Recovery via shared seed phrase - proof of history v.0.0.4

Overview: The objective of the protocol is to allow a user to recover its private key if it is lost in an environment where peers exchange social content. The idea is to use the shared memory of users from chat histories as a proof of identiy/backup mechanism for passwords. In order to regain access to its private key, a recovering user has to iteratively identify and associate social content (media files, links, texts) from a chat history to the correct peers, similar to reCAPTCHA. A successful protocol run statiscally indicates the user had prior knowledge of the private key.

Procedure:

Assumption: A secure channel exists S_o, the user has still access to its private key. The user has interacted with enough peers to establish a solid social graph which allows the algorithm to determine suitable peers for recovery.

The private key s_o or a derivative is locally split in k pieces, where: $s_o = \oplus_{i=1}^{k} s_i$

The XOR works as a provisional one time pad. (# salt necessary, plaintextspace not large enough, padding from encryption later might be sufficient #)

An additional secure channel S_r must be established preliminary during setup phase only for the recovery process, while the user still has access to its private keys, yielding a keypair p_r, s_r. The pieces are encrypted: the public key is distributed through the original secure channel S_o and stored by the peers.
The pieces are steganographically encoded in messages $m_1, ..., m_k$ (an image or media file, consider client side privacy filter, to prevent privacy leaking through triggering the recovery). The k pieces are distributed to peer devices, through the original secure channel. The encrypted pieces must be removed from the users device after generation.

On recovery, after social verification (via phone/video call) the recovering user has to identify the carrier messages $m_1, ..., m_k$ from a random set. The random set has to be generated on the peers side, assuming adversarial takeover attempt, the correct images must not be leaked during social verification, the set $m_1, ..., m_k$ does not have to be exposed to the user and can be determined automatically, preventing social engineering). If the user manages to identify the images with a low enough error rate, they get pulled in from the peers devices and can be decrypted to recover the seed phrase.

System parameters and analysis: n peers, k encoded pieces of the private key are distributed. $n/k > 1$ is the redundancy of the system in case peers left or unavalaible, where k increases both security and overhead of the protocol. With $j > k$, the number of pieces a recovering user is presented, where only k were part of the communication history, yields a probability of k/j to guess a correct piece. To further decrease the likelihood of an attacker to guess the images correctly, the user is presented a batch with batchsize $b < j$ of pieces and aborting the recovery procedure if the attacker guesses less than t_1 of a batch right or after overall t_2 wrong answers.

The probability to have $\geq t_1$ correct guesses in a batch with batchisze b is given by:

$$P(X \geq t_1) = \sum_{i=t_1}^{b} B(b; \frac{k}{j}; i) = \sum_{i=t_1}^{b} \binom{b}{i} \left(\frac{k}{j}\right)^i \left(1 - \left(\frac{k}{j}\right)\right)^{b-i}$$

The likelihood for an attacker to have $\geq t_2$ correct guesses from a random set of size j with k key-pieces is equal to:

$$P(Y \geq t_2) = \sum_{i=t_2}^{j} B(j; \frac{k}{j}; i) = \sum_{i=t_2}^{j} \binom{j}{i} \left(\frac{k}{j}\right)^i \left(1 - \left(\frac{k}{j}\right)\right)^{j-i}$$