ℙAD

Privacy-preserving Accountable Decryption

White Paper
Version 0.5

October 2021

1 Introducing a new model of secure information flow

Transacting online today allows us to operate in one of two modes: complete openness or complete privacy. Data is either shared immediately, regardless of whether there is an immediate need, or is hidden forever. Because of this, at times we share more than we should, and at other times we share less. Moreover, when data *is* shared, there is no trustworthy source of transparency about when that data is actually consumed.

We all generate data that may need to eventually be shared. ℙAD can help to improve privacy and oversight of that information, because ℙAD provides:

- A secure and highly available data store that allows a designated recipient to access information regardless of circumstances at any time in the future, without trusted third parties, master keys, or reliance on non-cryptographic operations-based security.
- A guarantee that data remains private and encrypted until the point that the information is requested by a designated recipient.
- Cryptographically-enforced transparency about who decrypts and accesses data and when, so that anyone who inappropriately uses data can be held accountable and be exposed to judicial or regulatory challenge.

ℙAD is a new approach to maintaining privacy in the management of sensitive information. It allows anyone to share the *ability* to access a piece of information, whilst creating transparency and proofs as to whether or not that information has actually been accessed.

We look forward to working with developers in the application of ℙAD to protection of user privacy, reduction in the unnecessary spread of personal information, and creation of new products and services based on improved management of confidential data.

2 Problem Setting

We consider one of Alice's electronic devices or online bank accounts secured by a secret password. She would like to ensure that Bob has access to her device or account under pre-agreed (and usually exceptional) circumstances when Bob cannot get in touch with Alice or vice versa. Alice does not wish that her password be available to Bob unless the agreed circumstances require it; she also insists that Bob can only access her password if such access is mandatorily disclosed to her as a matter of a perpetual, unalterable and trustless recording process.

In another situation, Alice may also benefit from signing an insurance policy for her high value car, which may require that she install a tracking device that is remotely linked to Bob at a monitoring station. Alice is concerned that the geolocation information about her journeys held by Bob may fall into the wrong hands. Giving Bob access to the car's most recent geolocation and driving data when Bob legitimately suspects that the car has been stolen while Alice is

unreachable on vacation, with such access by Bob to Alice's data necessarily creating a public record, solves the problem of keeping her journey data secure until exceptional situations require Bob to access the data. It also prevents Bob from accessing the data in an unauthorised manner since Alice will be informed of Bob's access to her private geolocation data so that she could sue him for breach of terms.

These scenarios, which involve necessarily recorded access by designated parties to private content, are examples of accountable decryption (see also Ryan [2017] and Ryan [2020]). In each case, unauthorised access is effectively discouraged, while it becomes possible to pursue due legal or regulatory remedies whenever such unauthorised access does happen.

Alternative security arrangements where law enforcement agencies are forced to exploit undocumented security flaws which function as covert backdoors, or where data administrators deploy master keys that weaken encryption for all system participants, are simply not acceptable.

Instead, any solution to the problem of balancing individual privacy with public benefit must adhere to the following principles:

- Complete denial of access to personal information for authorised persons or covert access to such content via methods that depend on obscure security flaws, 'master keys', or delegated authority to system administrators, are not compatible with the interest of individuals or the general public.
- Content shall be principally private. Designated authorised persons may gain access to the private content only in such a way that their exercise of such access rights and their identity are necessarily matters of record to Alice or to the public at large, either at once or with a specified time delay.
- Any proposed solution should not only focus on prevention of unauthorised access but should also enable system participants to react to instances of unauthorised access.
- No private person shall be compelled to trust any single institution, person, or company regarding the faithful reporting of any data access request.
- Rather than attempt to provide software-encoded conditional access to private content, the solution should not require specifying the conditions of permitted access upfront. In most real world applications, it is practically impossible and unrealistic to try and list such conditions formulaically. Instead, the solution should allow users to seek legal or regulatory remedies in the event of unauthorised access.

Our solution is 'backward-compatible' with existing legal institutions and processes in the sense that ℙAD mirrors in the digital world certain widely used practices from the non-digital world, where for example warrants must be secured and presented to private persons before their homes can be searched. If an authorised entity exercises the access right outside the scope of their public duties, then they may be taken to court where the matter can be reviewed and dealt with in all its practical and situational complexities.

With such an accountable decryption available, much more robust or practically unbreakable encryption algorithms may become legally acceptable in countries where this is currently not so, offering more privacy where it matters and controlled public access where it is personally desired or socially warranted.

In Section 3, we set out the proposed cryptographic ℙAD protocol. It does not require fundamental innovation of cryptographic primitives and can be built out of standard and time tested cryptographic schemes.

3 ℙAD Protocol for Accountable Decryption

3.1 Protocol Protagonists and Cryptographic Components

We describe the accountable decryption protocol in relation to five protagonist types:

Alice wishes (or is required by law) to provide accountable access to Bob for one of her devices, online accounts, or data stores, which is protected by a secure password or is encrypted by Alice.

Bob is granted (or has by law the right to) accountable access to Alice's device or account.

The ℙAD system comprising one or more **accountability ledgers**, each of which is an append-only, tamper-proof, and time-stamped digital ledger. This ledger permanently holds all decryption requests from Bob, along with responses posted by other parties (see below).

Trustee$_i$ is one of many ($i = 1, 2, \ldots, n$) third party service providers who collectively constitute and deliver the services needed to implement ℙAD. Trustees respond to valid decryption requests and post messages to the accountability ledger that enable decryption.

Validator$_i$ is one of many ($i = 1, 2, \ldots, n'$) service providers who deliver necessary ledger maintenance and verification services. Validators help with prevention and recovery from certain unlikely attacks.

The detailed ledger architecture depends on our choice of blockchain and the secure ledger infrastructure. In this section, we describe the protocol in a way that is agnostic to these choices. In Section 4, we provide details about our implementation of the ledger.

The proposed protocol makes use of conventional public and private key cryptography, digital signatures, hash-based authentication techniques and secret sharing schemes. These will be used routinely to deliver secrecy and authentication, as well as ensuring that any successful attack on the protocol requires wide-scale malicious collaboration by many parties.

3.2 ℙAD in a nutshell

The ℙAD protocol has been designed with great care in order to maximise performance, reduce computational complexity, and minimise the attack surface. The main ideas can be described

succinctly with only modest simplifications.

Secret generation and encryption. Imagine that a user Alice holds a secret, or perhaps creates a new one every second. With her personal device, Alice encrypts her secret with a fresh symmetric key that only she knows. She can then store the encryption of her secret with ℙAD and/or have the ciphertext be sent to Bob. Alice uses a cryptographic secret sharing scheme to split the secret key into shares: one share for each of the trustees and validators. These shares contain no information about the secret key unless a large fraction of the trustees work together to combine their shares. Finally each share is encrypted and sent to its associated trustee or validator.

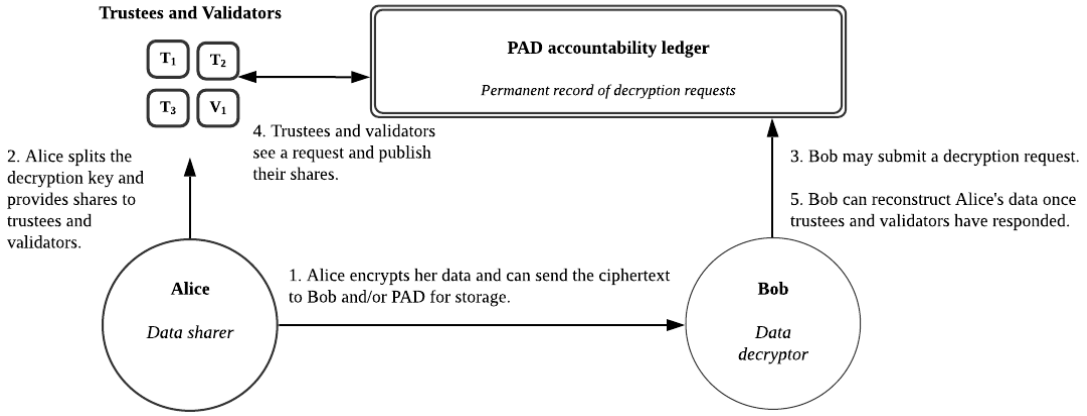
Submitting a decryption request and trustee responses. At any time, Bob can request access to Alice's secret by posting a message to the accountability ledger. The business of trustees and validators is to monitor the ledger for such requests; in response they decrypt their secret share and post it to the ledger.

Recovering the original data. Once a threshold number of trustees and validators have responded to the data decryption request, Bob can reconstruct the secret key. Bob can now access Alice's secret by decrypting the ciphertext he received from Alice at the start of the protocol.

Trust model. Accountability is provided by entries on ℙAD's public immutable, append-only ledger: the entry posted by Bob to request decryption and the entered responses from trustees that allowed Bob to decrypt. The way in which Alice has encrypted her secret requires that at least Bob and a threshold number of trustees and validators are required to uncover her secret. In unusual circumstances, perhaps one or two trustees might be willing to work with Bob and attempt to decrypt Alice's secret without checking that Bob's request is on the public immutable ledger. All such malicious efforts will fail and reveal nothing about Alice's secret unless a large coalition all operates maliciously at the same time.

Proofs of decryption and of non-decryption. If there is no malicious coalition containing at least a threshold number of trustees and validators, then ℙAD provides both positive and negative proofs about whether a decryption occurred. A decryption request and sufficient responses from trustees and validators guarantee that Bob has accessed Alice's secret. There is no feasible method for Bob to circumvent ℙAD and decrypt without leaving a record on the ledger. Therefore, the absence of these ledger entries is proof that Bob has not decrypted.

Simplified ℙAD schematic



3.3 The protocol in detail

We set out the full ℙAD protocol step by step. We presume that necessary cryptographic material has been securely created in advance. Further, we assume that the particular choices of cryptographic primitives are publicly known ahead of time, and that all parties have the capability to safeguard their own data and run secure computations locally.

Securing a secret

Step 1: Let s denote Alice's sensitive information. She chooses an arbitrary token value tkn, which is used to reference this particular secret. She then generates a fresh symmetric key K uses it to encrypt her secret and the token:

$$C = Enc_K(tkn, s)$$

Step 2: Alice uses her signing key sk_A to sign C and then uses Bob's public key pk_B to encrypt:

$$C' = Enc_{pk_B}(C, Sig_{sk_A}(C))$$

Step 3: Alice chooses a random masking value R and masks the symmetric key K by computing $\tilde{K} \leftarrow K \oplus R$. She then computes n secret shares $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_n)$ of the data for the trustees:

$$\boldsymbol{\sigma} := \textsc{Share}(\tilde{K}).$$

Step 4: Alice now generates the n' shares $\boldsymbol{\psi} = (\psi_1, ..., \psi_{n'})$ for the validators. She computes:

$$\boldsymbol{\psi} := \textsc{Share}(R, pk_B).$$

Alice enables each validator to verify independently that shares from the trustees (once they appear on the ledger) have not been tampered with. For each $j = 1, ..., n'$, Alice encrypts to validator j the vector of hashes of trustee shares:

$$Enc_{pk_{V_j}}(H(\sigma_1), ..., H(\sigma_n))$$

Step 5: For $i = 1, ..., n$, we use pk_{T_i} to denote the public key of the ith trustee. Similarly, for $j = 1, ..., n'$, let pk_{V_j} denote the public key of the jth validator. For each i and j, Alice encrypts the shares: $Enc_{pk_{T_i}}(\sigma_i)$ and $Enc_{pk_{V_j}}(\psi_j)$.

Step 6: Alice sends the following to the IPAD system. These data may be stored in a public blockchain or a secure database:

- The hash of the token value: $H(tkn)$
- All of the encrypted secret shares: $Enc_{pk_{T_i}}(\sigma_i)$ and $Enc_{pk_{V_j}}(\psi_j)$ for $i = 1, ..., n$ and $j = 1, ..., n'$.
- The encryption of the secret: C'.

Step 7: Alice provides Bob the token value tkn, which allows him to request decryption, and her verification key vk_A so that he can verify her signature.

Requesting a secret

Step 8: Bob wishes to gain access to Alice's secret s. Bob posts anonymously to the ledger the request message tkn. The anonymity of this request message ensures that trustees are not biased in their responses. Alice is aware that Bob is the source of the request as long as she maintains a mapping between her third-parties and the tokens she delegated in Step 7.

Step 9: The IPAD system finds the information provided by Alice in Step 6 by computing $H(tkn)$ and referencing its database. The system then publishes on the ledger all of the encrypted secret shares and C'.

Step 10: Trustees actively monitor the ledger, decrypt the secret share that is encrypted with their key, and publish the plaintext share on the ledger.

Step 11: Once a threshold number of trustees have completed Step 10, the validators do the same: they decrypt their shares and publish the plaintext shares on the ledger, along with the identity of Bob in the form of his public key. Validators check integrity of trustee shares and post to the ledger which, if any, published trustee shares are invalid.

Recovering a secret

Step 12: Bob can see C' on the ledger and use his private key to decrypt to C and $Sig_{sk_A}(C)$. Bob verifies the signature using the verification key he received in Step 7. Using the secret

sharing reconstruction procedure, Bob can recover $\tilde{K} = K \oplus R$ from the trustee shares and R from the validator shares. He then computes $K \leftarrow \tilde{K} \oplus R$. With K, Bob can decrypt C to obtain s, which is Alice's secret. No one else can learn s by observing the ledger because of the encryption of C' using Bob's public key. However, anyone viewing the ledger can conclude that the secret has been decrypted by Bob.

3.4 Risk analysis and security against collusion

There are in principle two main types of possible attack resulting from collusion among sufficiently large coalitions of parties: *data reconstruction* and *protocol censorship* attacks. Data reconstruction attacks require coalitions consisting of sufficiently many adversaries across multiple types of parties. Protocol censorship attacks require an overwhelmingly large fraction of one type of party.

3.4.1 Data reconstruction attacks

These attacks result in some secret data being recovered by the attacking coalition; it will thus be a primary concern for Alice. For the sake of her privacy, she wants to be quite certain that Bob cannot gain access to her secret password unless he requests such access in a manner that is public and hence knowable to her as a matter of record.

We suppose that the secret sharing thresholds for the trustees and validators are k and k', respectively. Any coalition consisting of at least k trustees, at least k' validators and Bob have sufficient information to decrypt s. By tuning the ratios k/n and k'/n', the likelihood of such a coalition forming can be made arbitrarily low.

3.4.2 Protocol censorship attacks

Just as Alice wants to enforce public disclosure of Bob's query for her secret, Bob wants to reduce the risk that his request might receive an insufficient number of responses from the n trustees or from the n' validators.

There are the risks (i) that some of the chosen trustees or validators fail, (ii) that the trustee job of monitoring the ledger for decryption requests becomes too expensive, and finally (iii) that a blocking number of trustees decide to boycott the protocol for any reason.

The mitigation of the first of these risks rests on the increasing redundancy that comes with lower k/n and k'/n' ratios. Alice demands a high probability that a threshold number of trustees respond only to requests published on the ledger, whereas Bob insists that his legitimate queries on the ledger reliably receive a sufficient number of responses. The secret sharing parameters k, k', n, and n' provide flexibility to tune IPAD according to a risk budget.

Regarding the second of these risks, the efforts by trustees and validators to hunt for query matches on the ledger - and by validators to check correctness of trustee responses - may be supported by commercial incentives.

The final type of risk involves a direct effort by trustees or validators to block the protocol from proceeding. In the case of trustees, if any k shares are published then the protocol can proceed; thus a trustee coalition of size at least $n - k + 1$ must collude to block the protocol from proceeding. The value for k should typically be smaller than half of n, so that the attack requirement of $n - k + 1$ dishonest trustees is as unrealistic as possible. The same logic applies to the validators and the parameters k', n'.

Validator protection against trustee bias. Censorship attacks may also occur on the basis of the requesting party Bob. The protocol protects against (even large) coalitions of trustees from bias against Bob: the secret shares of the trustees, taken jointly, do not reveal the identity of Bob. This is only revealed later by the validators, who are incentivised and more trusted to enforce the proper functioning of ℙAD. Trustees that wish to boycott the protocol may choose to not respond to valid data requests. A more meaningful attack would be to post invalid shares in response to a request. The determination of which subset of shares are valid would significantly slow the protocol. This attack is prevented because validators post to the ledger to identify any invalid trustee shares (see Step 4 in the general scheme).

4 The ℙAD system

ℙAD had been available as an API since September 2021. The API is designed to minimise the complexity that developers face when building an application using the ℙAD protocol. The API handles secret storage, provides trustees and maintains blockchain-based accountability ledgers. All underlying cryptography is abstracted away where possible, and provided in explicit code samples when necessary.

The ℙAD system and accountability ledgers. At present, the accountability ledgers are operated by the ℙAD team as a permissioned Hyperledger Fabric blockchain. Each ℙAD ledger is represented naturally as a channel within Hyperledger Fabric. In practice, it is preferable for different ledgers to exist for different purposes; anyone may request the creation of a new ℙAD instance and become its operator. An instance utilises its own ledger, and operators have the right to select trustees and choose the secret sharing thresholds. All decryption requests and trustee and validator responses are published on this ledger. Prior to a data request, encrypted secret shares and ciphertexts (see Step 6 in Section 3.3) are held off the blockchain in a ℙAD system database. We emphasise that the storage of this information confers no technical capability to view any secret data or allow its decryption without permanent publication of a decryption request.

Trustees. The ℙAD team runs several trustees that are ready to serve any ℙAD instance. Each trustee consists of a Raspberry Pi computer that frequently consults a ℙAD ledger and responds to requests as needed. We note that the storage requirement of trustees is independent of the number of secrets secured with a ℙAD instance - trustees only need to hold their own key

pair. We expect that application developers will wish to recruit or run their own trustees, so we provide all of the code necessary to become a trustee in our documentation.

Validators. The validator role is presently handled solely by the PAD team and we will decentralise this role as needed. We emphasise that control of validation does not allow the team to access any secrets - it simply enables us to help maintain the ledger and play a custodian role in the event of abuse of the system.

Encryptors and decryptors. Our documentation provides all the code needed to encrypt data with the system and secure it with PAD via the API. Developers can integrate this code into their application of PAD in any way that they see fit.

5 Alternative and future functionalities

Decentralisation of the PAD system. Crucial to PAD is the guarantee that no secret can be decrypted without accountability. To minimise the need for trust in the PAD system, the PAD protocol is built to support trustee attestation of their view of the ledger state. These attestations are published periodically and prevent the PAD system from maliciously performing a split-view attack in which certain requests are shown to certain trustees but not to others. These attestations and the cryptographic design of the PAD protocol provide a strong assurance that all decryption is accountable. Presently, any member of the public can become a trustee. To further reduce the need for trust, we intend to enable the public to run a *trustee peer*, which is a trustee that also operates as a consensus node in an accountability ledger. This will decentralise control of the blockchain. In addition, trustee peers and decentralisation of the blockchain will increase the likelihood that the service remains online and functional over time. Encryptors and decryptors will benefit from higher confidence that secrets will not be lost and will be available upon request at any time in the future.

Limited disclosure of decryptors. The protocol as described in Section 3.3 ensures that Bob's identity is revealed to the public by the posted shares of the validators. An alternative mechanism is that the validator shares do not reveal the identity of Bob. This is useful if Alice prefers that her selected data recipient remains anonymous. If the identity of Bob should be hidden by default, but Alice should be capable of proving later the identity of the decryptor, the protocol can easily be adjusted to allow this: Alice can require that Bob provide her a digital signature on *tkn* during the set-up phase (in Step 7).

Compensation mechanisms. Trustees may eventually wish to seek compensation for the service they provide. We expect that applications built using the PAD protocol will provide their own compensation mechanism, but the PAD team also aims to provide PAD-native compensation mechanisms both within and outside the crypto space.

Revocation. Depending on the use case, it may be desirable or legally required to give Alice the power to revoke Bob's ability to request decryption of her secret. Revocations are no different than decryptions in that they must be recorded on the ledger and publicly verifiable. This enables legal recourse in the event that a revocation is not respected. A revocation may simply consist of a digitally signed message on the ledger indicating that Alice has revoked a token *tkn*. This action alerts trustees and validators that they are not permitted to respond to data requests that contain the token value *tkn*. Bob is then prevented from using his token to request the secret. An equally simple protocol may be used in the event that Alice's identity is not to be revealed to trustees or the public. In particular, upon set-up, Alice may choose a random *revocation token* from a sufficiently large domain and include its hash on the ledger. Revocation will consist of her publishing the revocation token, which is validated by checking that it is a pre-image to the hash value originally present on the ledger.

6 Collaboration and applications

ℙAD is a new approach to maintaining privacy in the management of sensitive information. It allows anyone to share the *ability* to access a piece of information, whilst creating transparency as to whether that information has actually been accessed.

Rather than wade into the domain of jurisprudence, we have built the technology that allows accountable data sharing and provides indisputable evidence of what has taken place and not taken place. This makes ℙAD compatible with existing institutions of law and with contractual agreements.

Privacy and accountability are at the centre of the social contract. As a society, we need to collaborate on creating the tools, technologies and systems that solve the many flavours of privacy and accountability problems we face. The ℙAD team is recruiting a vibrant developer community to build privacy management applications and new products that are appropriate for deployment in the real world.

References

Asmuth, C. Bloom, J. (1983) *A modular approach to key safeguarding.* IEEE Transactions on Information Theory, IT-29, **2**, pp. 208-2011

Blakley, G.R. (1979) *Safeguarding cryptographic keys.* Proc. AFIPS 1979 National Computer Conference, pp. 313-317

Brickell, E.F. (1989) *Some ideal secret sharing schemes.* Journal of Combinatorial Mathematics and Combinatorial Computing, **6**, pp. 105-113

Diffie, W. & Hellman, M.E. (1976) *New Directions in Cryptography.* IEEE Transactions on Information Theory, IT-22, pp. 644-654

ElGamal, T. (1985) *A public key cryptosystem and a signature scheme based on discrete logarithms.* IEEE Trans. Inform. Theory, IT-31, pp. 469-472

Ito, M, Saito, A., Nishizeki, T. (1987) *Secret sharing scheme realising general access structure.* IEEE Proceedings Globecom 1987, pp. 99-107.

Karnin, E.D., Greene, J.W., Hellman, M.E. (1983) *On secret sharing systems.* IEEE Transactions on Information Theory, IT-29, pp. 35-41

Koblitz, N. (1998) *Algebraic Aspects of Cryptography* Berlin: Springer Verlag

Lindell, Y. & Katz, J. (2014) *Introduction to modern cryptography* Chapman and Hall/CRC

Rivest, R.L., Shamir, A., Adleman, L.M. (1978) *A method for obtaining digital signatures and public-key cryptosystems.* Comm. of the ACM, 21, pp. 120-126

Ryan, M. (2017) *Making Decryption Accountable.* Springer LNCS, 25th Security Protocols Workshop

Ryan, M. (2020) *Verifying information access to enhance transparency using accountable decryption.* Barcelona: FAT, January 27-30, 2020

Shamir, A. (1979) *How to share a secret.* Communication of the ACM, **22**, pp. 612-613.