AnoA: A Framework For Analyzing Anonymous Communication Protocols

Unified Definitions and Analyses of Anonymity Properties

Michael Backes[1,2] Aniket Kate[3] Praveen Manoharan[1]

Sebastian Meiser[1] Esfandiar Mohammadi[1]

[1] Saarland University, [2] MPI-SWS, [3] MMCI, Saarland University

{backes, manoharan, meiser, mohammadi}@cs.uni-saarland.de

aniket@mmci.uni-saarland.de

February 6, 2014

Abstract

Protecting individuals' privacy in online communications has become a challenge of paramount importance. To this end, anonymous communication (AC) protocols such as the widely used Tor network have been designed to provide anonymity to their participating users. While AC protocols have been the subject of several security and anonymity analyses in the last years, there still does not exist a framework for analyzing complex systems such as Tor and their different anonymity properties in a unified manner.

In this work we present AnoA: a generic framework for defining, analyzing, and quantifying anonymity properties for AC protocols. AnoA relies on a novel relaxation of the notion of (computational) differential privacy, and thereby enables a unified quantitative analysis of well-established anonymity properties, such as sender anonymity, sender unlinkability, and relationship anonymity. While an anonymity analysis in AnoA can be conducted in a purely information theoretical manner, we show that the protocol's anonymity properties established in AnoA carry over to secure cryptographic instantiations of the protocol. We exemplify the applicability of AnoA for analyzing real-life systems by conducting a thorough analysis of the anonymity properties provided by the Tor network against passive adversarys. Our analysis significantly improves on known anonymity results from the literature.

Note on this version Compared to the CSF version [BKM+13], this version enhances the AnoA framework by making the anonymity definitions as well as the corresponding privacy games *adaptive*. It also introduces the concept of *adversary classes* for restricting the capabilities of adversaries to model realistic attack scenarios.

Contents

1 Introduction

Protecting individuals' privacy in online communications has become a challenge of paramount importance. A wide variety of privacy enhancing technologies, comprising many different approaches, have been proposed to solve this problem. Privacy enhancing technologies, such as anonymous communication (AC) protocols, seek to protect users' privacy by anonymizing their communication over the Internet. Employing AC protocols has become increasingly popular over the last decade. This popularity is exemplified by the success of the Tor network [Tor03].

There has been a substantial amount of previous work [STRL00, DSCP02, SD02, Shm04, MVdV04, HO05, SW06, D06́, FJS07a, FJS07b, GTD$^+$08, APSVR11, FJS12] on analyzing the anonymity provided by various AC protocols such as dining cryptographers network (DC-net) [Cha88], Crowds [RR98], mix network (Mixnet) [Cha81], and onion routing (e.g., Tor) [RSG98]. However, most of the previous works only consider a single anonymity property for a particular AC protocol under a specific adversary scenario. Previous frameworks such as [HS04] only guarantee anonymity for a symbolic abstraction of the AC, not for its cryptographic realization. Moreover, while some existing works like [FJS12] consider an adversary with access to *a priori* probabilities for the behavior of users, there is still no work that is capable of dealing with an adversary that has arbitrary auxiliary information about user behavior.

Prior to this work, there is no framework that is both expressive enough to unify and compare relevant anonymity notions (such as sender anonymity, sender unlinkability, and relationship anonymity), and that is also well suited for analyzing complex cryptographic protocols.

1.1 Contributions

In this work, we make four contributions to the field of anonymity analysis.

As a first contribution, we present the novel anonymity analysis framework AnoA. In AnoA we define and analyze anonymity properties of AC protocols. Our anonymity definition is based on a novel generalization of differential privacy, a notion for privacy preserving computation that has been introduced by Dwork et al. [Dwo06, DMNS06]. The strength of differential privacy resides in a strong adversary that has maximal control over two adjacent settings that it has to distinguish. However, applying differential privacy to AC protocols seems impossible. While differential privacy does not allow for leakage of (potentially private) data, AC protocols inherently leak to the recipient the data that a sender sends to this recipient. We overcome this contradiction by generalizing the adjacency of settings between which an adversary has to distinguish. We introduce an explicit *adjacency function* α that characterizes whether two settings are considered adjacent or not. In contrast to previous work on anonymity properties, this generalization of differential privacy, which we name α-IND-CDP, is based on IND-CDP [MPRV09] and allows the formulation of anonymity properties in which the adversary can choose the messages—which results in a strong adversary—as long as the adjacent challenge inputs carry the same messages. Moreover, AnoA is compatible with simulation-based composability frameworks, such as UC [Can01], IITM [KT13], or RSIM [BPW07]. In particular, for all protocols that are securely abstracted by an ideal functionality [Wik04, CL05, DG09, KG10, BGKM12], our definitions allow an analysis of these protocols in a purely information theoretical manner.

As a second contribution, we formalize the well-established notions of sender anonymity, (sender) unlinkability, and relationship anonymity in our framework, by introducing appropriate adjacency functions. We discuss why our anonymity definitions accurately capture these notions, and show for sender anonymity and (sender) unlinkability that our definition is equivalent to the definitions from the literature. For relationship anonymity, we argue that previous formalizations captured recipient anonymity rather than relationship anonymity, and we discuss the accuracy of our formalization. Moreover, we show relations between our formalizations of sender anonymity, (sender) unlinkability, and relationship anonymity: sender anonymity implies both (sender) unlinkability and relationship anonymity, but is not implied by either of them.

As a third contribution, we extend our definitions by strengthening the adversary and improving the flexibility of our framework. We strengthen the adversary by introducing an adaptive challenger and adaptive anonymity notions for our privacy games. In many real-world scenarios an adversary has less control over individual protocol participants. We improve the flexibility of AnoA by introducing so-called *adversary classes* that allow such a fine-grained specification of the influence that the adversary has on a scenario. Even though, in general, adversary classes are not sequentially composable for adaptive adversaries, we identify a property that suffices for proving sequential composability.

Finally, as a fourth contribution, we apply our framework to the most successful AC protocol—Tor. We leverage previous results that securely abstract Tor as an ideal functionality (in the UC framework) [BGKM12]. Then, we illustrate that proving sender anonymity, sender unlinkability, and relationship anonymity against passive adversaries boils down to a combinatoric analysis, purely based on the number of corrupted nodes in the network. We further leverage our framework and analyze the effect of a known countermeasure for Tor's high sensitivity to compromised nodes: the entry guards mechanism. We discuss that, depending on the scenario, using entry guards can have positive or negative effect.

Outline of the Paper. In Section 2 we introduce the notation used throughout the paper. Section 3 presents our anonymity analysis framework AnoA and introduces the formalizations of sender anonymity, unlinkability, and relationship anonymity notions in the framework. Section 4 compares our anonymity notions with those from the literature as well as with each other. In Section 5 we extend the definitions with adaptively chosen inputs and with adversary classes. In Section 6, we demonstrate compatibility of AnoA with a simulation-based composability framework (in particular, the UC framework), and, in Section 7, we leverage our framework for proving of the anonymity guarantees the Tor network. In Section 8, we discuss and compare related work with our framework. Finally, we conclude and discuss some further interesting directions in Section 9.

2 Notation

Before we present AnoA, we briefly introduce some of the notation used throughout the paper. We differentiate between two different kinds of assignments: $a := b$ denotes a being assigned the value b, and $a \leftarrow \beta$ denotes that a value is drawn from the distribution β and a is assigned the outcome. In a similar fashion $i \overset{R}{\leftarrow} I$ denotes that i is drawn uniformly at random from the set I.

Probabilities are given over a probability space which is explicitly stated unless it is clear from context. For example $\Pr[b = 1 : b \overset{R}{\leftarrow} \{0, 1\}]$ denotes the probability of the event $b = 1$ in the probability space where b is chosen uniformly at random from the set $\{0, 1\}$.

Our security notion is based on interacting Turing Machines (TM). We use an oracle-notation for describing the interaction between an adversary and a challenger: $\mathcal{A}^{\mathcal{B}}$ denotes the interaction of TM $\mathcal{A}$ with TM $\mathcal{B}$ where $\mathcal{A}$ has oracle access to $\mathcal{B}$. Whenever $\mathcal{A}$ activates $\mathcal{B}$ again, $\mathcal{B}$ will continue its computation on the new input, using its previously stored state. $\mathcal{A}$ can then again activate $\mathcal{B}$ with another input value, and $\mathcal{B}$ will continue its computation with the new input, using its previously stored state. This interaction continues until $\mathcal{A}$ returns an output, which is considered the output of $\mathcal{A}^{\mathcal{B}}$.

In this paper we focus on computational security, i.e. all machines are computationally bounded. More formally, we consider *probabilistic, polynomial time* (PPT) TMs, which we denote with PPT whenever required.

3 The AnoA Framework

In this section, we present the AnoA framework and our formulations of sender anonymity, sender unlinkability, and relationship anonymity (Section 3.3). These formulations are based on a novel generalization of differential privacy that we describe in Section 3.2. Before we introduce this notion, we first describe the underlying protocol model. Using our protocol model, AC protocols are closely related to mechanisms that process databases, a fact that enables us to apply a more flexible form of *differential privacy.*

3.1 Protocol model

Anonymous communication (AC) protocols are distributed protocols that enable multiple users to anonymously communicate with multiple recipients. Formally, an AC protocol is an interactive Turing machine.[1] We associate a protocol with a user space $\mathcal{U}$, a recipient space $\mathcal{R}$ and an auxiliary information space Aux. Users' actions are modeled as an input to the protocol and represented in the form of an ordered *input table.* Each row in the input table contains a user $u \in \mathcal{U}$ that performs some action, combined with a list of possible recipients $r_i \in \mathcal{R}$ together with some auxiliary information aux. The meaning of aux depends on the nature of the AC protocol. Based on the AC protocol, auxiliary information can specify the content of a message that is sent to a recipient or may contain a symbolic description of user behavior. We can think of the rows in the input table as a list of successive input to the protocol.

Definition 1 (Input tables)**.** *An* input table *D of size t over a user space $\mathcal{U}$, a recipient space $\mathcal{R}$ and an auxiliary information space Aux is an ordered table $D = (d_1, d_2, \ldots, d_t)$ of tuples $d_j = (u_j, (r_{j_i}, \mathsf{aux}_{j_i})_{i=1}^{\ell})$, where $u_j \in \mathcal{U}, r_{j_i} \in \mathcal{R}$ and $\mathsf{aux}_{j_i} \in \mathsf{Aux}$.*

A typical adversary in an AC protocol can compromise a certain number of parties. We model such an adversary capability as static corruption: before the protocol execution starts $\mathcal{A}$ may decide which parties to compromise.

Our protocol model is generic enough to capture multi-party protocols in classical simulation-based composability frameworks, such as the UC [Can01], the IITM [KT13] or the RSIM [BPW07] framework. In particular, our protocol model comprises ideal functionalities, trusted machines that are used in simulation-based composability frameworks to define security. It is straightforward to construct a wrapper for such an ideal functionality of an AC protocol that translates input tables to the expected input of the functionality. We present such a wrapper for Tor in Section 7.

3.2 Generalized computational differential privacy

For privacy preserving computations the notion of *differential privacy* (DP) [Dwo06, DMNS06] is a standard for quantifying privacy. Informally, differential privacy of a mechanism guarantees that the mechanism does not leak any information about a single user–even to an adversary that has auxiliary information about the rest of the user base. It has also been generalized to protocols against

[1] We stress that using standard methods, a distributed protocol with several parties can be represented by one interactive Turing machine.

computationally bounded adversaries, which has led to the notion of computational differential privacy (CDP) [MPRV09]. In computational differential privacy two input tables are compared that are *adjacent* in the sense that they only differ in one row, called the *challenge row*. The definition basically states that no PPT adversary should be able to determine which of the two input tables was used.

For anonymity properties of AC protocols, such a notion of adjacency is too strong. One of the main objectives of an AC protocol is communication: delivering the sender's message to the recipient. However, if these messages carry information about the sender, a curious recipient can determine the sender (see the following example).

<u>*Example 1: Privacy.*</u> *Consider an adversary $\mathcal{A}$ against the "computational differential privacy" game with an AC protocol. Assume the adversary owns a recipient* evilserver.com*, that forwards all messages it receives to $\mathcal{A}$. Initially, $\mathcal{A}$ sends input tables D_0, D_1 to the IND-CDP challenger that are equal in all rows but one: In this distinguishing row of D_0 the party Alice sends the message "I am Alice!" to evilserver.com and in D_1, the party Bob sends the message "I am Bob!" to evilserver.com. The tables are adjacent in the sense of computational differential privacy (they differ in exactly one row). However, no matter how well the identities of recipients are hidden by the protocol, the adversary can recognize them by their messages and thus will win the game with probability* 1. $\diamond$

Our generalization of CDP allows more fine-grained notions of adjacency; e.g., adjacency for sender anonymity means that the two tables only differ in one row, and in this row only the user that sends the messages is different. In general, we say that an adjacency function α is a randomized function that expects two input tables (D_0, D_1) and either outputs two input tables (D_0', D_1') or a distinguished error symbol $\bot$. Allowing the adjacency function α to also modify the input tables is useful for shuffling rows, which we need for defining relationship anonymity (see Definition 6).

CDP, like the original notion of differential privacy, only considers trusted mechanisms. In contrast to those incorruptible, monolithic mechanisms we consider arbitrary protocols, and thus even further generalize and strengthen CDP: we grant the adversary the possibility of compromising parties in the mechanism in order to accurately model the adversary.

For analyzing a protocol $\mathcal{P}$, we define a challenger $\textsc{Ch}(\mathcal{P}, \alpha, b)$ that expects two input tables D_0, D_1 from a PPT adversary $\mathcal{A}$. The challenger $\textsc{Ch}$ calls the adjacency function α on (D_0, D_1). If α returns $\bot$ the challenger halts. Otherwise, upon receiving two (possibly modified) tables D_0', D_1', $\textsc{Ch}$ chooses D_b', depending on its input bit b, and successively feeds one row after the other to the protocol $\mathcal{P}$.[2] We assume that the protocol upon an input $(u, (r_i, \mathsf{aux}_i)_{i=1}^{\ell})$, sends $(r_i, \mathsf{aux}_i)_{i=1}^{\ell}$ as input to party u. In detail, upon a message $(\mathsf{input}, D_0, D_1)$ sent by $\mathcal{A}$, $\textsc{Ch}(\mathcal{P}, \alpha, b)$ computes $(D_0', D_1') \leftarrow \alpha(D_0, D_1)$. If $(D_0', D_1') \neq \bot$, $\textsc{Ch}$ runs $\mathcal{P}$ with the input table D_b' and forwards all messages that are sent from $\mathcal{P}$ to $\mathcal{A}$ and all messages that are sent from $\mathcal{A}$ to $\mathcal{P}$. At any point, the adversary may output his decision b^*.

Our definition depends on two parameters: ϵ and δ. As in the definition of differential privacy, ϵ quantifies the degree of anonymity (see Example 3). The anonymity of commonly employed AC protocols also break down if certain distinguishing events happen, e.g., when an entry guard of a Tor user is compromised. Similar to CDP, the probability that such a distinguishing event happens is quantified by the parameter δ. However, in contrast to CDP, this δ is typically non-negligible and depends on the degree of corruption in the AC network. As a next step, we formally define (ε, δ)-α-IND-CDP.

[2]In contrast to IND-CDP, we only consider PPT-computable tables.

```
Upon message(input, $D_0, D_1$) (only once)
  compute $(D_0', D_1') \leftarrow \alpha(D_0, D_1)$
  if $(D_0', D_1') \neq \bot$ then
    run $\mathcal{P}$ on the input table $D_b'$ and forward all messages that are sent by $\mathcal{P}$ to the adversary
    $\mathcal{A}$ and send all messages by the adversary to $\mathcal{P}$.
```

Figure 1: The challenger $\textsc{Ch}(\mathcal{P}, \alpha, b)$ for the adjacency function α

Definition 2 ((ε, δ)-α-IND-CDP)**.** *Let* Ch *be the challenger from Figure 1. The protocol $\mathcal{P}$ is* (ε, δ)-α-IND-CDP for α, *where $\varepsilon \geq 0$ and $0 \leq \delta \leq 1$, if for all* PPT-*adversaries $\mathcal{A}$:*

$$\Pr[b = 0 : b \leftarrow \mathcal{A}^{\textsc{Ch}(\mathcal{P},\alpha,0)}] \\ \leq e^{\varepsilon} \cdot \Pr[b = 0 : b \leftarrow \mathcal{A}^{\textsc{Ch}(\mathcal{P},\alpha,1)}] + \delta$$

A note on the adversary model. While our adversary initially constructs the two input tables in their entirety, our model does not allow the adversary to adaptively react to the information that it observes by changing the behaviors of users. This is in line with previous work, which also assumes that the user behavior is fixed before the protocol is executed [FJS07a, FJS12].

As a next step towards defining our anonymity properties, we formally introduce the notion of challenge rows. Recall that challenge rows are the rows that differ in the two input tables.

Definition 3 (Challenge rows)**.** *Given two input tables $A = (a_1, a_2, \ldots, a_t)$ and $B = (b_1, b_2, \ldots, b_t)$ of the same size, we refer to all rows $a_i \neq b_i$ with $i \in \{1, \ldots, t\}$ as* challenge rows*. If the input tables are of different sizes, there are no challenge rows. We denote the challenge rows of D as $\mathsf{CR}(D)$.*

3.3 Anonymity properties

In this section, we present our (ε, δ)-α-IND-CDP based anonymity definitions in which the adversary is allowed to choose the entire communication except for the challenge rows, for which he can specify two possibilities. First, we define sender anonymity, which states that a malicious recipient cannot decide, for two candidates, to whom he is talking even in the presence of virtually arbitrary auxiliary information. Second, we define user unlinkability, which states that a malicious recipient cannot decide whether it is communicating with one user or with two different users, in particular even if he chooses the two possible rows. Third, we define relationship anonymity, which states that an adversary (that potentially controls some protocol parties) cannot relate sender and recipient in a communication.

Our definitions are parametrized by ε and δ. We stress that all our definitions are necessarily quantitative. Due to the adversary's capability to compromise parts of the communication network and the protocol parties, achieving overwhelming anonymity guarantees (i.e., for a negligible δ) for non-trivial (and useful) AC protocols is infeasible.

3.3.1 Sender anonymity

Sender anonymity requires that the identity of the sender is hidden among the set of all possible users. In contrast to other notions from the literature, we require that the adversary is not able to decide which of two *self-chosen* users have been communicating. Our notion is stronger than the

```
$\alpha_{\mathrm{SA}}(D_0, D_1)$
  if $||D_0|| \neq ||D_1||$ then
    output $\bot$
  if $\mathsf{CR}(D_0) = ((u_0, R)) \wedge \mathsf{CR}(D_1) = ((u_1, R))$ then
    output $(D_0, D_1)$
  else
    output $\bot$
```

Figure 2: The adjacency function α_{SA} for sender anonymity.

```
$\alpha_{\mathrm{UL}}(D_0, D_1)$
  if $||D_0|| \neq ||D_1||$ then
    output $\bot$
  if $\mathsf{CR}(D_0) = ((u_0, R_u), (u_0, R_v)) =: (c_{0,u}, c_{0,v})$
   $\wedge \mathsf{CR}(D_1) = ((u_1, R_u), (u_1, R_v)) =: (c_{1,u}, c_{1,v})$
  then
    $x \overset{R}{\leftarrow} \{0, 1\}$, $y \overset{R}{\leftarrow} \{u, v\}$
    Replace $c_{x,y}$ with $c_{(1-x),y}$ in $D_x$
    output $(D_x, D_{1-x})$
  else
    output $\bot$
```

Figure 3: The adjacency function α_{UL} for sender unlinkability.

usual notion, and in Section 4 we exactly quantify the gap between our notion and the notion from the literature. Moreover, we show that the Tor network satisfies this strong notion, as long as the user in question did not choose a compromised path (see Section 7).

We formalize our notion of sender anonymity with the definition of an adjacency function α_{SA} as depicted in Figure 2. Basically, α_{SA} merely checks whether in the challenge rows everything except for the user is the same.

Definition 4 (Sender anonymity)**.** *A protocol $\mathcal{P}$ provides* (ε, δ)-sender anonymity *if it is* (ε, δ)-α-IND-CDP *for α_{SA} as defined in Figure 2.*

Example 2: Sender anonymity. *The adversary $\mathcal{A}$ decides that he wants to use users Alice and Bob in the sender anonymity game. It sends input tables D_0, D_1 such that in the challenge row of D_0 Alice sends a message m^* of $\mathcal{A}$'s choice to a (probably corrupted) recipient, e.g. evilserver.com, and in D_1, instead of Alice, Bob sends the same message m^* to the same recipient evilserver.com. The adjacency function α_{SA} makes sure that only one challenge row exists and that the messages and the recipients are equal. If so, it outputs D_0, D_1 and if not it outputs $\bot$.* $\diamond$

Notice that analogously recipient anonymity (α_{RA}) can be defined: the adjacency function then checks that the challenge rows only differ in one *recipient*.

```
$\alpha_{\mathrm{Rel}}(D_0, D_1)$
  if $||D_0|| \neq ||D_1||$ then
    output $\perp$
  if $\mathsf{CR}(D_0) = ((u_0, R_u)) \wedge \mathsf{CR}(D_1) = ((u_1, R_v))$ then
    $x \overset{R}{\leftarrow} \{0,1\}$, $y \overset{R}{\leftarrow} \{0,1\}$
    if x=1 then
      Set $\mathsf{CR}(D_0)$ to $((u_1, R_v))$
    if y=1 then
      Set $\mathsf{CR}(D_1)$ to $((u_0, R_v))$
    else
      Set $\mathsf{CR}(D_1)$ to $((u_1, R_u))$
    output $(D_0, D_1)$
  else
    output $\perp$
```

Figure 4: The adjacency function α_{Rel} for relationship anonymity.

3.3.2 The value of ε

In Section 7, we analyze the widely used AC protocol Tor. We show that if every node is uniformly selected then Tor satisfies sender anonymity with $\varepsilon = 0$. If the nodes are selected using preferences, e.g., in order to improve throughput and latency, ε and δ may increase.[3]

Recall that the value δ describes the probability of a distinguishing event, and if this distinguishing event occurs, anonymity is broken. In the sender anonymity game for Tor this event occurs if the entry guard of the user's circuit is compromised. If a user has a preference for the first node, the adversary can compromise the most likely node. Thus, a preference for the first node in a circuit increases the probability for the distinguishing event (δ). However, if there is a preference for the second node in a circuit, corrupting this node does not lead to the distinguishing event but can still increase the adversary's success probability by increasing ε. Consider the following example.

Example 3: The value of ε. *Assume that the probability that Alice chooses a specific node N as second node is $\frac{1}{40}$ and the probability that Bob uses N as second node is $\frac{3}{40}$. Further assume that for all other nodes and users the probabilities are uniformly distributed. Suppose the adversary $\mathcal{A}$ corrupts N. If $\mathcal{A}$ observes communication over the node N, the probability that this communication originates from Bob is 3 times the probability that it originates from Alice. Thus, with such preferences Tor only satisfies sender anonymity with $\varepsilon = \ln 3$.* $\diamond$

3.3.3 Sender unlinkability

A protocol satisfies *sender unlinkability*, if for any two actions, the adversary cannot determine whether these actions are executed by the same user [PH10]. We require that the adversary does not know whether two challenge messages come from the same user or from different users. We formalize this intuition by letting the adversary send two input tables with two challenge rows, respectively. Each input table D_x carries challenge rows in which a user u_x sends a message to two recipients R_u, R_v. We use the shuffling abilities of the adjacency function α_{UL} as defined in Figure 3, which

[3]Previous work discusses the influence of node selection preferences on Tor's anonymity guarantees, e.g., [AYM12].

makes sure that D'_0 will contain the same user in both challenge rows, whereas D'_1 will contain both users. As before, we say a protocol $\mathcal{P}$ fulfills sender unlinkability, if no adversary $\mathcal{A}$ can sufficiently distinguish $\textsc{Ch}(\mathcal{P}, \alpha_{\mathrm{UL}}, 0)$ and $\textsc{Ch}(\mathcal{P}, \alpha_{\mathrm{UL}}, 1)$. This leads to the following concise definition.

Definition 5 (Sender unlinkability)**.** *A protocol $\mathcal{P}$ provides (ε, δ)-*sender unlinkability *if it is (ε, δ)-*α-IND-CDP *for α_{UL} as defined in Figure 3.*

Example 4: Sender unlinkability. The adversary $\mathcal{A}$ decides that he wants to use users Alice and Bob in the unlinkability game. He sends input tables D_0, D_1 such that in the challenge rows of D_0 Alice sends two messages to two recipients and in D_1, Bob sends the same two messages to the same recipients. Although initially "the same user sends the messages" would be true for both input tables, the adjacency function α_{UL} changes the challenge rows in the two input tables D_0, D_1. In the transformed input tables D'_0, D'_1, only one of the users (either Alice or Bob) will send both messages in D'_0 , whereas one message will be sent by Alice and the other by Bob in D'_1. $\diamond$

3.3.4 Relationship anonymity

$\mathcal{P}$ satisfies *relationship anonymity*, if for any action, the adversary cannot determine sender and recipient of this action at the same time [PH10]. We model this property by letting the adjacency α_{Rel} check whether it received an input of two input tables with a single challenge row. We let the adjacency function α_{Rel} shuffle the recipients and sender such that we obtain the four possible combinations of user and recipient. If the initial challenge rows are (u_0, R_0) and (u_1, R_1), α_{Rel} will make sure that in D'_0 one of those initial rows is used, where in D'_1 one of the rows (u_0, R_1) or (u_1, R_0) is used.

We say that $\mathcal{P}$ fulfills relationship anonymity, if no adversary can sufficiently distinguish $\textsc{Ch}(\mathcal{P}, \alpha_{\mathrm{Rel}}, 0)$ and $\textsc{Ch}(\mathcal{P}, \alpha_{\mathrm{Rel}}, 1)$.

Definition 6 (relationship anonymity)**.** *A protocol $\mathcal{P}$ provides (ε, δ)-*relationship anonymity *if it is (ε, δ)-*α-IND-CDP *for α_{Rel} as defined in Figure 4.*

Example 5: Relationship anonymity. The adversary $\mathcal{A}$ decides that he wants to use users Alice and Bob and the recipients Charly and Eve in the relationship anonymity game. He wins the game if he can distinguish between the scenario "0" where Alice sends m_1 to Charly or Bob sends m_2 to Eve and the scenario "1" where Alice sends m_2 to Eve or Bob sends m_1 to Charly. Only one of those four possible input lines will be fed to the protocol.

$\mathcal{A}$ sends input tables D_0, D_1 such that in the challenge row of D_0 Alice sends m_1 to Charly and in D_1, Bob sends m_2 to Eve. Although initially 'scenario 0" would be true for both input tables, the adjacency function α_{Rel} changes the challenge rows in the two input tables D_0, D_1 such that in D'_0 one of the two possible inputs for scenario "0" will be present (either Alice talks to Charly or Bob talks to Eve) and in D'_1 one of the two possible inputs for scenario "1" will be present (either Bob talks to Charly or Alice talks to Eve). $\diamond$

4 Studying our Anonymity Definitions

In this section, we show that our anonymity definitions indeed capture the anonymity notions from the literature. We compare our notions to definitions that are directly derived from informal descriptions in the seminal work by Pfitzmann and Hansen [PH10]. Lastly, we investigate the relation between our own anonymity definitions.

```
**Upon message** (input, $D$) **(only once)**
  **if** $\exists!$ challenge row in $D$ **then**
    Place user $u$ in the challenge row of $D$
    run $\mathcal{P}$ on the input table $D$ and forward all messages to $\mathcal{A}$
```

Figure 5: The challenger $\text{SACH}(\mathcal{P}, u)$

4.1 Sender anonymity

The notion of sender anonymity is introduced in [PH10] as follows:

> Anonymity of a subject from an adversary's perspective means that the adversary cannot sufficiently identify the subject within a set of subjects, the anonymity set.

From this description, we formalize their notion of sender anonymity. For any message m and adversary $\mathcal{A}$, any user in the user space is equally likely to be the sender of m.

Definition 7 (δ-sender anonymity)**.** *A protocol $\mathcal{P}$ with user space $\mathcal{U}$ of size N has* δ-sender anonymity *if for all* PPT-*adversaries $\mathcal{A}$*

$$Pr\left[u^* = u : u^* \leftarrow \mathcal{A}^{\text{SACH}(\mathcal{P},u)}, u \overset{R}{\leftarrow} \mathcal{U}\right] \leq \frac{1}{N} + \delta,$$

where the challenger SACH *as defined as in Figure 5.*

Note that SACH slightly differs from the challenger $\text{CH}(\mathcal{P}, \alpha, b)$ in Figure 1: It does not require two, but just one input table in which a single row misses its sender. We call this row the challenge row.

This definition is quite different from our interpretation with adjacency functions. While α_{SA} requires $\mathcal{A}$ to simply distinguish between two possible outcomes, Definition 7 requires $\mathcal{A}$ to correctly guess the right user. Naturally, α_{SA} is stronger than the definition above. Indeed, we can quantify the gap between the definitions: Lemma 8 states that an AC protocol satisfies $(0, \delta)$-α_{SA} implies that this AC also has δ-sender anonymity. The proofs for these lemmas can be found in Appendix A.2.In this section, we only present the proof outlines.

Lemma 8 (sender anonymity)**.** *For all protocols $\mathcal{P}$ over a (finite) user space $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for α_{SA}, $\mathcal{P}$ also has δ-sender anonymity as in Definition 7.*

Proof outline. We show the contraposition of the lemma: an adversary $\mathcal{A}$ that breaks sender anonymity, can be used to break α-IND-CDP for α_{SA}. We construct an adversary B against α-IND-CDP for α_{SA} by choosing the senders of the challenge rows at random, running $\mathcal{A}$ on the resulting game, and outputting the same as $\mathcal{A}$. For $\mathcal{A}$ the resulting view is the same as in the sender anonymity game; hence, B has the same success probability in the α-IND-CDP game as $\mathcal{A}$ in the sender anonymity game. □

In the converse direction, we lose a factor of $\frac{1}{N}$ in the reduction, where N is the size of the user space. If an AC protocol $\mathcal{P}$ provides δ-sender anonymity, we only get $(0, \delta \cdot N)$-α_{SA} for $\mathcal{P}$.

Lemma 9. *For all protocols $\mathcal{P}$ over a (finite) user space $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has δ-sender anonymity as in Definition 7, $\mathcal{P}$ also has* $(0, \delta \cdot N)$-α-IND-CDP *for α_{SA}.*

```
**Upon message** $(\mathsf{input}, D)$ **(only once)**
  **if** exactly 2 rows in $D$ are missing the user **then**
    $u_0 \overset{R}{\leftarrow} \mathcal{U}, u_1 \overset{R}{\leftarrow} \mathcal{U} \setminus \{u_0\}$
    **if** $b = 0$ **then**
      Place $u_0$ in both rows.
    **else**
      Place $u_0$ in the first and $u_1$ in the second row.
    run $\mathcal{P}$ on input table $D$ and forward all messages to $\mathcal{A}$
```

Figure 6: The challenger ULCh$(\mathcal{P}, b)$

Proof outline. We show the contraposition of the lemma: an adversary $\mathcal{A}$ that breaks α-IND-CDP for α_{SA}, can be used to break sender anonymity. We construct an adversary B against sender anonymity by running $\mathcal{A}$ on the sender anonymity game and outputting the same as $\mathcal{A}$. If the wishes of $\mathcal{A}$ for the challenge senders coincide with the sender that the challenger chose at random, the resulting view is the same as in the α-IND-CDP game for α_{SA}; hence, B has a success probability of δ/N in the sender anonymity game if $\mathcal{A}$ has a success probability of δ in the α-IND-CDP game for α_{SA}. □

4.2 Unlinkability

The notion of unlinkability is defined in [PH10] as follows:

> Unlinkability of two or more items of interest (IOIs, e.g., subjects, messages, actions, ...) from an adversary's perspective means that within the system (comprising these and possibly other items), the adversary cannot sufficiently distinguish whether these IOIs are related or not.

Again, we formalize this in our model. We leave the choice of potential other items in the system completely under adversary control. Also, the adversary controls the "items of interest" (IOI) by choosing when and for which recipient/messages he wants to try to link the IOIs. Formally, we define a game between a challenger ULCh and an adversary $\mathcal{A}$ as follows: First, $\mathcal{A}$ chooses a input table D, but leaves the place for the users in two rows blank. The challenger then either places one (random) user in both rows or two different (random) users in each and then runs the protocol and forwards all output to $\mathcal{A}$. The adversary wins the game if he is able to distinguish whether the same user was placed in the rows (i.e. the IOIs are linked) or not.

Definition 10 (δ-sender unlinkability)**.** *A protocol $\mathcal{P}$ with user space $\mathcal{U}$ has* δ-sender unlinkability *if for all* PPT-*adversaries $\mathcal{A}$*

$$\left|Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\text{ULCh}(\mathcal{P},0)}\right] - Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\text{ULCh}(\mathcal{P},1)}\right]\right| \leq \delta$$

where the challenger ULCh *is as defined in Figure 6.*

We show that our notion of sender unlinkability using the adjacency function α_{UL} is much stronger than the δ-sender unlinkability Definition 10: $(0, \delta)$-α_{UL} for an AC protocol directly implies δ-sender unlinkability; we do not lose any anonymity.

Lemma 11 (sender unlinkability)**.** *For all protocols $\mathcal{P}$ over a user space $\mathcal{U}$ it holds that if $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for α_{UL}, $\mathcal{P}$ also has δ-sender unlinkability as in Definition 10.*

Proof outline. We show the contraposition of the lemma: an adversary $\mathcal{A}$ that breaks sender unlinkability, can be used to break α-IND-CDP for α_{UL}. We construct an adversary B against α-IND-CDP for α_{UL} by choosing the senders of the challenge rows at random, running $\mathcal{A}$ on the resulting game, and outputting the same as $\mathcal{A}$. For $\mathcal{A}$ the resulting view is the same as in the sender unlinkability game; hence, B has the same success probability in the α-IND-CDP game for α_{UL} as $\mathcal{A}$ in the sender unlinkability game. □

For the converse direction, however, we lose a factor of roughly N^2 for our δ. Similar to above, proving that a protocol provides δ-sender unlinkability only implies that the protocol is $(0, \delta \cdot N(N-1))$-α-IND-CDP for α_{UL}.

Lemma 12 (sender unlinkability)**.** *For all protocols $\mathcal{P}$ over a user space $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has δ-sender unlinkability as in Definition 10, $\mathcal{P}$ also has* $(0, \delta \cdot N(N-1))$-α-IND-CDP *for α_{UL}.*

Proof outline. We show the contraposition of the lemma: an adversary $\mathcal{A}$ that breaks α-IND-CDP for α_{UL}, can be used to break sender unlinkability. We construct an adversary B against sender unlinkability by running $\mathcal{A}$ on the sender unlinkability game and outputting the same as $\mathcal{A}$. If the senders from the challenge from of $\mathcal{A}$ coincide with the senders that the challenger chose at random, the resulting view is the same as in the α-IND-CDP game for α_{UL}; hence, B has a success probability of $\delta/N(N-1)$ in the sender unlinkability game if $\mathcal{A}$ has a success probability of δ in the α-IND-CDP game for α_{UL}. □

Again, proofs can be found in Appendix A.2.

4.3 Relationship anonymity

While for sender anonymity and sender unlinkability our notions coincide with the definitions used in the literature, we find that for relationship anonymity, many of the interpretations from the literature are not accurate. In their Mixnet analysis, Shmatikov and Wang [SW06] define relationship anonymity as 'hiding the fact that party A is communicating with party B'. Feigenbaum et al. [FJS07b] also take the same position in their analysis of the Tor network. However, in the presence of such a powerful adversary, as considered in this work, these previous notions collapse to recipient anonymity since they assume knowledge of the potential senders of some message.

We consider the notion of relationship anonymity as defined in [PH10]: the anonymity set for a message m comprises the tuples of possible senders and recipients; the adversary wins by determining which tuple belongs to m. However, adopting this notion directly is not possible: an adversary that gains partial information (e.g. if he breaks sender anonymity), also breaks the relationship anonymity game, all sender-recipient pairs are no longer equally likely. Therefore we think that approach via the adjacency function gives a better definition of relationship anonymity because the adversary needs to uncover both sender and recipient in order to break anonymity.

4.4 Relations between anonymity notions

Having justified the accuracy of our anonymity notions, we proceed by presenting the relations between our notions of anonymity. AnoA allows us to formally argue about these relations. Figure 7

$$\begin{tikzcd}
\alpha_{\mathrm{UL}} \arrow[r, Rightarrow, bend right, "/" marking] & \alpha_{\mathrm{SA}} \arrow[l, Rightarrow, bend right] \arrow[r, Rightarrow, bend left] & \alpha_{\mathrm{Rel}} \arrow[l, Rightarrow, bend left, "/" marking]
\end{tikzcd}$$

Figure 7: The relations between our anonymity definitions

illustrates the implications we get based on our definitions using adjacency functions. In this section, we discuss these relations. The proofs can be found in Appendix A.2.

Lemma 13 (Sender anonymity implies relationship anonymity.). *If a protocol $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for* α_{SA}*, is also has* $(0, \delta)$-α-IND-CDP *for* α_{Rel}.

Proof outline. Relationship anonymity requires an adversary to acquire information about both sender and recipient. If a protocol has sender anonymity, this is not possible. Hence, sender anonymity implies relationship anonymity. □

Similarly, recipient anonymity implies relationship anonymity.

Lemma 14 (Sender anonymity implies sender unlinkability). *If a protocol $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for* α_{SA}*, $\mathcal{P}$ also has* $(0, \delta)$-α-IND-CDP *for* α_{UL}.

Proof outline. Our strong adversary can determine the behavior of all users; in other words, the adversary can choose the scenario in which it wants to deanonymize the parties in question. Thus, the adversary can choose the payload messages that are not in the challenge row such that these payload messages leak the identity of their sender. Hence, if an adversary can link the message in the challenge row to another message, it can determine the sender. Thus, sender anonymity implies sender unlinkability. □

A protocol could leak the sender of a single message. Such a message does not necessarily help an adversary in figuring out whether another message has been sent by the same sender, but breaks sender anonymity.

Lemma 15 (Sender unlinkability does not imply sender anonymity). *If a protocol $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for* α_{UL}*, $\mathcal{P}$ does not necessarily have* $(0, \delta')$-α-IND-CDP *for* α_{SA} *for any* $\delta' < 1$.

Proof outline. We consider a protocol Π that satisfies sender anonymity. We, moreover, consider the modified protocol Π' that leaks the sender of a single message. Since by Lemma 14 Π satisfies unlinkability, we conclude that the modified protocol Π' satisfies sender unlinkability: a single message does not help the adversary in breaking sender unlinkability. However, Π' leaks in one message the identity of the sender in plain, hence does not satisfy sender anonymity. □

Relationship anonymity does not imply sender anonymity in general: for example, a protocol may reveal information about senders of the messages, but not about recipients or message contents.

Lemma 16 (Relationship anonymity does not imply sender anonymity). *If a protocol $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for* α_{Rel}*, $\mathcal{P}$ does not necessarily have* $(0, \delta')$-α-IND-CDP *for* α_{SA} *for any* $\delta' < 1$.

Proof outline. We consider a protocol Π that satisfies sender anonymity. We, moreover, consider the modified protocol Π' that for each message leaks the sender. Since by Lemma 13 Π satisfies unlinkability, we conclude that the modified protocol Π' satisfies relationship anonymity: the sender alone does not help the adversary in breaking relationship anonymity. However, Π' leaks the identity of the sender in plain, hence does not satisfy sender anonymity. $\square$

This concludes the formal definition of our framework.

5 Adaptive AnoA

The definitions presented so far force the adversary to fix all inputs at the beginning, i.e., they only allow static inputs. In this section, we generalize AnoA for adaptive inputs and adjust the anonymity games accordingly. To model fine-grained interactions, we allow the adversary to send individual messages (instead of sessions), such that messages within one session can depend on the observations of the adversary. Since many AC protocols (such as Tor) guarantee anonymity on a session basis (cf. Section 7.2), we present both anonymity notions for individual messages and for whole sessions.

We improve the flexibility of the framework by formalizing classes of adversaries that only have limited influence on the inputs of the users. It turns out that for technical reasons such *adversary classes* are in general not sequentially composable. In Section 5.3, we characterize under which conditions an adversary class is sequentially composable.

5.1 Defining adaptive adversaries

The AnoA challenger from Definition 2 requires the adversary to initially decide on all inputs for all users, i.e., the challenger enforces static inputs. Against an adversary that controls all servers interactive scenarios, such as a chatting user or transactions between two users, can be expressed with static inputs.[4] However, as soon as we consider adversaries that do not control all servers, static-input adversaries do not allow to model arbitrary interactive scenarios because all responses have to be fixed in the initial input table.

We present an extension of AnoA that allows the adversary to adaptively choose its inputs. The adversary can, for every message, choose sender, message and recipient. The adjacency functions are not applied to input tables anymore, but to individual challenges, which correspond to the challenge rows from the previously used input tables. In contrast to our previous definition, these generalized adjacency functions allow us to additionally formulate settings with streams of messages that consist of more (or less) than one circuit. More precisely, we modify the challenger from Section 3.2 as follows.

Instead of databases, the challenger now receives single user actions $r = (\mathcal{S}, \mathcal{R}, m)$, corresponding to individual *rows*, that are processed by the protocol in having a user $\mathcal{S}$ send the message m to a recipient $\mathcal{R}$ and challenge actions (r_0, r_1) that correspond to *challenge rows.* The single user actions are forwarded to the environment immediately, whereas the adjacency function is invoked on the challenge actions and the resulting action $r^* \leftarrow \alpha(r_0, r_1, b)$ is sent to the environment (unless the adjacency function outputs an error symbol $\perp$).

To model more complex anonymity notions, such as anonymity for a *session*, the adjacency function is allowed to keep state for each challenge. To simplify the notation of our adjacency

[4]The adversary would internally simulate the communication and replay it by composing the input table and choosing the server responses accordingly.

functions, they get the bit b from the challenger as an additional input. Consequently, the adjacency functions can model the anonymity challenges on their own and simply output a row r^* that is sent to the challenger. The adaptive challenger for AnoA (for n challenges) follows these simple rules:

- Run the protocol on all messages (input, m) from the adversary without altering them.
- Only allow for n challenges. This is done by restricting the set of possible *challenge tags* Ψ, which are used by the adversary to distinguish the challenges.[5]
- For every challenge that is not in the state over, apply the adjacency function with its correct state – fresh for newly started challenges, some other state st_Ψ if the challenge is already active – and simulate the protocol on the output of the adjacency function.
- Make sure that the sessions that are used for challenges are not accessible via the challenger. This is done by choosing and storing session IDs sid_{real} for every challenge separately as follows. Whenever a message with a new session ID S is to be sent to the protocol, randomly pick a fresh session ID sid_{real} that is sent instead, and store sid_{real} together with S and the challenge tag Ψ (if it is a challenge) or zero (if it is not a challenge).

The full description of the session challenger is available in Figure 8. By definition, the sessions created by challenge messages are isolated from sessions created by input messages. Thus, an adversary cannot use the challenger to highjack sessions. The interface of the adaptive adjacency functions now is

$$\alpha(r_0 = (\mathcal{S}_0, \mathcal{R}_0, m_0, S_0), r_1 = (\mathcal{S}_1, \mathcal{R}_1, m_1, S_1), b)$$

We define adaptive α-IND-CDP as follows, where n is the number of challenges that an adversary is allowed to start.

Definition 17 ((n, ϵ, δ)-α-IND-CDP)**.** *A protocol $\mathcal{P}$ is (n, ϵ, δ)-α-IND-CDP for a class of adversaries A, with $\varepsilon \geq 0$ and $0 \leq \delta \leq 1$, if for all PPT machines $\mathcal{A}$,*

$$\Pr\left[0 = \langle \mathrm{A}(\mathcal{A}(n)) || \mathrm{C{\small H}}(\mathcal{P}, \alpha, n, 0)\rangle\right] \leq e^{n\varepsilon} \Pr\left[0 = \langle \mathrm{A}(\mathcal{A}(n)) || \mathrm{C{\small H}}(\mathcal{P}, \alpha, n, 1)\rangle\right] + e^{n\varepsilon} n\delta$$

where the challenger $\mathrm{C{\small H}}$ *is defined in Figure 8.*

For the adaptive notions the structure of the rows is slightly different. In Section 3.2, every row consists of a sequence of user inputs that is processed by the protocol, e.g., our Tor interface translates each row to a stream of messages that use one circuit. In an adaptive game the adversary, depending on its observations, might want to modify the messages within one session. Thus, every user action only consists of one single message that is sent to a single recipient, so messages within one session can depend on the observations of the adversary.

We explicitly allow to analyze the difference between anonymity for individual messages and anonymity on a session level. To this end we formulate new variants of the anonymity notions both with and without sessions.

[5] Several challenges that are within different stages might interleave at the same time; e.g., the adversary might first start an unlinkability challenge (with tag Ψ_1). Before this challenge is over, the adversary might start another challenge (with another tag Ψ_2) or continue the first one (with Ψ_1).

```
**Adaptive** ANOA **Challenger** $\mathbf{Ch}(\mathcal{P}, \alpha, n, b)$
**Upon message** $(\mathsf{input}, r = (\mathcal{S}, \mathcal{R}, m, S))$
  RunProtocol$(r, 0)$

**Upon message** $(\mathsf{challenge}, r_0, r_1, \Psi)$
  **if** $\Psi \notin \{1, \ldots, n\}$ **then**
    Abort.
  **else**
    **if** $\Psi \in \mathbb{T}$ **then**
      Retrieve $st := st_\Psi$
      **if** $st = \mathsf{over}$ **then**
        Abort.
    **else**
      $st := \mathsf{fresh}$
      Add $\Psi$ to $\mathbb{T}$
    Compute $(r^*, st_\Psi) \leftarrow \alpha(st, r_0, r_1, b, \Psi)$
    RunProtocol$(r, \Psi)$

**RunProtocol**$(r = (\mathcal{S}, \mathcal{R}, m, S), \Psi)$
  **if** $\neg\exists y$ such that $(S, y, \Psi) \in S$ **then**
    Let $\mathrm{sid}_{\mathrm{real}} \leftarrow \{0,1\}^k$
    Store $(S, \mathrm{sid}_{\mathrm{real}}, \Psi)$ in $S$.
  **else**
    $\mathrm{sid}_{\mathrm{real}} := y$
  Run $\mathcal{P}$ on $r = (\mathcal{S}, \mathcal{R}, m, \mathrm{sid}_{\mathrm{real}})$ and forward all messages that are sent by $\mathcal{P}$ to the adversary
  $\mathcal{A}$ and send all messages by the adversary to $\mathcal{P}$.
```

Figure 8: Adaptive ANOA Challenger

5.1.1 Adaptive anonymity notions

The anonymity notions we describe here are almost identical to their non-adaptive counterparts from Section 3.3. However, an adaptive adversary is able to adapt its strategy in constructing user input depending on the protocol output. However, we model sessions explicitly. Usually, such sessions are started by and linked to individual senders, but not to recipients. Thus, we model the anonymity notions in a way that allows the adversary to use different messages and recipients for each individual challenge message, but not to exchange the senders (within one challenge). If a protocol creates sessions for recipients, the respective anonymity notions can easily be derived by following our examples in this section.

Session sender anonymity. For session sender anonymity, we make sure that all messages that belong to the same challenge have both the same sender and the same session ID. We do not restrict the number of messages per challenge (but the protocol might restrict the number of messages per session).

```
$\alpha_{\mathrm{SSA}}(st, r_0 = (\mathcal{S}_0, \mathcal{R}_0, m_0, \_), r_1 = (\mathcal{S}_1, \_, \_, \_), b)$
  **if** $st = \mathsf{fresh} \lor st = (\mathcal{S}_0, \mathcal{S}_1)$ **then**
```

```
    output $((\mathcal{S}_b, \mathcal{R}_0, m_0, 1), st := (\mathcal{S}_0, \mathcal{S}_1))$
```

Session relationship anonymity. Relationship anonymity describes the anonymity of a relation between a user and the recipients, i.e., an adversary cannot find out both the sender of a message and the recipient.

```
$\alpha_{\mathrm{SRel}}(st, r_0 = (\mathcal{S}_0, \mathcal{R}_0, m_0, \_), r_1 = (\mathcal{S}_1, \mathcal{R}_1, \_, \_), b)$
  if $st = \mathsf{fresh}$ then
    $a \leftarrow \{0, 1\}$
  else if $\exists x.\ st = (\mathcal{S}_0, \mathcal{S}_1, x)$ then
    $a := x$
  if b=0 then
    output $((\mathcal{S}_a, \mathcal{R}_a, m_0, 1), st := (\mathcal{S}_0, \mathcal{S}_1, a))$
  else
    output $((\mathcal{S}_a, \mathcal{R}_{1-a}, m_0, 1), st := (\mathcal{S}_0, \mathcal{S}_1, a))$
```

Session sender unlinkability. Our notion of session sender unlinkability allows the adversary (for every challenge) to first run arbitrarily many messages in one session (stage 1) and then arbitrarily many messages in another session (stage 2). In each session there is only one sender used for the whole session (that is chosen at random when the first challenge message for this challenge is sent), but depending on b, either the same or the other user is used for the second session.

```
$\alpha_{\mathrm{UL}}(st, r_0 = (\mathcal{S}_0, \mathcal{R}_0, m_0; S_0), r_1 = (\mathcal{S}_1, \_, m_1, \_), b)$
  if $m_1 = \mathsf{stage\ 1}$ then
    $t := \mathsf{stage\ 1}$
  else
    $t := \mathsf{stage\ 2}$
  if $st = \mathsf{fresh}$ then
    $a \leftarrow \{0, 1\}$
    output $((\mathcal{S}_a, \mathcal{R}_0, m_0, 1), (t, a))$
  else if $\exists x.\ st = (\mathsf{stage\ 1}, x)$ then
    $a := x$
    output $((\mathcal{S}_a, \mathcal{R}_0, m_0, 1), (t, a))$
  else if $\exists x.\ st = (\mathsf{stage\ 2}, x)$ then
    $a := x$
    if $b = 0$ then
      output $((\mathcal{S}_a, \mathcal{R}_0, m_0, 2), (\mathsf{stage\ 2}, a))$
    else
      output $((\mathcal{S}_{1-a}, \mathcal{R}_0, m_0, 2), (\mathsf{stage\ 2}, a))$
```

5.2 Adversary classes

For restricting the adversary we propose to formally introduce classes of adversaries: an *adversary class* is a PPT wrapper that restricts the adversary $\mathcal{A}$ in its possible output behavior, and thus, in its knowledge about the world. The following example shows a scenario in which an (unrestricted) adaptive adversary might be too strong.

Example 6: Tor with entry guards. Consider the AC protocol Tor with so-called entry guards. *Every user selects a small set of entry nodes (his guards) from which he chooses the entry node of every* *circuit. New guards are chosen only after several months. As a compromised entry node is fatal for the security guarantees that Tor can provide, the concept of entry guards helps in reducing the risk of choosing a malicious node. However, if such an entry guard is compromised the impact is more severe since an entry guard is used for a large number of circuits.*

An adaptive adversary can choose its targets adaptively and thus perform the following attack. It (statically) corrupts some nodes and then sends (polynomially) many messages $(\mathsf{input}, r = (\mathcal{S}, \_, \_, \_))$ *for different users* $\mathcal{S}$*, until one of them, say Alice, chooses a compromised node as its entry guard. Then* $\mathcal{A}$ *proceeds by using Alice and some other user in a challenge. As Alice will quite likely use the compromised entry guard again, the adversary wins with a very high probability (that depends on the number of guards per user).* ◇

Although this extremely successful attack is not unrealistic (it models the fact that some users that happen to use compromised entry guards *will* be deanonymized), it might not depict the attack scenario that we are interested in. Thus, we define an adversary class that makes sure that the adversary cannot choose its targets. Whenever the adversary starts a new challenge for (session) sender anonymity, the adversary class draws two users at random and places them into the challenge messages of this challenge.

Definition 18. *An* adversary class *is defined by a* PPT *machine* $\mathrm{A}(\cdot)$ *that encapsulates the adversary* $\mathcal{A}$*. It may filter and modify all outputs of* $\mathcal{A}$ *to the challenger in an arbitrary way, withhold them or generate its own messages for the challenger. Moreover it may communicate with the adversary.*

We say that a protocol is secure against an adversary class $\mathrm{A}(\cdot)$*, if it is secure against all machines* $\mathrm{A}(\mathcal{A})$*, where* $\mathcal{A}$ *is an arbitrary* PPT *machine.*

5.3 Sequentially composable adversary classes

In this section we show that adaptive α-IND-CDP against adversaries that only use one challenge immediately implies adaptive α-IND-CDP against adversaries that use more than one challenge. The quantitative guarantees naturally depend on the number of challenges. We present a composability theorem that is applicable for all adversary classes that are *composable* for the adjacency function in use. Before we present the theorem, we define and discuss this property.

5.3.1 Requirements for composability

Composability is relatively straightforward for the non-adaptive AnoA, as well as for adaptive adversaries that are not restricted by an adversary class. However, by restricting the adversary, the number of challenges that this adversary sends might make a qualitative difference instead of a quantitative one.

Example 7: Non-composable adversary class. *Consider the following adversary class* A_{nc}*. The adversary class relays all input messages to the challenger and replaces the messages of the first challenge by (say) random strings or error symbols. The adversary class simply relays all messages except the ones belonging to the first challenge to the challenger.*

Every protocol, even if completely insecure, will be α-IND-CDP *secure for one challenge for the class* A_{nc} *(as the adversary cannot gain any information about the bit* b *of the challenger), but it might not necessarily be secure against more than one challenge.* ◇

We proceed by defining when an adversary class A is composable. First, we introduce notation for games in which some of the queries are simulated. This simulatability is a necessary condition for the composability of an adversary class.

$$\begin{tikzcd}
\textsc{Ch} \arrow[r, dash] \arrow[d, dash] & \mathcal{A} \arrow[d, dash] \\
\mathrm{A} \arrow[r, dash] & {}
\end{tikzcd}$$

ACReal

$$\begin{tikzcd}
\textsc{Ch} \arrow[r, dash] \arrow[d, dash] & \mathcal{A} \arrow[d, dash] \\
\mathrm{A} \arrow[r, dash] & \mathcal{S}
\end{tikzcd}$$

ACSim

Figure 9: The two games from Construction 1

Construction 1. *Consider the two scenarios defined in Figure 9:*

ACReal(b, n)*:* $\mathcal{A}$ *communicates with* A *and* $\mathcal{A}$ *communicates with* Ch(b, n) *(as protocol). The bit of the challenger is* b *and the adversary may send challenge tags in* $\{1, \dots, n\}$*.*

ACSim$_{\mathrm{M}_z}(b, n)$*:* $\mathcal{A}$ *communicates with* $\mathrm{M}_z(b)$ *that in turn communicates with* A *and* $\mathcal{A}$ *communicates with* Ch(b, n) *(as protocol).The bit of the challenger is* b *and the adversary may send challenge tags in* $\{1, \dots, n\}$*.*

Definition 19 (Consistent simulator index)**.** *A* simulator index *(for* n *challenges) is a bitstring* $z = [(z_1, b_1), \dots, (z_n, b_n)] \in \{0,1\}^{2n}$*. A pair of simulator indices* $z, z' \in \{0,1\}^{2n}$ *(for* n *challenges) is consistent w.r.t.* b *if*

$$\forall i \in \{1, \dots, n\} \ s.t. \ z_i \neq z'_i . (z_i = \mathsf{sim} \Rightarrow b_i = b) \wedge (z'_i = \mathsf{sim} \Rightarrow b'_i = b).$$

Consider an adversary class A and an adjacency function α. The anonymity guarantee defined by α for a protocol against all adversaries A$(\cdot)$ is composable, if additional challenges do not allow an adversary to perform qualitatively new attacks; i.e., the adversary does not gain any conceptually new power. This can be ensured by requiring the following three conditions from an adversary class A with respect to α. It must not initiate challenges on its own (*reliability*), it must not behave differently depending on the challenge tag that is used for a challenge (*alpha-renaming*) and all its challenges have to be simulatable by the adversary (*simulatability*).

We define the third condition, simulatability, by requiring the existence of a simulator M for the adjacency function. Intuitively, the simulator withholds challenge messages from the adversary class. Instead M precomputes the adjacency function (for a bit b') on those challenge messages and sends the resulting rows as input messages. More precisely, M makes sure that neither the adversary, nor the protocol (not even both in collaboration) can distinguish, whether the adversary's challenge messages are received by the adversary class and processed by the challenger, or whether M mimics the challenge messages with input messages before. The simulation only has to be indistinguishable, if the simulator uses the correct bit b of the challenger for simulating a challenge.

Definition 20 (Condition for adversary class)**.** *An adversary class* A *is* composable *for an anonymity function* α*, if the following conditions hold:*

1. ***Reliability:*** A$(\mathcal{A})$ *never sends a message* $(\mathsf{challenge}, \_, \_, \Psi)$ *to the challenger before receiving a message* $(\mathsf{challenge}, \_, \_, \Psi)$ *from* $\mathcal{A}$ *with the same challenge tag* Ψ*.*

2. ***Alpha-renaming:*** A *does not behave differently depending on the challenge tags* Ψ *that are sent by* $\mathcal{A}$ *except for using it in its own messages* $(\mathsf{challenge}, \_, \_, \Psi)$ *to the challenger and in the (otherwise empty) message* $(\mathsf{answer\ for}, \_, \Psi)$ *to* $\mathcal{A}$*.*

3. ***Simulatability:*** *For every $n \in \mathsf{N}$ and every list $z = [(z_1, b_1), \ldots, (z_n, b_n)] \in \{0,1\}^{2n}$ there exists a machine M_z such that:*

   (a) For every $i \in \{1, \ldots, n\}$. If $z_i = \mathsf{sim}$ then M_z never sends a message $(\mathsf{challenge}, \_, \_, i)$ to A.

   (b) The games $\mathrm{ACR\scriptstyle{EAL}}(b, n)$ and $\mathrm{ACS\scriptstyle{IM}}_{\mathrm{M}_{z_{\mathsf{dontsim}}}}(b, n)$ (cf. Construction 1) are computationally indistinguishable, where $z_{\mathsf{dontsim}} = [(\mathsf{dontsim}, \_), \ldots, (\mathsf{dontsim}, \_)] \in \{0,1\}^{2n}$ for M_z and A.

   (c) for all simulator indices $z, z' \in \{0,1\}^{2n}$ that are consistent w.r.t. b (see Definition 19) $\mathrm{ACS\scriptstyle{IM}}_{\mathrm{M}_z}(b, n)$ and $\mathrm{ACS\scriptstyle{IM}}_{\mathrm{M}_{z'}}(b, n)$ are indistinguishable.

5.3.2 Sequential composability theorem for adversary classes

We finally present our composability theorem. For all adversary classes that are composable (for an anonymity notion α) it suffices to show anonymity against single challenges. The theorem then allows for deriving guarantees for more than one challenge.

Theorem 21. *For every protocol $\mathcal{P}$, every anonymity function α, every $n \in \mathsf{N}$ and every adversary class A that is composable. Whenever $\mathcal{P}$ is $(1, \epsilon, \delta)$-α-IND-CDP for A, with $\varepsilon \geq 0$ and $0 \leq \delta \leq 1$, then $\mathcal{P}$ is $(n, n \cdot \epsilon, n \cdot e^{n\epsilon} \cdot \delta)$-$\alpha$-IND-CDP for A.*

Proof outline. We show the composability theorem by induction over n. We assume the theorem holds for n and compute the anonymity loss between the games $\mathrm{ACR\scriptstyle{EAL}}(0, n+1)$, where we have $b = 0$ and $\mathrm{ACR\scriptstyle{EAL}}(1, n+1)$, where we have $b = 1$ via a transition of indistinguishable, or differentially private games.

We start with $\mathrm{ACR\scriptstyle{EAL}}(0, n+1)$ and introduce a simulator that simulates one of the challenges for the correct bit $b = 0$. We apply the induction hypothesis for the remaining n challenges (this introduces an anonymity loss of $(n \cdot \epsilon, n \cdot e^{n\epsilon} \cdot \delta)$). The simulator still simulates one challenge for $b = 0$, but the bit of the challenger is now $b = 1$. We then simulate all remaining n challenges for $b = 1$ and thus introduce a game in which all challenges are simulated. As the bit of the challenger is never used in the game, we can switch it back to $b = 0$ again and remove the simulation of the first challenge. We can apply the induction hypothesis again (we loose (ε, δ)) and switch the bit of the challenger to $b = 1$ again. In this game, we have one real challenge (for $b = 1$) and n simulated challenges (also for $b = 1$). Finally, we remove the simulator again and yield $\mathrm{ACR\scriptstyle{EAL}}(1, n+1)$.

We refer to Appendix A.4 for the full proof. □

Why do we need the condition from Definition 20? The adversary class must not allow for behavior, where the security of a protocol is broken depending on, e.g., the number of messages that are sent. Imagine a protocol $\mathcal{P}$ that leaks all secrets as soon as the second user sends a message to any recipient. Now imagine an adversary class A that only forwards challenges to the challenger and blocks all input messages. The protocol is clearly not composable.

Downward composability. The adversary classes that we define here are downwards composable for all protocols. More precisely, if a protocol $\mathcal{P}$ is α secure for an adversary class $\mathrm{A}_1(\cdot)$ and if A_2 is an arbitrary adversary class, then $\mathcal{P}$ is also α secure for an adversary class $\mathrm{A}_1(\mathrm{A}_2(\cdot))$.

This observation follows directly from the fact that within the adversary classes, arbitrary PPT Turing machines are allowed, which includes wrapped machines $\mathrm{A}_2(\cdot)$.

5.3.3 Examples for adversary classes

Adversaries that may not choose their target. In the previous definitions, the adversary can choose which users it wants to deanonymize. As a consequence, in the sender anonymity game the adversary can, adaptively, choose which user(s) it wants to target. This worst-case assumption models a very strong adversary and is for many scenarios unrealistic. In these scenarios it might be more accurate to analyze the anonymity of the average user, which means that an adversary has to deanonymize a randomly chosen user to break the anonymity property. We model such a scenario by an adversary class A_r that, for every (new) challenge with tag Ψ chooses two different users at random and puts them into the challenge messages for this tag.

Non adaptive adversaries. In combination with adversary classes, the adaptive variant of AnoA is a generalization of AnoA and, among many new and interesting scenarios, still allows for modeling the (non adaptive) basic variant of AnoA. Non-adaptive adversaries can be defined by an adversary class $\mathrm{A}_{\mathsf{non-adaptive}}$ that first waits for the adversary to specify all inputs and challenges (or rather: one challenge) and only then outputs them, one by one, to the challenger.

User profiles. In [JWJ$^+$13], realistic, but relatively simple user profiles are defined that determine the behavior of users. We can model such profiles by defining a specific adversary class. The adversary class initiates profiles such as the *typical* users (they use, e.g., Gmail, Google Calendar, Facebook and perform web search at certain points in time) or *BitTorrent* users (they use Tor for downloading files on other times) as machines and measures time internally. If the adversary sends an input message, the adversary class initiates a profile. The adversary class might also initiate more profiles for random users at the beginning, which corresponds to "noise". If the adversary sends a challenge message, the adversary class initiates profiles for two different users (or two different profiles, depending on the anonymity notion).

On its own, the adversary class machine runs a loop that activates the profiles and increases a value t for time every now and then, until the adversary decides to halt with its guess b^* of the challenge bit b. Although the adversary class activates the profiles in a random order, it makes sure that all profiles have been activated before it proceeds to the next point in time. It additionally tells the activated profiles the point in time, such that they can decide whether or not they want to output a valid user action r or an error symbol $\perp$. The adversary class then sends input messages or challenge messages, depending on how the profiles have been initialized.

6 Leveraging UC realizability

Our adversary model in AnoA is strong enough to capture well-known simulation-based composability frameworks (e.g., UC [Can01], IITM [KT13] or RSIM [BPW07]). In Section 7 we apply AnoA to a model in the simulation-based universal composability (UC) framework.

In this section, we briefly introduce the UC framework and then prove that α-IND-CDP is preserved under realization. Moreover, we discuss how this preservation allows for an elegant crypto-free anonymity proof for cryptographic AC protocols.

6.1 The UC framework

The UC framework allows for a modular analysis of security protocols. In the framework, the security of a protocol is defined by comparing it with a setting in which all parties have a direct and private connection to a trusted machine that provides the desired functionality. As an example consider an authenticated channel between two parties Alice and Bob. In the real world Alice calls a protocol that signs the message m to be communicated. She then sends the signed message over the network and Bob verifies the signature. In the setting with a trusted machine T, however, we do not need any cryptographic primitives: Alice sends the message m directly to T[6]. T in turn sends m to Bob, who trusts T and can be sure that the message is authentic. The trusted machine T is called the *ideal functionality.*

Security in the UC framework is defined as follows: A protocol is secure if an execution of this protocol is indistinguishable from an execution of the corresponding ideal functionality.

More formally, the notion of indistinguishability is captured in UC in terms of *realization*: A protocol π *UC-realizes* an ideal functionality $\mathcal{F}$ if for all PPT adversaries $\mathcal{A}$ there is a PPT simulator S such that no PPT machine can distinguish an interaction with π and $\mathcal{A}$ from an interaction with $\mathcal{F}$ and S. The distinguisher is connected to the protocol and the adversary (or the simulator). A full definition can be found in Appendix A.3.

6.2 Preservation of α-IND-CDP

We prove that α-IND-CDP is preserved by UC realization. This result is motivated by the ideas presented in the result of integrity property conservation by simulation-based indistinguishability shown by Backes and Jacobi [BJ03, Thm. 1].

As a consequence of this lemma, it suffices to apply AnoA to ideal functionalities: transferring the results to the real protocol weakens the anonymity guarantees only by a negligible amount.

Lemma 22 (Preservation lemma)**.** *Let $\mathcal{P}$ be (ϵ, δ)-α-IND-CDP and Π be a protocol. If Π UC-realizes $\mathcal{P}$ then Π is (ϵ, Δ)-α-IND-CDP with $\Delta = \delta + \delta'$ for some negligible value δ'.*

Proof outline. The proof of the preservation lemma is straightforward: If the success probability of an adversary in the real world differs in more than a negligible value from the ideal world, we can use this adversary to distinguish the real from the ideal game. □

The full proof can be found in Appendix A.3. This preservation lemma, in combination with an ideal functionality for an AC protocol, is useful for analyzing the AC protocol with respect to our strong anonymity definitions. In the next section, we exemplify approach by using an ideal functionality for Tor [BGKM12] and showing that the anonymity analysis of Tor boils down to a purely combinatorical analysis.

7 Analyzing Tor Anonymity

The onion routing (OR) [RSG98] network Tor [Tor03] is the most successful anonymity technology to date: hundreds of thousands individuals all over the world use it today to protect their privacy over the Internet. Naturally, Tor is our first choice for applying our AnoA framework.

We start our discussion by briefly describing the Tor protocol [DMS04] and its UC definition [BGKM12]. We then formally prove (ε, δ)-α-IND-CDP for Tor's UC definition and quantify anonymity provided by the Tor network in terms of the anonymity properties defined in Section 3. Finally, we consider a selection of system-level attacks (e.g., traffic analysis) and adaptations (e.g., entry guards) for Tor, and analyze their effects on Tor's anonymity guarantees.

[6]Recall that T and Alice are directly connected, as well as T and Bob.

```
**Upon input** $(r, m, S)$
  **if** lastsession $= (S, \mathcal{C})$ **then**
    send message $(\mathsf{send}, \mathcal{C}, (r, m))$ to $P_u$
  **else**
    $\mathcal{P} \leftarrow \text{RandomParties}(P_u)$
    send message $(\mathsf{createcircuit}, \mathcal{P})$ to $P_u$
    wait for response $(\mathsf{created}, \mathcal{C})$
    lastsession $:= (S, \mathcal{C})$
    send message $(\mathsf{send}, \mathcal{C}, (r, m))$ to $P_u$
RandomParties$(P_u)$:
  $l \overset{R}{\leftarrow} \{1, \ldots, n\}$
  $N := \{1, \ldots, n\}$
  **for** $j = 1$ to $l$ **do**
    $i_j \overset{R}{\leftarrow} N$
    $N := N \setminus \{i_j\}$
  **return** $(P_u, P_{i_1}, \ldots, P_{i_l})$
```

Figure 10: Wrapper module $\textsc{Env}_u$ for onion proxy P_u

7.1 Tor — the OR network

An OR network such as Tor [DMS04] consists of a set of *OR nodes (or proxies)* that relay traffic, a large set of users and a directory service that maintains and provides cryptographic and routing information about the OR nodes. Users utilize the Tor network by selecting a sequence of OR nodes and creating a path, called a *circuit*, over this set. This circuit is then used to forward the users' traffic and obscure the users' relationship with their destinations. It is important that an OR node cannot determine the circuit nodes other than its immediate predecessor and successor. In the OR protocol, this is achieved by wrapping every message in multiple layers of symmetric-key encryption. Symmetric keys are agreed upon between each OR node in the circuit and the user during the circuit construction phase.

Tor was designed to guarantee anonymity against partially global adversarys, i.e., adversarys that do not only control some OR nodes but also a portion of the network. However, an accurate anonymity quantification is not possible without formally modeling the OR protocol and its adversary. In an earlier work, a formal UC definition (an ideal functionality $\mathcal{F}_{\mathrm{OR}}$) for the OR network was presented, and a practical cryptographic instantiation was proposed [BGKM12]. We employ this ideal functionality $\mathcal{F}_{\mathrm{OR}}$ for instantiating the AnoA framework.

7.2 Anonymity analysis

We start our Tor analysis with a brief overview of the $\mathcal{F}_{\mathrm{OR}}$ functionality and refer the readers to [BGKM12] for more details. An excerpt of relevant details can also be found in Appendix B. $\mathcal{F}_{\mathrm{OR}}$ presents the OR definition in the message-based state transitions form, and defines sub-machines for all OR nodes in the ideal functionality. These sub-machines share a memory space in the functionality for communicating with each other. $\mathcal{F}_{\mathrm{OR}}$ assumes an adversary who might possibly control all communication links and destination servers, but cannot view or modify messages between

```
**Upon message** $m$ **from** $\mathcal{F}_{\text{NET}}$ **or** $\mathcal{F}_{\text{OR}}$
    send $m$ to the challenger
    reflect the message $m$ back to sender
```

Figure 11: Dummy-adversary in $\mathcal{F}_{\text{OR}}$

uncompromised parties due to the presence of secure and authenticated channels between the parties. In $\mathcal{F}_{\text{OR}}$ these secure channels are realized by having each party store their messages in the shared memory, and create and send corresponding handles $\langle P, P_{next}, h\rangle$ through the network. Here, P and P_{next} are the sender and the recipient of a message respectively and h is a handle, or pointer, for the message in the shared memory. Only messages that are visible to compromised parties are forwarded to $\mathcal{A}$.

We consider a partially global, passive adversary for our analysis using AnoA, i.e., $\mathcal{A}$ decides on a subset of nodes before the execution, which are then compromised. The adversary $\mathcal{A}$ then only reads intercepted messages, but does not react to them.

Tor sets a time limit (of ten minutes) for each established circuit. However, the UC framework does not provide a notion of time. $\mathcal{F}_{\text{OR}}$ models such a time limit by only allowing a circuit C to transport at most a constant number (say ttl_C) of messages.

In the context of onion routing, we interpret an input message $(\mathsf{input}, r = (\mathcal{S}, \mathcal{R}, m, sid))$ as a message that is transmitted through the OR-network, where m is the message sent from the user $\mathcal{S}$ to the recipient $\mathcal{R}$ in session that corresponds to the session with ID sid.The number of messages per session is bounded by ttl_C.

In order to make $\mathcal{F}_{\text{OR}}$ compatible with our α-IND-CDP definition, we require an additional wrapper functionality, which processes the input rows forwarded from the challenger Ch. This functionality is defined in Figure 10. Env_u receives a row, which had u as its user, as its input. It then initiates the circuit construction for the new session and sends all messages in the row through this circuit.

Messages intercepted by compromised nodes are sent to a network adversary $\mathcal{F}_{\text{NET}}$ described in Fig. 11. $\mathcal{F}_{\text{NET}}$ forwards all intercepted messages to the challenger, who in turn forwards them to $\mathcal{A}$.

We show that the Tor analysis can be based on a *distinguishing event* $\mathcal{D}$, which has already been identified in the first onion routing anonymity analysis by Syverson et al. [STRL00, Fig. 1]. The key observation is that the adversary can only learn about the sender or recipient of some message if he manages to compromise the entry- or exit-node of the circuit used to transmit this message. We define the distinguishing event $\mathcal{D}_\alpha$ for each of the anonymity notions defined in section 3.

(Session) Sender Anonymity (α_{SSA}). Let $\mathcal{D}_{\alpha_{\text{SSA}}}$ be the event that the entry-node of the challenge session is compromised by $\mathcal{A}$. This allows $\mathcal{A}$ to determine the sender of the challenge session and therefore break sender anonymity.

(Session) Sender Unlinkability (α_{SUL}). Let $\mathcal{D}_{\alpha_{\text{SUL}}}$ be the event that $\mathcal{A}$ successfully compromises the entry nodes for both challenge sessions in the unlinkability game. This allows $\mathcal{A}$ to determine whether the sessions are linked or not, and hence break the unlinkability game.

(Session) Relationship Anonymity (α_{SRel}). Let $\mathcal{D}_{\alpha_{\text{SRel}}}$ be the event that $\mathcal{A}$ successfully compromises entry- and exit-node of the challenge session. This allows him to link both sender and recipient of the session.

We first prove Lemma 23. It captures anonymity provided by $\mathcal{F}_{\text{OR}}$ in case $\mathcal{D}$ does not happen. We then use Lemma 23 to prove (ε, δ)-α-IND-CDP for $\mathcal{F}_{\text{OR}}$ in general.

We introduce random strings $r_{\mathcal{A}}$ and r_{CH} as additional input to the adversary and the challenger respectively. This allows us to handle them as deterministic machines and simplifies the proof for Lemma 23. Accordingly, all subsequent probabilities are taken over those random strings. In the following, we present a proof outline here.

The full proof is available in Appendix C.1.

Lemma 23. *Given an adjacency function $\alpha \in \{\alpha_{\text{SSA}}, \alpha_{\text{SUL}}, \alpha_{\text{SRel}}\}$, the following two probabilities are equal:*

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,0,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}); r_{\mathcal{A}}, r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\\ =&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,1,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}); r_{\mathcal{A}}, r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\end{aligned}$$

Proof outline. We fix the random strings $r_{\mathcal{A}}$ and r_{CH} of the adversary and the challenger. This in turn fixes both the inputs created by the adversary and the circuits drawn by $\mathcal{F}_{\text{OR}}$ for each session. As circuits are drawn independently from the transmitted messages, $\mathcal{F}_{\text{OR}}$ draws the set of circuits independently of the bit b.

We assume the event $\neg\mathcal{D}_\alpha$. For α_{SSA} the messages intercepted by $\mathcal{A}$ do not carry critical information and look the same, regardless of the bit b of the challenger. Since we also fixed $r_{\mathcal{A}}$, $\mathcal{A}$ returns the same value after processing the set of intercepted messages in either case.

For α_{SUL} and α_{SRel}, $\mathcal{A}$ might learn partial information, e.g., the sender of one of the sessions. However, as we did not fix the randomness of the adjacency function, those observations are equally likely for both possible bits b. Hence we get

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,0,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\\ =&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,1,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\end{aligned}$$

and from this

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,0,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}(r_{\text{CH}}, r_{\mathcal{A}}); r_{\mathcal{A}}, r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\\ =&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,1,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}(r_{\text{CH}}, r_{\mathcal{A}}); r_{\mathcal{A}}, r_{\text{CH}} \overset{R}{\leftarrow} \{0,1\}^{p(\eta)}]\end{aligned}$$

as required. □

With this result we obtain (ε, δ)- α-IND-CDP for $\mathcal{F}_{\text{OR}}$ by simple manipulation of equations.

Theorem 24. *$\mathcal{F}_{\text{OR}}$ is $(0, \delta)$ - α-IND-CDP for $\alpha \in \{\alpha_{\text{SSA}}, \alpha_{\text{SUL}}, \alpha_{\text{SRel}}\}$, i.e*

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,0,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0]\\ \leq&\Pr[\mathcal{A}^{\text{CH}(\mathcal{F}_{\text{OR}},\alpha,1,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0]) + \delta\end{aligned}$$

with $\delta = \Pr[\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}})]$.

Here δ is exactly the probability for the event $\mathcal{D}_\alpha$ that allows $\mathcal{A}$ to distinguish between both scenarios. Interestingly, we get the parameter value $\varepsilon = 0$. This implies that as long as $\mathcal{D}_\alpha$ does not happen, $\mathcal{F}_{\text{OR}}$ provides perfect anonymity for its users.

7.3 Anonymity quantification

We now evaluate the guarantees provided by Theorem 24 and consider further results we can derive from it for the special case of sender anonymity.

7.3.1 Distinguishing events

We measure the probability of the distinguishing event $\mathcal{D}$ using combinatorial observations. For an OR network of n OR nodes such that k of those are compromised, probabilities associated with the various anonymity notions are as follows:

Sender Anonymity (α_{SSA}). The probability that $\mathcal{D}_{\alpha_{\text{SSA}}}$ happens and sender anonymity is broken, is

$$Pr[\mathcal{D}_{\alpha_{\text{SSA}}}] = 1 - \frac{\binom{n-1}{k}}{\binom{n}{k}} = \frac{k}{n}$$

Sender Unlinkability (α_{SUL}). The probability that $\mathcal{D}_{\alpha_{\text{SUL}}}$ happens and sender unlinkability is broken, is

$$Pr[\mathcal{D}_{\alpha_{\text{SUL}}}] = \left(\frac{k}{n}\right)^2$$

Relationship Anonymity (α_{SRel}). The probability that $\mathcal{D}_{\alpha_{\text{SRel}}}$ happens and relationship anonymity is broken, is

$$Pr[\mathcal{D}_{\alpha_{\text{SRel}}}] = \frac{\binom{n-2}{k-2}}{\binom{n}{k}} = \frac{k(k-1)}{n(n-1)}$$

Figure 12 illustrates these results. The graph shows the probability of the distinguishing events depending on the fraction $\frac{k}{n}$ of corrupted OR nodes. We assume a system with 3000 OR nodes, which is consistent with current numbers in the real world Tor network [Tor]. We observe that the success probability of event $\mathcal{D}_{\alpha_{\text{SSA}}}$ always remains above the success probabilities of events $\mathcal{D}_{\alpha_{\text{SUL}}}$ and $\mathcal{D}_{\alpha_{\text{SRel}}}$. Therefore, sender anonymity is indeed a stronger notion than both relationship anonymity and sender unlinkability, and correspondingly more difficult to achieve.

Note that the above analysis and the underlying model assume all OR nodes to be identical, and can perform all roles. Respecting OR node operators' legal boundaries, the real-world Tor network allows OR nodes to function in specific roles. To some extent, this simplifies $\mathcal{A}$'s task of identifying entry- or exit-nodes for circuits.

Multiple Challenges. In this section we considered the number of allowed challenges to be set to $n = 1$. However, scenarios in which the adversary can use more than one challenge can directly be derived by application of Theorem 21.

Adversary classes. The analysis of Tor conducted in this section does not explicitly mention adversary classes (cf. Definition 18). However, since a PPT adversary class $\mathrm{A}(\cdot)$ applied to a PPT adversary is again a PPT adversary, naturally, the analysis holds for arbitrary adversary classes.

7.4 System-level attacks and adaptations

Next, we consider attacks that are not directly covered by our model and explore how the strong adversary we employ helps to deal with them. We then analyze the *entry guard* mechanism, a feature of the Tor protocol, and its influence on sender anonymity.

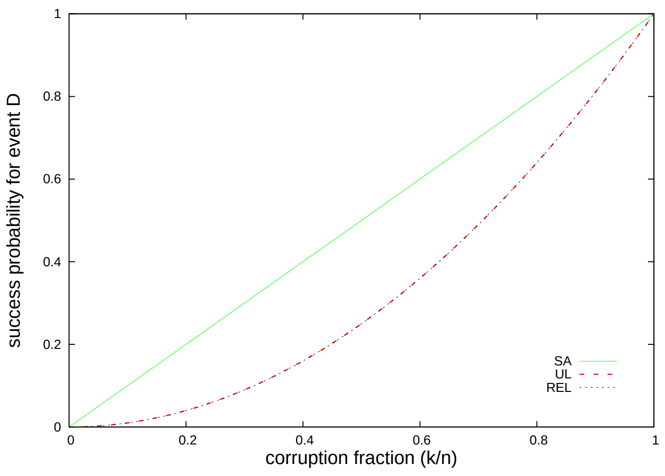

Figure 12: Probability of $\mathcal{D}$ for the different anonymity notions, depending on the corruption $\frac{k}{n}$ for 3000 OR nodes

7.4.1 Traffic analysis attacks

Many of the known attacks on Tor nowadays depend on so called side-channel information, i.e. throughput and timing information an adversary might gather while watching traffic routed through the Tor network. Since the UC framework does not allow time-sensitive attacks, traffic analysis is outside of the scope of this work. However, due to the strong adversary we deploy, we can still cover all known attacks by making suitable assumptions. In the following we look at two well known traffic analysis attacks and how we can cover them in our model.

Traffic correlation. These forms of traffic analysis attacks observe traffic going out from the sender and into the receiver and try to correlate them based on different features like volume, direction or inter-packet delay [OB09, WRW02]. We model these attacks by over-approximating the adversary. We allow adversaries that only send a single challenge. If the adversary controls the links or nodes that are needed for a traffic correlation attack, i.e., the entry and exit node or the link from the user to the entry node and the link from the exit node to the server, then the adversary can correlate the traffic.

Website fingerprinting. Fingerprinting attacks try to classify user traffic based on a catalog of fingerprints derived for a large set of web pages beforehand and matching the observed traffic to those fingerprints [PNZE11, CZJJ12, DCRS12]. This kind of attack can be modeled by assuming that it is enough for the adversary to compromise the entry node (i.e. we define a new distinguishing event $\mathcal{D}_{WF}$ that captures this) to find the recipient, as he will then be able to launch the fingerprinting attack. The δ in Theorem 24 then changes to

$$\delta = Pr[\mathcal{D}_{WF}] \cdot Pr[\mathcal{S}]$$

where $\mathcal{S}$ is the event that the website fingerprinting attack successfully classifies the traffic.

7.4.2 Entry guards

When regarding adversaries that may use many challenges, it is interesting to analyze the concept of *entry guards* [WALS03, ØS06], which are used in the current implementation of Tor. Entry guards are a small subset of the whole set of onion routers that are chosen by a user before the initiation of a Tor communication. They are then used as entry nodes for any subsequent communication. As we discussed in Section 5, entry guards can be problematic if an adversary may choose adaptively which users it targets. Such an adversary can initiate many messages from (fresh) users until a user chooses a corrupted node as its entry guard. Then the adversary starts a challenge with this user and some other user and wins the sender anonymity game with a very high probability (as it has corrupted an entry guard of one of the users). The situation is vastly different if we consider adversaries that may not adaptively choose their targets, although they can adaptively choose their messages and thus still utilize their adaptivity. This can be formalized by the adversary class A_r as discussed in Section 5.3.3.

Consider a single, random user u who communicates using Tor over a long period of time, initiating a total of d new sessions. Without entry guards, the probability that $\mathcal{A}$ de-anonymizes u is bounded by

$$1 - \frac{\binom{n-d}{k}}{\binom{n}{k}}$$

which converges to 1 the bigger d gets. If we do use a set of m entry guards on the other hand, the probability for de-anonymizing u will stay constant at

$$1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}.$$

In order to prevent loss of performance, entry guards are also replaced at regular intervals. Let l be the maximum number of sessions possible per entry-guard-interval. The probability for de-anonymization can then be bounded by

$$1 - \frac{\binom{n-\lceil \frac{d}{l} \rceil m}{k}}{\binom{n}{k}}$$

which is smaller than the original value, but still converges to 1 at some point. Note that these upper bounds only make sense if the sessions initiated per entry-guard-interval also use each entry-guard at least once. Dropping this assumption requires a more fine-grained analysis.

The problem with entry guards is the following: while the probability for de-anonymization is smaller, u will effectively stay de-anonymized as soon as $\mathcal{A}$ manages to find u's entry guards (for as long as these entry guards are used). Also, while the above value attains its minimum for $m = 1$, choosing a small value for m will realistically also incur loss in performance for the whole system. The exact analysis is unfortunately out-of-scope for our approach, but further elaboration on the parameters and their influence on anonymity and performance using simulation can be found in Elahi et al. [EBA+12].

7.5 Link corruption

So far we have only been concerned with an adversary $\mathcal{A}$ that compromises nodes in the onion routing network in order to learn about the transmitted messages. But our model also supports an adversary that compromises links between nodes and learns about messages transmitted through these links.

Thus, the event $\mathcal{D}_{\alpha_{\mathrm{SSA}}}$ alone is not enough to capture all bad events. For sender anonymity, we also lose if the adversary manages to compromise the link between the user and the entry node of the circuit used to transmit the challenge row. Let $\mathcal{L}_{\alpha_{\mathrm{SSA}}}$ be the event that this entry link is compromised and let q be the number of compromised links. Naturally, it is in the best interest of the adversary to not compromise links between user/server and already compromised nodes, as he will not learn anything new that way. Hence we have that

$$Pr[\mathcal{L}_{\alpha_{\text{SSA}}}] \leq \frac{q}{n-k}$$

In order to extend our δ by the event $\mathcal{L}_{\alpha_{\text{SSA}}}$, we can now consider the "bad event" $\mathcal{B}_{\alpha_{\text{SSA}}}$ depending on $\mathcal{D}_{\alpha_{\text{SSA}}}$:

$$\begin{aligned}
Pr[\mathcal{B}_{\alpha_{\text{SSA}}}] &= \Pr[\mathcal{B}_{\alpha_{\text{SSA}}} | \mathcal{D}_{\alpha_{\text{SSA}}}] \cdot Pr[\mathcal{D}_{\alpha_{\text{SSA}}}] \\
&\qquad + Pr[\mathcal{B}_{\alpha_{\text{SSA}}} | \neg\mathcal{D}] \cdot Pr[\neg\mathcal{D}_{\alpha_{\text{SSA}}}] \\
&= Pr[\mathcal{D}_{\alpha_{\text{SSA}}}] + Pr[\mathcal{L}_{\alpha_{\text{SSA}}}] \cdot Pr[\neg\mathcal{D}_{\alpha_{\text{SSA}}}] \\
&\leq \frac{k}{n} + \frac{q}{n-k}\frac{n-k}{n} \\
&= \frac{k+q}{n}
\end{aligned}$$

For more than one challenge session this can be extended in a similar way as before, by just adjusting the event for successful link corruption. Let $\mathcal{L}^*_{\alpha_{\text{SSA}}}$ be the event that in one of the challenge sessions, an entry-link was successfully compromised. Doing a similar analysis as for the node corruption, we get the following upper bound, which is tight if the user for all challenge sessions is the same.

$$Pr[\mathcal{L}^*_{\alpha_{\text{SSA}}}] \leq 1 - (1 - \frac{q}{n-k})^d \tag{1}$$

The full derivation of Inequality 1 can be found in Appendix C.2.

This concludes the formal analysis of the Tor network with the AnoA framework. We illustrated how AnoA can be used by using it on $\mathcal{F}_{\text{OR}}$. We showed that $\mathcal{F}_{\text{OR}}$ is $(0, \delta)$-α-IND-CDP for the different anonymity notion we defined in Section 3 and also explored further aspects of OR anonymity accessible through the AnoA framework. Still, we barely scratched the surface with our analysis and see many different directions for future work in the Tor analysis with AnoA. We further elaborate on these directions in Section 9.

Note that, although we only considered the ideal functionality $\mathcal{F}_{\text{OR}}$ in our analysis, Theorem 22 allows us to lift our results to any (cryptographic) protocol that realizes $\mathcal{F}_{\text{OR}}$.

8 Related Work

Pfitzmann and Hansen [PH10] develop a consistent terminology for various relevant anonymity notions; however, their definitions lack formalism. Nevertheless, these informal definitions form the basis of almost all recent anonymity analysis, and we also adopt their terminology and definitions in our AnoA framework.

Our relaxation of differential privacy is not the first variation of differential privacy. Gehrke et al. recently introduced the stronger notion of zero-knowledge privacy [GLP11] and the relaxed notion of crowd-blending privacy [GHLP12]. Similar to differential privacy, these notions are not well suited for the analysis of AC protocols. However, extending the crowd-blending privacy notion with corruptible distributed mechanisms and flexible adjacency functions would allow capturing the notion of k-anonymity for AC protocols. We could imagine applying the resulting concept to Mixnets, in which each mix waits for a certain amount of time: if at least k messages arrived, these messages are then processed, otherwise they are discarded; however, discarding messages in such a way may not be acceptable in a real world application.

Efforts to formally analyze anonymity properties have already been made using communicating sequential processes (CSP) [SS96], epistemic logic [SS99, HO05], Kripke structures [HS04], and probabilistic automata [APSVR11]. However, these formalisms have only being applied to simple protocols such DC-net. Since it's not clear if these frameworks can capture an adversary with auxiliary information, it seems difficult to model complex protocols such as onion routing and its traffic analysis attacks. It still presents an interesting challenge to relate the probabilistic notions among those mentioned above (e.g. [HO05, APSVR11]) to our anonymity framework.

There have been analyses which focus on a particular AC protocol, such as [SD02, D0́6, SW06, GTD$^+$08] for Mixnet, [BP05, APSVR11] for DC-net, [DSCP02, Shm04] for Crowds, and [STRL00, MVdV04, FJS07a, FJS07b, FJS12] for onion routing. Most of these study a particular anonymity property in a particular scenario and are not flexible enough to cover the emerging system-level attacks on the various AC protocols. (We refer the readers to [FJS12, Sec. 5] for a detailed survey.) The most recent result [FJS12] among these by Feigenbaum, Johnson and Syverson models the OR protocol in a simplified black-box abstraction, and studies a notion of relationship anonymity notion which is slightly different from ours: here the adversary wishes to identify the destination of a user's message. As we discussed in Section 4.3, this relationship anonymity notion is slightly weaker than ours. Moreover, their model is not flexible enough to extend to other system-level scenarios such fingerprinting attacks [PNZE11, CZJJ12, DCRS12].

Hevia and Micciancio [HM08] introduce an indistinguishability based framework for the analysis of AC protocols. While they take a similar approach as in AnoA, there are some notable differences: The first difference is that their anonymity definition does not consider compromised parties; as a consequence, they only define qualitative anonymity guarantees. While the authors discuss corruption as a possible extension, for most real world AC protocols they would have to adjust their notion to a quantitative anonymity notion as in AnoA. The second difference is the strength of the adversary: we consider a stronger adversary which determine the order in which messages are sent through the network, whereas Hevia and Micciancio only allow the adversary to specify which party sends which messages to whom. Building on the work of Hevia and Micciancio, Gelernter and Herzberg published, concurrently with this work, an expressive framework that extends the work of Hevia and Micciancio with active adversaries that adaptively send inputs [GH13]. They apply their methodology to obtain a strong impossibility result for a strong combination of sender anonymity and unobservability, which they coin "ultimate anonymity". However, as the work of Hevia and Micciancio, they only consider qualitative anonymity notions (i.e., protocols that only allow a negligible chance of de-anonymizing users), which does not allow them to quantify the anonymity loss in low-latency anonymous communication networks, such as Tor. Moreover, they do not provide a mechanism, such as adversary classes, to flexibly relax the strength of the adversary.

9 Conclusion and Future Directions

In this paper we have presented our generic framework AnoA. We have defined new, strong variants of anonymity properties like sender anonymity, sender unlinkability and relationship anonymity based on a novel relaxation of computational differential privacy, and presented how to concisely formulate them in AnoA. We have shown that our definitions of the anonymity guarantees accurately model prominent notions in the literature. Moreover, we have generalized those definitions to allow for adaptive adversaries that may be restricted in a controlled and modular manner. We have also applied AnoA to the UC framework and shown that the results shown for ideal functionalities carry over to their secure cryptographic protocols.

Additionally, we have conducted an extensive analysis of the Tor network. We have validated the inherent imperfection of the current Tor standard in the presence of a significant fraction of compromised nodes, and we have given quantitative measures of the different forms of anonymity against passive adversaries that statically corrupt nodes.

Future directions. In our analysis of Tor we did not consider the impact of user preferences for path selection. If certain nodes are more likely for a given user (e.g. for efficiency reasons), anonymity can (and will) decrease. As illustrated in Example 3, when analyzing Tor with preferences, the value for ε can be larger than zero (depending on the preferences). We plan to analyze the influence of Tor's node selection preferences [DM09] on Tor's anonymity guarantees and to investigate active attacks on Tor such as elective DoS attacks [BDMT07].

The next step is to analyze anonymity guarantees in scenarios where the adversary may not choose the user behavior, e.g., in cases where the behavior is defined by profiles. Moreover, we will apply AnoA to other AC protocols such as Mixnets [Cha81] and the DISSENT system [CGF10].

On the framework level we will investigate other anonymity notions such as unobservability and undetectability [PH10], and their relation to the notions we already defined in this paper.

Acknowledgment

We thank Aaron Johnson and the anonymous reviewers for their useful suggestions. This work was partially supported by the German Universities Excellence Initiative, the ERC Grant End-2-End Security, and the Center for IT-Security, Privacy and Accountability (CISPA).

References

[APSVR11] E. Andrés, Miguel, Catuscia Palamidessi, Ana Sokolova, and Peter Van Rossum. Information Hiding in Probabilistic Concurrent System. *Journal of Theoretical Computer Science (TCS)*, 412(28):3072–3089, 2011.

[AYM12] Masoud Akhoondi, Curtis Yu, and Harsha V. Madhyastha. LASTor: A Low-Latency AS-Aware Tor Client. In *Proc. of the 2012 IEEE Symposium on Security and Privacy (S& P)*, pages 476–490. IEEE Computer Society, 2012.

[BDMT07] Nikita Borisov, George Danezis, Prateek Mittal, and Parisa Tabriz. Denial of Service or Denial of Security? In *Proc. 14th ACM Conference on Computer and Communications Security (CCS)*, pages 92–102, 2007.

[BGKM12] Michael Backes, Ian Goldberg, Aniket Kate, and Esfandiar Mohammadi. Provably Secure and Practical Onion Routing. In *Proc. 26st IEEE Symposium on Computer Security Foundations (CSF)*, pages 369–385, 2012.

[BJ03] Michael Backes and Christian Jacobi. Cryptographically Sound and Machine-Assisted Verification of Security Protocols. In *Proceedings of 20th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 675–686, 2003.

[BKM$^+$13] Michael Backes, Aniket Kate, Praveen Manoharan, Sebastian Meiser, and Esfandiar Mohammadi. AnoA: A Framework For Analyzing Anonymous Communication Protocols — Unified Definitions and Analyses of Anonymity Properties. In *Proc. 27th IEEE Symposium on Computer Security Foundations (CSF)*, 2013.

[BMP00] V. Boyko, P. D. MacKenzie, and S. Patel. Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. In *Advances in Cryptology — EUROCRYPT*, pages 156–171, 2000.

[BP05] Mohit Bhargava and Catuscia Palamidessi. Probabilistic Anonymity. In *CONCUR*, pages 171–185, 2005.

[BPW07] Michael Backes, Birgit Pfitzmann, and Michael Waidner. The Reactive Simulatability (RSIM) Framework for Asynchronous Systems. *Information and Computation*, 205(12):1685–1720, 2007.

[Can01] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 136–145, 2001.

[CGF10] Henry Corrigan-Gibbs and Bryan Ford. Dissent: Accountable Anonymous Group Messaging. In *Proc. 17th ACM Conference on Computer and Communication Security (CCS)*, pages 340–350, 2010.

[Cha81] D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 4(2):84–88, 1981.

[Cha88] David Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *J. Cryptology*, 1(1):65–75, 1988.

[CL05] J. Camenisch and A. Lysyanskaya. A Formal Treatment of Onion Routing. In *Advances in Cryptology — CRYPTO*, pages 169–187, 2005.

[CZJJ12] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *Proc. 19th ACM Conference on Computer and Communication Security (CCS)*, pages 605–616, 2012.

[D0́6] Claudia Díaz. Anonymity Metrics Revisited. In *Anonymous Communication and its Applications*, 2006.

[DCRS12] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *Proc. 33th IEEE Symposium on Security and Privacy*, pages 332–346, 2012.

[DG09] G. Danezis and I. Goldberg. Sphinx: A Compact and Provably Secure Mix Format. In *Proc. 30th IEEE Symposium on Security and Privacy*, pages 269–282, 2009.

[DM09] R. Dingledine and S.J. Murdoch. Performance Improvements on Tor or, Why Tor is slow and what we're going to do about it. *Online: http://www. torproject. org/press/presskit/2009-03-11-performance. pdf*, 2009.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Proc. 10th Theory of Cryptography Conference (TCC)*, pages 265–284, 2006.

[DMS04] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proc. 13th USENIX Security Symposium (USENIX)*, pages 303–320, 2004.

[DSCP02] Claudia Díaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards Measuring Anonymity. In *Proc. 2nd Workshop on Privacy Enhancing Technologies (PET)*, pages 54–68, 2002.

[DvOW92] W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and Authenticated Key Exchanges. *Des. Codes Cryptography*, 2(2):107–125, 1992.

[Dwo06] Cynthia Dwork. Differential Privacy. In *ICALP (2)*, pages 1–12, 2006.

[EBA$^+$12] Tariq Elahi, Kevin S. Bauer, Mashael AlSabah, Roger Dingledine, and Ian Goldberg. Changing of the Guards: A Framework for Understanding and Improving Entry Guard Selection in Tor. In *Proc. 11th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 43–54, 2012.

[FJS07a] J. Feigenbaum, A. Johnson, and P. F. Syverson. A Model of Onion Routing with Provable Anonymity. In *Proc. 11th Conference on Financial Cryptography and Data Security (FC)*, pages 57–71, 2007.

[FJS07b] J. Feigenbaum, A. Johnson, and P. F. Syverson. Probabilistic Analysis of Onion Routing in a Black-Box Model. In *Proc. 6th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 1–10, 2007.

[FJS12] Joan Feigenbaum, Aaron Johnson, and Paul F. Syverson. Probabilistic Analysis of Onion Routing in a Black-Box Model. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):14, 2012.

[GH13] Nethanel Gelernter and Amir Herzberg. On the limits of provable anonymity. In *Proc. 12th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 225–236, 2013.

[GHLP12] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. Crowd-Blending Privacy. In *Advances in Cryptology — CRYPTO*, pages 479–496, 2012.

[GL01] O. Goldreich and Y. Lindell. Session-Key Generation Using Human Passwords Only. In *Advances in Cryptology — CRYPTO*, pages 408–432, 2001.

[GLP11] Johannes Gehrke, Edward Lui, and Rafael Pass. Towards Privacy for Social Networks: A Zero-Knowledge Based Definition of Privacy. In *Proc. 8th Theory of Cryptography Conference (TCC)*, pages 432–449, 2011.

[GTD$^+$08] Benedikt Gierlichs, Carmela Troncoso, Claudia Díaz, Bart Preneel, and Ingrid Verbauwhede. Revisiting a Combinatorial Approach toward Measuring Anonymity. In *Proc. 7th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 111–116, 2008.

[HM08] Alejandro Hevia and Daniele Micciancio. An Indistinguishability-Based Characterization of Anonymous Channels. In *Proc. 8th Privacy Enhancing Technologies Symposium (PETS)*, pages 24–43, 2008.

[HO05] Joseph Y. Halpern and Kevin R. O'Neill. Anonymity and Information Hiding in Multiagent Systems. *Journal of Computer Security*, 13(3):483–512, 2005.

[Hof11] D. Hofheinz. Possibility and Impossibility Results for Selective Decommitments. *J. Cryptology*, 24(3):470–516, 2011.

[HS04] Dominic Hughes and Vitaly Shmatikov. Information Hiding, Anonymity and Privacy: a Modular Approach. *Journal of Computer Security*, 12(1):3–36, 2004.

[JWJ$^+$13] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users get routed: Traffic correlation on tor by realistic adversaries. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS '13)*. ACM, 2013.

[KG10] A. Kate and I. Goldberg. Using Sphinx to Improve Onion Routing Circuit Construction. In *Proc. 14th Conference on Financial Cryptography and Data Security (FC)*, pages 359–366, 2010.

[KT13] Ralf Küsters and Max Tuengerthal. The IITM Model: a Simple and Expressive Model for Universal Composability. *IACR Cryptology ePrint Archive*, 2013:25, 2013.

[MPRV09] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil P. Vadhan. Computational Differential Privacy. In *Advances in Cryptology — CRYPTO*, volume 5677, pages 126–142, 2009.

[MVdV04] S. Mauw, J. Verschuren, and E. de Vink. A Formalization of Anonymity and Onion Routing. In *Proc. 9th European Symposium on Research in Computer Security (ESORICS)*, pages 109–124, 2004.

[OB09] G. O'Gorman and S. Blott. Improving Stream Correlation Attacks on Anonymous Networks. In *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC)*, pages 2024–2028, 2009.

[ØS06] L. Øverlier and P. F. Syverson. Locating Hidden Servers. In *Proc. 27th IEEE Symposium on Security and Privacy*, pages 100–114, 2006.

[PH10] Andreas Pfitzmann and Marit Hansen. A Terminology for Talking about Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, August 2010. v0.34.

[PNZE11] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website Fingerprinting in Onion Routing Based Anonymization Networks. In *Proc. 10th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 103–114, 2011.

[RR98] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1):66–92, 1998.

[RSG98] M. Reed, P. Syverson, and D. Goldschlag. Anonymous Connections and Onion Routing. *IEEE J-SAC*, 16(4):482–494, 1998.

[SD02] Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity. In *Proc. 2nd Workshop on Privacy Enhancing Technologies (PET)*, pages 41–53, 2002.

[Shm04] Vitaly Shmatikov. Probabilistic Analysis of an Anonymity System. *Journal of Computer Security*, 12(3-4):355–377, 2004.

[SS96] Steve Schneider and Abraham Sidiropoulos. CSP and Anonymity. In *Proc. 4th European Symposium on Research in Computer Security (ESORICS)*, pages 198–218, 1996.

[SS99] Paul F. Syverson and Stuart G. Stubblebine. Group Principals and the Formalization of Anonymity. In *World Congress on Formal Methods*, pages 814–833, 1999.

[STRL00] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an Analysis of Onion Routing Security. In *Proc. Workshop on Design Issues in Anonymity and Unobservability (WDIAU)*, pages 96–114, 2000.

[SW06] Vitaly Shmatikov and Ming-Hsiu Wang. Measuring Relationship Anonymity in Mix Networks. In *Proc. 7th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 59–62, 2006.

[Tor] Tor Metrics Portal. `https://metrics.torproject.org/`. Accessed Feb 2013.

[Tor03] The Tor Project. `https://www.torproject.org/`, 2003. Accessed Feb 2013.

[WALS03] Matthew Wright, Micah Adler, Brian Neil Levine, and Clay Shields. Defending Anonymous Communication Against Passive Logging Attacks. In *Proc. 24th IEEE Symposium on Security and Privacy*, pages 28–43, 2003.

[Wik04] Douglas Wikström. A Universally Composable Mix-Net. In *Proc. of the 1st Theory of Cryptography Conference (TCC)*, pages 317–335, 2004.

[WRW02] Xinyuan Wang, Douglas S. Reeves, and Shyhtsun Felix Wu. Inter-Packet Delay Based Correlation for Tracing Encrypted Connections through Stepping Stones. In *Proc. 7th European Symposium on Research in Computer Security (ESORICS)*, pages 244–263, 2002.

A Framework

In this section we provide the proofs for the claims made in section 4.

A.1 Expressivity

We first recall the definition of δ-sender anonymity derived from the literature.

Definition 7. *[sender anonymity]A protocol $\mathcal{P}$ with user space $\mathcal{U}$ of size N has* δ-sender anonymity *if for all ppt-adversaries $\mathcal{A}$*

$$Pr\left[u^* = u : u^* \leftarrow \mathcal{A}^{\mathrm{SAC\textsc{h}}_u(\mathcal{P})}, u \overset{R}{\leftarrow} \mathcal{U}\right] \leq \frac{1}{N} + \delta$$

where the challenger $\mathrm{SAC\textsc{h}}$ *is as defined in Figure 5.*

Our definition with the adjacency function α_{SA} implies definition 7.

Lemma 8. *[sender anonymity] For all protocols $\mathcal{P}$ over a (finite) user space $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has* $(0,\delta)$-α-IND-CDP *for α_{SA}, $\mathcal{P}$ also has δ-sender anonymity as in Definition 7.*

Proof. We proof this by contradiction: given an adversary $\mathcal{A}^*$ that breaks δ-sender anonymity, we construct an adversary $\mathcal{A}$ that breaks $(0,\delta) - \alpha$-IND-CDP for α_{SA}. Let $\mathcal{A}^*$ be a ppt-adversary that wins against $\mathrm{C\textsc{h}}_b(\mathcal{P}, \alpha_{\mathrm{SA}})$ with probability more than $\frac{1}{N} + \delta$. Let $\mathcal{A}$ be defined as follows:

```
**$\mathcal{A}$ with oracle access to Ch$_b(\mathcal{P}, \alpha_{\mathrm{SA}})$**
  Call $\mathcal{A}^*$ and receive a input table $D$
  if $\exists!$ challenge_line in $D$ then
    $u_0 \overset{R}{\leftarrow} \mathcal{U}$, $u_1 \overset{R}{\leftarrow} \mathcal{U} \setminus \{u_0\}$
    Let $D_0$ be $D$ with $u_0$ in the challenge_line.
    Let $D_1$ be $D$ with $u_1$ in the challenge_line.
    Send $(\mathsf{input}, D_0, D_1)$ to the oracle $\mathrm{C\textsc{h}}_b(\mathcal{P}, \alpha_{\mathrm{SA}})$
    Send every message from $\mathrm{C\textsc{h}}$ to $\mathcal{A}^*$.
    Receive $u^* \leftarrow \mathcal{A}^*$
    if $u^* =\in \{u_0, u_1\}$ then
      Output $b^*$
    else
      output $c \overset{R}{\leftarrow} \{0,1\}$
  else
    Halt.
```

Note that independent of the bit of our challenger $\mathrm{C\textsc{h}}_b$, we simulate a perfect challenger $\mathrm{SAC\textsc{h}}$ for sender anonymity. The choice for u_1 is completely ignored by $\mathrm{C\textsc{h}}_0$ and the choice for u_0 is ignored by $\mathrm{C\textsc{h}}_1$ as the adjacency function α_{SA} for sender anonymity does not modify its inputs.

We calculate the success probability for our adversary $\mathcal{A}$. Since the probability of success is independent of the bit b, we will show the calculation parametric in b. For readability we will use the following notation:

- $\mathcal{A}_b(u_b) := \mathcal{A}^{\mathrm{C\textsc{h}}_b(\mathcal{P},\alpha_{\mathrm{SA}})}$ the adversary $\mathcal{A}$ with oracle access to $\mathrm{C\textsc{h}}_b(\mathcal{P}, \alpha_{\mathrm{SA}})$ that chooses the value $u_b \overset{R}{\leftarrow} \mathcal{U}$ respectively.

- $\mathcal{A}^*_{u_b} := \mathcal{A}^{*\mathcal{A}_b(u_b)}$, The adversary $\mathcal{A}^*$ that has oracle access to the simulated challenger $\text{SACH}_{u_b}(\mathcal{P})$ as simulated by $\mathcal{A}$, where $u_b \overset{R}{\leftarrow} \mathcal{U}$ is chosen by $\mathcal{A}$.

$$\begin{aligned}
& Pr\left[b \leftarrow \mathcal{A}_b(u_b)\right] \\
=& Pr\left[b \leftarrow \mathcal{A}_b(u_b) | u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \cdot Pr\left[u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \\
+& Pr\left[b \leftarrow \mathcal{A}_b(u_b) | u^* \neq u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \cdot Pr\left[u^* \neq u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \\
\overset{(1)}{=}& 1 \cdot Pr\left[u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \\
+& \left(\frac{1}{2} \cdot \frac{N-2}{N-1}\right) \cdot Pr\left[u^* \neq u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \\
=& Pr\left[u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \\
+& \left(\frac{1}{2} \cdot \frac{N-2}{N-1}\right) \cdot \left(1 - Pr\left[u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right]\right) \\
=& Pr\left[u^* = u_b; u^* \leftarrow \mathcal{A}^*_{u_b}\right] \cdot \left(1 - \frac{1}{2} \cdot \frac{N-2}{N-1}\right) \\
+& \left(\frac{1}{2} \cdot \frac{N-2}{N-1}\right) \\
\overset{(2)}{>}& \left(\frac{1}{N} + \delta\right) \cdot \left(1 - \frac{1}{2} \cdot \frac{N-2}{N-1}\right) + \left(\frac{1}{2} \cdot \frac{N-2}{N-1}\right) \\
=& \left(\frac{1}{N} + \delta\right) \cdot \left(1 - \frac{1}{2} \cdot \left(1 - \frac{1}{N-1}\right)\right) + \left(\frac{1}{2} \cdot \left(1 - \frac{1}{N-1}\right)\right) \\
=& \left(\frac{1}{N} + \delta\right) \cdot \left(\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{N-1}\right) + \left(\frac{1}{2} - \frac{1}{2} \cdot \frac{1}{N-1}\right) \\
=& \frac{1}{2}\left(\left(\frac{1}{N} + \delta\right) \cdot \left(1 + \frac{1}{N-1}\right) + \left(1 - \frac{1}{N-1}\right)\right) \\
=& \frac{1}{2}\left(\frac{1}{N} + \frac{1}{N(N-1)} + \delta + \frac{\delta}{N-1} + 1 - \frac{1}{N-1}\right)
\end{aligned}$$

$$\begin{aligned}
=& \frac{1}{2}\left(\frac{N-1}{N(N-1)} + \frac{1}{N(N-1)} + \delta + \frac{\delta}{N-1} + 1 - \frac{N}{N(N-1)}\right) \\
=& \frac{1}{2}\left(\delta + \frac{\delta}{N-1} + 1\right) \\
=& \frac{1}{2} + \frac{1}{2}\delta + \frac{\delta}{2(N-1)} \\
\geq& \frac{1}{2} + \frac{1}{2}\delta
\end{aligned}$$

Equation (1) holds true since:

- If $\mathcal{A}^*$ guesses the correct $u^* = u_b$, $\mathcal{A}$ will always output the correct b.
- If $\mathcal{A}^*$ guesses a wrong u^*, $\mathcal{A}$ will output the correct b only if the coin toss $c \overset{R}{\leftarrow} \{0, 1\}$ matched b (which happens with probability $\frac{1}{2}$) and if additionally $\mathcal{A}^*$ did not guess $u^* = u_{1-b}$ by chance (which happens with probability $\frac{N-2}{N-1}$ independently of the behavior of $\mathcal{A}^*$, since u_{1-b} is drawn uniformly at random, but is discarded by $\text{CH}_b(\mathcal{P}, \alpha_{\text{SA}})$ and thus never actually used).

For equation (2) we make use of our assumption that $\mathcal{A}^*$ indeed wins against an honest challenger $\text{SACH}_u(\mathcal{P})$ with probability more than $\frac{1}{N} + \delta$.

Finally we can compute the difference:

$$\begin{aligned}
&Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\text{CH}_0(1^\eta,\mathcal{P},\alpha)}(1^\eta)\right] \\
-&Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\text{CH}_1(1^\eta,\mathcal{P},\alpha)}(1^\eta)\right] \\
=&Pr\left[0 \leftarrow \mathcal{A}_0(u_0)\right] - Pr\left[0 \leftarrow \mathcal{A}_1(u_1)\right] \\
=&Pr\left[0 \leftarrow \mathcal{A}_0(u_0)\right] - (1 - Pr\left[1 \leftarrow \mathcal{A}_1(u_1)\right]) \\
=&Pr\left[0 \leftarrow \mathcal{A}_0(u_0)\right] + Pr\left[1 \leftarrow \mathcal{A}_1(u_1)\right] - 1 \\
>&\frac{1}{2} + \frac{1}{2}\delta + \frac{1}{2} + \frac{1}{2}\delta - 1 = \delta,
\end{aligned}$$

Thus, $\mathcal{A}$ is a ppt-adversary that breaks $(0,\delta)$-α-IND-CDP for α_{SA} for $\mathcal{P}$. □

The otehr direction also holds, i.e. sender anonymity implies α-IND-CDP for α_{SA}, but only with a changed parameter

Lemma 9. *For all protocols $\mathcal{P}$ over a (finite) userspace $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has $\frac{\delta}{N}$-sender anonymity as in Definition 7 $\mathcal{P}$ also has* $(0,\delta)$-α-IND-CDP *for* α_{SA}.

Proof. Given an adversary $\mathcal{A}^*$ that breaks $(0,\delta)$-α-IND-CDP for α_{SA}, we construct an adversary against $\frac{\delta}{N}$-sender anonymity as follows:

```
$\mathcal{A}$ with oracle access to $\mathbf{SACh}_u(\mathcal{P})$
  Call $\mathcal{A}^*$ and receive two input tables $D_0, D_1$
  if $\alpha_{\text{SA}}(D_0, D_1) \neq \perp$ then
    Let $u_0, u_1$ be the users in the challenge-rows of $D_0, D_1$.
    Send $(\mathsf{input}, D')$ to the oracle $\text{SACH}_u(\mathcal{P})$, where $D'$ is $D_0$ without $u_0$ in the challenge-row.
    Send every message from the oracle to $\mathcal{A}^*$.
    Receive $b^* \leftarrow \mathcal{A}^*$
    Output $u_{b^*}$
  else
    Halt.
```

If by chance one of the users u_0, u_1 has been picked by SACH, $\mathcal{A}^*$ has an advantage. Otherwise, u_{b^*} will hit the correct uniformly drawn user with probability $\frac{1}{N-2}$.

We now compute the success probability of $\mathcal{A}$. For readability we use the following notation:

- $\mathcal{A}u_b := \mathcal{A}^{\mathcal{A}_b^*(u_b)}$, The adversary $\mathcal{A}$ that has oracle access to the challenger $\text{SACH}_{u_b}(\mathcal{P})$ where $u_b \overset{R}{\leftarrow} \mathcal{U}$ is chosen by this challenger.
- $\mathcal{A}_b^*(u_b) := \mathcal{A}^{*\text{CH}_b(\mathcal{P},\alpha_{\text{SA}})}$ the adversary $\mathcal{A}^*$ with oracle access to the simulated challenger $\text{CH}_b(\mathcal{P}, \alpha_{\text{SA}})$ that chooses the value $u_b \overset{R}{\leftarrow} \mathcal{U}$ respectively.

$$
\begin{aligned}
&Pr\left[u \leftarrow \mathcal{A}(u)\right]\\
=&Pr\left[u \leftarrow \mathcal{A}(u) | u = u_0\right] \cdot Pr\left[u = u_0\right]\\
+&Pr\left[u \leftarrow \mathcal{A}(u) | u = u_1\right] \cdot Pr\left[u = u_1\right]\\
+&Pr\left[u \leftarrow \mathcal{A}(u) | u \notin \{u_0, u_1\}\right] \cdot Pr\left[u \notin \{u_0, u_1\}\right]\\
=&Pr\left[0 \leftarrow \mathcal{A}_0^*(u_0)\right] \cdot Pr\left[u = u_0\right] + Pr\left[1 \leftarrow \mathcal{A}_1^*(u_1)\right] \cdot Pr\left[u = u_1\right]\\
+&Pr\left[u \leftarrow \mathcal{A}(u) | u \notin \{u_0, u_1\}\right] \cdot Pr\left[u \notin \{u_0, u_1\}\right]\\
=&Pr\left[0 \leftarrow \mathcal{A}_0^*(u_0)\right] \cdot Pr\left[u = u_0\right]\\
+&Pr\left[1 \leftarrow \mathcal{A}_1^*(u_1)\right] \cdot Pr\left[u = u_1\right] + 0\\
=&Pr\left[0 \leftarrow \mathcal{A}_0^*(u_0)\right] \cdot Pr\left[u = u_0\right]\\
+&(1 - Pr\left[0 \leftarrow \mathcal{A}_1^*(u_1)\right]) \cdot Pr\left[u = u_1\right]\\
=&\frac{1}{N} \cdot (Pr\left[0 \leftarrow \mathcal{A}_0^*(u_0)\right] - Pr\left[0 \leftarrow \mathcal{A}_1^*(u_1)\right] + 1)\\
>&\frac{1}{N} \cdot \delta + \frac{1}{N}
\end{aligned}
$$

□

We recall the definition of δ-sender unlinkability

Definition 10. *[δ-sender unlinkability] A protocol $\mathcal{P}$ with user space $\mathcal{U}$ has δ-*sender unlinkability *if for all ppt-adversaries $\mathcal{A}$*

$$\left| Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\mathrm{ULCH}_0(\mathcal{P})}\right] - Pr\left[b = 0 : b \leftarrow \mathcal{A}^{\mathrm{ULCH}_1(\mathcal{P})}\right] \right| \leq \delta$$

where the challenger ULCh *is as defined in Figure 6.*

Lemma 11. *[sender unlinkability] For all protocols $\mathcal{P}$ over a user space $\mathcal{U}$ it holds that if $\mathcal{P}$ has* $(0, \delta)$-α-IND-CDP *for α_{UL}, $\mathcal{P}$ also has δ-sender unlinkability as in Definition 10.*

Proof. We proof the lemma by contradiction. Given an adversary $\mathcal{A}^*$ that breaks δ-unlinkability, we will construct an adversary $\mathcal{A}$ that breaks $(0, \delta) - \alpha$-IND-CDP for α_{UL}. Let $\mathcal{A}^*$ be a ppt-adversary that wins against ULCh with probability more than δ. We define $\mathcal{A}$ as follows:

```
$\mathcal{A}$ with oracle access to Ch$_b(\mathcal{P}, \alpha_{\mathrm{UL}})$
  Call $\mathcal{A}^*$ and receive a input table $D$
  if in exactly 2 rows $i$ and $j$ in $x$ the user is missing then
    $u_0 \overset{R}{\leftarrow} \mathcal{U}$, $u_1 \overset{R}{\leftarrow} \mathcal{U} \setminus \{u_0\}$
    Let $D_0$ be $D$ with $u_0$ put in rows $i$ and $j$.
    Let $D_1$ be $D$ with $u_1$ put in rows $i$ and $j$.
    Send $(\mathsf{input}, D_0, D_1)$ to oracle Ch$_b(\mathcal{P}, \alpha_{\mathrm{UL}})$.
    Send every message from the oracle to $\mathcal{A}^*$.
    Upon receiving $b^* \leftarrow \mathcal{A}^*$, output $b^*$.
  else
    Halt.
```

The adjacency function α_{UL} changes the data sets such that D_0 contains only one of the users u_0, u_1, while in D_1 both will be present. Together with the challenger Ch$_b(\mathcal{P}, \alpha_{\mathrm{UL}})$ we simulate a challenger ULCh perfectly.

Whenever b^* is the correct answer for the (simulated) challenger ULCh, it also is the correct answer for $\text{Ch}_b(\mathcal{P}, \alpha_{\text{UL}})$.

Since by assumption $\mathcal{A}^*$ breaks δ-sender unlinkability with probability more than δ, $\mathcal{A}$ breaks $(0, \delta)$-α-IND-CDP. □

Lemma 12. *For all protocols $\mathcal{P}$ over user space $\mathcal{U}$ of size N it holds that if $\mathcal{P}$ has $\frac{1}{N(N-1)}\delta$-sender unlinkability as in Definition 10, $\mathcal{P}$ also has $(0,\delta)$-α-IND-CDP for α_{UL}.*

Proof. Assume $\mathcal{P}$ provides $\frac{\delta}{N(N-1)}$-sender unlinkability and assume we have an adversary $\mathcal{A}^*$ which breaks $(0,\delta)$-α-IND-CDP. We construct an adversary $\mathcal{A}$ for $\frac{\delta}{N(N-1)}$-sender unlinkability.

```
**$\mathcal{A}$ with oracle access to $\text{ULCh}_b(\mathcal{P}, \alpha_{\text{UL}})$**
  Call $\mathcal{A}^*$ and receive two input tables $D_0, D_1$
  **if** $\alpha_{\text{UL}}(D_0, D_1) \neq \perp$ **then**
    Let $u_0, u_1$ be the users in the challenge-rows of $D_0, D_1$.
    Send $(\mathsf{input}, D')$ to oracle,where $D'$ is $D_0$ with no users in the challenge rows.
    Send every message from the oracle to $\mathcal{A}^*$.
    Receive $b^* \leftarrow \mathcal{A}^*$
    Output $b^*$
  **else**
    Halt.
```

We directly compute the difference $|Pr[A^{\text{ULCh}_0} = 0] - Pr[A^{\text{ULCh}_1} = 0]|$. Let u and u' be the users chosen by ULCh. The first part then computes to

$$\begin{aligned}
&Pr[A^{\text{ULCh}_0} = 0] \\
=&Pr[A^{\text{ULCh}_0} = 0 | u \in \{u_0, u_1\}] \cdot Pr[u \in \{u_0, u_1\}] \\
&+ Pr[A^{\text{ULCh}_0} = 0 | u \notin \{u_0, u_1\}] \cdot Pr[u \notin \{u_0, u_1\}] \\
=&Pr[A^{*\text{Ch}_0(\alpha_{\text{UL}})} = 0] \cdot \frac{2}{N} \\
&+ Pr[A^{*\text{ULCh}_0} = 0 | u \notin \{u_0, u_1\}] \cdot (1 - \frac{2}{N})
\end{aligned}$$

and $Pr[A^{\text{ULCh}_1} = 0]$ computes to

$$\begin{aligned}
&Pr[A^{\text{ULCh}_1} = 0] \\
=&Pr[A^{\text{ULCh}_1} = 0 | \{u, u'\} = \{u_0, u_1\}] \\
&\cdot Pr[\{u, u'\} = \{u_0, u_1\}] \\
&+ Pr[A^{\text{ULCh}_1} = 0 | \{u, u'\} \neq \{u_0, u_1\}] \\
&\cdot Pr[\{u, u'\} \neq \{u_0, u_1\}] \\
=&Pr[A^{*\text{Ch}_1(\alpha_{\text{UL}})} = 0] \cdot \frac{2}{N(N-1)} \\
&+ Pr[A^{*\text{ULCh}_1} = 0 | \{u, u'\} \neq \{u_0, u_1\}] \cdot (1 - \frac{2}{N(N-1)})
\end{aligned}$$

For readability we use following place holders for the different events

- $M_0 : A^{*\text{Ch}_0(\alpha_{\text{UL}})} = 0$

- $M_1 : A^{*\textsc{Ch}_1(\alpha_{\mathrm{UL}})} = 0$
- $L_0 : A^{*\mathrm{ULC}\textsc{h}_0} = 0 \mid u \neq \{u_0, u_1\}$
- $L_1 : A^{*\mathrm{ULC}\textsc{h}_1} = 0 \mid \{u, u'\} \neq \{u_0, u_1\}$

With this we get by suitably adding 0 several times

$$\begin{aligned}
&|Pr[A^{\mathrm{ULC}\textsc{h}_0} = 0] - Pr[A^{\mathrm{ULC}\textsc{h}_1} = 0]| \\
=&|\frac{2}{N(N-1)}(Pr[M_0] - Pr[M_1]) \\
&+ (\frac{2}{N} - \frac{2}{N(N-1)})Pr[M_0] \\
&- (\frac{2}{N} - \frac{2}{N(N-1)})Pr[L_0] \\
&+ (1 - \frac{2}{N(N-1)})(Pr[L_0] - Pr[L_1])|
\end{aligned}$$

Here we make following assumptions

- $Pr[M_0] \geq Pr[L_0]$: As $\mathcal{A}^*$ is an adversary especially constructed for $\textsc{Ch}_b(\alpha_{\mathrm{UL}})$ we can assume that he also works better against this adversary. Otherwise we can construct an even better adversary $\mathcal{A}^{*\prime}$ which makes the same decisions as $\mathcal{A}^*$ with oracle $\mathrm{ULC}\textsc{h}_b$ and use him instead of $\mathcal{A}^*$.
- $-\frac{1}{N(N-1)} \leq Pr[L_0] - Pr[L_1] \leq \frac{1}{N(N-1)}$, as otherwise $\mathcal{A}^*$ already is an adversary which breaks δ-unlinkability, which is a contradiction.

Together with our initial assumption, that $\mathcal{A}^*$ breaks (ε, δ)- α-IND-CDP for α_{UL}, we get

$$\begin{aligned}
&|\frac{2}{N(N-1)}(Pr[M_0] - Pr[M_1]) \\
&+ (\frac{2}{N} - \frac{2}{N(N-1)})Pr[M_0] \\
&- (\frac{2}{N} - \frac{2}{N(N-1)})Pr[L_0] \\
&+ (1 - \frac{2}{N(N-1)})(Pr[L_0] - Pr[L_1])| \\
>&\frac{2}{N(N-1)}\delta - (1 - \frac{2}{N(N-1)})\frac{\delta}{N(N-1)} \\
>&\frac{\delta}{N(N-1)}
\end{aligned}$$

as required. □

A.2 Relations among the various notions

In this section we explore the relations among our notions of *sender anonymity*, *sender unlinkability* and relationship anonymity.

Below follow the lemmas and their proofs for the various relations between the anonymity notions, as visualized in Fig. 7.

Lemma 13. *[Sender anonymity implies relationship anonymity.] If a protocol $\mathcal{P}$ has $(0, \delta)$-α-IND-CDP for α_{SA}, is also has $(0, \delta)$-α-IND-CDP for α_{Rel}.*

Proof. Given an adversary $\mathcal{A}^*$ against $(0, \delta)$-α-IND-CDP for α_{Rel}, we construct an adversary against $(0, \delta)$-α-IND-CDP for α_{SA} as follows: $\mathcal{A}$ **with oracle access to** $\mathbf{Ch}_b(\mathcal{P}, \alpha_{\text{SA}})$

```
Call $\mathcal{A}^*$ and receive input tables $D_0, D_1$.
Let $b \overset{R}{\leftarrow} \{0, 1\}$.
Replace $R_c = (r, \mathsf{aux})^k$ in the challenge-row of $D_b$ by $R_c$ in $D_{1-b}$.
Send $(\mathsf{input}, D_0, D_1)$ to oracle.
Send every message from the oracle to $\mathcal{A}^*$.
Upon receiving $b^*$ from $\mathcal{A}^*$, output $b^*$.
```

The adversary $\mathcal{A}$ perfectly simulates a challenger against relationship anonymity. However, since he chooses R_c to be the same before sending the input tables to its oracle, it also fits the requirements for sender anonymity. Whenever $\mathcal{A}^*$ wins the relationship anonymity game, $\mathcal{A}$ wins the sender anonymity game. □

Lemma 14. *[Sender anonymity implies sender unlinkability] If a protocol $\mathcal{P}$ has $(0, \delta)$-α-IND-CDP for α_{SA}, $\mathcal{P}$ also has $(0, \delta)$-α-IND-CDP for α_{UL}.*

Proof. Given an adversary $\mathcal{A}^*$ against $(0, \delta)$-α-IND-CDP for α_{UL}, we construct an adversary against $(0, \delta)$-α-IND-CDP for α_{SA} as follows: $\mathcal{A}$ **with oracle access to** $\mathbf{Ch}_b(\mathcal{P}, \alpha_{\text{SA}})$

```
Call $\mathcal{A}^*$ and receive input tables $D_0, D_1$.
Let $(D'_0, D'_1) \leftarrow \alpha_{\text{UL}}$.
Send $(\mathsf{input}, D'_0, D'_1)$ to oracle.
Send every message from the oracle to $\mathcal{A}^*$.
Upon receiving $b^*$ from $\mathcal{A}^*$, output $b^*$.
```

The adversary $\mathcal{A}$ perfectly simulates a challenger against sender unlinkability (α_{SA} does not change the input tables). However, since exactly one challenge row exists in (D'_0, D'_1) it also fits the requirements for sender anonymity. Whenever $\mathcal{A}^*$ wins the relationship anonymity game, $\mathcal{A}$ wins the sender anonymity game. □

Lemma 15. *[Sender unlinkability does not imply sender anonymity] If a protocol $\mathcal{P}$ has $(0, \delta)$-α-IND-CDP for α_{UL}, $\mathcal{P}$ does not necessarily have $(0, \delta')$-α-IND-CDP for α_{SA} for any $\delta' < 1$.*

Proof. Consider the following counterexample protocol $\mathcal{P}$. It processes the input table row by row, perfectly hiding all information about the user and broadcasting the message. At any point, an adversary might ask it to reveal the user of a given message, but only once. Obviously this protocol does not have sender anonymity, as the adversary can deanonymize any user. For unlinkability, however, deanonymizing the user of one *single message* does not suffice. □

Lemma 16. *[relationship anonymity does not imply sender anonymity] If a protocol $\mathcal{P}$ has $(0, \delta)$-α-IND-CDP for α_{Rel}, $\mathcal{P}$ does not necessarily have $(0, \delta')$-α-IND-CDP for α_{SA} for any $\delta' < 1$.*

Proof. Consider the following counterexample protocol $\mathcal{P}$. It processes the input table row by row, perfectly hiding all information about the message and possible recipients, but leaking the users that send messages. An adversary can easily distinguish input tables, where e.g. a user only is present in D_1 but not in D_0, but for breaking relationship anonymity, information about the messages and/or recipients has to be present. □

A.3 Leveraging UC

In this section we derive the proof for Lemma 22. We first give the required definitions from the UC-framework and then give the proof.

Definition 25 (Indistinguishability [Can01])**.** *Two binary distribution ensembles X and Y are indistinguishable, denoted $X \approx Y$, if for every $c \in \mathbb{N}$ there is a $\eta_0 \in \mathbb{N}$ such that for all $\eta > \eta_0$ and all x we have that*

$$|Pr[X(\eta, x)] = 1 - Pr[Y(\eta, x)] = 1| < \delta' = \eta^{-c}$$

The real world. For the process in the real world we introduce the random variable $Real_{\Pi,\mathcal{A},D}(\eta, x)$ which captures the interaction of a protocol Π with an adversary $\mathcal{A}$, observed by a distinguisher $D.Real_{\Pi,\mathcal{A},D}$ will denote the ensemble of all those distributions.

The ideal world. Similarly, we introduce the random variable $Ideal_{\mathcal{F},S,D}(\eta, x)$ which captures the interaction of an ideal functionality $\mathcal{F}$, a simulator S and the distinguisher. $Ideal_{\mathcal{F},S,D}$ will again denote the ensemble of such random variables.

Definition 26 (Realization in UC)**.** *A protocol* Π *UC-realizes an ideal functionality* F *if for every* PPT *adversary* $\mathcal{A}$ *of* Π *there exists a* PPT *simulator* S *such that for every* PPT *distinguisher* D *it holds that*

$$Real_{\Pi,\mathcal{A},D} \approx Ideal_{\mathcal{F},S,D}$$

In order to stay consistent with the notation used in the main body of the paper, but still catch all technical details of UC, we adopt following notation: The adversary we used in AnoA in order to define α-IND-CDP is now part of the environment Env. We capture the interaction of adversary – as denoted above, and protocol by taking both as arguments into the challenger Ch, i.e. we write $\text{Ch}_b(\Pi, \mathcal{A}, \alpha, n)$ instead of just $\text{Ch}_b(\Pi, \alpha, n)$.

Given the realization of a (ϵ, δ)-differentially-private ideal functionality by a protocol Π, we get differential privacy for Π.

Lemma 22. *Let* $\mathcal{F}$ *be* (ϵ, δ)-α-IND-CDP *and* Π *be a protocol. If* Π *UC-realizes* $\mathcal{F}$ *then* Π *is* (ϵ, Δ)-α-IND-CDP *with* $\Delta = \delta + \delta'$ *for some negligible value* δ'.

Proof. Given an (ϵ, δ)-α-IND-CDP functionality $\mathcal{F}$, assume Π UC-realizes $\mathcal{F}$, but Π is not (ϵ, Δ)-α-IND-CDP, i.e. there exist an adversary $\mathcal{A}$ s.t.

$$\begin{aligned} &Pr[b = 0 : b \leftarrow \text{Env}^{\text{Ch}_0(\Pi,\mathcal{A},\alpha,1)}] \\ &> e^{\epsilon} Pr[b = 0 : b \leftarrow \text{Env}^{\text{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] + \Delta \end{aligned}$$

where $\Delta \geq \delta + \delta'$ for a non negligible value δ'.We construct the following PPT distinguisher D that uses $\mathcal{A}$ in order to separate Π from $\mathcal{F}$:

1. choose $b \overset{R}{\leftarrow} \{0, 1\}$ uniformly at random
2. Simulate the challenger Ch on b
3. depending on the output b^* of the environment:
   (a) if the environment returns $b^* = b$, decide that you observed $(\Pi, \mathcal{A})$ and output 1

(b) otherwise decide that you observed $(\mathcal{F}, S)$ and output 0

We now bound the probabilities $\Pr[Real_{\Pi,\mathcal{A},D}(\eta) = 1]$ and $\Pr[Ideal_{\mathcal{F},S,D}(\eta) = 1]$ as required for Lemma 22. Using the assumption that Π is not (ϵ, Δ)-differentially private, the first expression computes to

$$\begin{aligned}
&\Pr[Real_{\Pi,A,D}(\eta) = 1] \\
=&\Pr[b = b^* : b \leftarrow \textsc{Env}^{\textsc{Ch}_{b^*}(\Pi,\mathcal{A},\alpha,1)}] \\
=&\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] \cdot \Pr[b = 1 : b \xleftarrow{R} \{0,1\}] \\
&+ \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_0(\Pi,\mathcal{A},\alpha,1)}] \cdot \Pr[b = 0 : b \xleftarrow{R} \{0,1\}] \\
=&\frac{1}{2}\Big(\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] \\
&+ \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_0(\Pi,\mathcal{A},\alpha,1)}]\Big) \\
>&\frac{1}{2}\Big(\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] \\
&+ e^{\varepsilon} \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,A,\alpha)}] + \Delta\Big) \\
=&\frac{1}{2}\Big(1 - \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] \\
&+ e^{\varepsilon} \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,A,\alpha)}] + \Delta\Big) \\
=&\frac{1}{2}\Big((e^{\varepsilon} - 1) \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\Pi,\mathcal{A},\alpha,1)}] + \Delta + 1\Big)
\end{aligned} \tag{2}$$

Using the (ϵ, δ)-differential privacy of $\mathcal{F}$, the second expression can be bound as follows

$$\begin{aligned}
&\Pr[Ideal_{\mathcal{F},S,D}(\eta) = 1] \\
=&\Pr[b = b^* : b \leftarrow \textsc{Env}^{\textsc{Ch}_{b^*}(\mathcal{F},S,\alpha,1)}] \\
=&\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] \cdot \Pr[b = 1 : b \xleftarrow{R} \{0,1\}] \\
&+ \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_0(\mathcal{F},S,\alpha,1)}] \cdot \Pr[b = 0 : b \xleftarrow{R} \{0,1\}] \\
=&\frac{1}{2}\Big(\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] \\
&+ \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_0(\mathcal{F},S,\alpha,1)}]\Big) \\
\leq&\frac{1}{2}\Big(\Pr[b = 1 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] \\
&+ e^{\varepsilon} \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] + \delta\Big) \\
=&\frac{1}{2}\Big(1 - \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] \\
&+ e^{\varepsilon} \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] + \delta\Big) \\
=&\frac{1}{2}\Big(1 + (e^{\varepsilon} - 1) \Pr[b = 0 : b \leftarrow \textsc{Env}^{\textsc{Ch}_1(\mathcal{F},S,\alpha,1)}] + \delta\Big)
\end{aligned} \tag{3}$$

Putting Equations 2 and 3 together, we get

$$\begin{aligned}
&\Pr[Real_{\Pi,A,D}(\eta) = 1] \\
&\quad - \Pr[Ideal_{\mathcal{F},S,D}(\eta) = 1] \\
>&\frac{1}{2}\Big((e^{\varepsilon} - 1)(\Pr[b = 0 : b \leftarrow \mathcal{A}^{\textsc{Ch}_1(\Pi,\alpha,1)}] \\
&- \Pr[b = 0 : b \leftarrow S^{\textsc{Ch}_1(\mathcal{F},\alpha,1)}]) + \Delta - \delta\Big) \\
=&\frac{1}{2}\Big((e^{\varepsilon} - 1)(\Pr[b = 0 : b \leftarrow \mathcal{A}^{\textsc{Ch}_1(\Pi,\alpha,1)}] \\
&- \Pr[b = 0 : b \leftarrow S^{\textsc{Ch}_1(\mathcal{F},\alpha,1)}]) + \delta'\Big)
\end{aligned} \tag{4}$$

By assumption, Π UC-realizes $\mathcal{F}$. Hence $\Pr[b = 0 : b \leftarrow \mathcal{A}^{\textsc{Ch}_1(\Pi,\alpha,1)}] - \Pr[b = 0 : b \leftarrow S^{\textsc{Ch}_1(\mathcal{F},\alpha,1)}]$ is negligible. As δ' is not negligible, the difference 4 stays non-negligible, and positive. Hence we get that

$$\begin{aligned}
&|\Pr[Real_{\Pi,A,D}(\eta, (x_0, x_1)) = 1] \\
&\qquad - \Pr[Ideal_{\mathcal{F},S,D}(\eta, (x_0, x_1)) = 1]| \\
&> \delta''
\end{aligned}$$

for some non negligible value δ'', contradicting the UC-realization of $\mathcal{F}$ by Π (Def. 26). Therefore our initial assumption is wrong and Π is (ϵ, Δ)-α-IND-CDP. □

A.4 Composability theorem

In this section, we present the full proof for our composability theorem: for all adversary classes that are composable (for an anonymity notion α) it suffices to show anonymity against single challenges. The theorem then allows for deriving guarantees for more than one challenge

Theorem 21. *For every protocol $\mathcal{P}$, every anonymity function α, every $n \in \mathsf{N}$ and every adversary class* A *that is composable. Whenever $\mathcal{P}$ is $(1, \epsilon, \delta)$-α-IND-CDP for* A*, with $\varepsilon \geq 0$ and $0 \leq \delta \leq 1$, then $\mathcal{P}$ is $(n, n \cdot \epsilon, n \cdot e^{n\epsilon} \cdot \delta)$-$\alpha$-IND-CDP for* A.

Proof. We will show this theorem inductively. Assume that $\mathcal{P}$ is $(i, i \cdot \epsilon, e^{i\varepsilon} \cdot i \cdot \delta)$-$\alpha$-IND-CDP. We show that $\mathcal{P}$ is also $(i + 1, (i + 1) \cdot \epsilon, e^{(i+1)\cdot\epsilon}(i + 1) \cdot \delta)$-$\alpha$-IND-CDP.

Let $\mathcal{A}$ be an adversary that sends at most $i + 1$ challenges. To do so, we construct several games:

- **Game:** G_0 is the normal game $\text{ACR}\textsc{eal}(0, i + 1)$ with up to $i + 1$ challenges where $b = 0$.
- **Game:** G_1 is an intermediate game $\text{ACS}\textsc{im}_{\text{M}_{z_{\mathsf{dontsim}}}}(0, i + 1)$. Here every message from $\mathcal{A}$ to A (and otherwise) goes through the simulator $\text{M}_{z_{\mathsf{dontsim}}}$. However, this simulator does not need to simulate anything, as there are still up to $i + 1$ challenges and $b = 0$.

  Claim: G_0 and G_1 are computationally indistinguishable.

  Proof: By item 3b Definition 20 the simulator $\text{M}_{z_{\mathsf{dontsim}}}$ exists and the games are indistinguishable.
- **Game:** G_2 is an intermediate (hybrid) game $\text{ACS}\textsc{im}_{\text{M}_z}(0, i + 1)$ with $b = 0$ and fixed input messages instead of the challenge with tag $i + 1$ (so there are at most i challenges left). This

is done by using the simulator M_z for $z = [(\mathsf{dontsim}, \_), \ldots, (\mathsf{dontsim}, \_), (\mathsf{sim}, 0)] \in \{0,1\}^{i+1}$, i.e., the simulator simulates the $i+1$st challenge for $b = 0$.

Claim: G_1 and G_2 are computationally indistinguishable.

Proof: By item 3 of Definition 20, we know that the simulator M_z exists. Since the simulator M_z from G_2 uses the correct bit $b_{i+1} = 0$ for the simulated challenge, Item 3c of Definition 20 implies that the games are indistinguishable.

- **Game:** G_3 is the intermediate (hybrid) game $\text{ACS}\textsc{im}_{\mathrm{M}_z}(1, i+1)$ where the simulator stays M_z but the challenger changes to $b = 1$.

  Claim: G_2 and G_3 are $(i\varepsilon, e^{i\varepsilon} i\delta)$-indistinguishable.

  Proof: The adversary $\mathrm{M}_z(\mathcal{A})$ makes at most i queries with challenge tags in $\{1, \ldots, i\}$. From the reliability property of the adversary class (item 1 of Definition 20) we know that thus $\mathrm{A}(\mathrm{M}_z(\mathcal{A}))$ uses at most i challenge tags in $\{1, \ldots, i\}$. The claim immediately follows from the induction hypothesis: $\mathcal{P}$ is $(i, i \cdot \epsilon, i \cdot \delta)$-$\alpha$-IND-CDP.

- **Game:** G_4 is a game $\text{ACS}\textsc{im}_{\mathrm{M}_{z'}}(1, i+1)$ where the simulator $\mathrm{M}_{z'}$ with $z' = [(\mathsf{sim}, 1), \ldots, (\mathsf{sim}, 1), (\mathsf{sim}, 0)]$ simulates all challenges from $\mathcal{A}$. For the challenge tags 1 to i, $\mathrm{M}_{z'}$ simulates the challenges for $b_1 = \ldots = b_i = 1$, whereas for the tag $i+1$ it still simulates it for $b_{i+1} = 0$. The challenger uses $b = 1$.

  Claim: G_3 and G_4 are computationally indistinguishable.

  Proof: Since the simulator $\mathrm{M}_{z'}$ from G_4 uses the correct bit $b_1 = \ldots = b_i = 1$ for the challenges that are not simulated in M_z, Item 3c of Definition 20 implies that the games are indistinguishable.

- **Game:** G_5 is the game $\text{ACS}\textsc{im}_{\mathrm{M}_{z'}}(0, i+1)$ where we use the same simulator $\mathrm{M}_{z'}$ but we have $b = 0$ again.

  Claim: G_4 and G_5 are computationally indistinguishable.

  Proof: Since there are no challenge messages (everything is simulated, as by item 3a $\mathrm{M}_{z'}$ does not send any messages $(\mathsf{challenge}, \_, \_, \Psi)$), changing the bit b of the challenger does not have any effect. Hence, the games are indistinguishable.

- **Game:** G_6 is the game $\text{ACS}\textsc{im}_{\mathrm{M}_{z''}}(0, i+1)$ where we use the simulator $\mathrm{M}_{z''}$ with $z'' = [(\mathsf{sim}, 1), \ldots, (\mathsf{sim}, 1), (\mathsf{dontsim}, \_)]$. In other words, we do not simulate the challenge for $i+1$ with $b_{i+1} = 0$, but we use the challenger again (also with $b = 0$).

  Claim: G_5 and G_6 are computationally indistinguishable.

  Proof: Since the simulator $\mathrm{M}_{z'}$ from G_5 uses the correct bit $b_{i+1} = 0$ for the simulated challenge (which the simulator $\mathrm{M}_{z''}$ does not simulate), Item 3c of Definition 20 implies that the games are indistinguishable.

- **Game:** G_7 is $\text{ACS}\textsc{im}_{\mathit{translator}(\mathrm{M}_{z''})}(0, i+1)$ where we build around the simulator $\mathrm{M}_{z''}$ we an interface $\mathit{translator}(\cdot)$ that translates the challenge tag from $i+1$ to 1 and vice versa in all messages $(\mathsf{challenge}, \_, \_, \Psi)$ from $\mathrm{M}_{z''}$ to A and in all messages $(\mathsf{answer\ for}, \_, \Psi)$ from A to $\mathrm{M}_{z''}$..

  Claim: G_6 and G_7 are information theoretically indistinguishable.

  Proof: Item 2 of Definition 20 requires that the renaming of challenge tags does not influence the behavior of A. It also does not influence the behavior of the challenger (by definition) or the protocol (that never sees challenge tags). Thus, the games are indistinguishable.

- **Game:** G_8 is the game $\text{ACS}\textsc{im}_{\mathit{translator}(\mathrm{M}_{z''})}(1, i+1)$ where the simulator is defined as in G_7 but $b = 1$.

  Claim: G_7 and G_8 are (ε, δ) indistinguishable.

  Proof: By assumption of the theorem, the protocol $\mathcal{P}$ is $(1, \epsilon, \delta)$-α-IND-CDP for A. Moreover, by definition of z'' and by item 3a, the adversary $\mathit{translator}(\mathrm{M}_{z''}(\mathcal{A}))$ only uses at most one challenge tag, namely the tag 1. From the reliability property of the adversary class (item 1 of Definition 20) we know that thus $\mathrm{A}(\mathit{translator}(\mathrm{M}_{z''}(\mathcal{A})))$ uses only the challenge tag 1. Thus, G_7 and G_8 are (ε, δ) indistinguishable.

- **Game:** G_9 is $\text{ACS}\textsc{im}_{\mathrm{M}_{z''}}(1, i+1)$ where we remove the translation interface again.

  Claim: G_8 and G_9 are information theoretically indistinguishable.

  Proof: As before, Item 2 of Definition 20 requires that the renaming of challenge tags does not influence the behavior of A. It also does not influence the behavior of the challenger (by definition) or the protocol (that never sees challenge tags). Thus, the games are indistinguishable.

- **Game:** G_{10} is the normal game $\text{AC}\textsc{Real}(1, i+1)$ where $b = 1$.

  Claim: G_9 and G_{10} are computationally indistinguishable.

  Proof: Since $\mathrm{M}_{z''}$ uses the correct bit $b_1 = \ldots = b_i = 1$ for all simulations, we can replace it with $\mathrm{M}_{z_{\mathsf{dontsim}}}$, that, in turn, is indistinguishable from $\text{AC}\textsc{Real}(1, i+1)$.

We slightly abuse notation in writing $\Pr[0 = \mathcal{A}(G_0)]$ for $\Pr[0 = \langle \mathrm{A}(\mathcal{A}(n)) || \textsc{Ch}(\mathcal{P}, \alpha, n, 0)\rangle]$, $\Pr[0 = \mathcal{A}(G_1)]$ for $\Pr[0 = \langle \mathrm{A}(\mathrm{M}_z(b, \mathcal{A}(n))) || \textsc{Ch}(\mathcal{P}, \alpha, n, 0)\rangle]$, etc..

$$\begin{aligned}
&\Pr[0 = \mathcal{A}(G_0)] \\
&\le \Pr[0 = \mathcal{A}(G_1)] + \mu_1 \\
&\le \Pr[0 = \mathcal{A}(G_2)] + \mu_2 + \mu_1 \\
&\le e^{i\varepsilon} \Pr[0 = \mathcal{A}(G_3)] + e^{i\varepsilon} i\delta + \mu_2 + \mu_1 \\
&\le e^{i\varepsilon} \Pr[0 = \mathcal{A}(G_4)] + e^{i\varepsilon}(\mu_3 + i\delta) + \mu_2 + \mu_1 \\
&\le e^{i\varepsilon} \Pr[0 = \mathcal{A}(G_5)] + e^{i\varepsilon}(\mu_4 + \mu_3 + i\delta) + \mu_2 + \mu_1 \\
&\le e^{i\varepsilon} \Pr[0 = \mathcal{A}(G_6)] + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + i\delta) + \mu_2 + \mu_1 \\
&= e^{i\varepsilon} \Pr[0 = \mathcal{A}(G_7)] + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + i\delta) + \mu_2 + \mu_1 \\
&\le e^{i\varepsilon}(e^{\varepsilon} \Pr[0 = \mathcal{A}(G_8)] + \delta) + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + i\delta) + \mu_2 + \mu_1 \\
&= e^{(i+1)\varepsilon} \Pr[0 = \mathcal{A}(G_8)] + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + (i+1)\delta) + \mu_2 + \mu_1 \\
&= e^{(i+1)\varepsilon} \Pr[0 = \mathcal{A}(G_9)] + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + (i+1)\delta) + \mu_2 + \mu_1 \\
&\le e^{(i+1)\varepsilon} \Pr[0 = \mathcal{A}(G_{10})] + e^{(i+1)\varepsilon}\mu_6 + e^{i\varepsilon}(\mu_5 + \mu_4 + \mu_3 + (i+1)\delta) + \mu_2 + \mu_1 \\
&\le e^{(i+1)\varepsilon} \Pr[0 = \mathcal{A}(G_{10})] + e^{(i+1)\varepsilon}(i+1)\delta
\end{aligned}$$

□

B Abstracting Tor in UC

We cite the the description of the ideal functionality $\mathcal{F}_{\text{OR}}$. Sections B.1 and B.2 are taken from [BGKM12].

B.1 System and adversary model

We consider a fully connected network of $n + m$ parties $\mathsf{N} = \{P_1, \ldots, P_n, \ldots, P_{n+1}, \ldots, P_{n+m}\}$. We consider the parties $P_1, \ldots, P_n$ to be OR nodes, and the parties $P_{n+1}, \ldots, P_{n+m}$ to only be users. We furthermore assume that the set of OR nodes is publicly known. The onion routers can be compromised by the adversary by sending compromise messages. The users, however, can in our model not be compromised, since the adversary can just act as a user of the OR network. Formally, $P_{n+1}, \ldots, P_{n+m}$, consequently, do not react towards compromise messages.

Tor has not been designed to resist against global adversarys. Such an adversary is too strong for many practical purposes as it can simply break the anonymity of an OR protocol by holding back all but one onion and tracing that one onion though the network. However, in contrast to previous work, we do not only consider local adversarys, which do not control more than the compromised OR routers, but also partially global adversarys that control a certain portion of the network. Analogous to the network functionality $\mathcal{F}_{\text{SYN}}$ proposed by Canetti [Can01], we model the network as an ideal functionality $\mathcal{F}_{\text{NET}}$, which bounds the number of adversary-controlled links to $q \in [0, \binom{n}{2}]$. For adversary-controlled links the messages are forwarded to the adversary; otherwise, they are directly delivered.

Let S represent all possible destination servers $\{S_1, \ldots, S_\Delta\}$ which reside in the network abstracted by a network functionality $\mathcal{F}_{\text{NET}^q}$.

We stress that the UC framework does not provide a notion of time; hence, the analysis of timing attacks, such as traffic analysis, is not in the scope of this work.

Adaptive Corruptions. Forward secrecy [DvOW92] is an important property for onion routing. In order to analyze this property, we allow adaptive corruptions of nodes by the adversary $\mathcal{A}$. Such an adaptive corruption is formalized by a message compromise, which is sent to the respective party. Upon such a compromise message the internal state of that party is deleted and a long-term secret key *sk* for the node is revealed to the adversary. $\mathcal{A}$ can then impersonate the node in the future; however, $\mathcal{A}$ cannot obtain the information about its ongoing sessions. We note that this restriction arises due to the currently available security proof techniques and the well-known selective opening problem with symmetric encryptions [Hof11], and the restriction is not specific to our constructions [BMP00, GL01]. We could also restrict ourselves to a static adversary as in previous work [CL05]; however, that would make an analysis of forward secrecy impossible.

B.2 Ideal functionality

The presentation of the ideal functionality $\mathcal{F}_{\text{OR}}$ is along the lines of the description OR protocol Π_{OR} from Section [BGKM12, Section 2.4]. We continue to use the message-based state transitions from Π_{OR}, and consider sub-machines for all n nodes in the ideal functionality. To communicate with each other through messages and data structures, these sub-machines share a memory space in the functionality. The sub-machine pseudocode for the ideal functionality appears in Figure 13 and three subroutines are defined in Figure 14. As the similarity between pseudocodes for the OR protocol and the ideal functionality is obvious, rather than explaining the OR message flows again, we concentrate on the differences.

The only major difference between Π_{OR} and $\mathcal{F}_{\text{OR}}$ is that cryptographic primitives such as message wrapping, unwrapping, and key exchange are absent in the ideal world; we do not have any keys in $\mathcal{F}_{\text{OR}}$, and the OR messages *WrOn* and *UnwrOn* as well as the 1W-AKE messages *Initiate*, *Respond*, and *ComputeKey* are absent.

The ideal functionality also abstracts the directory server and expects on the input/output

```
upon an input (setup):
  draw a fresh handle $h$; a set $\mathsf{registered\_flag} \leftarrow \mathit{true}$
  store $\mathit{lookup}(h) \leftarrow (\mathsf{dir}, \mathsf{registered}, \mathcal{N})$
  send $(h, \mathsf{register}, P)$ to $\mathcal{A}$
  wait for a msg $(\mathsf{dir}, \mathsf{registered}, \mathcal{N})$ via a handle
  output $(\mathsf{ready}, (P_j)_{j=1}^n) = (\mathsf{ready}, \mathcal{N})$
upon an input $(\mathsf{createcircuit}, \mathcal{P} = \langle P, P_1, \ldots, P_\ell \rangle)$:
  store $\mathcal{P}$ and $\mathcal{C} \leftarrow \langle P \rangle$; $\mathit{ExtendCircuit}(\mathcal{P}, \mathcal{C})$
upon an input $(\mathsf{send}, \mathcal{C} = \langle P \overset{cid_1}{\Longleftrightarrow} P_1 \Longleftrightarrow \cdots P_\ell \rangle, m)$:
  if $\mathit{Used}(cid_1) < ttl_C$ then
      $\mathit{Used}(cid_1)$++; $\mathit{SendMessage}(P_1, cid_1, \mathsf{relay}, \langle \mathsf{data}, m \rangle)$
  else
      $\mathit{DestroyCircuit}(\mathcal{C}, cid_1)$; output $(\mathsf{destroyed}, \mathcal{C}, m)$
upon receiving a handle $\langle P, P_{next}, h \rangle$ from $\mathcal{F}_{\text{NET}}$:
  send $(msg) \leftarrow \mathit{lookup}(h)$ to a receiving submachine $P_{next}$
upon receiving a msg $(P_i, cid, \mathsf{create})$ through a handle:
  store $\mathcal{C} \leftarrow \langle P_i \overset{cid}{\Longleftrightarrow} P \rangle$; $\mathit{SendMessage}(P_i, cid, \mathsf{created})$
upon receiving a msg $(P_i, cid, \mathsf{created})$ through a handle:
  if $\mathit{prev}(cid) = (P', cid')$ then
      $\mathit{SendMessage}(P', cid', \mathsf{relay}, \mathsf{extended})$
  else if $\mathit{prev}(cid) = \bot$ then
      $\mathit{ExtendCircuit}(\mathcal{P}, \mathcal{C})$
upon receiving a msg $(P_i, cid, \mathsf{relay}, O)$ through a handle:
  if $\mathit{prev}(cid) = \bot$ then
      if $\mathit{next}(cid) = \bot$ then
          get $(\mathsf{type}, m)$ from $O$
      else $\{P', cid'\} \leftarrow \mathit{next}(cid)$
  else
      $(P', cid') \leftarrow \mathit{prev}(cid)$
  switch (type)
  case extend:
    get $P_{next}$ from $m$; $cid_{next} \leftarrow \{0,1\}^\kappa$
    update $\mathcal{C} \leftarrow \langle P_i \overset{cid}{\Longleftrightarrow} P \overset{cid_{next}}{\Longleftrightarrow} P_{next} \rangle$
    $\mathit{SendMessage}(P_{next}, cid_{next}, \mathsf{create})$
  case extended:
    update $\mathcal{C}$ with $\mathsf{P_{ex}}$; $\mathit{ExtendCircuit}(\mathcal{P}, \mathcal{C})$
  case data:
    if (P = OP) then output $(\mathsf{received}, C, m)$
    else if $m = (S, m')$
       generate or lookup the unique $sid$ for $cid$
       send $(P, S, sid, m')$ to $\mathcal{F}_{\text{NET}^q}$
  case $\mathit{corrupted}$: /*corrupted onion*/
     $\mathit{DestroyCircuit}(\mathcal{C}, cid)$
  case default: /*encrypted forward/backward onion*/
    $\mathit{SendMessage}(P', cid', \mathsf{relay}, O)$
upon receiving a msg $(sid, m)$ from $\mathcal{F}_{\text{NET}}$:
  obtain $\mathcal{C} = \langle P' \overset{cid}{\Longleftrightarrow} P \rangle$ for $sid$
  $\mathit{SendMessage}(P', cid, \mathsf{relay}, \langle \mathsf{data}, m \rangle)$
upon receiving a msg $(P_i, cid, \mathsf{destroy})$ through a handle:
  $\mathit{DestroyCircuit}(\mathcal{C}, cid)$
upon receiving a msg $(P_i, P, h, [\mathsf{corrupt}, T(\cdot)])$from $\mathcal{A}$:
  $(\mathit{message}) \leftarrow \mathit{lookup}(h)$
  if $\mathsf{corrupt} = \mathit{true}$ then
      $\mathit{message} \leftarrow T(\mathit{msg})$; set $\mathit{corrupted}(\mathit{message}) \leftarrow \mathit{true}$
  process $\mathit{message}$ as if the receiving submachine was $P$
upon receiving a msg $(\mathsf{compromise}, P)$ from $\mathcal{A}$:
  set $\mathit{compromised}(P) \leftarrow \mathit{true}$
  delete all local information at $P$
```

Figure 13: The ideal functionality $\mathcal{F}_{\text{OR}}^{\mathcal{N}}$ (short $\mathcal{F}_{\text{OR}}$) for Party P [BGKM12, Fig.5]

```
$ExtendCircuit(\mathcal{P} = (P_j)_{j=1}^{\ell}, \mathcal{C} = \langle P \overset{cid_1}{\Longleftrightarrow} P_1 \Longleftrightarrow \cdots P_{\ell'} \rangle)$:
  determine the next node $P_{\ell'+1}$ from $\mathcal{P}$ and $\mathcal{C}$
  if $P_{\ell'+1} = \perp$ then
    output $(\mathsf{created}, \mathcal{C})$
  else
    if $P_{\ell'+1} = P_1$ then
      $cid_1 \leftarrow \{0,1\}^{\kappa}$; $SendMessage(P_1, cid_1, \mathsf{create})$
    else
      $SendMessage(P_1, cid_1, \mathsf{relay}, \{\mathsf{extend}, P_{\ell'+1}\})$
$DestroyCircuit(\mathcal{C}, cid)$:
  if $next(cid) = (P_{next}, cid_{next})$ then
    $SendMessage(P_{next}, cid_{next}, \mathsf{destroy})$
  else if $prev(cid) = (P_{prev}, cid_{prev})$ then
    $SendMessage(P_{prev}, cid_{prev}, \mathsf{destroy})$
  discard $\mathcal{C}$ and all streams
$SendMessage(P_{next}, cid_{next}, \mathsf{cmd}, [\mathsf{relay\text{-}type}], [\mathsf{data}])$:
  create a $msg$ for $P_{next}$ from the input
  draw a fresh handle $h$ and set $lookup(h) \leftarrow msg$
  if $compromised(P_{next}) = true$ then
    let $P_{last}$ be the last node in the continguous $compromised$ path starting in $P_{next}$
    if $(P_{last} = OP)$ or $P_{last}$ is the exit node then
      send the entire msg to $\mathcal{A}$
    else
      send $\langle P, P_{next}, \ldots, P_{last}, cid_{next}, \mathsf{cmd}, h \rangle$ to $\mathcal{A}$
  else
    send $\langle P, P_{next}, h \rangle$ to $\mathcal{F}_{\mathrm{NET}^q}$
```

Figure 14: Subroutines of $\mathcal{F}_{\mathrm{OR}}$ for Party P[BGKM12, Fig.6]

interface of $\mathcal{F}_{\mathrm{REG}}^{\mathcal{N}}$ (from the setting with Π_{OR}) an initial message with the list $\langle P_i \rangle_{i=1}^{n}$ of valid nodes. This initial message corresponds to the list of onion routers that have been approved by an administrator. We call the part of $\mathcal{F}_{\mathrm{OR}}$ that abstracts the directory servers dir. For the sake of brevity, we do not present the pseudocode of dir. Upon an initial message with a list $\langle P_i \rangle_{i=1}^{n}$ of valid nodes, dir waits for all nodes P_i ($i \in \{1, \ldots, n\}$) for a message (register, P_i). Once all nodes registered, dir sends a message (registered, $\langle P_i \rangle_{i=1}^{n}$) with a list of valid and registered nodes to every party that registered, and to every party that sends a retrieve message to dir.

Messages from $\mathcal{A}$ and $\mathcal{F}_{\mathrm{NET}}$. In Figure 13 and Figure 15, we present the pseudocode for the adversary messages and the network functionality, respectively. For our basic analysis, we model an adversary that can control all communication links and servers in $\mathcal{F}_{\mathrm{NET}}$, but cannot view or modify messages between parties due to the presence of the secure and authenticated channel. Therefore, sub-machines in the functionality store their messages in the shared memory, and create and send handles $\langle P, P_{next}, h \rangle$ for these messages in $\mathcal{F}_{\mathrm{NET}}$. The message length does not need to be leaked as we assume a fixed message size (for all $M(\kappa)$). Here, P is the sender, P_{next} is the receiver and h is a handle or a pointer to the message in the shared memory of the ideal functionality. In our analysis, all $\mathcal{F}_{\mathrm{NET}}$ messages flow to $\mathcal{A}$, which may choose to return these handles back to $\mathcal{F}_{\mathrm{OR}}$ through $\mathcal{F}_{\mathrm{NET}}$ at its own discretion. However, $\mathcal{F}_{\mathrm{NET}}$ also maintains a mechanism through *observedLink* flags for the non-global adversary $\mathcal{A}$. The adversary may also corrupt or replay the corresponding messages; however, these active attacks are always detected by the receiver due to the presence of a secure

```
upon receiving a msg (observe, $P, P_{next}$) from $\mathcal{A}$:
  set $\mathit{observedLink}(P, P_{next}) \leftarrow \mathit{true}$
upon receiving a msg (compromise, $S$) from $\mathcal{A}$:
  set $\mathit{compromised}(S) \leftarrow \mathit{true}$; send $\mathcal{A}$ all existing $\mathit{sid}$
upon receiving a msg $(P, P_{next}/S, m)$ from $\mathcal{F}_{\mathrm{OR}}$:
  if $P_{next}/S$ is a $\mathcal{F}_{\mathrm{OR}}$ node then
    if $\mathit{observedLink}(P, P_{next}) = \mathit{true}$ then
      forward the msg $(P, P_{next}, m)$ to $\mathcal{A}$
    else
      reflect the msg $(P, P_{next}, m)$ to $\mathcal{F}_{\mathrm{OR}}$
  else if $P_{next}/S$ is a $\mathcal{F}_{\mathrm{NET}}$ server then
    if $\mathit{compromised}(S) = \mathit{true}$ then
      forward the msg $(P, S, m)$ to $\mathcal{A}$
    else
      output $(P, S, m)$
upon receiving a msg $(P/S, P_{next}, m)$ from $\mathcal{A}$:
  forward the msg $(P/S, P_{next}, m)$ to $\mathcal{F}_{\mathrm{OR}}$
```

Figure 15: The Network Functionality $\mathcal{F}_{\mathrm{NET}}$ [BGKM12, Fig.7]: A/B denotes that as a variable name either A or B is used.

and authenticated channel between any two communicating parties and we need not model these corruptions.

The adversary can compromise a party P or server S by sending a compromise message to respectively $\mathcal{F}_{\mathrm{OR}}$ and $\mathcal{F}_{\mathrm{NET}}$. For party P or server S, the respective functionality then sets the *compromised* tag to *true*. Furthermore, all input or network messages that are supposed to be visible to the compromised entity are forwarded to the adversary. In principle, the adversary runs that entity for the rest of the protocol and can send messages from that entity. In that case, it can also propagate corrupted messages which in Π_{OR} can only be detected during *UnwrOn* calls at OP or the exit node. We model these corruptions using $\mathit{corrupted}(\mathit{msg}) = \{\mathit{true}, \mathit{false}\}$ status flags, where $\mathit{corrupted}(\mathit{msg})$ status of messages is maintained across nodes until they reach end nodes. Furthermore, for every corrupted message, the adversary also provides a modification function $T(\cdot)$ as the end nodes run by the adversary may continue execution even after observing a *corrupted* flag. In that case, $T(\cdot)$ captures the exact modificaiton made by the adversary.

We stress that $\mathcal{F}_{\mathrm{OR}}$ does not need to reflect reroutings and circuit establishments initiated by the adversary, because the adversary learns, loosely speaking, no new information by rerouting onions.[7] Similar to the previous work [CL05], a message is directly given to the adversary if all remaining nodes in a communication path are under adversary control.

C Tor

In this section we show the proofs for the claims from the Tor-Analysis section 7.

[7]More formally, the simulator can compute all responses for rerouting or such circuit establishments without requesting information from $\mathcal{F}_{\mathrm{OR}}$ because the simulator knows all long-term and session keys. The only information that the simulator does not have is the routing information, which the simulator gets in case of rerouting or circuit establishment.

C.1 Formal Analysis

We prove that $\mathcal{F}_{\text{OR}}$ is $(0, \delta)$-α-IND-CDP. For this, we first prove Lemma 23, which we then use in order to proof Theorem 24.

Lemma 23. *Let* $r_{\mathcal{A}}, r_{\text{CH}} \xleftarrow{R} \{0,1\}^{p(\eta)}$. *For* $\alpha \in \{\alpha_{\text{SSA}}, \alpha_{\text{SUL}}, \alpha_{\text{SRel}}\}$*, it holds that*

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}_0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}})]\\ =&\Pr[\mathcal{A}^{\text{CH}_1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}})]\end{aligned}$$

Proof. We fix the random string r_{CH}. This in turn fixes the circuits drawn by $\mathcal{F}_{\text{OR}}$ for each row. As circuits are drawn independently from the messages transmitted, $\mathcal{F}_{\text{OR}}$ draws the same set of circuits to transmit either input table.

The adversary $\mathcal{A}$ only observes messages of the form $m = (h, P_1, P_2[, P_3, \ldots, P_k, m'])$ that only contain the following information:

a) which party P in $\mathcal{F}_{\text{OR}}$ the message m comes from,

b) which party P' in $\mathcal{F}_{\text{OR}}$ the message m is sent to,

c) which circuit was used by the curcuit-ID (cid),

d) if the following nodes are compromised: $P_3, \ldots, P_k$

e) if the exit node node or the link from the exit node to the server is compromised: the message that is sent.

Observer that the message m' is the same in both scenarios and the handles h are freshly chosen and, hence, do not leak any information about the path or the sender.

For $\alpha \in \{\alpha_{\text{SSA}}, \alpha_{\text{SUL}}\}$, we know that by $\neg\mathcal{D}_\alpha$ the adversary does not compromise the entry node, or the link from the user to the entry node. Let $R \subseteq \{0,1\}^{p(\eta)}$ be the subset of all random strings $r_{\mathcal{A}}$, for which $\mathcal{A}^{\text{CH}_0(\mathcal{F}_{\text{OR}},\alpha,r_{\text{CH}})}(r_{\mathcal{A}}) = 0$. As $\mathcal{A}^{\text{CH}_b(\mathcal{F}_{\text{OR}},\alpha,r_{\text{CH}})}(r_{\mathcal{A}})$ is a deterministic machine, also $\mathcal{A}^{\text{CH}_1(\mathcal{F}_{\text{OR}},\alpha,r_{\text{CH}})}(r_{\mathcal{A}}) = 0$ exactly for every $r_{\mathcal{A}} \in R$, as both CH_0 and CH_1 forward the same messages to $\mathcal{A}$.

If $\alpha = \alpha_{\text{SRel}}$, $\mathcal{A}$ might learn partial information by compromising either entry- or exit-node. But this only allows him to reduce the set of possible input tables to two, each of which could have been selected by only one of the challengers. By the same argument as above, if we fix $r_{\mathcal{A}}$, $\mathcal{A}$ returns the same value, regardless of which challenger he interacts with.

Hence $\mathcal{A}$ does not learn about the challenger's decision, and we get for any random string r_{CH}

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}_0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}}]\\ =&\Pr[\mathcal{A}^{\text{CH}_1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}}]\end{aligned} \tag{5}$$

As the probabilities are the same for any random string r_{CH}, we then get

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}_0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}})]\\ =&\frac{1}{2^{p(\eta)}} \cdot \sum_{r_F \in \{0,1\}^{p(\eta)}} \Pr[\mathcal{A}^{\text{CH}_0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}}]\\ \overset{(5)}{=}&\frac{1}{2^{p(\eta)}} \cdot \sum_{r_{\text{CH}} \in \{0,1\}^{p(\eta)}} \Pr[\mathcal{A}^{\text{CH}_1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}}), r_{\text{CH}}]\\ =&\Pr[\mathcal{A}^{\text{CH}_1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}}) = 0 \mid \neg\mathcal{D}_\alpha(r_{\text{CH}}, r_{\mathcal{A}})]\end{aligned}$$

□

Using Lemma 23, we now prove the main theorem.

Theorem 24. *$\mathcal{F}_{\text{OR}}$ is $(0,\delta)$ - α-IND-CDP for α, i.e*

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0]\\ \leq&\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0]+\delta\end{aligned}$$

with $\delta=\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]$.

Proof.

$$\begin{aligned}&\Pr[\mathcal{A}^{\text{CH}0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0]\\
=&\Pr[\mathcal{A}^{\text{CH}0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}}))]\\
&+\Pr[\mathcal{A}^{\text{CH}0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
\overset{C.1}{=}&\Pr[\mathcal{A}^{\text{CH}0(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&+\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
\leq&\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}}))]\\
&+\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
\leq&\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&+\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\neg\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&+\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0\mid\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
&\qquad\cdot\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]\\
=&\Pr[\mathcal{D}_\alpha(r_{\text{CH}},r_{\mathcal{A}})]+\Pr[\mathcal{A}^{\text{CH}1(\mathcal{F}_{\text{OR}},\alpha,1,r_{\text{CH}})}(r_{\mathcal{A}})=0]\end{aligned}$$

□

C.2 Link Corruption

We extend link corruption for more than one challenge session. Let $\mathcal{L}^*_{\alpha_{\text{SSA}}}$ be the event that in one of the challenge sessions, an entry-link is successfully compromised. Let $i_j, 1\leq j\leq d$ denote the number of possible entry links compromised by $\mathcal{A}$ for the user of the jth challenge-row. We then get

$$\begin{aligned}Pr[\neg\mathcal{L}^*_{\alpha_{\text{SSA}}}]&=\frac{\binom{n-k-i_1}{1}}{\binom{n-k}{1}}\frac{\binom{n-k-i_2}{1}}{\binom{n-k}{1}}\cdots\frac{\binom{n-k-i_d}{1}}{\binom{n-k}{1}}\\&=\frac{(n-k-i_1)(n-k-i_2)-\ldots-(n-k-i_d)}{(n-k)^d}\end{aligned}$$

The probability $Pr[L^*]$ is maximal if the i_j are maximal. This happens when the users for all challenge sessions are the same and the adversary can invest all link corruptions for this single user. We then get

$$\begin{aligned} Pr[\mathcal{L}_{\alpha_{\text{SSA}}}] &\leq 1 - Pr[\neg\mathcal{L}_{\alpha_{\text{SSA}}}] \\ &= 1 - (\frac{n-k-q}{n-k})^d \\ &= 1 - (1 - \frac{q}{n-k})^d \end{aligned}$$