Data Vaults: Strategies for storing, verifying and sharing private data

Arnold Almeida
arnold@brightnetwork.io

Flinn Dolman
flinn@brightnetwork.io

Jaime van Oers
jaime@brightnetwork.io

Version 1.0
15th August 2018
https://www.brightnetwork.io/

Abstract. Bright aims to create a system that will address small businesses' lack of access to finance and scalability issues faced by lenders in the financial services industry at large. To achieve this, it is necessary to securely store, verify and share information. This paper describes the requirements of a system that enables these data operations in a way which meets our higher architectural principles. We introduce three strategies, Storage, Attestation and Snapshotting that adhere to these requirements. Through cryptography, we demonstrate how data's potential can be unlocked. Our system enables novel utilisation of private data and catalyses value in any marketplace built upon the Bright Network. In conclusion, we demonstrate how the Data Vault meets the conditions of Autonomous Data and how these strategies have been implemented in Bright's private alpha.

1 Introduction

Bright Network is a decentralised system built upon an open data sharing protocol. It allows small business data to be stored, verified and consumed for the purpose of accessing financial services. Its goal is to address the lack of access to finance these types of businesses face, in addition to scalability issues faced by lenders in the financial services sector.

In order to address the problems posed by taking on such an ambitious goal Bright proposes Autonomous Data.

Autonomous Data

Autonomous Data is a term that encompasses both data and rules that are ascribed to dictate how that data can be utilised.

Autonomous Data Attributes

- Data is only viewable by its owner.
- Data owners must explicitly consent to share data with third parties.
- No unauthorised parties may view others data.
- All data is encrypted at rest and while in transit.
- Data is self-describing and schema aware.
- Data is censorship resistant and integrity checked.
- Data is traversable.
- Meta-data cannot be used to undermine privacy.

Autonomous Data Rules

- There must be a scheme for the data to be stored.
- There must be a scheme for the data to be shared
- There must be a scheme to verify that data is owned by any entity claiming to own it.

1.1 Bright Solution

It is our belief that implementation of a system whereby the requirements of Autonomous Data are met provides the best chance of realising Bright's ambition of improving MSME access to financial services. To meet these requirements Bright has developed what we term the **Data Vault**. We define this object as a collection of data coupled with the technical and cryptographic infrastructure required in order to meet the conditions put forward by Autonomous Data. This paper documents and justifies the design decisions we have taken to develop the Data Vault system and how our chosen design abides by Bright's architectural principles[1].

2 Problem statement

The following scenarios are the core primitives needed in order to meet the requirements of Autonomous Data. They can be combined to create more sophisticated network activities:

1. A Bright Network agent *Alice* wishes to store some private data (e.g. financial transactions, identity information).
2. A Network agent *Alice* wishes to share a subset of her stored data with n Network agents.
3. A Network agent *Carol* needs to verify that the data from *Alice* is legitimate and has not been tampered with.

These can be solved with cryptography, peer attestations and an appropriate data model.

2.1 Peer attestations through cryptography

A peer attestation or just 'attestation' is a claim of legitimacy provided by a Bright Network agent about data shared with them by another Network agent. For example, *Alice* can attest that an invoice claimed by *Bob* was accurate and has been paid, and *FictionalBank* can attest that *Carol* has positive cash flow. By assessing both a business' data and attestations on that data, a model of trust can be built. Notably, attestations can be made on individual or collected records of data, and each attestation can also be treated as data which can itself be attested.

2.2 Data Vault Requirements

Expanding on the interaction primitives above with attestation in mind yields the following requirements:

(i) **Bright Network agents should be able to add data to their Data Vault.**

(ii) **Network agents should be able to add data to other agent's Vaults.**

(iii) **Network agents should be able to attest data added to their Vault, to the benefit of n Network parties.**

(iv) **Network agents should be able to select a subset of their data and create a *Snapshot* in their Vault.**

(v) **Network agents should be able to share a Snapshot with n Network parties.**

[1] See Appendix B

3 Data Vault

Levering robust existing cryptographic primitives such as RSA[1], AES[2] and ECDSA[3] in conjunction with emergent decentralised storage offerings such as IPFS we can construct a modular architecture for agile research and development whilst also providing a solid foundation for innovation.

3.0.1 RSA

Asymmetric cryptography allows data to be encrypted by anyone using a public key in such a way that only the holder of the associated private key can decrypt it. Accordingly these systems allow users to store information in a form that only the owner of the private key can interpret. This is fundamental in a system like ours where users are storing potentially sensitive data about themselves. RSA is a widely adopted and heavily audited asymmetric cryptosystem.

3.0.2 AES

Symmetric cryptography allows data to be encrypted and decrypted using a single (equivalent) key. This is particularly useful in a system such as ours where users wish to securely share data with other parties. AES is amongst the best performing symmetric key algorithms both in terms of security and performance. Accordingly we have adopted it to aid in facilitation of data sharing between parties.

3.0.3 ECDSA

Elliptic curve cryptography allows a users to create practically unforgeable signatures. Data signatures allow users to confirm authorship, assess relevancy of attestations and prevent impersonation. Our choice is motivated by the fact that elliptic curve based methods are efficient whilst not compromising on security. ECDSA is widely adopted for use in Blockchains for these reasons. Bitcoin is a prime example.

3.0.4 IPFS and IPLD

IPFS[9] is a file storage protocol which embraces and enables our desired properties of Autonomous Data through decentralisation and immutability. In addition, Vault data can be structured to make use of optimisations that arise from content-addressable linked data formats, such as IPLD[4].

Specifically, IPLD provides a JSON inspired representation of data or "content" that shares much of its logic with Merkle trees[8]. This content is linked to other content through what are known as Merkle-links (content hashes in the context of IPFS). These links enable data objects to provide an immutable representation of their children, as the nature of content-addressable hashes means that alteration of children would require a new, unique parent to be created. These Merkle-links allow us to perform traditionally complex operations such as traversals between points of data with relative ease. For more details see the IPLD documentation[4]

Our use case has high affinity for this kind of structure as it allows us to hierarchically compose our data into related components without affecting data navigability. A tangible example of this is linking a data record representing a peer-to-peer attestation to its respective attested record. Constructing data in such a way creates exciting opportunities for the study and analysis of user trustworthiness in the network.

It is important to note that IPFS and IPLD is only used as a persistence layer which provides decentralisation, immutability and optimisations for data access. Technically a Data Vault can be reconstructed by its owner from the logs of our smart contract platform components as facilitated by the Bright Network protocol[2]. However, this is not expected to be a common event. The Data Vault system as a whole allows for flexibility in the event of any alternative emergent decentralised storage solution.

[2] See Appendix A.1

3.1 Strategy: Storage

In the previous sections we motivated the formulation of decentralised immutable data stores that we termed Data Vaults, and justified their implementation via IPFS. We now describe how to facilitate secure data sharing, attestation, and analysis of data stored within them as is required by Autonomous Data. As was established in Section 2, we require the realisation of three scenarios. To solve the first, which can be paraphrased as secure data transmission, we propose what we term the Storage strategy.

This strategy satisfies the requirements (i)-(ii) as described in section 2.2.

Strategy 1 Storage

Inputs

Action: $A :=$ STORE
Data: $\mathcal{D}$
Alice private-public RSA keypair: (s_{rsa}, p_{rsa})
Alice private-public ECDSA keypair: (s_{ec}, p_{ec})

Operators

RSA: $\Re$
AES: $\Im$
ECDSA: $\mathscr{Z}$

Goal

Alice will add $\mathcal{D}$ to her Data Vault

Strategy

(a) *Alice* randomly generates a key for AES: k_r and an unique id for the payload *id*.

(b) She uses k_r to encrypt the Data, $E_k := \Im(k_r, \mathcal{D})$.

(c) She encrypts k_r, $T_k := \Re(p_{rsa}, k_r)$ and constructs the tuple $C_s := [T_k, E_k]$ known as the cryptographic constituents.

(d) She constructs a tuple $b_1 := [C_s, id]$.

(e) She signs b_1, $sgn := \mathscr{Z}(s_{ec}, b_1)$ and constructs $b_2 := [sgn, b_1]$.

(f) She constructs a tuple $D_p := [b_2, A]$ known as the data package.

(g) She delegates management of D_p to the Bright Network protocol. The protocol handles the process of adding D_p to her Data Vault.

In the case of inter-agent transmission to a Bright Network agent *Bob*, *Bob*'s public key $p_{B_{rsa}}$ is used instead of *Alice*'s p_{rsa} and A is TRANSMIT instead of STORE.

3.1.1 The stages of submission to Alice's Data Vault:

1. *Alice* randomly generates a unique payload *id* and a new AES key then symmetrically encrypts a data record with it. She then asymmetrically encrypts the AES key with her own RSA public key and packages it with the encrypted data payload to form the cryptographic constituents.

2. *Alice* cryptographically signs these constituents alongside the *id* and bundles the signature, cryptographic constituents and *id* together.

3. Along with an action which dictates what operation to perform, this bundle is handed over to the Bright Network protocol.

4. The protocol handles the process of adding the bundle to her Data Vault.

3.1.2 The stages of submission to Bob's Data Vault:

1. *Alice* randomly generates a unique payload *id* and a new AES key then symmetrically encrypts a data record with it. She then asymmetrically encrypts the AES key with *Bob*'s public key and packages it with the data payload and the other cryptographic constituents.

2. *Alice* cryptographically signs these constituents alongside the *id* and bundles the signature, cryptographic constituents and *id* together.

3. Along with an action which dictates what operation to perform, this bundle is handed over to the Bright Network protocol.

4. The protocol handles the process of adding the bundle to *Bob*'s Data Vault.

3.2 Strategy: Attestation

This strategy is an application of the Storage strategy, with a specific payload representing the peer attestation of a record (2.1) from a Bright Network agent.

This strategy satisfies the requirement (iii) as described in section 2.2.

3.2.1 The stages of attestation submission to both Alice and Bob's Data Vaults:

1. *Bob* deconstructs the data package and decrypts the encrypted AES key using his private key. He then uses the AES key to decrypt the data record.

2. Assuming he agrees to attest the record, he constructs a new data record representing an attestation and copies it. This new attestation record will reference the original data record via its *id* parameter. The first copied record will confirm *Alice*'s claim for *Bob*'s benefit, whereas the second copy will attest *Alice*'s claim for her benefit, thereby giving each user a symmetrically-valuable attested record.

3. The two attestation records are then encrypted with the same procedures as in the Storage strategy, however the first record's random AES key is encrypted with *Bob*'s public key and the second record's random AES key is encrypted using *Alice*'s public key.

4. The packages are bundled with actions and handed over to the Bright Network protocol.

5. The protocol handles the process of submitting the two encrypted packages to *Alice*'s and *Bob*'s respective Data Vaults accordingly.

Figure 1 shows how the Bright Network protocol takes submitted data and distributes it between Bright Network agents.

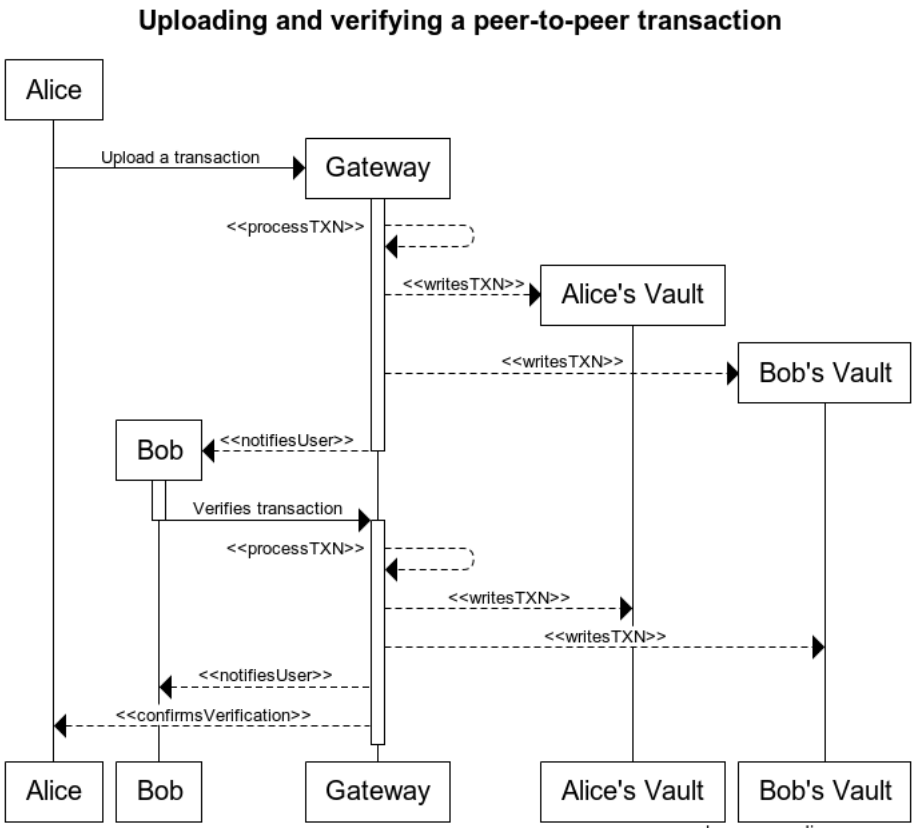

Figure 1: Sequence diagram describing the process of uploading and verifying some transaction data with a Data Vault.

3.2.2 Attestations using linked data

Under the linked data model, attestations of a record would be required to reference the ID of their 'parent' records. This referencing logic can be extended so that there is no reason that a Network agent couldn't have their attestation verified by a hierarchical chain of successive parties, i.e. attestations of attestations. Whether or not this is a desirable activity is for Bright Network agents to determine. Regardless, analysis of such data structures could certainly be used to provide rich insight into the trustworthiness of data and create new opportunities for Network agents such as credit scorers.

3.3 Strategy: Snapshotting

Section 2 also describes scenarios where a Bright Network agent wishes to provide simultaneous access to many pieces of Vault data. Under the Storage strategy in Section 3.1, doing so provides additional layers of complexity since decryption will take place using many different keys. We propose a new strategy, known as Snapshotting, which circumvents this issue. It provides the means to take Snapshots of multiple data entries in a way that allows them to be shared, referenced, analysed and attested as independent groups.

This strategy satisfies the requirements (iv)-(v) as described in section 2.2.

Strategy 3 Snapshotting

Inputs

Action: $A :=$ STORE.
n previously stored data packages: $[\mathcal{D}_i : \ i \in [1, ..., n]]$.
Alice private-public RSA keypair: (s_{rsa}, p_{rsa}).
Public RSA keys of n Network agents: $keys := [p_i : \ i \in [1, ..., n]]$.
Alice private-public ECDSA keypair: (s_{ec}, p_{ec}).

Operators

RSA: $\Re$
AES: $\Im$
ECDSA: $\mathscr{Z}$

Goal

Alice will add a Snapshot: $\mathcal{S}_p$ to her Data Vault using Snapshotting.

Strategy

(a) *Alice* randomly generates a key for AES: k_r and an unique id for the payload: id.

(b) She decrypts the payloads in each $\mathcal{D}_i$ and constructs a tuple $S := [d_i, ..., d_n]$ where d_i is the decrypted payload corresponding to the ith data package.

(c) She uses k_r to encrypt each element in S, $e_i := \Im(k_r, d_i)$ and constructs a corresponding tuple, $E_k := [e_i, ..., e_n]$.

(d) She encrypts a copy of k_r for each key in $keys$: $T_{k_i} := \Re(p_i, k_r)$ and creates a mapping of each T_{k_i} to its corresponding p_i, $\mathcal{K} := [p_i \rightarrow \ T_{k_i} | \ i \in [1, ..., n]]$. The set of maps $\mathcal{K}$ is known as the key lookup table (KLT).

(e) She constructs a tuple $C_s := [\mathcal{K}, E_k, id]$ known as the cryptographic constituents.

(f) She constructs a tuple $b_1 := [C_s, id]$.

(g) She signs b_1, $sgn := \mathscr{Z}(s_{ec}, b_1)$ and constructs $b_2 := [sgn, b_1]$

(h) She constructs a tuple $S_p := [b_2, A]$ known as the Snapshot.

(i) She delegates management of S_p to the Bright Network protocol. The protocol handles the process of adding S_p to her Data Vault.

Snapshots capture the current state of a portion of a Bright Network agents Data Vault at the time in which they are generated. For example, *Alice* could create a Snapshot of her Vault and provide a credit risk analysis service that is on the Bright Network access to the Snapshot via the KLT. This service could then decrypt the shared data and issue a credit score tagged to that Snapshot as a result of analysing the data. Sharing that same Snapshot with other Network agents such as lenders in future would then also reveal the associated credit score. This enables, as an example, lending decisions to be made.

To summarise, Snapshotting allows *Alice* to selectively share a portion of the data stored in her Vault with an arbitrary number of other Network agents.

3.4 Maintaining our principles

A Data Vault combines the strategies described in this paper (Storage, Attestation, and Snapshotting) to achieve our architectural principles[3] under the requirements of Autonomous Data.

[3] See Appendix B

3.4.1 Decentralisation

A Bright Network agent's data is stored in a decentralised manner. Data Vaults can be recreated by their owner without compromising on security.

3.4.2 Privacy

All data is secured according to ubiquitous, trusted cryptographic methodologies. Data can only be seen when explicitly shared with another Network agent and it can be selectively disclosed. Snapshotting in particular simplifies the sharing process by allowing a Network agent to securely share a subset of their Data Vault with n Bright Network counterparties.

3.4.3 Modularity and Extensibility

The component-wise design of a Data Vault allows for strategies to be interchanged and improved if doing so is desirable. Additionally the modularity of these strategies allows a Data Vault to facilitate other types of data utilisation yet to be considered by Bright.

3.4.4 Simplicity

We have aimed for a simplistic yet secure and effective design. This motivated our choice for using more well known and documented cryptographic procedures.

We are aware of how developing strategies that utilise concepts such as multi party computation[5][6] and zero knowledge proofs[7] may increase the privacy element of our network further. We will continue to critically evaluate where and how the aforementioned technologies can compliment the more traditional cryptographic techniques employed in our current strategies.

4 Conclusion

This document has defined and described a system adhering to what we term as Autonomous Data as well as Bright's architectural principles. The Data Vault system as it came to be known relies on two strategies constructed to realise the rules that Autonomous Data ascribes. The first, Storage, is designed to facilitate the transmission of a Bright Network agents data into either their own or another agent's Data Vault. Other agents can then attest to this data cryptographically, indicating their belief of its legitimacy. The second, Snapshotting, is a strategy in which an agent can share access to a subset of their Vault data to n agents.

In addition to these strategies Data Vaults define a schema required in order for the conditions set forth by the attributes of Autonomous Data. By meeting all prequisites of Autonomous Data, Data Vaults unlock the potential of business data improving small businesses' access to financial services in addition to addressing scalability issues faced by lenders in the financial services industry as a whole. We have implemented the system put forward in this document as part of Bright's private alpha.

Appendix A Definitions

A.1 Bright Network protocol

The Bright Network protocol is a protocol which can be used to perform actions between Bright Network agents and Data Vaults trustlessly. For example, an agent may use the protocol to submit data they own to their Data Vault. Due to the strategies and data model used, the protocol never needs to decrypt data or access private user information to function. In this document we shall only refer to the Bright Network protocol rather than explaining its inner workings. A full description is reserved for a future technical document.

Appendix B Architectural Principles

At Bright we subscribe to a number of architectural principles that we believe are fundamental to the growth of the Bright Network and the wider Blockchain ecosystem. These principles will guide our technologies and decision making in order to develop a sustainable, trustworthy, responsible, and inclusive ecosystem.

They are as follows:

Decentralisation	Provide services which have no centralised authority wherever possible. Where centralisation might be required during development, plan for decentralisation.
Privacy	Embrace privacy as the foundation for all operations. Never require disclosing private data, keys or identity. Mitigate privacy attacks wherever possible.
Open Protocols	Develop and publish open protocols wherever possible. Where closed protocols might be required during development, plan to open source.
Flexibility	Prefer decoupled components and protocols which scale to a myriad of use-cases.
Simplicity	User-facing software should always target a minimally-educated user. Architectural components should strive for simplicity.
Ecosystem Development	Contribute resources and code to improving the common Blockchain ecosystem wherever possible.

References

[1] Rivest, R.L., Shamir, A. and Adleman, L., 1978. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), pp.120-126.

[2] Daemen, J. and Rijmen, V., 1991. The design of Rijndael:AES—the Advanced. Journal of Cryptology, 4(1), pp.3-72.

[3] Junru, H., 2011, August. The improved elliptic curve digital signature algorithm. In Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on (Vol. 1, pp. 257-259). IEEE.

[4] David Dias: IPLD Specification
`https://github.com/ipld/specs/blob/master/IPLD.md`

[5] Yao, A.C., 1982, November. Protocols for secure computations. In Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on (pp. 160-164). IEEE.

[6] Andrew Chi-Chih Yao. 1986. How to generate and exchange secrets. In Proceedings of the 27th Annual Symposium on Foundations of Computer Science (SFCS '86). IEEE Computer Society, Washington, DC, USA, 162-167.

[7] Quisquater, J.J., Quisquater, M., Quisquater, M., Quisquater, M., Guillou, L., Guillou, M.A., Guillou, G., Guillou, A., Guillou, G. and Guillou, S., 1989, August. How to explain zero-knowledge protocols to your children. In Conference on the Theory and Application of Cryptology (pp. 628-631). Springer, New York, NY.

[8] R.C. Merkle, "Protocols for public key cryptosystems," In Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society, pages 122-133, April 1980.

[9] Benet, J., 2014. IPFS-content addressed, versioned, P2P file system. arXiv preprint arXiv:1407.3561.