Received 12 January 2023, accepted 27 January 2023, date of publication 6 February 2023, date of current version 27 February 2023.
Digital Object Identifier 10.1109/ACCESS.2023.3242870

SURVEY

A Survey on Anonymous Communication Systems With a Focus on Dining Cryptographers Networks

MOHSEN SHIRALI[1], **TOBIAS TEFKE**[2], **RALF C. STAUDEMEYER**[2], **AND HENRICH C. PÖHLS**[3]
[1]Department of Computer Science and Engineering, Shahid Beheshti University, Tehran 19839-63113, Iran
[2]Faculty of Computer Science, Schmalkalden University of Applied Sciences, 98574 Schmalkalden, Germany
[3]Chair of IT-Security, University of Passau, 94036 Passau, Germany
Corresponding authors: Mohsen Shirali (m_shirali@sbu.ac.ir) and Ralf C. Staudemeyer (r.staudemeyer@hs-sm.de)

ABSTRACT Traffic analysis attacks can counteract end-to-end encryption and use leaked communication metadata to reveal information about communicating parties. With an ever-increasing amount of traffic by an ever-increasing number of networked devices, communication privacy is undermined. Therefore, Anonymous Communication Systems (ACSs) are proposed to hide the relationship between transmitted messages and their senders and receivers, providing privacy properties known as anonymity, unlinkability, and unobservability. This article aims to review research in the ACSs field, focusing on Dining Cryptographers Networks (DCNs). The DCN-based methods are information-theoretically secure and thus provide unconditional unobservability guarantees. Their adoption for anonymous communications was initially hindered because their computational and communication overhead was deemed significant at that time, and scalability problems occurred. However, more recent contributions, such as the possibility to transmit messages of arbitrary length, efficient disruption handling and overhead improvements, have made the integration of modern DCN-based methods more realistic. In addition, the literature does not follow a common definition for privacy properties, making it hard to compare the approaches' gains. Therefore, this survey contributes to introducing a harmonized terminology for ACS privacy properties, then presents an overview of the underlying principles of ACSs, in particular, DCN-based methods, and finally, investigates their alignment with the new harmonized privacy terminologies. Previous surveys did not cover the most recent research advances in the ACS area or focus on DCN-based methods. Our comprehensive investigation closes this gap by providing visual maps to highlight privacy properties and discussing the most promising ideas for making DCNs applicable in resource-constrained environments.

INDEX TERMS Privacy-preservation, anonymity, anonymous communication system (ACS), dining cryptographers network (DCN), unobservability.

I. INTRODUCTION

The continuously growing collection of data by pervasive computing techniques and great advances in communication during the current information age are bringing many benefits to society. This encompasses transformative changes and opportunities created in many aspects of daily life, e.g. healthcare, transportation, education and social interaction [1], [2]. However, much of these collected data might be sensitive or contain personal information. Therefore, their collection and transmission poses serious privacy concerns. These could prevent a wider incorporation of new technologies into daily lives [3], [4].

Furthermore, people might desire strong communications privacy and anonymity on the Internet in many situations. These include circumstances in which people need to report information they may have on unlawful activities without fear of retribution or punishment. Moreover, people who live under regimes that try to limit what their citizens can say and do on the Internet need solutions to circumvent censorship and restrictions concerning the freedom of speech. Additionally, even private citizens may want to be able to freely browse

The associate editor coordinating the review of this manuscript and approving it for publication was Mehdi Sookhak.

the web, without third-parties collecting statistics on their browsing habits and selling that personal information to other companies [5].

End-to-end encryption is often used to combat this situation and to ensure the confidentiality of transmitted messages over intermediate links. Then, only the intended recipient can read the message [6]. However, encryption only hinders third parties from reading transmitted information. It cannot hide the fact that the message exchange is taking place and parties are communicating [7], [8]. Even over encrypted channels, an observer on the network might be able to gather the so-called metadata, which includes information like communication endpoints, the sheer size of exchanged packets [9], the frequency and timing of packets in correlation to other packets, events[1] and location details [11]. This information, when extracted and combined with a priori knowledge, statistics and processed (e.g. by machine learning algorithms) can be rich enough to even bypass end-to-end encryption [12], [13]. The attacks on the users' privacy, which work without the attacker having knowledge of the communications' contents, are called traffic analysis attacks [14], [15]. While the mere knowledge that devices exchange information might not be interesting on its own, the whole situation changes dramatically, when we can map devices to locations and device types, or we can find out the used services from the communication patterns (The interested reader will find more information about traffic analysis and de-anonymization attacks as well as implemented examples in Appendix B-D).

In order to counter such traffic analysis attacks and to minimise any kind of information disclosure occurring as part of the exchange of metadata, more than end-to-end encryption is necessary. For providing privacy to metadata, an additional layer of privacy protection running on top of the existing communication protocols is needed. This layer must hide the fact that communication takes place [5], [11]. Different protocols and mechanisms, that are generally known as ACSs, have been proposed to this aim. ACSs hide the relation between transmitted messages and their senders and/or receivers. Thereby, they allow their users to communicate privately within a network environment [16]. For instance, in (wireless) networks, anonymous communication features could prevent traffic analysis by making the real network traffic indistinguishable from random noise [13].

The ACSs can offer different levels of protection, e.g. anonymity, unlinkability and unobservability (these communication properties are defined in Section II in detail). However, providing different levels of guarantee for privacy comes at a price of performance, scalability limitations and increases practical deployment complexity. So that, all protocols and techniques proposed to enable anonymous communication have to choose a trade-off between efficiency in terms of throughput, latency, and scalability on the one hand and security and privacy guarantees on the other hand.

The basic building blocks of nearly all widely known developed ACSs are two concepts proposed by Chaum: Mix networks in 1981 [17] and Dining Cryptographers Network (DCN) in 1988 [18]. Mix networks (also called Mixes) take a bunch of messages and then scramble, delay and re-encode them. In this way, an eavesdropper can no longer easily correlate incoming with outgoing messages [11]. The strength of Mix techniques relies on multiple transmission and routing of messages. However, this introduces delays between the time a message is sent and the time it arrives at the intended recipient [5]. Besides, DCN is a broadcast round-based protocol wherein only one member can publish one l-bit message per round. The privacy of DCN relies on information coding, and DCN provides unconditional secure unobservable communication [13].

The Virtual Private Networks (VPNs), proxies, Mix-based solutions and onion-based routings like The Onion Router (Tor) [19] are popular ACSs and have been widely adopted in practice due to their support for most network protocols [13]. Nevertheless, they mostly offer limited anonymity protection and an observer who traces packets can still mount traffic analysis attacks to break anonymity guarantees [20], [21]. On the contrary, with DCN-based ACSs an adversary monitoring the users is unable to distinguish messages carrying actual content from random noise. However, these solutions have their own challenges, such as providing round (or slot)-reservation techniques and dealing with disruptions. Moreover, the initial DCN-based ACSs suffered from high computational and communication overheads and lack of scalability [12]. For this reason, even though the history of DCN dates back to almost three decades ago, they were rarely implemented in real-world anonymous communications until solutions were proposed to improve efficiency and make them more realistic [5].

Due to the increasing importance of privacy and the variety of proposed ACSs and their applications, several surveys have been conducted on private and anonymous communication so far. Some of these surveys will be mentioned in Section III-G. However, despite a large number of these articles, there are still issues that need to be addressed. For instance, despite traffic-analysis resistance provided by DCN protocols, their recent improvements or implementations in constrained environments (like [12], [22], and [23]) have not been addressed in survey articles so far. The proposed methods for protection against disruption and contributions that have been made to reduce high computation and latency overheads of DCN approaches in order to make them more practical and efficient to be used in constrained environments are less discussed. Additionally, to the best of our knowledge, there has not been a recent survey article that included newly published ACSs' research and projects, such as cMix [24] and Nym [25] or DCN-based solutions like PriFi [13], Arbitrary length k-anonymous [26] and Shared-Dining [27].

[1]Events occur when the state of an object within a communication changes significantly [10].

The challenge is a lack of a comprehensive study to present recent innovations for anonymous communication, which motivates us to conduct a panoramic review of novel ACSs focusing on DCN-based systems. Our contribution is as follows: Firstly, we evaluate the most recent progress and developments in the area of ACSs, in particular, we analyse DCN-based methods from the past to the present because of their information-theoretic privacy features.

Beyond looking at ACSs and grouping them into families according to their key design decisions, secondly, we look at the ACSs from the privacy perspective. We analysed them regarding their achievements in offering various privacy properties, which is a challenging task without having a common understanding of privacy. This article fills this gap: We observe that the properties to describe anonymous communications do not follow a common definition, which leads to confusion and challenges in comparing different methods. Hence, we define the main commonly used terms based on the literature. With a harmonized terminology as a common ground, we then investigate the reviewed ACSs based on the privacy properties they offer.

Thirdly, the article contributes to focus on the fitness of ACSs when used in everyday applications on the Internet or within networks with more resource-limited nodes, like in the Internet of Things.Finally, this survey concludes with a discussion of the most promising ideas developed by different protocols to mitigate the current ACSs challenges.

The remainder of this article is structured as follows: Section II reviews and harmonizes the definitions of privacy properties. Section III briefly overviews general ACSs methods such as Mix networks, Onion Routing based solutions and Multi-Party Computation (MPC) and offers our analysis of the privacy properties achieved by the different ACSs proposals (see the map of privacy properties of ACSss in Section III-H). Then, the original DCN protocol with its main challenges and ACSs based on DCNs are studied comprehensively in Section IV and Section V, respectively. Again we offer an overview map of the privacy properties achievable with DCNs methods in Section V-K. Finally, Section VI concludes this contribution. Supplementary materials are provided in the Appendices including the abbreviations list in Appendix A and the extensions of the terminology (network and security), adversarial model and traffic-analysis attacks in Appendix B to provide common background knowledge for interested readers. Finally, a detailed review of all the main ACSs which are discussed is provided in Appendix C to wrap up this survey.

II. PRIVACY TERMINOLOGY

A complete body of terminology for talking about privacy has been proposed by Pfitzmann and Hansen [28]. Since then, their definitions have been cited highly in the anonymous communication publications, and they have become the reference terminology. In this survey, we largely follow the most recent version of their terminology [29] and definitions based on it in the literature.

A. ANONYMITY

''The state of being not identifiable within a set of subjects, which is called the *anonymity set*''. ''**Not identifiable** within the anonymity set'' means that only using the information the attacker has at his discretion, the subject is ''not uniquely characterized within the anonymity set'' or the subject is ''not distinguishable from the other subjects within the anonymity set''.[2]

In simple words, anonymity is provided when multiple subjects form a set, for instance in message transmission, it cannot be distinguished who sends or receives the message. Hence, the anonymity set is the set of all possible subjects or actors within a system. Identically in network communication, all the nodes that could have been involved will form the anonymity set. The anonymity property can be refined further based on the role that a specific subject has. Regarding a specific message exchange; the subject can be either the *sender* or *recipient* of a message. The set of subjects which could have sent a specific message is called the *sender anonymity set*. Similarly, all subjects who could have received a particular message form the *recipient anonymity set* [5].

In some cases, sender and recipients want to identify each other while achieving *third-party anonymity*, meaning that they want to be sure that they are interacting with the intended party – while not wanting any other external party to be able to determine that they are communicating with each other [25].

The probability that a verifier can successfully determine the real subject is exactly $\frac{1}{n}$, where n is the number of members in the anonymity set [7]. As this definition implies, an ACS must consist of at least two subjects in order to provide anonymity property, so the anonymity set should always have more than one member.

Reiter and Rubin [30] widened the term of anonymity by adding the degree of anonymity. The degree of anonymity is an indicator telling how exposed a sender and/or receiver is on the spectrum between having absolute privacy and being provably exposed. Later, Shields and Levine [31] refined this indicator by adding specific mathematical definitions to the degrees of anonymity.

To summarize, if a system provides anonymity, it hides the identity of each subject within a set of subjects. Conversely, identifiability means that the attacker can sufficiently identify the subject within a set of subjects and is the opposite of anonymity [29]. Hence, in the literature, *unidentifiability* is sometimes used as an equivalent of anonymity. Moreover, *undetectability* improves the unidentifiability property by making it impossible for an attacker to figure out whether a specific subject exists [12], [29].

B. UNLINKABILITY

A user may make multiple uses of resources or services; however, others are unable to determine whether the same user caused certain specific operations in the system. [32].

[2]Direct quotes show where the exact wording of [29] is used in the definitions.

In an abstract sense, *unlinkability* refers to the inability to determine which pieces of data available at different parts of a system may or may not be related to each other [25]. A network providing unlinkability ensures that neither messages nor network nodes can be correlated. Therefore, the probability of finding relations between senders, recipients and messages stays the same before and after eavesdropping on the traffic [33].

Unlinkability relates to anonymity as follows: *Sender anonymity* means that a particular message is not linkable to any sender, and no sender is linkable to any message. Respectively, *recipient anonymity* means that a particular message is not linkable to any recipient, and no recipient is linkable to any message. Pfitzmann and Hansen further defined the unlinkability between senders and recipients in an anonymous system as *relationship anonymity* [29].

Relationship anonymity means that each message is unlinkable to each potentially communicating pair of subjects and is a **weaker** property than each of sender anonymity and recipient anonymity (sender anonymity or recipient anonymity each alone implies relationship anonymity) [29].

If the unlinkability property holds, an adversary observing senders and recipients in the network is not able to discern any relationship between communicating nodes and cannot distinguish who is communicating with whom [5].

C. UNOBSERVABILITY

Ensures that the communication pattern between senders and recipients remains hidden from the adversary. It conceals the activities of users and adds idle users to the anonymity set [25]. Thus, unobservability hides the fact that a subject is sending or receiving a message, and it is achieved through the use of *"cover"* (or *"dummy"*) traffic [5].

Unobservability is **stronger** privacy feature than unlinkability and anonymity [34]. Unobservability always reveals only a subset of the information that anonymity reveals [29] (with respect to the same attacker, when we have unobservability, we have anonymity as well). In a network with anonymity; when a user sends a message, the adversary cannot identify which of the observed output message corresponds to the user; while unobservability means that the adversary cannot even determine whether the user is sending any message at all, or whether it is just being idle [25], [34].

Similar to anonymity sets, we have unobservability sets that describe the unobservability for a set of subjects considering the subjects' role in communication. *Sender unobservability* or *Sender online unobservability* means that it is impossible to tell whether a sender within the unobservability set currently transmits a message. In other words, sender unobservability is the inability of an adversary to decide whether a specific sender (for any concurrently online sender of the adversary's choice) is communicating with any potential or not [35]. Sender unobservability directly implies the notion of sender anonymity where the adversary tries to distinguish between two possible senders communicating with a target recipient.

Likewise, *recipient unobservability* means that it is impossible to tell if a recipient within the unobservability set currently receives a message. Therefore, it is defined as the inability of an adversary to decide whether any sender is communicating with a specific recipient or not, for any recipient of the adversary's choice [35].

Relationship unobservability then means that it is impossible to figure out whether anything is sent out of a set of could-be senders to a set of could-be recipients [34]. In other words, it is not noticeable if a message is transmitted by any of all possible sender-recipient pairs within the relationship unobservability set [7].

In summary, unobservability ensures that all activities between network nodes remain unnoticeable to eavesdroppers. The messages carrying actual information are not distinguishable from random noise messages and cannot be correlated [11]. As a result, unobservability does not only hide communicating parties. It also hides which subjects exchanged messages during a period of observation [5]. Unobservability ensures unlinkability and unidentifiability, under the assumption of a continuous flow of dummy traffic.

Anonymous communication systems seek all, but provide at least a subset of these privacy properties [5]. However, they are mostly intended and recommended ensuring the highest level: **unobservable communication**. Unobservability is beneficial to any application and data as it frustrates traffic analysis by an attacker who observes local traffic [12].

D. PERFECT PRESERVATION

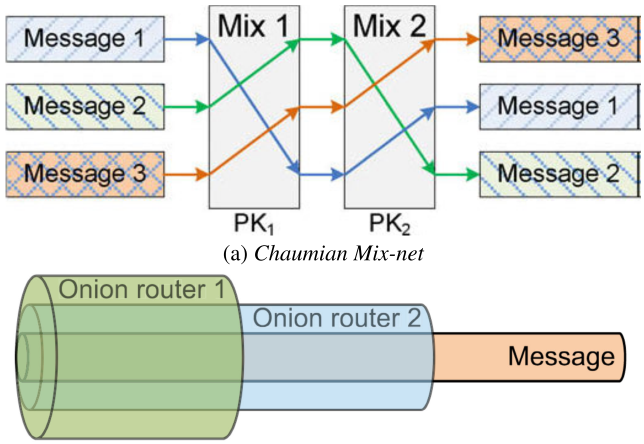
The privacy properties can be evaluated in terms of their change over time. The *perfect preservation* of a property means that its value will not decrease with regard to the attacker's knowledge from the current time relative to the attacker's background knowledge (the a-priori-knowledge of the attacker). For instance, perfect preservation of a subject's anonymity means the anonymity property stays the same over time. Indeed, anonymity does not change when it is compared by taking the attacker's observation into account (new knowledge) with the attacker's background knowledge. The change in anonymity over time- which may be reflected in a decrease in the size of the anonymity set- is called *Anonymity Delta* [29].

Moreover, it should be noted that the security and privacy properties of a system may be held conditionally or unconditionally. A conditional secure system only provides security under certain circumstances: Its security depends on the hardness of a computational problem or the limitation of the adversary's computational power [36]. If an ACS functionality depends on conditionally secure cryptographic algorithms, it is categorised as conditionally secure or cryptographically secure ACS [18]. On the other hand, a system is proved to be unconditional if its security will not be broken by an attacker who is computationally unbounded and has an unlimited amount of time [7]. For instance, an ideal ACS which uses one-time random keys that are at least as long as the message is an unconditional secure –but unfortunately impractical– ACS. In this way, the conditionality/unconditionality of the privacy properties provided by an ACS determines its resilience against attackers.

Furthermore, within the ACSs, the privacy properties are investigated from a *global* perspective; the level of privacy provided by the whole system to all of its users together, and not for each *individual* subject. As an example, global anonymity refers to the anonymity provided by a system to all of its users together, while individual anonymity is the anonymity of one individual subject.

III. GENERAL OVERVIEW OF ANONYMOUS COMMUNICATION SYSTEMS

This article studies ACSs with a focus on DCN-based anonymous communication methods. In order to provide the readers with the broader concept of ACSs, their main methods and some important approaches based on them are briefly discussed in this section. An overview of the offered privacy properties by different ACSs and their classification are presented to conclude this section. This overview investigates the alignment of ACSs with the provided privacy terminology in Section II.



(a) *Chaumian Mix-net*

(b) *Onion routing*

FIGURE 1. The basic ideas of (a) Chaumian Mix-nets and (b) Onion routing (derived from [7]).

A. MIX NETWORKS

Throughout the literature and various categories listed for ACSs in the publications, Mix networks or Mix-based solutions are one of the main ACSs concepts. All Mix-based protocols use a set of mix servers (Mixes) which receive asymmetric encrypted messages from different sources and put them in a queue. Mixes delete replays, collect and decrypt the received messages and if a certain amount of messages has been queued, they are all pushed out in a rearranged order [12]. By doing this, the Mix servers attempt to make the communication paths (including those of the sender and the receiver) ambiguous (see in Figure 1a). The desired anonymity is reached by relying on statistical properties of background traffic (also called cover traffic) [7]. In fact, a Mix network is a mean of sender anonymity.

One trustful Mix among the multiple Mixes of the network, which are selected to pass the traffic, is sufficient to ensure anonymity in Mix-based methods. Hence, the strength of a Mix-based protocol is based on the trust relationship between Mixes, and Mix networks are not capable of providing unconditional anonymity [7]. However, several stages of encryption and message transmission impose large delays on messages, making Mix-based systems undesirable for real-time communication. Furthermore, this highlights the need for ACSs providing low latency message exchange.

Concerning Mixes, the topology of the network strongly influences overhead and anonymity [37]. Therefore, several Mix network prototypes have been developed to address various applications' requirements, particularly the need for low-latency communication, which, for instance, is needed for web browsing and online chats.

Loopix [35] is a Mix network which groups nodes into different layers, where nodes in each layer can communicate with all the nodes in the immediately previous and following layers. Loopix adds independent delays to incoming messages (Poisson mixing) in order to obfuscate message timing. cMix [24] also is a precomputation-based Mix network with fixed cascade architecture. It completely eliminates computationally expensive public-key operations during runtime at the senders, recipients and Mix-nodes. This protocol uses multi-party group homomorphic encryption in order to create a shared secret in the precomputation phase. The decreased real-time cryptographic latency and lowered computational costs for clients made cMix a well-suited system for low-latency applications with lightweight clients. Moreover, MiXiM [38] provides a flexible simulation framework for Mix networks to evaluate different design options and their trade-offs. The MiXiM framework allows assessing combinations of Mix network building blocks by running experiments and providing results for metrics including anonymity, end-to-end latency and traffic overhead [39].

In addition to these prototypes, Privacy and Accountability in Networks via Optimized Randomized Mix-nets (PANORAMIX) [40] and Resilient Anonymous Communication for Everyone (RACE) [41], programs funded by European Union (EU) and United States (US) respectively, along with Nym [25] and Elixxir [42], as two commercial ventures, have strengthened the interest in developing Mix networks during recent years [43]. For instance, among the mentioned projects, the Nym network [25] is proposed to provide a generic infrastructure to be integrated flexibly within myriad services and applications. Its decentralized and incentivized infrastructure offers stronger privacy guarantees to its users by providing a massive anonymity set. The Nym is composed of a decentralized Mix network and an anonymous credential cryptosystem. The anonymous credentials allow users to prove their ''right of use'' when privately accessing services over the Mix network. In fact, the credentials are used as Nym tokens to reward the nodes which provision a high-quality service by adequately routing the traffic. In addition, a blockchain maintained by users decentralizes the operations of the entire Nym network (including the membership and configuration of the Mix network, the issuing of anonymous credentials and the distribution of rewards). Thus, the Nym network is able to provide a scalable privacy infrastructure to protect network traffic metadata for a broad range of message-based applications and services [25].

B. PROXIES AND VPNs

While Mixes explicitly batch and reorder incoming messages, proxies, as the simplest solutions for anonymous communication, merely forward all incoming traffic (e.g. a Transmission Control Protocol (TCP) connection) immediately without any packet reordering [5]. Moreover, Virtual Private Networks (VPNs) – which are popular due to their low latency and support for most network protocols on the Internet [13] – are also designed to establish a secure tunnel between a client and a VPN server. VPNs can be used to encrypt and secure the whole network traffic, not just HTTP or SOCKS requests from a browser (like a proxy server). In such a way, it is not possible to easily map the incoming and outgoing traffic when a proxy or a VPN is used. However, message frequencies and flows can still be analysed. Hence, an observer with access to traffic entering and leaving the proxy or network over extended periods can reveal the communication relation. While VPNs are convenient, using them offers a very limited degree of anonymity [11], [12], [13].

C. ONION ROUTING BASED SOLUTIONS

Later, according to the principle of Chaum's Mix cascades [17], onion routing based methods were introduced by Goldschlag [44], [45], [46] as an equivalent of Mix networks within the context of circuit-based routing. Onion routing differs from Mixes by not routing each packet separately. Instead, the client chooses a path and then opens a circuit through the network by sending the first message and labelling the chosen path. In a circuit, each onion router knows its predecessor and successor, but it does not have any information about other nodes in the circuit [7]. After establishing the circuit, each message having a particular label is routed on this predetermined path. In the end, a message can be sent to close the path [34].

The onion data structure, or simply onion, is composed of layer upon layer of encryption wrapped around the payload (as shown in Figure 1b). When each onion router receives the fixed-length messages, it performs cryptographic operations on each message and thereby removes a layer of encryption by using its own private key. This also uncovers the routing instructions for the next onion router in the circuit, then, the message will be forwarded to the next node. This process is being repeated until the message is delivered to the final onion router. In this way, intermediary nodes have no knowledge of the origin, destination and content of the message [7]. Often, the information travelling through each of the labelled circuits is referred to as an anonymous stream [34].

The ACSs based on onion routing provide an application-independent socket connection. Therefore, they can be easily used by many applications (e.g. web browsing, SSH and instant messaging) [7]. However, onion routing-based ACSs differ regarding how the onion routers are organized, how encryption algorithms are applied, how the tunnels are established, whether the transport-layer uses TCP or UDP, or whether the clients relay traffic to other clients [33]. Thus, a large number of ACSs based on onion routing as underlying approach have been deployed. These solutions have attracted millions of users due to the low-latency connections they offer.

Tor [19], [47] is a distributed-trust, circuit-based low-latency anonymous communication network built upon the onion routing design [48]. Tor is an overlay network (a communication network constructed on top of another network [49]. It consists of a set of voluntary servers called onion routers, which are used to build circuits and relay messages [7]. The Invinsible Internet Project (I2P) is another message-oriented system offering anonymization services by using peer-to-peer low-latency communication. In fact, I2P is another overlay network, mainly designed to enable fully anonymous communication between two parties inside the network [50], [51].

Crowds [30], Hordes [52], LASTor [53], Torsk [54], High-speed Onion Routing at the Network Layer (HORNET) [55], and Non-interactive Anonymous Router (NIAR) [56] are based on onion routing as well. Crowds was specially designed to hide a specific user's action within the actions of many others during web browsing [30], [57], [58]. Instead of operating a set of onion routers, Crowds' clients relay the traffic of others. To create an anonymous web request, the representative process of each client (called *jondo*) establishes a random path by choosing another *jondo* from the crowd and forwarding the request to it [5]. Upon receiving the request by the selected *jondo*, it decides to either randomly forward the request again to another *jondo* or forward it to the intended recipient. The server's response will also be routed in the reverse path through Crowds.

Analogously, Hordes provides a similar degree of anonymity but with a significant performance advantage (in terms of latency in data delivery and the amount of participants' required work by using multicast communication to anonymously route the reply to the initiator) [52]. Another protocol which is known as Freerider-Resilient Scalable Anonymous Communication Protocol (RAC) [59] also bases on the principle of onion routing protocols. RAC provides better scalability and resolves the free-riders issue. In this context free-riders are users who have no interest in acting as relay and drop the messages they are supposed to relay [59].

Furthermore, anonymous broadcast messaging systems based on Mixing and onion routing have also been investigated. Vuvuzela [60], Pung [61], Stadium [62] and Karaoke [63] are designed for private message sharing in order to support a large number of users and provide a 'cover' for sensitive use cases. However, these methods also have their own vulnerabilities and impose large delays that prevent them from being accepted. As an example, Vuvuzela [60] works by routing user messages through a chain of servers and adopts ideas from differential privacy [64] to prove strong guarantees about the level of privacy provided by dummy traffic. It is designed for private message sharing in which both sender and receiver pull some information from the system. Vuvuzela can scale up to two million online users and achieves a throughput of four messages per minute per client with a 37-second end-to-end latency on commodity servers. But, all messages must have fixed size and the server pads them to the largest message size making its adoption to online storage services inefficient [65]. More crucially, it cannot hide the fact that a user is connected to its network [60].

Moreover, there have been significant works on designing file-sharing systems that allow people to anonymously store, publish, and retrieve data (see survey [5] for more information). Peer-to-peer storage and retrieval systems such as Freenet [66], FreeHaven [67] and, later, GnuNet [68], [69] provide anonymous persistent data stores. They use multiple hops to retrieve data associated with a key in a distributed data store.

In summary, solutions based on onion routing employ an application-layer overlay routing and public key cryptography in order to provide sender anonymity. The most popular protocols of them, such as Tor, offer large anonymity sets in the order of millions of users [16]. However, in addition to the latency imposed by the required sequential operations on different servers, the stateful nature of hops or routers makes traffic analysis a serious threat to onion routing approaches [70]. Thus, it must be said that onion routing protocols are not designed to protect users against global adversaries [71]. It has been shown that onion routing protocols are susceptible to a variety of traffic analysis attacks [72], [73], [74], [75], [76], even those performed by local adversaries [77], [78], [79], [80]. For example, in 2014, a study [81] showed that more than 81% of Tor clients can be de-anonymized via traffic analysis.

D. BROADCAST/MULTICAST-BASED SOLUTIONS

Anonymous Communications Systems based on broadcast or multicast methods are designed to provide anonymity, in a scalable manner, through one-to-many communications among hosts [57]. When using broadcast or multicast communication, the message is sent to all (or a set of) nodes of a network, and it protects the receiver's anonymity. In this case, instead of using an implicit destination address to enable the only intended recipient to recognise the message, public-key encryption can be utilised. Every message broadcasts to every participant, then all recipients attempt to decrypt them, whereas only the intended will succeed by using the correct private key. This also ensures confidentiality, integrity and authenticity [82]. If broadcast or multicast techniques are used, the senders send their messages to a group of recipients (while these recipients look the same). According to the definition of anonymity (see Section II), a higher number of recipients lowers the chance for an attacker to guess who the real receiver is. Hence, broadcast-/multicast-based systems can increase the anonymity of its participants.

There exist multiple ACSs using broadcast or multicast-based communication. Even DCN-based methods can be categorised into this type. However, since DCN-based ACSs also provide sender anonymity and are mainly derived from Chaum's protocols; therefore, it is more preferable to consider them as a main separated category. Peer-to-Peer Personal Privacy Protocol (P5) [83], K-Anonymity [84], Multicasting Mixes for Efficient and Anonymous Communication (M2) [85], Mutual Anonymous Multicast (MAM) [86] and Broadcast Anonymous Routing (BAR) [16] can be named as broadcast-/multicast-based ACSs. For instance, P5 creates a broadcast hierarchy, in which different hierarchy levels provide different levels of anonymity at the cost of communication bandwidth and reliability [83]. In P5, all messages sent to a certain receiver are transmitted from a single upstream node. Thus, the receiver does not know the original message sender, also the sender does not know who the receiver is (or which host or address the receiver is using). P5 provides individual participants with a trade-off between the degree of anonymity and communication efficiency. The users always have the flexibility to decrease their level of anonymity in order to increase their performance [7], [57].

Another example of broadcast-based ACSs is Broadcast Anonymous Routing (BAR) which proposed a scalable anonymous Internet communication system that combines broadcast features with layered encryption of Mix networks [16]. In this system, a selective broadcast mechanism can provide significantly lower broadcast costs. Moreover, an efficient filter mechanism allows users to filter out noise traffic and selectively decrypt only those messages intended for them. Unlike Mix network systems, it provides sender, receiver and unlinkability with forward secrecy. The system consists of three different parties: users, the BAR servers acting as the broadcast servers, and a system coordinator whose role is to publish system parameters and support its operation [16]. The BAR design is not distributed, there is a coordinator that acts as a single entity to manage the users, servers and clusters (which must be available in real-time). Hence, the system performance completely depends on the coordinator plus the number of broadcast servers, since each server can only handle up to some hundreds of users according to the implementation results [87].

E. OBLIVIOUS TRANSFER

The first form of Oblivious Transfer (OT) was introduced by Rabin in [88], which used as a secret exchange protocol between two parties [89]. An OT protocol enables a sender to transfer a record of information from a sequence of records to a receiver, while the sender remains oblivious about which record is selected, also the protocol hides the rest of the records from the receiver [90], [91]. A slightly more advanced form of OT is *'chosen one-out-of-two'* OT, denoted as OT_1^2 [92], where the sender has two private inputs (X_1, X_2), and the receiver can choose to get either X_1 or X_2 and learns nothing about the other input [91], [93].

Similarly, the generalised form of OT_1^2 was introduced by Brassard et al. [94] under the name all or nothing disclosure of secrets (ANDOS). In *1-out-of-n* denoted OT_1^n, the sender has n private inputs and the receiver can choose to get one of them on her choice, without learning anything about the other inputs and without knowing the sender which input is transferred [90], [93].

Indeed, OT is a cryptographic primitive that provides the capability for selecting and transferring data between two parties and can be used as a building block in different contexts where there is a requirement to hide or limit the information about data transfer [90]. This property of OT is used to design a delivery mechanism in a novel ACS called Anonymization by Oblivious Transfer (AOT) [95]. AOT is a protocol that uses OT to facilitate anonymous two-way communication and deliver the messages to the recipients. The AOT is a system based on Mix network architecture that comprises three levels of nodes, where each one performs a different function. The senders send the encrypted payloads along with their corresponding tags to the network. The tags are derived from secret keys, which are shared in advance between the sender and receiver of each message and will be later published on a public bulletin board. Then in the network, Level-1 nodes strip the sender information of messages and send them to Level-2 nodes in batch, while Level-2 nodes add dummy messages and send the reordered batches of dummy and real messages to Level-3. At the end, Level-3 nodes publish tags associated with messages on the bulletin board. By identifying any tag, a user knows a message is prepared for him and uses OT to request the message associated with that tag from a Level-3 node. Using OT hides which messages are received by users from a larger set of messages, hence a network adversary cannot link senders and receivers. In summary, the combination of OT and Mix networks increases the anonymity provided by the system, also users do not have to register with the system; any entity who knows the public key of any middle-layer node in the system can send messages with AOT [90], [95].

In the same fashion as OT, Private Information Retrieval (PIR) protocols allow a client to retrieve (or fetch) an item from a destination in possession of the client, a database, without revealing which record is retrieved [65], [96], [97]. Riffle [98] uses hybrid mix networks and PIR techniques to implement anonymous messaging with an acceptable privacy guarantee, but it cannot handle changes in the network topology [99]. Riposte [100] also uses PIR techniques in a system with multiple servers to provide anonymous message broadcasting (Riposte will be explained in detail in Section V-E). Pung [61], Private Keyword-Based Push and Pull (P3) [101], and Private Information Retrieval for Everyone (XPIR) [102] are other anonymous communication systems based on PIR, which use a key-value store to allow clients to deposit and retrieve messages without anyone learning the existence of conversation. Although using smart database organization helps these methods to scale to a large number of users, they exhibit substantial client load. As a result, PIR techniques require high bandwidth and computation, and they usually provide data anonymity meaning the destination knows the client, but, it does not realise what records are read or written.

F. MULTI-PARTY COMPUTATION

Secure Multi-Party Computation (MPC) is a method which enables a group of distrusting parties $P_1, \ldots, P_n$ to collaboratively compute a function *f*. Hereby, each party P_i can contribute an input x_i to the computation; no other party learns these values, but all learn the result $f(x_1, \ldots, x_n)$ [103], [104]. Indeed, the only information each party learns about the inputs of other participants is this result. Therefore, MPC can be used by multiple independent data owners, who do not trust each other or any common third party, to carry out a distributed computing task that depends on all of their private inputs in a secure manner [105]. Oblivious Transfer, which is described in the previous item, is one of the cryptographic primitives for building secure multi-party computations to enable the utilisation of data without compromising privacy.

In general, ACS solutions which use MPC techniques [106] provide strong anonymity guarantees. Sometimes they use mixing to pass the intermediary results through the network, although the sequential mixing is time-consuming and slows down the protocol. For instance, MCMix casts the problem of anonymous messaging in the setting of MPC [99]. The MCMix system enables clients to use a dialling functionality to call other clients and establish a random tag. Subsequently, the dialler and dialee use this tag in the conversation functionality to send messages (even concurrently) [99].

Furthermore, there have been several efforts to use MPC as MPC as a system-as-a-service (MPSaas) [107], [108]. AsynchroMix is an application of the MPSaas approach to anonymous broadcast communication. In a typical client-server setting, the clients send their confidential messages to server nodes which continuously process those encrypted inputs from the clients. The system selects a subset of clients whose inputs are mixed together before making them public. AsynchroMix employs a MPC implementation called HoneyBadgerMPC [109], which relies on the pre-processing paradigm. Thus, it features robust online phases along with non-robust but efficient offline phases [108].

G. EXISTING SURVEYS ON ANONYMOUS COMMUNICATION SYSTEMS

Interested readers who want to study ACSs in more detail can refer to other surveys such as the following articles, while each one has further elaborated ACSs with their perspective.

To name a few, Danezis and Diaz in [34] reviewed briefly the underlying principles, the advantages and disadvantages of various ACSs. Further, an overview of the research in anonymous communications in terms of their basic definitions, cryptographic primitives, network protocols and their applications is conducted in [7]. Also, a summary of ACSs is provided by [110]. Moreover, Fernández in [58] categorised different schemes of anonymous communication according to their architecture, client-server and peer-to-peer. These systems are compared based on their resistance against the most notorious attacks, and several aspects of them are exposed, including scalability, anonymity and unobservability, security and censorship-resistance. Even among more recent studies, a survey on ACSs in the Internet of Things (IoT) is published in [111] to explore them based on computational offloading and lightweight cryptography. Other than that, different types of anonymous communication based on different anonymous mechanisms such as routing, broadcast, etc. and a formalization of the notion of anonymity for measuring its degree are provided in [57].

However, one of the most cited studies on anonymity is the one presented by Edman and Yener [5]. In this paper, major concepts and technologies to design, develop, and deploy systems for enabling private and anonymous communication on the Internet are described. Anonymous systems are classified into high-latency and low-latency systems depending on their intended application and their latency tolerance. Considering this classification, high-latency anonymity systems are able to provide strong anonymity, but they impose significant delays between sending and receiving transmitted messages (up to multiple hours and random message drops). The high-latency systems, also known as message-based systems [5], are typically applicable for non-interactive applications that can tolerate delays of several hours or more. Hence, they can only be used for asynchronous services which accept notable delays [112]. In contrast, low-latency anonymous systems offer better performance and can be adopted for use-cases which require higher levels of interactivity and bidirectional communication channels, e.g. web browsing or Secure Shell (SSH) sessions. Consequently, low-latency systems are also called connection-based systems [5], [112]. Furthermore, DCN-based systems are introduced as a means to offer unobservability in this survey.

According to the literature review, there is a need for a comprehensive study of DCN-based methods. For this reason, the next section of this article is dedicated to a detailed review of these methods.

H. OVERVIEW OF PROVIDED PRIVACY BY ACSs

In this section, a comparison of the offered privacy features by different ACS methods is prepared in the form of a chart, which we introduce as privacy map. To draw this map, privacy properties (anonymity, unlinkability, unobservability and their role-based subcategories) are considered in line with the definitions provided in Section II. Then, the main ACS methods cited in Section III are investigated in terms of the quality of their offered privacy properties. In fact, the privacy map visualises the alignment of ACS methods with the privacy terminology.

The size of anonymity/unobservability sets is represented by circles in three sizes to show the maximum number of supported users in the sets (*small, medium or large*). The circle sizes are determined based on reported evaluation results for each method and the highest number of supported participants at their most efficient setup (acceptable overhead and performance in practice) is represented as the degree of anonymity/unobservability.

In addition, the preservation of anonymity/unobservability properties are presented by the circles' style; *solid* colour represents the methods which provide perfect preservation of anonymity/unobservability, while the *hatched* circles are used to show the methods which are not able to provide perfect preservation. The size of anonymity/unobservability sets of a not-perfect ACS changes over time. Furthermore, *double-circle* icons are used to illustrate the methods which offer variable or time-dependent properties. In these methods, the size of anonymity/unobservability sets depends on the participation of nodes/users during a specific time interval (i.e. usually known as epochs or time slots). Hence, the anonymity set does not have the same size at all times, and the maximum supported sets are much larger than the actual sets. The assumptions on the trust model are also mentioned in the privacy map. Figure 2 presents the privacy map to compare general ACSs.

According to the defined privacy properties in Section II, three different classes can be imagined for the ACSs, in general; (1) Anonymity-support group with VPNs and mostly onion routing-based systems. (2) unlinkability-support group including systems, which uses Mix networks. (3) And unobservability-support group, which provides the highest protection for the users and is supported by systems like recent implementations of Mix networks like cMix [24] and Loopix [35]. This classification is also visible in Figure 2, in addition, this privacy map clearly highlights the effectiveness of different approaches in providing privacy protection.

Forwarding and redirecting the packets alone in the way that happens in onion routing methods such as Tor [47] and I2P [51] is not enough to provide strong privacy protection, and it can only offer anonymity within typically a large anonymity set which should be expected to be degraded over time. Further, batching, re-ordering messages or imposing random delays during the message transmission is necessary to hide the relationship between messages and their links to the senders/receivers. The approach taken by Loopix [35] is good evidence for this claim. Additionally, when a message can be accessed by multiple subjects at the same time, for instance, by broadcasting the message or publishing it on

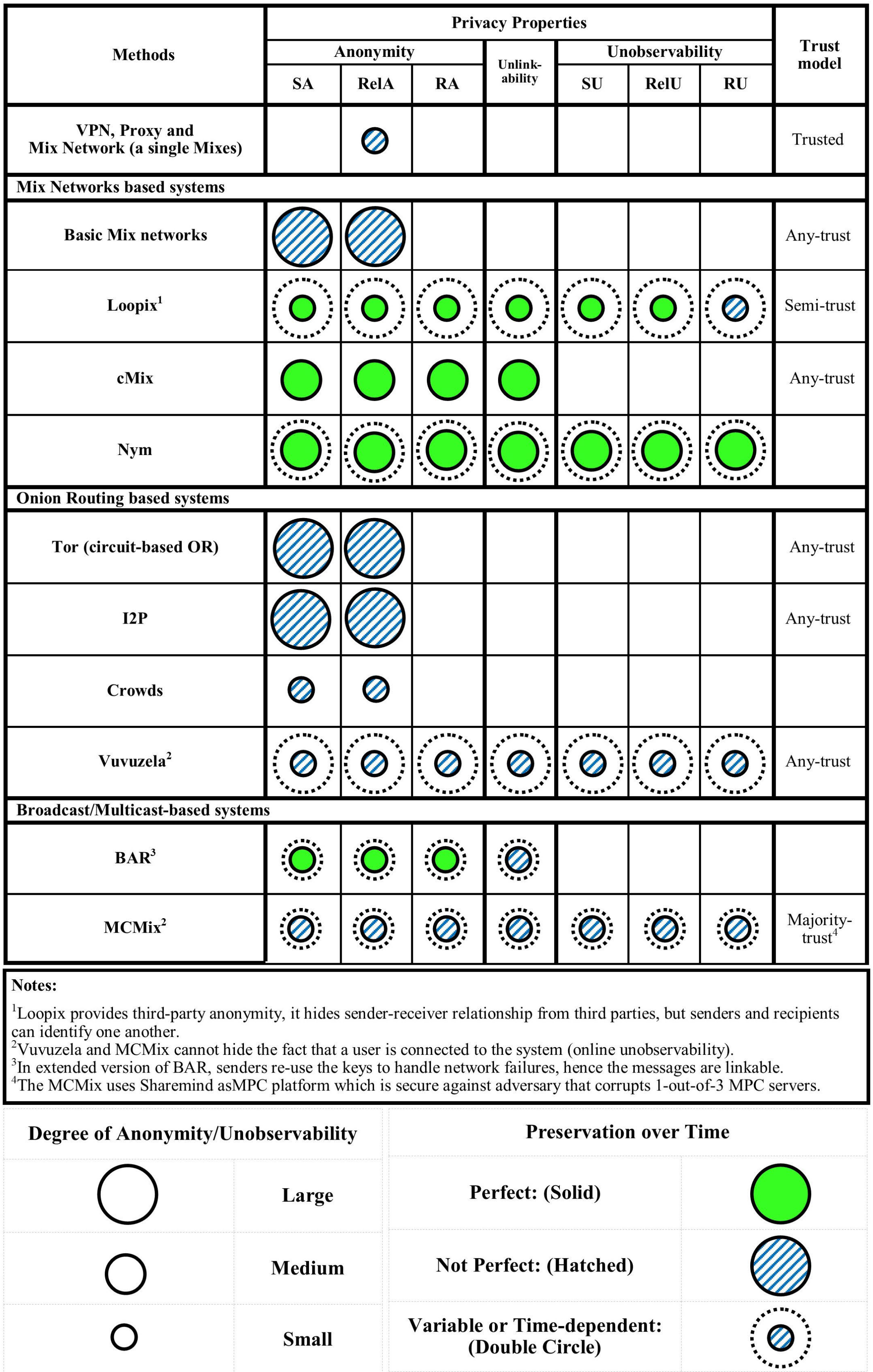


FIGURE 2. Comparison of main general ACSs from the privacy properties perspective.

a shared board/server, this will lead to receiver anonymity. BAR [16] is an example of this approach. The last finding is that the only way to make the actual activities of subjects indistinguishable is to hide them by utilising randomly generated dummy traffic. In this way, the attacker can not understand the difference between real messages and noise, and thus communications will be unobservable. Indeed, Loopix [35], Nym [25] and MCMix [99] all use random traffic to offer unobservability.

Apart from looking at the ACSs from the privacy perspective and categorising them based on their offered privacy or grouping the ACSs based on their main employed idea to protect privacy, it is also possible to classify these methods based on their desired application. In this way, ACSs can be mostly divided into two groups, latency-sensitive and latency-tolerant.

The main criteria for users who want to protect their privacy during latency-sensitive applications such as instant private messaging, web browsing and financial transactions is to experience seamless and (near) real-time communication. Therefore, the ACSs, which impose low or medium latency during the communication phase, can be classified in the latency-sensitive class. VPN and Tor [47], which are the most widely used and well-known methods, are included in this class, along with the recently proposed Loopix [35] and cMix [24] methods.

On the other hand, the methods, which are designed to provide higher privacy protection by adding extra random delays, such as the delayed version of Loopix [35] and MCMix [99], are counted in the latency-tolerant class. The methods of this category are for example desirable to be used for posting on blogs, file sharing and email communications. Detailed information on the ACSs' desired applications is provided in Table 1.

IV. ANONYMOUS COMMUNICATION SYSTEMS BASED ON DINING CRYPTOGRAPHERS NETWORKS

DCN is a broadcast round-based protocol, which provides unconditional secure unobservable communication. The name of this protocol, DCN, comes from a little story, first introduced by David Chaum in 1988 [18]- Dining Cryptographers' Network; "Three cryptographers meet for dinner in a restaurant which has paid beforehand. They are curious who paid for the dinner — either one of the cryptographers or it was sponsored by the employer (for instance the National Security Agency (NSA) or the government). In case one of the cryptographers paid, however, they do not want to reveal who exactly." Chaum came up with the DCN protocol, which provides unconditional security for messages' originators in a closed group to solve this problem [12], [113]. Unconditional secure means that it can be proven that it is impossible to find out who paid [18]. The seminal protocol only allowed participants to unobservably publish a 1-bit message per round, which is called "superposed sending" [18], [114], [115].

In this section, we discuss the DCN protocol, its offered level of anonymity and features besides the main challenges, which have caused DCN to relatively remain neglected in the first years. In addition, the comprehensive review of the proposed solutions based on DCN protocol is provided in V to investigate the recent contributions in the development and implementation of DCN-based ACSs.

A. THE BASIC DCN PROTOCOL

In the following, an example taken from [5] is used to demonstrate the basic principle behind DCN protocol for n cryptographers (or players, participants, users). "Let assume n cryptographers seated in circle as nodes in an undirected circular graph. Every link in this graph, between two neighbouring cryptographers, represent a one-bit secret shared key between the nodes. Let $x_{i,j}$ be the bit shared between neighbouring cryptographers i and j. Further, let s_i be cryptographer i's secret bit indicating whether or not he paid for the meal. Thus, each cryptographer i is announcing the result of $z_i = x_{(i-1),i} \oplus x_{i,(i+1)} \oplus s_i$ to the network. In this way, by receiving a message from each cryptographer, in the end, all the cryptographers can compute the result as shown in Equation 1.

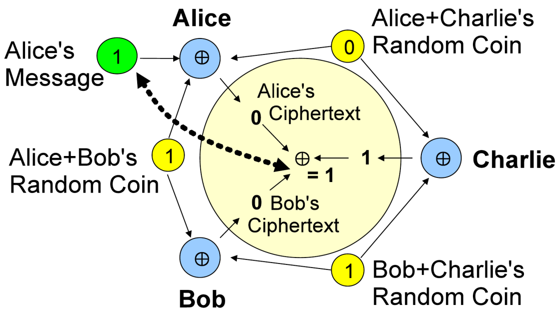
$$\begin{aligned} Z &= z_1 \oplus z_2 \oplus \ldots \oplus z_n \\ &= (x_{n,1} \oplus x_{1,2} \oplus s_1) \oplus (x_{1,2} \oplus x_{2,3} \oplus s_2) \oplus \ldots \\ &\quad \oplus (x_{(n-1),n} \oplus x_{n,1} \oplus s_n) \\ &= x_{n,1} \oplus x_{n,1} \oplus s_1 \oplus x_{1,2} \oplus x_{1,2} \oplus s_2 \oplus \ldots \\ &\quad \oplus x_{(n-1),n} \oplus x_{(n-1),n} \oplus s_n \\ &= s_1 \oplus s_2 \oplus \ldots \oplus s_n \end{aligned} \tag{1}$$

where the third step follows from a simple reordering of terms. If no cryptographer paid for the meal, $s_i = 0$ for all i. Otherwise, there is precisely one s_i that is non-zero and the result $Z = 1$."

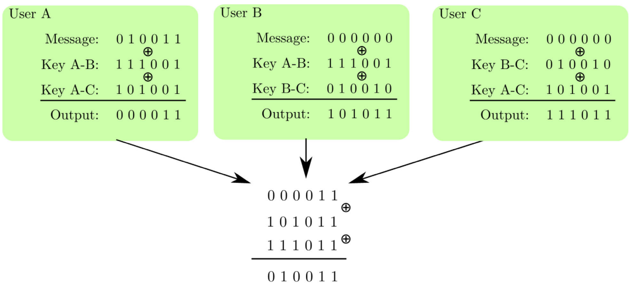
Later, DCN were enhanced to support arbitrary message lengths and shared secret sizes. By running this protocol in several rounds and assigning each round to only one user, this user can anonymously publish an l-bit message. The following example describes DCN for six-bit message transmission.

Examples: In Figure 3a, the DCN protocol is shown in its most basic way: three parties would like to exchange a one-bit message. In this example, taken from [116], each member pairs up with his neighbours. Both flip a coin and agree on a secret result. Afterwards, each member exclusive ors (XORs) all results he knows. The member publishing the message also XORs the message would like to publish with the result of the previous XORs operation. Then, the results of the mathematical operations which took place are published and each member XORs the messages he receives. The result is the message that was published anonymously [116].

In [117], it is described how the basic DCN protocol works with an example, which is also illustrated in Figure 3b. Three participants want to exchange six-bit messages and every participant has a shared symmetric key (six-bit key) with each other participant. By assuming participant A is the node who wants to send a message in this round, she XORs her message



(a) Basic DCN protocol to exchange a one-bit message [116].



(b) A DCN example with three participants to show how this protocol works to transmit a six-bit message per round [117].

FIGURE 3. **The basic DCN protocol flow for three users in a setting in which (a) one-bit and (b) six-bit messages are being transmitted.**

with all the keys she shares with the other participants. The result is an output that A sends out to every participant of the DCN. The other participants perform the same procedure, but use a zero message instead of a meaningful one. All outputs of all participants are XORed together to reveal the meaningful message of participant A, because keys are cancelling out (each symmetric key is used twice). This completes a single round of the DCN, during which, one participant can transfer (broadcast) one message; in the next round, another participant transmits a message until all participants are done transmitting [117].

B. THE STRENGTH OF DCN: UNCONDITIONAL UNOBSERVABILITY

Since, the construction of DCN relies on the 'reliable broadcast assumption' [118], providing receiver anonymity is easy. As mentioned in Section III, broadcast-based concepts offer full receiver anonymity. In addition, the reliability of broadcast means the broadcasted message should be received by every receiver unaltered which can be ensured by verifying the integrity [113].

In terms of preserving sender anonymity, the designated protocol must be able to eliminate the possibility of identifying the message originator, under any circumstances. The privacy is compromised, if an attacker can learn something about a participant's role from the network. The strongest possible attacker, in this case, is able to observe all communications and can collude with all participants except two honest ones (Colluding means the attacker knows all the exchanged keys and the individual messages of the participants [113]. In addition, it is worth mentioning that there is no anonymity if all participants are colluding against one single victim [22]). Nevertheless, in the case of using DCN protocol, the attacker still cannot discover the exact role of the honest participants, i.e. acting as a sender or receiver, even if they are only two.

Indeed, for every message, every participant in a network can be in the role of the receiver, sender, none or both [113]. Therefore, the attacker is not able to distinguish whether honest nodes are communicating, or just being idle, which is consistent with the definition of unobservability. Obviously, the assured unobservability set of a single node in the DCN protocol is equivalent to the set of honest nodes participating in the protocol and the node has already shared secrets with them [22].

Moreover, it should be noted that, to set up keys before message transmission, every participant exchanges random keys with everyone else and add them to the value he sends on the network [113]. Therefore, if the participants of a DCN, share secrets through an unconditional secure channel, the DCN provides unconditional unobservability (or in general privacy). While, if the key exchange is done through a public-key cryptography system, the privacy of a DCN degrades to the degree of computationally robustness of the security for the used public-key cryptography [12].

Subsequent to keys establishment, players may accomplish their anonymous message transmission in a single broadcast round, with no player-to-player communication [119]. This very attractive and compelling feature of the basic DCN, as formulated by Chaum, is called non-interactivity, which is not possible in other privacy-preserving tools like Mix networks based communication protocols [7].

In the end, it can be guaranteed that the basic privacy offering of DCN are much stronger than solutions like Tor [22], and they guaranteed a higher privacy level by providing provable sender and receiver unobservability without relying on a trusted third party [7].

C. MAIN CHALLENGES OF DCN

The listed advantages of DCN protocol and achieving to unconditional unobservability are very decisive to choose an appropriate privacy preserving solution; however, they come at the cost of low throughput and higher computation and delay in particular when scaling to many participants [117]. In addition, there are major drawbacks that are large obstacles to the development of protocols based on DCN. The following can be considered as the main challenging issues and practical problems of DCN adoption and implementation in reality:

1) SCHEDULING (COLLISION PREVENTION)

If two participants send in the same round, their messages collide and become unusable [117]. Even when all participants are honest and adhere to the protocol, there is still no perfect means of enabling them to select distinct rounds in order to transmit their messages in a non-interactive manner [119].

This problem can be avoided by scheduling the rounds or slot reservation in advance. In doing so, each participant needs to know when it is his or her turn to send a message, and it is mandatory to schedule the rounds before any transmission. Therefore, the task of slot reservation or scheduling is to agree on a transmission schedule in a way that each participant knows when to send, but does not learn who is sending in the other slots [117].

For this reason, many of the standard slot-reservation protocols are not applicable due to compromising anonymity [117]. In addition, the probability of a collision never reaches zero with reservation procedures, and in any case there is some possibility that two (or more) players attempt to transmit messages in the same round (or slot). Hence, in all cases, DCN protocols involve collisions (whether of messages or even reservation requests) which mandates retransmissions and causes more cost and delay in delivering messages. The general approaches for slot reservation could be divided into three classes including reservation-map methods, collision-resolution algorithms and secure multi-party computation [117].

2) DISRUPTION AND JAMMING PROTECTION

Besides accidental collisions, a single malicious or dishonest insider – a participant who wants to disrupt the communication – can straightforwardly jam the network by intentionally corrupt or block the transmission of messages from honest participants. These disruptions by malicious participants can prevent the delivery of messages, either by broadcasting invalid messages via tampering bits in encrypted messages of others or even simply by dropping out of the protocol [119]. What makes things worse is that tracing the jamming source, in this case, is challenging due to the privacy guarantee in DCN and each participant is as anonymous as any message originator. Hence, the malicious participant can choose to send a message every round or not following the protocol to launch a denial-of-service attack and disrupt the entire DCN communications without being identified [7]. Detecting cheating players comes at a cost as well; multiple broadcast rounds, high computational and communications overhead and fault recovery are required [119]. For this reason, DCN are known to be vulnerable to disruption attacks (i.e., jamming) [18], and many solutions like using trap rounds, relying on commitments or blame mechanisms have been proposed in the literature to find and exclude the disruptor [13].

3) CHURN HANDLING

Churn handling means the ability of participants to join or leave the network and is another fundamental DCN challenge. A single missing ciphertext in each round prevents the discovery of message. Therefore, unlike other ACSs, the disconnection of any participant invalidates the current communication, forces re-transmission of the data and leads to global downtime where no one can communicate. Hence, it should be handled intelligently in order to prevent imposing extra overheads in each round [13].

4) TOPOLOGY AND SCALABILITY

The initial DCN design requires a shared secret between every pair of participants. In this way, the number of nodes in the network to run DCN protocol and their topology dictates the number of required shared secret key pairs, latency, overhead and the scalability [13]. As the public becomes increasingly concerned about threats to personal privacy, the number of anonymity systems' users is likely to grow and ACSs must be able to support more users [5]. However, since DCN requires every user of the network to participate in every round of the protocol, it quickly becomes impractical as the number of users grows [100]. For this reason, scalability is one of the main problems prevent DCN to be implemented in current real-world scenarios [22].

V. REVIEW OF EXISTING DCN-BASED METHODS

As stated in the DCN challenges, they originally offer unobservability at the cost of high-latency and communication overheads, and they scale poorly [120]. However, over the years, DCNs have been re-used in several ACSs and various improvements have been made to fix the challenges and decrease their cost. This section reviews the efforts in developing methods based on DCNs and explore how their contributions tried to mitigate the weaknesses and vulnerabilities of DCNs.

A. PRELIMINARY STUDIES ON DCN

Chaum himself made the early improvements; a ring topology to decrease the overhead of broadcasting messages and the number of required paired keys [18]. With this topology, each global broadcast message has to travel twice through the ring in order to be received completely by all members. Further, Chaum pointed trap mechanisms out to address disrupter problems. This way, an honest sender places a trap by sending a random message with a secret key in its reserved slot instead of sending an actual message. Then, if the attacker tries to disrupt the communication in this reserved round, the honest node that placed the trap detects it and signals the other participants [7], [22]. Unfortunately, even a computationally limited attacker can forge a trap for an arbitrary slot [7].

Later, Waidner and Pfitzmann generalise the concept of superposed sending by deciding to take the advantage of Abelian finite group $(F, \bigoplus)$ instead of the XOR operation [118], [121]. They also propose a multi-round solution to the disruption problem, which is only guaranteed to identify one dishonest player for a given "trap", but without any chance for fault recovery [7]. In addition, Waidner used additive groups of integers modulo m to improve the original reservation technique. In this case, after broadcasting reservation vectors, participants can sum the positions up to find slots that more than one participant wants to reserve (the addition result is higher than 1). Then, all slots with collisions are skipped, and the DCN protocol is executed only in successfully reserved slots. Franck used the Pfitzmann's collision resolution algorithm again in 2014 in a protocol, which is called Successive Inference Cancellation Tree Algorithm (SICTA). This scheduling protocol operates with multiplication of ciphertexts instead of addition [122].

Additionally, other schemes were proposed to resolve collision [123] and to make DCN robust to disruption by employing cryptographic proofs of correctness rather than traps to detect cheating players. However, the bandwidth costs and inefficiencies of the protocol remain high. For instance, in DCN variant by Von Ahn [84], participants initially partitioned into autonomous groups, then a secret sharing protocol used to establish secrets and the correctness of pads is proven via a cut-and-choose protocol. The communications complexity of this scheme, in the worst case in the presence of cheating player, rises to $O(n^4)$.

B. HERBIVORE

Herbivore is a peer-to-peer scalable ACS introduced in 2002 by Goel et al. [124]. It takes a hierarchical two-level approach. At the lower level, a round protocol governs how bits are sent among the participating nodes, while, at the next level, a global topology control algorithm is employed to divide the network into smaller anonymizing groups [7]. When a new user joins the network, it is assigned to one of many smaller groups of users called cliques. Herbivore guarantees that each clique will have at least k users (between k and $3k$ users), where k is a predetermined constant that describes the degree of anonymity offered by the system. If a clique grows too large, it can be separated into smaller cliques. Similarly, once a clique becomes too small, its users are merged into other cliques. Users within each clique are logically arranged in a star topology, with one user at the centre, and all other nodes communicate via this centre node using DCNs. Each user in the clique still has a shared key with every other member of the clique. At a higher level, cliques are arranged in a ring topology, allowing inter-clique communication [5].

Further contributions of the Herbivore system design are reservation maps and an enhancement in collision avoidance via optimisation in the size of the scheduling message by allowing some collisions during the reservation cycle depending on the message size (collisions for smaller messages are more likely) [117], [124]. Herbivore attempted to adapt DCNs into a design that would make them more efficient and suitable for use in low latency, real-time Internet applications [5].

C. DINING CRYPTOGRAPHERS REVISITED

The ‘Dining Cryptographer revisited’ paper, presented in 2004 [119], is one of the papers related to jamming detection and proposes asymmetric constructions to detect cheating. It describes the intuition behind DCNs based on Chaum’s original paper [18] and reviews the solutions using traps in multi-round protocols such as [18] and [118]. The paper shows that catching cheating players with an overwhelming probability comes at the price of higher computation and communication costs. Hence, it proposed two new DCN constructions to achieve non-interactivity and high-probability detection in identification of cheating players.

For this purpose, by assuming the presence of a reliable broadcast channel and that all messages have authenticity, a different strategy for pad computation (messages or dummy traffic transmitted by players) has been used and more computationally efficient cryptographic proofs of correctness are employed to proof the pad computation. In these new asymmetric constructions, the players only employ bilinear maps to identify the cheating players with a cost that is linear in the number of participating players, which is reasonable for small sets of players. Moreover, in the case of cheating, full fault recovery is possible with just a single additional broadcast and there is no need for repeating the whole transmission round. However, the issues of collisions are not covered in these constructions and DCN considered as a primitive to provide partial throughput [119].

D. ADDING ACCOUNTABILITY TO DCNs WITH VERIFIABLE SHUFFLES (DISSENT, DISSENT IN NUMBERS AND VERDICT)

Another set of enhancements to DCN was made by presenting the secret shuffle methods [125], [126]. In 2007, Studholme and Blake proposed a way to implement secret shuffle based on multi-party computation [127]. In this method, DCN participants are organized in a Mix network and use it to transmit their encrypted round reservation vectors. By passing the reservation vectors through the entire Mix, a secretly permuted vector is obtained, so that each participant can recognise his own request only after the permutation is completed. Therefore, the participant nodes can derive their corresponding reserved round number from the position of their requests within the secretly shuffled vector without any possibility to find the round owners by others [117].

Later, Franck used this idea to derive a fully verifiable variant of DCN [122]. Then in 2010, Corrigan-Gibbs and Ford built an accountable anonymous messaging protocol upon Verifiable shuffles and called it Dining-cryptographers Shuffled-Send Network (Dissent) [128]. This protocol was designed to be used for sending a message anonymously in a small-distributed group to a single recipient or the whole group.

To achieve this goal, the topology in Dissent differs with the complete decentralized approach of the DCN; Dissent uses a client-server architecture and communications always happens among client-server and server-server, but never between two clients directly (Figure 4). Hence, each client node only shares keys with the servers, but not with other clients [22].

The communication protocol in Dissent is implemented as follows; first, each client creates a secret pseudonym key k and sends it encrypted to a server. The servers receive and decrypt the keys, then, shuffle them in a random and secret manner to create a vector with the secret permutation of all

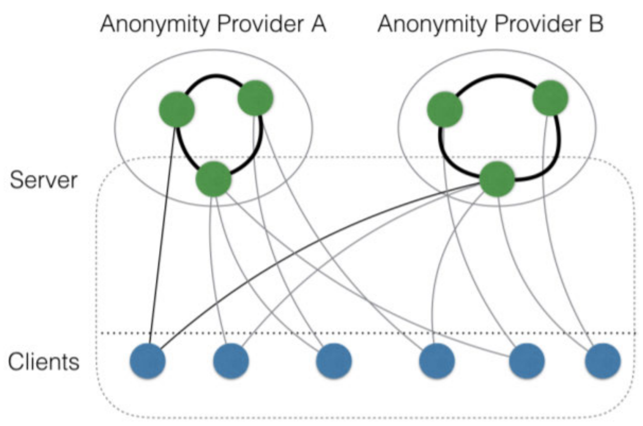


FIGURE 4. The Client-Server architecture of Dissent (adopted from [129]).

clients' received keys. This vector is sent to the clients, then, each client can determine his own key within the shuffled key vector without being able to discover the position of other clients' keys. In this way, the secret shuffling works as a scheduling algorithm and one slot is assigned for each key included in the secret shuffled vector, which is reserved for the corresponding key owner.

Next, for each communication round, according to the assigned slot number, each client can determine if he can actively send his payload, or he has to participate passively by sending a message with zeros as payload. Afterwards, the client creates pseudo-random strings from the secret keys he shared with every server and combines those strings with XOR, then sends the output to at least one server. The servers synchronize messages from each other (by ignoring messages from the same client through the pseudonym keys) and continue the protocol if the number of participating clients in the round is above a certain threshold (to not threaten the anonymity of clients sending in the current round). Finally, each client receives the output of this round, which is signed by every server. Then, each client can extract the round message after verifying the signatures[3] [22], [128].

The approaches used in Dissent's design has added several advantages to it. First, Dissent offers participants the ability to send variable-sized messages by announcing the length of their intended message instead of sending equal-sized messages in original DCN. The length-dissemination phase scales linearly in the number of group members [26].

The next positive point of Dissent is that it allows servers to leave out clients that were leaving the network or clients that are too slow during each communication round. In Herbivore, the ring topology or traditional DCN, the global result of each round relies on all participants' committed messages and the slowest client determines the latency and bandwidth characteristics of the network. However, a fixed receiving window is used in Dissent to tackle this bottleneck to overall performance; clients, who send their messages too late or do not send anything at all, are left out for that round. In addition, a threshold for minimum number of participating nodes, who have to participate in a given round, is considered to avoid the possibility for attackers to isolate currently sending client. If the threshold is not met, the servers will abort the round, or they increase the receiving windows [12].

As a final point, each node in Dissent shares secrets with each server. If there is at least one honest server, the anonymity of a client cannot easily be compromised. In this implementation, the client does not have to know which one of the servers is classified as the honest server, which is commonly referred to as the "any-trust model" [130].

In brief, by using DCN and verifiable shuffle algorithms, Dissent provides integrity, anonymity guaranty and supports variable-length messages [16]. In addition, it holds members accountable, by ensuring that no malicious member can abuse his strong anonymity to disrupt the groups' operation [128]. A prototype of Dissent protocol is implemented in the C++ programming language with the Qt-Framework. In practice, normal TCP traffic can be tunneled through the SOCKS server provided by Dissent and therefore normal application level protocol can transparently use the DCN implementation [22].

However, Dissent has limitations of course, and it is not intended for large scale, 'open-access' anonymous messaging or file sharing. Dissent works for the cases in which very few clients share very small messages and suffers from poor performance at scale. Moreover, since this protocol is designed to ensure resilience to any traffic analysis, its imposed overhead grows linearly with the size of the anonymity set scale [65], [128]. Besides, Dissent's accountability properties assume closed groups and are ineffective if a malicious member can leave and re-join the group under a new (public) identity after expulsion. The serialized shuffle protocol also imposes a per-round start-up delay that makes Dissent impractical for latency-sensitive applications [128].

Later, the same group of authors proposed an improved implementation of Dissent protocol in 2012. Dissent in Numbers [129] is an extension of the original Dissent. The protocol uses a few powerful core servers as anonymity providers in a round-based multiphase protocol, to make it accessible for a large number of clients [26].

In addition, a verifiable DCN was implemented in 2013 under the name Verdict [116]. The advantage of Verdict is that it allows switching between traditional DCN and verifiable DCN, depending on the presence of disruption and for scheduling, it uses a similar approach as Dissent [127], [128], [129]. Additionally, Verdict uses zero-knowledge proofs to proactively exclude misbehaving users before jamming the communication. The proactive exclusion of insider disruption attacks relieves the system from the need to trace a disruptor after the attack. In contrast, Verdict relies on public key cryptography for message encryption, which increases the computation cost in relative to traditional DCN. Verdict is suitable for low-latency communications for small

[3] The signatures are generated and verified using a public key cryptosystem. Each server uses its private key to sign a message and clients verify the message by using the server's public key in the verification step. In Dissent's implementation, 1024-bit RSA-OAEP is used for this purpose [128].

groups of users. However, similar to other approaches based on the verifiable shuffle, it does not scale well.

E. RIPOSTE

A Riposte [100] deployment consists of a few servers, which collectively maintain a shared write-private database and a large number of clients that are allowed to write into it. To post a message, which should be proceeded in regular time intervals, called epochs, a client generates a write request to encode his message and the row index at which he wants to write. Then, the client splits this write request into many shares via secret sharing and sends one share to each Riposte server. A coalition of servers smaller than a pre-specified threshold cannot learn anything about the contents of the write request or write location.

During each epoch time, the servers collect write requests and apply them to their local state. When the servers agree that the epoch has ended, they publish the aggregation of the write requests they received during the epoch to reveal the plaintexts represented by the write requests.

When using a Riposte as a platform for anonymous message broadcasting, the messages are limited to be long enough as database row size; hence, there is a limit for uploading a message. In addition, at the end of each epoch, anyone can recover the set of posts uploaded, so the identity of the entire set of clients who posted during this epoch is known, but no one can link a client to a post. A particular client's anonymity set consists of all the honest clients who submitted write requests to the servers during each epoch. Thus, each time epoch must be long enough to ensure that many honest clients are able to participate. Therefore, the definition of what constitutes an epoch is a crucial decision for the level of offered anonymity.

In comparison with Dissent systems, which also use partially trusted distributed servers to provide anonymity guarantee, Dissent requires a weaker trust assumption than Riposte. At least one of the Dissent servers must be honest, and it has any-trust model, whereas, for a Riposte system at least three non-colluding servers are required, and it has a three-server protocol. On the opposite side, when Dissent clients want to send an l-bit message, they must send the whole message length to all servers, but Riposte clients split their write request into a number of shares, and they have to send the shares in fewer bits to servers, which leads to bandwidth-efficiency by Riposte.

F. FOOTPRINT SCHEDULING

In [117], a new reservation-map method called footprint scheduling was proposed to address the 'slot reservation' issue. Footprint scheduling modifies the original map-reservation algorithm described in the Chaum's DCN [18]. This scheduling uses footprints in B bits ($B > 1$ bit) instead of 1-bit per slot in the reservation vector to decrease the likelihood of undetected collisions in the transmission phase, which occurs when an odd number of players attempt to reserve the same slot. In this case, although the number of bits to represent each slot is multiplied by the B factor, the number of slots in the reservation vector can be decreased.

To reserve a slot in footprint scheduling, players should change the corresponding bits of a slot to a random value. If each slot in the final reservation vector, which results from XORing of all participants' individual vectors, contains a footprint of a participant, the slot is reserved successfully. In contrast, if participants cannot find their original footprints in the round result, then the corresponding participants detect the collision and due to multiple scheduling rounds in a cycle, players can try again to choose another slot. Additionally, skipping non-reserved slots at the end of the scheduling and the possibility to reserve multiple slots which can hide the number of actively sending users, are other improvements of the footprint scheduling method.

A publicly available simulation [131] was used to define optimum values for the footprint scheduling parameters (the number of bits per slot, number of slots and number of scheduling rounds per scheduling cycle) based on the scheduling overhead imposed on each participant for slot reservation. The simulation results for scenarios with three different activity rates show that footprint scheduling yields excellent results, particularly in very dynamic networks with a frequently changing set of participants and frequently changing activity rate. Therefore, using the footprint approach will reduce the probability of undetected collisions in the reservation vector; participants can negotiate for communication slots without losing anonymity, while at the same time, the number of actively sending users will be hidden.

G. DINING CRYPTOGRAPHERS GROUP

Dining Cryptographers groups refer to the protocols where only a part of the whole network participates in the execution of Dining Cryptographers (DC) protocol. This idea is being used in different state-of-the-art protocols such as Dissent variations [116], [128], [129] and k-anonymous groups [84]. Performing DC within the groups with a smaller number of participants provides efficiency in communication with strong privacy guarantee. However, the non-cooperating participants in a protocol based on DC might force the true originator to step up which leads to jeopardising its anonymity. This problem in protocols based on DC groups such as [132] creates additional risks. The common approach to address this issue is to punish and exclude the misbehaving nodes from the group. A better alternative is to incentivize nodes to participate as in the Shared-Dining approach [27], designed in a way that messages can only be read when enough participants cooperate to cross a threshold.

Shared-Dining introduces a combination of Shamir's secret sharing and classical DCN to provide a manageable performance impact on dissemination while enforcing the anonymity guarantees of the protocol throughout the network. In this approach, the broadcasting of a message to all participants in the original DC is replaced by the transmission of n distinct parts. Therefore, in the first phase called Split, the message parts are created by splitting the original message into n shares using a (n, k) Shamir's secret sharing technique. Then, in Distribute phase as the second phase of the protocol, each part is transmitted simultaneously during a modified DC round, resulting in each participant ending up with a single share of the message. In the third phase, Broadcast and Combine, all group members broadcast their received share through the network, allowing any recipient of k shares to reconstruct the message, enforcing anonymity.

If at least k participants broadcast their message, every recipient can decode the original message. When exactly $k - 1$ participants broadcast, only non-broadcasting participants of the group can decode the message, as they possess the last share required to decrypt the message. If $k-2$ or fewer group members broadcast their shares, no one can decode the message, thus preventing privacy breaches for the originator.

The shared-Dining approach is designed to address privacy requirements for financial information. In this regard, a proof-of-concept prototype of Shared-Dining is implemented in Java [133] and its performance in terms of latency and throughput rates are investigated according to different system parameters. The anonymous transmission of transaction data for blockchains in peer-to-peer networks is selected as an evaluation scenario and the results show throughput rates between 10 and 100 kB/s [27].

H. PriFi

PriFi [13] is another DCN based anonymous communication protocol which assures connected users on the Local Area Network (LAN)/Wireless Local Area Network (WLAN) to be indistinguishable from other users. This protocol is similar to a low-latency proxy service (e.g., a VPN or SOCKS tunnel) working within a LAN, creating tunnels between clients and the PriFi relay (e.g., the LAN's router) and these tunnels protect honest client's traffic from eavesdropping attacks.

The main technical contribution of PriFi is a low-latency 3-layer architecture, which removes an important latency bottleneck seen in Mix networks and eliminates the need for costly client-server communication, while adding to the security of all PriFi clients. The clients only 'stream' ciphertexts to the relay and this design dramatically reduces the latency experienced by the clients. The PriFi system goals are anonymity, traffic-analysis resistance, low-latency, accountability and scalability. Additionally, server does not need to be trusted, i.e., security properties hold in case of compromise.

The PriFi communication protocol can be deployed to existing infrastructures with minimal changes. Consider a set of n *clients*, $\{C_1, \ldots, C_n\}$, which are connected within an organizational network through a *relay*, R, which acts as a gateway and connects the LAN to the Internet. The relay can process regular network traffic in addition to running the anonymity protocol (PriFi software). Furthermore, on the Internet, there is a small set of m servers called *guards*, $\{S_1, \ldots, S_m\}$, whose role is to assist the relay in the anonymization process. These guards could be maintained by independent third parties or sold as a 'privacy service' by companies, and it is preferable to be distributed around the world, across different jurisdictions to maximize diversity and collective trustworthiness.

The PriFi protocol is jointly executed by clients, guards and the relay and proceeds in time slots to allow an l-bit anonymous message transmission within each slot according to a defined schedule. The protocol starts with a *Setup* phase to establish a schedule and share secrets for a predetermined timespan (e.g. 10 minutes) which is called epoch. The configuration (i.e. shared secrets and schedule) does not change during an epoch and when its time expires or due to network churn, a new epoch will be created by re-executing of the *Setup* phase. During *Setup* phase, each client (C_i) authenticates itself to the relay using its individual long-term public keys, generates a fresh ephemeral key-pairs, then runs authenticated Diffie-Hellman key exchange protocol with each guard (S_j) to agree on a shared secret (r_{ij}) which is used later to compute the DCN's ciphertexts.

The clients need to know how they should participate in the protocol. For this purpose, the relay prepares a vector of n client's ephemeral public keys and sends it sequentially to all guards in order to perform a verifiable shuffle on the vector. Finally, after the finishing of last server shuffling, relay broadcasts the resulted random permutation of the vector to all clients. At the end of scheduling, each client uses its own ephemeral private key to recognise the corresponding pseudonym key in the vector to find its allocated slot.

After *Setup*, all nodes continuously run the *Anonymize* phase. At each time slot, all the clients and guards participate in a DCN protocol. Each guard seeds a Pseudorandom Generator (PRG) function with all of its shared secrets with clients and XOR all these N values to compute one l-bit pseudo-random message and sends it to the relay. On the other side, all clients, except the slot owner, perform likewise and generate pseudo-random numbers by using the same PRG functions for all shared secrets with guards and XOR all of them to compute one l-bit pseudo-random message to send to the relay. The client owning the time-slot (C_i) additionally includes its upstream message(s) - m_i - in the computation. The upstream message is one or more Internet Protocol (IP) packet(s) without source address, up to a total length l. If the slot owner has nothing to transmit, it sets m_i to 0 in l-bit.

Once the relay receives the ciphertexts from all clients and guards ($n+m$ messages), it XORs them together to obtain m_k. The values of each PRG(r_{ij}), $i \epsilon \{1 \ldots n\}$, $j \epsilon \{1 \ldots m\}$, appears twice in the computation and cancel out, hence $m_k = m_i$, if the protocol is executed correctly. Finally, if m_k is a full IP packet, the relay replaces the null source IP in the header by its own (just like in Network Address Translation (NAT)) and forwards it to its destination. If it is a partial packet, the relay buffers it and completes it during the next schedule. Besides, by receiving an answer to an anonymous message sent in some time-slot, the relay encrypts it under the (anonymous) slot-owner's public key, then broadcasts the ciphertext to all clients. In addition, client churn can be handled in background either as a graceful churn or as abrupt disconnection.

In the threat model of PriFi, all nodes including clients, guards and relay are honest if they follow the protocol faithfully and does not collude or leak sensitive information to any other node. The relay might be malicious and actively tries to de-anonymize honest users but considered trusted for availability, means it will not perform actions that affect the availability of PriFi communications such as delaying, corrupting or deleting clients' messages. In addition, clients can be malicious (controlled by an adversary), but at least two honest clients at all times are required; otherwise, the anonymization will be meaningless. The guards are all highly available and follow the any-trust model (clients do not need to know which one). Thus, the PriFi protects an honest user's traffic among all honest user's traffic and suggested further ideas to hide global/aggregate communication volumes or time series of packets. However, the proposed solutions are not perfect against intersection attacks and could just make them harder.

In terms of security and practical considerations, the PriFi protocol provides techniques for protection against disruption attacks by malicious clients and equivocation attacks by relays that try to de-anonymize clients. An open-source prototype of PriFi was evaluated by Barman and Wolinsky with realistic datasets. The experiments results are publicly available for further investigation [134], [135].

I. VERIFIABLE DCNs BY ADDING COMMITMENTS AND ZERO-KNOWLEDGE PROOFS

For a long time, DCNs were considered unpractical for real-world applications because of the tremendous communication and computation overhead they introduce, in particular for handling malicious participants who disrupt protocol [23]. The advances in cryptographic techniques, such as commitment schemes and Zero-Knowledge Proofs (ZKPs), provide a great opportunity to reduce the communicational cost of modern DCNs and a possibility to detect misbehaving participants more efficiently than before. This led researchers to re-assess the DCN-based solutions and consider a more fundamental role for DCN in future communication [136].

In 2015, Franck extended the DC scheme with the Pedersen commitments to provide Zero-Knowledge verification and unconditional anonymity at the same time [136]. The Pedersen commitments computationally bind participants to their secret keys, and then they could be used to construct ZKPs about the retransmission of data. A ZKP allows a prover to convince a verifier that he knows a witness, which verifies a given statement without revealing the witness or giving the verifier any other information [136]. This verifiable DC scheme does not require any kind of reservation phase prior to the message transmission; hence, it shows a significant improvement over the reservation based techniques [137].

Later in 2021, Franck and his colleagues introduced a library specifically designed to efficiently implement the cryptographic primitives they proposed. The X64ECC is a self-contained library for Elliptic Curve Cryptography (ECC) developed from scratch to fulfil all public-key operations needed by modern DCNs: key exchange, digital signatures, Pedersen commitments, and ZKP [23].

The use of ECC in the library implementation allows keeping the cryptography as compact and as efficient as possible. Furthermore, X64ECC supports three different levels of security, which can be chosen independently for each of the four high-level functionalities. This makes X64ECC easy to use for the implementation of DCNs with arbitrary message sizes, and trade-offs between the cryptographic strength and throughput are possible. Additionally, the arithmetic functions of the X64ECC are parameterized to provide a high level of flexibility and scalability. Also, compiler intrinsics are used to speed up performance-critical arithmetic operations [23].

J. ARBITRARY LENGTH K-ANONYMOUS DC-BASED COMMUNICATION

In the most recent research effort in 2021, Mödinger et al. highlighted the lack of privacy for blockchain systems in transactions' dissemination within peer-to-peer networks connecting all participants and took the advantage of DCN based protocols to address this need [26]. In their work, a number of developed network-layer protocols are reviewed, and a design for a privacy-preserving protocol is proposed with strong privacy guarantees. The main idea of this design is derived from two previous works on DCNs:

1) The k-anonymous message-transmission protocol (which is earlier mentioned as a broadcast based anonymous method in Section III) by von Ahn et al. [84]. This sender- and receiver-k-anonymous protocol is realised to mitigate the scalability weakness of DCNs. For this purpose, participants are assigned to several disjoint groups with only k members and a message is first shared anonymously within a local group providing k-anonymity. Then, all participants of this group send the message to all members of the target group. The participants also create commitments on all messages and broadcast them to the group to provide a blame protocol to identify malicious actors in case of misbehaviour detection. In addition, participants are allowed to reserve more than one slot per round. As a result, the k-anonymous protocol provides fairness and robustness at a higher overhead than basic DCN in malicious environments.
2) 3P3 design [138]. In 2020, the same group of authors, Mödinger et al., addressed the major limitation of predetermined message size and extended the k-anonymous message transmission protocol by proposing a design to support arbitrary-length message transmission. The 3P3 design consists of two consecutive phases, which are each built on a DCN protocol. In the initial phase, the participants anonymously

announce the size of their messages while trying to reserve a slot. For this purpose, each participant shares a vector, which includes the length of his message in his randomly selected slot and sets the remaining slots to zeros. The protocol then merges the vectors of all participants using secure multi-party computation. During the next phase, participants will follow the DCN protocol in successfully reserved slots to disseminate the actual messages. To do this, every participant prepares a message with a length equal to the sum of the sizes announced for this slot during the initial round. If the slot number and the aggregated announced length of message for this slot are equal with the participant's request in the initial phase, hence, this slot is properly reserved for him. Therefore, he should include his actual message; otherwise, he sends a zero message in the specified length. This protocol assumed single group topology, so the final inter-group transmissions of von Ahn et al. protocol are not required.

The 3P3 design is extended and its realisation within a real-world use-case is provided in [26]. In addition, since this protocol has massive overhead to secure its operations in a malicious environment, an unsecured variant (without creating commitments) is proposed to severely reduce the overhead in environments where maliciousness is the exception, i.e., in normal operations. This construction results in a more complex protocol state, it starts with the unsecured variant and will switch to the secure variation once a likely attack is detected. Thus, the performance of the extended 3P3 is optimised by reducing the overhead for the most common cases. A prototype implementation of 3P3 in C++ is used in the simulation to evaluate the expected performance of the extended protocol in blockchain applications and results show the protocol provides enough throughput for most applications by the secured and unsecured variants [139]. The fully secure version is applicable for highly security-relevant applications such as blockchain transactions, while, the version only using the secure version as a fallback mechanism can easily be used for many less critical text-based applications [26].

K. DISCUSSION AND OPEN ISSUES OF DINING CRYPTOGRAPHERS NETWORKS

The anonymity offered by original DCNs is information-theoretically secure, which makes DCNs unique among other solutions. The DCNs' security cannot be broken given unlimited time and computation power [18]. Hence, they are the perfect choice to provide unconditional privacy under any circumstances in current communications through the Internet. As discussed in this section, many efforts have been made up until now to make DCNs more practical by degrading their unconditional privacy to provide more applicable methods with reduced computation and communication overhead. Figure 5 illustrates a privacy map to compare main DCN-based ACSs regarding their offered privacy properties.

The privacy map of DCN-based ACSs clearly shows that these methods have no problem in terms of privacy and are able to provide unobservability. However, the performance of these methods drastically drops when they are scaled up to be used by a large number of participants. Indeed, their imposed computation and communication overhead in association with latency have made them quite inefficient and infeasible in practice. For instance, the basic DCN [18] is almost impractical for groups with more than 10 participants (due to its overhead for disruption protection). The Shared-Dining [27] is designed to implement DCN in groups, and the results of other methods, including Arbitrary length k-anonymous [26] and verifiable DCN [23], [137], also show that they have too much overhead when used by large number of participants and are not applicable. This problem is reflected by small-sized circles in the privacy map.

Considering the performance and overhead issues that hinder the implementation of scalable and practical DCNs, the challenges of DCNs are addressed in several ways. For instance, there have been enhancements to the original DCN to support the transmission of variable-length messages. Moreover, due to the major impact of scheduling algorithms on the fairness, bandwidth utilisation and the perceived latency of the clients, several methods have been proposed to reduce the probability of collision and the amount of computation in the setup phases. However, by scheduling in advance, participants have one or more slots per round, which force them to buffer messages. Still, all active nodes in the network must participate in every round, and they should transmit a message. Even in some protocols, messages must be encrypted before the transmission, which makes things worse. Nevertheless, if a node wants to be inactive for a while in order to avoid this overhead, it has to leave and re-join the network. This change in the network members consequently imposes additional overhead on the remaining participants to establish fresh ephemeral key pairs with others and perform setup and scheduling phases again. Also, the network will be not available for message transmission during the required actions to handle this churn. Therefore, DCN-based methods need more investigation and contribution in this regard.

The next decisive factor is the network topology and architecture. A two-tier architecture, clients and servers, provides higher scalability and reduces the number of required shared keys since each client no longer needs to exchange secrets with all other client. In contrast, the need of relaying messages by servers during message transmission will have a significant negative impact on latency. It requires several client-to-server and server-to-server messages to ensure integrity, accountability and the ability of handling churn.

Furthermore, even if the high overhead is tolerated in a particular scenario, the jamming and disruption attacks should not be overlooked, due to their impact on the overall network performance. Nevertheless, the methods proposed to solve the disruption problem are time-consuming and always computationally intensive. The advances in cryptographic primitives such as commitments and ZKPs help

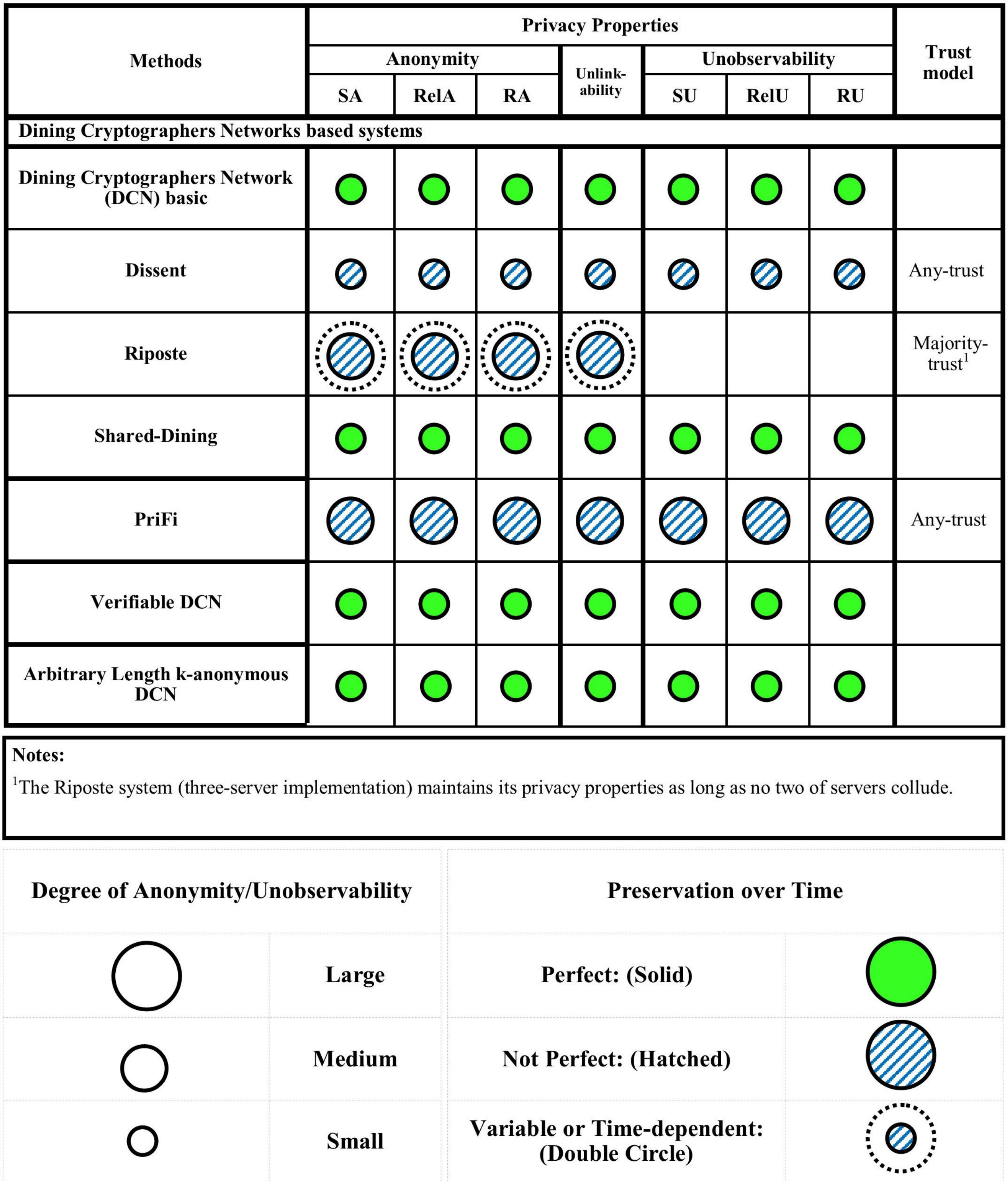


FIGURE 5. Comparison of main DCN-based ACSs from the privacy properties perspective.

detect misbehaving participants efficiently with fewer computations; however, the feasibility of using them in real scenarios needs more investigation. Therefore, it might be preferable to use a DCN-based method, which is able to switch between two different modes. One to be used in environments where disruption is an exception and requires reduced overhead, and the other is a secure or resilient mode, which employs commitments and robust proof of correctness to discover disruptors for a stronger privacy guarantee.

Additionally, regarding the privacy-support classification of the ACSs, as described in section III-H, the privacy map provided in Figure 5 foregrounds that DCN-based systems are placed in the unobservability-support class. In addition, the investigation of DCN variations mostly highlights that except for recent implementations such as PriFi [13], Verifiable DCN [23], [137] and Arbitrary length k-anonymous [26], the legacy methods are inappropriate for latency-sensitive applications.

In the end, despite the definite necessity of privacy preservation for users on the Internet; however, the applicability of DCN-based methods in constrained environments has rarely been explored. The recent innovative solutions and adaptations applied to modern DCN variations have broken down the barriers in its implementation. Some of the most promising innovations are energy-efficient cryptographic primitives, highly scalable architecture designs with reduced latency, and the flexibility of supporting multiple modes of a protocol to provide different levels of privacy and resiliency. Therefore, if a novel variant of DCN-based ACS utilises a combination of these recent innovations, it might eventually be able to facilitate the integration of DCNs in daily real-world scenarios, including the equipment within resource-constrained environments such as IoT devices. In conclusion, DCN-based methods with efficient overhead are worth to be considered; hence, further evaluation of DCN-based methods in IoT testbeds and simulations using realistic datasets is recommended. To facilitate this evaluation, the detailed analysis of de-anonymising techniques could also help system developers realise the security threats and detect possible vulnerabilities of anonymous communications in a better manner by identifying more attack models.

VI. CONCLUSION

Although data encryption algorithms protect the content of communications from unauthorised access, traffic analysis methods can be used to extract valuable information about communicating parties. Anonymous Communication Systems (ACSs) are able to additionally provide strong privacy properties, like anonymity and ideally unobservability. Hence, the growing trend of privacy-sensitive data collection, for instance, via IoT devices, and the increasing usage of wireless communications in data transmission require us to rethink to applying the privacy protections offered by ACSs.

These systems are proposed to prevent traffic-analysis attacks by making the traffic of the network's users indiscernible, and this article was devoted to a comprehensive survey of ACSs with a focus on DCN-based methods, including the latest contributions. In addition, this article provides a common ground privacy terminology to define privacy properties. Then, we investigate the ACSs from the privacy perspective and visualised their alignment with the terminology. It is highlighted that modern ACSs, in particular, DCN-based methods offer better privacy protections. However, their intrinsically introduce significant computation and communications overhead and are less scalable, which often makes them initially appear impractical for real-world scenarios.

Reduction of overhead becomes the next impetus to design and develop practical, low-latency and robust ACSs. In this direction, DCN-based methods with guaranteed unconditional unobservability and provable traffic-analysis resistance seem to be the most appropriate option to choose because they can offer the highest guarantees. The analysed enhancements, such as variable-length message transmission support, commitment techniques or using two-tier architectures in the more recent variants of DCN-based methods have reduced latency, computation and communication overhead and offer higher scalability compared to the original DCN protocol. Still, the DCN-based methods are limitedly realised in real-world use cases such as the constrained environments or for blockchain's transaction dissemination privacy, which has revealed the necessity for further improvements. However, the latest advancements highlight again the strength of DCN-based methods (and also other types of ACSs) for privacy preservation but make these methods more operational in practice.

Thus, we expect a reinforcement in the research on those types of ACSss that bring the most suitable privacy guarantees to the application and hope that they will finally become woven into future communication networks to intrinsically protect our privacy.

APPENDIX A
ABBREVIATIONS

A complete list of appeared abbreviated terms in this article and their correspondig definitions are provided below.

ACS	Anonymous Communication System.
ANDOS	all or nothing disclosure of secrets.
AOT	Anonymization by Oblivious Transfer.
BAR	Broadcast Anonymous Routing.
DC	Dining Cryptographers.
DCN	Dining Cryptographers Network.
Dissent	Dining-cryptographers Shuffled-Send Network.
ECC	Elliptic Curve Cryptography.
EU	European Union.
HORNET	High-speed Onion Routing at the Network Layer.
HTTP	Hypertext Transfer Protocol.
I2P	Invinsible Internet Project.
IoT	Internet of Things.
IP	Internet Protocol.
ISO	International Organization for Standardization.
ITU	International Telecommunication Union.
LAN	Local Area Network.

LTE	Long-Term Evolution.
M2	Multicasting Mixes for Efficient and Anonymous Communication.
MAC	Media Access Control Address.
MAM	Mutual Anonymous Multicast.
MPC	Multi-Party Computation.
MPSaas	MPC as a system-as-a-service.
NAT	Network Address Translation.
NIAR	Non-interactive Anonymous Router.
NSA	National Security Agency.
OT	Oblivious Transfer.
P3	Private Keyword-Based Push and Pull.
P5	Peer-to-Peer Personal Privacy Protocol.
PANORAMIX	Privacy and Accountability in Networks via Optimized Randomized Mix-nets.
PIR	Private Information Retrieval.
PRG	Pseudorandom Generator.
RAC	Freerider-Resilient Scalable Anonymous Communication Protocol.
RACE	Resilient Anonymous Communication for Everyone.
SSH	Secure Shell.
TCP	Transmission Control Protocol.
Tor	The Onion Router.
UDP	User Datagram Protocol.
US	United States.
VPN	Virtual Private Network.
WLAN	Wireless Local Area Network.
XOR	exclusive or.
XPIR	Private Information Retrieval for Everyone.
ZKP	Zero-Knowledge Proof.
SICTA	Successive Inference Cancellation Tree Algorithm.
SOCKS	Socket Secure.

APPENDIX B
BACKGROUND KNOWLEDGE

To have a common understanding of the properties used to describe anonymous communications, we must introduce the commonly used terms from the literature in a harmonized way to avoid any confusion and to establish a common ground for later usage. We start by presenting networking terms in Section B-A. Then, in Section B-B, the well-known security properties are presented to describe the foundational requirements of a secure information system. Afterwards, in Section B-C, we lay the foundation to characterise an adversary, against whom an ACS seeks to defend. Finally, in Section B-D, a brief overview of traffic analysis and de-anonymization attacks is given.

A. NETWORK TERMINOLOGY

The communications that should be hidden take part between multiple participants which form a network. There are different terms that are used to describe the participants of a network and the actions occurring in it. In order to make clear the networking terminology, which is used throughout the article, these terms are explained here briefly.

1) SENDER

A networking device acts as a sender whenever it sends a signal to the network (e.g. in order to communicate with another device) [140, pp. 35-36].

2) RECIPIENT

In difference from being a sender, a device is a recipient whenever it receives a signal from the network [140, pp. 35-36].

3) PACKET

Data transmission over a network happens using so-called 'datagrams'. A datagram may be too large to be transmitted over a network. Then, it has to be fragmented into multiple smaller packets [141, p. 8].

4) NODE

A network node is an element which takes part in communication. A node can either be an endpoint (a sender or recipient) or a point which receives a packet and redistributes it to another one [140, p. 473].

5) METADATA

Metadata is, as the name indicates, data about data. This term is widely used for describing data that is generated when data exchanges occur. If packets are transmitted between network nodes, there is also data exchanged besides the payload, e.g. the Media Access Control Addresss (MACs) of the involved nodes. This data is then called metadata [142, pp. 1-2].

6) INTERNET OF THINGS (IoT)

The term Internet of Things is now more and more broadly used; however, it is hard to find a common definition or understanding of what IoT actually encompasses [143] and manifold definitions are presented within the research community. For instance, the International Telecommunication Union (ITU) defines the Internet of Things as 'a global infrastructure for the Information Society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies' [144, p. 1].

The basic idea of the IoT concept is the ubiquitous presence of a variety of things or objects around us, which are able to interact with each other and cooperate with their neighbours to reach common goals [145]. In this way, an IoT system logically can be depicted as a collection of smart devices that interact on a collaborative basis to fulfil a common goal [146].

B. SECURITY PROPERTIES

The main goals of information security are confidentiality, integrity and availability [147, pp. 2-4]. Regarding communication, we define them as properties following the

International Organization for Standardization (ISO)[4] as follows:

1) CONFIDENTIALITY
The 'property that information is not made available or disclosed to unauthorised individuals, entities or processes' [148, def. 3.36.4]. Thus, confidentiality protected information is only accessible to the people or systems which were allowed to access it. Confidentiality can be ensured by using access control mechanisms, e.g. encryption and giving a decryption key only to authorised people or systems.

2) INTEGRITY
Defined as the property that allows to proof that the message content has not been altered, deliberately or accidentally in unauthorised manner during transmission [148, def. 3.46 & 3.79]. This allows maintaining the correctness and trustworthiness of data, as unauthorised third parties are not able to change the data without detection.

3) AVAILABILITY
The 'property of being accessible and useable upon demand by an authorised entity' [148, def. 3.27.1], individual or process. It means a legitimate user or system can always access a system or a resource without suffering restrictions.

In addition to maintaining integrity, we often need to explicitly make sure that received information is from a certain origin, e.g. in communication, the receiver would like to ensure that a message originated from a certain sender. By intuition the authentication of the origin is thought to be included in integrity, but not clearly by definition [149], thus we explicitly define it for later use in this contribution:

4) AUTHENTICITY OF DATA
A positive output of the 'process of corroboration that the source of data received is as claimed' [148, def. 3.23.3] indicates authenticity. Thus making sure that when we require authentication of messages the origin of the data or message can be verified.

C. ADVERSARY MODEL
Due to the wide range of attacks applicable on ACSs, modelling realistic capabilities of the adversary can provide better insight to assess the resiliency of various ACSs against attacks. Adversaries are mostly defined according to their goals, i.e. breaking a privacy or security property, and their strengths; the following properties as suggested by Raymonds [14] can be used in combination to describe the strength of an adversary [5]:

1) CAPABILITY
Defines the ability of an adversary to monitor or manipulate the network traffic. A *passive adversary* is able to monitor and record the traffic on the network links. This also applies to metadata about network flows. Whereas, an *active adversary* has all capabilities of a passive attacker and is also able to manipulate network traffic. This can happen by either controlling one or more network links or by operating a node in the network.

2) VISIBILITY
Determines how much of the network is passively monitored or actively manipulated by the adversary. A *global adversary* is able to access and observe all communication lines within a network [12]. In contrast, a *partial* (or local) *adversary* is only able to monitor a subset of links or nodes of the network.

3) MOBILITY
Categorises the adversary depending on the ability to select a specific subset of a network. An *adaptive adversary* can choose subsets of the network to monitor, while a *static adversary* is not able to change the observable subset of the network at will.

4) PARTICIPATION
Characterises adversaries based on their engagement in the network protocol. An *internal adversary* is one who participates in the anonymity network's protocol as a node; the adversary can be a client (node at endpoint) or perhaps operate a piece of the network's infrastructure by running a server in the network. An *external adversary*, on the other hand, does not participate in the anonymity network nor in its protocol. Instead, the adversary compromises the communication medium used by nodes (i.e., their network links) in order to monitor or manipulate their network traffic.

It is a prudent practice in information security to try to defend against a worst-case scenario. The adversary model most often assumed in the literature on ACSs is a passive global adversary (who can access and observe the whole network) [5]. This could be made worse by assuming also an internal and adaptive, as well as an active adversary.

D. TRAFFIC ANALYSIS AND DE-ANONYMIZATION ATTACKS
Traffic analysis attacks are inherently not detectable as they occur [12]. These attacks ignore the content of messages and instead try to obtain as much information as possible from only network traffic metadata, such as packet arrival time and message lengths. The attackers passively collect metadata about the messages or traffic flows and try to correlate senders and recipients [5]. In the same way that attackers can use traffic analysis techniques to compromise the anonymity of users in a system, system designers also can use these techniques to find the vulnerabilities within their designs [150]. Attacks based on traffic analysis have been the subject of several studies [58], [151]. Hence, a brief introduction to the most significant traffic analysis attacks – that have been applied to ACSs – are presented in the following.

[4]Note, ISO maintains a free terminology database at www.iso.org/obp.

TABLE 1. Detailed overview of ACSs from the performance aspect.

ACS	Computation Cost	Transmission overhead	Latency	Disadvantages	Main Features	Desired Applications	Traffic Analysis Attacks
VPN, Proxy	Low	No Overhead	Low	• Single point of failure • Fully trusted party (server)	Simple	Supports most network protocols	✓ • Timing • Fingerprinting
				Mix-Networks Systems			
Basic Mix networks [17]	High	Low	High	• Anonymity and overheads are topology-dependent • Poor performance • Undesirable for real-time applications	Re-ordering and transferring messages cryptographically	Delay-tolerant-Applications	✓ Timing
Loopix [35]	High	High	• Low (without delay) • High (added delay at Mixes)	• Third-party anonymity/unobservability • Message processing and packaging is the most computationally expensive part	• Obfuscate timing (via independent poisson mixing delay) • Tunable latency/anonymity trade-off • Tunable real/cover traffic load • Continuous (does not operate in synchronized rounds) • Supports delivering messages to offline users	• Private emails • Instant messaging	N/A
cMix [24]	• Setup: Medium • Precomputation: Zero • Real-time: Low	• Setup: Medium • Precomputation: Low • Real-time: Low	• Setup: Medium • Precomputation: Medium • Real-time: Medium	User activity at any given time is known	• Precomputation based • No public key encryption at run time • Cascade architecture • Multi-party group homomorphic cryptographic system • Resistance to traffic analysis and intersection attacks • Linear scalability	Low latency applications with lightweight clients	N/A
Nym [25]	N/A	N/A	N/A	• Security analysis is not available • implementation details are not available	• Generic - Incentivized - Decentralized and stratified infrastructure • Continuous-time • Using loops of cover traffic • Better privacy and performance trade-offs when scales up to support multiple services and applications	Ranging from instant messaging to cryptocurrency transactions	N/A
				Onion Routing Based Systems			
TOR (Curcuit-based Onion Routing) [47]	High	Medium	Low	Does not perform permutation of messages	• Centralized (any)trusted design • Low-latency • Circuit-based	generic	✓ • Correlation attacks • Timing • Fingerprinting • Disclosure attacks
I2P [50], [51]	Medium	High	High	• Does not hide running I2P and is publicly known • Aim is defending against local network adversaries	• Decentralized with no trusted parties • Message-oriented (packet-switched) • Peer-to-peer overlay network • Encrypted unidirectional tunnels • Stores routing and contact information in distributed hash tables (DHT)	• Web hosting • Web browsing • Email and chat • File sharing	✓ De-anonymizing and disclosure attacks
Crowds [30]	Small	Small	Medium	No protection against global adversaries	Crowds members cannot identify message originator	Web browsing	✓ Timing attacks
Vuvuzela [60]	High	• Conversion: Medium • Dialing: High	• Conversion: Medium • Dialing: High	• High bandwidth cost (almost 30GB over a month) • Operates in rounds • Users can send and receive only one message per round • Offline users might miss messages	• Scalable with linear cost • Two protocols (dialing and point-to-point conversation) • Using differential privacy • Preserve fixed message sizes and rate	Private messaging	N/A
				Broadcast/Multicast-based Systems and MPC-Based Systems			
BAR [16]	• Registration: High • Communication: High	• Registration: High • Communication: Medium	• Registration: High • Communication: Medium	High bandwidth and latency cost	• Trade-off between bandwidth and latency with selective Broadcast mechanism • Filtering mechanism to distinguish between noise and unobservable bilateral communication	Anonymous broadcasting	N/A
MCMix [99]	• Registration: Low • Dialing: Medium • Conversation: Medium	Low	• Registration: Medium • Dialing: High • Conversation: High	Each user can send and receive one message per conversation round	• Unobservable bilateral communication • Completely hides metadata • Using MPC and oblivious sorting algorithm for efficient implementation • Proceeds in rounds via two types of protocols (Dialing and Conversation)	Point-to-point messaging	N/A
				DCN-based Systems			
Basic DCN [18]	Medium	High	High	• Disruptions • Only supports transmission of one-bit messages per round	Information-theoretically secure		
Dissent [128], [129]	Medium	High	High	• Linear increase of overhead with anonymity set size • Not intended for large scale • Per round start-up delay due to serialized shuffle protocol	• Client-server architecture • Scheduling via secret shuffling • Support variable-size message transmissions • Minimum participating users threshold (minimum anonymity set) • Leaves out slow users	• Latency tolerant messaging • File sharing	✓ Intersection Attacks
Riposte [100]	Medium	High	High	• Trust model (three non-colluding servers) • Long epochs to ensure enough user participation • Uploading limit for message sizes • Identity of active users in each epoch is known	• Secret sharing of write request • Bandwidth-efficiency (transmission of message shares in fewer bits)	Anonymous message broadcasting	✓ Intersection attacks
Shared-Dining [27]	High	High	High	• Supports group with small number of participants • Fixed-length messages	• Combination of secret sharing with classical DCN • Incentivize nodes to participate in the protocol • Prevents privacy breach by considereing a threshold for participating users in a round	Applications with high privacy requirements, e.g. financial systems	N/A
PriFi [13], [120]	• Setup: High • Anonymize: Medium	• Setup: Medium • Anonymize: High	• Setup: Medium • Anonymize: Low	• Fix-length messages • High overhead for disruption attacks	• 3-layer low-latency architecture • Keeps packets inside usual local low-latency path • Deployable to existing infrastructures with minimal changes	Organizational communication networks	✓ Intersection attacks
Verifiable DCN [23], [137]	Medium	Medium	Medium		• Development of self contained ECC library for all public key operations • Trade-off between cryptographic strength and throughput • Arbitrary-length messages • Efficient misbehaving user detection • No need for reservation phase		N/A
Arbitrary-Length k-Anonymous DCN [26]	• Unsecured: Low • Secured: High	• Unsecured: Medium • Secured: High	• Unsecured: Low • Secured: High	• Massive overhead to secure operations • Only single group transmissions are implemented in the current version	• Two variant and reducing the overhead for the most common cases • Secured variant is available as a fallback mechanism • Arbitrary-length messages	From highly security relevant applications to less critical applications	N/A

1) WEBSITE FINGERPRINTING

Encrypted communication still reveals who is communicating with whom and how much data is transmitted through the network [152]. To start a fingerprinting attack, an attacker connects to websites and records the generated metadata and traffic. Then, the attacker analyses this metadata (packets' length and quantity) and uses a supervised classifier to build a fingerprint of what the website's response looks like when it is fetched via an encrypted connection [5], [58]. Afterwards, the attacker can compare the observed traffic pattern of target users against the stored fingerprint database to figure out which websites the observed users are visiting [33], [153]. Active, passive, and semi-passive are three classes of fingerprinting techniques [154]. To mitigate this attack vector, most ACSs split messages into equal-length packets.

2) TIMING ATTACKS

A passive global adversary, who is able to observe connections entering and exiting the anonymity network, observes the duration of communication between nodes. Then, the attacker is maybe able to correlate incoming and outgoing packets through timing analysis [5], [12], [37]. The adversary uses the patterns of packet inter-arrival times to link the network participation's patterns [66].

3) DISCLOSURE ATTACKS

When users engage in repeated or persistent communications, their frequent communication partners may eventually be uncovered just by observing the edges of the network and correlating the activity at the two ends of the communication [34]. These attacks exploit the fact that a sender's correspondents will appear more frequently when that sender is active [43]. Therefore, the adversary observes multiple sets of recipients for consecutive message transmissions by that user. Each of these sets contains precisely one communication partner of the sender. In a long run, the adversary refines these sets by intersecting observed new recipient sets with previous sets and continues to do this until he reduces all sets to only a single element, thus de-anonymizing each correspondent of the sender [5], [34], [58]. The disclosure attacks were first presented in [155] and have also been referred to as intersection attacks in the literature [156]. Rather than mounting exact disclosure attacks that precisely identify users' communication partners, an improved statistical variant of these attacks is also presented in [156], [157], and [158] to effectively reveal probable communication partners. These kinds of attacks, in fact, explore fundamental limitations for any systems that select trusted parties, at random, from a larger set of potentially corrupt nodes to provide security [34]. No efficient method for absolutely preventing intersection attacks has been found so far. However, inserting dummy traffic is a measure often proposed to reduce the effectiveness of such attacks [5].

The traffic analysis attacks applied in Tor networks [153] - the most popular overlay network to provide anonymous communication by redirecting traffic with the largest anonymity set-, cellular LTE communications [9] and several feasibility studies in various environments, such as smart home use-cases by utilising network traffic rates [11], [151], [159], are examples in this regard.

APPENDIX C
DETAILED OVERVIEW OF ACSs

A summary of the major ACSs methods from the overall performance aspects is presented in Table 1. In this table, the main affecting factors in the overall performance of a method including computational and communication overheads, latency, disadvantages, main features and their possible attacks are listed. The computations cost, transmission overhead and latency of various ACSs are evaluated according to the efforts required for different phases of each method such as setup, transmission and recovery from the failures or disruptions. Moreover, the likelihood of conducting successful traffic analysis attacks for each method is also represented in table and ✔ symbols in different sizes have been used for this purpose.

ACKNOWLEDGMENT

The authors would like to express their gratitude to Prof. Jorge Cuellar for his constructive feedback, particularly on the literature review section. The collaboration on this article was established during the research stay of the corresponding author at the Chair of IT-Security, University of Passau. Hence, the authors Mohsen Shirali and Ralf C. Staudemeyer would also like to acknowledge Prof. Joachim Posegga and Dr. Mona Ghassemian for their support in making his research visit happen and for their valuable comments. The research stay of the first corresponding author was (partially) supported by DAAD scholarship under grant no. 57440917.

REFERENCES

[1] L. Cranor, T. Rabin, V. Shmatikov, S. Vadhan, and D. Weitzner, "Towards a privacy research roadmap for the computing community," 2016, *arXiv:1604.03160*.

[2] R. Mendes and J. P. Vilela, "Privacy-preserving data mining: Methods, metrics, and applications," *IEEE Access*, vol. 5, pp. 10562–10582, 2016.

[3] H. C. Pöhls, V. Angelakis, S. Suppan, K. Fischer, G. Oikonomou, E. Z. Tragos, R. D. Rodriguez, and T. Mouroutis, "RERUM: Building a reliable IoT upon privacy- and security-enabled smart objects," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2014, pp. 122–127.

[4] D. J. Cook, G. Duncan, G. Sprint, and R. L. Fritz, "Using smart city technology to make healthcare smarter," *Proc. IEEE*, vol. 106, no. 4, pp. 708–722, Apr. 2018.

[5] M. Edman and B. Yener, "On anonymity in an electronic society: A survey of anonymous communication systems," *ACM Comput. Surveys*, vol. 42, no. 1, pp. 1–35, Dec. 2009.

[6] E. Z. Tragos, H. C. Pöhls, R. C. Staudemeyer, D. Slamanig, A. Kapovits, S. Suppan, A. Fragkiadakis, G. Baldini, R. Neisse, P. Langendörfer, Z. Dyka, and C. Wittke, "Securing the Internet of Things-security and privacy in a hyperconnected world," in *Building the Hyperconnected Society—Internet of Things Research and Innovation Value Chains, Ecosystems and Markets*. Denmark, Europe: River, Sep. 2015, pp. 189–219.

[7] J. Ren and J. Wu, "Survey on anonymous communications in computer networks," *Comput. Commun.*, vol. 33, no. 4, pp. 420–431, Mar. 2010.

[8] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," in *Proc. IEEE Symp. Secur. Privacy*, May 1997, pp. 44–54.

[9] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "Breaking LTE on layer two," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 1121–1136.

[10] W. van der Aalst, *Process Mining*. Cham, Switzerland: Springer, 2016.

[11] R. C. Staudemeyer, H. C. Pohls, and M. Wojcik, "The road to privacy in IoT: Beyond encryption and signatures, towards unobservable communication," in *Proc. IEEE 19th Int. Symp. World Wireless, Mobile Multimedia Networks (WoWMoM)*, Jun. 2018, pp. 14–20.

[12] R. C. Staudemeyer, H. C. Pöhls, and M. Wójcik, "What it takes to boost Internet of Things privacy beyond encryption with unobservable communication: A survey and lessons learned from the first implementation of DC-net," *J. Reliable Intell. Environments*, vol. 5, no. 1, pp. 41–64, Feb. 2019.

[13] L. Barman, I. Dacosta, M. Zamani, E. Zhai, A. Pyrgelis, B. Ford, J. Feigenbaum, and J.-P. Hubaux, "PriFi: Low-latency anonymity for organizational networks," in *Proc. Privacy Enhancing Technol.* Berlin, Germany: Walter de Gruyter, no. 4, Aug. 2020, pp. 24–47.

[14] J.-F. Raymond, "Traffic analysis: Protocols, attacks, design issues, and open problems," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2001, pp. 10–29.

[15] G. Danezis and R. Clayton, *Introducing Traffic Analysis*. Boca Raton, FL, USA: Auerbach, 2007.

[16] P. Kotzanikolaou, G. Chatzisofroniou, and M. Burmester, "Broadcast anonymous routing (BAR): Scalable real-time anonymous communication," *Int. J. Inf. Secur.*, vol. 16, no. 3, pp. 313–326, Feb. 2017.

[17] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[18] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *J. Cryptol.*, vol. 1, no. 1, pp. 65–75, Jan. 1988.
[19] R. Dingledine, N. Mathewson, and P. Syverson, "TOR: The second-generation onion router," in *Proc. 13th Conf. USENIX Secur. Symp. (SSYM)*, San Diego, CA, USA, vol. 13. USA: USENIX Association, 2004, p. 21.
[20] L. Nguyen and R. Safavi-Naini, "Breaking and mending resilient mix-nets," in *Proc. Int. Workshop Privacy Enhancing Technol.* Berlin, Germany: Springer, 2003, pp. 66–80.
[21] B. Pfitzmann, "Breaking an efficient anonymous channel," in *Advances in Cryptology (EUROCRYPT'94)*, Cham, Switzerland: Springer, 1994, pp. 332–340.
[22] J. Bauer and R. C. Staudemeyer, "From dining cryptographers to dining things: Unobservable communication in the IoT," in *Proc. 22nd Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Jun. 2017, pp. 1–7.
[23] B. Dupont, C. Franck, and J. Großschädl, "Fast and flexible elliptic curve cryptography for dining cryptographers networks," in *Mobile, Secure, and Programmable Networking*. Berlin, Germany: Springer, 2021, pp. 89–109.
[24] D. Chaum, D. Das, F. Javani, A. Kate, A. Krasnova, J. D. Ruiter, and A. T. Sherman, "cMix: Mixing with minimal real-time asymmetric cryptographic operations," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur. (ACNS)*. Cham, Switzerland: Springer, 2017, pp. 557–578.
[25] C. Diaz, H. Halpin, and A. Kiayias, "The NYM network: The next generation of privacy infrastructure," Nym Technol. SA, Switzerland, White Paper, Version 1.0, 2021, vol. 1.
[26] D. Mödinger, A. Heß, and F. J. Hauck, "Arbitrary length K-anonymous dining-cryptographers communication," 2021, *arXiv:2103.17091*.
[27] D. Mödinger, J. Dispan, and F. J. Hauck, "Shared-dining: Broadcasting secret shares using dining-cryptographers groups," in *Proc. Int. Conf. Distrib. Appl. Interoperable Syst. (DAIS)*. Berlin, Germany: Springer, 2021, pp. 83–98.
[28] A. Pfitzmann and M. Hansen, "Anonymity, unobservability, and pseudonymity—A proposal for terminology," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science), vol. 2009. Berlin, Germany: Springer-Verlag, 2000, pp. 1–9.
[29] (2010). *A Terminology for Talking About Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management*. [Online]. Available: http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf
[30] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, 1998.
[31] C. Shields and B. N. Levine, "A protocol for anonymous communication over the internet," in *Proc. 7th ACM Conf. Comput. Commun. Secur.*, Nov. 2000, pp. 33–42.
[32] *Information Security, Cybersecurity and Privacy Protection—Requirements for Attribute-Based Unlinkable Entity Authentication*, International Organization for Standardization (ISO), Standard ISO/IEC 27551, 2021.
[33] E. Erdin, "Anonymous communication networks: Usage analysis and attack mechanisms," M.S. thesis, Dept. Comput. Sci. Eng., Univ. Nevada, Reno, NV, USA, 2012.
[34] G. Danezis and C. Diaz, "A survey of anonymous communication channels," Microsoft Research, Cambridge, U.K., Tech. Rep. MSR-TR-2008-35, 2008.
[35] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis, "The Loopix anonymity system," in *Proc. 26th USENIX Secur. Symp. (USENIX Security)*, 2017, pp. 1199–1216.
[36] H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications* (Information Security and Cryptography), 3rd ed. Cham, Switzerland: Springer, 2015. [Online]. Available: https://books.google.de/books?id=KwmkCgAAQBAJ
[37] C. Diaz, S. J. Murdoch, and C. Troncoso, "Impact of network topology on anonymity and overhead in low-latency anonymity networks," in *Privacy Enhancing Technologies*. Berlin, Germany: Springer, 2010, pp. 184–201.
[38] I. B. Guirat, D. Gosain, and C. Diaz, "MiXiM: A general purpose simulator for mixnet," in *Proc. Privacy Enhancing Technol. Symp. HotPETs Workshop*. Berlin, Germany: Walter de Gruyter, 2020, pp. 1–3.
[39] I. Ben Guirat, D. Gosain, and C. Diaz, "MiXiM: Mixnet design decisions and empirical evaluation," in *Proc. 20th Workshop Workshop Privacy Electron. Soc.*, Nov. 2021, pp. 33–37.
[40] The European Commission. (2019). *PANORAMIX: Privacy and Accountability in Networks Via Optimized Randomized Mix-Nets*. Accessed: Jun. 26, 2022. [Online]. Available: https://cordis.europa.eu/project/id/653497
[41] Defense Advanced Research Projects Agency (DARPA). (2018). *Resilient Anonymous Communication for Everyone (RACE)*. Accessed: Jun. 26, 2022. [Online]. Available: https://www.darpa.mil/program/resilient-anonymous-communication-for-everyone#:~:text=The%20Resilient%20Anonymous%20Communication%20for,any%20participant%20in%20the%20system
[42] (Jul. 2021). *Elixxir—A Decentralized Blockchain Platform*. Accessed: May 4, 2022. [Online]. Available: https://boostylabs.com/cases/elixxir
[43] C. Diaz, "Mix networks," in *Encyclopedia of Cryptography, Security and Privacy*. Springer, 2022, pp. 1–5.
[44] D. M. Goldschlag, M. G. Reed, and P. F. Syverson, "Hiding routing information," in *Proc. Int. Workshop Inf. Hiding (IH)*. Berlin, Germany: Springer, 1996, pp. 137–150.
[45] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 482–494, May 1998.
[46] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Commun. ACM*, vol. 42, no. 2, pp. 39–41, 1999.
[47] P. Syverson, R. Dingledine, and N. Mathewson, "TOR: The second generation onion router," in *Proc. 26th USENIX Secur. Symp. (USENIX Security)*, 2004, pp. 303–320.
[48] B. Conrad and F. Shirazi, "A survey on Tor and I2P," in *Proc. 9th Int. Conf. Internet Monitor. Protection (ICIMP)*, 2014, pp. 22–28.
[49] W. Galuba and S. Girdzijauskas, *Overlay Network*, L. Liu and M. T. ÖZsu, Eds. Boston, MA, USA: Springer, 2009, doi: 10.1007/978-0-387-39940-9_1231.
[50] J. P. Timpanaro, C. Isabelle, and F. Olivier, "Monitoring the I2P network," Ph.D. dissertation, INRIA Nancy-Grand Est, France, 2011.
[51] J. P. Timpanaro, I. Chrisment, and O. Festor, "I2P's usage characterization," in *Proc. Int. Workshop Traffic Monitor. Anal. (TMA)*. Berlin, Germany: Springer, 2012, pp. 48–51.
[52] B. N. Levine and C. Shields, "Hordes: A multicast based protocol for anonymity1," *J. Comput. Secur.*, vol. 10, no. 3, pp. 213–240, Jul. 2002.
[53] M. Akhoondi, C. Yu, and H. V. Madhyastha, "LASTor: A low-latency as-aware Tor client," in *Proc. IEEE Symposium Security Privacy*, May 2012, pp. 476–490.
[54] J. McLachlan, A. Tran, N. Hopper, and Y. Kim, "Scalable onion routing with torsk," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, Nov. 2009, pp. 590–599.
[55] C. Chen, D. E. Asoni, D. Barrera, G. Danezis, and A. Perrig, "HORNET: High-speed onion routing at the network layer," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1441–1454.
[56] E. Shi and K. Wu, "Non-interactive anonymous router," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2021, pp. 489–520.
[57] T. Lu, Z. Du, and Z. Jane Wang, "A survey on measuring anonymity in anonymous communication systems," *IEEE Access*, vol. 7, pp. 70584–70609, 2019.
[58] L. F. Franco et al., "A survey and comparison of anonymous communication systems: Anonymity and security," Universitat Oberta de Catalunya, Barcelona, Spain, 2012.
[59] S. B. Mokhtar, G. Berthou, A. Diarra, V. Quema, and A. Shoker, "RAC: A freerider-resilient, scalable, anonymous communication protocol," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst.*, Jul. 2013, pp. 520–529.
[60] J. van den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich, "Vuvuzela: Scalable private messaging resistant to traffic analysis," in *Proc. 25th Symp. Operating Syst. Princ.*, Oct. 2015, pp. 137–152.
[61] S. Angel and S. Setty, "Unobservable communication over fully untrusted infrastructure," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 551–569.
[62] N. Tyagi, Y. Gilad, D. Leung, M. Zaharia, and N. Zeldovich, "Stadium: A distributed metadata-private messaging system," in *Proc. 26th Symp. Operating Syst. Princ.*, Oct. 2017, pp. 423–440.
[63] D. Lazar, Y. Gilad, and N. Zeldovich, "Karaoke: Distributed private messaging immune to passive traffic analysis," in *Proc. 13th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2018, pp. 711–725.
[64] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, Aug. 2014.

[65] M. Akhoondi, "Scalable techniques for security and anonymity in distributed systems," Ph.D. dissertation, Dept. Comput. Sci. Eng., Univ. California, Riverside, CA, USA, 2015.

[66] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "FreeNet: A distributed anonymous information storage and retrieval system," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2001, pp. 46–66.

[67] R. Dingledine, M. J. Freedman, and D. Molnar, "The free haven project: Distributed anonymous storage service," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science). Berkeley, CA, USA: Springer, 2001, pp. 67–95.

[68] K. Bennett and C. Grothoff, "GAP–practical anonymous networking," in *Proc. Int. Workshop Privacy Enhancing Technol. (PETS)*. Berlin, Germany: Springer, 2003, pp. 141–160.

[69] C. Grothoff, K. Bennett, and B. Matheny, "GNUnet: A secure peer-to-peer framework," in *Proc. 4th Annu. Inf. Secur. Symp. (CERIAS)*, 2003, p. 1.

[70] I. A. Sergeev, "Network coding for anonymous broadcast," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 2013.

[71] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson, "Users get routed: Traffic correlation on tor by realistic adversaries," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 337–348.

[72] M. K. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 4, pp. 489–522, 2004.

[73] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *Proc. Symp. Secur. Privacy (SP)*, 2005, pp. 183–195.

[74] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against tor," in *Proc. ACM Workshop Privacy Electron. Soc.*, Oct. 2007, pp. 11–20.

[75] N. Hopper, E. Y. Vasserman, and E. Chan-Tin, "How much anonymity does network latency leak?" in *Proc. ACM Trans. Inf. Syst. Secur. (TISSEC)*, Feb. 2010, vol. 13, no. 2, pp. 1–28.

[76] P. Mittal, A. Khurshid, J. Juen, M. Caesar, and N. Borisov, "Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting," in *Proc. 18th ACM Conf. Comput. Commun. Secur. (CCS)*, Oct. 2011, pp. 215–226.

[77] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proc. ACM Conf. Comput. Commun. Secur.*, Oct. 2012, pp. 605–616.

[78] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: Passive deanonymization of Tor hidden services," in *Proc. 24th USENIX Secur. Symp. (USENIX Security)*, 2015, pp. 287–302.

[79] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proc. 10th Annu. ACM Workshop Privacy Electron. Soc.*, Oct. 2011, pp. 103–114.

[80] T. Wang and I. Goldberg, "Improved website fingerprinting on Tor," in *Proc. 12th ACM Workshop Privacy Electron. Soc.*, Nov. 2013, pp. 201–212.

[81] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis, and A. D. Keromytis, "On the effectiveness of traffic analysis against anonymity networks using flow records," in *Proc. Int. Conf. Passive Act. Netw. Meas. (PAM)*. Cham, Switzerland: Springer, 2014, pp. 247–257.

[82] R. C. Staudemeyer, H. C. Pöhls, and B. W. Watson, "Security and privacy for the Internet of Things communication in the smartcity," in *Designing, Developing, and Facilitating Smart Cities*. Cham, Switzerland: Springer, Dec. 2017, pp. 109–137.

[83] R. Sherwood, B. Bhattacharjee, and A. Srinivasan, "P5: A protocol for scalable anonymous communication," *J. Comput. Secur.*, vol. 13, no. 6, pp. 839–876, Dec. 2005.

[84] L. Von Ahn, A. Bortz, and N. J. Hopper, "K-anonymous message transmission," in *Proc. 10th Conf. Comput. Commun. Secur. (CCS)*, Oct. 2003, pp. 122–130.

[85] G. Perng, M. K. Reiter, and C. Wang, "M2: Multicasting mixes for efficient and anonymous communication," in *Proc. 26th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2006, p. 59.

[86] L. Xiao, X. Liu, W. Gu, D. Xuan, and Y. Liu, "A design of overlay anonymous multicast protocol," in *Proc. 20th IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, Apr. 2006, pp. 10.

[87] P. K. G. Chatzisofroniou. (2016). *Broadcast Anonymous Routing—A Scalable System for Efficient Anonymous Communications*. Accessed: Apr. 10, 2022. [Online]. Available: https://github.com/sophron/BAR

[88] M. O. Rabin, "How to exchange secrets by oblivious transfer," Aiken Comput. Lab., Harvard Univ., Cambridge, MA, USA, Tech. Rep. TR-81, 1981.

[89] V. Gupta, T. S. Vineeth, and V. Aggarwal, "Make your query anonymous with oblivious transfer," in *Proc. 6th Int. Conf. Comput. Commun. Technol.*, Sep. 2015, pp. 345–349.

[90] F. Javani, "Privacy-preserving protocols with oblivious transfer and mix networks," Ph.D. dissertation, Dept. Comput. Sci. Elect. Eng., Univ. Maryland, Baltimore, MD, USA, 2021.

[91] B. Schoenmakers, "Oblivious transfer," in *Encyclopedia of Cryptography and Security*, H. C. A. van Tilborg, Ed. Boston, MA, USA: Springer, 2005, pp. 445–446, doi: 10.1007/0-387-23483-7_285.

[92] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," *Commun. ACM*, vol. 28, no. 6, pp. 637–647, 1985.

[93] M. Naor and B. Pinkas, "Computationally secure oblivious transfer," *J. Cryptol.*, vol. 18, no. 1, pp. 1–35, Jan. 2005.

[94] G. Brassard, C. Crepeau, and J. Robert, "All-or-nothing disclosure of secrets," in *Advances in Cryptology (CRYPTO)*, vol. 263. Cham, Switzerland: Springer, 1987, pp. 234–238.

[95] F. Javani and A. T. Sherman, "AOT: Anonymization by oblivious transfer," 2021, *arXiv:2105.10794*.

[96] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proc. 36th Annu. Found. Comput. Sci.*, 1995, pp. 41–50.

[97] Y. Gilad, "Metadata-private communication for the 99%," *Commun. ACM*, vol. 62, no. 9, pp. 86–93, Aug. 2019.

[98] A. Kwon, D. Lazar, S. Devadas, and B. Ford, "Riffle: An efficient communication system with strong anonymity," in *Proc. Privacy Enhancing Technol. Symp. (PETS)*, vol. 2016, no. 2. Berlin, Germany: Walter de Gruyter, 2015, pp. 115–134, doi: 10.1515/popets-2016-0008.

[99] N. Alexopoulos, A. Kiayias, R. Talviste, and T. Zacharias, "MCMix: Anonymous messaging via secure multiparty computation," in *Proc. 26th USENIX Secur. Symp. (USENIX Security)*, 2017, pp. 1217–1234.

[100] H. Corrigan-Gibbs, D. Boneh, and D. Mazieres, "Riposte: An anonymous messaging system handling millions of users," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 321–338.

[101] L. Kissner, A. Oprea, M. K. Reiter, D. Song, and K. Yang, "Private keyword-based push and pull with applications to anonymous communication," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur. (ACNS)*. Cham, Switzerland: Springer, 2004, pp. 16–30.

[102] C. Aguilar-Melchor, J. Barrier, L. Fousse, and M.-O. Killijian, "XPIR: Private information retrieval for everyone," in *Proc. Privacy Enhancing Technol. Symp. (PETS)*, vol. 2. Berlin, Germany: Walter de Gruyter, Dec. 2016, pp. 155–174.

[103] M. von Maltitz, S. Smarzly, H. Kinkelin, and G. Carle, "A management framework for secure multiparty computation in dynamic environments," in *Proc. NOMS IEEE/IFIP Netw. Operations Manage. Symp.*, Apr. 2018, pp. 1–7.

[104] J. Mueller-Quade and H. Imai, "Anonymous oblivious transfer," 2000, *arXiv:cs/0011004*.

[105] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Found. Trends Privacy Secur.*, vol. 2, nos. 2–3, pp. 70–246, 2018.

[106] A. Kwon, H. Corrigan-Gibbs, S. Devadas, and B. Ford, "Atom: Horizontally scaling strong anonymity," in *Proc. 26th Symp. Operating Syst. Princ.*, Oct. 2017, pp. 406–422.

[107] A. Barak, M. Hirt, L. Koskas, and Y. Lindell, "An end-to-end system for large scale P2P MPC-as-a-service and low-bandwidth MPC for weak participants," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 695–712.

[108] D. Lu, T. Yurek, S. Kulshreshtha, R. Govind, A. Kate, and A. Miller, "HoneyBadgerMPC and AsynchroMix: Practical asynchronous MPC and its application to anonymous communication," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 887–903.

[109] (2020). *HoneyBadgerMPC—Robust MPC-Based Confidentiality Layer for Blockchains*. Accessed: Jun. 2, 2022. [Online]. Available: https://github.com/initc3/HoneyBadgerMPC

[110] D. Kelly, R. Raines, R. Baldwin, M. Grimaila, and B. Mullins, "Exploring extant and emerging issues in anonymous networks: A taxonomy and survey of protocols and metrics," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 2, pp. 579–606, 2nd Quart., 2012.

[111] S. Wang, Y. Du, T. Lu, J. Wu, and T. Wang, "A survey of anonymous communication methods in Internet of Things," in *Proc. IEEE 9th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2019, pp. 627–633.

[112] A. Serjantov and P. Sewell, "Passive attack analysis for connection-based anonymity systems," in *Proc. Eur. Symp. Res. Comput. Secur. (ESORICS)*, E. Snekkenes and D. Gollmann, Eds. Berlin, Germany: Springer, 2003, pp. 116–131.

[113] I. Scholz, "Dining cryptographers—The protocol," in *Proc. 24th Chaos Commun. Congr.*, 2007, pp. 1–5.

[114] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Commun. ACM*, vol. 28, no. 10, pp. 1030–1044, Oct. 1985.

[115] A. Pfitzmann and M. Waidner, "Networks without user observability," *Comput. Secur.*, vol. 6, no. 2, pp. 158–166, Apr. 1987.

[116] H. Corrigan-Gibbs, D. I. Wolinsky, and B. Ford, "Proactively accountable anonymous messaging in verdict," in *Proc. 22nd USENIX Secur. Symp. (USENIX Security)*, 2013, pp. 147–162.

[117] A. Krasnova, M. Neikes, and P. Schwabe, "Footprint scheduling for dining-cryptographer networks," in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2016, pp. 385–402.

[118] M. Waidner and B. Pfitzmann, "The dining cryptographers in the disco: Unconditional sender and recipient untraceability with computationally secure serviceability," in *Advances in Cryptology—(EUROCRYPT)*, vol. 434. Berlin, Germany: Springer, 1989, p. 690.

[119] P. Golle and A. Juels, "Dining cryptographers revisited," in *Advances in Cryptology—(EUROCRYPT)*. Berlin, Germany: Springer, 2004, pp. 456–473.

[120] L. Barman, I. Dacosta, M. Zamani, E. Zhai, A. Pyrgelis, B. Ford, J.-P. Hubaux, and J. Feigenbaum, "PriFi: Low-latency anonymity for organizational networks," 2017, *arXiv:1710.10237*.

[121] M. Waidner, "Unconditional sender and recipient untraceability in spite of active attacks," in *Proc. Workshop Theory Appl. Cryptograph. Techn. (EUROCRYPT)*. Cham, Switzerland: Springer, 1989, pp. 302–319.

[122] C. Franck, "New directions for dining cryptographers," M.S. thesis, Dept. Comput. Sci., Univ. Luxembourg, Luxembourg, Europe, 2008.

[123] J. Bos and B. D. Boer, "Detection of disrupters in the DC protocol," in *Proc. Workshop Theory Appl. Cryptograph. Techn. (EUROCRYPT)*. Berlin, Germany: Springer, 1989, pp. 320–327.

[124] S. Goel, M. Robson, M. Polte, and E. Sirer, "Herbivore: A scalable and efficient protocol for anonymous communication," Dept. Comput. Inf. Sci., Cornell Univ., Ithaca, NY, USA, Tech. Rep. 2003-1890, 2003.

[125] J. Furukawa and K. Sako, "An efficient scheme for proving a shuffle," in *Advances in Cryptology (CRYPTO)*. Berlin, Germany: Springer, 2001, pp. 368–387.

[126] C. A. Neff, "A verifiable secret shuffle and its application to E-voting," in *Proc. 8th Conf. Comput. Commun. Secur. (CCS)*, Nov. 2001, pp. 116–125.

[127] C. Studholme and I. Blake, "Multiparty computation to generate secret permutations," Cryptol. ePrint Arch., Paper 2007/353, 2007. [Online]. Available: https://eprint.iacr.org/2007/353

[128] H. Corrigan-Gibbs and B. Ford, "Dissent: Accountable anonymous group messaging," in *Proc. CCS*, Oct. 2010, pp. 340–350.

[129] D. I. Wolinsky, H. Corrigan-Gibbs, B. Ford, and A. Johnson, "Dissent in numbers: Making strong anonymity scale," in *Proc. 10th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2012, pp. 179–182.

[130] *Scalable Anonymous Group Communication in the Anytrust Model*, Naval Res. Lab., Washington, DC, USA, 2012.

[131] M. Neikes. (2015). *DCnet-Simulator—Simulator to Explore and Compare Various Scheduling Algorithms for DC-Networks*. Accessed: Apr. 10, 2022. [Online]. Available: https://github.com/25A0/DCnet-simulator

[132] D. Mödinger, H. Kopp, F. Kargl, and F. J. Hauck, "A flexible network approach to privacy of blockchain transactions," in *Proc. 38th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1486–1491.

[133] J. Gutjuri. (2020). *Threshold Dining Cryptographers Simulation*. Accessed: May 31, 2022. [Online]. Available: https://github.com/vs-uulm/thc-in-dc-simulation

[134] B. F. L. Barman and D. Wolinsky. (2020). *PriFi, A Low-Latency, Local-Area Anonymous Communication Network*. Accessed: Apr. 10, 2022. [Online]. Available: https://github.com/dedis/prifi

[135] L. Barman. (2020). *PriFi Experiments—Raw Data and Plots for PriFi*. Accessed: Apr. 10, 2022. [Online]. Available: https://github.com/lbarman/prifi-experiments

[136] C. Franck and J. van de Graaf, "Dining cryptographers are practical," 2014, *arXiv:1402.2269*.

[137] C. Franck, "Dining cryptographers with 0.924 verifiable collision resolution," 2014, *arXiv:1402.1732*.

[138] D. Mödinger and F. J. Hauck, "3P3: Strong flexible privacy for broadcasts," in *Proc. 19th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Dec. 2020, pp. 1630–1637.

[139] D. M. A. Hess. (2020). *3P3 Evaluation*. Accessed: Apr. 10, 2022. [Online]. Available: https://github.com/vs-uulm/3p3-evaluation

[140] J. Schiller, *Mobile Communications*, 2nd ed. Reading, MA, USA: Addison-Wesley, 2003.

[141] J. Postel, "Internet protocol," Univ. Southern California, Los Angeles, CA, USA, Tech. Rep. RFC0791, Sep. 1981. [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc791.txt.pdf

[142] R. Gartner, *Metadata. Shaping Knowledge from Antiquity to the Semantic Web*. Berlin, Germany: Springer Nature, 2016.

[143] F. Wortmann and K. Fluchter, "Internet of Things," *Bus. Inf. Syst. Eng.*, vol. 57, no. 3, pp. 221–224, 2015.

[144] *Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks Next Generation Networks–Frameworks and Functional Architecture Models: Overview of the Internet of Things*, ITU Telecommunication Standardization, document ITU-T Y.2060, 2012. [Online]. Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2060-201206-I!!PDF-E&type=items

[145] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Jun. 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128610001568

[146] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Comput. Netw.*, vol. 76, pp. 146–164, Jan. 2015.

[147] M. Bishop, *Introduction to Computer Security*. Reading, MA, USA: Addison-Wesley, 2004.

[148] *Health Informatics—Trusted End-to-End Information Flows*, International Organization for Standardization (ISO), Standard ISO/TS 21089, 2018.

[149] H. C. Pöhls, "Contingency revisited: Secure construction and legal implications of verifiably weak integrity," in *Proc. IFIP Int. Conf. Trust Manage.*, vol. 401. Berlin, Germany: Springer, Jun. 2013, pp. 136–150, doi: 10.1007/978-3-642-38323-6_10.

[150] R. Staudemeyer, D. Umuhoza, and C. Omlin, "Attacker models, traffic analysis and privacy threats in IP networks," in *Proc. 12th Int. Conf. Telecommun. (ICT)*, 2005.

[151] A. Hashemi, "Fingerprinting the smart home: Detection of smart assistants based on network activity," M.S. thesis, Dept. Elect. Eng., California Polytech. State Univ., San Luis Obispo, CA, USA, 2018.

[152] A. Hintz, "Fingerprinting websites using traffic analysis," in *Proc. Int. Workshop Privacy Enhancing Technol. (PETS)*. Berlin, Germany: Springer, 2002, pp. 171–178.

[153] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2014, pp. 263–274.

[154] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Trans. Dependable Secure Comput.*, vol. 2, no. 2, pp. 93–108, Apr. 2005.

[155] O. Berthold, A. Pfitzmann, and R. Standtke, "The disadvantages of free mix routes and how to overcome them," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2001, pp. 30–45.

[156] G. Danezis and A. Serjantov, "Statistical disclosure or intersection attacks on anonymity systems," in *Proc. Int. Workshop Inf. Hiding (IH)*. Berlin, Germany: Springer, 2004, pp. 293–308.

[157] G. Danezis, "Statistical disclosure attacks," in *Security and Privacy in the Age of Uncertainty*. Boston, MA, USA: Springer, 2003, pp. 421–426.

[158] N. Mathewson and R. Dingledine, "Practical traffic analysis: Extending and resisting statistical disclosure," in *Proc. Int. Workshop Privacy Enhancing Technol. (PETS)*. Berlin, Germany: Springer, 2004, pp. 17–34.

[159] N. Apthorpe, D. Reisman, and N. Feamster, "A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic," 2017, *arXiv:1705.06805*.

MOHSEN SHIRALI received the B.Sc. degree (Hons.) in software engineering from Islamic Azad University-Tehran North Branch, in 2010, and the M.Sc. degree in computer architecture from Shahid Beheshti University (SBU), in 2013, where he is currently pursuing the Ph.D. degree, with a focus on using the Internet of Things in the health-care domain by considering privacy and energy efficiency. He worked at SBU and Ahwaz Sama University as a Lecturer and a Visiting Researcher at the Chair of IT-Security, Passau Institute of Digital Security (PIDS), from 2019 to 2020. His research stay at the University of Passau was funded by two scholarship awards from the DAAD and Iran MSRT. His research interests include the Internet of Things, security and privacy, anonymous communication, and process mining.

TOBIAS TEFKE received the B.Sc. degree in computer science from the Schmalkalden University of Applied Sciences, Germany, in 2022, where he is currently pursuing the degree in applied computer science. He enjoys software development. His research interests include IT security and privacy, particularly on mobile and constrained devices.

RALF C. STAUDEMEYER received the Ph.D. and German Diploma degrees in computer science (diplom-informatiker). He is currently a Professor of IT security with the Faculty of Computer Science, Schmalkalden University of Applied Sciences, Germany. He is also a designated expert in IT security and privacy, computer networks, and deep learning. He lectured at several academic institutions in Germany, South Africa, and the Fiji Islands. He is the author of more than 30 publications. He was a recipient of various research fellowships and supported EU research projects as a coordinator, a work-package leader, and a researcher.

HENRICH C. PÖHLS received the Diploma degree in computer science from the University of Hamburg, Germany, the master's degree in information security from RHUL, U.K., and the Ph.D. degree from the University of Passau, Germany. He is currently a Senior Researcher and a Lecturer on secure AI systems and other topics with the University of Passau. His research interests include the IoT, AI, and cloud security, with a focus on cryptographic integrity protection mechanisms. He serves on several security conferences' committees and is an Appointed Expert of the German National Standardization Body (DIN) in the areas of information security (SC27) and artificial intelligence (SC42). He is an Editor of several ISO/IEC standards, such as ISO/IEC 23264-1 on the redaction of authentic data.