will receive few interactions; it is recommended to be used for short-lived use cases.

The DID document of the DID:KEY method can be generated by the Document Creation Algorithm, which expands the DID identifier to create it [23].

DID:web

The DID is a web method that does not use a registry; instead, it uses a website page accessible through the HTTPS protocol to store DID documents. In such a way, any person or organization with an internet domain can store on their website their DID Document without relying on a blockchain registry [20], which may lead to an easier adoption to whom does not want to interface with such registries. There is still decentralization as each subject retains control of their DID by hosting it on their domain. Trust in this method is established given the security of the domain's infrastructure and using HTTPS for document access.

2.5.2 VCs the W3C Data Model, JOSE and JWT

VCs, as defined by the W3C Data Model specification, are an instance of a digital certificate that holds a set of claims about a subject [24]. VCs are cryptographically secure and privacy-respecting. They are issued by the Issuer actor, who performs a digital signature using their DID private key and stores it by their holder in a digital wallet. Each VC contains the DID of the Issuer and the holder it refers to, ensuring that there is a link to their identities. The holder of VCs can create Verifiable Presentations (VPs) containing one or more of their VCs and send them to a Verifier. The Verifier will verify the signatures of the issuers by looking up their DIDs in a Verifiable Data Registry, which usually corresponds to a Blockchain but could also be a website in the case of the Issuer using a DID with method DID:WEB.

We may imagine a real-world scenario involving VCs, VPs, and DIDs. The scenario is as follows: Alice is applying for a Tech Corp job requiring a University degree and an English proficiency certificate. Alice holds both of these credentials as VCs in their digital wallet:

- The **University** issues a VC to Alice for their degree. The University digitally signs this VC using its DID private key.
- The **English Language Institute** issues a Verifiable Credential to Alice for their English proficiency level, also signed using its DID private key.

Alice creates a Verifiable Presentation (VP) that includes the University degree and the English proficiency certificate. She submits this VP to

Tech Corp as part of their job application. Tech Corp, acting as the Verifier, checks the digital signatures on the VCs by retrieving the public keys from the Verifiable Data Registry. If the signatures are valid and the integrity of the VP is correct, Tech Corp accepts Alice's credentials as authentic. We can see the workflow we detailed in Figure 2.2. Note that the DID methods of the issuers are not necessarily the same; the VPs holding their issued VCs can still be verified by the Verifier going to the appropriate Verifiable Data Registry accordingly.

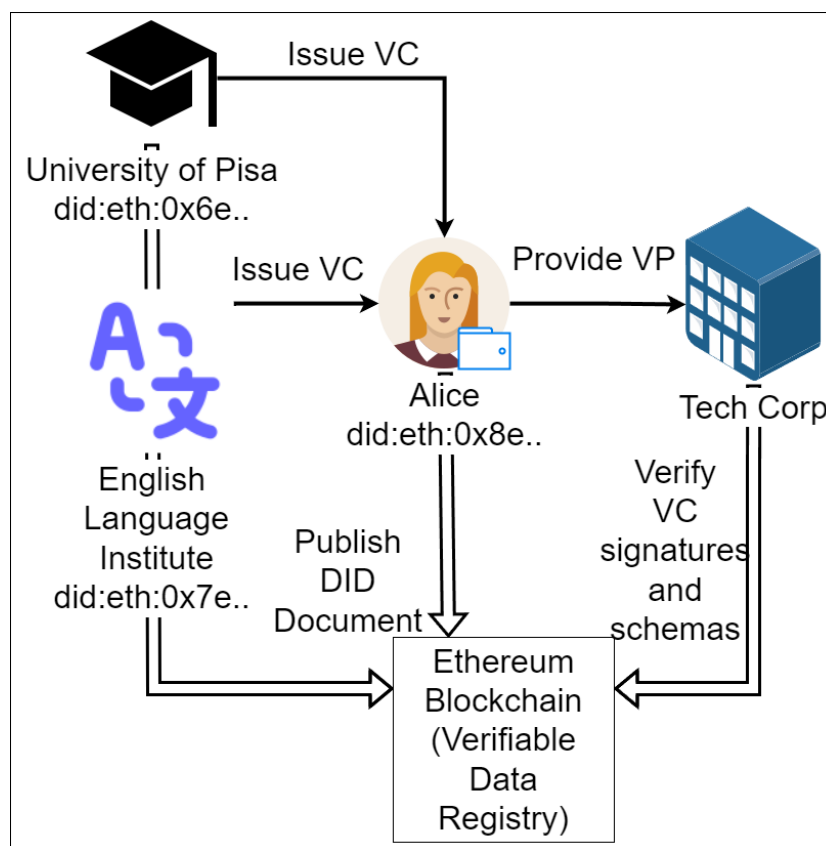


Figure 2.2: The of the workflow of VCs issiomg. A VC is issued by the University and Certificate Issuer and presented by Alice to a Tech Corp. The Verifiable Data Registry holds the DID and (data) schema involved in the VC

VCs use two different types of data schemas that enforce a structure on the data contained within them: 1) **Data verification schemas**, which ensure that the structure of a VC adheres to a published schema, defin-

ing its fields and expected data types; and 2) **Data encoding schemas**, which map the contents of a VC to a specific format for encoding. In the ‘credentialSchema’ field of a VC, the Issuer can include one or more schemas that the Verifier can reference to validate the conformity of the VC’s structure and data. These schemas can vary in structure, but the most widely used format is the Verifiable Credentials JSON Schema, as defined by its specification. Based on these schemas, a verifier can perform syntactic checks, and the JSON Schema validation standard rules specify the verification mechanisms [25].

VCS have two classes of securing mechanisms, which are not mutually exclusive and are related to the digital signature from the Issuer.

- An embedded proof, where the signature is written inside the *plain-text VC* with the verification method.
- An enveloping proof, which embeds the whole *plain-text VC* in a **cryptographic hash**. This cryptographic hash is produced by the Issuer using a digital signature process.

The embedded proof is generally inside the *plain-text VC*, which can be enveloped by a JSON Web Token (JWT)[26]. JWTs are often used to securely transmit the VC, providing an additional cryptographic layer through signing and encryption, as specified by the JOSE (JSON Object Signing and Encryption) framework.

The enveloping proof allows the **JWT** to be decoded back to the *plain-text VC*, ensuring the information’s authenticity and integrity.

The most common enveloping proof mechanism is specified by the Verifiable Credential data model with JSON Object Signing and Encryption (JOSE) specification [27]. Veramo, the SSI implementation tool we exploited in this thesis, uses JWTs within enveloping proofs to ensure secure credential sharing.

2.6 Large Language Models (LLMs)

Language models (LMs) are computational models that can understand and generate human language. LMs have the transformative ability to predict the likelihood of word sequences or generate new text based on a given input [28]. Large Language Models (LLMs) are LMs with a large number of parameters, leading to very complex models with exceptional

learning capabilities. This complexity means that LLMs require an enormous quantity of training data, much more than traditional machine learning models, to ensure they have high generalization capabilities, meaning their ability to work on unseen data instances.

Before delving into specific models like GPT and BERT, it is essential to understand how these models process text. They start the text processing by transforming it into **tokens**. Tokens are smaller units that represent segments of text. Tokenization involves breaking down text into words, subwords, or characters, depending on the tokenization strategy used. For instance, the sentence "The quick brown fox" might be tokenized into individual words like "The," "quick," "brown," "fox," or even smaller units if using a more granular approach.

Once the text is tokenized, each token needs to be represented in a way that captures not just the token itself but its meaning and relation to other tokens. This representation is known as a text embedding. An embedding is a vector representation of a token, which places each token into a **vector space**. In LLMs, embeddings are crucial because they allow the model to work with text data in a continuous vector space, converting tokens into fixed-size vectors and enabling the model to understand and generate text based on these dense, meaningful representations. In this vector space, tokens with similar meanings are located in said space closer together[29], allowing the model to understand and leverage semantic and syntactic similarities and differences.

2.7 Transformers: GPT and BERT

General Purpose Transformers (GPT) and Bidirectional Encoder Representations from Transformers (BERT) are both large language models (LLMs) built on a transformer architecture. Transformers are a particular type of neural network. A neural network with a Transformer architecture comprises an encoder and decoder unit.

- An encoder, given an input vector of **input features**, generates an **internal representation**, which is another vector.

²<https://github.com/dvgodoy/dl-visuals>

³<https://creativecommons.org/licenses/by/4.0/>

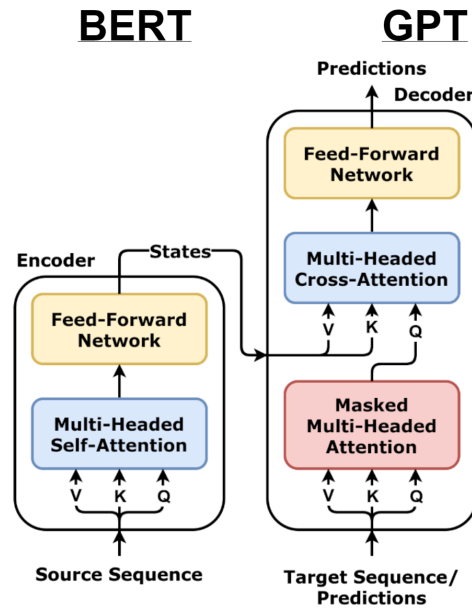


Figure 2.3: The Transformer model architecture, showing a separation between the encoder on the left, used by BERT, and the decoder on the right, used by GPT. Image by dvgodoy² / licensed under CC BY 4.0³.

- A decoder takes the **internal representation** and tries to reconstruct the given **input features** vector.

The substantial difference between GPT and BERT lies in the following:

- GPT primarily uses the Transformer's decoder part for generative purposes, as it predicts the next word in a sequence. GPT considers only the context to the left of the next word to generate it, making it a unidirectional model.
- BERT uses the Transformer's encoder part to produce contextual representations. Unlike static representations, these are influenced by the surrounding tokens in a sentence. This means the token's left and proper contexts condition each token's representation. For instance, the token "bank" would be represented differently in the contexts of "river bank" versus "bank account," thanks to these contextual representations, which are a particular type of embedding.

Figure 2.3 shows the Transformer model architecture [30], highlighting

how the decoder part (used by GPT) is on the right side while the encoder part (used by BERT) is on the left side. BERT is deeply bidirectionally trained, unlike OpenAI's GPT model, which is trained unidirectionally [31], as GPT considers only the tokens to the left of the current position.

2.8 Named Entity Recognition (NER)

We describe NER models by defining a Named Entity (NE). "A NE is a word or a phrase that identifies one item from a set of other items with similar attributes." We see a Named Entity Recognition Task as locating a word or phrase referencing a particular entity within a text[32].

NER input is an indexed sequence of tokens, and the output is

- Start index of an NE defined by one or multiple tokens
- Start index of a named NE defined by one or multiple tokens
- the NE type from a selected domain

A NER model can be classified into three: rule-based, unsupervised learning, and feature-based supervised learning, in particular with the use of Neural Networks and Deep Learning, allowing learning the representation of input data with multiple levels of abstraction[33].

Chapter 3

Related Work

Given the widespread usage of Social Media, the problem of the trustworthiness of Social Content is a well-researched subject. Social Media represents a set of software tools, such as online social networks (OSNs), which allow individuals to publish content online. Those OSN are platforms on which the publishing of content, which can be articles, videos, news, or pictures, is easy [34].

On the one hand, Social Media allows anyone to express themselves on online platforms and provide quality content to a large audience of internet users.

On the other hand, the surge in the quantity of content produced means that the quality of trustworthiness is not guaranteed. Anyone can write a **statement** about a topic or news. In this thesis, we will look at the Social Content of the textual form, particularly **online statements**, which represent news or articles. Furthermore, the literature is rich in discussing the problem of fake news and offering solutions. Given this, we will continue this chapter by defining fake news, its characteristics, and some solutions proposed to them.

There exist many definitions of fake news; some definitions are broad, such as "fake news is false news" or narrow, as "fake news is intentionally false news published by a news outlet" [35]. In the literature, some definition includes the concept of propagation via media, including social media; one definition is "A news article or message published and propagated through media carries false information regardless of the

means and motives behind it" [36].

We must consider that even non-journalists can reach a mass audience with social media. As such, the quantity of In some instances, even harmful but true news may be considered on the same spectrum as fake news, for example, bad news about a company that is factual (true) but harms a company (negative news, e.g., layoffs). Some researchers found a direct correlation between fake news and the stock market for a company. A conclusion reached by researchers is that fake news has a detrimental effect (albeit small) on a company's valuation, potentially leading to a decrease in its stock value [37]. Companies struggling in the stock market, receiving negative social media mentions, and employees facing stress are more likely to be targeted by fake news.

Given the criticality of correctly identifying what we are mitigating and our aim of protecting the freedom of expression while preserving the privacy of who shares information, we will define fake news as: ***Fake news refers to any statement published online across various platforms, including OSN, that is disseminated by sources that may or may not have malicious intent and is propagated by users who do not critically assess the veracity of the information and the authorativeness of the source.***

3.1 Propagation Characteristics

"Lies spread faster than truth" is a consensus among researchers. False information is 70% more likely to be shared than true information, often because fake news is perceived as novel. Studies indicate that false news attracts more attention and is shared more widely due to its novelty, while true news has the trend of spreading more slowly.

Researchers found that using fact-checking resources like Snopes.com influences user behavior.

When false news is commented on with a link to Snopes, the likelihood of users canceling a reshare increases.

Analysis of information cascades on Facebook shows that false news is more likely to receive such comments, leading to fewer subsequent reshares.

In contrast, true news tends to result in deeper and larger cascades of

reshares [38].

3.2 Widely used solutions and discussed characteristics

This section highlights some solutions and discussions proposed in the literature revolving around fake news, its propagation, and the tools and approaches to recognize it. We can categorize the mitigations in three method categories [36]:

- **Content-Based Identification:** This method focuses on verifying the content of the information. Refer to 3.1 for specific methods and their descriptions, and 3.4 for an overview of the category pros and cons.
- **Feedback-Based Methods:** These methods classify news based on user responses on social media. For details on specific feedback-based techniques, see 3.2; for a discussion on the pros and cons, refer to 3.5.
- **Intervention-Based Solutions:** This approach encompasses computational solutions for actively identifying and containing the spread of false information, thereby mitigating the impact of exposure to false information. Specific intervention methods can be found in 3.3 and the associated pros and cons in 3.6.

Table 3.1: Content-Based Fake News Detection Methods: Descriptions

Specific Method	Description
Cue and Feature-Based	Analyzes textual characteristics and linguistic cues for misinformation.
Deep Learning Methods	Employs CNNs and RNNs to automatically extract complex features from content.

The following subsections provide more in-depth discussions of fake news and the techniques to counter it, which are covered in multiple types of research.

Table 3.2: Feedback-Based Fake News Detection Methods: Descriptions

Specific Method	Description
User Responses & Propagation Patterns	Utilizes patterns of user engagement and propagation on social media as indicators.
Response Text Analysis	Analyzes text from user responses to identify credibility indicators.

3.2.1 Study on Fake News Credibility

Researchers found that *non-credible news published on an unreliable website, forwarded by an untrustworthy user is more likely fake news than news posted by an authoritative and credible user*[35].

It may be possible to detect fake news by *detecting unreliable website source*.

Headline Credibility

When a headline is very catchy and generates a strong emotional response, there is a high risk of fake news. We refer to catchy headlines as **clickbait**. News with a clickbait tile may be true news, but we highlight that given the cognitive load of reading many of them, it may be possible to fit in between them fake news that can be recognized as true by a reader.

A way to mitigate fake news is to use clickbait detection as an indirect way to detect fake news. One solution proposed is to use **decision trees gradient boosted to detect clickbait headlines**. Said solution can be applied because headlines have characteristic adjectives that convey authority and sensationalism.

Note: To avoid feature engineering, they also used Deep Learning, which is capable of learning a representation of the headlines automatically, without manually creating features describing them.

Table 3.3: Intervention and Identification Strategies: Descriptions

Specific Method	Description
Decontamination & Competing Cascades	Models diffusion of true news to counteract misinformation using strategies like decontamination.
Multi-Stage Intervention	Adapts interventions based on observed fake news propagation, using dynamic strategies.
Network Monitoring	Involves monitoring suspected sources of misinformation through various means.
Crowd-sourcing	Employs community feedback mechanisms for flagging content as potentially fake.
User Behavior Modeling	Uses models of user behavior and news-sharing patterns to prioritize fact-checking.

News Source Credibility

It is possible to use a source website's credibility, quality, and political bias to determine the quality and credibility of news.

In truth, some researchers concluded that *that PageRank and HITS can be used to assess the credibility of a page to put on top of search engine*. The problem is that **there may be some web spam to improve the ranking**.

Web spam can be:

Table 3.4: Content-Based Fake News Detection Methods Pros and Cons

Specific Method	Pros	Cons
Cue and Feature-Based	Direct analysis of misinformation content.	Needs new cue sets for different contexts; limited generalizability.
Deep Learning Methods	Captures complex patterns and nuances in data.	Requires extensive labeled data sets; attackers crafting fake content to resemble truth challenges detection.

Table 3.5: Feedback-Based Fake News Detection Methods Pros and Cons

Specific Method	Pros	Cons
User Responses & Propagation Patterns	Exploits the dynamics of social interactions for detection.	Reliant on substantial and often late-stage of spreading user interaction data.
Response Text Analysis	Leverages direct user feedback on content.	More suited for qualitative insights; limited detection accuracy.

- Including spamming title, body, meta-tags, anchor, and URL
- Cloaking and redirection
- Link spam to get a high score with HITS algorithms

There have been some websites *credibility assessment systems*, which check the content and link features passed on a machine learning framework (on one paper). Another paper uses a multi-layer probabilistic model.

Table 3.6: Intervention and Identification Strategies Pros and Cons

Specific Method	Pros	Cons
Decontamination & Competing Cascades	Offers a method to reverse misinformation effects.	Resource-intensive and operates as a corrective measure rather than preventative.
Multi-Stage Intervention	Enables dynamic, responsive measures to combat misinformation spread.	Assumes prior identification and tracking of fake news, adding complexity.
Network Monitoring	Allows for early interception of potential misinformation.	Can be costly and requires constant updates to remain effective.
Crowd-sourcing	Broadens the scope of monitoring through community participation.	Vulnerable to misuse by adversaries; flags may be unreliable.
User Behavior Modeling	Reduces the need for wide-scale interventions by focusing on high-priority cases.	Based on assumptions of simplistic network structures; limited in complex real-world scenarios.

News Comments Credibility

In this case, moderators or automated systems can analyze a particular **the user posts are** to detect fake news. Exploring the statements they post, based on:

- User stance
- Opinion

User comments on news websites and social media carry information on stances and opinions. It's essential to check the credibility of comments,

as the news can receive many compliments by "fake supporters." This phenomenon may happen for e-commerce websites.

This has been a research field in reviewing spam detection.

Researchers classify Comment credibility that can into:

- Content-based: extract features from comments (as fake news style-based detection)
- Behavior-based: spam users tend to have bursts of positive and negative reviews, while normal users are stable and uncorrelated in *their temporal rating pattern (may be positive or negative)*
- Graph (network)-based models: use graph models

Fake News Spreader Credibility

We can group Users into two categories:

- Malicious users: those who have low credibility
- Normal users: high credibility

The credibility can be measured:

- Through posts indirectly
- Via interconnection
- **Directly**

It may be possible that normal users may frequently and unintentionally spread fake news. We can distinguish participants in fake news spread activities in:

- Malicious users
- Naive users

Researchers found that *"false news was more novel than true news, which suggests that people were more likely to share novel information. While false stories inspired fear, disgust, and surprise in replies, true stories inspired anticipation, sadness, joy, and trust. Contrary to conventional wisdom, robots accelerated the spread of true and false news*

at the same rate, implying that false news spreads more than the truth because humans, not robots, are more likely to spread" [39]

Malicious Users Spread Fake News

Malicious users may spread fake news for monetary or non-monetary (benefits, entertainment) goals. The identity of the malicious user can be:

- It can be a **bot, a software application running an automated task over the internet.**
- Troll: a persona that posts inflammatory, digressive, extraneous, or off-topic messages to provoke emotional responses from targets.
- Cyborg: an account registered by a human as a camouflage to execute
 - Automated programs performing attack activities

Some studies analyze the bots and trolls' involvement in fake news, classifying them with their profiles. We can do this with a process of *modeling their behavior and content posted*; there exist models having high precision and recall in performing this.

Naïve Users

Humans are generally bad at assessing fake news and mistake fake news for truth. All influential factors may stem from:

- Social influence: peer pressure or network structure happens as exogenous factors (from the outside). How a trustworthy individual considers a piece of information and how likely it is for that information to be forwarded depends on *the physical network structure that models the potential exposure*.
 - Peer pressure also impacts user behavior towards fake news-related activities with a bandwagon effect, meaning that peers will share the same fake news one after the other.
- Self-influence: Users will more likely engage with fake news when fake news confirms pre-existing knowledge. We can approximate

the pre-existing knowledge by looking at the user's past contents (posts).

3.2.2 Fact-Checking by Mainstream Media Organizations

It emphasizes the role of online fact-checking resources fact-checking in the literature [2].

Mainstream media organizations perform fact-checking. There are truthfulness levels attributed to the news. A binary classification result cannot fully explain the overall problem.

3.2.3 Sources of Fact-Checking

There exist different sources of fact-checking:

- **Classify.news**: uses machine learning with labeled news to train the models and implement daily learning on credible/non-credible website sampling. They use a Naive Bayes classifier for the content model and a context-only model with an adaptive Boosting classifier.
- **FactCheck.org**: employs experts to evaluate the accuracy of statements.
- **Factmata.com**: uses ML and NLP techniques to identify and check "statistical claims by numerical relation extraction". They detect and verify information and provide an informed opinion to the world.
- **PolitiFact.com**: checks statements by elected officials, bloggers, and journalists in general, and performs a fact-check with ratings: "True, Mostly True, Half True, Mostly False, False, and Pants on Fire (blatantly false)." **Editors perform manual word analysis and complete context evaluation to determine the reliability of statements.** They offer an API for users to access the full text of statements, stories, promises, and updates on which they have fact-checked the truthfulness.
- **Snopes.com**: provides evaluations with truth ratings from experts.

- **TruthOrFiction:** checks rumors, warnings, hoaxes, virus warnings, and humorous and inspirational stories sent via email. They rate stories with categories such as Truth, Fiction, Reported to be Truth, Unproven, Truth and Fiction, Previously Truth, Disputed, and Pending Investigation.
- **OpenSecrets:** maintains an updated database of fake, false, conspiratorial, misleading news.
- many others..

100% automated detection is complex and has yet to be fully implemented as of today in the most popular OSN. Human intervention is required to maintain detection systems. Platforms such as *classify.news* and *Factmata* check claims and statements with statistical information. While *classify.news* uses ML, **they need high-quality training datasets and labels.**

3.2.4 Propagation characteristics

"Lies spread faster than truth" is the consensus reached by several authors. For example, researchers found that false information is more likely to be posted by 70% than true information, and they found that this is because "novel news" is more likely to be shared as the human mind is attracted to it. Users find it worthwhile to spread novel news, and there is the issue that fake news is more novel than true news [40].

Researchers found that using a fact-checking resource classifying true and false news, such as *Snopes.com*, influences users.

Researchers found that shared fake news, which has linked a comment to a post with a link to *Snopes*, increases the likelihood of a user canceling a reshare. Researchers did this study by analyzing an information cascade of news sharing on Facebook [38]. A result found is that false news is more likely to be commented on with a link to *Snopes.com* and that if it receives said comment, the cascade after the post receiving the comment will have fewer children. True news will be more viral, resulting in deeper and larger cascades of reshares, while there will be more cascades of false news that are shallower and less wide.

3.3 Solutions/Ideas Similar to ours

Our solution mainly leverages the blockchain as a trust anchor and Self-Sovereign Identity (SSI). In particular, blockchain has been widely discussed as a privacy-preserving mitigation against malicious users in OSNs who forward fake or undesirable content. Non-Fungible Tokens (NFTs), which are contracts on the blockchain that attest to ownership—in this case, of social content—allow the tracking of content sources to stop its forwarding by others [41]. In our case, we propose that social media platforms generate an NFT of ownership tied to the authors of posts, which can be blocked if their content is flagged as fake news by AI models or moderators/fact-checkers. Furthermore, we delve into the following sections in other proposed solutions:

3.3.1 TRUSTD

TRUSTD is a blockchain-based solution that utilizes Decentralized Identifiers (DIDs) and a collective signature-based ecosystem to enhance content credibility. In this system, authors, known as Content Creators (CC), sign the content they publish. Readers or fact-checkers, called Appraiser Actors (AA), then sign their evaluations of the content to provide a community-backed assessment of its credibility and accuracy [42].

3.3.2 Tracing with Blockchain

A blockchain-based solution has been proposed to trace news spread and analyze its propagation [43]. This solution involves:

1. Recording each instance of news sharing or uploading as a transaction for transparency and traceability.
2. Tracking the origin of fake news through transaction links.
3. Monitoring the rate and density of message forwarding to identify and limit the spread of dangerous content.
4. Storing transaction data on the blockchain while keeping actual media files in the cloud.
5. Encrypting media with symmetric key encryption and using keyed

watermarking to ensure future accountability and prevent tampering.

A solution to fake news leveraging Ethereum smart contracts, IPFS, and Machine Learning techniques has also been proposed in the literature [44].

3.3.3 W3C Credentials Community workshop: Verifiable Statements

In a community workshop, the W3C credentials community proposed creating a "new" base datatype and a derivation of it. A **Statement** base datatype was defined as "an assertion by an entity." They propose deriving it into **Verifiable Statement**: A tamper-resistant statement, where verifiers can cryptographically verify authorship. As such, it naturally could be included in a VC to become verifiable in the manner that we have seen in section 2.5.2.

Chapter 4

Veramo, IPFS, ESCO Ontology, and AI Tools

This chapter describes the tools used as building blocks to define the proposed solution detailed in chapter 5 and implemented in 6. We begin by detailing Veramo, which is utilized for creating Verifiable Credentials (VCs), Verifiable Presentations (VPs), and managing Decentralized Identifiers (DIDs), essential for leveraging Self-Sovereign Identity (SSI). VCs are generated to demonstrate the trustworthiness of online posts based on the user’s credentials. To achieve this, KeyBert is employed to extract keywords from posts, and Nesta’s Skill Extractor maps the keywords to skills. The ESCO ontology is the reference for both the skills extracted from posts and those in the authors’ VCs. Additionally, IPFS is used to store the VCs in a decentralized manner to assess the trustworthiness of posts. Finally, GPT is introduced as an alternative to KeyBert for keyword extraction and a tool for generating synthetic datasets for simulations discussed in Chapter 8.

4.1 Veramo

Veramo is an open-source JavaScript-based framework designed to manage DIDs and VCs easily and flexibly [45], as it follows a plugin architecture allowing the use of particular modules in applications, allowing extensibility to build on top of other modules.

Veramo offers tools to create, manage, and verify DIDs, as well as issue

and verify VCs. It abstracts the complexities of working with various blockchain technologies and standards, providing a unified interface for identity management.

In the implementation done for this thesis, Veramo interacts with Ethereum-based DIDs and DID having DID Methods: DID:WEB and DID:KEY (all described in section 2.5.1). We decided to use in our implementation in chapter 5 only the DID:ETHR method explained in subsection 2.5.1 and supported by Veramo; the DID:WEB and DID:KEY methods were considered theoretically for alternative SSI mappings in chapter 8. We supported both DID:ETHR on the Ethereum mainnet and the Sepolia testnet, as considered interacting with both the Ethereum blockchain and the Sepolia testnet. Veramo uses Infura to connect with the Ethereum blockchains. Infura is a managed service provider [46] that allows decentralized applications to connect to Ethereum and other blockchain networks without requiring developers to run their full nodes.

4.1.1 Veramo DID Agent and Veramo Javascript library methods

In Veramo, a Veramo DID Agent (Veramo agent) is a software program that performs the SSI framework operations using the Veramo plugin architecture modules. An agent is defined as the entry point for the Veramo framework. In particular, it is possible to enrich an agent's capabilities by extending it with modules. For instance, an Agent can be extended with a storage module, allowing it to have a wallet to store DIDs with their private keys and VCs. Furthermore, the agent can be extended with a module for issuing and verifying VCs. The following methods from its Javascript library are used in the implementations done for this thesis :

- `didManagerCreate`: This method creates a new DID and associates it with a wallet address or alias.
- `didManagerGetByAlias`: This method retrieves a DID based on a given alias which is given in creation.
- `didManagerImport`: This method imports an existing DID into the Veramo agent using a provided private key and alias.

- **dataStoreSaveVerifiableCredential**: This method stores a Verifiable Credential (VC) in the data store managed by Veramo.
- **createVerifiableCredential**: This method issues a new VC digitally signed by the issuer's DID. The signature is done by creating an enveloping proof of the VC. This signature is a hash that can be decoded to return the plain-text VC, as detailed in subsection 2.5.2.
- **dataStoreORMGetVerifiableCredentials**: This method retrieves Verifiable Credentials from the data store based on specified query parameters.
- **dataStoreORMGetVerifiablePresentations**: This method retrieves Verifiable Presentations from the data store based on specified query parameters.
- **dataStoreGetVerifiableCredential**: This method fetches a specific Verifiable Credential from the data store using its hash.

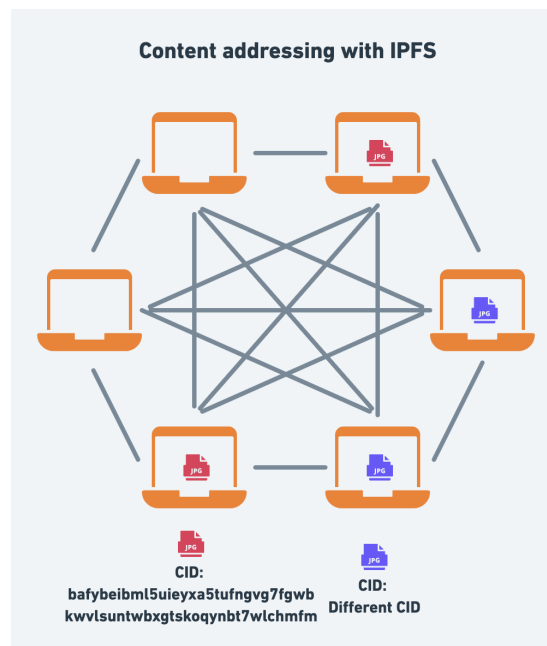


Figure 4.1: Figure: IPFS as a P2P network with multiple nodes storing two files, one red and one blue, with different CIDs [47]. Image by Daniel Norman⁴ / CC BY 3.0⁵.

4.2 IPFS

The InterPlanetary File System (IPFS)[48] is a distributed file storage protocol that allows users to store and share data in a decentralized manner. Instead of relying on a central server, IPFS utilizes a peer-to-peer network where each file is assigned a unique Content Identifier (CID). The CID is generated as a hash of the content data, ensuring that the same content always results in the same CID. This allows users to retrieve content from any peer on the network that hosts it.

We may see a representation of IPFS in Figure 4.1; we note how all nodes are interconnected and different nodes store different contents, which a CID addresses. We use IPFS to store digital assets associated with VCs, ensuring their integrity and availability across the network.

4.2.1 IPFS using Helia Library

In the implementation part of this thesis, the Helia JavaScript(TypeScript) library is used to interact with IPFS, providing a simple and flexible that follows a modular architecture [49], to manage data storage and IPNS (InterPlanetary Naming System) operations, allowing to create name files which can be mutable and be identified by the same identifier, contrary to CID. The Helia library provides essential functionalities for integrating IPFS (InterPlanetary File System) into applications. In this section, we discuss the primary functions provided by Helia, focusing on their usage.

- **createHelia**
 - Initializes an IPFS node.
 - Configures the node with a block store for persistent data storage.
 - Acts as the core entity for interacting with the IPFS network.
- **unixfs**
 - Provides an interface to interact with the Unix File System (UnixFS) within IPFS.

⁴<https://blog.ipfs.tech/2022-06-09-practical-explainer-ipfs-gateways-1/>

⁵<https://creativecommons.org/licenses/by/3.0/>

- Facilitates the addition and retrieval of files using a hierarchical structure.
- Abstracts the complexities of the IPFS DAG (Directed Acyclic Graph), offering a file system-like interface.

- **keychain**

- Manages cryptographic keys within Helia's `libp2p` module.
- Handles the creation, export, and import of cryptographic keys.
- Ensures secure signing and verification of IPNS records, maintaining the integrity of mutable links.

- **ipns**

- Manages the InterPlanetary Naming System (IPNS), enabling mutable content on IPFS, as it provides a mechanism for creating mutable links within the immutable IPFS environment.
- Associates a stable `peerId` with a changing CID, allowing content updates to be referenced by a consistent identifier. To do so, it uses cryptographic key pairs to sign and publish records mapping `peerId` to specific CIDs
- Publishes and resolves IPNS records, ensuring that the latest content can be accessed through the `peerId`. In this way, it enables dynamic content management by allowing updates to be retrieved via a consistent, human-readable name.

4.3 KeyBert

In our project, we exploited KeyBert [50] to extract from statements (posts) made online the keywords that are the most representative of it. KeyBert, in its default configuration, utilizes as BERT model the "all-MiniLM-L6-v2" model [51], which MiniLM architecture is lighter-weight than standard BERT models, making it more suitable for *online serving in real life applications for latency and capacity constraints*[52]. The algorithm KeyBert uses to extract keywords from text is illustrated in Figure 4.2. KeyBert uses the BERT model to generate text embeddings

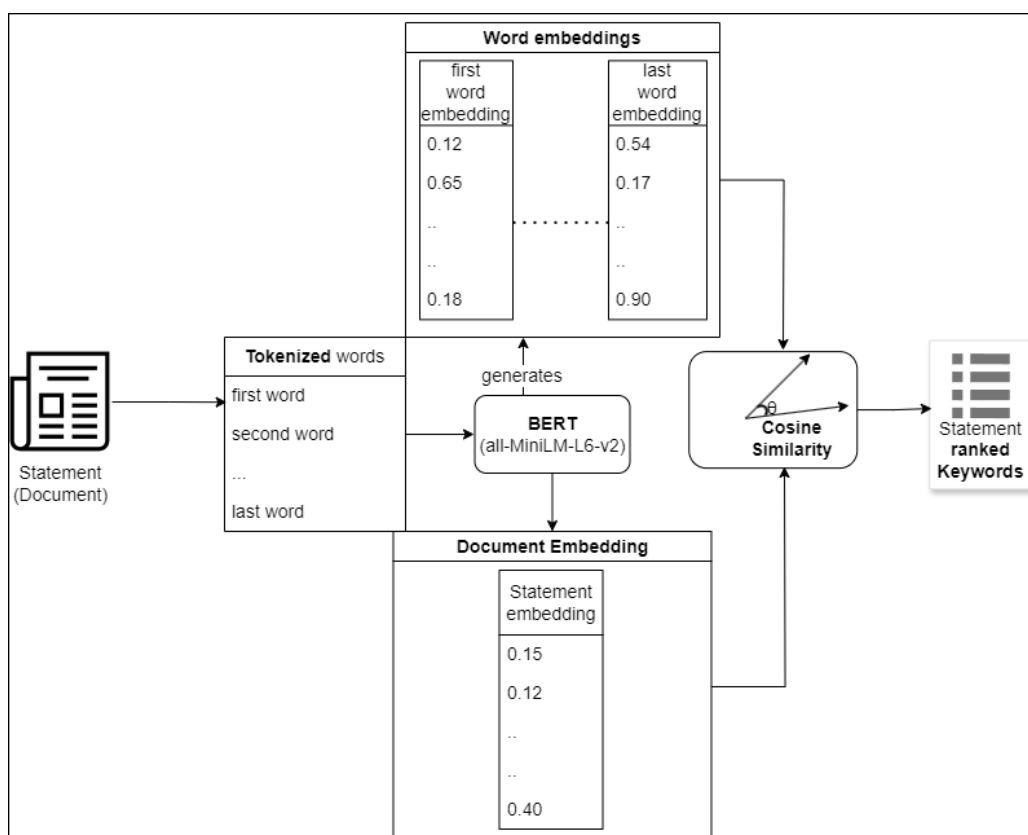


Figure 4.2: KeyBert tool keyword extraction process, an algorithm tokenized a statement, then a BERT model generates embeddings, ultimately an algorithm applies cosine similarity between the Document embedding and the Word embedding to generate a list of keywords

of two different lengths from a text; the concept of text embeddings was introduced in section 2.6. The text embeddings generated are :

- **Document embeddings:** embeddings representing the entire statement in vector form.
- **Word Embeddings:** embeddings representing individual words or short phrases (candidate keywords) within the statement in vector form.

keyBERT algorithm identifies the keywords that best describe the statement by using cosine similarity between Word embeddings and Document embedding. In particular, the Word embeddings with the statement Document Embedding are the highest-ranked keywords to represent the

whole statement. The cosine similarity between two vectors $\mathbf{A}$ and $\mathbf{B}$ is defined as:

$$\text{cosine similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

Where:

- $\mathbf{A} \cdot \mathbf{B}$ is the dot product of vectors $\mathbf{A}$ and $\mathbf{B}$,
- $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ are the magnitudes (or norms) of vectors $\mathbf{A}$ and $\mathbf{B}$, respectively.

Additionally, the KeyBERT tool allows using an LLM model for keyword extraction; in particular, it queries a provided OpenAI GPT API. To provide a more efficient extraction with LLM models, a pipeline in which the BERT model is used first, and the keywords it finds are passed to the LLM model can be created.

4.4 ESCO ontology

Our proposed system uses mainly the ESCO Skills & competencies ontology as the reference taxonomy for Skills. The tool we use to map keywords to skill supports also has its own toy taxonomy and the Light-cast taxonomy, which we do not consider.

The ESCO ontology is a project of the European Commission defined as *the European multilingual classification of Skills, Competencies, and Occupations. ESCO is a dictionary that describes, identifies, and classifies professional occupations and skills relevant to the EU labor market, as well as education and training.*

The ontology provides each skill with a primary and optional secondary label, description, and reusability grade. The ESCO ontology is provided in 28 languages, and we use the English version.

*Note that ESCO is structured in two main pillars: skills and occupations. Additionally, ESCO organizes skills into four subgroups: skills, knowledge, attitudes, and language skills, which provide a granular classification of competencies. Nonetheless, for convenience, we will refer to them as skills and detail them as being, for instance, from **skills from the skill subgroup or skills from the knowledge subgroup.***

Furthermore, ESCO is crucial in understanding the skills and competencies built upon qualifications. For qualification, we intend a certificate issued by an authority such as a university that attests to an individual that they have reached a particular learning outcome[53], such as a bachelor's degree. In particular, those certifying bodies can issue a digital credential with an electronic signature, the so-called European Digital Credentials for Learning, and it is highly suggested by the European Commission to add in the description of the learning outcomes the ESCO skills acquired.

Furthermore, ESCO skills can be seen as a descriptor of a person's authoritativeness, as the website Europass, which allows the creation of CVs, suggests based person profile ESCO Skills to add in their CV. This can be seen in Figure 4.3.

Regarding the ESCO skill Pillar, it is divided into four sub-classifications those are *i) Knowledge*, *ii) Language skills and knowledge*, *iii) Skills*, *iii) Transversal Skills*.

For each skill, there is a set of metadata, which is composed of the properties:

- **Preferred Term:** The primary term used to describe the skill or knowledge, referred to as a **label**.
- **Non-preferred Terms:** Secondary or alternative terms, also known as **secondary labels**, that are less frequently used.
- **Hidden Terms:** Terms that may be relevant but are not visible in the general taxonomy.
- **Description:** A general explanation of the skill or competence.
- **Formal Definition:** A precise and structured concept definition.
- **Scope Note:** Clarifies the context in which the skill or knowledge applies.
- **Skill Type:** Defines whether the concept is categorized as knowledge or skill/competence.
- **Skill Reusability Level:** Indicates whether the skill is sector-specific or transferable across different sectors.
- **Essential Occupations:** A list of jobs for which the knowledge,

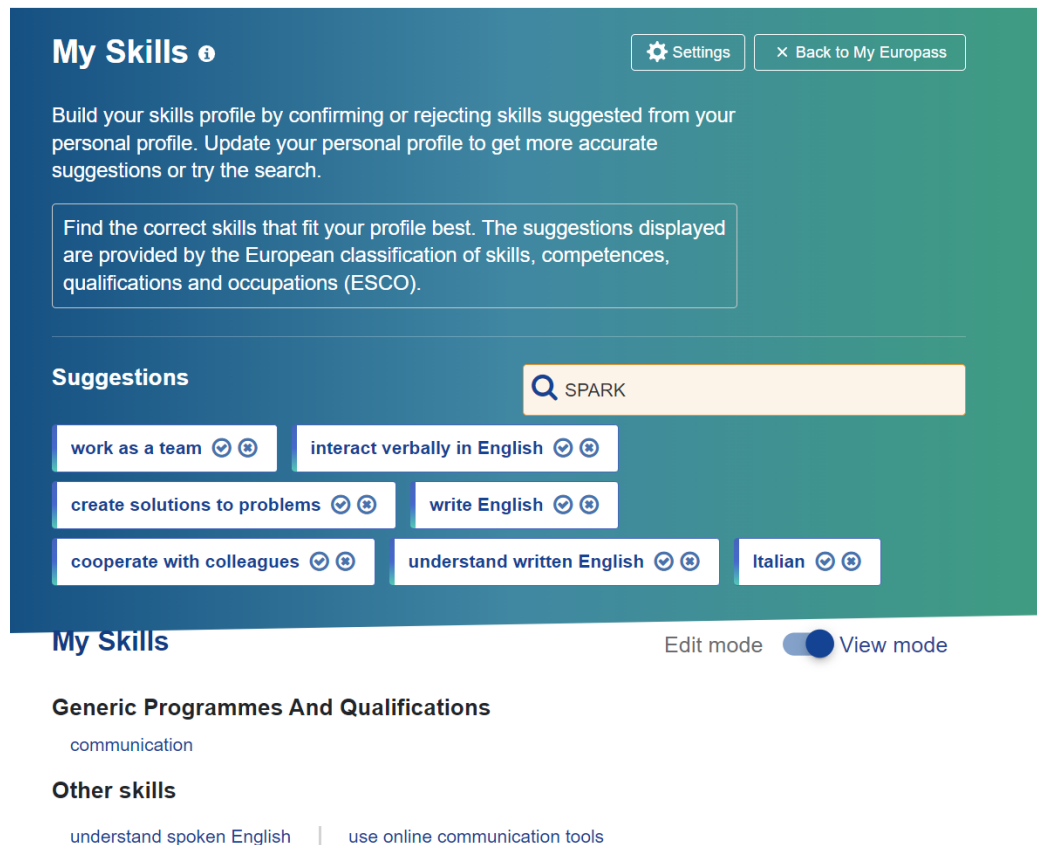


Figure 4.3: The interface of europa.eu/Europass for the addition of skills and the already added skill based on ESCO ontology for a person profile

skill, or competence is a core requirement.

- **Optional Occupations:** A list of jobs where the knowledge, skill, or competence is beneficial but not essential.
- **URI:** The unique resource identifier links to the skill's reference page on the ESCO website.

Furthermore, the ESCO Skills have a hierarchy, where a skill can be higher in the hierarchy when it is more general than another. Notice that a skill may be reached from a high-level skill contained in one subgroup or another. For instance, the skill **Java (computer programming)** can be reached from both the **Knowledge** subgroup and the **Skill** subgroup. Here are the paths in the hierarchy from more general skills to the **Java (computer programming)** skill:

- From the **Knowledge** subgroup:
Information and Communication Technologies (ICTs) -> Software and Applications Development and Analysis -> Java (computer programming)
- From the **Skill** subgroup:
Working with Computers -> Using Digital Tools for Collaboration, Content Creation, and Problem Solving -> Digital Content Creation -> Computer Programming -> Java (computer programming)

Additionally, we can see how said skill is presented on the ESCO website in Table 4.1.

4.5 Nesta's Skills Extractor Library: **ojd daps skills**

Note that this tool is both referred to as Nesta Skills Extractor and **ojd daps skills** by their creators and by us in this thesis; the two names refer to the same tool.

This tool uses a NER model based on a Neural Network, introduced in section 2.8, to map statements to a skill ontology, which can be provided to the tool. Its functioning can be seen in Figure 4.4

This skill extractor has been designed with three taxonomies as its

⁶<http://data.europa.eu/esco/skill/19a8293b-8e95-4de3-983f-77484079c389>

Table 4.1: Java (computer programming) skill from ESCO metadata

Category	Details
Skill Name	Java (computer programming)
Description	The techniques and principles of software development, such as analysis, algorithms, coding, testing, and compiling of programming paradigms in Java.
Alternative Labels	Java
Skill Type	Knowledge
Skill Reusability Level	Sector-specific skills and competences
Broader Concepts	Software and applications development and analysis Computer programming
Essential For	Cloud software developer, Computer-aided design operator, Computer science lecturer, Numerical tool and process control programmer
Optional For	ICT system analyst, Web developer, ICT network engineer, Mobile application developer, Integration engineer, Blockchain developer, Software tester, ICT technician,...
Concept URI	http://data.europa.eu/esco/skill/19a.. ⁶
Status	Released

target: a toy taxonomy, Lightcast’s Open Skills, and the ESCO ontology. We decided to extract skills only in the ESCO ontology. The Nesta Skill Extractor library extracts Skills. First, it recognizes skills from keywords or text and translates them into an internal taxonomy(the toy taxonomy), then it maps them in the designed one (ESCO ontology or Lightcast’s Open Skills)

⁷https://web.archive.org/web/20230329004058/https://nestauk.github.io/oj_daps_skills/build/html/about.html

⁸https://web.archive.org/web/20230531234250/https://nestauk.github.io/oj_daps_skills/build/html/license.html

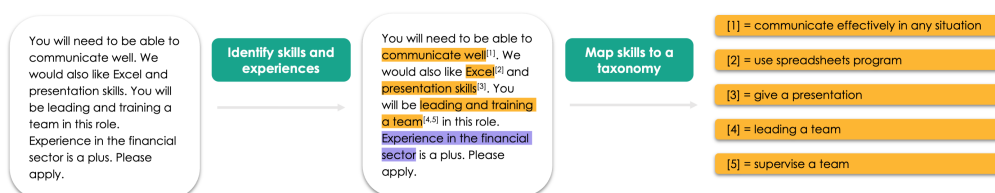


Figure 4.4: Figure showing the mapping from text to skill taxonomy performed by the Nesta Skill Extractor Library. Image by NestaUK⁷ / licensed under MIT⁸.

- NER model, trained on Job Adverts to predict Skills based on text. Given that we apply the tool to general statements, we are performing the so-called Transfer Learning, using a model trained for a task (extract skills from Job Adverts) to another task (extract skills from online statements)
- SVM model, which recognizes if a Skill predicted from the NER model is a single skill or a multi-skill, the latter represents a general skill, which is divided by the tool in individual skills.
- NER model, which is coincidentally all-MiniLM-L6-v2 too, using it to ultimately map the extracted skills to the targeted taxonomy by creating embeddings of what the NER model recognizes as skills to then map them with cosine similarity with the skills of the taxonomy.

4.5.1 A critical look to the tool

We acknowledge that the tool performance is average, based on the measures in the following table that the tool authors provided: The

Table 4.2: Performance metric of the Nesta’s Skill extractor tool for Skill mapping

Entity	F1	Precision	Recall
Skill	0.586	0.679	0.515

results were measured by matching, looking at partial entity matching from the tool reevaluate. We analyze only the skills results; the F1-score is average, with the precision being relatively high, meaning that when

the tool identifies something as a "Skill," it is often correct. This indicates that the tool minimizes false positives in this category. Meanwhile, Recall 0.515 signifies that the tool misses half of the skills it should predict. Given the limitations of the tool and as suggested by developers, it is fundamental to review the output of the tool output and not use it mindlessly [54].

The internal taxonomy-matched skills are then matched to the actual taxonomy selected, in our case, the ESCO ontology. Matching to an ESCO skill uses cosine similarity, assigned to an ESCO skill with a similarity greater than 0.70. If there is no clear match, an iterative process starts, where we find a parent skill group inside the ESCO ontology to assign this as output. We may see the process in Figure 4.5 At the end of the

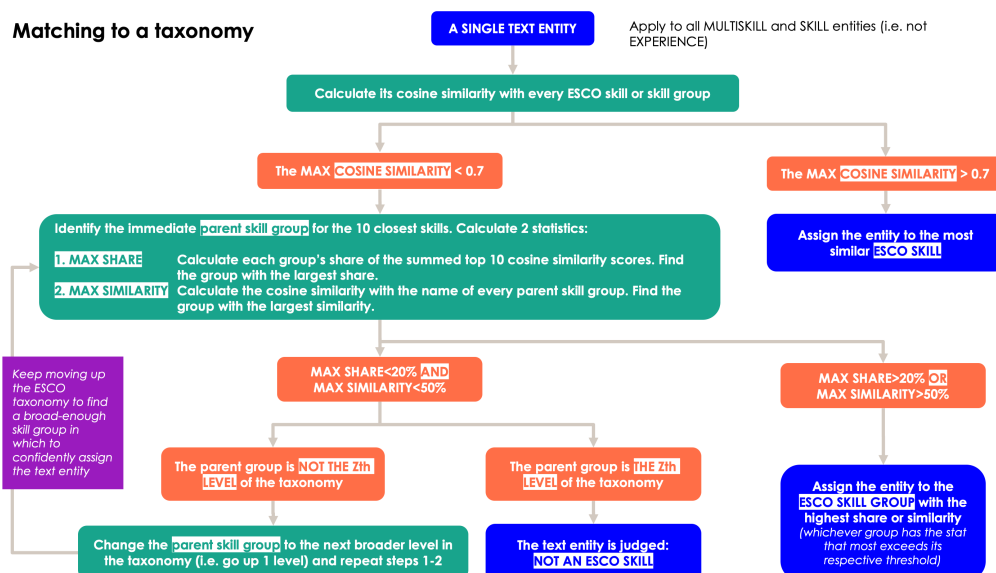


Figure 4.5: Figure showing the match flow from internally represented skill to ESCO ontology skills performed by the Nesta Skill Extractor Library. Image by NestaUK⁹ / licensed under MIT¹⁰.

tool processing, a list of skills is provided in the designated taxonomy.

⁹https://web.archive.org/web/20230329004058/https://nestauk.github.io/ojd_daps_skills/build/html/about.html

¹⁰https://web.archive.org/web/20230531234250/https://nestauk.github.io/ojd_daps_skills/build/html/license.html

4.5.2 Example

In this subsection, we illustrate an example by starting from a text, obtaining keywords from it with keyBert, and then mapping those to skills. Given the text in this listing 4.1.

Listing 4.1: Article from which to extract keywords

```

1 "title": "The Role of Bloomberg Terminal in Market Analysis",
2 "content": "The Bloomberg Terminal has revolutionized market analysis by providing
  financial professionals with real-time access to market data, analytics, and news.
  This powerful tool allows analysts to monitor global financial markets, track
  securities, and evaluate market trends. Features such as charting tools, price
  analytics, and economic indicators enable comprehensive analysis, aiding in informed
  decision-making. The Terminal's integration with other financial tools, like Excel
  and SQL databases, allows for seamless data analysis and reporting. Its extensive
  database and analytical capabilities make it an essential resource for performing
  thorough due diligence, identifying investment opportunities, and managing financial
  portfolios in an increasingly volatile market environment."

```

The keywords extracted from it are:

Bloomberg, Analytics, Excel, Analysts, Charting, Data, Bloomberg Terminal, Financial tools, Market tracking, Price analytics, Charting tools, Data analysis, Economic indicators, Securities, Identifying investment opportunities, Analytical capabilities, Database.

The skills extracted from `ojd_daps_skills` from the keywords can be seen in two tables 4.3 and 4.4. As can be seen, the tool returns the Skill label (primary term) and the Skill URI.

4.6 OpenAI GPT APIs

GPT models were fundamental for generating a synthetic dataset of OSN profiles and articles for the simulation in chapter 8. The OpenAI APIs allow to send instructions (prompts) to the models offered by OpenAI. The APIs are HTTP requests, with libraries written for Python and Javascript [55] that generate and send those requests. The API's so-called "chat completion" section is the primary type for text generation. Each request to chat completion contains a prompt for which the selected model generates a textual response. Besides the prompt, the requests hold some parameters that influence the text generation. The one that we

Table 4.3: Keywords and Skills Mapped from Nesta Skill Extractor Library (Part 1)

Keyword	Skill and URI
Charting tools	Transport services http://data.europa.eu/esco/iscd-f/1041
Excel	Use spreadsheets program http://data.europa.eu/esco/skill/1973..
Markets track	Marketing and advertising http://data.europa.eu/esco/iscd-f/0414
Economic indicators	Economic analysis http://data.europa.eu/esco/skill/995..
Identifying investment	Investment analysis http://data.europa.eu/esco/skill/bf12b461-1c..
Securities	Securities http://data.europa.eu/esco/skill/749ce4c2-4d..
Bloomberg	Finance, banking and insurance http://data.europa.eu/esco/iscd-f/0412
Data analysis	Data analysis http://data.europa.eu/esco/skill/7ee4c2ea-b3..
Analytics	Data analytics http://data.europa.eu/esco/skill/c5e8abde-d2..

found to be the most fundamental is the **temperature** parameter; we delve into this and other main parameters here below:

- **temperature**: this parameter controls the tradeoff between randomness and determinism in the model's response. The values range from 0 to 2. In practice, temperature changes how the next word to generate is selected given the previous words in a response. A low value such as 0.25 makes the answer more deterministic [56]. This parameter works in conjunction with the seed parameter. It was noticed that when giving instructions to adhere strictly to a particular response format, a low-temperature value was needed to enforce it; otherwise, the model would change it or add more content than necessary to the response. For instance, a low-temperature value is essential when a JSON output with a

Table 4.4: Keywords and Skills Mapped from Nesta Skills Extractor (Part 2)

Keyword	Skill and URI
Of Bloomberg	Finance, banking and insurance http://data.europa.eu/esco/iscd-f/0412
Analysts	Management and administration http://data.europa.eu/esco/iscd-f/0413
Analytical capabilities	Finance, banking and insurance http://data.europa.eu/esco/iscd-f/0412
Bloomberg terminal	Transport services http://data.europa.eu/esco/iscd-f/1041
Data	Data http://data.europa.eu/esco/skill/7ee4c2ea-b3..
Analytics and	Data analytics http://data.europa.eu/esco/skill/c5e8abde-d2..
Charting	Transport services http://data.europa.eu/esco/iscd-f/1041
Financial tools	Financial instruments http://data.europa.eu/esco/skill/0bd4c62a-c3..
Database	Database http://data.europa.eu/esco/skill/43ae58b9-5e..

predefined set of properties is required.

- **seed:** this parameter allows the programmer to set a set for the random number generator that allows random choices. If not provided, the seed changes every time the request is sent. However, when provided along with a low temperature, it makes the responses more deterministic, allowing to replicate always the output.
- **model:** this parameter allows the selection of the model that will process the request and generate the text. The different choices have different monetary costs. An incomplete list of models available, which have also have available specific snapshots of them are, as of September 2024: gpt-4o plus specific snapshots of it, gpt-4o-mini which has lower cost, o1-preview, gpt-4 and gpt-4-turbo,

gpt-3.5-turbo-0125 and many others. For instance, GPT-4-turbo is designed for faster and cost-effective processing, while GPT-3.5-turbo offers cheaper alternatives for less demanding applications.

Chapter 5

Assessing Social Content Trustability with SSI

In this chapter, we define the Social Content trustworthiness problem and fake news spreading issue, with a privacy concern, and we propose our solution by providing a system that proves the authoritativeness of Social Media content generators using the tools of chapter 2.

5.1 Problem statement and overview

Social content trustworthiness and fake news are issues that authorities have tackled with different strategies. Machine Learning models are usually used to flag false content or fact-checkers, who are responsible for reviewing posts on OSN to remove any false information posted. However, we highlight how a clear mitigation for this issue has yet to be found. In particular, most solutions try to stop fake news after it is propagated in an OSN system using human or automated fact-checking tools. This resolution is inefficient, as the propagation of false information will do some damage if not arrested promptly. As such, we define the problem of spreading fake news in OSN from the start of the diffusion process.

The diffusion process starts when the Author publishes the original **statement** online. In particular, a **statement** may be an article, editorial, news report, or just an OSN post or comment.

We see the problem from the point of view of trustworthiness in the

authors and their authoritativeness on the topic they write a statement about. Authors must prove their expertise on the topic they discuss. For example, the case of Belle Gibson highlights the consequences of non-authoritative individuals presenting themselves as experts online. They falsely claimed, online and in their books, to have cured cancer with an alternative approach to traditional medicine without any real medical qualification. This rumor led to many believing their claims, profiting from them, and hurting their health. **As such, a good mitigation to the problem of false information presented online is to assess the authoritativeness of the Author.**

On OSN, users are usually pseudo-anonymous and behind a made-up username. Even if they are competent in a subject, other persons or institutions may recognize their words as un-trustworthy. For instance, in 2019, on Reddit and YouTube, the username of Roaring Kitty, real name being Keith Gill, posted under his username some theories on the market value of a company called Gamestop; his theories were correct and led to him and others who invested following his advice some great return on their investment.

However, his initial claims were met with skepticism by a traditional financial institution, which shows how his authoritativeness was not perceived, even if Keith Gill is a Chartered Financial Analyst (CFA) [57], that is a certification issued by the Research Foundation of CFA Institute and a business degree graduate at the Stonehill College. Those credentials were not presented clearly alongside the statement in online posts and videos that they made, making them feel lacking in authoritativeness.

Currently, the statement authors only sometimes provide valuable information about their competence on the topic at hand when making a statement. This is understandable, as authors may wish to maintain privacy, and then they avoid exposing traditional credentials, such as a university degree with his name and surname when discussing politics. Authors of statements on OSN would like to express their opinions, even controversial ones, without being traced back to their real identities. In this thesis, an identity is seen as a set of personally identifiable information (PII), e.g., name, surname, residence address, spouse name, and many others. PII is defined as *"any information that can be used to distinguish or to trace an individual and any information that is linked (information associated) or linkable (linkable logically to other infor-*

mation) to an individual" [58], As an example, the proposed SSI-based solution would have been helpful in the case of Timnit Gerbu, a researcher in AI ethics, who published a paper that strongly criticized AI-generated languages as being centered toward generating content that reflects the wealthiest countries and communities' cultures [59]. Consequently, she lost her job because her employer had high stakes in developing language models such as BERT. While this example is not centered around OSN, it highlights how sharing a controversial opinion with a person's whole identity being disclosed leads to repercussions.

Another example where the proposed solution could have been successfully adopted is that of Dr. Li Wenliang, who warned his colleagues on a WeChat group chat about an infectious disease, which turned out to be COVID-19. Unfortunately, his warning messages were leaked to the general population. Unfortunately for the doctor, four days after the leak, public authorities made them sign a letter that stated that the comments were false, and public police investigated them for the spread of fake news [60].

5.2 Key actors and proposed solution

We define the **main goal** of our solution to be: *the assessment of statement trustworthiness by evaluating the Author's authority over the topic of the statement in a privacy-preserving manner*, i.e., by enabling the Author to provide proof of their authoritativeness on that topic without disclosing their real identity.

We break from past solutions in which an author publishes a statement, and human or bot fact checkers judge it as true or false. Our solution makes publishing a **credibility assessed statement** a collaborative process between the Author of the statement, the fact-checking tool, and the Reader, thereby ensuring everyone's active participation in the fight against fake news spread.

We acknowledge the limitation that **an assumption is needed: an authoritative person about a subject will likely make a statement that is true and correct** but we acknowledge that there is the possibility of **a reputable source spreading fake news either unknowingly or not**.

We define two objectives to accomplish to reach our main goal:

1. provide the Author of a statement as a way to **prove their authority on the subject in a privacy-preserving and verifiable manner**
2. to craft a system that provides the reader an **informative and verifiable** overview of the **statement author authority** over the subject allowing the reader to make their **own judgment subject**

For both objectives *SSI is the technological enabler allowing us to accomplish them.*

5.2.1 Key Actors

The key actors that we identify as being involved in the diffusion of statements are listed in this section. We list them here and see the mapping to our proposed system in subsection 5.5.

- **Certificate Issuer:** The Issuer is any entity responsible for providing credentials that verify a person's skills, competencies, or expertise in a subject. This could be an institution, organization, or authority capable of certifying an individual's capabilities. For instance, a university or a course provider.
- **Author:** An author can be a journalist, researcher, or any field expert who writes and publishes a statement (post), such as an article or opinion piece, on an OSN, potentially with controversial views. The Author attaches credentials to their statement, showcasing their relevant skills or expertise related to the content. This allows the Author to demonstrate their authority on the subject matter without exposing sensitive personal details. The certificates enable the disentangling of competencies and PII. This is because the certificates issued should or contain the PII, or those are hidden when presenting them. This will enable them to showcase their competencies without exposing their identity.
- **Spreader:** A spreader shares the Author's statement across the same or various online platforms, such as an OSN or a blog. The Spreader could be, first and foremost, the original Author or a reader such as an OSN user, another journalist, or a reader who finds the content valuable and wants to disseminate it. By sharing

the content, the Spreader contributes to its diffusion. Furthermore, it wants to indicate the credibility of the original statement by referencing the source from which it has seen the statement and the credentials attached to the source of the post from the authors.

- **Reader:** The Reader is the final consumer of the content. They interact with the statement in the place they find it from the Spreader that propagates it. They want to have the ability to verify its trustworthiness by checking the credentials associated with the Author. The Reader wants to trace the diffusion chain back and ensure the original statement and credentials are authentic. They were not tampered with. Besides, they want to map the skill in the statement locally with the skill in the credentials.

5.3 User Stories and High-level Overview

Below, we craft a narrative for each actor involved in our solution. Furthermore, we define some requisites, which are their wants and needs. Those are indicated by the first letter of the actor and a number (i.e., R1 for the first requisite of the Reader). In addition, those requisites will be referenced in this thesis to express where they have been met.

For the Reader, the crafted narrative is:

As a reader,

(R1) *I want to see an indication of the Author's verified authority on the topic before reading the statement;*

So that (R2) I can trust the information and make informed judgments about its credibility before reading it.

Meanwhile, for the Author, our narrative is:

As an author, I want to:

A1) *prove my expertise on the topic across many platforms*

A2) *without disclosing my whole identity.*

So that readers can have confidence in the accuracy and credibility of my statements without compromising my identity.

At last, for the Spreader, our envisioned narrative is:

As a spreader;

I want to (S1) share content from an authoritative source and (S2) provide a link to my source and the diffusion path of the content;

So that I am seen as a spreader of credible information, supporting my shared content with the original Author's verified expertise and the content's history.

The secondary actors that are involved in our system are the following:

- Certificate Issuers: an entity that produces a certificate with a set of skills for a person.
- Online Social Network Platform or Host Website: Any platform on which an author may publish the statement author published: Facebook, X(Twitter), LinkedIn, Reddit, or a news outlet website. It is fundamental for our solution to be pluggable as a software component on the majority of platforms to be the most effective in mitigating fake news.

The complete comprehensive conceptual overview, with the requisite points detailed before, can be seen in Figure 5.1. In particular, the workflow from certificate issuing to presentation to the Reader of the statement can be seen in Figure 5.2.

5.3.1 Use case

In this subsection, we detail a use case from the point of view of two actors, one that acts as the Author and another that acts as the Reader and Spreader.

On the one hand, considering the Author's point of view, we present this use case where Peter, the journalist, wants to publish an article on an OSN, acting as an Author. For instance, let us suppose Peter has a degree in computer science. Additionally, he took a course on business market pricing. Peter writes an article on market pricing using software for automating product pricing and their best practices, with some controversial opinions.

In this case, Peter does not want his real name to be associated with the

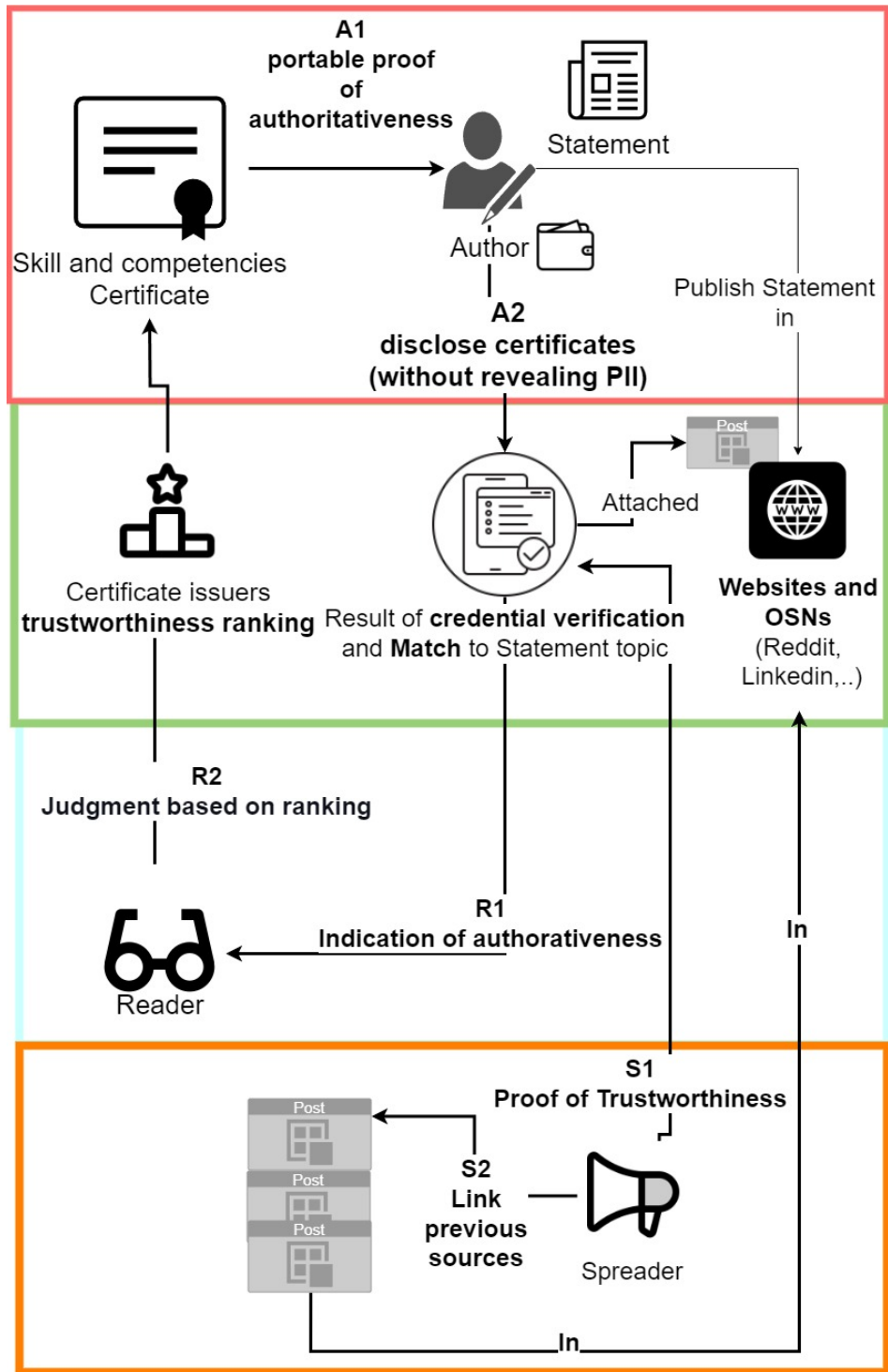


Figure 5.1: A high-level view of our systems with the involved actors. It covers all the requisite points in section 5.3.

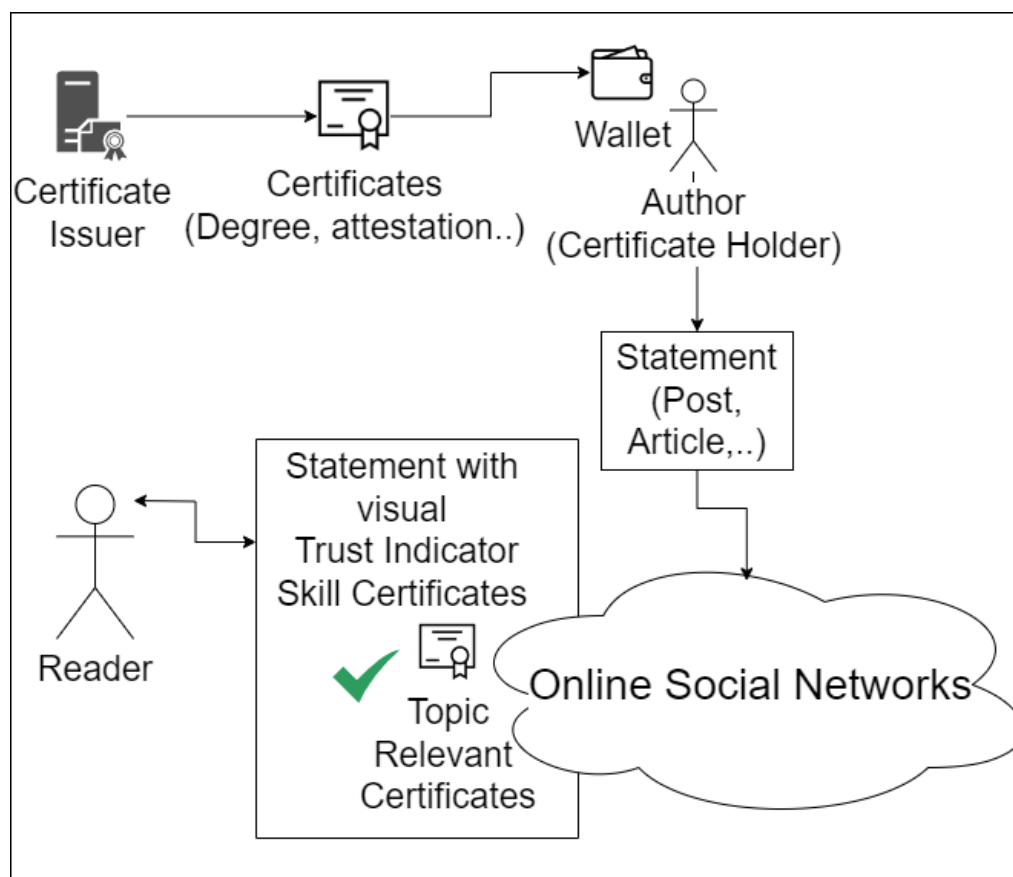


Figure 5.2: Figure showing the workflow from the Author to the Reader of a statement, highlighting the attachment of certificates issued by Certificate Issuers and the presentation of them alongside the statement

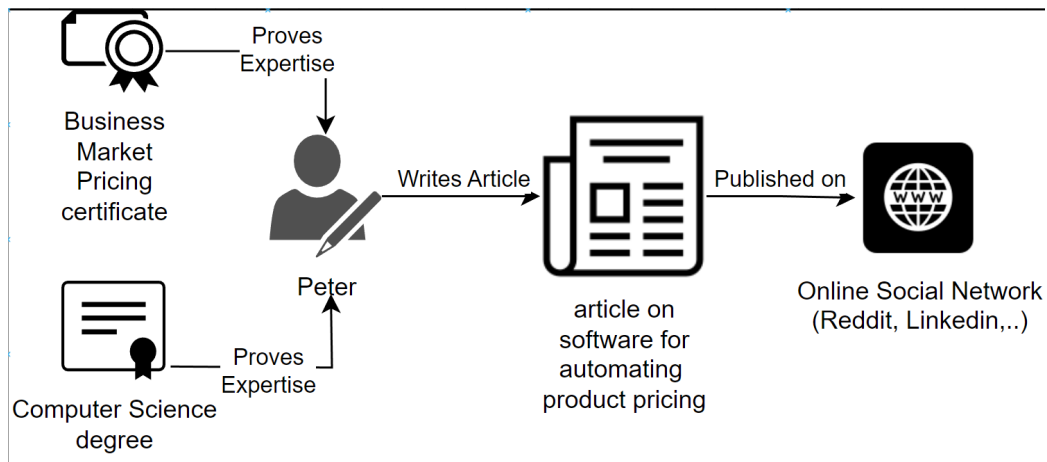


Figure 5.3: A use case for our system. Peter owns two certificates and writes an article related to it. He then publishes said article on an Online Social Network (OSN)

article (as requisite **(A2)**). Nonetheless, he would like to give weight to his words, providing their certificates (requisite **(A1)**). The situation may be seen in Figure 5.3. We see in subsection 5.5.2 how our solution solves this case.

On the other hand, we can see the point of view of Jane, an online influencer who acts as a Reader and a Spreader. Jane, as a Reader, has the following desiderata:

- She wants to know if Peter has the proper certificates that back their authority on talking about software for product pricing
- She wants to verify those certificates' validity easily without contacting the issuing institutions.
- For the institution issuing certificates, Jane intends to assess its trustworthiness based on her own information and information that may be produced by a different authority, such as her friends/colleagues, and reference points, such as our system; this contributes to assessing the trustworthiness of the statement which is requisite **(R2)**.
- Before reading the article, which can be time-consuming and re-

quire effort, she wants to be sure that the certificates provided match the content topic (stated in requisite **(R1)**). Given that, to discuss a specific topic, a writer such as Peter must possess certain skills and competence to cover that topic. For instance, a Business Market Pricing certificate provides the skill to discuss those matters.

Furthermore, Jane, as a Spreader, wants to repost an OSN post in which she links the article; alongside it, she desires to provide a way of tracing back the article from the platform/post where Jane saw it up to the original link of the article (requisite **(S2)**), in addition to also providing a link to the certificates she looked up before (requisite **(S1)**).

5.4 Match Statement and Certificates Skills

Our approach defines a way to assess the relevancy of a certificate to the statement to satisfy the requisites **(R1)** and **(S1)**. As such, our system must correctly **confirm the correspondence** between the provided Author's certificates containing **skills** with the statement contents. To do so, it must be capable of stating what certificates contain skills that match the statement's topic.

The standard approach has been to have fact checkers assess the truthfulness of a statement online. However, this approach requires human action, which is slower and costlier than automated software. On the contrary, using the AI tools described in chapter 3, we **propose the automation of the match between competencies in the certificates and the statement content**.

However, the automated process faces an issue: no standardized certificate format and credential ontology is widely accepted worldwide. Nonetheless, while it may be possible to account for certificates having multiple formats, we found a more suitable approach to defining skills, the ESCO ontology described in section 4.4. Said ontology is proposed as a standardized way for placing the skills and competencies acquired by an individual through certificates and matching them with a particular occupation. We found it also is suitable for our case to ask an Author to provide appropriate certificates for a statement.

For instance, consider the example in subsection 5.2.1, in which a

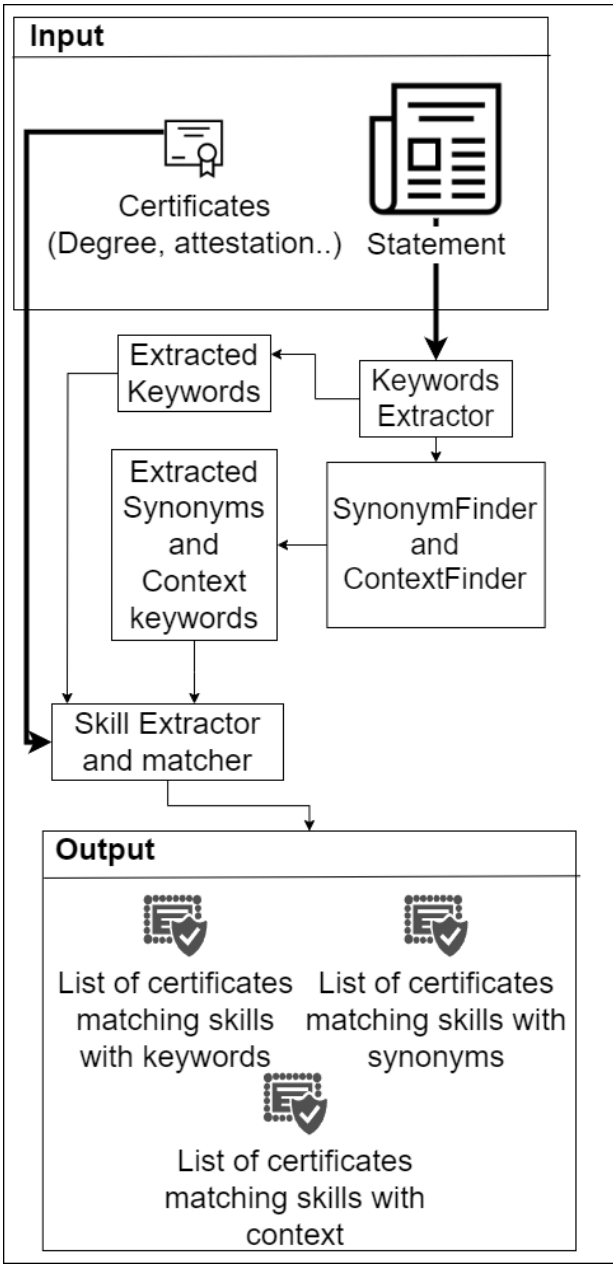


Figure 5.4: workflow of the statements versus certificates skill matching phase, from keywords to skills and matching to certificates

journalist writes a statement about a "market analysis tool." We apply the process described in Figure 5.4 to process the statement and match the skills. Notably, the pipeline starts with the extraction of keywords extraction from the statement. In our example, some extracted keywords

are Financial tools, Database, Identifying investment, Excel, and many others. Afterward, those may be passed to a software tool that may find synonyms or more general terms to refer to the elements, such as:

- Synonyms: Financial tools: fiscal instrument, monetary tool; Database: data bank, data file; Identifying investments: recognizing investments, finding investment opportunities; Excel: spreadsheet program, data analysis software
- Contexts: Financial tools: Economic entity, Financial instrument; Database: Information system, Data structure; Identifying investments: Economic activity, Investment strategy; Excel: Software application, Office productivity software

The extracted keywords, along with those additional ones, are then sent to the Skill Extractor, which **extracts the skills from the statement**, and afterward, the extracted skills are **matched to the Certificates declared skills**. For example, the Author holds and provides three certificates with ESCO Skills, with PII such as name hidden; Those certificates are:

- Certificate 1: "Use spreadsheets program", **matching with Excel keyword**
- Certificate 2: "Investment analysis" and "Social Work Theory", **where only the first matches with a keyword, the Identifying investment keyword**
- Certificate 3: Academic English, **which does not match with any skill necessary to show competency on the statement**

The output of the process will then contain only Certificate 1 and Certificate 2, which allow to satisfy the requisite **(A1)** and **(A2)** for the Author to prove their expertise without disclosing their name. For the Reader, it allows to satisfy **(R1)**, as the tool assesses the expertise of the Author with an indicator of trustworthiness being the presence of the certificates, same for **(S1)** of the Spreader.

Given this detailed process of matching statements and certificates, we introduce the concepts of SSI to create a trust verification system. This will allow the presentation of certificates to be disentangled from the Author's PII using DID (introduced in section 2.5.1) to identify the

Author instead of names and make them verifiable with VC (introduced in section 2.5.2).

5.5 Credibility Assessment and Verification System

In this section, the mappings between our system and the concepts of SSI are presented. We define the centerpiece of this architecture **CAVS**: Credibility Assessment and Verification System. This system will act as a trust anchor and is central to the trustworthiness assessment process for social content. **Key functions of CAVS:**

1. **Skill Matching:** CAVS uses the AI tools keyBert introduced in section 4.3 and Nesta's skills extractor introduced in section 4.5 to extract skills from an online statement and match them to skills contained in an author certificates VCs. Those are crucial to satisfy the following requisite from section 5.3: **(R1)**, as it is the tools and not humans that extract and match skills.
2. **Certificate Verification:** Acting as a **trust anchor**, CAVS verifies the authenticity of the VCs paired with the matched skills. This is crucial to satisfy the following requisites from section 5.3: **(R2)**, **(A1)** and **(A2)**, and **(S1)**
3. **Ranking of Certificates:** CAVS ranks the certificates based on the issuer's authority. Which allows to satisfy **(R2)**). Providing the Reader with additional information about the certificate's trustworthiness.
4. **Issue a VC to the Author :** Given the results of the previous functions, it then issues a VC to the Author of a statement, which may provide it alongside the statement, showing that CAVS certified it. **CAVS may issue a VC if it has its own DID and can have as subject also services besides persons.** This functionality is fundamental to satisfy the author requirements of **(A1)** and **(A2)**.

An alternative and more decentralized system may involve only providing a VC that certifies the skill matches with extracted skills; said way, the Reader of a post may obtain said VC and the author VCs with skills and proceed to perform the ranking of their issuers and the verification

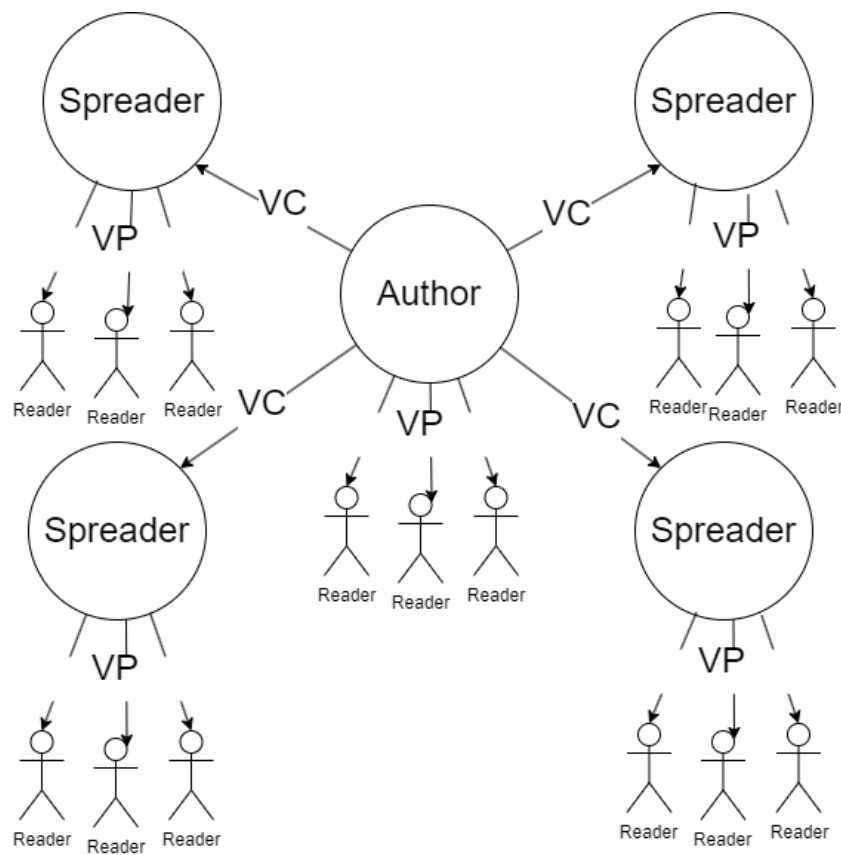


Figure 5.6: Representation of how the Author actor or Spreader actors may present the VC with a VP

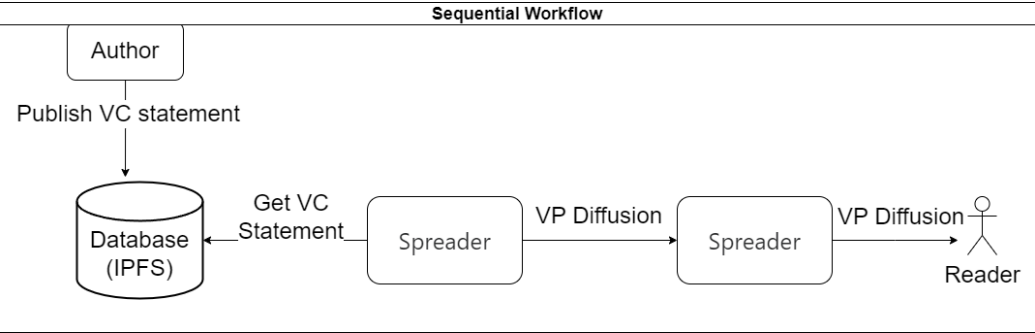


Figure 5.7: CAVS Workflow from Author actor that publishes on IPFS their VC Statement, it highlights how the Spreader actor retrieves the VC Statement and presents it with a VP Diffusion to other spreaders, up until it reaches a reader that does now spread further the statement

- **Author:** An author can be a journalist, researcher, or field expert who publishes a statement on a website (social media platform, blog, online journal). Consider that the certificates issued to authors will not include any PII, given that the certificates are issued to the DID of an author, allowing to satisfy requisite **(A2)**. In particular, it presents the statement and a set of **ESCO Skills VC** which they find related to it. It then obtains from **CAVS** a **Verifiable Credential (VC) Statement**, which includes the skills the **ESCO Skills VC** system finds related to the statement.

The Author then publishes it on IPFS as explained in 4.2, which allows it to store it in a distributed manner and to identify it with a unique CID. Using IPFS enables the exploitation of the properties of CID since the immutability of data behind a CID makes the VC Statement not tamper-proof.

- **Spreader:** The Spreader may be the Author (for example, a researcher publishing a paper), journalists spreading a fact, or any social media user. As we can see in Figure 5.7, they fetch the **Verifiable Credential (VC) Statement** and use it to create a **Verifiable Credential Diffusion** (for presentation), exploiting the concept of the Verifiable Credential Model 2.0 of holder claims to self-issue it and then present it in a Verifiable Presentation (VP) Diffusion. We delve more into it in the next section 5.5.1
- **Reader:** The Reader may verify all VP Diffusion, tracing back all of them and the VC Statement, which is retrievable via its CID from any VP Diffusion.

The portability of VCs allows them to satisfy the requirement **(A1)**. Meanwhile, their being issued to the Author DID allow them to satisfy **(A2)**.

5.5.1 Types of Diffusion: VP Diffusion of type "origin" and type "diffusion"

In our system, two main types of diffusion play a role in a statement's lifecycle. A VP Diffusion is created by the Author of a statement when it is first shared and from a Spreader when it is reshared. We use a VP to present to the Reader a signature (with their DID private key) of the sharing of a statement from both of those actors; in this way, they can

also be traced by their DID.
Note that design the VP Diffusion to hold a **VC** self issues by the

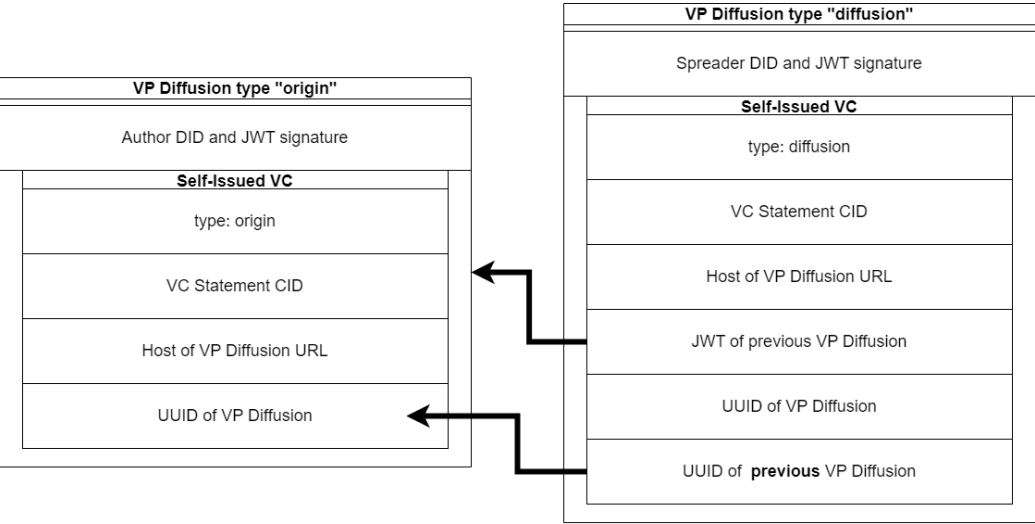


Figure 5.8: Structure of the two types of VP Diffusions, with a link from the VP Diffusion of type "diffusion" to one of type "origin." The link is in the embedded self-issued VC inside the VP Diffusion

Author/Spreader. Said *self-issued VC* leverages **holder claims** from the Verifiable Credential Model 2.0, where the issuer of a VC is the designated holder of said VC, and it issues self-claims to assert their relationship with the content. In such a way, the statement's Author can embed the VC Statement alongside other information, such as the URL that hosts said VC Statement inside the VP Diffusion, given that it presents this self-issues VC. Note that we have the self-issued VC named VC Diffusion. However, we will refer to the VP Diffusions, which we can see in their two configurations in Figure 5.8 and we detail in the following subsections.

VP Diffusion of type "origin"

The diffusion process starts with the Author of the content. The Author generates a **VP Diffusion of type "origin"** when they first publish their statement to attach to it the VC Statement. This VP Diffusion:

- Embeds the original VC Statement CID from publishing it on IPFS, which guarantees its immutability and verifiability; the VC Statement has already been issued by CAVS and linked to a set of ESCO

VC Skill that establishes the Author's credibility on the subject matter. Providing the VC Statement CID satisfies the requirement **(S1)**.

- Includes the *self-issued VC* that categorizes VP Diffusion as having type: "origin", asserting that this VP Diffusion is the initial publication of the content by the Author in the hosting website. This approach is necessary to provide an end to the chain which will satisfy requirement **(S2)**.
- It includes the website URL where the VP Diffusion will be presented, such as a blog or an OSN user profile.
- It includes a UUID of the VP Diffusion, which is useful for identifying it.

The VP Diffusion of type "origin" is the starting point for the statement's credibility chain, making the Author's original publication the root of all future interactions with the content.

VP Diffusion of type "diffusion"

When any other actor wants to assume the role of *Spreader*, given that it sees Author's VP Diffusion of type origin. The Spreader creates this VP Diffusion that embeds a *self-issued VC* with those characteristics:

- It embeds the self-issues VC that has **attribute type equal to Diffusion**
- It embeds JWT of the previous VP Diffusion they have seen before. *This is fundamental to satisfy the requirement (S2).*
- It embeds the CID to the Author's original VC Statement on IPFS. As said before, this satisfies **(S1)**.
- It includes a UUID of the VP Diffusion, as the other type.
- It includes a UUID of the previous VP Diffusion to record it in a shorter string than the JWT.

Chain of VP Diffusion

Each new **VP Diffusion** in this process includes a set of attributes within the *self-issued VC*:

- A reference to the original **VC Statement** using the CID.
- type either being origin if first in the chain or diffusion
- A reference to the previous **VP Diffusion** in the chain (with a JWT) if of type diffusion

As shown in Figure 5.7, the VP Diffusion process repeats as the statement is disseminated across platforms, with each new VP Diffusion embedding the preceding chain. This ensures traceability, as any reader can verify the authenticity of a statement by reviewing its entire diffusion chain (critical for requisite **(R2)** and **(S2)**), which can be seen in Figure 5.9. Any reader can confirm from the VP Diffusion that it saw the VP Diffusion connection to the source as they can trace it back to the VP Diffusion of type "origin" going through the VP Diffusion of type "diffusion." The VP Diffusions are chained by including the previous one in the chain JWT. Furthermore, each VP Diffusion attached to a statement has embedded the VC Statement CID on IPFS. In this way, the diffusion types ensure

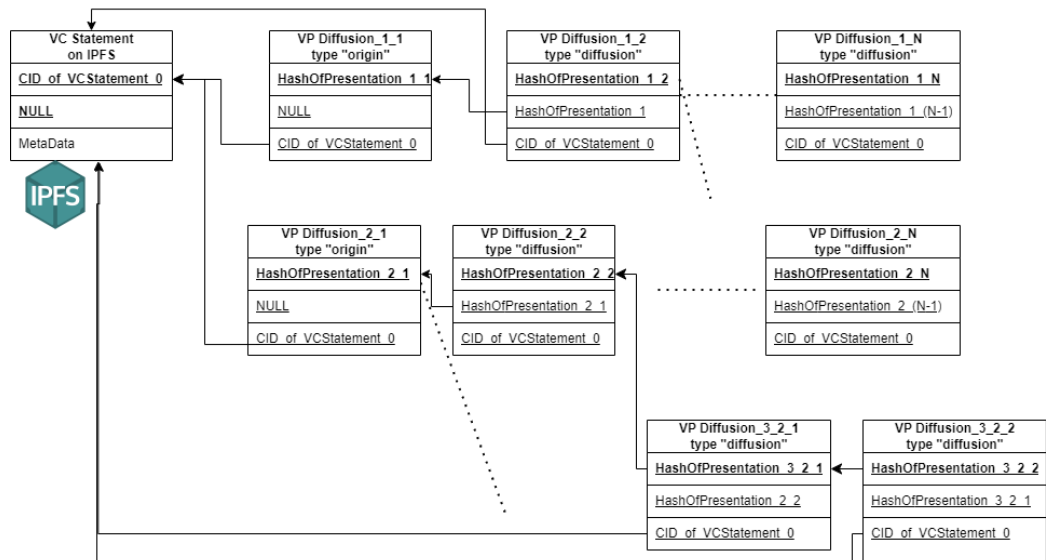


Figure 5.9: The chain of VP Diffusion, which traces back to the VC Statement.

that any actor interacting with or spreading the content contributes to a transparent and verifiable chain of credibility. Every VP Diffusion links back to the original statement and *self-issued VC* created by the Author, forming an unbroken chain that guarantees the integrity and

provenance of the shared content.

5.5.2 CAVS and use case

This subsection shows how CAVS and SSI apply to the use case described in subsection 5.3.1. In particular, we see in Figure 5.10 the issuing of certificates with a digital certificate that is an ESCO VC Skill. We can see the skills are included inside the ESCO VC Skill themselves; the skills are described by their name and URI.

Given the ESCO VC Skill credentials stored in Peter’s wallet, they then

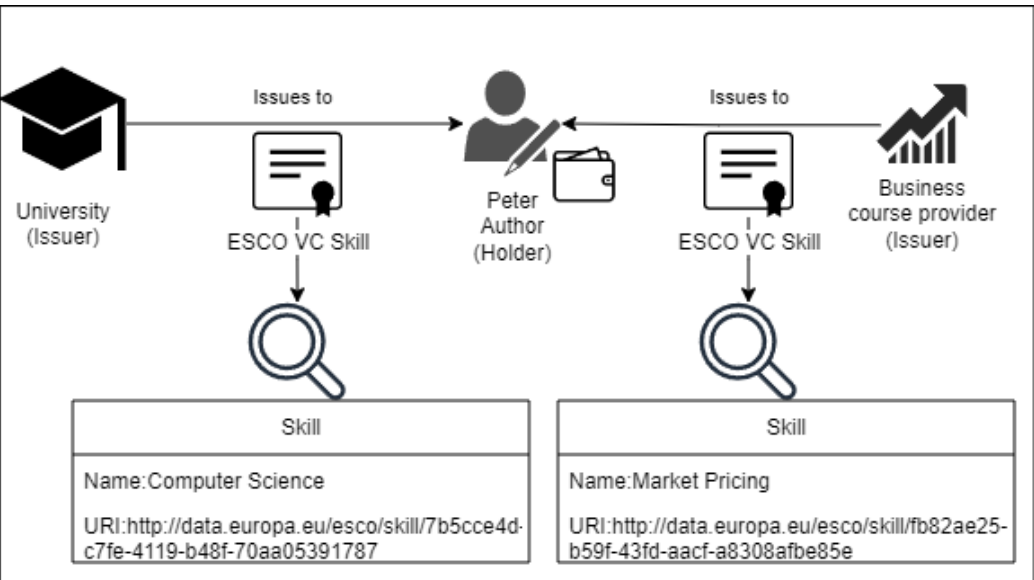


Figure 5.10: The issuing process from two issuers, a University and Business course provider, to an author of statements, Peter (holder), of ESCO VC Skill credentials.

present them to CAVS along with a statement (an article) on which they want to prove their authoritativeness on the statement topic. As seen in Figure 5.11, CAVS produces a VC Statement that Peter can present; this can be seen in Figure 5.12. Suppose CAVS successfully maps the keywords of the statement to the skills provided in ESCO VC Skill credentials, which are verified to be true. In that case, those skills will be included in the VC Statement alongside their issuer’s CAVS rank from 0 to 10 of trust in their issuers. Embedding the CAVS rank in the VC Statement allows satisfying the requisite **(R2)** of the Reader to make

an informed judgment about a statement, given that the trust in the Certificate Issuer is a significant factor in the quality of the statement content. Additionally, the list of keywords extracted from the statement and their synonyms will be in the VC Statement.

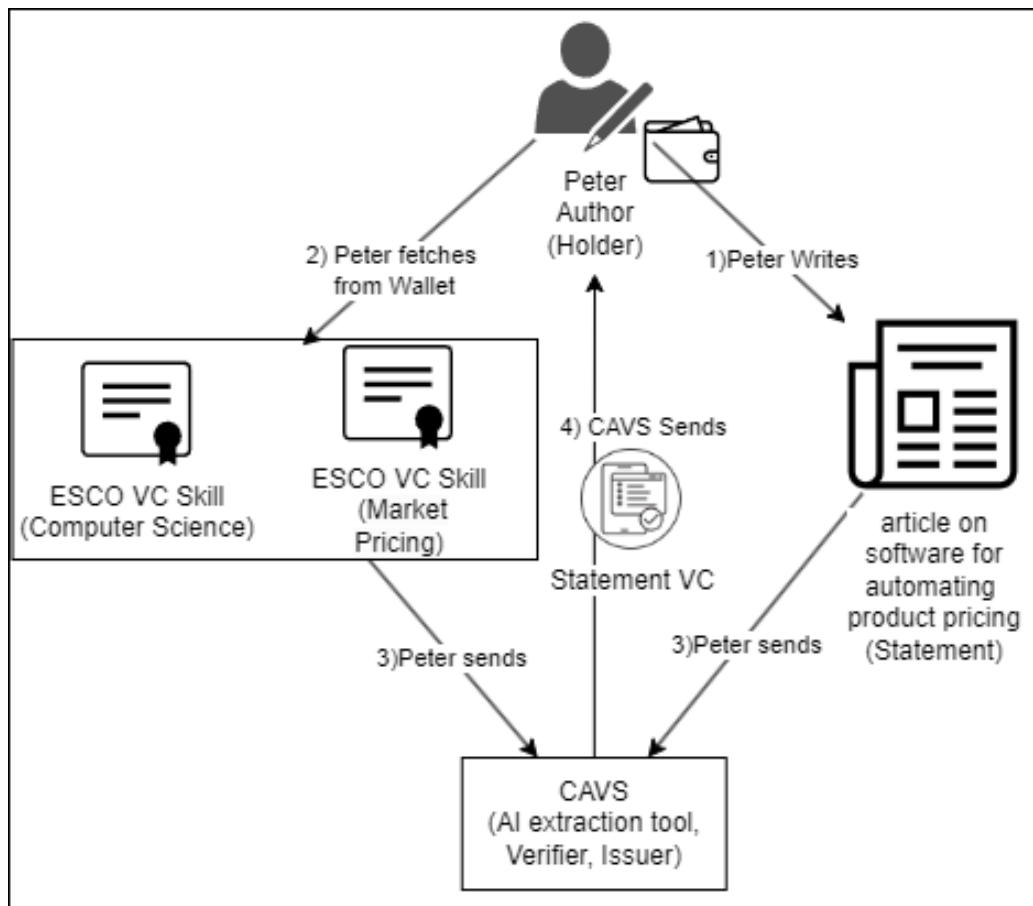


Figure 5.11: The process of creating a VC Statement for Peter, the Author of a statement, which provides two ESCO VC Skills. Afterward, CAVS produces a VC Statement and sends it to Peter.

Given that the ESCO VC Skill credentials are in the VC Statement and alongside the AI tools checksum, Jane is acting as a Reader. First, she verifies the VC Statement. They may first execute locally the AI tools matching the checksum; afterward, if the ESCO VC Skill corresponds to the skills extracted, they may verify the ESCO VC Skill if she does not trust the CAVS verification process and want to be double-sure

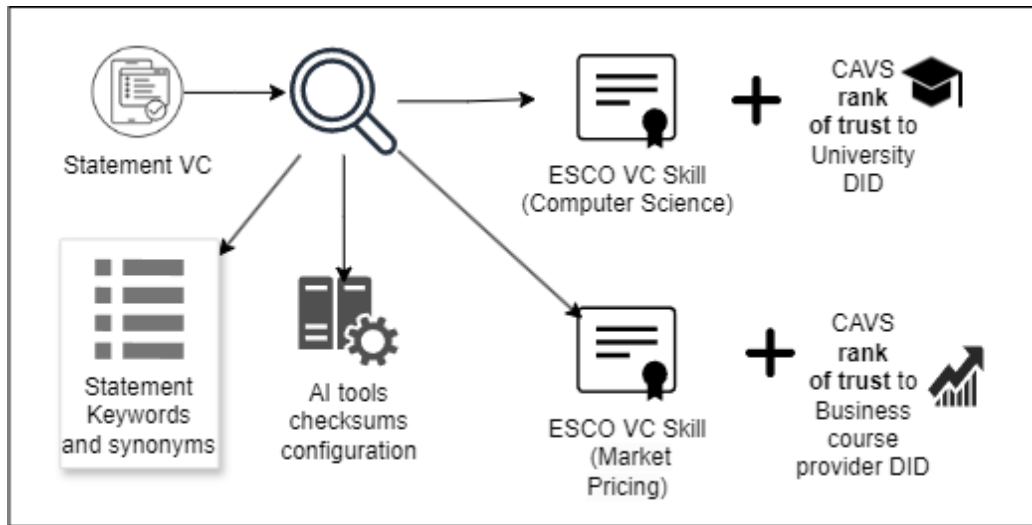


Figure 5.12: The content of Peter's VC Statement, composed of two ESCO VC statements with the ranking that CAVS places in their issuers and the CAVS AI tools' keywords extracted. Additionally, checksums of the AI model extracting keywords and mapping to skills are present

(important for requisite **(R1)**, allowing the Author to know that the indication of trust, being the list of certificates with skills, is correct and to replicate it).

Furthermore, Jane can create a VP Diffusion of type diffusion. With it, they reference the origin of the VP Diffusion from Peter and the VC Statement, allowing any reader to verify all the VP Diffusions that create a chain. We allowed readers to reach each VP Diffusion's VC Statement on IPFS. Furthermore, tracing back to VP Diffusion and verifying all VPs in the chain allows for a verified chain of statement diffusion.

Chapter 6

CAVS implementation

The code illustrated in the following chapter is available on GitHub at the repository:caltr98/CAVS ¹¹ (with execution instructions). Throughout the chapter, we will refer to the requisites introduced in section 5.3.

We implemented CAVS using a microservice architecture. The idea was to create a solution with many pieces that someone could chain and interchange with its alternative, given that it can interact with their REST interface.

The components run on Docker containers; the "Docker compose" command connects them automatically and creates a network for the CAVS whole system. The BERT model extracts keywords from each statement, using KeyBERT described in section 4.3. A microservice querying one model or the other is defined, where the keyBERT microservice runs on the local machine, while the microservice with GPT-3.5 queries the OpenAI APIs described in section 4.6 .

It may be possible to enrich the keywords by looking at related keywords or the general field of the keywords, and the author actor can do this when providing a statement to CAVS, which will call an internal microservice connected to the Yago knowledge base or another microservice using GPT-3.5, this latter task if performed in the keyLLM microservice.

¹¹<https://github.com/caltr98/CAVS>

For the keywords mapping to skills, we have a microservice running `ojd_daps_skills` model, which we described in section 4.5, which, as stated, allows us to extract ESCO ontology-defined skills from the text.

CAVS uses a microservice with a Veramo Agent with its own DID. DID concept and their usefulness (required for requisite **(A2)**) were stated in section 2.5.1, while Veramo and the concept of Veramo agent in section 4.1. The Veramo Agent microservice enables those tasks for different actors: **1.** Create DID with key-pairs, **2.** Import DID, **3.** Issue VC, **4.** Verify VC/VP, **5.** Store VC, **6.** Create VP.

In particular, the CAVS system allows the verification of the ESCO Skills VC and then the creation of a VC Statement. This allows to fulfill the requisite of sec 5.3 of portable VCs for requisite **(A1)** and indication of credibility for **(R1)**.

For the author, it allows the storing of the VCs mentioned above and the creation of VPs Diffusion, the latter also for Spreaders.

For readers, it allows to verify the VPs and the VC Statement to which the VPs Diffusion refers (satisfying requisite **(R2)**).

6.0.1 Architecture details

We may see the system's architecture in Figure 6.1. All the microservices besides CAVS are stateless, allowing them to fail, and a daemon process or a person can restart them without issues. The Client Webapp can change the state of the CAVS microservice, which includes:

- the selected engine for keyword extraction between keyBERT and its subconfigurations. In the microservice `keyBertService`, only the "all-MiniLM-L6-v2" BERT model is used for skill extraction, which is set up to extract at most five keywords of one token length and five keywords of two token lengths. Meanwhile, in `keyLLMService`, two configurations can be called in requests, including a mixture of BERT and GPT-3.5 and another only calling GPT-3.5 in `keyLLMService`. Both microservices use KeyBert library, **they functionalities are not in the same microservice**, but separated in two microservices, because we expect `KeyBertService` to be the most widely used as it just runs on the local machine, without costs incurred in calling OpenAI APIs. Given this, we created the `KeyBert` microservice that only runs the local BERT model, keeping the microservice

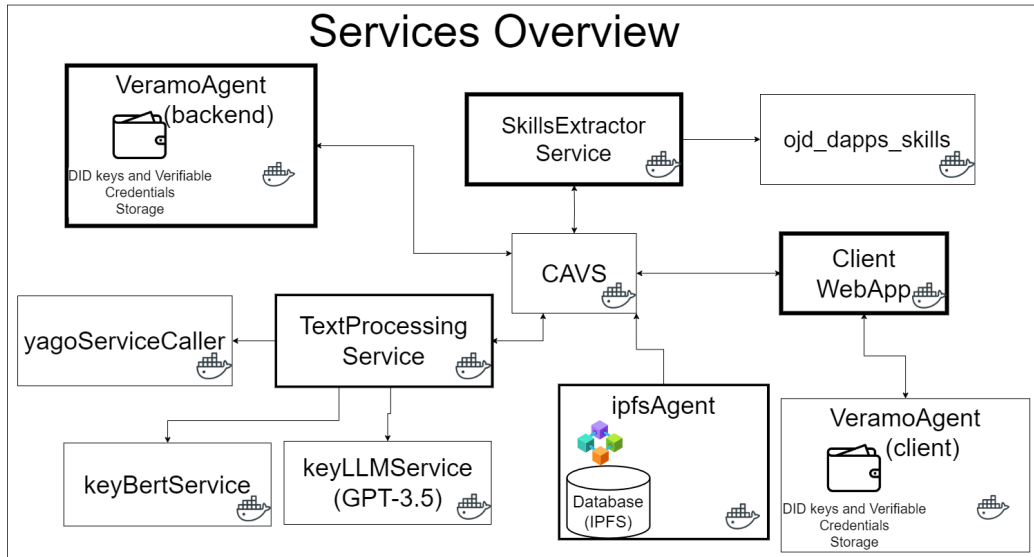


Figure 6.1: An overview of the microservices we created for the CAVS system and their interconnection. In thick lines, the microservices are directly connected to the CAVS microservice

lightweight. Changing the engine will change the REST API calls that CAVS performs to TextProcessingService and, in turn, the REST API calls it performs and on which endpoint.

- the selected engine for skill extraction, which is currently only `ojd_dapps_skills`. Changing the engine will change the REST API calls that CAVS performs to SkillExtractorService

Since `ojd_dapps_skills` and `keyBert` are models with libraries written in Python, the microservices enclosing them were written in Python. The microservice interfacing with OpenAI GPT-3.5 is also written in Python, standardizing the interaction with AI tools via Python. We use Python's Flask framework to serve HTTP requests for said microservices.

6.0.2 Trust in AI Models

To allow the verification of results from TextProcessingService and SkillExtractorService, we include a checksum of the trained base models used by the tools inside the VC Statement. This is fundamental to satisfy requisite **(R2)**, as the reader may run the tools themselves to make their

judgments. In particular:

- **keyBert**: we include the all-MiniLM-L6-v2 hash of the model state dictionary with trained parameters (weights and biases) and architecture of the NN model. The checksum is generated with sha256.
- **keyLLM**: we provide the checksum above for extraction with the BERT model in conjunction with the LLM model version, which in our case will be a version of OpenAI GPTs available API callable models. Furthermore, if keyLLM is used to enrich keywords, a link to the prompt, with the GPT version used and the parameters for each call, will be provided.
- **ojd_daps_skill**: for this, we make two checksums for two models used by the tool, one as before for the BERT model used for the mapping skills extracted in the toy taxonomy to a reference taxonomy and the other for the extraction of the skill to the toy taxonomy for the NER model. For the NER model, we hash the config.cfg file.

6.0.3 Testing AI tools

Note that the web app called TextWebApp at localhost:13001 (when executing the command Docker compose) allows testing all the AI text processing tools independently from the SSI concepts in CAVS. The results from its usage can be seen in section 4.5.2. Furthermore, we may see the interface of the web app in Figure 6.2 and Figure 6.3

6.1 Webapp and CAVS in detail

We designed a web app that serves as a client for the microservices illustrated.

It allows the generalization of a DID for the Sepolia testnet, a testnet of Ethereum, but also to import a DID of the DID method ETH mentioned in section 2.5.1 by importing it in the Wallet address.

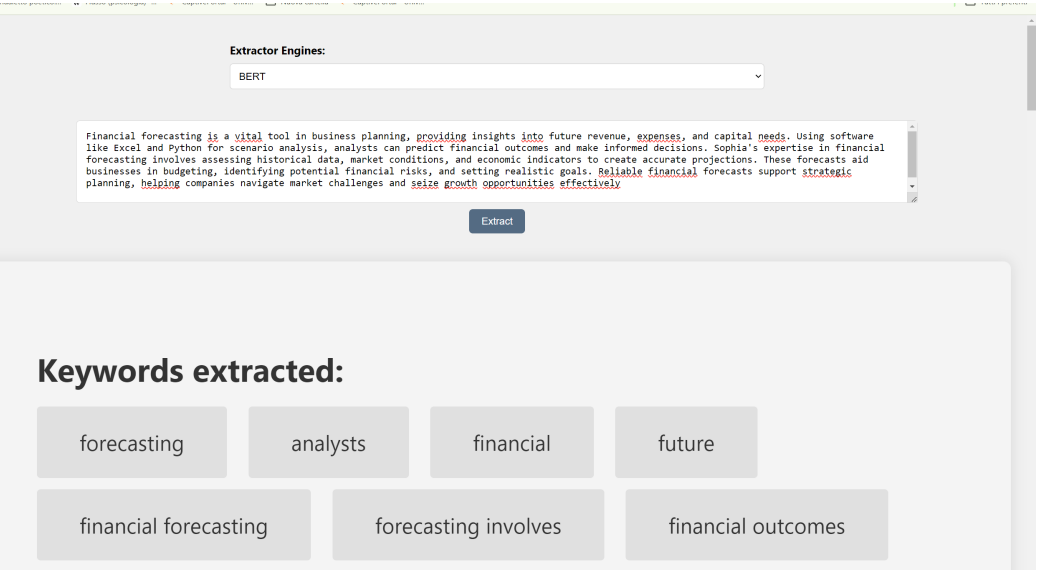


Figure 6.2: The TextWebApp section that allows to extract keywords from text

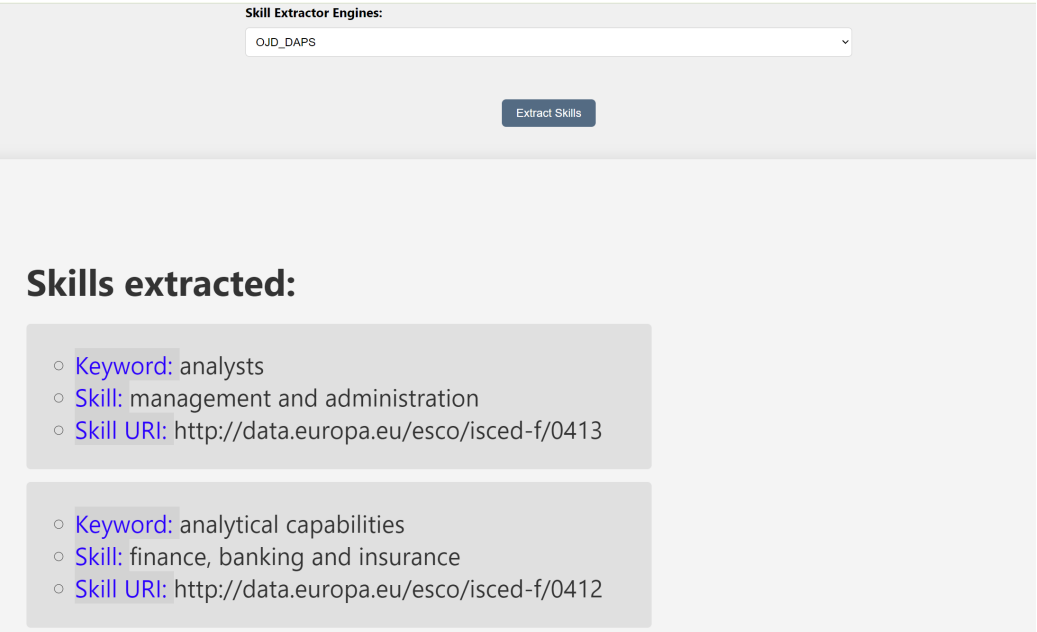


Figure 6.3: The TextWebApp section that allows to map keywords to skills

6.1.1 Certificate Issuer section

Figure 6.4 shows the dashboard for the CertificateIssuer, which grants ESCO Skills VCs, with the skills extracted from a text or added manually. The field is composed of:

- New Holder DID: the DID of the Holder to be assigned the certificates
- New Holder Endpoint DID: the endpoint URL to send the ESCO Skill VC after it is created; it is the VeramoAgent microservice on the client side.
- Certificate Description: a description of the certificate to assign, allowing the extraction of ESCO Skills from said description using the CAVS microservice.
- Not seen in Figure, there are fields to input the ESCO Skill manually, with their name and URI.

6.1.2 CAVS dashboard

The CAVS dashboard is intended to be its web app, which the system administrator uses. We included it in the whole web app to showcase the system features. In this section, we may set up the CAVS microservice backend with the engines for VC Statement, keyword extraction, synonyms generator with general field extraction, and skills extraction. As shown in the figure, we can choose the configuration and see the current one 6.5. The possible choices are:

- Extractor Engines: BERT, BERT+LLM, LLM. Where LLM refers to GPT-3.5, BERT refers to the microservice keyBertService; BERT+LLM refers to the microservice on keyLLMService.
- Enricher Engines: Yago and GPT, two microservices calling the online APIs of each tool.
- Skill Extractor Engines: OJD_DAPS_SKILL (Nesta's skill extractor) microservice serving requests to said tool.

The screenshot displays a web application interface for issuing a verifiable credential. It features several input fields and a large text area, all set against a light gray background.

- Issue Certificate Verifiable Credential:** This section contains two input fields:
 - New Holder DID:** A text input field.
 - New Holder Endpoint URL:** A text input field.
- Certificate Description To Skills:** A large text area with the placeholder text "Enter Certificate Description...".
- Keyword Extractor Engines:** A dropdown menu currently showing "BERT".
- Skill Extractor Engines:** A dropdown menu currently showing "OJD_DAPS", followed by a blue button labeled "Extract Skills from Description".

Figure 6.4: The section of the Web app for CertificateIssuer

CAVS available at <http://localhost:4200>

Extractor Engines:

BERT

Enricher Engines:

YAGO

Skill Extractor Engines:

OJD_DAPS

Update Config

Current Setup: BERT, NONE, OJD_DAPS

Figure 6.5: The section of the Web app for CAVS setup

Trust Certificate Issuer

Given that not all Certificate Issuers are to be trusted as authoritative in issuing a certificate, we provide a trust score to assign as a solution to them. The system administrator can review a Certificate Issuer and may input their DID and a score from 1 to 5 assigned to it, which the CAVS system will consider during VC Statement creation.

We may see in the figure the interface that allows this task 6.6

Certificate Issuer DID:

did:ethr:sepolia:0x03d3a2ba9a3b5378f4012166e7fc706038d7cfc5ddb2adb253f4cc15c9441809b6

Trust Score [1-5]:

3

Rank Issuer

Ready to Add

Figure 6.6: The section of the Web app for adding the DID of Certificate Issuer and their ranking

6.1.3 Author section

Fetch ESCO Skills Verifiable Credential

Select Credentials for VC

✓

Credential #1

Hash: QmR6Apps9KPdnKQEIMx81gcgRndY8VFQNhZDzQsuXuB6Az

Issuance Date: 2024-08-17T13:49:49.000Z

Credential Type: VerifiableCredential, ESCO_type_VerifiableCredential

Issuer DID: did:ethr:sepolia:0x03d3a2ba9a3b5378f4012166e7fc706038d7cfc5ddb2adb253f4cc15c9441809b6

Holder DID: did:ethr:0x877545E3910550Ce27c2c51Bd2FF14837acB6566

```
{
  "skills": [
    {
      "mechanical vehicle components",
      "http://data.europa.eu/esco/skill/575f79e8-8500-4ddf-91ec-5e6de73d35a3"
    }
  ],
}
```

Selected Credentials: 1

Figure 6.7: The section of the Web app that the author uses to provide the statement to present to CAVS

The author of a statement using the functionalities of CAVS may obtain a VC Statement. To request one, the author presents the ESCO VC Skill

Statement Category(Economics, Medicine, Science..)

Impact of 3D Printing on Mechanical Prototyping

Statement Host URL

<https://manufacturingtoday.com/3d-printing-mechanical-prototyping>

Statement Type:

Article

Statement:

3D printing, or additive manufacturing, is transforming the way mechanical prototypes are developed. This technology allows for the rapid production of complex parts and components, reducing lead times and costs. Danielu2019s experience with 3D printing technology enables the creation of accurate and functional prototypes, facilitating iterative testing and refinement. Understanding materials science is key, as the choice of materials impacts the performance and durability of printed parts. The adoption of 3D printing in mechanical engineering fosters innovation, enabling the production of custom and intricate designs that were previously difficult or impossible to manufacture.

Send to CAVS

Ready to Request

Figure 6.8: The section of the Web app that the author uses to see their ESCO VC Skill and choose to present to CAVS

to CAVS. This can be seen in Figure 6.7, then the author provides the statement, the URL of where the statement is hosted, and the statement category, which can be seen in Figure 6.8. We may know the structure of the VC Statement created after the request of the previous figure in the JSON 6.1. It can be noticed how there are the keywords extracted, and the ESCO VC Skill with the skills that matched said keywords. The ESCO VC Skills are placed in the following categories:

- **Direct Keyword Match:** inside **credentials_for_skills**
- **Synonyms Match:** inside **credentials_for_similar_concepts_-_skills**
- **General Field Match:** inside **credentials_for_general_concepts_-_skills**

The latter two categories are optional and only used if CAVS is set up to obtain the synonyms and extract the general field. Each category has an array of ordered rankings of the issuers based on the issuers' provided DID rankings.

Listing 6.1: VC Statement containing information about the configuration and keywords used for skills verification

```

1 {
2   "cavs_config": "BERT+NONE+OJD_DAPS",
3   "keywords": ["manufacturing", "3d", "mechanical", "printing", "materials", "prototypes",
4     "technology", "engineering", "designs", "3d printing", "additive manufacturing", "
    manufacturing is ", "understanding materials", "of 3d", "prototypes are ", "
    intricate designs", "engineering", "way mechanical"],
5   "keywords_model": {

```

```

5     "model_checksum": "16c9297ae82f8565b5369459db1b7f310e719798d1d13145d349125ca58cf84d",
6     "name": "all-MiniLM-L6-v2"
7 },
8     "skills_model": {
9         "OJD_DAPS_BERMODELCHECKSUM": "16
10             c9297ae82f8565b5369459db1b7f310e719798d1d13145d349125ca58cf84d",
11         "OJD_DAPS_BERTMODELNAME": "sentence-transformers/all-MiniLM-L6-v2",
12         "OJD_DAPS_NERMODELCHECKSUM": "398
13             f0be769f8b1e017bed2e93372ca2a7b74dddb66f27063d0cf5de06c7a9fe",
14         "OJD_DAPS_NERMODELNAME": "spacy_ner_model"
15     },
16     "similar_concepts_keywords": [],
17     "general_concepts_keywords": [],
18     "statementType": "News",
19     "statementCategory": "Mechanics",
20     "statementHostURL": "mechanicsonlinewebsite.com",
21     "credentials_for_skills": [
22         {
23             "issuanceDate": "2024-08-23T16:42:20.000Z",
24             "@context": ["https://www.w3.org/2018/credentials/v1", "https://www.w3.org/ns/
25                 credentials/v2"],
26             "type": ["VerifiableCredential", "ESCO_type_VerifiableCredential"],
27             "credentialSubject": {
28                 "skills": [
29                     ["engineering", "http://data.europa.eu/esco/skill/b8ac179c-4a29-4467-9177-031daf1321d6"],
30                     ["CAD software", "http://data.europa.eu/esco/skill/a2b5dcf3-5b6a-453d-876c-cff540c0faf1"],
31                     ["a manufacturing process", "http://data.europa.eu/esco/skill/3786b61f-f22e-48d1-af8d-ad4c354534db"],
32                     ["materials (glass, paper, plastic and wood)", "http://data.europa.eu/esco/iscde-f/0722"],
33                     ["mechanics and metal trades", "http://data.europa.eu/esco/iscde-f/0715"],
34                     ["work skills", "http://data.europa.eu/esco/iscde-f/0417"],
35                     ["management and administration", "http://data.europa.eu/esco/iscde-f/0413"],
36                     ["Prototyping development", "http://data.europa.eu/esco/skill/ed1d8bd4-cd2a-4c64-b665-56d70651026c"],
37                     ["mechanics", "http://data.europa.eu/esco/skill/efa141df-f382-418f-9121-bd88fd735669"],
38                     ["software and applications development and analysis", "http://data.europa.eu/esco/iscde-f/0613"],
39                     ["material mechanics", "http://data.europa.eu/esco/skill/2964011e-f828-4c6c-b98b-51e984d66365"],
40                     ["engineering", "http://data.europa.eu/esco/skill/b8ac179c-4a29-4467-9177-031daf1321d6"],
41                     ["business, administration and law not elsewhere classified", "http://data.europa.eu/esco/iscde-f/049"],
42                     ["engineering and engineering trades", "http://data.europa.eu/esco/iscde-f/071"],
43                     ["industrial engineering applications", "http://data.europa.eu/esco/skill/a1816dfd-dbd4-4499-bef3-ff81dd895b33"],
44                     ["CAD software", "http://data.europa.eu/esco/skill/a2b5dcf3-5b6a-453d-876c-cff540c0faf1"],
45                     ["information and communication technologies ( icts)", "http://data.europa.eu/esco/iscde-f/061"],
46                     ["materials science", "http://data.europa.eu/esco/skill/142f3f7f-f15f-412e-a5fe-f75755b5dbe0"],
47                     ["Prototyping development", "http://data.europa.eu/esco/skill/ed1d8bd4-cd2a-4c64-b665-56d70651026c"],
48                     ["the science of materials", "http://data.europa.eu/esco/skill/142f3f7f-f15f-412e-a5fe-f75755b5dbe0"],
49                     ["engineering", "http://data.europa.eu/esco/skill/b8ac179c-4a29-4467-9177-031daf1321d6"]],
50                 "id": "did:ethr:0x90b1041EEfea68Ea564574f6C576bD3c902A8f32"
51             },
52             "issuer": {
53                 "id": "did:ethr:sepolia:0x036f320763e5cb6006ee0d272a6ec80be654bb70dde26d8717ef1bbc6566002728"
54             },
55             "proof": {
56                 "type": "JwtProof2020",
57                 "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eX0iOiJKV1QifQ..."
58             }
59         }
60     ]
61 }

```

```

35     }
36   ],
37   "credentials_for_similar_concepts_skills": [],
38   "credentials_for_general_concepts_skills": [],
39   "ranks_for_skills": [0],
40   "ranks_for_similar_concepts_skills": [],
41   "ranks_for_general_concepts_skills": [],
42   "id": "did:ethr:0x90b1041EEfea68Ea564574f6C576bD3c902A8f32"
43 }

```

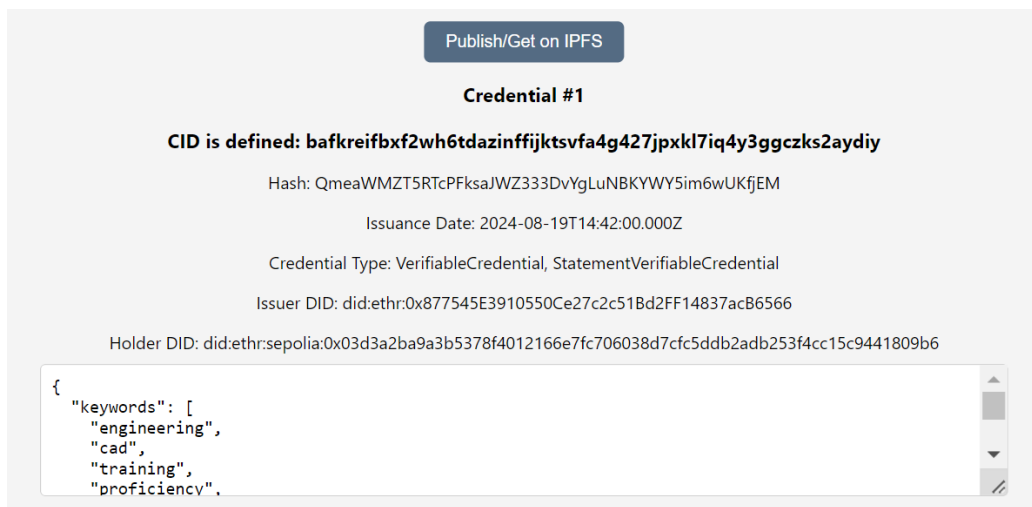


Figure 6.9: The section of the Web app that the author uses to publish a VC Statement on IPFS or obtain its CID if it has already been published

After obtaining the VC Statement, an author may publish it to IPFS; as shown in Figure 6.9, the CID is obtained after publishing it. This is fundamental to satisfy the requisite (**S1**) for the Spreader to have a way of proving the trustworthiness of their sources.

Using the CID of the VC Statement, the author can create a VP Diffusion, which the system requires to create and distribute alongside the statement. In this case, the VP Diffusion has the type "origin," as it's the start of the chain of VP Diffusion directing to the VC Statement. We may see a VC Diffusion of type origin created in Figure 6.10.

6.1.4 Spreader section and VP Diffusion

A Spreader reshapes an original statement or a previous resharing of it. To do so, it uses the JWT of the VP Diffusion, which it found last in the chain of resharing, and uses it to create their VP Diffusion of type

Figure 6.10: The section of the Web app that the author uses to create a VP Diffusion of type origin

We may see in Figure 6.11 how it does so, providing the JWT, the CID of the VP Diffusion, and the UUID of the previous VP Diffusion. The UUID is integrated to provide unique identifiers to VP Diffusions in case they are published elsewhere rather than IPFS and so without a unique CID. For example, VP Diffusions may be hosted on web pages as QR codes or plain JSON rather than published on IPFS. We may see an example of VP Diffusion of type diffusion in the listing 6.1.4

```

1  {
2    "verifiablePresentation" :{
3      "verifiableCredential" :[
4        {
5          "credentialSubject" :{
6            "statement_vc_cid_or_url":
7              "bafkreihi4lfunkhwcw4x5xpxk7u4k3sifc6nvvgqtxysvfzfzfyavnlbn4",
8            "prev_vp_uuid": "00c86dfe-b1d2-496d-b446-030f751e078e",
9            "url_or_cid_jwt_prev": "bafkreifdavmkmqgzdbdzembynm6rb4dpvos2yf267m5mzqjitzpwnr7ru",

```

Figure 6.11: The section of the Web app that Spreader uses to create a VP Diffuusion of type diffusion

```

9         "type": "diffusion",
10        "host_URL": "facebook.com/post/...",
11        "assertion": "This VP is submitted by the subject as evidence of VC
        propagation",
12        "id": "46ea4b57-698e-4d40-8fc7-f5c834d2124d"
13    },
14    "issuer" :{
15        "id": "did:ethr:0x877545E3910550Ce27c2c51Bd2FF14837acB6566"
16    },
17    "type" :[
18        "VerifiableCredential",
19        "VCPresentation",
20        "VCforStatementDiffusion_VPv01"
21    ],
22    "@context":[
23        "https://www.w3.org/2018/credentials/v1",
24        "https://www.w3.org/ns/credentials/v2"
25    ],
26    "issuanceDate": "2024-08-17T15:09:15.000Z",
27    "proof" :{
28        "type": "JwtProof2020",
29        "jwt": "eyJ...."
30    }
31 }
32 ],
33 "holder": "did:ethr:0x877545E3910550Ce27c2c51Bd2FF14837acB6566",
34 "id": "46ea4b57-698e-4d40-8fc7-f5c834d2124d",
35 "type" :[
36     "VerifiablePresentation",
37     "StatementDiffusion_VPv01"
38 ],
39 "@context":[
40     "https://www.w3.org/2018/credentials/v1",
41     "https://www.w3.org/ns/credentials/v2"
42 ],
43 "issuanceDate": "2024-08-17T15:09:16.000Z",
44 "proof" :{
45     "type": "JwtProof2020",
46     "jwt": "eyJhb...."
47 }
48 }
49 }

```

6.1.5 VCs and QRCode

Given that the VCs and VPs are JSONS with a JWT, that is the signature made by the issuer in case of a VC or the VP creator. That signature is a hash of the VCs/VPs, which can be reverted to the plain-text version. From the JWT, creating a QR code for them is possible. For example, we may create the QRcode of a VP Diffusion and place it on a webpage alongside a statement. A spreader or an author may place a QRCode on a webpage. We see the section dedicated to the QRCODE creation a

QR CODE generated for a VP Diffusion in Figure 6.12



Figure 6.12: A QR CODE of a VP Diffusion

6.1.6 Reader section

The dashboard for the reader, which can be seen in Figure 6.13, allows it to:

- decode a VP Diffusion from a JWT, possible since its signature is embedding proof that can be reversed to the plain-text VP. Details on embedding proofs are in subsection 2.5.2.
- decode a VP Diffusion from the image of a QR CODE generated from a JWT, said JWT(

Then, after having the VP Diffusion decoded, the reader may:

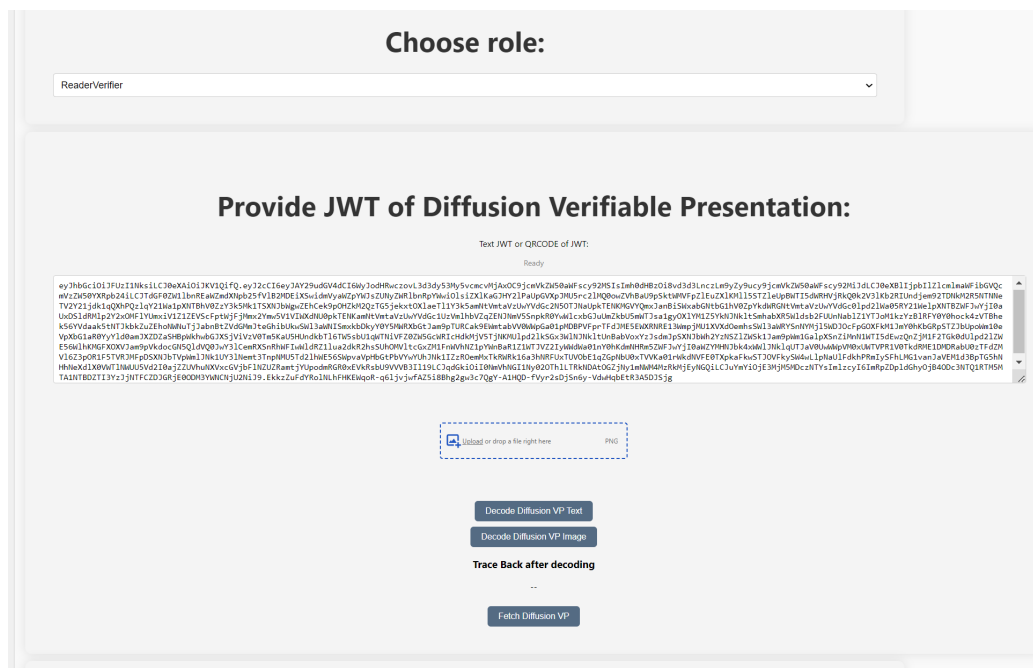


Figure 6.13: Reader section showing how they may decode a VP Diffusion from a QRCODE or a JWT

- Trace back the VP Diffusion, following the URL or CID chain of VP Diffusions up to the VP Diffusion of type origin
- Fetch the VC Statement generated by CAVS, which includes the keywords extracted, the skills, and the ranking of the ESCO Skill VC given by the Certificate Issuer to the statement author. This can be seen in Figure 6.15.
- Verify all the VP Diffusions, VC Statement recovered, and the ESCO Skill VC inside the VC Statement, as can be seen in Figure 6.15

The verification after having traced back VPDiffusions can be seen in Figure 6.14

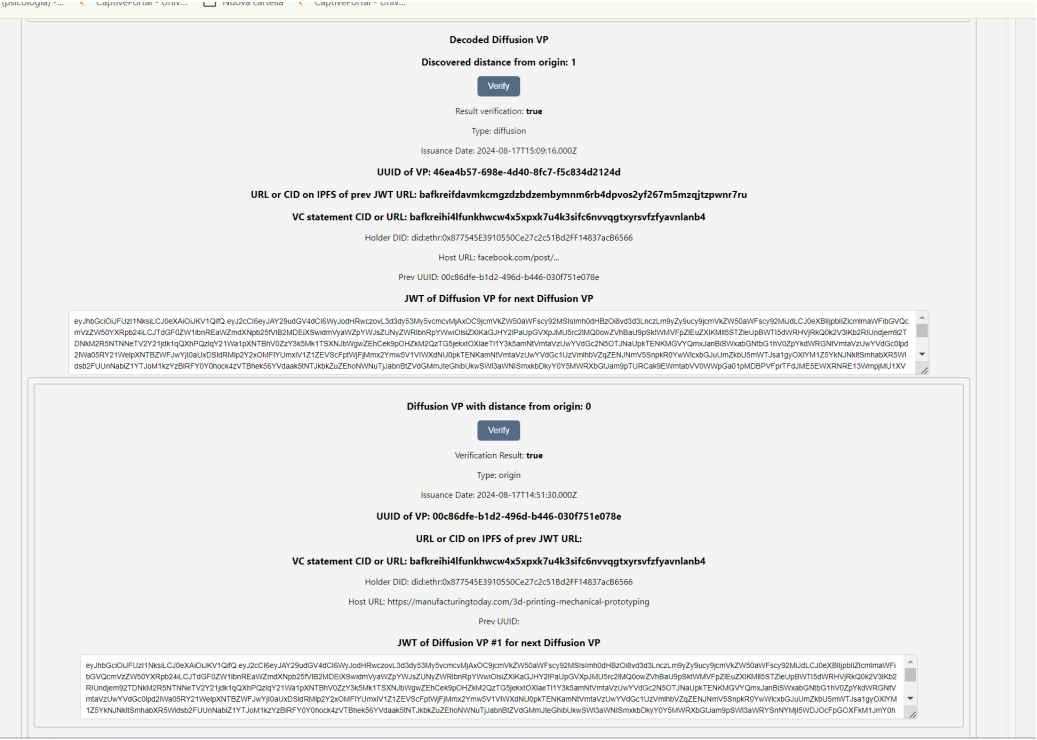


Figure 6.14: the traceback outcome of VP Diffusion and the result of their verification

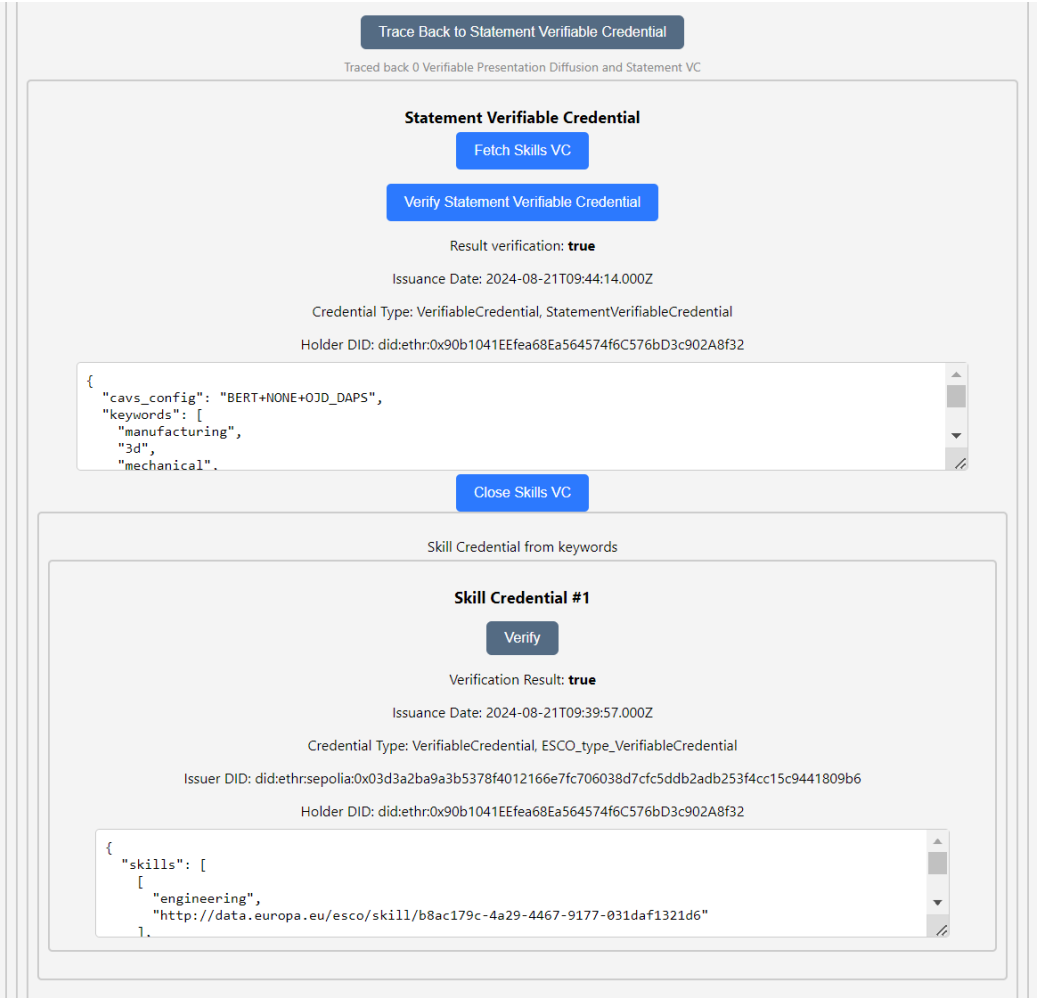


Figure 6.15: A fetched VC Statement with one ESCO Skill VC that was provided to CAVS for its creation

Chapter 7

Use Case on Real Online Social Network

We integrated the CAVS system into an existing OSN to validate the proposed concept and to assess the challenge of incorporating it into a real-world environment.

We evaluated different OSNs for integration, ranging from traditional ones, such as Facebook, to more decentralized and novel ones, such as Mastodon and blockchain-based ones. Furthermore, we highlight that blockchain-based Social media face trouble with the problem of bot accounts, given that they give rewards for content. In particular, there is the identity checking problem, which, given the openness of the blockchains as a ledger, leads to bots freely registering and performing malicious actions [61] to get rewards. We propose CAVS as a solution to stop the spread of posts generated by said bots.

The blockchain-based platforms considered for the integration Mirror.xyz¹², Steemit.com¹³ and Warpcast.com¹⁴. In particular, the social network Minds.com¹⁵ was also considered because it pairs a DID to each profile. Unfortunately, integrating the CAVS system within Minds.com was impossible because, although OSN generates a DID for each account, such a DID is entirely managed by the OSN, and even the

¹²<https://mirror.xyz>

¹³<https://steemit.com>

¹⁴<https://warpcast.com/>

account owner cannot access the corresponding private key. Additionally, the source code is closed-source, necessitating a browser extension to work with it.

Ultimately, the chosen OSN was DeSo¹⁶; the reason for choosing it is explained in the following section.

7.1 DeSo

We chose DeSo because it allows each of its accounts to be mapped to an Ethereum wallet address. This means we can link each DeSo account to a DID using the DID method mentioned in section 2.1.

DeSo is an OSN supported by its blockchain (a layer-1 blockchain, meaning it is not built on top of another). In the following, we will use the term "DeSo blockchain" to refer to the underlying blockchain infrastructure, while "DeSo" will denote the ecosystem of decentralized social applications built on it.

7.1.1 DeSo blockchain

The DeSo blockchain can be viewed on the dedicated explorer website¹⁷, where all the mined blocks and transactions can be viewed. Furthermore, statistics can be seen regarding the number of DeSo users' accounts identified by a different wallet. Other statistics are the number of posts, comments, and a leaderboard of users by followers. That information can be extracted from the DeSo blockchain, given that it's written on that. In particular, the unique feature of DeSo is that it is specifically designed to be an OSN, with a set of functionalities targeted to support OSN features.

DeSo developers argue that traditional blockchains and applications built on top of them operate on a finite state, with, for example, a ledger of balances that is periodically updated. However, OSNs require the ability to create new entities, such as profiles or posts, meaning an infinite state is needed. DeSo supports this, including capabilities for

¹⁵<https://www.minds.com>

¹⁶Project overview: <https://www.deso.com/> and OSN: <https://node.deso.org/>

¹⁷<https://explorer.deso.com/>

ordered lists directly within the node functionalities, which are not typically supported by general-purpose blockchains. The developers suggest that the DeSo blockchain may also support applications beyond OSNs, provided there are well-defined schemas for the data to store [62].

7.1.2 DeSo Wallet and Mapping DeSo Accounts to DID

As said previously, a unique feature of DeSo is a 1-to-1 correspondence between a DeSo account and an Ethereum wallet address. Starting from the DeSo account's public key, DeSo allows the creation of an account that utilizes MetaMask, which functions as an Ethereum wallet. DeSo uses Ethereum's same cryptographic structure for wallet addresses, with a different encoding.

To understand this, we need to introduce derivation paths, which derive specific keys within a hierarchy of keys for a single cryptocurrency Wallet [63]. The difference between DeSo and Ethereum accounts lies in these derivation paths.

The DeSo wallet consists of a keypair generated using the BIP39 [64] standard for mnemonic seed generation [65] and BIP44 for key derivation, with the derivation path specified as `m/44'/0'/0'/0/n` (same as Bitcoin). While Ethereum wallets also utilize the BIP39 standard for seed generation, they follow a different derivation path: `m/44'/60'/0'/0/n`. This difference in the `coin_type` (0' for Bitcoin/DeSo and 60' for Ethereum). As shown in Figure 7.1, the `coin_type` specifies a key's format, which differs depending on the blockchain.

Consequently, given the two different derivation paths, using the exact seed phrase, two different key pairs will be generated in Ethereum wallets, such as MetaMask and DeSo ones. However, it is possible to convert the DeSo wallet address to the Ethereum wallet address by mathematical operations, and this is the solution we used in our integration. However, we highlight that DeSo provides a way of using MetaMask wallet addresses by creating a derived key, [66], which enables the signing of transactions on the DeSo blockchain on behalf of the Ethereum key. Instead, *to map the Ethereum Wallet Address to the DeSo Wallet Address* we used the function provided by the DeSo APIs for JavaScript [67] that allows us to convert an address from the Bitcoin/DeSo type to the Ethereum type and vice versa by mathematical operations. Said

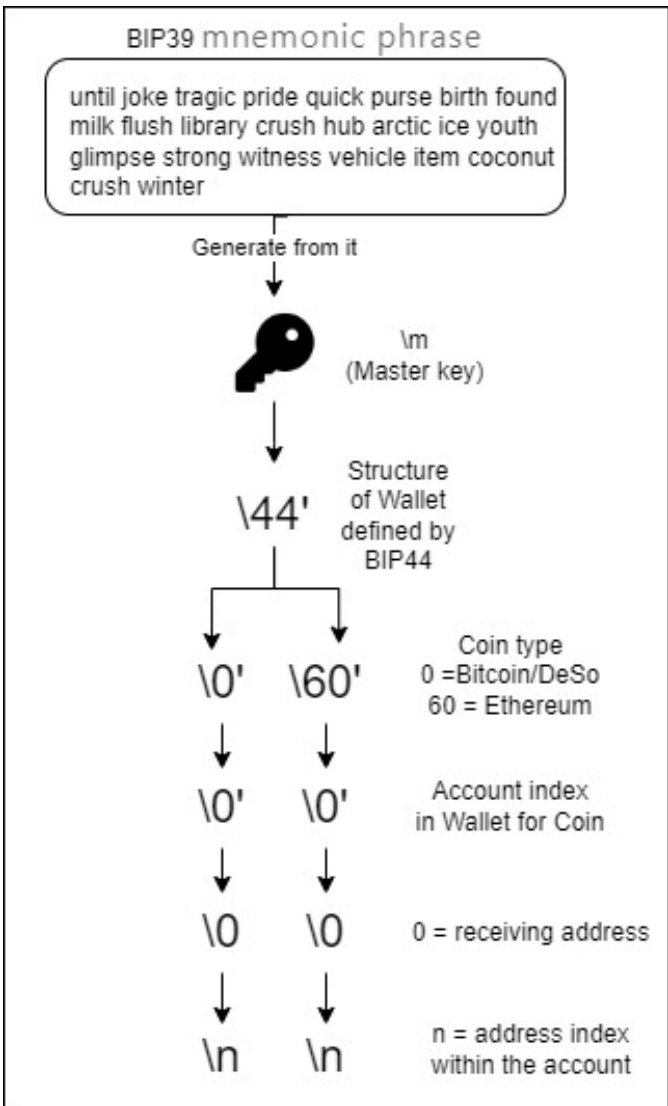


Figure 7.1: Diagram illustrating the BIP39 mnemonic phrase and its derivation path as defined by BIP44, showing the process of generating a master key, coin types for Bitcoin/DeSo and Ethereum, and the hierarchical structure of wallet addresses and account indices

function can be seen in the code box 7.1 and Figure 7.2; from the DeSo address, the process is detailed as follows:

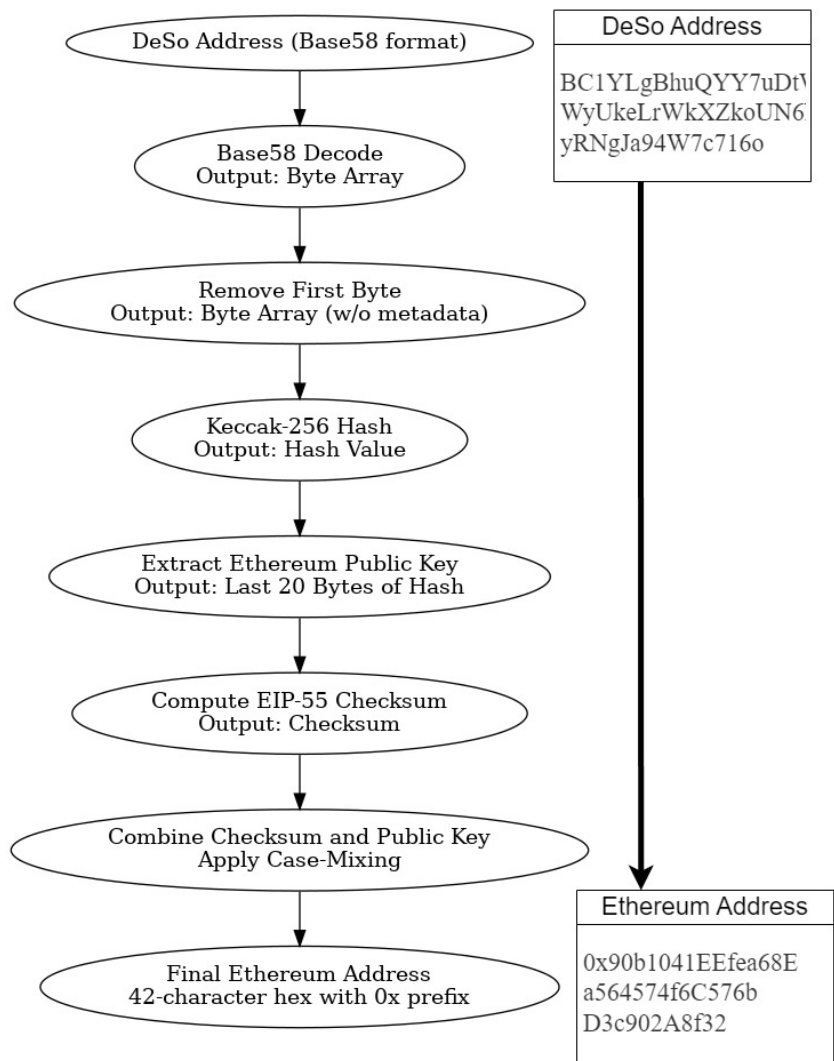


Figure 7.2: Diagram illustrating conversion from DeSo Address to Ethereum Address

1. **Base58 Decode:** Convert the provided DeSo address from its Base58 format into bytes using a Base58 decoding function.
2. **Remove First Byte:** Remove the first byte from the byte array once decoded. This is necessary because the first byte represents metadata that is not needed for the conversion to Ethereum Address.
3. **Keccak-256 Hash:** Apply the Keccak-256 hashing algorithm to

the remaining bytes. This unique hash will form the basis of the Ethereum public key.

4. **Extract Ethereum Public Key:** Extract the Ethereum public key by taking the last 20 bytes of the Keccak-256 hash. This is achieved by removing the first 24 characters from the hexadecimal representation of the hash.
5. **Compute EIP-55 Checksum:** compute the checksum using the Keccak-256 hash of the Ethereum public key. This step follows the EIP-55 standard[68], which ensures that the address is valid and recognized by Ethereum-compatible wallets.
6. **Final Ethereum Address:** Combine the checksum with the Ethereum public key to generate the final Ethereum address in hexadecimal format. To do so, during the combination, the address generated is formatted by applying a case-mixing mechanism, where characters are capitalized or left in lowercase based on the corresponding value in the checksum. The final Ethereum address is a 42-character hexadecimal string prefixed with '0x'.

Listing 7.1: Function to convert DeSo address to Ethereum Address provided by DeSo library for Javascript

```
desoAddressToEthereumAddress(address: string) {
  const desoPKBytes = bs58PublicKeyToBytes(address).slice(1);
  const ethPKHex = ecUtils.bytesToHex(keccak_256(desoPKBytes)).slice(24);
  const checksum = ecUtils.bytesToHex(keccak_256(ethPKHex));

  return Array.from(ethPKHex).reduce(
    (ethAddress, char, index) =>
      ethAddress +
      (parseInt(checksum[index], 16) >= 8 ? char.toUpperCase() : char),
    '0x'
  );
}
```

As such, the first step of our integration was creating an extension of CAVS by adding a new microservice, SSIBridgeIntegration, which interfaces with the DeSo frontend [69] and Veramo. The functionalities we implemented in the microservice are:

- **Obtain the User's Ethereum Wallet Address:** This is achieved by calling the `desoAddressToEthereumAddress` function from the DeSo protocol API for JavaScript, using the address of a DeSo account.

- **Get Key Pair:** This allows the retrieval of key pairs from a hexadecimal seed in DeSo. This function is currently not used.

Furthermore, by obtaining the Ethereum Wallet Address, we get the DID:ETHR linked to it, which is ultimately linked to the DeSo account. Additionally, we extended the JavaScript application with an Angular-based frontend for DeSo[69], incorporating calls to the SSIIntegrationHelper, VeramoAgent for CAVS, and an IPFS agent.

On the frontend, we made it so that the user will see the Ethereum address displayed as bound to their DeSo account, which is invisible in the standard web app frontend.

We made it so that the DeSo frontend WebApp is interfaced with a Veramo Agent service, and we made it so that to use the CAVS functionalities the user provides in the web app, their Ethereum wallet's private key to import the DID:ETHR into their own Veramo Agent. The architecture of the integration can be seen in Figure 7.3.

7.2 Integration View

There are multiple frontend applications; the most prominent ones today are desoworld.com and diamond.com. Meanwhile, the basic frontend application that forms the foundation of these is at the address node.deso.org (<https://node.deso.org>), which is open-source and available on GitHub. We have extended it to include the functionalities of CAVS.

Our extension implementation is available on GitHub at github.com/caltr98/frontend (<https://github.com/caltr98/frontend>) (with execution instruction); at the moment, it requires the CAVS microservices running on Docker.

Figure 7.5 shows an overview of the Author's views when creating a post on the integration front. The mapping of the DeSo Wallet address to ETH DID is shown in Figure 7.7. Furthermore, the selection of ESCO Skill VC can be seen in Figure 7.8.

Functionally, we made it so that the user experience of a DeSo user is enriched by the functionalities related to CAVS. In the following subsections, we highlight our integration of CAVS into the DeSo frontend web app.

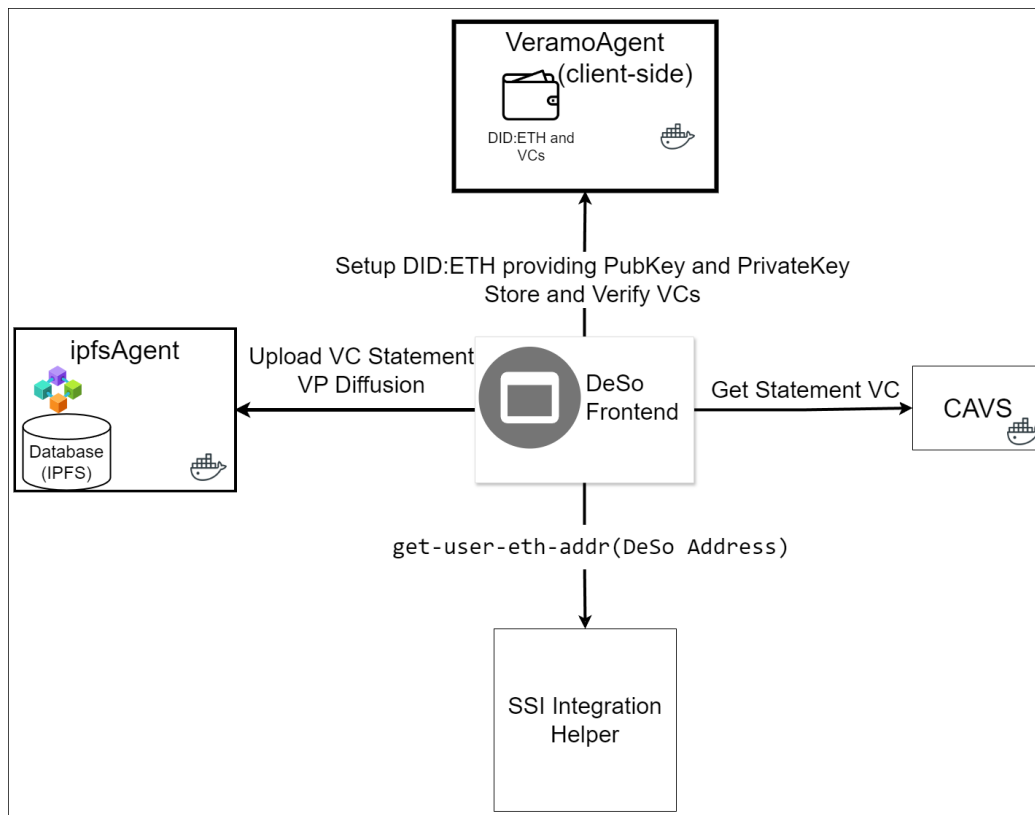


Figure 7.3: The connection between the DeSo frontend and the microservices described in chapter 5

7.2.1 DeSo address to Ethereum address

In Figure 7.7, we can see in blue our addition to the DeSo web app frontend that details the logged account funds; we displayed the User ETH Address and the associated DID:ETHR. This allows the user to see the DID they currently use for presenting ESCO VC Skills, which they receive from the CAVS VC Statement and with which they create VP Diffusions.

7.2.2 Author Posting

For an **Author** can be seen in Figure 7.6, our integration added the possibility of selecting ESCO VC Skills present in the wallet and issued to the DID:EHTR of the account as can be seen in 7.8 to generate a VC Statement to embed within the post. The post is seen as a **statement**

and sent to CAVS, which will issue a VC Statement. Given this VC Statement, we made it so that the Veramo Agent microservice of the Author automatically creates a VP Diffusion of type origin for the post loaded on IPFS. Furthermore, we made it so that the embedding of the VP Diffusion is done by loading it on IPFS, and given its CID, we changed the front end to concatenate it at the end of the post. We can see the result of a post-publication with the CID in Figure 7.4.

7.2.3 Spreader

We also extended for the Spreader the posting section of the web app, as we show in Figure 7.6, by providing the information of a previous VP Diffusion (its JWT and UUID), we made it so that a VP Diffusion of type Diffusion will be attached to the post made, and that points to the previous post that shares the statement from the Author which received a VC Statement from CAVS.

7.2.4 Reader tracing back and verifying

We made it so that the Reader of a post can verify the VP Diffusion embedded within it when there is a CID at the end of a post. We made the front end look for a string "CID:<cid_value>" at the end of a post; if present, it will display the VP Diffusion attached to the post by fetching it automatically from IPFS.

We provided a reader the possibility to trace the current VP Diffusion to the VP Diffusion of the type of origin of the statement. Furthermore, it also fetches the VC Statement inside. As such, we allow to verify it can be seen in Figure 7.9. We implemented the verification by sending the VC Statement embedded in the VP Diffusion from the front end to the Veramo Agent microservice of the user that proceeds to do it and returns the result to the front end to display it.

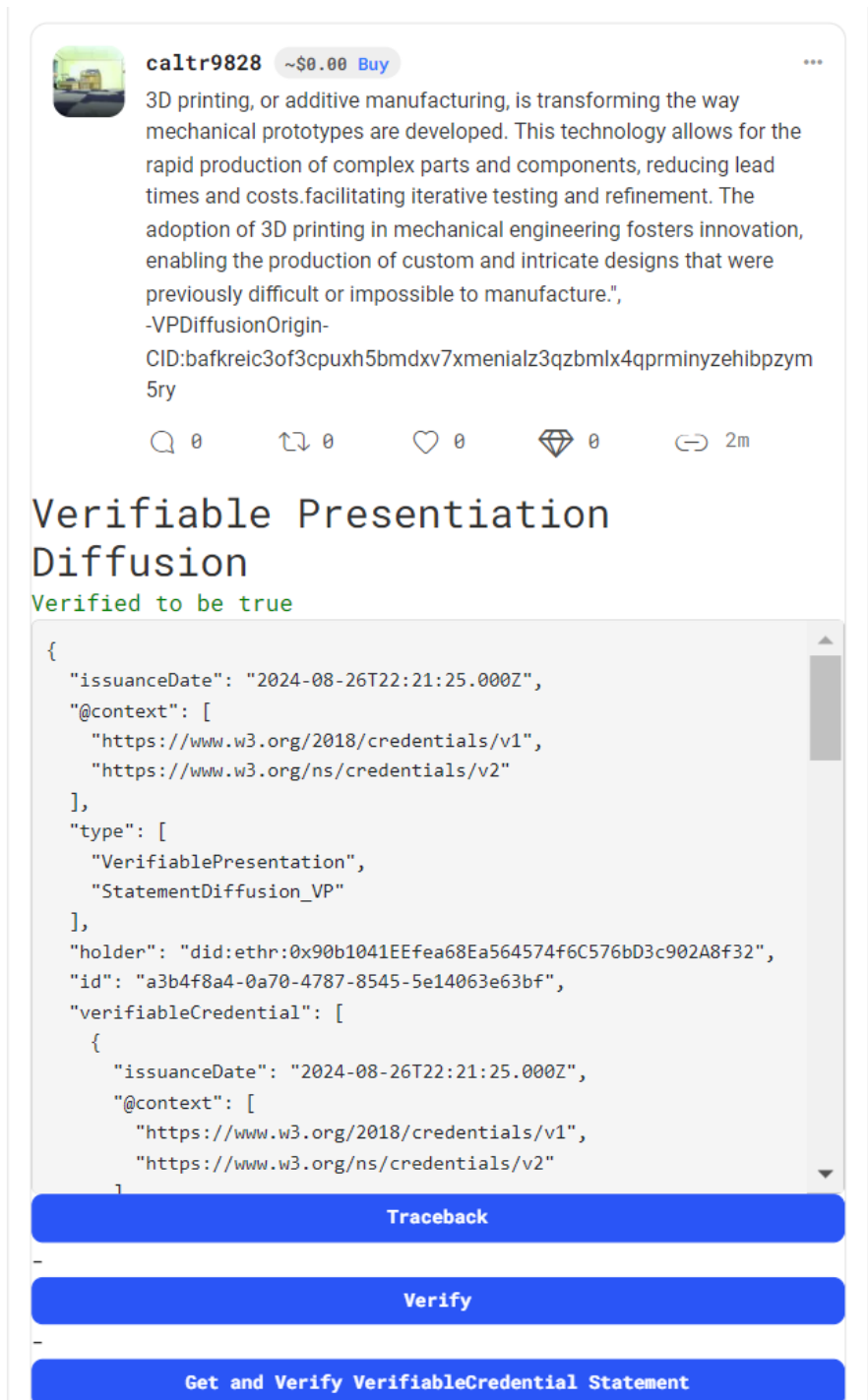


Figure 7.4: The interface of DeSo integrated with CAVS showing a statement published on DeSo, alongside its verified VP Diffusion of type origin.

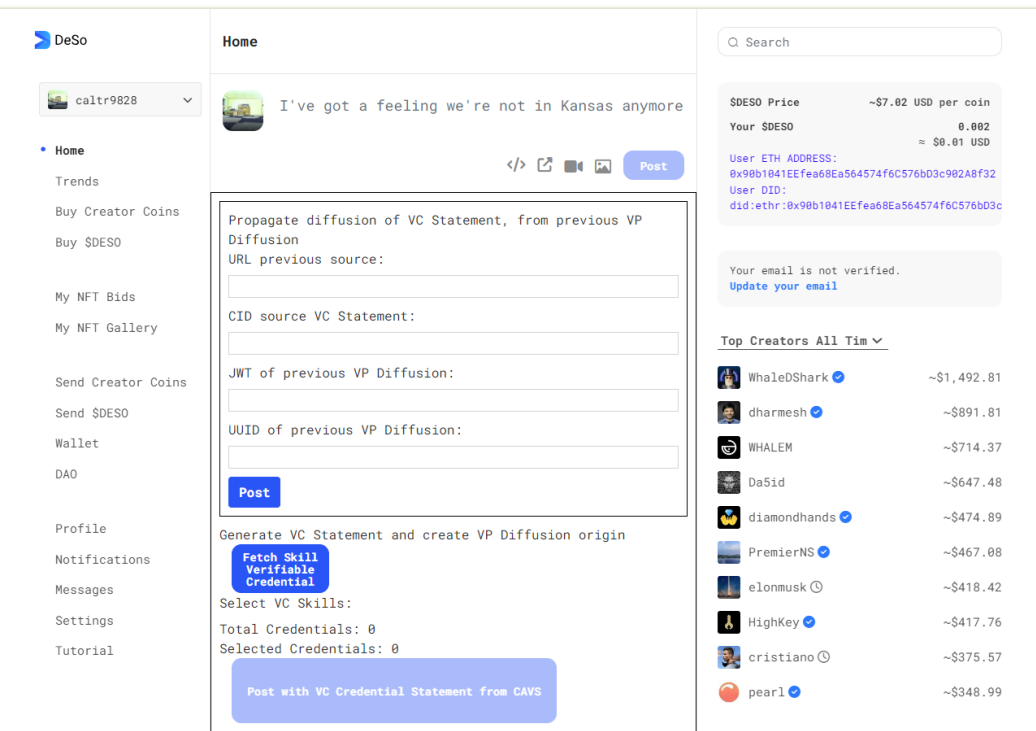


Figure 7.5: The interface of DeSo integrated with CAVS for posting with VC Statement or resharing a post from a VP Diffusion

Home


I've got a feeling we're not in Kansas anymore

</>
Share
Video
Image
Post

Propagate diffusion of VC Statement, from previous VP Diffusion
URL previous source:

CID source VC Statement:

JWT of previous VP Diffusion:

UUID of previous VP Diffusion:

Post

Generate VC Statement and create VP Diffusion origin
Fetch Skill Verifiable Credential
Select VC Skills:
Total Credentials: 0
Selected Credentials: 0
Post with VC Credential Statement from CAVS

Figure 7.6: The interface of DeSo integrated with CAVS *on center*: propagate a VP Diffusion, *on top and bottom*: provide a new statement and attach to it ESCO VC Skills for the creation of a VC Statement and VP Diffusion Origin

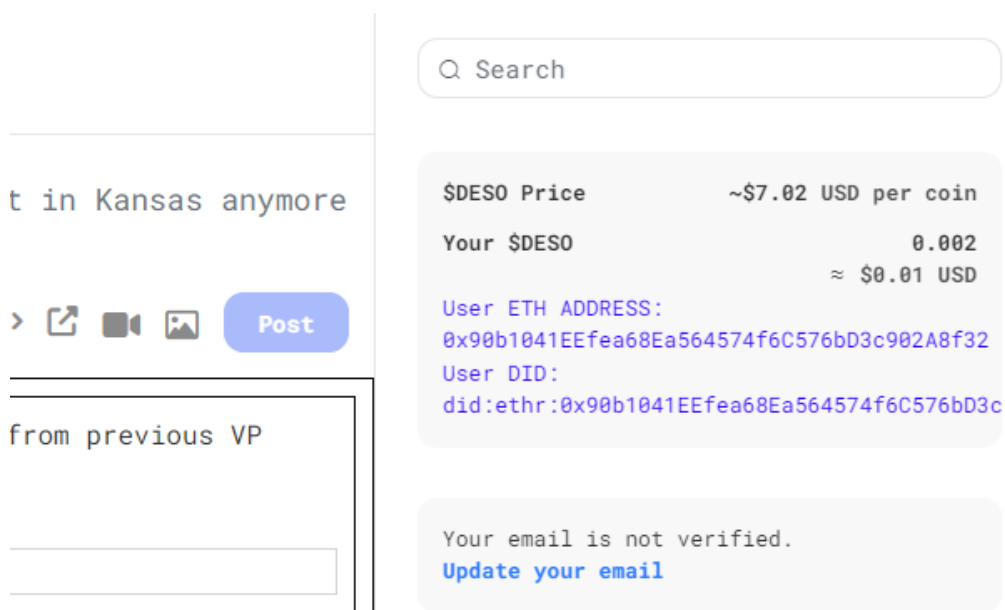


Figure 7.7: The interface of DeSo integrated with CAVS, there is the mapping of the DeSo account to an Ethereum wallet address and the corresponding DID:ETHR

Issuance Date: 2024-08-23T16:02:39.000Z

Credential Type: VerifiableCredential,
ESCO_type_VerifiableCredential

Issuer DID:
did:ethr:sepolia:0x036f320763e5cb6006ee0d272a6ec80be654bb70dde

Holder DID:
did:ethr:0x90b1041EEfea68Ea564574f6C576bD3c902A8f32

```
{
  "skills": [],
  "id":
    "did:ethr:0x90b1041EEfea68Ea564574f6C576bD3c902A8f32"
}
```

Credential #3



Hash: QmPzaneq61G56FKwFL4dzDrQdVBezfpHbH8Zbay9oUMRS5

Issuance Date: 2024-08-23T16:42:20.000Z

Credential Type: VerifiableCredential,
ESCO_type_VerifiableCredential

Issuer DID:
did:ethr:sepolia:0x036f320763e5cb6006ee0d272a6ec80be654bb70dde

Holder DID:
did:ethr:0x90b1041EEfea68Ea564574f6C576bD3c902A8f32

```
{
  "skills": [
    "engineering",
    "http://data.europa.eu/esco/skill/b8ac179c-4a29-4467-9177-031daf1321d6"
  ]
}
```

Processing statement by CAVS...

Figure 7.8: The interface of DeSo integrated with CAVS, showing the selection of ESCO Skill VC inside the DeSo integration

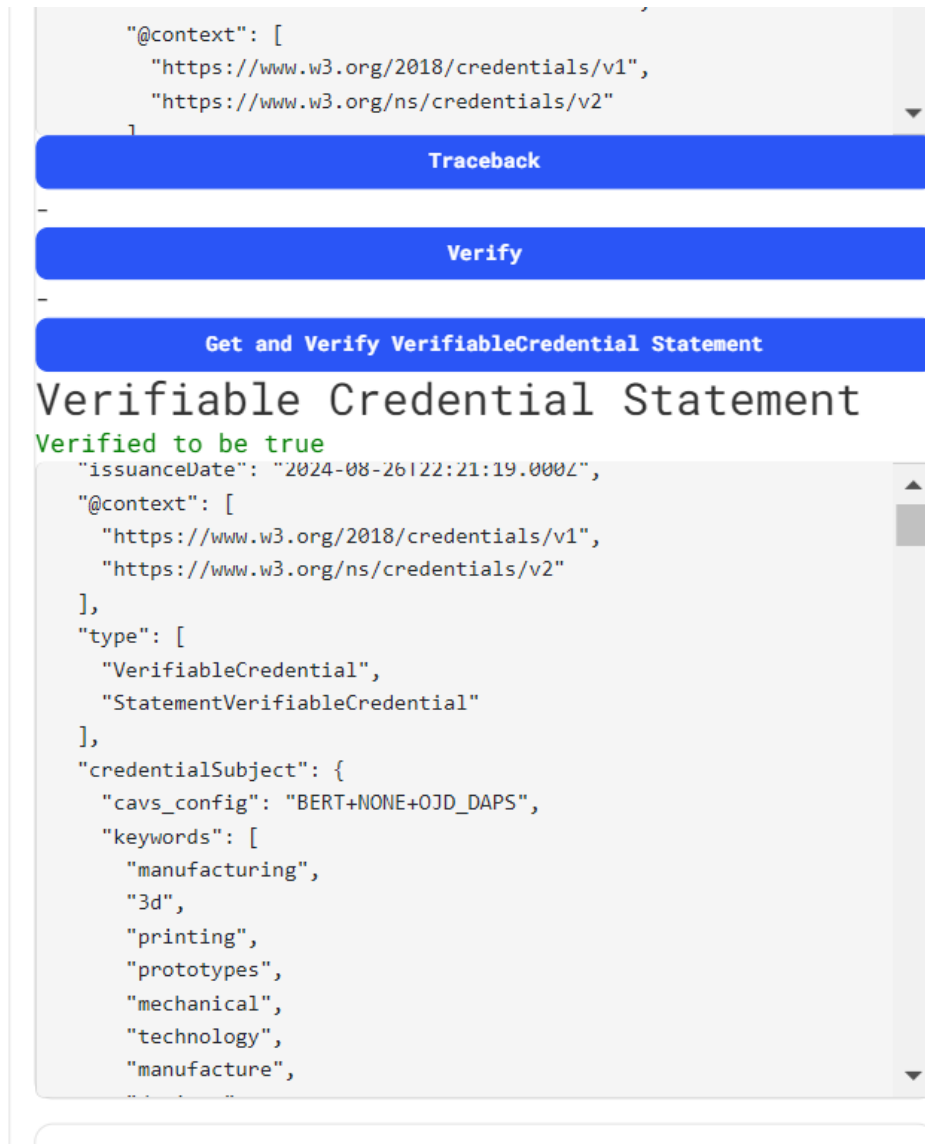


Figure 7.9: The interface of DeSo integrated with CAVS showing a verification resulted to be true of a VC Statement on the DeSo integration

Chapter 8

Experimental results and Analysis

In this chapter, we present an OSN simulation to validate the effectiveness of the mitigation measures introduced by CAVS, security analysis with threats and mitigations, some alternative designs for an SSI-based system, and performance tests to assess the effectiveness of the CAVS components.

First, our OSN simulation considers two types of actors: Users (base) and Checkers; the latter is a subset of users who fact-check any post before resharing or publishing it. The OSN is crafted by creating User objects and setting up their relationships, where one user may follow another with a random probability.

In this section, we compare the results of two simulations, Ground Truth and Skill-Based Checkers simulations. The first simulation assumes that the Checkers know the Ground Truth, i.e., they know whether a post contains fake news or **true news**, and **they share only the latter**; the other simulation, instead, assumes that Checkers does not know the Ground Truth and that *they decide whether to share posts assessing how much the Author's skills match the post content*, which is our proposed solution. For instance, if the skill match count is below a predefined threshold, the Checker will not share the post because they don't consider the authors an authoritative source for that kind of content, thus assessing the post as possible fake news.

To run our simulation, we created a synthetic dataset which **embeds authors' bio and skills, the statements (articles) they post, and for each statement, a count of the Author's skills matching the ones associated with the statement itself.**

8.1 Testbed Overview: Ground Truth and Skill-Based simulations

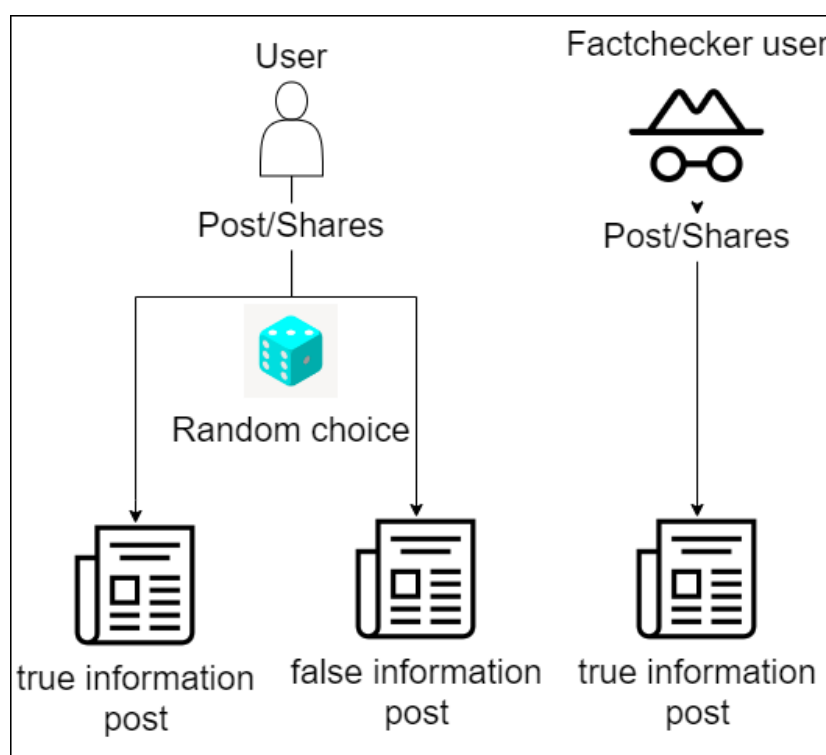


Figure 8.1: A representation of what the two classes of users, normal and Checkers, share as posts, being posted with true or false information, it highlights how normal users make a random choice for the choice, while Checkers only post true information

We consider two classes of users and simulate their behavior:

- Normal users: they may share true or false information without checking its truthfulness. They can be authors of said information as a post, or they may reshare a post containing it.

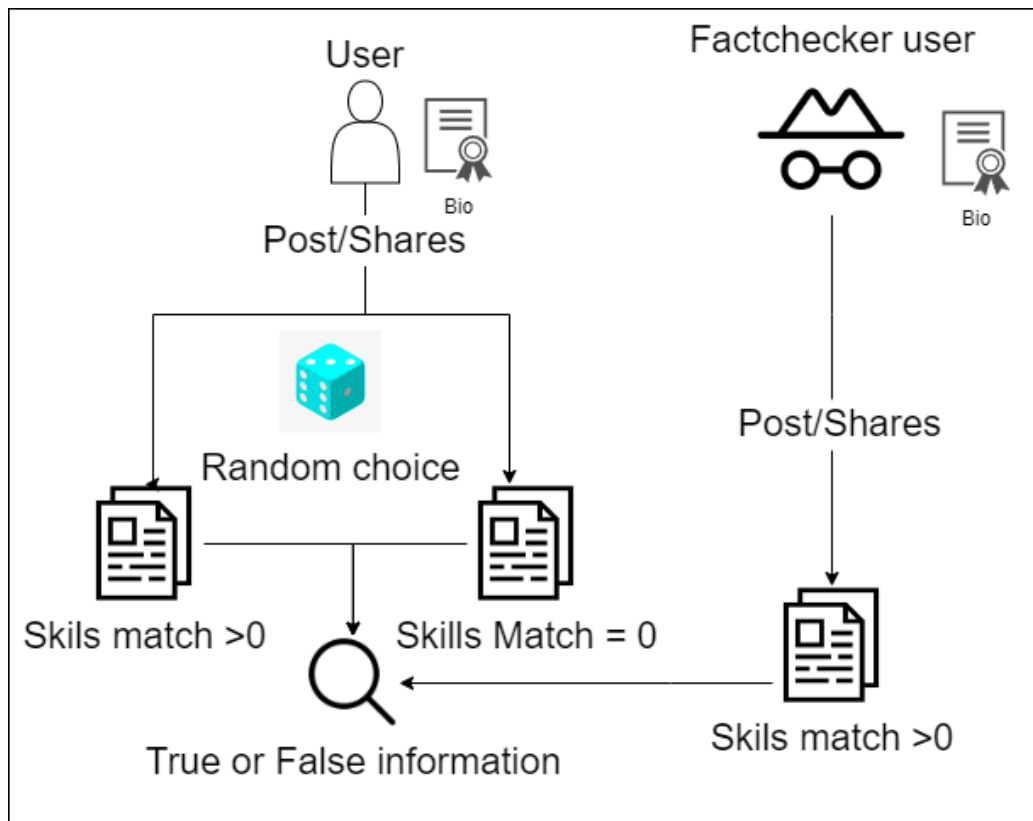


Figure 8.2: The two classes of users, normal and Checker. Highlighting how they share as posts those with skills match count either is non zero or zero, which may ultimately be true or false information; it highlights how normal users make a random choice, while Checkers only post information which matches with the skills from the bio

- Checkers users: Those who assess the truthfulness of information before creating a post or resharing a post holding the information.

Those two classes differ in the two simulations with Checkers that know the Ground Truth, visible in Figure 8.1 and those that use the skills matching count that can be seen in Figure 8.2. In particular, the first simulation is detailed as follows:

- Normal users: those users post/reshare true or false information in a post depending on a random choice
- Checkers users: that user can see if the information is false or true; if it's false, they do not share it as a post. Furthermore, if they see

a post with false information, they do not share it.

In the simulation with Skill-based Checkers, each user is characterized by:

- a bio: a description of the user of an OSN
- a list of five articles: three are true, and two are false as labels.

Furthermore, the articles are characterized by:

- title: description of the article
- text: the statement that the Author makes in the article
- skill_count: the count of skills that match the Author's competencies; said the Author in the bio states competencies.
- label: stating if the article holds true information or false information

There can be cases where the label is false, but the skill count is greater than 0, this highlights **our assumption that** in some instances, even competent authors on a topic may spread misinformation. Furthermore, articles with a label of true may have a skill count equal to zero.

All users of the simulated OSN will have a personal feed of posts, which is shared with all their followers. The following relationship between all users is set up randomly at the start of the simulation. Users may publish a new post in their feed or can share a post from other users they follow, putting them in their feed.

8.2 Dataset Generation for the Testbed

We have generated such a synthetic dataset with an LLM such as GPT-4o because it is a resource-effective way to obtain labeled data without human labor [70]. The data generated are persons' OSN profiles, including a Bio that highlights their skills according to the ESCO ontology and a set of articles. The articles will be the content of the posts inside the simulation; note that a user may produce more posts with the same article, as we generated only 5 articles for convenience, while a user's posts may be more.

Articles include title, text, and whether they are fake or true news (false or true statement). Figure 8.3 illustrates how the dataset is generated

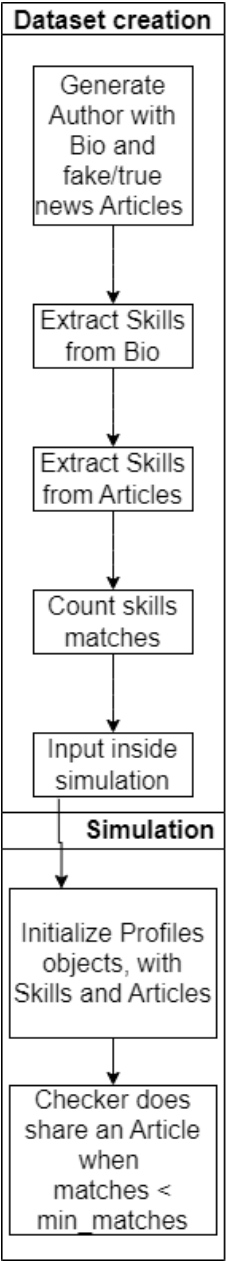


Figure 8.3: The process of generating the synthetic dataset and incorporating it in the simulation

and processed. After GPT-4o generates the data, we extract the skills

using the `ojd_daps_skills` microservice from Articles and Profiles Bio. The processed dataset is a JSON file of profiles, each with articles that hold the skills match count with the bio of their Author. The processed dataset is given to the simulation to generate Profiles of the Users and Checkers.

The dataset is made available at kaggle¹⁸ The code to create the synthetic dataset we illustrate below is in the CAVS GitHub repository¹⁹. The dataset generation code is divided into two files: `gen.py`, which communicates to OpenAI GPT APIs to generate profiles, and `skillify.py`, which communicates with CAVS to extract skills from profiles and their articles.

The prompt (In the AI/ML field, the term "prompt" refers to the instructions or input given to the model) to generate the synthetic dataset, which are instructions to define a GPT assistant, can be seen in listing 8.1:

Listing 8.1: Instructions for Generating Profiles using ESCO Ontology

```

1 You are an article/news writer AI. Your task is to use the ESCO ontology to generate
  profiles of individuals for creating news articles. Each profile must include the
  person's Name, Surname, gender, and a BIO that maps directly to ESCO ontology skills.
2
3 Ensure that the technologies and knowledge expressed in the articles are related to the
  ESCO skills listed in the BIO, even in name. The output should be in JSON format.
4
5 The input will request you to create a profile; your response should be a unique profile
  with the following specifications:
6
7 1. Create an equal number of male and female profiles (50/50 gender split).
8 2. Generate a bio of exactly 1000 characters highlighting skills that align with ESCO
  ontology.
9 3. True news articles must relate to the ESCO skills listed in the bio. Fake news articles
  should generally be unrelated to the person's ESCO skills. According to the Oxford
  Internet Institute Report (2018), 90% of fake news is from sources lacking
  credibility. Therefore, at least 90% of the fake news articles should be completely
  unrelated to the person's ESCO skills.
10 It is mandatory that the technologies and knowledge expressed in the articles are related
  to the ESCO in the bio, even in name.
11 Ensure that all true news articles relate to the skills included in the person bio.
12 Structure each profile as follows, IT IS MANDATORY TO FOLLOW THIS EXACT STRUCTURE FOR EACH
  PROFILE:
13 {
14   "name": "First Name",
15   "surname": "Last Name",
16   "gender": "male | female",
17   "bio": "A detailed summary of the person's skills and expertise related to ESCO
  ontology.",

```

¹⁸<https://www.kaggle.com/datasets/calogeroturco/synthetic-articles-with-authors-having-esco-skills>

¹⁹<https://github.com/caltr98/CAVS/ProfilesAndNewsGenerator>

```

18     "articles": [
19         {
20             "title": "Article Title",
21             "content": "Exactly 1000 characters of content related to the article.",
22             "news-status": "true-news | false-news",
23             "url": "URL of the news source"
24         }
25     ]
26 }
27
28 For each profile, generate five news articles:
29 - 3 true news articles related to the person's ESCO skills.
30 - 2 fake news articles generally unrelated to the person's ESCO skills.
31
32 The ESCO dataset version must be v1.2.0. Only use the labels from the specified list.
33
34 Each profile must be unique. Use the seed provided as a base for the generation.

```

Then, we query the assistant by sending a message that holds a seed for randomized generation, and the message sent to OpenAI's GPT-4o can be seen in listings 8.2

Listing 8.2: Instructions for Generating Profiles using ESCO Ontology

```

1 Per your instruction, create a profile with ESCO SKILLS in the bio. Ensure that the exact
  number of articles specified in your instructions are generated and that the profile
  is unique with respect to any other past generated. Make sure that no reference to
  the person's name, surname, or gender is included in the title or content of the
  articles. It is mandatory that the technologies and knowledge expressed in the
  articles are related to the ESCO in the bio, even in name. Make sure that all true
  news articles are related to the skills included in the person bio, while false news
  articles are generally unrelated to the skills. Make sure that the profile format
  corresponds to that defined in your instruction, both in json structure and json keys
  being lowercase. Use as seed: SEEDVALUE

```

Re-enforcing the instructions to define the assistant led to better results in following a precise output structure to produce a standard JSON for all the profiles.

8.3 Ground Truth Simulation

The simulation consists of a Susceptible-Infectious-Recovered (SIR) model as done previously in the literature [71]. It is synthesized in Figure 8.4 Moreover, the states for users are defined as follows:

- **Susceptible:** A user in this state only posts true news. It can evolve into the Infectious state by **sharing fake news from their feed**, which includes the users it follows posts, or it can be Infectious

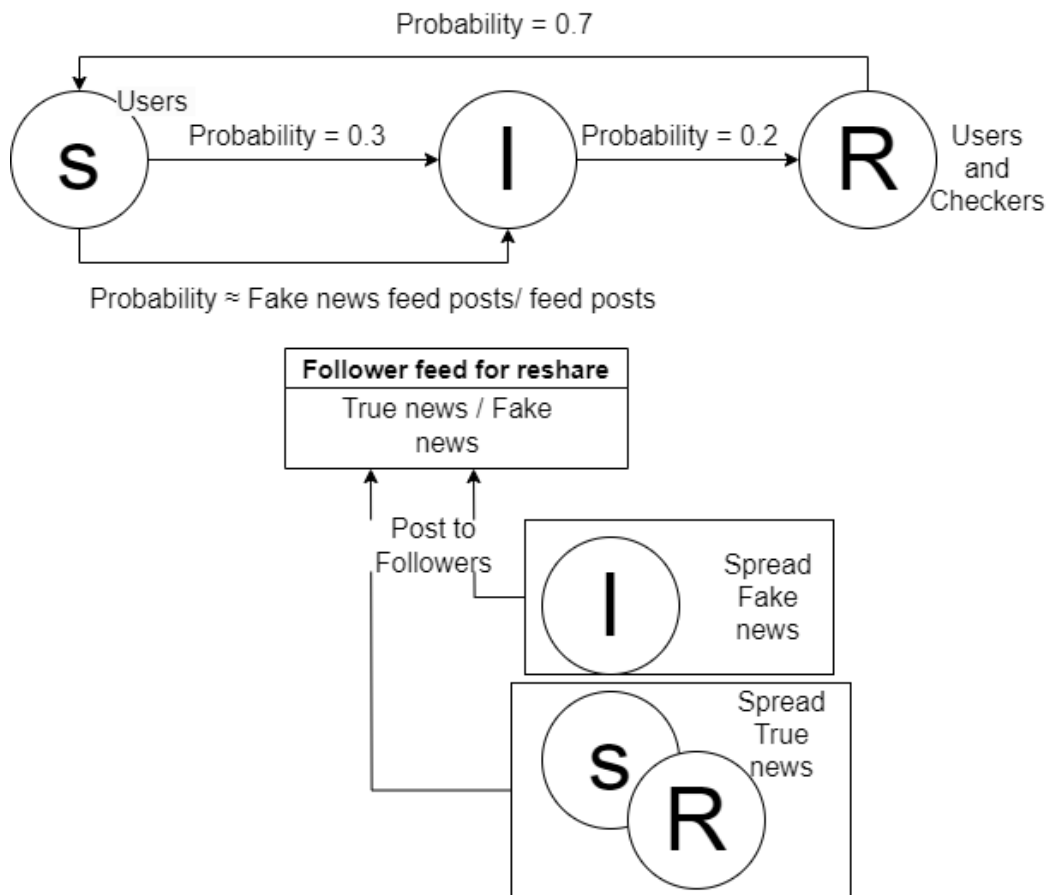


Figure 8.4: The SIR model in the Grount Truth simulation. We can see with what probability state changes and what type of posts users share in different states

at each turn with a probability of 30% probability from a random choice. A standard user (non-Checker) will start in this state.

- **Infectious:** A user in this state will only post fake news and reshare it. This state can evolve in the recovered state with a probability of 20% of a random choice.
- **Recovered:** A user in this state will only post true news and share it. A recovered user can become Susceptible with a probability of 70% of a random choice. A Checker will start and remain in this state during the entire simulation.

8.4 Skill-Based Simulation

In this case, the (simulation on the dataset) Checker may unknowingly share fake news, which can become Infectious. The Checker starts from the Recovered state. What may happen is that:

- a Checker may reshare a post from their feed for which the number of skills matched is above a minimum threshold set at the start of the simulation. Notice that it may be fake news. In this case, the Checker remains Recovered if it posts true news.
- a Checker may post fake news that still obtained skill matches greater than or equal to the minimum threshold. This leads the Checker to the Infectious state.
- the Checker may return to the Recovered state with the 70% probability of a random choice at each turn or by posting/resharing true news.

8.5 Running and Comparison

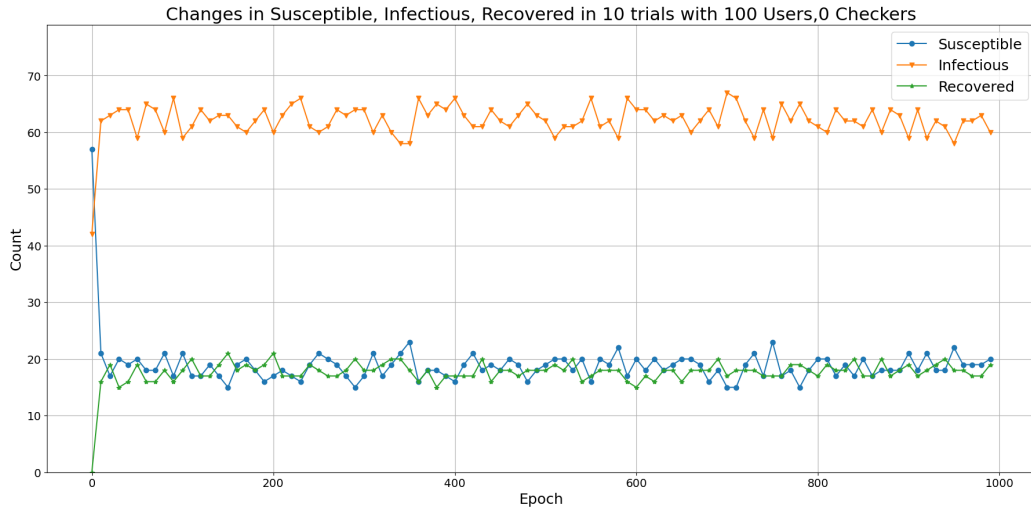
This section shows the results of our experiments to validate our proposal. Each experiment simulates the behavior of 100 OSN users (including Checkers) in 1000 epochs. To collect metrics, we average the measures collected during five execution of the simulation. In each epoch, each user may perform the actions "post" or "reshare," with a probability of 30% and 70%, respectively.

Note that there may not be any posts to reshare or that a user's feed was exhausted in the selection process, so no reshare occurred in an epoch. The reshare process starts from the post on top of the feed; the user selects it for resharing with a probability of 50%; if not reshared, it proceeds down in the feed.

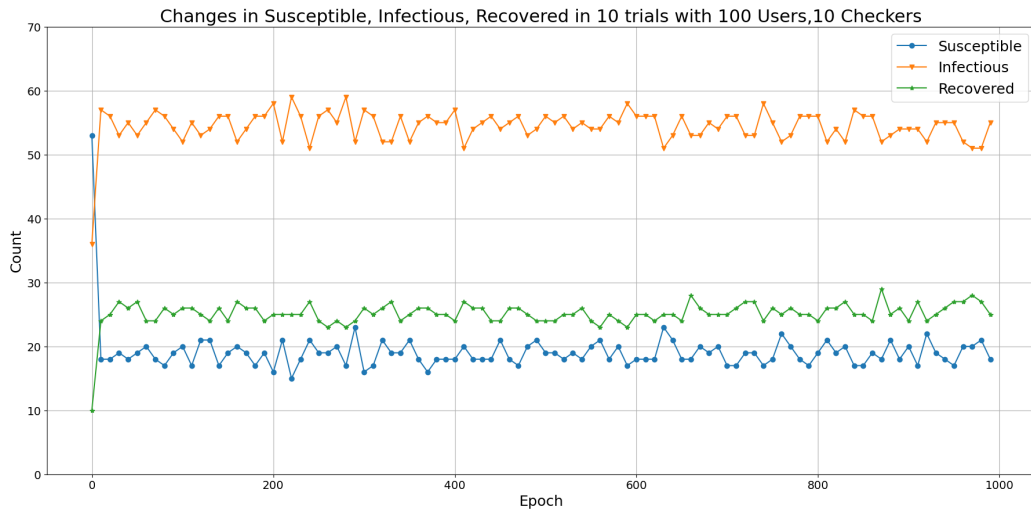
The following relationships between each user are recreated for each simulation, where, as stated previously, a user may follow another user with a random probability.

8.5.1 User State Evolution

In this subsection, we detail the *state of the users for each epoch* (according to the states defined in Section 8.3); in this way, we can have a



(a) SIR model state changes with 0 Checkers (Ground Truth simulation).

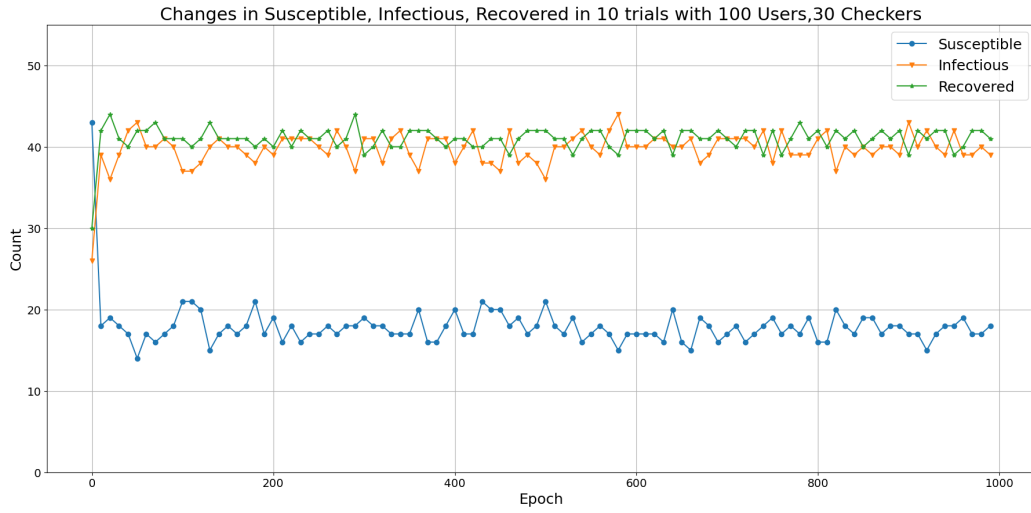


(b) SIR model state changes with 10 Checkers (Ground Truth simulation).

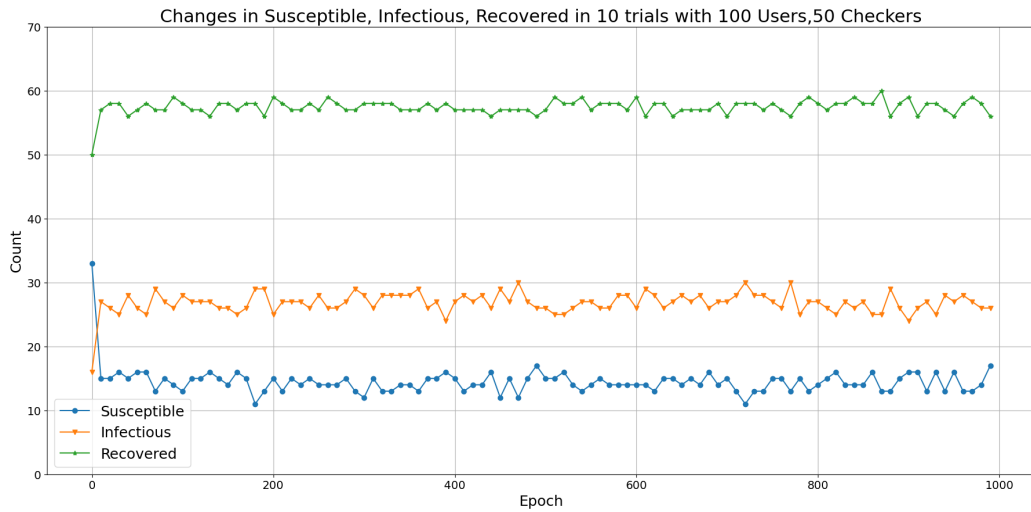
Figure 8.5: SIR model state changes with 0 (top) vs. 10 (bottom) Checkers in Ground-Truth simulation. The number of users in a particular state is shown at each of the 1000 epochs.

picture of how the state changes over time, depending on the number of Checkers.

In particular, we present graphs showing the user numbers in a particular state for each epoch, **comparing the results of the experiments for the Ground Truth Simulation with the ones obtained for the**



(a) SIR model state changes with 30 Checkers (Ground Truth simulation).



(b) SIR model state changes with 50 Checkers (Ground Truth simulation).

Figure 8.6: SIR model state changes with 30 (top) vs. 50 (bottom) Checkers in Ground-Truth simulation. The number of users in a particular state is shown at each of the 1000 epochs.

Skill-based simulation (on the synthetic dataset), to showcase how the Checkers effectiveness changes. In the figures, we highlight the simulation differences when increasing the number of Checkers for each of the two simulations.

Figure 8.5 shows the users' state changes in the simulation exploiting the Ground Truth simulation over 100 users. The top graph shows the

state changes with 0 Checkers and the bottom with 10 Checkers. We can see that by adopting 0 Checkers, the Infectious users are above 60 for most of the epochs; introducing 10 Checkers reduces the Infectious users between 50 and 60. However, the reduction can be seen as natural, given that Checkers are never infectious. However, we do not notice an effect on the whole OSN user base as an increase from the case with 0 Checkers is slight. To get an assessment of the Ground Truth simulation with more Checkers, Figure 8.6 shows the state changes in the Ground Truth simulation with 30 Checkers (top graph) and with 50 Checkers (bottom graph). We can see how, concerning the previous diagrams with 10 Checkers, the reduction in Infectious is more significant; this is a symptom. Having 30 Checkers reduces the exposition to fake news, allowing each epoch above 40 users after the first few epochs. As we also noticed for the 50 Checkers case, there are around 10 more users in the Recovered state besides the said Checkers. In this bottom case, the reduction in infections is more significant, as the infections are around 30 in each epoch, and the Recovered users are more than the Infectious ones.

For the Skill-based simulation, we previously introduced the concept of considering the content of a post as reliable according to a threshold defining the minimum number of authors' skills that should match the post's topics. To assess the effectiveness of increasing such a threshold, we can look at Figure 8.7, which shows the state changes in the Skill-based simulation with 10 Checkers by setting the skill matches threshold to 1 (top graph) and 3 (bottom graph). We may notice how increasing to 3 the threshold has a negligible effect. As a matter of fact, in both cases, there are around 50-60 users in an Infectious state.

Figure 8.8 shows the user state changes in the Skill-based simulation with 30 and 50 Checkers (respectively, top and bottom graph), setting the skill match threshold with a value equal to 1. We can see that in the case of 30 checkers, the Infectious is above 40, slightly more than the Ground Truth simulator with 30 Checkers. Furthermore, if we use 50 Checker, we may see that the results are similar to the Ground Truth simulation. The Infectious users are below 40 and stay between 30 and 40 for each epoch, showing a positive effect from 30 to 50 Checkers in the Skill-Based simulation.

Finally, we compare the Ground Truth (top graph) and Skill-based simulations (bottom graph) in Figure 8.9. In both simulations, we use 70

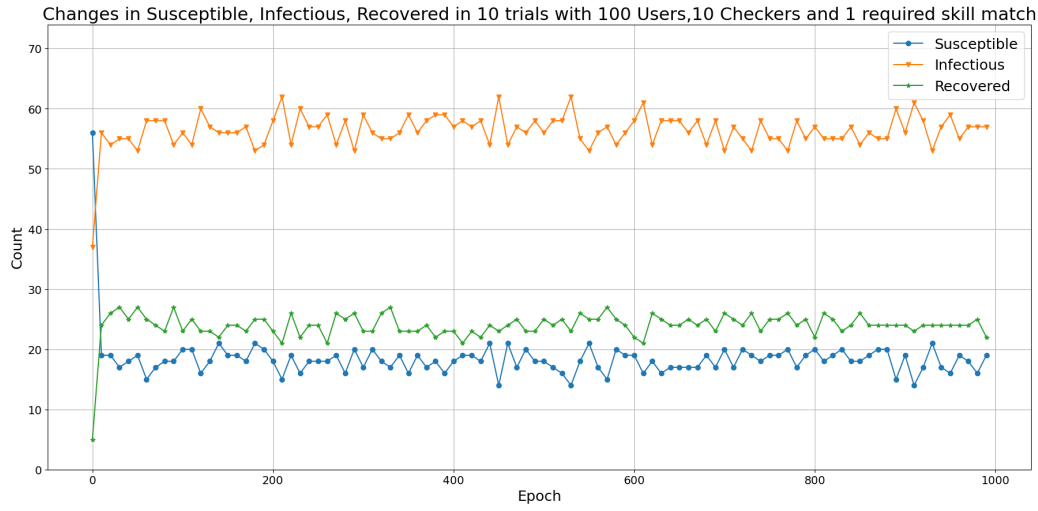
Checkers, and we set to 1 the skill matches threshold. The figure shows that the two simulations have similar results, as the Infectious stays between 10 to 20 at each epoch for the Ground Truth simulation. In contrast, in the Skill-based simulation, they go above 20 only in a negligible number of epochs. We may see how the Checkers are always in the recovered state in the Ground Truth case, while in the dataset case, they can become infectious. Nonetheless, the addition of Checkers in both cases has an influence benefit in creating an OSN where most of the users are in the recovered state, at least in most of the epochs, as we can see in the 50 Checkers case for the dataset, there may be viral spread of fake posts with matching skills that may cause the infectious user to be more than the recovered one in some epochs.

Table 8.1 highlights the number of posts containing false news or true news containing posts for different simulations, as we compare the Skill-Based simulations to the Ground Truth Simulation when having from 10 to 70 Checkers. In particular, for the Skill-Based Simulation, 10-1 and 10-3 are the first simulations when the skill threshold is set to 1, and the latter where it's set to 3. We may see how introducing more Checkers in both simulations causes a significant reduction in posts containing fake news, where the dataset simulation improvements are not far off the Ground Truth simulation results.

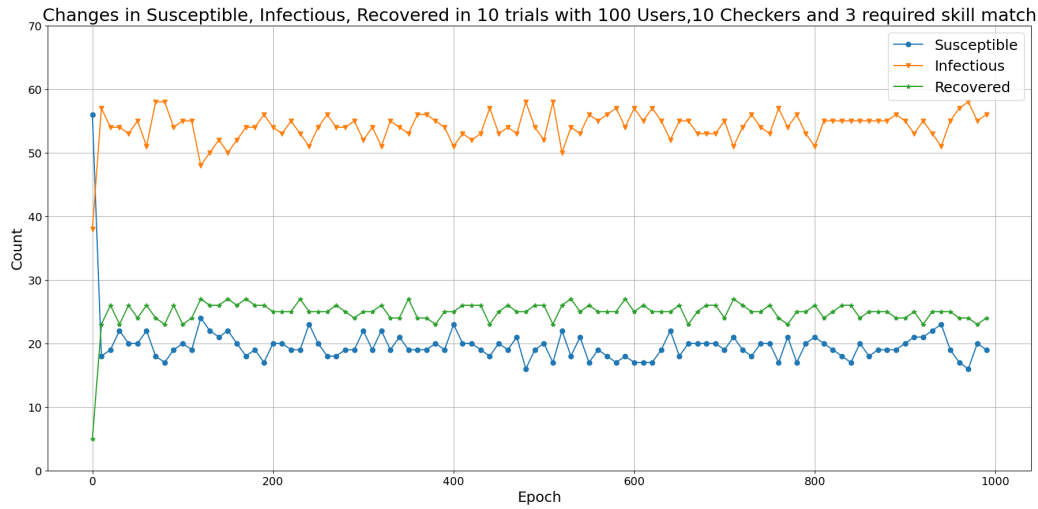
8.5.2 Posts propagation modeled as tree

To better understand how the post spreads, we may model the lifetime of a post with a tree. The node of a tree represents a user that shared(root) or reshared the data, and the leaves represent terminal users whose reshared post was not reshared afterward. We may visualize it as in Figure 8.10 Table 8.2 shows the metrics of post spread as a tree, showing the depth of the post share tree, the breadth, and the whole size of the tree, both for fake vs. true news. In this Table, as the one before, we compare the skill-based simulations to the ground truth simulation, which has 10 to 70 Checkers, and the skill-based simulation on 10 Checkers, with the skill match threshold set to 1 and 3. The Table highlights how, with the addition of Checkers in the Ground Truth simulation, there is a decrease in tree depth of the fake news posts resharing.

It may be noted that in the dataset simulation, the tree depth of fake



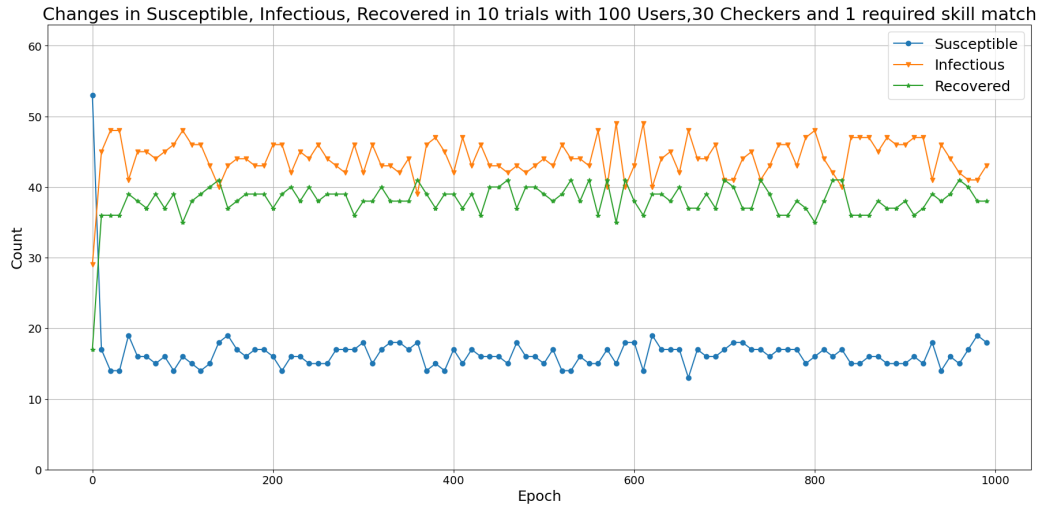
(a) SIR model state changes with 10 Checkers and one minimum skill match (Skill-Based simulation).



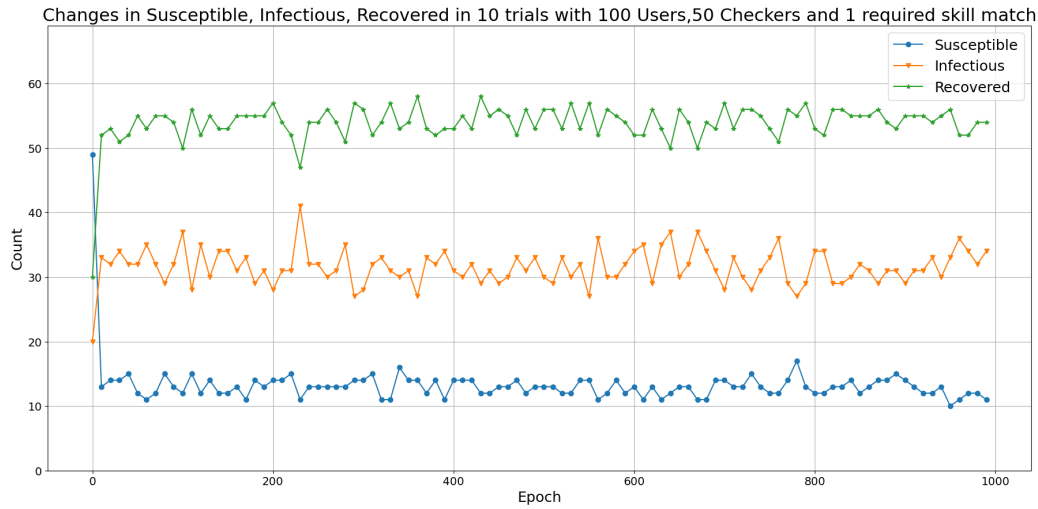
(b) SIR model state changes with ten Checkers and three minimum skill matches (Skill-Based simulation).

Figure 8.7: SIR model state changes with 10 Checkers in Skill-Based simulations with Checkers requiring 1 (top) vs. 3 (bottom) minimum skill matches. The number of users in a particular state is shown at each of the 1000 epochs.

news posts increases instead of decreases. This may be a sign that **given that there is less fake news circulating, those that will be more**



(a) SIR model state changes with 30 Checkers and one minimum skill matches (Skill-Based simulation)

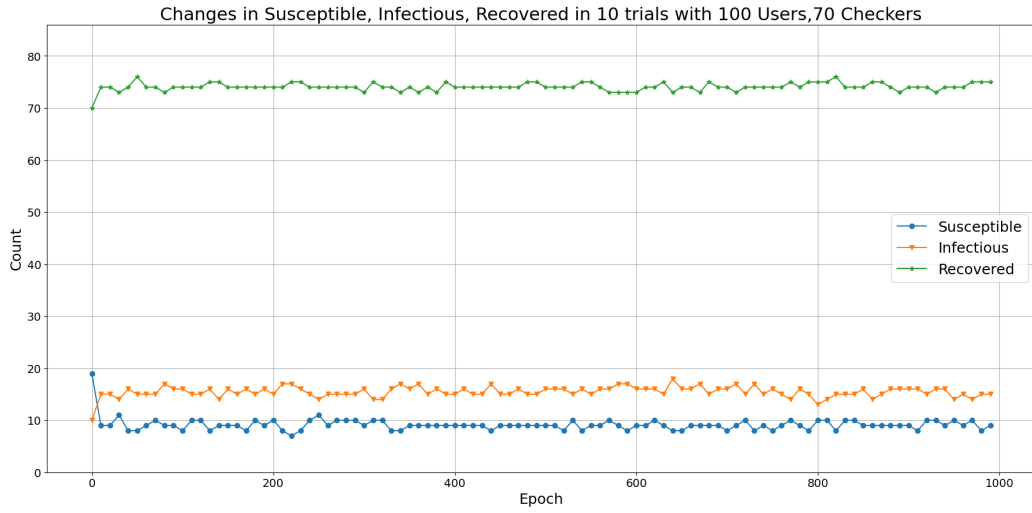


(b) SIR model state changes with 50 Checkers and one minimum skill matches (Skill-Based simulation)

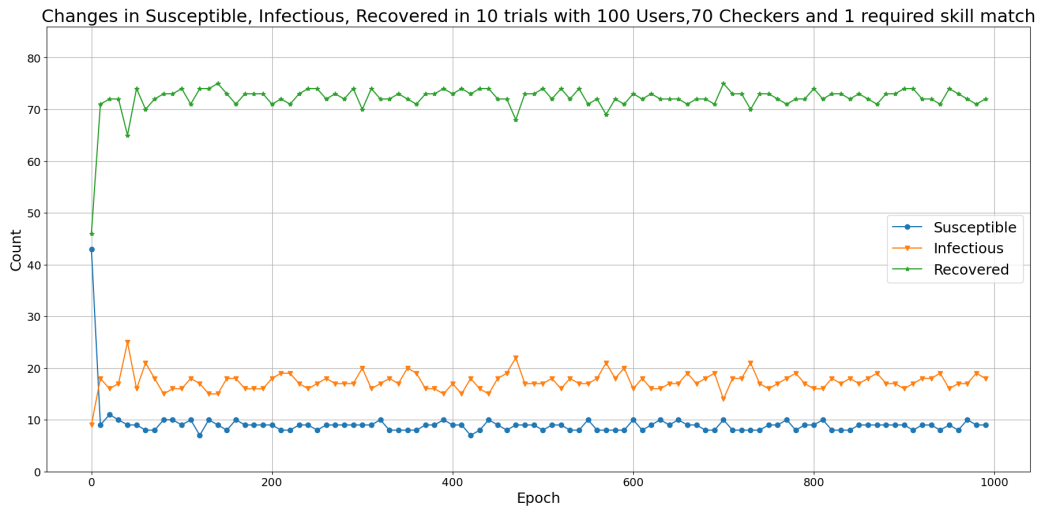
Figure 8.8: SIR model state changes with 30 (top) vs. 50 (bottom) Checkers in Skill- simulations. The number of users in a particular state is shown at each of the 1000 epochs.

shared more, this may provide clear advantages to moderators of an OSN, which may spot them with more ease.

We may see that the tree size of true news is the same for the Ground



(a) SIR model state changes with 70 Checkers in Ground Truth simulation



(b) SIR model state changes with 70 Checkers and one minimum skill match in Skill-Based simulation simulation

Figure 8.9: SIR model state changes with 70 Checkers in Ground Truth (top) vs. Skill-based (bottom) simulations. The number of users in a particular state is shown at each of the 1000 epochs.

Truth simulation and the dataset simulation, showing that the pattern of sharing true posts is similar regarding true news. At the same time, the true news posts tree decreases in size because more true news is circulating, leading to more posts of true news with less resharing.

Table 8.1: Table showing the average number of posts generated by category fake news and true news on 10 simulations long 1000 epochs. The pos particular of fake news (Fake Posts) and true news (True Posts). **D** indicates that it is the Skill-Based simulation on the synthetic dataset, with the number of Checkers-min skills matching on the first column. **GT** indicates the Ground Truth simulation, with the number of Checkers. The 0 checkers case is equal for both simulations.

Averages $\pm$ stdev /simul. type	n.Posts	n.FakePosts	n.TruePosts
10-1 D	30033.9 ± 140.0	15365.8 ± 113.9	14668.1 ± 107.6
10-3 D	29995.1 ± 97.8	14930.7 ± 118.4	15064.4 ± 74.9
30-1 D	30005.0 ± 160.8	12471.1 ± 200.3	17533.9 ± 119.7
50.1 D	30043.6 ± 115.0	9132.8 ± 190.1	20910.8 ± 215.7
70-1 D	29963.4 ± 135.7	5331.2 ± 147.3	24632.2 ± 221.5
0 GT D	30056.8 ± 192.2	16866.6 ± 159.8	13190.2 ± 145.2
10 GT	30030.7 ± 96.7	14869.4 ± 129.5	15161.3 ± 96.8
30 GT	30012.9 ± 127.8	11332.6 ± 101.7	18680.3 ± 104.5
50 GT	30021.9 ± 129.8	7921.8 ± 88.2	22100.1 ± 110.0
70 GT	29964.0 ± 151.1	4679.6 ± 64.2	25284.4 ± 143.7

8.5.3 User centric statistics

Table 8.3 shows the Skill-based vs Ground Truth simulation and some of the final statistics on average for both simulations. The number of reshared posts and, in particular, fake news reshares. Additionally, the count of users for each state: Recovered, Infected, and Susceptible are shown. The Table highlights how the number of fake news reshared by

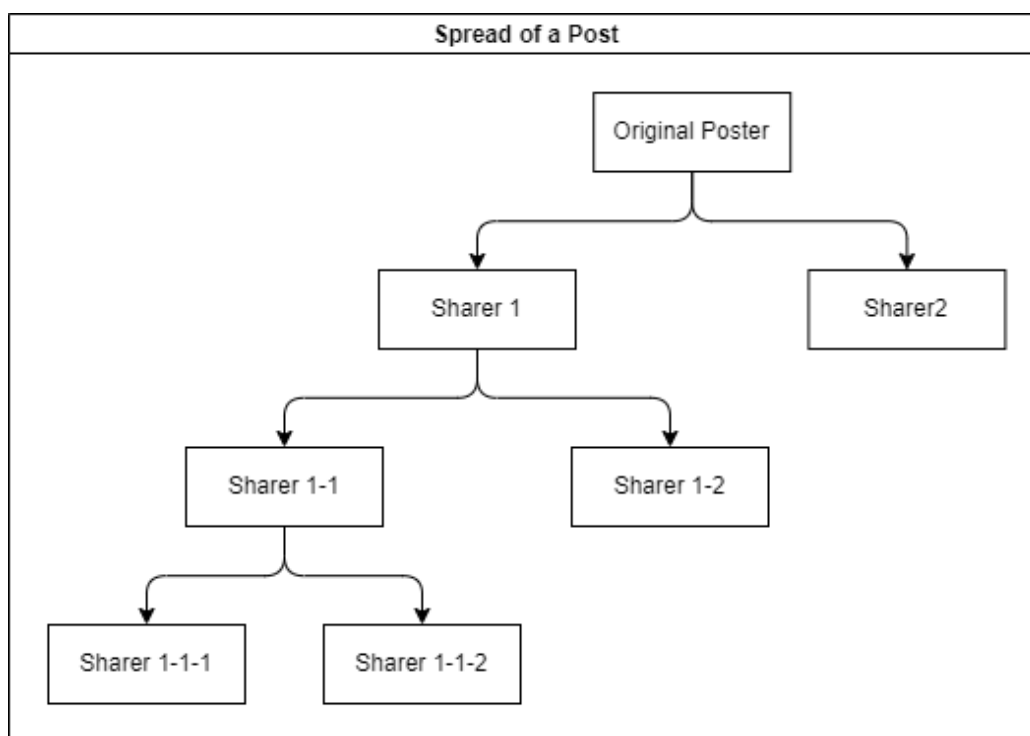


Figure 8.10: Tree diagram showing the spreading of a post from user to user, each user shares a post after having seen it shared/reshared by another user, which is their parent in the tree.

users decreases with the increase in Checkers. At the same time, we see close results in the simulations when looking at the user states at the end of the simulations.

8.5.4 Comment

Given the analysis performed, we may conclude that our tool can mitigate fake news while not inherently solving it. The tool shows comparable effectiveness concerning an oracle knowing the Ground Truth about news veridicality, at least with the tool applied to a synthetic dataset in the Skill-based simulation. In the future, it may be interesting to apply the tool to a real dataset that includes author posts and their skills, which is currently unavailable. The code in the simulation is available on GitHub SimulationFakeNews²⁰.

²⁰<https://github.com/caltr98/SimulationFakeNews>

Table 8.2: Table showing the measures collected from the Cascade Tree representing posts sharing. All averaged in 10 trials with 1000 epochs execution each. **D** indicates that it is the Skill-based simulation on the synthetic dataset, with the number of Checkers-min skills matching on the first column. **GT** indicates the Ground Truth simulation, with the number of Checkers. The 0 checkers case is equal for both simulations

Averages $\pm$ stdev /simul. type	Tree Max Depth Fake	Tree Max Breadth Fake	Tree Size Fake	Tree Max Depth True	Tree Max Breadth True	Tree Size True
10-1 D	18.3 $\pm$ 2.16	4.8 $\pm$ 0.63	2.19 $\pm$ 0.03	27.2 $\pm$ 1.55	5.5 $\pm$ 0.53	4.40 $\pm$ 0.04
10-3 D	16.0 $\pm$ 2.67	4.8 $\pm$ 0.63	2.07 $\pm$ 0.03	25.8 $\pm$ 1.48	5.7 $\pm$ 0.48	4.33 $\pm$ 0.04
30-1 D	18.2 $\pm$ 1.69	4.4 $\pm$ 0.52	1.99 $\pm$ 0.03	27.9 $\pm$ 1.79	5.8 $\pm$ 0.42	4.21 $\pm$ 0.03
50-1 D	20.0 $\pm$ 2.79	4.7 $\pm$ 0.48	1.79 $\pm$ 0.05	27.6 $\pm$ 2.01	5.9 $\pm$ 0.32	3.95 $\pm$ 0.01
70-1 D	20.1 $\pm$ 2.47	4.4 $\pm$ 0.70	1.52 $\pm$ 0.07	27.7 $\pm$ 1.57	5.7 $\pm$ 0.67	3.71 $\pm$ 0.02
0 GT	18.6 $\pm$ 1.78	5.3 $\pm$ 0.48	2.34 $\pm$ 0.02	27.2 $\pm$ 1.48	5.6 $\pm$ 0.52	4.46 $\pm$ 0.04
10 GT	14.9 $\pm$ 1.20	4.7 $\pm$ 0.48	2.07 $\pm$ 0.04	27.5 $\pm$ 2.88	5.6 $\pm$ 0.52	4.43 $\pm$ 0.06
30 GT	10.9 $\pm$ 1.73	4.1 $\pm$ 0.57	1.68 $\pm$ 0.02	28.5 $\pm$ 2.37	5.7 $\pm$ 0.48	4.26 $\pm$ 0.04
50 GT	7.2 $\pm$ 0.63	3.7 $\pm$ 0.48	1.39 $\pm$ 0.04	26.4 $\pm$ 1.51	5.9 $\pm$ 0.57	3.99 $\pm$ 0.04
70 GT	5.7 $\pm$ 0.67	2.8 $\pm$ 0.42	1.22 $\pm$ 0.03	28.0 $\pm$ 2.36	5.8 $\pm$ 0.42	3.71 $\pm$ 0.02

8.6 Performance Evaluation

In this section, we provide a set of benchmark tests on the critical operations performed by the CAVS system and the surrounding actors, such as the Author. The evaluation focuses on the performance of *skills matching, which is the most crucial operation for the functioning of the system*, considering that it is a multi-step process with first keyword extraction and then mapping. As such, we measure the performance

Table 8.3: This Table highlights the number of reshared posts and fake reshared posts of users, and the users' states (**Recovered**, **Infectious**, **Susceptible**) at the end of the simulation. All averaged in 10 trials with 1000 epochs execution each. **D** indicates that it is the Skill-Based simulation on the synthetic dataset, with the number of Checkers-min skills matching on the first column. **GT** indicates the Ground Truth simulation, with the number of Checkers

Averages $\pm$ stdev /simul. type	n.ReShared Posts	n.Fake ReShared Posts	End Rec.	End Infect	End Suscept.
10-1 D	69959.5 $\pm$ 134.4	18214.0 $\pm$ 483.9	23.4 $\pm$ 3.9	59.8 $\pm$ 6.9	16.8 $\pm$ 5.7
10-3 D	69999.2 $\pm$ 94.4	15990.9 $\pm$ 510.4	26.4 $\pm$ 2.9	54.1 $\pm$ 4.5	19.5 $\pm$ 4.9
30-1 D	69988.2 $\pm$ 152.4	12309.2 $\pm$ 455.9	37.7 $\pm$ 3.5	45.6 $\pm$ 6.8	16.7 $\pm$ 4.0
50-1 D	69951.7 $\pm$ 110.2	7199.8 $\pm$ 487.3	52.0 $\pm$ 7.3	34.8 $\pm$ 9.7	13.2 $\pm$ 4.6
70-1 D	70028.7 $\pm$ 128.5	2766.3 $\pm$ 397.1	74.5 $\pm$ 2.7	16.3 $\pm$ 4.1	9.2 $\pm$ 3.0
0 GT D	69938.5 $\pm$ 184.6	22521.9 $\pm$ 275.4	20.2 $\pm$ 3.4	61.4 $\pm$ 7.6	18.4 $\pm$ 5.8
10 GT	69964.7 $\pm$ 89.9	15984.6 $\pm$ 673.8	26.6 $\pm$ 3.7	53.5 $\pm$ 4.4	19.9 $\pm$ 4.7
30 GT	69979.3 $\pm$ 120.0	7660.6 $\pm$ 276.2	40.8 $\pm$ 2.5	43.1 $\pm$ 4.6	16.1 $\pm$ 3.7
50 GT	69972.9 $\pm$ 124.5	3072.0 $\pm$ 299.1	56.7 $\pm$ 1.5	27.1 $\pm$ 3.1	16.2 $\pm$ 3.6
70 GT	70029.6 $\pm$ 143.0	1028.0 $\pm$ 150.3	74.4 $\pm$ 2.2	15.4 $\pm$ 2.1	10.2 $\pm$ 2.7

of the AI tools. However, we also take into account the performance metrics of Veramo when generating VC and verifying them with different numbers of attributes.

The tests were conducted on a Lenovo Legion 5 15ARH7H with processor AMD Ryzen 7 6800H CPU having eight cores with 16 threads (by Simultaneous Multithreading), and as GPU an NVIDIA® GeForce RTX™ 3060 Laptop GPU.

The AI tools run locally and are not configured to work on GPUs; their performance is based on CPU execution time. In the following bulleted list, we highlight what tools are executed locally, remotely, or in other particular cases.

- Keyword extraction using GPT-3.5(LLM model): Executed remotely by calling the OpenAI APIs.
- Keyword extraction using only KeyBert (BERT model): Executed locally on the machine.
- Skill mapping using Nesta Skill Extractor: Executed locally, involving mapping keywords to skills within the ESCO ontology.
- Veramo VC issuing: Executed locally, without requiring a connection to the Ethereum Mainnet Blockchain.
- Veramo VC and VP verification: Requires a connection to the Ethereum Mainnet via Infura for retrieving public keys for verification.

For keyword extraction, the LLM model used is GPT-3.5, used by calling the OpenAI APIs as stated in Section 4.3; the keyword extraction with only KeyBert and skill mapping was instead executed just locally.

For what concerns Veramo, please note that the issuing of VCs does not entail a connection to the Ethereum Mainnet Blockchain, but the verification of VCs and VPs does, as stated in section 4.1; we tested the retrieval of the Issuer DID public key to verify a VC and also the holder DID public key to verify a VP by using Infura. The test on Veramo shows the complexity of issuing VCs and VPs with an increasing number of attributes, which gives us an idea of the performance in creating ESCO Skill VC and VC Statement along with the same increasing complexity for VP made from a holder claim to assess VP Diffusion complexity.

The call to the services for performance testing can be seen in a test file on GitHub ²⁰. We see the performance evaluation of the keyword extraction tool in diverse configurations in Table 8.4; the test was performed on an article from the synthetic dataset we produced for the Skill-based simulations, made of 630 characters and 80 words. The operation was executed 15 times, and an average is presented. What can be seen is

²⁰https://github.com/caltr98/CAVS/blob/master/performance_test.rest

Table 8.4: Table showing the average keywords extraction time (in seconds) for 630 characters (80 words) and 1280 characters (170 words) across 15 trials.

Extract Engine (15 trials)	630 characters (80 words) (seconds)	1280 characters (170 words) (seconds)
LLM	0.6913 ± 0.1865	0.7001 ± 0.1444
BERT	1.9385 ± 0.1982	2.0726 ± 0.1201
BERT + LLM	2.6312 ± 0.4756	2.6078 ± 0.2342

that the calls OpenAI APIs were faster than local execution with KeyBert.

The Nesta UK Skill (ojd_daps_skills) extractor tool was tested on the

Table 8.5: Table showing the average time (in seconds) for mapping to skills different numbers of keywords across five trials.

N.Keywords (5 trials)	Time in seconds (Average $\pm$ Stdev)
5	20.9498 ± 0.6424
10	20.6783 ± 0.1821
15	21.2301 ± 0.0835

performance of its mapping capabilities. In particular, the mapping tested was from keywords to the skills representation in the ESCO ontology. We may see the results in Table 8.5 averaging in 5 trials the mapping for increasing the number of keywords to skills. We notice that the time taken is consistently around 20 seconds, independent of the number of keywords, and we notice that this is the real bottleneck of the CAVS system, which requires further efficiency improvements.

For testing Veramo capabilities, we see the test of creating VC and verifying a VC based on a DID on the Ethereum Mainnet with an in-

creasing number of attributes in Table 8.6 averaged in 50 trials; we note that the time is much less than the time taken by AI tools as here we are in the order of milliseconds. We then see the time to create and

Table 8.6: Table showing the average time (in milliseconds) for different numbers of attributes across 50 trials of VC issue and verification.

N. of Attributes	Issuing Time (ms)	Verification Time (ms)
10	11.9618 $\pm$ 6.0303	398.6447 $\pm$ 90.2456
20	13.8403 $\pm$ 6.6283	419.9820 $\pm$ 71.1129
50	29.8172 $\pm$ 13.7308	411.5480 $\pm$ 99.5790

verify a VP, with inside the process of creating a VC with holder claims self-issued, which is fundamental to validate the performance with VP Diffusion issuing by the Spreader actors and verification from users, which were explained in the section 5.5.1. Table 8.7 shows this on VP with increasingly more attributes in a VC. The VP created and verified has one VC, for which the signature is verified, and has an additional signature, which is the one of the VP done by the holder. At last, we

Table 8.7: Table showing the average time (in milliseconds) for different numbers of attributes across 50 trials of creating a VP with a VC self-issued and verifying it.

N. of Attributes	Issuing Time (ms)	Verification Time (ms)
10	46.7599 $\pm$ 3.5011	387.5218 $\pm$ 91.1121
20	50.4558 $\pm$ 3.7031	380.7241 $\pm$ 56.3552
50	59.9865 $\pm$ 6.6332	397.4148 $\pm$ 88.3549

present in Figure 8.8 the average time required for an Author to request and receive a VC Statement issued by CAVS, the time to verify a VC Statement for a Reader, and the verification of a VP Diffusion of type "diffusion" generated by a Spreader and verified by a Reader. In particular, the Author, in this instance, presented 3 ESCO VC Skills with 12, 11, and 11 skills embedded and presented a statement to CAVS. The statement provided has a length in characters of 788 and a length in words of 101. The engine for the extraction of keywords is KeyBert. The matched skills were 17, with some multiplying from the 3 ESCO VC Skills, resulting in 20 matched skills. Consider that the request to CAVS of VC Statement is a multi-step process where first keywords are extracted from the VC

Statement, then are mapped to Skill, and ultimately, there is a match between the skills in the ESCO VC Skill. Afterwards, the ESCO VC Skill matched (all 3) are verified.

Table 8.8: Table showing the average time (in seconds) for issuing a VC Statement for a statement of length in characters of 788 and a length in words of 101. Furthermore, the VC Statement verification and the verification of a VP Diffusion of type "diffusion" is shown. The times are averaged on 5 requests.

Metric	VC Statement issuing (seconds)	VC Statement Verify (seconds)	VP Diffusion Verify (seconds)
Average (seconds)	30.207 $\pm$ 1.482	0.447 $\pm$ 0.179	0.644 $\pm$ 0.087

8.7 Security Analysis

In our attack model, we will assume that all actors involved, well configure their cryptography tools. In our cases, the DIDs key pairs (public key/ private key) created with DID method DID:ETHR, which we used in our system.

We see as possible attackers all the users involved in the CAVS system with which issue, hold, and verify VCs: Skills Certificate Issuer(Certifier), Author, Reader.

We assume that our CAVS system legitimately matches the skills provided by an author and the article ones, outputting the result of the AI models used without tampering with them, any CAVS misbehavior such as unmatched skills are considered issues of the AI tools below and their configuration. All results of the AI tools are reproducible since the VC Statement has a field that states the models used to process an article, as we indicated in subsection 6.0.2.

We will refer to the requisites A1,A2,R1,R2,S1,S2 of section 5.3 from this section to the end of the chapter.

- **Attack 1**
 - **Attackers:** AuthorX

- **Scenario:** AuthorX tries to steal the VC Statement to present it in one of their posts.
- **Mitigation:** The VC Statement holds the URL of the website holding the statement. Therefore, any attempt to use it to refer to another statement would be futile, providing a strong deterrent against such attacks. This aligns with requisite **(S2)**, as the source of a statement is indicated.
- **Attack 2**
 - **Attackers:** AuthorX
 - **Scenario:** AuthorX writes a statement that matches some of their skills. Afterward, it modifies its contents to create a statement different from the original one.
 - **Mitigation:** The VC Statement holds the keywords, their synonyms, and the general concepts extracted from the text, from which it extracts the skills. CAVS system includes all of those in each VC Statement. Providing as much information as possible entails a strong defense against such attacks as we allow verifiers and the ability to run the same keyword extraction tools and skill mapping tools specified in the VC Statement. The keyword extraction tool, skill mapping tool information, and a hash of the model are also provided, as stated in subsection 6.0.2, assisting individual execution from a verifier. This is a strong requirement for requisite **(R2)**, providing the means for the Reader to make their judgments.
- **Attack 3**
 - **Attacker:** AuthorX
 - **Scenario:** AuthorX writes **a portion of the statement** that includes all the necessary information to extract skills it owns but then proceeds to talk about something else.
 - **Mitigation:** The user may see the matching skills, and as such, it can reach their conclusions on where the Author was authoritative. As before, it is fundamental for **(R2)** to read for the readers to make their judgment.
 - **Further solutions:** We may include in the CAVS system

a tool capable of recognizing different contexts and adding indicators of what contexts are satisfied or not inside the VC Statement. This would satisfy the requisite **(R1)**

8.8 Trust Issue mitigations with CAVS

In this section, we tackle a problem that is effectively present in all SSI-based systems to various degrees. The problem concerns the trust that verifiers may have on certificate issuers.

Providing that a VC is true, there is the issue of the identity of the VC creator, given that in SSI, we use DIDs to express the identity of issuers and holders (fundamental for requisite **(A2)**), it is necessary for any system to link the issuers DID to a known identity, necessary for requisite **(A2)** and **(S1)**.

Let us see a possible attack scenario revolving around the trust issue **Attack 4**:

- **Attackers:** AuthorX, CertificatorY
- **Scenario:** A premise is that AuthorX and CertificatorY may be the same human person. We will describe the scenario in which CertificatorY may be a false or non-relevant institution; for example, it may be AuthorX rather than a University. CertificatorY may provide AuthorX with an ESCO Skills VC with precisely the skills matching a statement AuthorX wants to publish. This scenario violates **(S1)** (credibility of the statement for the Spreader) and R2 (the Reader's need for trust in the certificates ESCO Skill VC issuers authorities).

Note that the VC Statement holds the ESCO Skill VCs with skills found relevant by CAVS to the statement. CAVS or an OSN user can read them and give them a rank based on who the Issuer is of the ESCO Skill VC; this helps in *mitigating the violation of (R2)*.

To proceed with the mitigation, we consider various solutions:

8.8.1 DID:WEB and TLS/SSL Certificate Authorities

As stated in subsection 2.5.1, we have various DID methods. In CAVS, we mainly proposed the usage of DID:ETHR, particularly in the DeSo integration in chapter 7. However, it is entirely possible to use other DID methods, such as DID:WEB. DID:WEB method stores a DID document inside a webpage accessible with the HTTP(S) protocol, rather than a Blockchain.

Given this, we may have issuers of certificates using DID:WEB and exploit the certificate given by a Certification Authority of a website hosted on a server with HTTPS protocol to verify the Issuer's identity.

Provided that ESCO Skill VCs issuers utilize DID:WEB. We may extend CAVS to fetch the TLS/SSL certificates, with the issue that we must extract information about the website domain to understand better what type of organization is issuing the ESCO Skill VC.

For this task, we may use the Whois protocol to obtain information about a domain and then process said information with an AI tool such as GPT, which, given its *general knowledge*, might be used to give a score to the Issuer.

Consequently, the VC Statement would hold a rating for each skill about their issuers.

At the moment, we discard this solution to concentrate on our solutions based on the Ethereum blockchain as a registry for DID using the DID:ETHR method.

8.8.2 CAVS Centralized solution: known issuers list

CAVS includes a list of known certificate issuers and a score on their reputation(trust). As seen in section 6.1.2, the system manager may input the DID and the score of a particular certificate issuer. Whenever CAVS finds an issuer in said list, if a skill matches the ones in the statement, it places the issuer score alongside the ESCO Skill VC in the VC Statement. This solution provides a centralized and efficient manner to handle the problem. However, it requires manually creating a database of trusted issuers. It supports the requisites **(R2)** (ensuring the Reader can trust the certificates ESCO Skill VC issuers), and **(S1)** (ensuring the Spreader shares trustworthy content).

In Figure 8.11, we see the trust relationships between statements read-

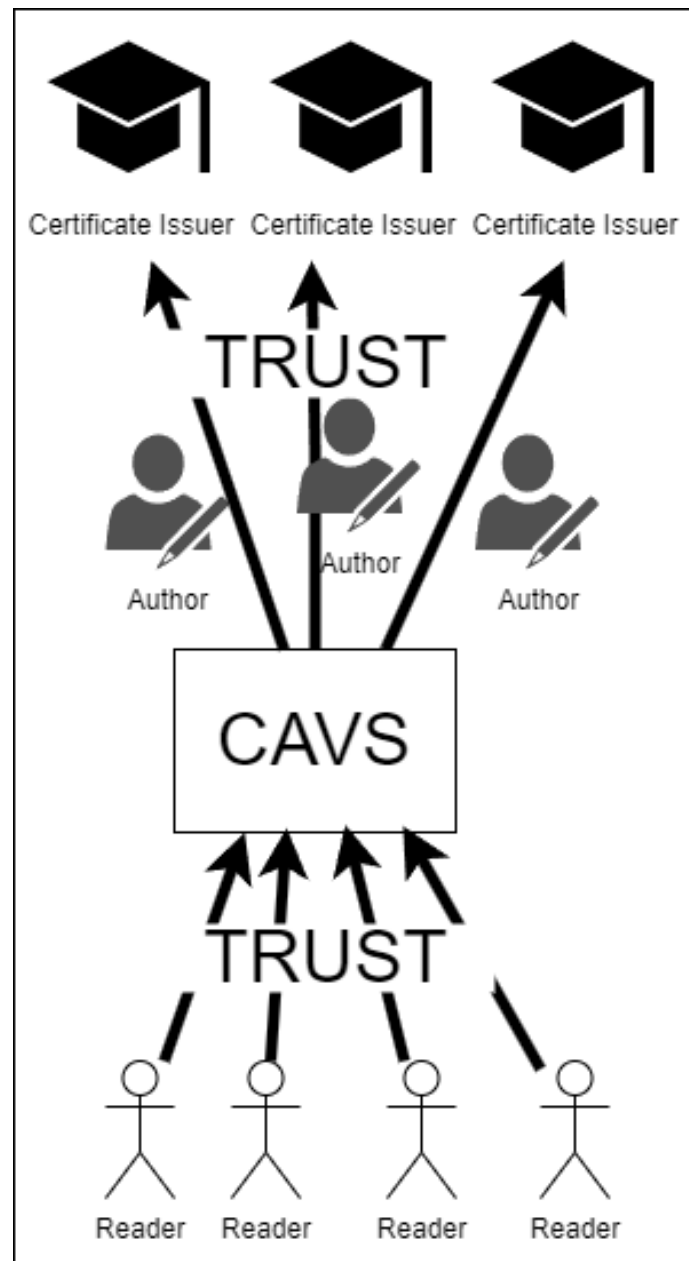


Figure 8.11: A representation of the trust from readers actors to CAVS, and from CAVS to Certificate Issuers. Certificate Issuers provide ESCO Certificate VC to Authors.

ers to CAVS and CAVS to certificate issuers. Practically, this solution relies on CAVS being a centralization point, as the figure shows. However, it is a more straightforward solution than the more decentralized

one that we will see in the next section 8.9, given that this solution does not burden the readers in making their list of certificate issuers DIDs but allows them to rely on CAVS.

8.9 Web of Trust and Alternative System: CMCS

A decentralized alternative to a centralized solution of CAVS giving trust score is using the Web of Trust[17], where users rate how much they trust one another. This model generates a graph where each user is represented as a node, and the edges between nodes represent trust relationships. A user trusts another user if a path exists from their node to the other's node [72]. Unlike the centralized CAVS system, where trust depends on a curated list of certificate issuers, the Web of Trust relies on user relationships. The verifier, acting as a statement reader, determines trust based on the graph of trust relationships. This user-centric approach eliminates central management and allows for dynamic, peer-based trust assessment.

Generally, we expect the users to declare themselves the trust relationships. But additionally, in the literature, there have been proposed systems that assign a trust score between users in an OSN based on the interaction that they had [73], this too may be a way of modeling the trust relationship that is performed by assignments of the score from a service using AI tools (e.g., Neuro-Behavioral Module)[73].

8.9.1 Decentralized Web of Trust System Architecture

We propose an alternative architecture to the centralized CAVS. We define this as the Competence Matching Certification System (CMCS), which we model to conceive trust as decentralized. With CMCS, we remove the need for a central system to match ESCO skills with statements and rank certificate issuers. We intend for Multiple CMCS instances to exist in this model and allow readers to choose which instance they trust. Figure 8.12 highlights how trust can be distributed across multiple CMCS services, allowing users to select which one to trust for skill matching and certification.

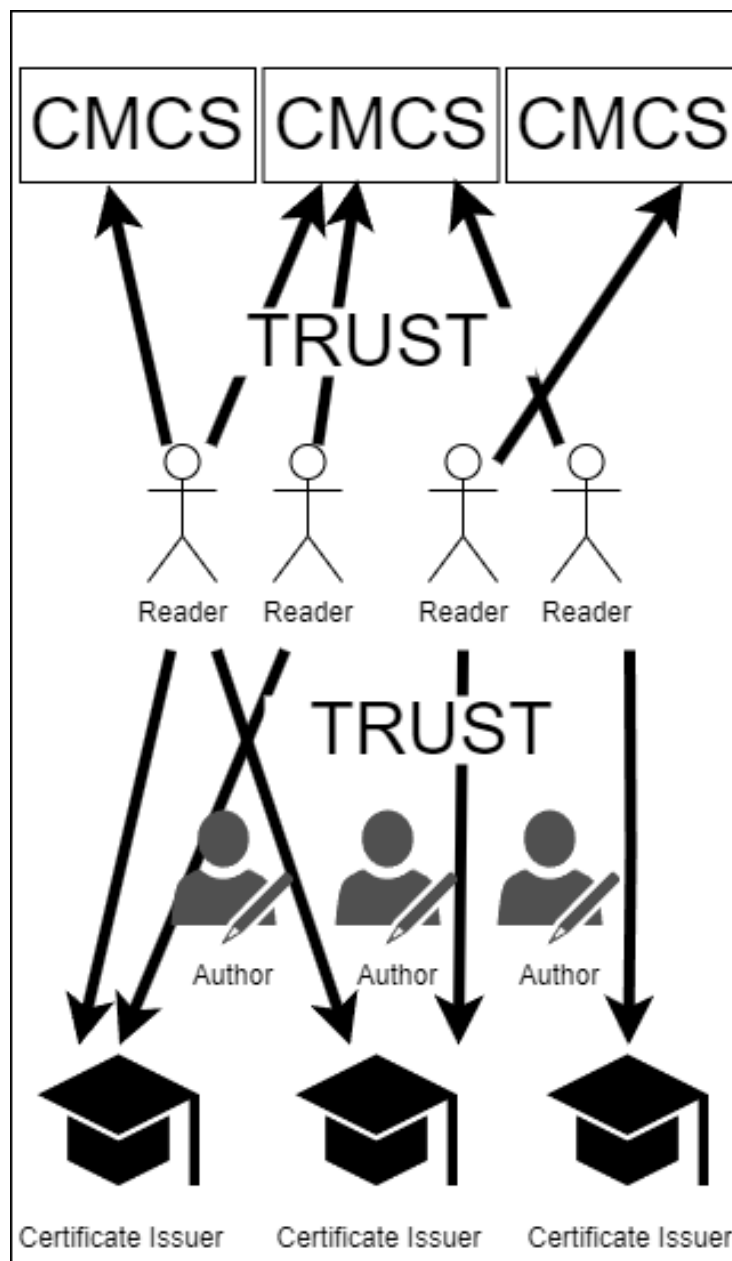


Figure 8.12: Diagram representing the trust relationships from readers to multiple CMCS instances for skill matching and from readers to Certificate Issuers. Certificate Issuers provide ESCO Certificate VC to authors.

The process of creating a post and assessing its trustworthiness can be seen in Figure 8.13 and is described as follows:

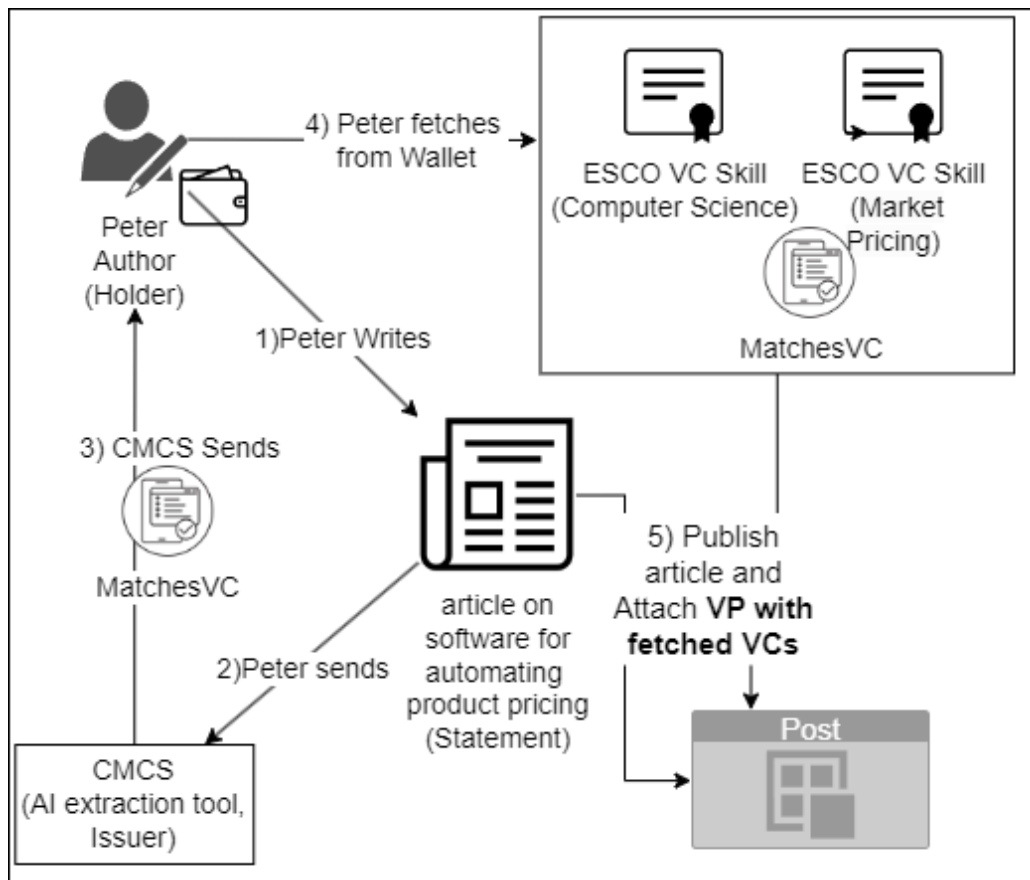


Figure 8.13: Diagram based on the use case in section 5.2.1, showing how the Author of a post requires from CMCS the VC Matches that provide matching between the statement and skills. Afterward, the Author attaches a VP with the VC Matches and relevant VC Skills.

1. The Author submits a statement and **post's URL** to CMCS, which matches with skills relevant to the statement and issues to the Author's DID a **VC Matches** holding the matching skills, the keywords of the statement and the URL of the post to reference it.
2. The Author generates a VP using both the **VC Matches** issued to their DID and the relevant **ESCO VC Skill**. This VP is displayed alongside the post as a QR code or JSON.
3. The Reader of the post verifies all the VCs presented with the post and decides the trustworthiness of both the ESCO Skill issuers and the CMCS instance that generated the VC Matches.

The Web of Trust system remains highly relevant for ranking issuers in this more decentralized approach, as multiple CMCS services can provide VC Matches. The trust placed in a specific CMCS service can be derived from the OSN's existing trust relationships.

8.9.2 Implementation in DeSo

In the DeSo implementation, we adapted the Web of Trust concept using the "following" relationships between users. Our extension to the DeSo front end retrieves the list of accounts followed by the logged-in user. The social graph of "following" relationships is explored up to a depth of 2, allowing the verifier to assess trustworthiness based on connections between the statement author and their followers, as well as the followers of those followers. Expanding deeper than this becomes impractical due to the principle of 7 degrees of separation, which implies that all users could be connected at greater depths. Studies, such as those on Facebook, show the average degrees of separation range between 4.4 and 5.7 [74].

8.9.3 DID-Based Alternative for Post-Level Trust

As an independent alternative, with a different mapping of DID to entities, we propose assigning a DID to each post containing a statement. This approach operates independently of CMCS and focuses on the post as the subject of trust.

This alternative ensures a greater emphasis on the satisfaction of requisite **(S2)** traceability of individual posts by DID rather than URL, and it supports **(A2)** allowing the Author to establish credibility without revealing PII, said PII was introduced in section 5.1. This is done by focusing on the post rather than the Author.

- DID subject: the post containing the statement.
- DID controller: the post's Author, who signs on behalf of the post (DID subject). **This enables the creation of Verifiable Presentations (VPs) using Verifiable Credentials (VCs) issued for the post.**

To minimize overhead, we use the DID method DID:KEY (introduced in Section 2.5.1), which allows for on-the-fly creation of DIDs without the

need for blockchain registration. This approach is optimal for handling billions of posts in an OSN, as its Author, who acts as the controller, generates the post's DID dynamically

In this approach, creating a post and assessing considers the VC

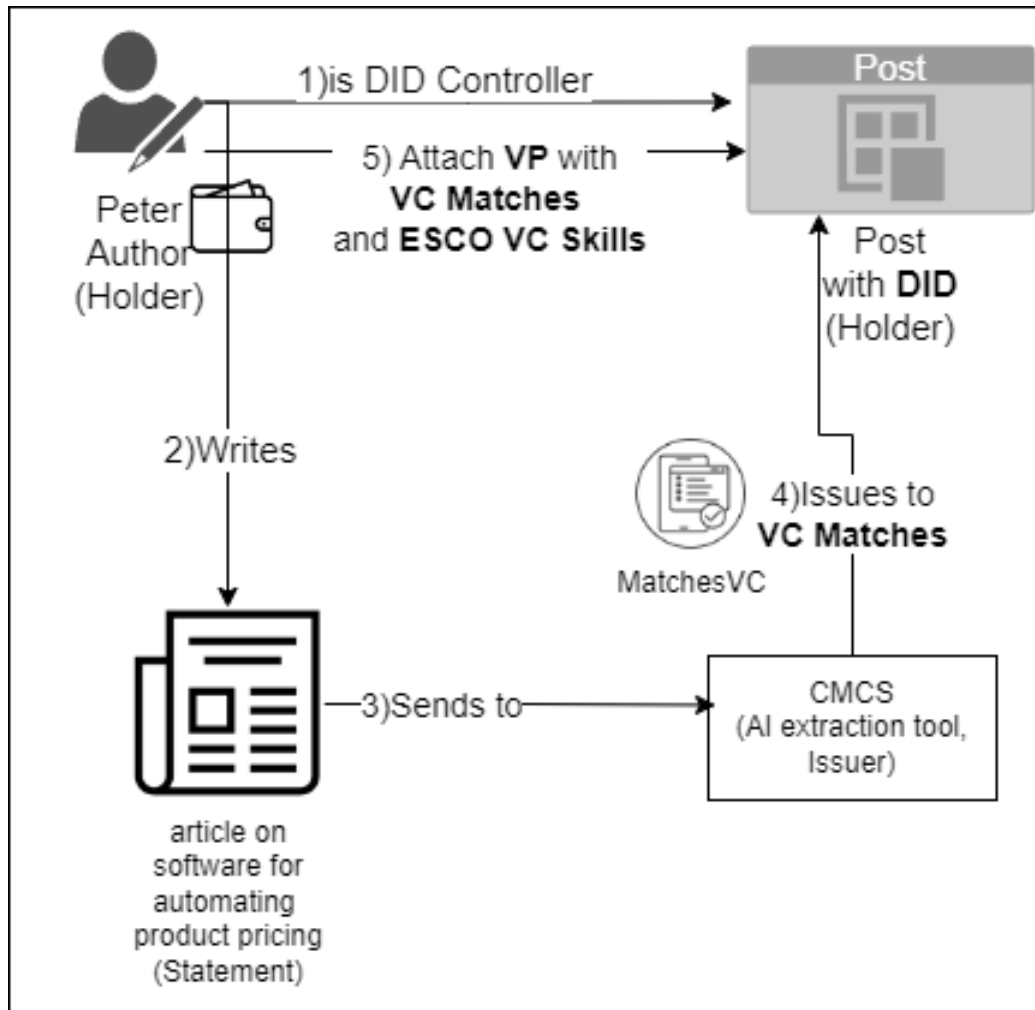


Figure 8.14: A representation of the use case in section 5.2.1, showing how the Author of a post requests from CMCS the VC Matches that provide matching between the statement and skills. In this case, CMCS issues it to the DID of the post.

Matches as being issued towards the post as holder DID. It can be seen in Figure 8.14 and in detail as follows:

1. The Author submits a statement and the **post's DID** to CMCS, which matches the skills with the statement relevant skills and issues a **VC Matches** towards the post's DID.
2. The Author generates a VP using both the **VC Matches** issued to the post's DID and the relevant **ESCO VC Skill**. They can do that as they are the DID controller of the post's DID. Furthermore, this VP is displayed alongside the post as a QR code or JSON.
3. The Reader of the post verifies all the VCs presented with the post and decides the trustworthiness of both the ESCO Skill issuers and the CMCS instance that generated the VC Matches.

This instance of the SSI mapping enhances the integrity and trustworthiness of the process, as the VC matches are issued specifically for the post rather than the Author. This ensures that any Reader can verify that the VC matches were generated precisely for the stated post. For example, this approach safeguards the Reader from malicious CMCS services that might embed a different link than the post's actual one within the VC matches. The post's authenticity is assured by referencing the post through its DID rather than just the VC matches' content.

Chapter 9

Conclusion and future works

In this thesis, we tackled the problem of Social Content trustworthiness, mitigating false news diffusion in OSN using the novel technology of Self-Sovereign Identity. In this way, we provided a privacy-preserving solution for assessing the authoritativeness of an author on the topic of the posts they made. To do so, we modeled the diffusion of statements via posts on OSN and defined Verifiable Credential (VC) and Verifiable Presentation (VP) for this task. Furthermore, we explored the usage of AI tools, which made it possible to extract the topic of a statement and match it with the VCs of a statement author. To achieve this, we have provided a set of definitions for fake news as we consider it as statements in OSN that possibly malicious or naive users make and users who do not double-check the content and authoritativeness of the source share.

Our proposed system, CAVS: Credibility Assessment and Verification System, is a significant step towards mitigating the fake news problem in a privacy-preserving manner. It aims to preserve freedom of expression while establishing the authoritativeness of those who share a statement. A system of this type can be a potential game-changer in the fight against fake news.

We discussed Self-Sovereign Identity (SSI), which is the paradigm that is central to CAVS. We provided the definitions of VC and decentralized identifiers (DID), and we note that the standard is still evolving continuously. We specified other used specifications, such as Verifiable Credentials with JSON Object Signing and Encryption. We use DID meth-

ods such as DID:KEY, DID:WEB, and, in particular, DID:ETHR, based on the Ethereum Blockchain. We used DID both for identifying humans and services, as CAVS produced VC with its DID.

We stress that Blockchain is fundamental as a technological enabler for SSI. We discussed in this distributed framework Veramo as an implementation of SSI and IPFS, which are tools that we found necessary to use.

To enable CAVS to work, we identified the necessary AI tools to extract the skills that a statement source must possess. We investigated the usage of the BERT model and the possibility of using models such as GPT, where the GPT usage was essential to us for the generation of a synthetic dataset of person profiles and articles that they wrote.

Furthermore, we discussed the tools that implement those models and that we used in CAVS which are OpenAI APIs, the KeyBert tool and NESTA Skills Extractor library (also called `ojd_daps_skill`).

In particular, we used the ESCO ontology of skills and competencies to provide an authoritative ontology to use, which may correspond in the future to what certificates issued in the EU are and is an indication of the names to use by an authoritative source such as the European Commission.

We developed CAVS with a highly adaptable microservice architecture in mind, leveraging Docker to create independent components connected by REST API calls. This architecture allowed for the easy integration of extensions and demonstrated the system's potential for scalability. To underscore the adaptability of CAVS, we also integrated it into DeSo, a social network based on blockchain technology.

We highlighted the limitations of SSI, such as the problem of establishing a root of Trust, and we provided some solutions, such as creating a Web of Trust between users of a social network and trusting friends. A simulation we made showed saw in a simulated test how fact-checkers who know the ground truth about news truthfulness compare to fact-checkers who decide the authoritativeness of authors by their skills, which CAVS includes in the VC Statements it produced. We saw that assessing the credibility of statements by counting the skills of the Author matching with the statement required is an improvement to not having a fact check, only slightly still less effective than having fact-checkers that know the ground truth; however, to reduce significantly the spread of

fake news it requires a large pool using CAVS and assessing credibility in this manner.

Furthermore, we note that the use of AI tools has a significant environmental impact and that they require costly processing capabilities. In section 8.9 proposed the CMCS: Competence Matching Certification System, which creates VCs for posts that hold statements that only certify the skills matches, providing the idea of a more decentralized solution than CAVS where the readers decide what issuers to trust instead of relying on CAVS.

9.1 Future Works

We identified some areas to research further and improve the CAVS system:

- Study DID with DID subjects services that leverage AI, such as GPT, with humans as DID controllers behind them
- Research on further solutions to leverage Trust in the system while preserving the privacy of all parties involved.
- Integrate the solution with different tools such as Hyperledger Aries and with other modules of Veramo that use different specifications for VC and DIDs
- Explore the usage of Selective Disclosure to provide only the necessary skills of an author to the CAVS system.
- Use and develop integrations with techniques of **Selective Disclosure**, usually based on **Zero-Knowledge proofs**, to reduce skills shared with CAVS to only the necessary ones.

SSI can be a great solution to the fake news problem while preserving privacy, and that future work and developments can make it more viable.

Bibliography

- [1] Adrien Guille et al. "Information diffusion in online social networks: a survey". In: *SIGMOD Rec.* 42.2 (July 2013), pp. 17–28. ISSN: 0163-5808. DOI: [10.1145/2503792.2503797](https://doi.org/10.1145/2503792.2503797). URL: <https://doi.org/10.1145/2503792.2503797>.
- [2] Xichen Zhang and Ali A. Ghorbani. "An overview of online fake news: Characterization, detection, and discussion". In: *Information Processing 'I&' Management* 57.2 (2020), p. 102025. ISSN: 0306-4573. DOI: <https://doi.org/10.1016/j.ipm.2019.03.004>. URL: <https://www.sciencedirect.com/science/article/pii/S0306457318306794>.
- [3] Simone Mariconda et al. "Fooling Them, Not Me?: How Fake News Affects Evaluators' Reputation Judgments and Behavioral Intentions". English. In: *Business and Society* (July 2024). ISSN: 1552-4205.
- [4] Bruce Y. Lee. "Fake Eli Lilly Twitter account claims insulin is free, stock falls 4.3%". In: *Forbes* (Sept. 2024). URL: <https://www.forbes.com/sites/brucelee/2022/11/12/fake-eli-lilly-twitter-account-claims-insulin-is-free-stock-falls-43/?sh=dda426641a3d>.
- [5] Claire Atkinson. *Fake news can cause irreversible damage to companies, sink their stock*. Sept. 2024. URL: <https://www.nbcnews.com/business/business-news/fake-news-can-cause-irreversible-damage-companies-sink-their-stock-n995436>.
- [6] Evelyn Douek. "The Rise of Content Cartels". In: *SSRN Electronic Journal* (Jan. 2020). DOI: [10.2139/ssrn.3572309](https://ssrn.com/abstract=3572309).
- [7] Karen Lumsden and Heather Morgan. "Media Framing of Trolling and Online Abuse: Silencing Strategies, Symbolic Violence and

- Victim Blaming". In: *Feminist Media Studies* iFirst (Apr. 2017). DOI: [10.1080/14680777.2017.1316755](https://doi.org/10.1080/14680777.2017.1316755).
- [8] A. Mühle et al. "A survey on essential components of a self-sovereign identity". In: *Computer Science Review* 30 (2018), pp. 80–86.
- [9] A. Tobin and D. Reed. *The Inevitable Rise of Self-Sovereign Identity*. Accessed: 2024-09-03. The Sovrin Foundation, 2017. URL: <https://sovrin.org/wp-content/uploads/2018/03/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf>.
- [10] Cameron Martel and David Rand. "Fact-checker warning labels are effective even for those who distrust fact-checkers". In: (Nov. 2023). DOI: [10.31234/osf.io/t2pmb](https://doi.org/10.31234/osf.io/t2pmb).
- [11] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014. ISBN: 9781466570269. URL: <https://www.crcpress.com/Introduction-to-Modern-Cryptography-Second-Edition/Katz-Lindell/p/book/9781466570269>.
- [12] R. L. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Commun. ACM* 21.2 (Feb. 1978), pp. 120–126. ISSN: 0001-0782. DOI: [10.1145/359340.359342](https://doi.org/10.1145/359340.359342). URL: <https://doi.org/10.1145/359340.359342>.
- [13] Andreas M Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies*. Second ed. "O'Reilly Media, Inc.", 2017.
- [14] Tatiana Revoredo. "Bitcoin Genesis Day". In: *Medium* (2023). URL: <https://tatianarevoredo.medium.com/bitcoin-genesis-day-63df91ec3ef3>.
- [15] Vitalik Buterin. *Ethereum White Paper: A Next-Generation Smart Contract and Decentralized Application Platform*. <https://ethereum.org/en/whitepaper/>. Accessed: 2024-08-27. 2014.
- [16] Gavin Wood. "Ethereum: A secure decentralised generalised transaction ledger". In: *Ethereum project yellow paper* 151 (2014). Accessed: 2024-08-27, pp. 1–32. URL: <https://ethereum.github.io/yellowpaper/paper.pdf>.
- [17] Web of Trust, Christopher Allen. *The Path to Self-Sovereign Identity*. <https://github.com/WebOfTrustInfo/self-sovereign-identity/blob/master/ThePathToSelf-SovereignIdentity.md>. Accessed: 2024-08-27. 2024.

- [18] Clemens Brunner et al. "DID and VC:Untangling Decentralized Identifiers and Verifiable Credentials for the Web of Trust". In: *Proceedings of the 2020 3rd International Conference on Blockchain Technology and Applications*. ICBTA '20. Xi'an, China: Association for Computing Machinery, 2021, pp. 61–66. ISBN: 9781450388962. DOI: [10.1145/3446983.3446992](https://doi.org/10.1145/3446983.3446992). URL: <https://doi.org/10.1145/3446983.3446992>.
- [19] Drummond Reed et al. *Decentralized Identifiers (DIDs) v1.0*. [Online; accessed 20. Aug. 2024]. July 2022. URL: <https://www.w3.org/TR/did-core>.
- [20] Christian Gribneau et al. *did:web Method Specification*. <https://w3c-ccg.github.io/did-method-web/>. Accessed: 2024-08-28.
- [21] Uport. *Ethereum DID Registry*. <https://github.com/uport-project/ethr-did-registry>. Accessed: 17-07-2024. 2024.
- [22] decentralized-identity. *ethr DID Resolver*. <https://github.com/decentralized-identity/ethr-did-resolver>. Accessed: 17-07-2024. 2024.
- [23] Dave Longley, Dmitri Zagidulin, and Manu Sporny. *DID Method Key*. Accessed: 2024-09-07. 2024. URL: <https://w3c-ccg.github.io/did-method-key/>.
- [24] Manu Sporny et al. *Verifiable Credentials Data Model v2.0*. [Online; accessed 20. Aug. 2024]. Aug. 2024. URL: <https://www.w3.org/TR/vc-data-model-2.0>.
- [25] Gabe Cohen et al. *Verifiable Credentials JSON Schema*. Accessed: 2024-08-29. 2024. URL: <https://www.w3.org/TR/vc-json-schema/#integrity-validation>.
- [26] M. Jones, J. Bradley, and N. Sakimura. *JSON Web Token (JWT)*. RFC 7519. May 2015. DOI: [10.17487/RFC7519](https://doi.org/10.17487/RFC7519). URL: <https://www.rfc-editor.org/rfc/rfc7519>.
- [27] Michael Jones et al. *Verifiable Credentials Data Integrity (JOSE/-COSE)*. Accessed: 2024-08-29. 2024. URL: <https://www.w3.org/TR/vc-jose-cose/>.
- [28] Yupeng Chang et al. *A Survey on Evaluation of Large Language Models*. 2023. arXiv: [2307.03109](https://arxiv.org/abs/2307.03109) [cs.CL]. URL: <https://arxiv.org/abs/2307.03109>.
- [29] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language*

- Models*. 3rd. Online manuscript released August 20, 2024. 2024. URL: <https://web.stanford.edu/~jurafsky/slp3/>.
- [30] Ashish Vaswani et al. "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.
- [31] Prafull Sharma and Yingbo Li. "Self-Supervised Contextual Keyword and Keyphrase Retrieval with Self-Labeling". In: *Preprints* (Aug. 2019). DOI: [10.20944/preprints201908.0073.v1](https://doi.org/10.20944/preprints201908.0073.v1). URL: <https://doi.org/10.20944/preprints201908.0073.v1>.
- [32] Rahul Sharnagat. *Named Entity Recognition: a Literature Survey*. URL: <https://api.semanticscholar.org/CorpusID:17131608>.
- [33] Jing Li et al. "A Survey on Deep Learning for Named Entity Recognition". In: *IEEE Transactions on Knowledge and Data Engineering* 34.1 (Jan. 2022), pp. 50–70. ISSN: 2326-3865. DOI: [10.1109/tkde.2020.2981314](https://doi.org/10.1109/tkde.2020.2981314). URL: <http://dx.doi.org/10.1109/TKDE.2020.2981314>.
- [34] Christian Fuchs. *Social media : a critical introduction*. English. 2nd ed. OCLC: 951226822. Los Angeles: Sage, 2017. ISBN: 978-1-4739-6682-6.
- [35] Xinyi Zhou and Reza Zafarani. "A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities". In: *ACM Comput. Surv.* 53.5 (Sept. 2020). ISSN: 0360-0300. DOI: [10.1145/3395046](https://doi.org/10.1145/3395046). URL: <https://doi.org/10.1145/3395046>.
- [36] Karishma Sharma et al. "Combating Fake News: A Survey on Identification and Mitigation Techniques". In: *ACM Trans. Intell. Syst. Technol.* 10.3 (Apr. 2019). ISSN: 2157-6904. DOI: [10.1145/3305260](https://doi.org/10.1145/3305260). URL: <https://doi.org/10.1145/3305260>.
- [37] Ke Zhou, Sanja Scepanovic, and Daniele Quercia. *Characterizing Fake News Targeting Corporations*. 2024. arXiv: [2401.02191](https://arxiv.org/abs/2401.02191) [cs.CY]. URL: <https://arxiv.org/abs/2401.02191>.
- [38] Adrien Friggeri et al. "Rumor Cascades". In: *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014* 8 (May 2014), pp. 101–110. DOI: [10.1609/icwsml.2014.8.101-110](https://doi.org/10.1609/icwsml.2014.8.101-110).
- [39] Soroush Vosoughi, Deb Roy, and Sinan Aral. "The spread of true and false news online". In: *Science* 359.6380 (2018), pp. 1146–1151. DOI: [10.1126/science.aap9559](https://doi.org/10.1126/science.aap9559).

- [40] Soroush Vosoughi, Deb Roy, and Sinan Aral. "The spread of true and false news online". In: *Science* 359.6380 (2018), pp. 1146–1151. DOI: [10.1126/science.aap9559](https://doi.org/10.1126/science.aap9559). eprint: <https://www.science.org/doi/pdf/10.1126/science.aap9559>. URL: <https://www.science.org/doi/abs/10.1126/science.aap9559>.
- [41] Mirko Franco et al. "A decentralised messaging system robust against the unauthorised forwarding of private content". In: *Future Generation Computer Systems* 145 (2023), pp. 211–222. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2023.03.025>. URL: <https://www.sciencedirect.com/science/article/pii/S0167739X23001024>.
- [42] Zakwan Jaroucheh et al. "TRUSTD: Combat Fake Content using Blockchain and Collective Signature Technologies". In: *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*. 2020, pp. 1235–1240. DOI: [10.1109/COMPSAC48688.2020.00-87](https://doi.org/10.1109/COMPSAC48688.2020.00-87).
- [43] Sakshi Dhall et al. "Blockchain-based Framework for Reducing Fake or Vicious News Spread on Social Media/Messaging Platforms". In: *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 21.1 (Nov. 2021). ISSN: 2375-4699. DOI: [10.1145/3467019](https://doi.org/10.1145/3467019). URL: <https://doi.org/10.1145/3467019>.
- [44] Maheen Unzeelah and Zulfiqar Ali Memon. "Fighting Against Fake News by Connecting Machine Learning Approaches with Web3". In: *2022 International Conference on Emerging Trends in Smart Technologies (ICETST)*. 2022, pp. 1–6. DOI: [10.1109/ICETST55735.2022.9922927](https://doi.org/10.1109/ICETST55735.2022.9922927).
- [45] Decentralized Identity Foundation. *Veramo Documentation: Introduction*. Accessed: 2024-08-30. 2024. URL: <https://veramo.io/docs/basics/introduction>.
- [46] ConsenSys, Inc. *Infura API Documentation*. Accessed: 2024-09-05. 2024. URL: <https://docs.infura.io/api>.
- [47] Daniel Norman. *A Practical Explainer: IPFS Gateways*. Accessed: 2024-09-03. June 2022. URL: <https://blog.ipfs.tech/2022-06-09-practical-explainer-ipfs-gateways-1/>.
- [48] Juan Benet. *IPFS - Content Addressed, Versioned, P2P File System*. 2014. arXiv: [1407.3561](https://arxiv.org/abs/1407.3561) [cs.NI]. URL: <https://arxiv.org/abs/1407.3561>.
- [49] IPFS Developers. *Helia*. <https://github.com/ipfs/helia>. Commit accessed: 2024-08-30. 2024.

- [50] Maarten Grootendorst. *KeyBERT*. <https://github.com/MaartenGr/KeyBERT>. 2024.
- [51] Nils Reimers Sentence Transformers. *sentence-transformers/all-MiniLM-L6-v2* · Hugging Face. [Online; accessed 22. Aug. 2024]. Aug. 2024. URL: <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>.
- [52] Wenhui Wang et al. *MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers*. 2020. arXiv: [2002.10957](https://arxiv.org/abs/2002.10957) [cs.CL]. URL: <https://arxiv.org/abs/2002.10957>.
- [53] ESCO. *Qualifications*. Accessed: 2024-09-17. 2024. URL: <https://esco.ec.europa.eu/en/classification/qualifications>.
- [54] Elizabeth Gallagher et al. *The Skills Extractor Library*. <https://www.escoe.ac.uk/the-skills-extractor-library/>. 2023.
- [55] OpenAI. *OpenAI Platform Documentation*. Accessed: 2024-09-18. 2024. URL: <https://platform.openai.com/docs>.
- [56] Michael V. Reiss. *Testing the Reliability of ChatGPT for Text Annotation and Classification: A Cautionary Remark*. 2023. arXiv: [2304.11085](https://arxiv.org/abs/2304.11085) [cs.CL]. URL: <https://arxiv.org/abs/2304.11085>.
- [57] ER Velasco. *Roaring Kitty Returns With \$180 Million GameStop Investment, Sends Stock Price Up By 70%*. 2024. URL: <https://thedeepdive.ca/roaring-kitty-returns-with-180-million-gamestop-investment-sends-stock-price-up-by-70/> (visited on 09/19/2024).
- [58] Erika McCallister, Timothy Grance, and Karen Scarfone. *Guide to Protecting the Confidentiality of Personally Identifiable Information (PII)*. en. Apr. 2010. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=904990.
- [59] Zoë Schiffer. “Timnit Gebru, a leading AI ethics researcher, says she was fired by Google”. In: *The Verge* (2020). Accessed: 2024-09-20. URL: <https://www.theverge.com/2020/12/7/22158501/timnit-gebru-team-google-public-statement-fired>.
- [60] Stephanie Hegarty. “China’s new law allows police to act on foreign and domestic threats”. In: *BBC* (2020). Accessed: 2024-09-20. URL: <https://www.bbc.com/news/world-asia-china-51364382>.
- [61] Barbara Guidi. “When Blockchain meets Online Social Networks”. In: *Pervasive and Mobile Computing* 62 (2020), p. 101131. ISSN: 1574-1192. DOI: <https://doi.org/10.1016/j.pmcj.2020>.

101131. URL: <https://www.sciencedirect.com/science/article/pii/S1574119220300195>.
- [62] DeSo Developers. *DeSo Blockchain Documentation*. <https://docs.deso.org/deso-blockchain/>. Accessed: 2024-08-24. 2024.
- [63] Crypto Safety First. *Derivation Path*. <https://cryptosafetyfirst.com/derivation-path/>. Accessed: 2024-09-14. 2024.
- [64] DeSo developers. *DeSo mnemonics*. 2024. URL: <https://offline.deso.com/#mnemonic> (visited on 08/24/2024).
- [65] Marek Palatinus, Pavol Rusnak, and Aaron Voisine ands Sean Bowe. *Mnemonic code for generating deterministic keys*. <https://github.com/uport-project/ethr-did-registry>. Accessed: 17-07-2024. 2013. URL: <https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki>.
- [66] DeSo Blog. *How MetaMask Works*. Accessed: 2024-08-24. 2024. URL: <https://blog.deso.com/blog/how-metamask-works>.
- [67] DeSo Developers. *deso-js*. <https://github.com/deso-protocol/deso-js>. Version 8ab3cb3fea806a61a7e644f1becc9b1c3f092b85. 2024. URL: <https://github.com/deso-protocol/deso-js> (visited on 08/24/2024).
- [68] Vitalik Buterin and Alex Van de Sande. *ERC-55: Mixed-case checksum address encoding*. <https://github.com/ethereum/ercs/blob/master/ERCS/erc-55.md>. 2016. (Visited on 08/24/2024).
- [69] DeSo Developers. *DeSo Protocol Frontend*. <https://github.com/deso-protocol/frontend>. 2024. URL: <https://github.com/deso-protocol/frontend> (visited on 08/24/2024).
- [70] Veniamin Veselovsky et al. *Generating Faithful Synthetic Data with Large Language Models: A Case Study in Computational Social Science*. 2023. arXiv: 2305.15041 [cs.CL]. URL: <https://arxiv.org/abs/2305.15041>.
- [71] Aleena Concepcion and Charlle Sy. "Modeling the Spread of Fake News on Social Networking Sites Using the System Dynamics Approach". In: *ASEAN Engineering Journal* 13.4 (Oct. 2023), pp. 69–78. DOI: 10.11113/aej.v13.19251. URL: <https://journals.utm.my/aej/article/view/19251>.
- [72] Andrea De Salve et al. "A multi-layer trust framework for Self Sovereign Identity on blockchain". In: *Online Social Networks and Media* 37-38 (2023), p. 100265. ISSN: 2468-6964. DOI: <https://doi.org/10.1016/j.osnem.2023.100265>. URL: <https://www.sciencedirect.com/science/article/pii/S2468696423000241>.

- [73] Barbara Guidi et al. "SentiTrust: A New Trust Model for Decentralized Online Social Media". In: *IEEE Access* 11 (2023), pp. 53401–53417. DOI: [10.1109/ACCESS.2023.3281194](https://doi.org/10.1109/ACCESS.2023.3281194).
- [74] Lars Backstrom et al. *Four Degrees of Separation*. 2012. arXiv: [1111.4570 \[cs.SI\]](https://arxiv.org/abs/1111.4570). URL: <https://arxiv.org/abs/1111.4570>.