SBC-IPFS: A Practical Secure Encryption Broadcast Channel on IPFS *(draft version)*

Gonçalo Pestana
gpestana@hashmatter.com

***Abstract*—We design, implement and study a decentralized secure broadcast encryption channel (SBC) on the InterPlanetary File System (IPFS) protocol stack. The SBC leverages a cryptographic broadcast encryption scheme that allows a set of participants (the sources) to encrypt messages with capabilities that only a subset of receivers can decrypt. The encrypted message is then published on a public channel where all the receivers attempt to recover the plaintext of the encrypted message. The decryption is successful only for the subset of receivers selected by the source. Furthermore, the SBC guarantees that every reader has the same view of the messages published in the broadcast channel and that the published messages are not tampered.**

This technical report shows that the IPFS protocol stack can be used to implement a practical and decentralized SBC. We demonstrate that the properties provided by the IPFS protocol stack such as data integrity, availability, censorship resistance, decentralization and available pub-sub semantics can be leveraged to implement a practical and secure broadcast encryption channel that can be used by a wide array of applications. Moreover, we implement SBC-IPFS, a decentralized and secure broadcast encryption channel and study its bandwidth overhead and performance in the context of real-world applications.

I. Introduction

A broadcast encryption channel is a widely studied and used primitive that enables multiple sources to broadcast data simultaneously to many receivers. The source attaches capabilities to the data published in the channel, so that only a subset of the receivers can utilize the data (selective disclosure). In addition, the source should be able to efficiently change the subset of qualified receivers and revoke access to previous receivers in each data broadcast. Broadcast encryption channel uses broadcast encryption schemes [1] to implement the selective disclosure. A broadcast encryption scheme allows a source to construct broadcast messages to a large set R, with $|R| = t$, such that only a subset $S \subseteq R$, with $|S| = r$, is capable of reading it. The subset of selected receivers might evolve dynamically such that the sender might choose the set of S at any time.

A secure broadcast encryption channel (SBC) is an instantiation of the broadcast channel primitive with the additional guarantees that all the receivers have access to untampered and authenticated data in the broadcast channel.

A real-world example of a broadcast encryption channel with selective disclosure is the emission of digital TV signals through the atmosphere. The encoded and encrypted signal is emitted by the TV broadcaster (the source). The signal is received, decoded and decrypted by the satellite dish and TV set by the receivers. Only viewers that have a TV set with the correct decryption key installed can watch the TV shows broadcast. Note, however, that this broadcast encryption channel is not secure.

More recently, many cryptographic schemes have relied on theoretical secure broadcast channels between participants. However, there are no practical SBC implementations that can be used by system and protocol developers. In this work, we fill this gap by designing, implementing and evaluating a practical secure broadcast encryption channel that leverages the IPFS [7] protocol stack and can be used in a variety of applications.

Broadcast Encryption

In broadcast encryption schemes the selective disclosure mechanism plays an important role in the performance and flexibility of the encryption channel. In recent works [5] [3] [2], the selective disclosure mechanisms are implemented through cryptography. In the trivial case, the source encrypts the data to be published in the public channel and distributes the decryption keys to the subset of receivers that are allowed to utilize the data. However, this approach is not practical as it requires the source to generate and distribute a new secret key every time the subset of receivers changes. Less trivially, the source can leverage public-key cryptography and key encapsulation schemes (KEM) [4] to generate and distribute the symmetric cryptographic key material to the receivers. Recent research work [2] [6] has improved the efficiency and overhead of KEM-based broadcast encryption schemes. These schemes leverage pairing cryptography to achieve low bandwidth overhead, optimize the computation time to decrypt the message by the receiver and provide security against any number of colluders. In addition, the pairing-based broadcast encryption schemes offer the flexibility to define the trade-off between the length of the receiver keys and the overhead of the ciphertexts.

In addition to pairing-based broadcast encryption, there are other schemes that implement the selective disclosure in broadcast channels. Attribute-based encryption [3] and functional encryption [5] are some examples. In this work we focus on studying the practical considerations of implementing a secure broadcast encryption channel using the IPFS protocol stack.

IPFS as a Secure Broadcast Channel

IPFS [7] is a content-addressing distributed file system that is implemented by a stack of modular protocols with well defined responsibilities and interfaces. Most notably, the network layer defines the data transport in the network; the routing layer is responsible for network peer and content discovery; the merkledag layer defines how to resolve content paths in the network and the naming layer defines how to immutably name resources stored in the network. On top of these layers, developers can build rich decentralized applications. The IPFS protocol stack allows developers to represent, replicate and route arbitrary data structures in a distributed network, unlocking the potential for developers to create secure, scalable, offline-first and distributed applications. A notable example of an application developed on IPFS is PulsarCast [8], a distributed pub-sub system which is built on the IPFS protocol stack to achieve high scalability, persistence and eventual delivery guarantees on a distributed network.

In this work, we implement SBC-IPFS and show how the IPFS protocol stack can be leveraged to implement a practical, decentralized and secure broadcast encryption channel. A practical SBC system requires a reliable and scalable transport bus, message authentication, message integrity and data availability. Moreover, pub-sub semantics help abstracting the message transport between participants of the broadcast channel and provide added flexibility. In addition to above-mentioned properties, the IPFS protocol stack provides availability and, potentially, censorship resistance in a distributed setting. These properties make IPFS a suitable fit for the infrastructure upon where to build a scalable, decentralized and secure broadcast encryption channel.

Contributions

The contributions of this work are:

- We describe and implement SBC-IPFS, a secure broadcast encryption channel built on top of the IPFS protocol stack. Our implementation leverages the properties provided by IPFS and pub-sub semantics to construct a practical, decentralized and secure broadcast encryption channel;
- We implement *beam-rs* [9] in Rust, a broadcast encryption cryptographic scheme based on [2] and [6] that provides a flexible broadcast encryption and selective disclosure mechanisms for SBC-IPFS;
- We implement *libp2p-sbc* [10], a libp2p client that, together with *beam-rs*, implements the source and receiver peer logic to participate in SBC-IPFS;
- We measure and evaluate the SBC-IPFS client implementation with regards to its network-level overhead and latency when running a high throughput application (100 messages/second) with multiple sources and receivers on IPFS;
- We demo and show how SBC-IPFS can be used for a wide variety of applications by the web3 ecosystem.

The remainder of this technical report is organized as follows. Next we describe the design, infrastructure and implementation details of SBC-IPFS and how it glues together the IPFS protocol stack and the pairing-based broadcast encryption scheme. In the last section, we outline the performance and overhead evaluation of SBC-IPFS when applied to two real-world use cases: a privacy-preserving and verifiable analytics system and a decentralized chat application.

II. SBC-IPFS: A SECURE BROADCAST CHANNEL ON IPFS

In this section we describe SBC-IPFS, an implementation of a practical and secure broadcast channel on IPFS[1]

III. EVALUATION

In this section we evaluate the SBC-IPFS implementation (Section II) applied to two use cases: a privacy-preserving and verifiable analytics system and a decentralized chat application[1].

REFERENCES

[1] Amos Fiat and Moni Naor. "Broadcast Encryption". In: *Advances in Cryptology — CRYPTO' 93*. Ed. by Douglas R. Stinson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 480–491. ISBN: 978-3-540-48329-8.

[2] Dan Boneh, Craig Gentry, and Brent Waters. "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys". In: *Advances in Cryptology – CRYPTO 2005*. Ed. by Victor Shoup. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 258–275. ISBN: 978-3-540-31870-5.

[3] Vipul Goyal et al. "Attribute-based encryption for fine-grained access control of encrypted data". In: vol. 89-98. Jan. 2006, pp. 89–98. DOI: 10.1145/1180405.1180418.

[4] Tatsuaki Okamoto. "Authenticated Key Exchange and Key Encapsulation in the Standard Model". In: *Advances in Cryptology – ASIACRYPT 2007*. Ed. by Kaoru Kurosawa. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 474–484. ISBN: 978-3-540-76900-2.

[5] Dan Boneh, Amit Sahai, and Brent Waters. "Functional Encryption: Definitions and Challenges". In: *Theory of Cryptography*. Ed. by Yuval Ishai. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 253–273. ISBN: 978-3-642-19571-6.

[6] Renaud Dubois, Aurore Guillevic, and Marine Sengelin Le Breton. "Improved Broadcast Encryption Scheme with Constant-Size Ciphertext". In: *Pairing-Based Cryptography – Pairing 2012*. Ed. by Michel Abdalla and Tanja Lange. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 196–202. ISBN: 978-3-642-36334-4.

[7] Juan Benet. *IPFS - Content Addressed, Versioned, P2P File System*. 2014. arXiv: 1407.3561 [cs.NI].

[1] Work in progress

[8] J. Antunes. *Pulsarcast - Scalable and reliable pub-sub over P2P network*. https://github.com/JGAntunes/pulsarcast. 2020.

[9] G. Pestana. *beam-rs - Pure rust implementation of a collusion resistant broadcast encryption scheme with pairings*. https://github.com/gpestana/beam-rs. 2021.

[10] G. Pestana. *libp2p-sbc - A Rust SBC client*. https://github.com/gpestana/libp2p-sbc. 2021.