Selective Content Disclosure using Zero-Knowledge Proofs

Nikos Fotiou, Vasilis Kalos, Yannis Thomas, George Xylomenos, Vasilios A. Siris, George C. Polyzos Mobile Multimedia Laboratory
Department of Informatics, School of Information Sciences and Technology
Athens University of Economics and Business, Greece
{fotiou,kalos20,thomasi,xgeorge,vsiris,polyzos}@aueb.gr

***Abstract*—Information-Centric Networking (ICN) is a Next Generation Internet architecture that facilitates content sharing. ICN natively supports content multi-sourcing, allowing content items to be stored in multiple storage nodes. In order to ensure data integrity, data owners can sign their content items. This, however, prevents storage nodes from sharing partial content items. We present a data sharing scheme where data owners store structured data items (e.g., IoT measurements) in semi-trusted storage nodes. We allow data consumers to express interest for a portion of a data item and we enable storage nodes to "hide" the remaining item without invalidating its integrity. We achieve our goal by leveraging BBS+ digital signatures that support selective data disclosure through Zero-Knowledge Proofs. We define a protocol for data owners to issue authorizations in the form of Verifiable Credentials, which indicate which parts of the data a consumer is allowed to access, and a protocol for consumers to send these authorizations inside ICN Interests. This allows storage nodes to implement fine grain access control, without having access to the secrets of the data owners, while data consumers can still verify the authenticity and integrity of the partially revealed data. In addition to its security advantages, our solution requires significantly less storage and communication overhead compared to an approach that relies on commonly used digital signature algorithms.**

***Index Terms*—BBS+, ICN, Verifiable Credentials**

I. Introduction

Information-Centric Networking (ICN) supports content replication, thus offering resilience to failures, origin server offloading and improved in-network forwarding [1]. Although this is efficient for "bulk" data, that is, data that only make sense when fully available, performance and security concerns are raised for data over which computations are made; this includes relational databases, key-value encoded data streams (e.g., IoT data), the append only logs of distributed ledgers [2] and distributed social networks [3], as well as the conflict-free replicated data types used in distributed messaging [4]. In these systems, it is desirable to implement fine grain access control, so that users can access only the portions of the data that they need or that they paid for. However, isolating data items from their context may jeopardize the correctness of the calculations that rely on them, if their users cannot be persuaded that they are authentic and untampered with.

In this paper, we present the initial design and implementation of a solution that achieves the following:

- It enables storage nodes, starting from the same data item encoding, to selectively disclose different portions to different consumers, while providing the same guarantees about the items' integrity and correctness.
- It specifies the algorithms implemented in ICN nodes and the protocol exchange using ICN "Interest" and "Data" packets allowing users to express the portion of a data item that they want, in a succinct and efficient way.
- It allows data owners to specify the access rights of a user in a way that can be verified by any entity, without having access to any secret or sensitive information.

To achieve these properties, we leverage a recent digital signature algorithm, known as BBS+, that allows selective hiding of elements of some *structured* data item, providing at the same time Zero-Knowledge Proofs about the correctness of the revealed portion. To the best of our knowledge, this is the first work that applies BBS+ and Zero Knowledge Proofs in an ICN context.

To motivate our solution, we consider an IoT system that outputs measurements in the form of key-value pairs. These pairs are generated by multiple IoT devices and are collected by a gateway that stores them in a file together with some metadata, including digital signatures ensuring their provenance. This file can be then replicated to multiple storage nodes. Our solution allows the data owner to define policies that restrict the measurements a consumer can access. Then, it allows a storage node to reveal to consumers only the information for which they are authorized, while also allowing consumers to verify the integrity and the correctness of the received information, e.g., it guarantees that the storage node cannot modify, switch, or delete measurements. Existing state of the art solutions employ either access control based on some sort of "access token" (e.g., [5], [1]), or access control enforced through the encryption of the protected content (e.g., [6], [7]). In the former, a storage node can perform selective disclosure but it must be trusted not to modify the transmitted content, whereas, in the latter, the integrity of the content is transmitted but the storage node cannot disclose partial content items. Our solution combines the best of both worlds.

The remainder of this paper is organized as follows. In Section II we introduce the building blocks of our system. We detail the design of our system in Section III and we present its implementations and evaluation n Section IV. Finally, we conclude our paper in Section V.

II. Background

A. Named-Data Networking

Named-Data Networking (NDN) [8] is the most popular realization of *Information-Centric Networking* (ICN) paradigm to rethinking the Internet's architecture. A basic tenet of NDN is that each piece of content has its own name, which is location-independent. This allows content to be made available from multiple origin servers, or even caching nodes; each such node is referred to as a *publisher*. Publishers in NDN *announce* the prefixes of the named content items that they *host* to the NDN network, thus allowing content *consumers* to retrieve the corresponding content.

A consumer in NDN expresses her interest for a content item by sending an *Interest* packet. Interests include the name of the desired item and they are routed by *content routers* (CRs) towards content publishers using routing state created by the content announcements. Each CR that forwards an Interest, populates accordingly a *Pending Interest Table* (PIT) which is used for forwarding data items in the reverse direction (i.e., from the publisher towards the consumer).

Content items in NDN are digitally signed by their owners, using a signature scheme that protects the items' integrity and provenance.

B. ZKPs using BBS+ signatures

Zero-knowledge proofs (ZKPs) are a fundamental notion in cryptography, in which an entity (the prover) proves knowledge of a piece of information that satisfies a certain relationship (for example, knowledge of a discrete logarithm), to another entity (the verifier), without revealing any information about the piece of information itself. This functionality, has led to the creation and implementation of a large number of cryptographic protocols using ZKPs.

One of those protocols is *BBS+*. BBS+ is a multi-message digital signature protocol, that also encapsulates ZKPs for a critical part of its functionality. It was first envisioned by [9] (from where it takes its name), and is currently under standardization [10]. BBS+ can be thought as a composition of two (interdependent) components; the digital signature and the ZKP protocol. As a digital signature, BBS+ provides the ability to sign an array of individual messages (each message consisting of a string of octets), with a single *constant size* signature. The signature can be validated given the signer's *Public Key* (PK) and the entire array of signed messages; this is equivalent to validating a "traditional" digital signature, if we consider the array of messages as a single compound message.

As a ZKP protocol, BBS+ enables any entity that knows the signature and the original signed array of messages, to create a proof of knowledge of the signature while selectively disclosing only a sub-array of the signed messages. The proof size will be linear to the number of un-revealed messages. The proof can be validated with only the signer's PK and the array of revealed messages. The whole protocol is zero-knowledge in the sense that, from this interaction, no information can be derived about the signature or the un-disclosed messages.

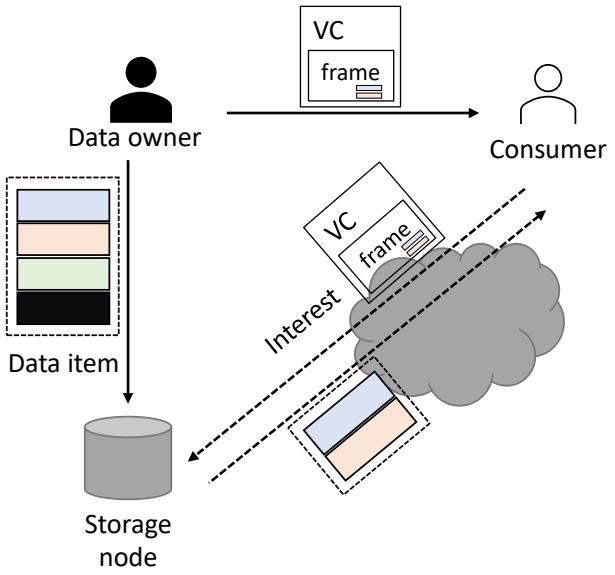

Fig. 1. Entities of our system and their interactions.

C. Verifiable Credentials

A *Verifiable Credential* (VC) [11] allows an *issuer* to assert some *attributes* about an entity referred to as the VC *subject*. A VC includes information about the issuer, the subject, the asserted attributes, as well as possible constraints (e.g., expiration date). Then, a VC *holder* (usually, the VC subject itself) can prove to a *verifier* that it owns a VC with certain characteristics. This is usually achieved by including in the VC an identifier (e.g., a public key), owned by the holder that enables the holder to generate a *proof of possession* (e.g., a digital signature with the corresponding private key). VC verification does not require communication with the issuer.

The VC data model allows different VC *types*, which define the attributes a VC should include. This provides great flexibility, since VC integrators can define their own types that fit the purposes of their systems. Our system uses a new VC type named *authorization* that "describes" which portion of a data item a user can access. This VC is issued by the data owner and is used by the data consumer to prove to a storage node that he or she has permission to see this portion of a stored data item.

III. Design

A. Entities and interactions

Our system considers the following entities interacting over an NDN-based architecture. A data *owner* that produces data items structured as key-value pairs and encoded as JSON objects, a storage node acting as the data *publisher*, and a data *consumer* wishing to access portion of the stored data. Owners *authorize* consumers by issuing to them a VC that gives them partial access to the stored data. This is achieved by including in the VC a data structure, referred to as a *frame*, that "describes" the "sub-item" the consumer can access (we detail this approach in section III-B1). Then, consumers interact with publishers using basic ICN functionality, i.e., publishers *advertise* data items as a whole and consumers send *interests*

for data items. Interests include application layer parameters that indicate the portion of the data the consumer wants to (and is allowed to) access.

Figure 1 gives an overview of our system entities and their interactions. In this figure, a data owner passes to a storage node an item that includes four "entries" and issues to a consumer a VC that gives it access to the two "top-most" entries. The consumer includes that VC in an Interest packet and receives back the corresponding sub-item.

Data owners maintain two public-private key pairs for this purpose: the first allows them to generate BBS+ signatures over the (complete) data items, and the second allows them to sign the VCs issued to consumers. Similarly, a consumer maintains a public-private key pair for this purpose: the public part is included in the VC issued by the data owner and the private part is used to generate a *proof of possession* of that VC. Consumers learn, using out-of-band mechanisms, the public key of the owner that can be used to verify BBS+ signatures. Similarly, storage nodes are configured with the public key of the owner that can be used to verify authorizations. In particular, storage nodes are configured with pairs of the form [$Owner_{Id}$, public key], where $Owner_{Id}$ is a data owner unique identifier, also included in the issued VCs. Storage nodes are not required to store any secret information related to the owners. Similarly, consumers are not required to maintain any information about storage nodes.

B. Algorithms and Protocols

1) Data framing: Framing refers to the derivation of a "sub-item", from a (signed in our system) item, that contains only part of the original one. Data framing is used to enable selective disclosure of the data item's information. More specifically, the framing algorithm accepts the original item and a *frame* as input, and returns a new item that only contains the key-value pairs specified by the frame. The frame itself is a JSON structure that specifies the parts of the original item that should be disclosed. For this purpose, the frame contains the keys (not the values) that the prover wants to reveal. For example, using the frame illustrated on the bottom left of Figure 2 with the data item shown on the top left of Figure 2, will result to the sub-item illustrated in the bottom middle of Figure 2.

Frames in our system are included in Interest messages sent by consumers, and the framing operation is executed by the corresponding storage node. The framing algorithm used in this work is inspired by the framing technique introduced and used for JSON Linked Data (JSON-LD) [12], but simplified and adapted to work on JSON-encoded items.

2) Data canonicalization: As discussed in Section II-B, BBS+ signatures act on arrays of messages, not on structured data formats like JSON. In order for an owner to be able to sign a data item and for a storage node to be able to derive ZKPs, data items must be *canonicalized*. Various canonicalization algorithms have been proposed [13], [14] by related efforts. A canonicalization algorithm serializes a JSON-encoded item into an array of messages, which can then be signed by a multi-message digital signature system like BBS+.

There are various security requirements that those algorithms must conform to, so as to avoid compromising the security of the system. In this work, we use the JCan algorithm [15] which is a lightweight, provably secure, JSON canonicalization proposal, designed to work with any data model.

A data item has to be canonicalized only once. This is done by the data owner who also generates a BBS+ signature over the canonicalized item. Then, the (canonicalized) item and the signature are stored at one or more storage nodes, alongside the original data item.

3) ZKP generation: A storage node can generate a sub-item of a content item based on a frame and provide a ZKP that proves its correctness as follows. Initially, the storage node applies the framing algorithm discussed in Section III-B1 to derive the sub-item. After framing, a storage node canonicalizes the resulting sub-item, gets the array of messages that correspond to the revealed information (from the security properties of the canonicalization algorithm, this array is guaranteed to be a subset of the signed array that resulted from the canonicalization of the original item) and uses that array to derive a ZKP using BBS+.

4) Consumer authorization: An owner can authorize a consumer to access a specific portion of a data item by *issuing* a *Verifiable Credential* (VC) that includes the frame a consumer is authorized to use. A VC can be encoded using different formats. In our system, VCs are encoded as JSON Web Tokens (JWT) and embedded by the data owner in JSON Web Signatures (JWS) [16]. A JWT encoded VC includes the following claims (specified by the VC data model):

- `iss`: A data owner specific identifier.
- `cnf`: The public key of the authorized consumer encoded as a JSON Web Key [17].
- `aud`: The identifier of the data item the VC concerns.
- `iat`: A timestamp indicating the VC's issuance time.
- `exp`: A timestamp indicating the VC's expiration time.
- `vc`: The actual frame.

5) Data access: A consumer can request a data item by including in the corresponding Interest message a VC it has received, as well as a *proof of possession*, which is a digital signature of the Interest message that can be verified using the public key included in the `cnf` claim of the VC. The proof or possession prevents an eavesdropper from receiving the data by replaying an overheard VC.

Upon receiving such an Interest message, a storage node verifies the consumer's access rights by following these steps:

1) It verifies that the provided VC has not expired, as well as that the `aud` claim of the VC is the same as the identifier of the requested item.
2) It validates the signature of the VC using one of the pre-configured public keys of the data owner.
3) It extracts the `cnf` claim included in the VC and verifies the provided proof of possession.

The storage node then extracts the frame included in the provided VC, it canonicalizes it, it applies it to the requested content item, and it generates the corresponding ZKP. Finally, it forwards the sub-item output by the framing operation, as

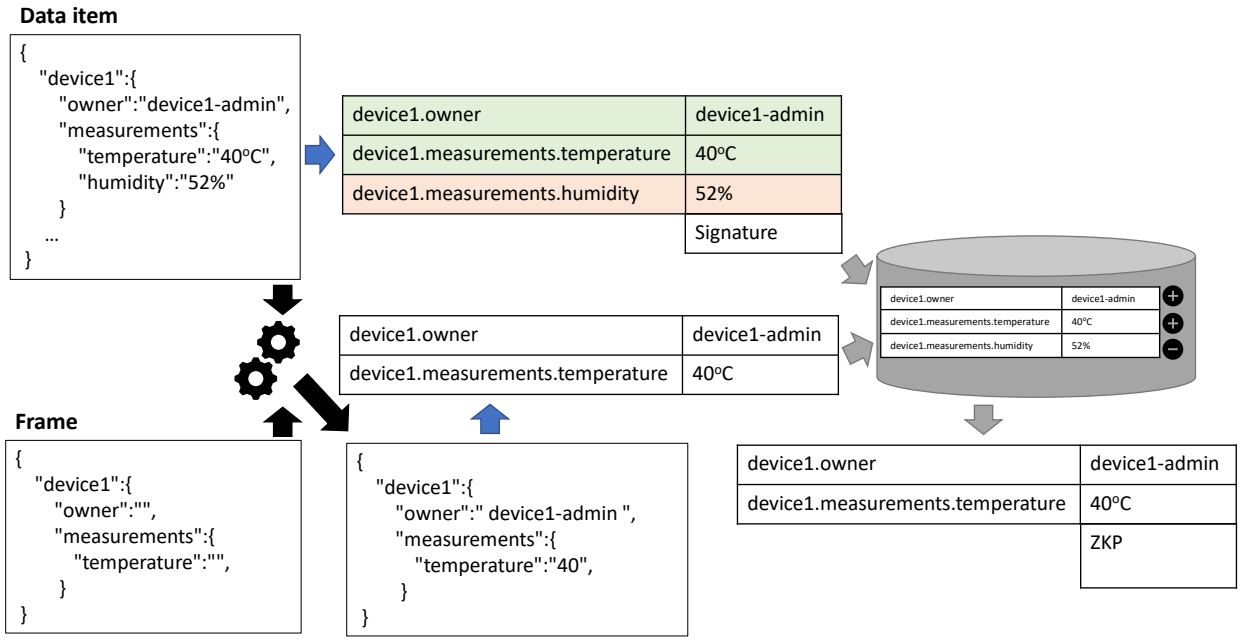

Fig. 2. Overview of the algorithms used by our system: framing (black arrows), canonicalization (blue arrows), and ZKP generation (grey arrows)

well as the BBS+ signature of the requested items and the generated ZKP, all in an ICN Data packet.

IV. Implementation and Evaluation

We implemented a proof of concept of our system[1] for the NDN architecture using python-ndn[2] and deployed it in the NDN testbed. BBS+ signatures are implemented using the Python wrapper of Hyperledger Ursa BBS Signatures library[3].

In our implementation we encode the parameters required by the *Data access* protocol in the *ApplicationParameters* option of NDN Interest packets.[4] This element is optional and it can carry any arbitrary data so as to parameterize an Interest for a Data packet according to the individual needs of the application. The NDN API does not restrict the usage of this element in any way, it simply defines the type of the parameter as *Binary String* (BinaryStr) which can serve any type of variable in its serialized form.

A. Overhead

We evaluate the overhead introduced to a storage node in an Ubuntu 18.04 machine equipped with an Intel i7-3770 CPU, 3.40GHz and 16GB of RAM. We consider a manually created data item composed of 100 key-value pairs. We calculate the time required to sign and verify sub-items that include from 1 to 99 records. Figure 3 shows the signature and verification time, measured in ms. We can see that as the number of items included in the sub-item increases, the signature creation time decreases, since for each hidden item a storage node has to perform a number of multiplications. On the other hand, the signature verification time remains almost constant. It should be noted that these measurements are obtained without any "pre-calculation"; in a real deployment, a storage node can pre-calculate many of the computations required for ZKP creation.

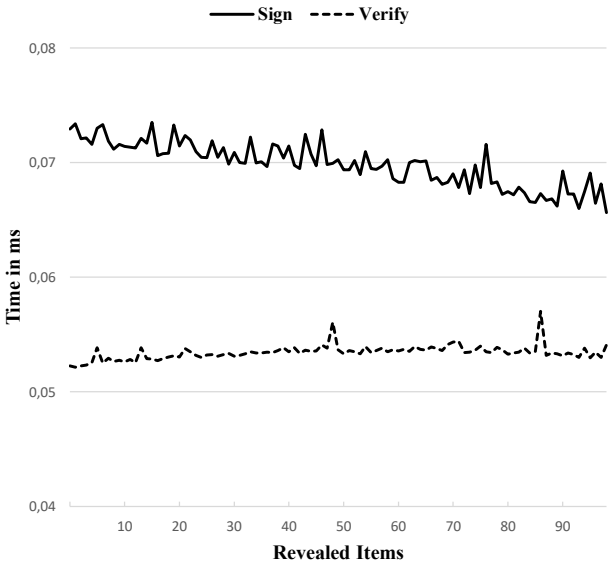

Fig. 3. Time to calculate a ZKP as a function of the revealed items

We now compare the communication overhead of our solution against a "naive" system in which each key-value pair of the data item is individually signed using an EdDSA digital signature [18]. We focus only on the overhead added by the digital signatures. Signatures in our system require $272 + 32 \times i$ bytes, where i is the number of hidden items. On the other hand, the "naive solution" requires $64 \times k$ bytes, where k is number of requested items. Figure 4 shows the communication overhead of these two approaches. We can see

[1]https://github.com/mmlab-aueb/zkp-toolkit/
[2]https://python-ndn.readthedocs.io/
[3]https://pypi.org/project/ursa-bbs-signatures/
[4]https://named-data.net/doc/NDN-packet-spec/current/interest.html

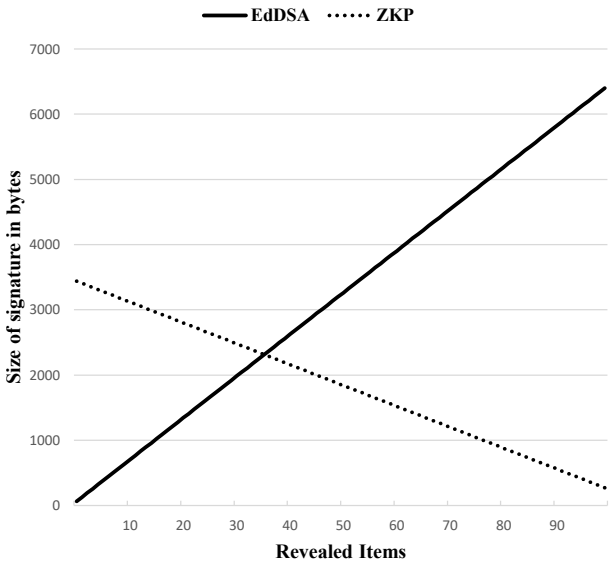

Fig. 4. Communication overhead introduced by the signatures as a function of the revealed messages

that if a consumer requests more than 36 items, the overhead of the signatures in our scheme is smaller.

V. Conclusions

In this paper we presented the initial design and evaluation of a solution that achieves selective disclosure of content by leveraging BBS+ signatures, which support Zero-Knowledge Proofs (ZKP). Our system not only protects the integrity and the correctness of the (partially) revealed content, but also enables storage nodes to perform access control with no access to secret information. Similarly, our solution requires content consumers to only know the public key of the content owner.

The work reported in this paper is an ongoing effort, currently undertaken by the NGIAtlantic.eu project SECOND.[5] Ongoing and future work in this area includes support for multiple frames per authorization, as well as, aggregation of requests for different portions of the same content item. Furthermore, our solution can be applied by any in-network node: this creates opportunities for "smart" cache population mechanisms. For example, even though a consumer may request a portion of an item, a storage node may decide that it is worthwhile to cache the whole item in the network, closer to the consumer; in that case it may send the whole item, alongside a frame and "instructions" to caches located close to the consumer to forward only the requested item portion.

Acknowledgments

The work reported in this paper has been funded in part by European Union's Horizon 2020 Research and Innovation Programme through the subgrant *Securing Content Delivery and Provenance (SECOND)* of the project *NGIatlantic.eu*, under grant agreement No 871582.

[5]https://mm.aueb.gr/projects/second

References

[1] C. Ghasemi, H. Yousefi, and B. Zhang, "Far cry: Will CDNs hear NDN's call?" in *Proc. of the 7th ACM Conference on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, 2020, pp. 89–98.

[2] Z. Zhang, V. Vasavada, R. King, and L. Zhang, "Proof of authentication for private distributed ledger," in *Proc. of the NDSS Workshop on Decentralised IoT Systems and Security (DISS)*. Reston, VA, USA: Internet Society, 2019.

[3] D. Tarr, E. Lavoie, A. Meyer, and C. Tschudin, "Secure scuttlebutt: An identity-centric protocol for subjective and decentralized applications," in *Proc. of the 6th ACM Conference on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, 2019, p. 111.

[4] C. Tschudin, "End-to-end encrypted scalable abstract data types over ICN," in *Proc. of the 5th ACM Conference on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, 2018, p. 8894.

[5] N. Fotiou, G. F. Marias, and G. C. Polyzos, "Access control enforcement delegation for information-centric networking architectures," in *Proc. of the 2nd ACM Workshop on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, 2012, pp. 85–90.

[6] A. Gawande, J. Clark, D. Coomes, and L. Wang, "Decentralized and secure multimedia sharing application over named data networking," in *Proc. of the 6th ACM Conference on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, 2019, pp. 19–29.

[7] Z. Zhang, Y. Yu, S. K. Ramani, A. Afanasyev, and L. Zhang, "NAC: Automating access control via named data," in *Proc. of the IEEE Military Communications Conference (MILCOM)*, 2018, pp. 626–633.

[8] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *SIGCOMM Computer Communications Review*, vol. 44, no. 3, pp. 66–73, Jul. 2014.

[9] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Proc. of the Annual International Cryptology Conference*. Heidelberg, DE: Springer, 2004, pp. 41–55.

[10] A. Whitehead, M. Lodder, T. Looker, and V. Kalos. (2022) The BBS signature scheme. Https://identity.foundation/bbs-signature/draft-bbs-signatures.html.

[11] Manu Sporny et al., "Verifiable credentials data model 1.0," 2019, https://www.w3.org/TR/verifiable-claims-data-model/.

[12] D. Longley, M. Sporny, G. Kellogg, M. Lanthaler, and N. Lindstrm. (2022) JSON-LD 1.1 framing. https://w3c.github.io/json-ld-framing/.

[13] R. Arnold and D. Longley. (2020) RDF dataset canonicalization. https://lists.w3.org/Archives/Public/public-credentials/2021Mar/att-0220/RDFDatasetCanonicalization-2020-10-09.pdf.

[14] M. Jeremie, W. David, and J. Michael, "JSON Web Proof," 2022, https://json-web-proofs.github.io/json-web-proofs/draft-jmiller-json-web-proof.html.

[15] V. Kalos and G. Polyzos, "Requirements and secure serialization for selective disclosure verifiable credentials," in *Proc. of the International Conference on ICT Systems Security and Privacy Protection (IFIP SEC)*, vol. 648. Heidelberg, DE: Springer, 2022.

[16] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Signature (JWS)," Internet Requests for Comments, IETF, RFC 7515, May 2015. [Online]. Available: https://tools.ietf.org/html/rfc7515

[17] M. Jones, "JSON Web Key (JWK)," Internet Requests for Comments, IETF, RFC 7517, May 2015. [Online]. Available: https://tools.ietf.org/html/rfc7517

[18] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of cryptographic engineering*, vol. 2, no. 2, pp. 77–89, 2012.