Lost and Found in the Fog of Trust

Özgür Kesim
Freie Universität Berlin
Berlin, Germany
Code Blau GmbH
Berlin, Germany
o.kesim@fu-berlin.de

Christian Grothoff
Bern University of Applied Sciences
Bern, Switzerland
christian.grothoff@bfh.ch

Abstract

The Fog of Trust protocol was supposed to allow a prover Peggy and a verifier Victor to perform a secure multiparty computation to determine the number of third parties that Victor trusted and that vouched for a certain property of Peggy. We intended to use formal methods to first state and then prove the privacy properties of the protocol. Instead, the analysis lead to the discovery of two previously unknown design failures that allow an adversary to break the privacy assurances the protocol was expected to provide. This paper briefly presents the Fog of Trust protocol and the vulnerabilities, which at this point we are unable to fix.

CCS Concepts: • **Security and privacy → Privacy-preserving protocols**.

Keywords: secure multiparty computation, formalization, attack

ACM Reference Format:
Özgür Kesim and Christian Grothoff. 2024. Lost and Found in the Fog of Trust. In *Proceedings of the Workshop Dedicated to Jens Palsberg on the Occasion of His 60th Birthday (JENSFEST '24), October 22, 2024, Pasadena, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3694848.3694853

1 Introduction

The main purpose of the Fog-of-Trust protocol is to enable two parties, who privately own sets A and B respectively, to calculate $|A \cap B|$, i.e. the size of the intersection, *without* disclosing the elements in their sets to the other party. Additionally, only one of the two parties actually should learn $|A \cap B|$, and to prevent the other party from cheating, their elements must be digitally signed, preventing them from including arbitrary elements in their set.

The idea for the Fog-of-Trust (FoT) protocol dates back to 2016 where it was proposed as a component in an online abuse detection system to securely and privately compute properties over the social graph in decentralized online social networks [4]. The protocol was formalized and implemented in 2023 [6], this time with a focus on addressing the well-known privacy issues of the widely used email encryption protocol OpenPGP and its web of trust [8] (i.e. the social graph resulting from trust relationships between individuals), with an eye on providing more automation to also address usability concerns with OpenPGP [9]. Beyond OpenPGP, a privacy-friendly method to securely associate public keys with identities in a decentralized way would be useful for a wide range of applications requiring secure communication, especially in the domain of peer-to-peer payments over the Internet. We analyzed the descriptions of the FoT protocol of [4] and [6] with the goal of formalizing the security and privacy properties and then proving those properties. Instead of a proof, we ended up finding fundamental flaws in the design of FoT that completely undermine its privacy promises.

The FoT is supposed to be a private set intersection cardinality protocol performed between two parties, *Peggy, the prover*, and *Victor, the verifier*. Both parties are assisted by many additional *Walters, the witnesses*. Peggy and Victor have interacted with the Walters before executing the core part of the FoT protocol. The Walters are thus not part of the online interaction. The motivation for the protocol is that Victor will accept a statement about Peggy if a sufficient number of Walters have affirmed that statement. The statement could be that Peggy:

- uses a certain public key (for example, to check the association of public keys with e-mail addresses as done by the Web of trust for OpenPGP),
- is part of a certain social group (a statement of group membership and limit abuse in online social networks as envisioned by [4]), or
- is a victim deserving special consideration (for example, refugees may be asked to provide witnesses for abuse in a real-world asylum process).

Before the core part of the FoT protocol is run, Peggy collected digital signatures in support of the statement from a number of witnesses (all called Walter). Victor similarly built a database of trusted Walters, as, given the possibility of Sybil attacks [3] where Peggy just digitally simulates the

JENSFEST '24, October 22, 2024, Pasadena, CA, USA

ACM ISBN 979-8-4007-1257-9/24/10
https://doi.org/10.1145/3694848.3694853

existence of additional Walters, Victor must only accept testimony of those Walters that he considers qualified as witnesses. Privacy requirements arise in scenarios where Peggy is not willing to disclose the identities of her witnesses (her personal connections), and also Victor is not willing to disclose the identities of his trusted acquaintances.

Formally speaking, the protocol starts with a situation where Victor has a set T of public keys of Walters that he trusts and Peggy has a message m encoding the statement she wants to convince Victor of, together with a finite set of pairs (p_i, σ_i) of public keys $P = \{p_i\}$ and corresponding signatures σ_i over m.

The aim of the Fog-of-Trust protocol is then for Victor to learn the size of the intersection $P \cap T$, that is the number of trusted witnesses for the statement m, while also providing Victor with a zero-knowledge proof that Peggy has valid signatures σ_i over m by p_i, but without Victor learning anything about P or Peggy learning anything about T. The latter was expected to be relaxed to cover obvious edge cases, such as the order of magnitude of the size of the sets (which may be sufficiently obscured by padding) or information that can be deduced from $|P \cap T|$ and T.

Our original objective was to provide formal definitions of the security properties of the protocol via securtiy games, and to provide their formal proofs. However, during the course of formulating appropriate security games for different aspects of the protocol, two of the games turned out to be winnable by an adversary, disclosing major flaws in the *design* of the protocol: $P \cap T$ is fully disclosed to a malicious Victor, and in addition, Peggy can — under certain circumstances — learn T.

In the following sections we will first formalize the Fog-of-Trust protocol and define the security proterties we are interested in via security games. We then show how one of the games can be won by an adversary by design, and another game can be won by the adversary in an concrete instantiation of the protocol with specific cryptographic primitives under certain circumstances.

2 Fog-of-Trust Protocol

To motivate the construction of the Fog-of-Trust protocol, consider the following simplified situation for the two parties: Victor has a finite set $T = \{t_1, \ldots, t_n\}$ of trusted public keys and Peggy has (σ, p, m) — a signed message m along with a signature σ that both can be verified with the public key p. Victor wants to know a) if σ is a valid signature for m, verifiable with p, and b) if $p \in T$. Peggy also wants to prove a) and b), but not disclose p itself.

To achieve this, let's assume both parties are provided with a so-called *blinding mechanism* that for a given message m transforms a valid (signature, public-key) pair (σ, p) into another valid (signature, public-key) pair (τ, q) for the same message m, and which is *unlinkable* to the original pair by either party participating in the transformation. Unlinkable here means that given some (τ, q) derived from either (σ_1, p_1) or (σ_2, p_2), it is not practical to determine whether (τ, q) was derived from (σ_1, p_1) or from (σ_2, p_2). Incidentally, the transformation can also operate only on the public keys.

If such a blinding mechanism exists and can be trusted by both parties, they can use it to 1.) substitute Victors's set of public keys T of trusted Walters with another equivalent set $Q = \{q_1, \ldots, q_n\}$ (without knowing which t_i corresponds with which q_j) and 2.) substitute Peggy's pair (σ, p) with another valid pair (τ, q) for the same message. That way, Victor can a) verify the signature τ for message m with public key q and b) check if $q \in Q$, but without learning which of the original t_i corresponds to q.

The aim of the Fog-of-Trust protocol is to create such a blinding mechanism that is trusted by both parties and achieves the wanted results with certainty, up to an adjustable error margin.

Following the description in [6], the core functionality of Fog-of-Trust will be defined in terms of five function signatures, which are explained below. These functions and their corresponding spaces must fulfill particular properties which will be defined next. An instantiation of the protocol can freely choose these functions, provided they fulfill the requirements. We will later provide a particular instantiation of the protocol with concrete functions based on cryptographic primitives originally proposed by [4] and also used in [6].

Without loss of generality, we shall assume $\mathbb{M} = \{0,1\}^*$ and $\mathbb{P}, \mathbb{K}, \mathbb{S}, \mathbb{H}, \Omega$ all denote subsets of $\mathbb{M}$, with $\mathbb{P}, \mathbb{K}, \mathbb{S}$ of finite size and $\Omega = \{0,1\}^\lambda$, $\mathbb{H} = \{0,1\}^\mu$ consisting of elements of fixed lengths $\lambda, \mu \in \mathbb{N}$. None of the sets must be prone to exhaustive search in polynomial time.

The five functions we need for the construction of the Fog-of-Trust protocol shall have the following function signatures:

$$\begin{aligned}
\mathsf{KeyGen}: \quad & \Omega \to \mathbb{P} \times \mathbb{K} \\
& \omega \mapsto (p, s) \\
\mathsf{Sign}: \quad & \mathbb{M} \times \mathbb{K} \to \mathbb{S} \\
& (m, s) \mapsto \sigma \\
\mathsf{Verify}: \quad & \mathbb{S} \times \mathbb{P} \times \mathbb{M} \to \mathbb{Z}_2, \\
& (\sigma, p, m) \mapsto b \\
\mathsf{Hash}: \quad & \mathbb{M} \to \mathbb{H} \\
& x \mapsto h \\
\mathsf{Blinding}_{(\cdot)}: \quad & \Omega \to \mathsf{Aut}(\mathbb{S}) \times \mathsf{Aut}(\mathbb{P}) \\
& \omega \mapsto \big(\omega_{\mathbb{S}}(\cdot), \omega_{\mathbb{P}}(\cdot)\big) = \mathsf{Blinding}_\omega
\end{aligned}$$

The functions KeyGen, Sign and Verify are expected to be a strong asymmetric signature-scheme, with additional requirements in relation to Blinding, see below. KeyGen shall

generate a pair of (public, private) keys from a given random value, Sign shall generate the signature for a given message m and a private key s, and Verify shall return 1 only when the provided signature σ for message m is valid for the given public key p. The function Hash is expected to be a cryptographically strong hash function, which allows Peggy to generate commitments to data without immediately disclosing the data to Victor.

As motivated in the introduction to this section, Blinding shall provide us with randomly chosen mechanisms to generate from any given pair $(\sigma, p) \in \mathbb{S} \times \mathbb{P}$ another pair (τ, q) of signatures and public keys. Such two pairs of signatures and public keys must both be able to verify the same message, and we therefore require the invariance of Verify under blinding and the commutativity of any two blinding operations. That is, by trivially extending

$$\mathsf{Blinding}^{+}_{\omega} = \left(\omega_{\mathbb{S}}, \omega_{\mathbb{P}}, id_{\mathbb{M}}\right)$$

we require a) for all $\omega \in \Omega$:

$$\mathsf{Verify} \circ \mathsf{Blinding}^{+}_{\omega} = \mathsf{Verify}$$

and b) for all $\alpha, \beta \in \Omega$:

$$\mathsf{Blinding}_{\alpha} \circ \mathsf{Blinding}_{\beta} = \mathsf{Blinding}_{\beta} \circ \mathsf{Blinding}_{\alpha}$$

In particular, two blindings must commute over $\mathbb{P}$:

$$\alpha_{\mathbb{P}} \circ \beta_{\mathbb{P}} = \beta_{\mathbb{P}} \circ \alpha_{\mathbb{P}}$$

We further denote with $\pi_{\mathbb{P}} : \mathbb{S} \times \mathbb{P} \to \mathbb{P}$ the projection onto $\mathbb{P}$ and extend Hash to multiple arguments $h : \mathbb{M}^n \to \mathbb{H}$, for any $n > 0$, by writing $h(x_1, \cdots, x_n) := \mathsf{Hash}(x_1 \| \cdots \| x_n)$. We also use salted hash functions $h_\omega(x) := h(x \| \omega)$, for $\omega \in \Omega$.

With these ingrediences, we can construct a trusted blinding mechanism, in which Victor and Peggy jointly establish the same double-blinding for their sets of public keys, reminiscent of the Diffie-Hellman key-exchange protocol [2]. Note that Peggy also has to show the correct application of the *same* blinding onto both sets (her public keys and the blinded public keys from Victor) by incorporating a zero-knowledge, cut&choose proof. The resulting protocol *Fog-of-Trust* is then defined in Figure 1.

3 Security Requirements and Games

The main privacy goals of the protocol is that Victor learns nothing about P or any element in $P \cap T$ during the protocol, and Peggy learns nothing about the trustees T of Victor.

If we assume an honest Peggy, who applies the blinding to both sets honestly, we can then drop the need for the cut&choose proof – or equivalently: set the cut&choose parameter $\kappa = 1$. As a result, the commitments and all redundant steps in the protocol can be removed and the whole protocol reduces to the one given in Figure 2.

However, even in this reduced variant of the protocol, Victor must not be able to learn anything about the *content* of the intersection of both double-blinded sets, and Peggy must not be able to learn anything about the contents of the set of trusted public keys of Victor.

To make these requirements more precise, we define two security games that are performed by an honest party and an adversary, who execute the reduced protocol. The games are defined such that winning means gaining knowledge about some secret information. The adversary is allowed to "cheat" in their steps during the execution of the protocol. The security requirements for the reduced protocol would then be defined as the inability of an adversary to win a game with higher than expected probability.

The two games for the reduced protocol are formalized in Figure 3 and are described as follows:

$\mathsf{Identification}_1(n)$**:** In this game Victor is the adversary, trying to gain information about Peggy's witnesses. Victor starts with a set of n public keys $\{p_i\}_{i \in \{1,\ldots,n\}}$ of suspected witnesses, and Peggy has a single valid signature for the public key p_c for some random $c \in \{1, \ldots, n\}$.
The objective for Victor is to determine the value of c (or equivalent p_c), and thus to identify Peggy's witness in his set of suspects.
The adversary playing as Victor is allowed to invoke an arbitrary polynomial time algorithm

$$\mathcal{A}_1 : \mathcal{P}(\mathbb{P}) \to \mathcal{P}(\mathbb{P})$$

on finite subsets of public keys as the initial, generalized blinding step, and later a polynomial time algorithm

$$\mathcal{A}_2 : \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \times \mathbb{P} \to \mathbb{P}$$

to identify the randomly chosen signatory from the (double) blinded set of trustees.
Given that the set size intersection in the game is fixed at 1 and Victor starts with n suspects, this implies that the adversary should only be able to guess c with probability n^{-1}.

$\mathsf{Identification}_2(n)$**:** In this game Peggy is the adversary, trying to gain information about Victor's trustees T. The adversary is allowed to collaborate with any of the trustees in order to achieve her goal. That is, the adversary (Peggy) is allowed to perform an arbitrary polynomial time algorithm on all blinded public keys of the trustees, the public *and* private keys of the signatories along with a blinded public key, and has to find out which key was blinded:

$$\mathcal{B} : \mathcal{P}(\mathbb{P}) \times \mathcal{P}\left(\mathbb{P} \times \mathbb{K}\right) \times \mathbb{P} \to \mathbb{P}$$

In particular, the adversary (Peggy) is challenged to recognize one of the public keys $\{p_j\}$ from the blinded set T' which Victor transmits in the first set of the reduced protocol. The game allows the adversary access

Victor

Has finite $T \subset \mathbb{P}$. // *Victor starts with a set of public keys of trusted witnesses.*

1 : $\alpha \leftarrow\!\$\ \Omega$ // *Victor picks a random number α for blinding,*

2 : $(\_, \alpha_{\mathbb{P}}) \leftarrow \mathsf{Blinding}_{\alpha}$ // *limits the blinding function to public keys*

3 : $\downarrow T' \leftarrow \alpha_{\mathbb{P}}(T)$ // *and sends Peggy a randomly blinded version of this set.*

Peggy

Has $m \in \mathbb{M}$, finite $S_m \subset \mathbb{S} \times \mathbb{P}$ // *Peggy starts with a message with signatures for multiple public keys.*

and let $P := \pi_{\mathbb{P}}(S_m)$.

4 : **for** $i \leftarrow 1, \ldots, \kappa$ // *For the initial cut&choose step, she prepares κ many variants*

5 : $(\beta_i, \omega_i) \leftarrow\!\$\ \Omega \times \Omega$ // *of randomly chosen blindings and salts for*

6 : $\mathbf{S}'[i] \leftarrow (h_{\omega_i} \circ \mathsf{Blinding}_{\beta_i})(S_m)$ // *1) commitments for her blinded signatures and public keys, and*

7 : $\mathbf{Q}[i] \leftarrow (h \circ \beta_{i,\mathbb{P}})(T')$ // *2) commitments for the (now double-)blinded trusted keys of Victor,*

8 : $\downarrow (m, \mathbf{S}', \mathbf{Q})$ // *which she all sends to Victor, together with her message.*

Victor

9 : $\downarrow \mathbf{C} \leftarrow\!\$\ \{0,1\}^{\kappa} \setminus \{(0, \ldots, 0)\}$ // *Victor decides which of the κ blindings Peggy must disclose.*

Peggy

10 : **for** $i \leftarrow 1, \ldots, \kappa$ // *Depending on each of the κ bits in* **C**, *Peggy either discloses*

11 : $\mathbf{L}[i] \leftarrow \begin{cases} \beta_i & \textbf{if } \mathbf{C}[i] = 0 \\ (\omega_i, \mathsf{Blinding}_{\beta_i}(S_m)) & \textbf{otherwise} \end{cases}$ // *her blinding, or the salt and her blinded set of (sig, key)-pairs,*

12 : $\downarrow \mathbf{L}$ // *and sends the result to Victor.*

Victor

13 : $R \leftarrow \emptyset$

14 : **for** $i \leftarrow 1, \ldots, \kappa$ // *Victor can now, for each of the κ bits in* **C**, *confirm that*

15 : **if** $\mathbf{C}[i] = 0$

16 : $\beta \leftarrow \mathbf{L}[i]$

17 : **assert** $(\mathbf{Q}[i] = (h \circ \beta_{\mathbb{P}})(T'))$ // *either Peggy's commitment for double-blinding was correct, or*

18 : **else**

19 : $(\omega, S'_m) \leftarrow \mathbf{L}[i]$

20 : **assert** $(\mathbf{S}'[i] = h_{\omega}(S'_m))$ // *her commitment for the blinded (signature, key)-pairs were correct,*

21 : **assert** $(\mathsf{Verify}(S'_m \times \{m\}) = \{1\})$ // *and the signatures are actually valid.*

22 : $Q \leftarrow (h \circ \alpha_{\mathbb{P}} \circ \pi_{\mathbb{P}})(S'_m)$ // *He then applies his original blinding α to Peggy's (blinded) keys,*

23 : $R \leftarrow R \cup \{|Q \cap \mathbf{Q}[i]|\}$ // *and saves the size of the intersection of the two double blinded sets.*

24 : **assert** $|R| = 1$ // *Only when all the intersections are of the same size,*

25 : **return** n $(n \in R)$ // *the protocol ran correctly, and it returns that size.*

Figure 1. The Fog-of-Trust protocol — The protocol runs from top to bottom, step by step, and the acting party is headed with a boxed name. The data transfered to the next party is marked with a downarrow ↓ *data*. Additional steps for storing session data and verification of transmitted data are left out for better readability.

Victor

Has finite $T \subset \mathbb{P}$.

$\alpha \leftarrow\!\!\$\ \Omega$

$(\_, \alpha_{\mathbb{P}}) \leftarrow \mathsf{Blinding}_{\alpha}$

$\downarrow T' \leftarrow \alpha_{\mathbb{P}}(T)$

Peggy

Has $m \in \mathbb{M}$, finite $S_m \subset \mathbb{S} \times \mathbb{P}$

$\beta \leftarrow\!\!\$\ \Omega$

$S' \leftarrow \mathsf{Blinding}_{\beta}(S_m)$

$T'' \leftarrow \beta_{\mathbb{P}}(T')$

$\downarrow (m, S', T'')$

Victor

assert $(\mathsf{Verify}(S' \times \{m\}) = \{1\})$

$Q \leftarrow (\alpha_{\mathbb{P}} \circ \pi_{\mathbb{P}})(S')$

return $|Q \cap T''|$

Figure 2. Reduced naïve variant of the Fog-of-Trust protocol — In this variant, Peggy is trusted to perform her part of the blinding honestly in the Fog-of-Trust protocol in Figure 1. The security parameter κ for the cut&choose proof in the original protocol is set to 1, and all commitments and redundant steps are removed from the protocol. As the reduced protocol exposes a strict *subset* of the data exposed by the full protocol, successful information disclosure attacks against the reduced protocol trivially extend to the full protocol.

to the full set of all public *and* private keys of all possible trustees. As we will see later, our attack only requires access to two private keys in the set, but with respect to the privacy assurances we have to assume that an adversary may collaborate with many potential trustees and then wants to learn which of their collaborators have earned Victor's trust. In the game, a random potential trustee p_c is singled out and the adversary is challenged to return p_c given only the blinded key $q_c \leftarrow \alpha_{\mathbb{P}}(p_c)$. Note that T' is provided as an unordered set, so the adversary cannot determine c simply by looking at the offset of $q_c \in T'$. Given that the game assures the adversary that $q_c \in T'$, the adversary should only be able to guess p_c with probability n^{-1}.

These two games only formalize a *subset* of the security and privacy objectives behind the full FoT protocol from Figure 1. However, without these two key properties, the rest of the protocol is basically pointless.

$\mathsf{Identification}_1(n)$

$\vec{\omega} = (\omega_1, \ldots, \omega_n) \leftarrow\!\!\$\ \Omega^n$

$(p_1, \ldots, p_n) \leftarrow (\pi_{\mathbb{P}} \circ \mathsf{KeyGen})^n(\vec{\omega})$

$P \leftarrow \{p_i\}_{i \in \{1,\ldots,n\}}$

$\overline{P} \leftarrow \mathcal{A}_1(P)$

$(c, \beta) \leftarrow\!\!\$\ \{1, \ldots, n\} \times \Omega$

$(\_, \beta_{\mathbb{P}}) \leftarrow \mathsf{Blinding}_{\beta}$

$Q \leftarrow \beta_{\mathbb{P}}(\overline{P})$

$q_c \leftarrow \beta_{\mathbb{P}}(p_c)$

$p \leftarrow \mathcal{A}_2(P, Q, q_c)$

return $\begin{cases} 1 & \textbf{if } p_c = p \\ 0 & \textbf{otherwise} \end{cases}$

$\mathsf{Identification}_2(n)$

$\vec{\omega} = (\omega_1, \ldots, \omega_n) \leftarrow\!\!\$\ \Omega^n$

$K := \{(p_i, s_i)\}_{i \in \{1,\ldots,n\}} \leftarrow \mathsf{KeyGen}^n(\vec{\omega})$

$(c, \alpha) \leftarrow\!\!\$\ \{1, \ldots, n\} \times \Omega$

$(\_, \alpha_{\mathbb{P}}) \leftarrow \mathsf{Blinding}_{\alpha}$

$T' \leftarrow (\alpha_{\mathbb{P}} \circ \pi_{\mathbb{P}})(K)$

$q_c \leftarrow \alpha_{\mathbb{P}}(p_c)$

$p \leftarrow \mathcal{B}(T', K, q_c)$

return $\begin{cases} 1 & \textbf{if } p_c = p \\ 0 & \textbf{otherwise} \end{cases}$

Figure 3. The games for identification of signatories and trustees in the reduced protocol — In the first game, Victor is considered the adversary, trying to identify which public key is the right witness for the message. In the second game, Peggy is the adversary, trying to identify a single trusted public key from Victor when given only a blinded version of it. In both games, the respective adversary is allowed to perform arbitrary polynomial timed algorithms on the provided data in their steps during the execution of the reduced Fog-of-Trust protocol from Figure 2.

4 Analysis

In the following we present for each game a strategy for an adversary to win the game. It is worth noting that the first game can be won for *any* instantiation of the procotol, i.e. it is independent of the particular choices of the required functions.

4.1 Winning Game $\mathsf{Identification}_1(n)$

The strategy to win the game $\mathsf{Identification}_1(n)$ by the adversary is by choosing *unique* blindings, one per $p \in P$. To

achieve this, the adversary starts with a random $\alpha \xleftarrow{\$} \Omega$, and derives a seed $h_\alpha(p)$ for any $p \in \mathbb{P}$ and picks the blinding $\alpha_p(\cdot) \in \mathsf{Aut}(\mathbb{P})$ to be

$$(\_, \alpha_p(\cdot)) \leftarrow \mathsf{Blinding}\big(h_\alpha(p)\big)$$

Note that each such blinding is now uniquely associated with an indivual $p \in P$. The first adversarial function is then defined as follows:

$$\mathcal{A}_1(\alpha) : \mathcal{P}(\mathbb{P}) \to \mathcal{P}(\mathbb{P})$$
$$P \mapsto \{\alpha_p(p) \mid p \in P\} =: \overline{P}$$

For $\mathcal{A}_2$, the adversary is given $(P, Q, \overline{p}) \in \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \times \mathbb{P}$, where $Q \leftarrow \beta_{\mathbb{P}}(\overline{P})$. They can now identify the chosen $p_c \in P$ by iterating over $q \in P$ and checking for which q

$$\alpha_q(\overline{p}) \stackrel{?}{\in} Q \tag{1}$$

holds in $O(n^2)$. This relationship will always be true for $q = p_c$ because

$$\alpha_q(\overline{p}) = \alpha_q(\beta_{\mathbb{P}}(p_c)) = \beta_{\mathbb{P}}(\alpha_q(p_c)) \in Q.$$

We now show that $q \in Q$ with $q = p_c$ is with high probability the only solution to Equation (1). Let $q' \in P \backslash \{p_c\}$ be another key evaluated by the adversary. The bijectivity of $\beta_{\mathbb{P}}$ implies that the probability of

$$\alpha_{q'}(\overline{p}) = \alpha_{q'}(\beta_{\mathbb{P}}(p_c)) = \beta_{\mathbb{P}}(\alpha_{q'}(p_c)) \stackrel{?}{\in} Q.$$

is equivalent to that of

$$\alpha_{q'}(p_c) \stackrel{?}{\in} \{\alpha_p(p) \mid p \in P\}$$

Given a collision-resistant hash function for $h_\alpha(p)$, the chance of a random collision between the two automorphisms (for a $q' \neq p_c$) is approximately $\frac{|P|}{|\mathbb{P}|}$, which for a large domain $\mathbb{P}$ is negliglible. Thus, the adversary almost always wins the game.

4.2 Winning Game $\mathsf{Identification}_2(\mathsf{n})$

For this game, we will formulate a winning strategy of the adversary in the context of the concrete instantiation of the FoT-protocol from [6] which is based on BLS signatures [1]. The choices of KeyGen, Sign and Verify are the corresponding functions from the BLS scheme.

Specifically, in BLS signatures, a private key x is a scalar and the element g^x in the underlying multiplicative group is the corresponding public key. For signing, messages must be mapped onto elements in the group, and a corresponding element h is then signed in BLS by computing the signature $\sigma := h^x$. Validation of signatures requires a non-trivial bilinear map $e : G_1 \times G_2 \to G_T$ for which then

$$e(\sigma, g) = e(h, g)^x = e(h, g^x)$$

holds for valid signatures. See [1] for details on constraints for suitable groups.

The function Blinding is defined simply as exponentiation in the underlying multiplicative group, for both signatures and public keys: Given signature $\sigma = h^x$ and corresponding public key g^x on message h, we can blind the signature σ and public key g^x with a scalar b:

$$e(\sigma^b, g) = e(h, g)^{xb} = e(h, g^{xb})$$

Thus, σ^b is a valid signature for the same message h, verifiable by the derived public key $(g^x)^b$.

In the game $\mathsf{Identification}_2(\mathsf{n})$ the adversary $\mathcal{B}$ gets

$$\big(T', K, q_c\big) = \big(\alpha_{\mathbb{P}}(\{p_i\}), \{(p_i, s_i)\}, \alpha_{\mathbb{P}}(p_c)\big), i \in \{1, \ldots, n\}$$

which in the BLS instantiation is

$$\big(\ \{g^{s_i \alpha}\}, \{(g^{s_i}, s_i)\}, g^{s_c \alpha}\ \big), i \in \{1, \ldots, n\}.$$

The adversary now iterates over all $s_j, j \in \{1, \ldots, n\}$ and all $x_i \in \{g^{s_i \alpha}\}, i \in \{1, \ldots, n\}$ and calculates $x_i^{-s_j}$, until they find two distinct pairs (i, j) and (i', j') where $x_i^{-s_j}$ and $x_{i'}^{-s_{j'}}$ are the same value — which in this case must be g^α with high probability (where the exponent α remains unknown to $\mathcal{B}$). The adversary then iterates again over the s_i and compares $(g^\alpha)^{s_i}$ to $g^{s_c \alpha}$ until they find a match and finally return the matching p_i.

5 Discussion

The presented games correspond to the information available and the computations performed by an adversary in the reduced protocol, and thus trivially extend to the full protocol. The special case of an intersection set size of 1 used in the first game is not fundamental to breaking the security; it simply makes it easier to express the desired privacy property. Similarly, for the second game, the attack does not require the adversary to have access to all private keys, as the approach works as soon as the adversary knows at least two private keys.

Giving an adversary access to private keys of parties that they do not play as might seem a bit unusual in a security model. However, especially in distributed systems we have to assume that an adversary may participate in protocols in multiple roles. The breaking of the second game is thus a bit reminiscent of Dominic Tarr's attack on 3DH, where he showed that disclosing one party's private key to the adversary provides the adversary with a "wildcard" which voids that party's ability to authenticate messages [7]. These types of attacks demonstrate that it is important for security games to model adversaries with (partial) access to sensitive information, including secret keys.

6 Conclusion

The Fog-of-Trust protocol of [4] is broken beyond repair. However, many applications are in need of a solution to the problem of threshold-validation of statements m by a set of semi-trusted witnesseses. The modern canonical answer for complex secure multiparty computations is the use of some kind of zk-SNARK [5] construction. However, in

practice, this particular application domain comes with additional challenges, such as the set of witnesses being possibly unknown at the beginning, witnesses signing statements independent of each other and prior to the execution of the actual protocol. A possible approach to these challenges could be the use of zk-SNARK proofs that operate recursively and potentially build an variable sized universal outer zk-SNARK, or to use a fixed size construction based on padded binary trees. But such constructions would require many constraints for each layer in the tree and might therefore be impractical for real application scenarios.

Acknowledgements

We thank Jeffrey Burdges for the comments and discussions on an earlier draft of the paper. This work was supported in part by the European Commision through the Horizon Europe program under project number 101135475 (TALER), by the German Federal Ministry of Education under grant ConcreteContracts and the Swiss State Secretariat for Education, Research and Innovation (Project number: 101135475).

Jens Palsberg's GnuPG fingerprint is
`08DD BA88 6B5A E2D1 AD23 EF83 E98F 4DF0 D29F 094C`
(personally confirmed by the authors, without FoT).

References

[1] Dan Boneh, Ben Lynn, and Hovav Shacham. 2001. Short Signatures from the Weil Pairing. *Journal of Cryptology* 17 (2001), 297–319.

[2] W. Diffie and M. Hellman. 1976. New directions in cryptography. *IEEE Transactions on Information Theory* 22, 6 (1976), 644–654. https://doi.org/10.1109/TIT.1976.1055638

[3] John Douceur. 2002. The Sybil Attack. In *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS 2002)*.

[4] Álvaro García-Recuero. 2017. *Discouraging Abusive Behavior in Privacy-Preserving Decentralized Online Social Networks*. Ph. D. Dissertation. University of Rennes 1, France. https://tel.archives-ouvertes.fr/tel-01548658

[5] Jens Groth and Amit Sahai. 2008. Efficient Non-interactive Proof Systems for Bilinear Groups. In *Advances in Cryptology – EUROCRYPT 2008*, Nigel Smart (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 415–432.

[6] Kevin Georg Schrag. 2023. *Fog of Trust*. Bachelor's Thesis. Bern University of Applied Sciences.

[7] Dominic Tarr. 2015. Designing a Secret Handshake: Authenticated Key Exchange as a Capability System. https://dominictarr.github.io/secret-handshake-paper/shs.pdf

[8] Alexander Ulrich, Ralph Holz, Peter Hauck, and Georg Carle. 2011. Investigating the OpenPGP Web of Trust. In *16th European Symposium on Research in Computer Security (ESORICS)*.

[9] Alma Whitten and J. D. Tygar. 1999. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *8th USENIX Security Symposium (USENIX Security 99)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/8th-usenix-security-symposium/why-johnny-cant-encrypt-usability-evaluation-pgp-50

Received 2024-05-24; accepted 2024-07-12