Evaluating Social Schemes for Recovering Control of an Identifier

a white paper from Rebooting the Web of Trust VIII

by Sean Gilligan, Gregory Jones, Adin Schmahmann, Andrew Hughes, and Christopher Allen

ABSTRACT

As systems where people are required to manage their own cryptographic keys become more popular, social recovery or reissuance of keys increases in importance. Such systems are inherently empowering to users but safeguarding keys is a hard problem.

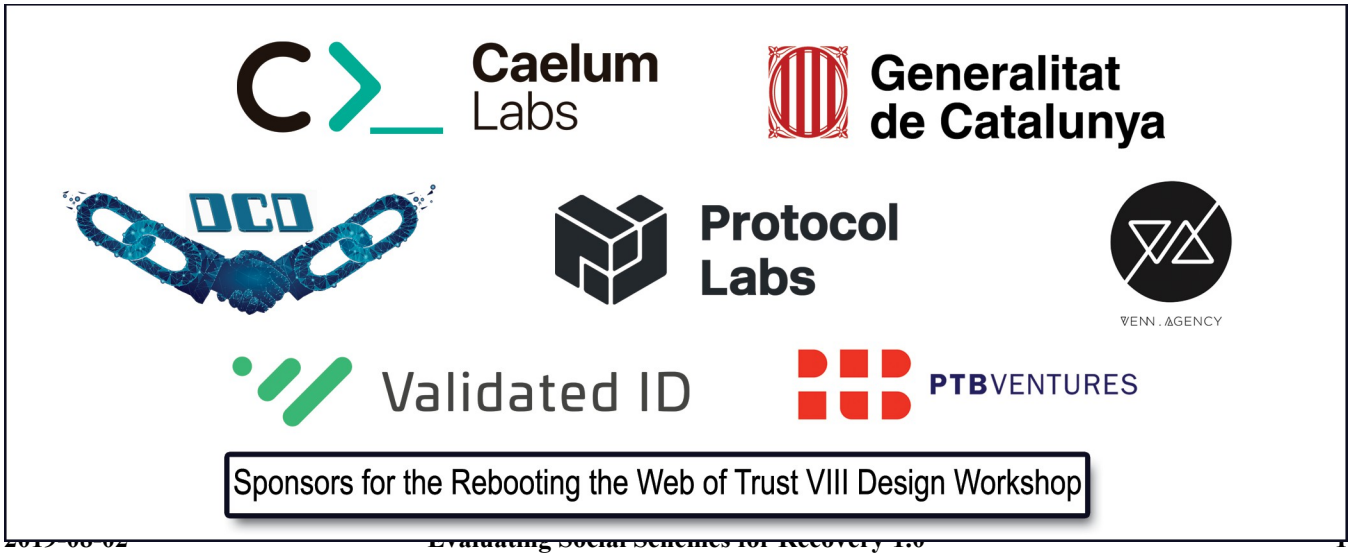
We focus on the social recovery of control of an identifier. There are several techniques to re-assert control over identifiers including key recovery and issuance of a new key. In many situations it is preferrable to establish a new key than recover the old one.

We propose a rubrik for evaluating such schemes, and give a brief overview of possible schemes to consider.

INTRODUCTION

With traditional multi-user systems, such as web-based services where users authenticate with a username and password, there is typically an authority who may revoke, re-issue, or modify identifiers when problems arise. In recent years, the limitations of this model have become more apparent, but there are also problems with giving individual users ultimate control of their identifiers.

Several proposals have been made for 'social recovery', generally involving consensus of a quorum of trusted peers chosen by the user themself. These offer a promising compromise but there are a multitude of social, technical, and contextual factors which should be considered before adopting such a scheme.



We address this problem by defining a rubrik to help determine the suitability of different social recovery schemes in a particular context.

DEFINITION OF TERMS USED IN THIS PAPER

Actors

- User: The entity whose key may need to be recovered.
- **Peers**: The entities who will be assisting the **User** in recovering their identity.
- **Adversaries**: Potential attackers who may want to hinder recovery, gain private data by eavesdropping, or impersonate the user or peers. This could involve both technical vulnerabilities and social engineering techniques.

Lifecycle of Indentifiers

- **Setup**: The user does any preparation work required to use the recovery scheme in the future.
- **Stasis**: The time between the user's setup phase and the recovery phase.
- **Recovery**: The recovery method is used to assert control over their identifier.

Identfier Recovery Scenarios

- **Lost Secret:** Control of the identifier has been lost, but it has not been compromised. For example, device has fallen in the sea.
- **Compromised Secret:** For example, device is stolen.
- **Inheritance:** After death or incapacitation we want to enable control of the identifier to our heirs.

A RUBRIK FOR EVALUATING RECOVERY SCHEMES

Actor Experience

1. Must peers know their involvement before the setup phase?

Is it possible for a peer to be a potential assistant in recovery without their knowledge? This has advantages in terms of security, as if a peer does not even know they could play a role in recovery it makes social engineering from an adversary considerably more difficult. However, it is more prone to error, as there is no possibility to gain confirmation from the peer that the setup phase was successful. There are also consent issues.

Furthermore, we should consider whether there any additional preparation required on the part of the user or peers, such as contacting them 'out of band' and asking them to install specific software.

2. Is it necessary to gain consent from trusted peers?

This might not seem like a big issue, since it concerns the potential loss of data for the user themselves, not the peers. But research from running role-playing workshops, as well as from actual experience, shows users are often uncomfortable about the idea of 'holding responsibility' of providing assistance in recovery.

This reluctance is considerably reduced if the potential peer is fully understands that they are not completely depended upon for recovery, for example in the case of threshold-based schemes, where there is a tolerance to unavailablilty of some peers. However, since these concepts are not widely understood, gaining consent can involve additional work on the part of the user, to make sure peers propery understand their roles.

3. What involvement is required from peers during stasis?

For example, do they need to keep a device online, or check a particular service for notifications?

3.1 If significant involvement is required, does the scheme have any mitigation efforts?

These could include:

- **External Incentives** (friendship, business, family). If the peer is a friend or family member, they may be willing to make an effort to be available to assist in recovery. If they are a colleage they may have a duty provide mutual support for team members.
- Utilising a peer's E**xisting Incentives.** If the scheme relies on a particular device or a private key which also serves some other purpose (such as securing funds or an account for some other service) the peer will already have a personal incentive to safeguard that device or key.
- **Explicit Rewards.** A scheme could have some formal incentive system, either by gamification (for example, the user must act as a peer for somebody else in exchange) or by a financial incentive such as cryptocurrency.

4. What involvement is required from the user during stasis?

This might include keeping a device online or checking for notifications from peers.

5. How does the scheme deal with a loss of confidence in one of the peers?

Relationships change over time and trust is not static. If a peer is a colleague, they may leave the organisation. If a peer is a friend, the relationship might go bad, or the peer's personal circumstances may change.

Furthermore a peer's personal device or key might be compromised by an adversary.

A good scheme should have a method of revoking trust from a specific peer.

Threat Model

6. Can the scheme work in a distributed architecture?

Distributed architectures or peer to peer networks have advantages for security schemes brought by the absence of a single point of failure, such as better resilience to surveillance, censorship, or takedown by an authority or the server owner, and a tolerance to particular connections being poor.

However these systems also have less desirable properties which should be considered when choosing a scheme, for example

high latency and partitioned networks where there is not connectivity between all nodes. Without a centralized 'single source of truth' it is impossible to know a given node has the most up-to-date information, which means there must be a strategy for resolving conflicts.

7. How well does the system deal with malicious behaviour of individual peers?

The assumption of threshold-based schemes is generally that peers can be trusted to the extent that *m*-of-*n* peers will not collude against the user. This means that the scheme should tolerate the misbehaviour of up to *m* - 1 peers.

8. Is there a single point of compromise during recovery?

The procedure of the recovery process should be considered carefully, as this is where operations take place that present a great threat if compromised, such as the reconstruction of a private key. Therefore one should consider factors such as where computation takes place, whether any information remain in memory, and where the results such as private keys are stored afterwards. In some cases it is appropriate to use Trusted Execution Environments (TEEs).

9. Is anonymity of the trusted peers maintained?

This can be divided into two subcategories:

- **Anonymity from outsiders**

If an adversary can determine who the peers are, they are in a better position to make an attack, either through social engineering or compromising devices or accounts belonging to the peers.

- **Anonymity between peers**

For added security, it may be desirable that the peers themselves do not know who each other are. This makes it difficult for them to maliciously collude against the user, but it also makes recovery more difficult in the inheritance case.

For lost or compromised indentifiers, the user can contact the peers (provided they can remember who they were). But in the inhertance case, where the user is incapacitated or dead, things are made complicated if the peers do not know who each other are.

One solution to this is a 'proof-of-custody' scheme. Suppose the peers all hold a common piece of data, some kind of unique identifier for the key they are protecting. In the case that a peer learns that something bad has happened to the user, they can 'broadcast' the hash of this piece of data. By broadcast we mean publish it publicly somewhere where other users of this scheme know to look. The other peers each respond **privately** with the actual piece of data, proving that they are also part of the recovery group, and they can then proceed with the recovery process.

10. What security assumptions does the scheme rely on?

This is an important part of threat modeling. We should be able to identify the hard problems an adversary faces, such as the strength of a paricular cryptographic algorithm.

External factors

11. How to measure/monitor the strength of the backup network?

- **Health Check.** Is there any way of knowing that the peers are still available and able to participate in recovery? Is there an automated means to regularly check this? Does this process require effort or costs on the part of the user or peers?

- **Transitive (indirect) Peers.** In the case that each peer's own identity is subsequently secured by a further set of peers, how can we be sure that these peers continue to exist? For security reasons, it is desirable that we do not know the identities of these secondary peers, but that we have a way of proving that the 'secondary health check' has passed successfully.

12. Does the scheme address the 'network growth problem'?

User can't rely on their key until they can rely on their peers and their peers can rely on their keys. It is desirable that each peer does 'due diligence' and ensures that their own identity is secured. Does the scheme provide a way of being able to prove this? Or at least encourage this behaviour? This requires 'mutual dependence' within the social graph, otherwise we would need an infinite number of peers to enforce this strictly.

OVERVIEW OF SCHEMES TO CONSIDER

Master Secret Recovery

Multiple keys or passwords can be generated deterministically from a single master key, which is generally stored in a seperate location, which could be offline, on paper, or even memorized.

While this is not a social scheme, it is worth noting this technique's application in indentifier recovery. A derived sub-key can be used to secure the indentifier or account. When it is lost or compromised, the master key is used to generate a new sub-key. Cryptographic techniques allow a potentially infinite number of sub-keys to be generated, so as long as the master key remains secure this process can be carried out many times.

QR codes can be used to store data on paper or transfer to an air-gapped (offline) device. Furthermore, the key can be stored as a mnemonic phrase consisting of dictionary words, which are easier for humans to read, write, or remember. Mnemonic phrased can also be engraved into hard metals such as titanium in order to survive fire.

The limitations of this technique are the existence of a single point of failure and that it relies on the good practices and resources of the user alone. For example, it is not well suited to users who are travelling. It also does not address the inheritance case.

Secret Sharing Schemes

Threshold based secret sharing schemes, originally proposed by Shamir and Blakley, can be used to recover lost keys by generating a set of shares from a given secret which are then distributed to trusted peers. Shares are points on a polynomial where coefficients are random except for the lowest, which is the secret. The order of polynomial used determines the threshold number of peers required for recovery.

Feldman, Pederson, and Berry proposed methods of introducing verification of shares. This mitigates the problem of not being able to identify which share has been maliciously or accidentally modified.

For encryption keys these schemes are very useful, as the integrity of the original key is critical for decrypting existing material. But for signing keys, when a key has been compromised we are often more interested in establishing a new key than recovering the old one. This is a major limitation, and means that such schemes are only useful for identifier recovery in the relatively rare scenario where we can be sure the key is lost but not compromised.

In this case, secret sharing could be used for identifier or account recovery as follows:

- The identifier holder, Alice, initially creates a signed message announcing that a root public key `R_pub` is the ultimate controller of Alice's identifier.
- Alice uses a secret sharing scheme to distribute the corresponding private key `R_priv` amongst m of n of her friends and deletes any traces of `R_priv` from her machine
- In the event of that Alice loses control of her identifier she contacts m of n group members and asks them to return their secret shares to her.
- Alice then computes `R_priv` and publishes a signed, timestamped message that reasserts Alice's control over her identifier.
- Any client software which seeks to validate Alice's identifier must resolve the identifiers current status by looking for messages published by `R_priv`.

Advantages:

- Anonymity of group members is maintained.
- This scheme is easily compatible with schemes where identifiers are controlled by private signing keys and where it is possible to define key or identifier hierarchies.

Issues:

- Alice must recover the root private key `R_priv` on a trusted device. If that single device is compromised Alice's identifier essentially becomes unrecoverable.
- Alice must trust her friends to protect the secrecy and integrity of her key shares, which they have no stake in.
    - If Alice's friends have public encryption keys she can encrypt the shares for her friends and store the encrypted shares in a location that is trusted for availability

Threshold Signature Schemes / Group Signatures

Group signatures, originally described by David Chaum in 1991, allow a group member to make a signature on behalf of a group, which proves they are a group member but does not reveal who they are.

Threshold-based group signatures require consensus of a specified quorum of group members, and as such have application in social recovery.

Boney-Lynn-Shacham

The Boneh–Lynn–Shacham signature scheme, which uses Weil pairings on an elliptic curve, have some desirable properties by allowing public key and signature aggregation.

Using BLS group signatures it is possible to have a signed message from *m* of *n* members of a group, with an identical aggregate signature and public key regardless of which group members signed.

This has implications for privacy, as verification is possible without needing to know which group members signed, or even the public keys of any individual group members.

MuSig

Threshold signatures are also possible with the MuSig signature scheme, which is based on Schnorr signatures and builds upon the Bellare-Nevan multi-signature scheme by also allowing key aggregation.

MuSig signatures are provably unmalleable, meaning that given a message and a signature, it is not possible to produce

another signature which is valid for that message with the same keypair.

Application of Threshold Signatures for Social Recovery

Threshold signatures could be used for identifier or account recovery as follows:

- The identifier holder, Alice, initially publishes a signed message announcing the aggregated public key of the group who are empowered to make assertions on her behalf.
- In the event of key loss or compromise, Alice generates a new keypair, sends her group members a signed, timestamped message containing her new public key, and contacts them out of band to confirm it was her.
- *m* of *n* group members each sign the message and their signatures are aggregated to produce and publish a single, signed message asserting Alice's new public key, which also serves to revoke the old one.
- Any client software which seeks to validate messages from Alice must resolve her current public key by looking for messages published by her trusted group.
- Alice publishes another signed message with her new key confirming the public key of her trusted group.
- If group members change, the new group's aggregate public key announcement must be signed by both Alice and the old group.

A major advantage of this scheme is that it addresses the case that a key has been compromised. Furthermore, anonymity of group members is maintained. Nobody but Alice can see which group members created the signature or who the group members are. Which makes it difficult for them to be targeted by someone who wanted to impersonate Alice.

A major disadvantage is that unlike secret sharing schemes like Shamirs, this recovery mechanism requires changes to the system using it. It cannot be used for existing systems that have not implemented this mechanism.

Social Identifiers Post-Recovery in Distributed Systems

Since identifiers are recovered by messages signed by a trusted group, in order for the network to learn about changes to an identifier it is important the these recovery messages are propagated.

In centralized systems this is trivial since the centralized authority is trusted to propagate messages from users.

In decentralized linear consensus systems (e.g. proof of stake blockchains) the propagation of messages can be tied to the consensus mechanism. For example, Alice knows Bob saw her identifier update method as long as Bob has seen data consensus from after Alice published her update. Alice can then assume Bob has seen her update as long as Bob would notice something wrong if the consensus system stalled/did not make progress.

In distributed systems, establishing a model for consensus is more difficult. However, one model that can be used is fork-consistency. With fork-consistency, Alice can assert that Bob has seen her update `U` as long as Bob has seen any update Alice has made since `U`. In these systems the ability to see fresh/accurate data is gated by the amount of information in the system. For example, with Bitcoin, Alice's protection against Bob seeing a forked blockchain is that if Bob has any interaction with a peer using the correct blockchain he will realize he should look for more information before using Alice's identifier.

CONCLUSION

Social recovery schemes have some very promising features but there are many social and technical factors that must be taken into account in order to use them appropriately. Due to the combination of cryptographic techniques and the behaviour of multiple human actors, the simulation and threat modelling of these schemes is particularly difficult.

While we cannot solve this, we hope that this rubrik draws attention to some factors which might otherwise have been overlooked and encourages further work in this area.

Lastly, since asserting an identifier often involves cryptographic signing, we would like to re-emphasise the distinction between the need to recover lost *encryption* keys, and the need to re-establish new *signing* keys. That is to say there is little point in recovering a key used for signing once it is assumed to be compromised. Rather, we need a mechanism for establishing a new one.

REFERENCES

- Boneh, Lynn & Shacham 'Short Signatures from the Weil Pairing' - Journal of Cryptology, Sept 2004 Vol 17 Issue 4 p297-319
- Gennaro Jarecki - Secure distributed key generation for discrete log
- Threshold Signatures on the keep network
- Parity's 'secret store' distributed key generation
- BLS Signatures: better than Snorr - Medium ariticle
- Practical Threshold Signatures - Victor Shoup on RSA, from 2000
- Dfinity's implementation of Distributed Key Generation using BLS
- web based demo of verifiable secret sharing using BLS
- Colic, Petar Hlad - Anonymous Threshold Signatures
- Back and Zheng - Identity-Based Threshold Signature Scheme from the Bilinear Pairings
- Gregory Maxwell, Andrew Poelstra1, Yannick Seurin, and Pieter Wuille - Simple Schnorr Multi-Signatures with Applications to Bitcoin
- Boneh, Drijvers and Neven, 'BLS Multi-Signatures With Public-Key Aggregation' 2018
- Permanent Revocation Systems - Brownstein, Gilboa,Dolev
- Chaum, David; van Heyst, Eugene (1991). 'Group signatures' Advances in Cryptology — EUROCRYPT '91. Lecture Notes in Computer Science. 547. pp. 257–265.

APPENDIX - THE RUBRIK QUESTIONS

Actor Experience

1. **Must peers know their involvement before the setup phase?**
2. **Is it neccessary to gain consent from trusted peers?**
3. **What involvement is required from peers during stasis?**
    1. **If significant involvement is required, does the scheme have any mitigation efforts?**
        - **Are there External incentives?**
        - **Are there Existing incentives?**
        - **Are there Explicit rewards?**
4. **What involvement is required from the user during stasis?**
5. **How does the scheme deal with a loss of confidence in one of the peers?**

Threat Model

6. **Can the scheme work in a distributed architecture?**
7. **ow well does the system deal with malicious behaviour of individual peers?**
8. **Is there a single point of compromise during recovery?**
9. **Is anonymity of the trusted peers maintained?**

- **Is there Anonymity from outsiders?**
- **Is there Anonymity between peers?**

10. **What security assumptions does the scheme rely on?**

External factors

11. **How to measure/monitor the strength of the backup network?**
    - **Is there some kind of 'health check'**
    - **Is there a secondary 'health check' for transitive (indirect) peers?**
12. **Does the scheme address the 'network growth problem'?**

Additional Credits

Lead Author: Gregory Jones

Authors: Sean Gilligan, Adin Schmahmann, Andrew Hughes, and Christopher Allen

Sample APA Citation:

Gregory Jones, Gilligan, S., Schmahmann, A., Hughes, A., and Allen, C. (2019). Evaluating Social Schemes for Recovering Control of an Identifier. *Rebooting the Web of Trust VIII*. Retrieved from https://github.com/WebOfTrustInfo/rwot8-barcelona/blob/master/final-documents/evaluating-social-recovery.pdf.

About Rebooting the Web of Trust

This paper was produced as part of the Rebooting the Web of Trust VIII design workshop. On March 1st to 3rd, 2019, over 80 tech visionaries came together in Barcelona, Spain to talk about the future of decentralized trust on the internet with the goal of writing at least 5 white papers and specs. This is one of them.

RWOT Board of Directors: Christopher Allen, Joe Andrieu, Kim Hamilton Duffy

Silver Sponsors: Caelum Labs, Digital Contract Design, Generalitat de Catalunya, Protocol Labs, Venn Agency

Additional Sponsors: Validated ID, PTB Ventures

Community Sponsors: Blockchain Commons, Digital Bazaar, In Turn Information Management Consulting, Learning Machine, Legendary Requirements

Workshop Credits: Christopher Allen (Founder), Joe Andrieu (Producer and Facilitator), Shannon Appelcline (Editor-in-chief), and Carlotta Cataldi (Graphical Recorder)

Thanks to our other contributors and sponsors!

What's Next?

The design workshop and this paper are just starting points for Rebooting the Web of Trust. If you have any comments, thoughts, or expansions on this paper, please post them to our GitHub issues page:

https://github.com/WebOfTrustInfo/rwot8/issues

The ninth Rebooting the Web of Trust design workshop is scheduled for September 3rd-6th 2019 in Prague, The Czech Republic. If you'd like to be involved or would like to help sponsor the event, email:

rwot-leadership@googlegroups.com
