# Distributed Event-Triggered Algorithms for Finite-Time Privacy-Preserving Quantized Average Consensus

Apostolos I. Rikos, *Member, IEEE*, Themistoklis Charalambous, *Senior Member, IEEE*, Karl Henrik Johansson, *Fellow, IEEE*, and Christoforos N. Hadjicostis, *Fellow, IEEE*

***Abstract*—In this article, we consider the problem of privacy preservation in the average consensus problem when communication among nodes is quantized. More specifically, we consider a setting where nodes in the network can be curious, while certain nodes in the network want to ensure that their initial states cannot be inferred exactly by these curious nodes. Curious nodes are not malicious, i.e., they try to identify the initial states of other nodes based on the data they receive during their operation (and some of them might even collude) but do not interfere in the computation in any other way. Each node in the network (including curious nodes) can opt to execute a privacy-preserving algorithm (so as not to reveal the initial state it contributes to the average calculation) or its underlying (plain) average consensus algorithm (if privacy is not a concern). We propose two privacy-preserving event-triggered quantized average consensus algorithms. Under certain topological conditions, both the proposed algorithms allow the nodes that adopt privacy-preserving protocols to preserve their privacy and at the same time to obtain, after a finite number of steps, the exact average of the initial states while processing and transmitting *quantized* information. We also present illustrative examples and comparisons of our algorithms against other algorithms in the existing literature and discuss a motivating application in which smart meters in a smart grid collect real-time demands in a privacy-preserving manner.**

***Index Terms*—Finite-time convergence, privacy-preserving average consensus, quantized communication.**

Manuscript received 12 July 2021; revised 1 November 2021, 5 November 2021, and 29 April 2022; accepted 5 June 2022. Date of publication 23 June 2022; date of current version 13 March 2023. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Grant 739551 (KIOS Research and Innovation Center of Excellence), in part by the Knut and Alice Wallenberg Foundation, and in part by the Swedish Research Council. An earlier version of this paper was presented at the 2020 IEEE Conference on Decision and Control [doi: 10.1109/CDC42340.2020.9303771]. We extend the results of the earlier version by: 1) proposing a second privacy-preserving algorithm and deriving its required topological conditions; 2) extending the simulation section, where we also compare the two proposed algorithms; and 3) presenting an application for computing power requests in smart grids in a privacy-preserving manner. Recommended by Associate Editor Y. Wang. *(Corresponding author: Apostolos I. Rikos.)*

Apostolos I. Rikos and Karl Henrik Johansson are with the Division of Decision and Control Systems, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden, and also with Digital Futures, SE-100 44 Stockholm, Sweden (e-mail: rikos@kth.se; kallej@kth.se).

Themistoklis Charalambous is with the Department of Electrical and Computer Engineering, School of Engineering, University of Cyprus, 1678 Nicosia, Cyprus, and also with the Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, 02150 Espoo, Finland (e-mail: charalambous.themistoklis@ucy.ac.cy).

Christoforos N. Hadjicostis is with the Department of Electrical and Computer Engineering, University of Cyprus, 1678 Nicosia, Cyprus (e-mail: chadjic@ucy.ac.cy).

Digital Object Identifier 10.1109/TCNS.2022.3185561

## I. Introduction

A PROBLEM of particular interest in distributed control is the *consensus* problem, in which nodes communicate locally with other nodes under constraints on connectivity [2]. In distributed averaging (a special case of the consensus problem), each node is initially endowed with a numerical state, which it updates in an iterative fashion by sending/receiving information to/from other neighboring nodes, so that it eventually computes the average of all the initial states. Average consensus has been studied extensively in settings, where each node processes and transmits real-valued states with infinite precision; see, for example, [3] and references therein.

The case where capacity-limited network links can only allow the messages of certain length to be transmitted between nodes has also received significant attention recently, as it effectively extends techniques for average consensus toward quantized consensus. Quantized processing and communication is better suited to the available network resources (e.g., physical memories of finite capacity and digital communication channels of limited data rate), while it also exhibits other advantages such as amenability to security and privacy enhancements [4]. For example, public-key cryptosystems require integer numbers to operate with, since nonquantized consensus algorithms would be subject to quantization errors in the final result [5], [6]. For these reasons, several probabilistic and deterministic strategies have been proposed for solving the quantized average consensus problem [7]–[10].

Average consensus algorithms require each node to exchange and disclose state information to its neighbors. This may be undesirable in case the state of some nodes is private or contains sensitive information. In addition, in many occasions, there might be nodes in the network that are curious and aim to extract private and/or sensitive information. In many emerging applications (e.g., healthcare and opinion forming over social networks), preserving the privacy of participating components is necessary for enabling cooperation between nodes without requiring them to disclose sensitive information. There have been different approaches for dealing with privacy preservation in such systems. For example, Kefayati *et al.* [11] proposed a method in which each node wishing to protect its privacy adds a random offset value to its initial state, thus ensuring that its true state will not be revealed to curious nodes that might be observing the exchange of data in the network. Kia *et al.* [12] calculate the dynamic average of the initial states in a privacy-preserving manner. They show that their algorithm tracks the average of the dynamic inputs with some steady-state error (which vanishes for special classes of input signals). Furthermore: 1) they analyze the algorithm under time-varying interaction topologies; 2) they present an extension that allows each node to control its own rate of convergence; and 3) they characterize the privacy preservation properties of the proposed algorithm. A related line of research is based on differential privacy [13], [14], in which nodes inject uncorrelated noise into the exchanged messages so that the data associated with a particular node cannot be inferred by a curious node during the execution of the algorithm. However, the exact average state is not eventually obtained due to the induced tradeoff between privacy and computational accuracy [14]. To overcome this tradeoff and guarantee convergence to the exact average, the injection of correlated noise at each time step and for a finite period of time was proposed in [15], thus allowing a node to avoid revealing its own initial state or the initial states of other nodes. Once this period of time ends, each node ensures that the accumulated sum of offsets it added in the iterative computation is removed. In [16], the nodes asymptotically subtract the initial offset values they added in the computation, while in [17], each node masks its initial state with an offset such that the sum of the offsets of each node is zero, thus guaranteeing convergence to the average. Rezazadeh and Kia [18] discuss the problem of calculating the average of the initial states over a continuous-time and weight-balanced system via perturbation signals (i.e., each node is adding admissible perturbation signals to the local dynamics and the signals that are transmitted by the agents). In [19], the average of the initial states over directed graphs is calculated in a privacy-preserving manner by using the consensus dynamics in order to embed privacy in random coupling weights between connected nodes. In [20], the average of the initial states is calculated in a privacy-preserving manner via a state-decomposition-based approach, where each node decomposes its state into two substates (whose mean is equal to the original state). Then, one of the two substates is used for computation and internode interactions, while the other substate interacts only with the first substate of the same node. Ridgley *et al.* [21] discuss the problem of calculating the dynamic average of the initial states in a privacy-preserving manner under certain topological conditions (i.e., privacy is preserved if each node has at least two out-neighbors and at least one of them is not colluding with other nodes). Furthermore, this strategy is hot pluggable, which means that the algorithm does not need reinitialization when there is a change over the network (e.g., when a node enters or leaves the network). Another approach that guarantees privacy preservation is via homomorphic encryption [6]. However, this approach requires the existence of trusted nodes and imposes heavier computational requirements on the nodes.

## A. Main Contributions

The main contributions of this article are as follows: 1) two novel distributed algorithms are proposed and shown to achieve quantized average consensus under privacy constraints and to converge after a finite number of time steps and 2) the application of these algorithms to power requests in smart grids under privacy-preserving requirements is demonstrated.

During its operation, each node that would like to protect its privacy from other curious (but not malicious) nodes follows one of the two finite-time event-triggered quantized average consensus protocols. The privacy-preserving algorithms essentially involve adding and subtracting offsets to each node's state in two different ways.

1) *The first algorithm injects offsets according to an event-based strategy for a predefined number of steps.* Specifically, when the token that triggers action arrives at a specific node for the first time, the node adds a substantial negative *quantized* offset to its initial state. This initial offset is determined by the node at the first triggering and is gradually removed at later triggerings (when certain conditions are satisfied) ensuring that the total accumulated sum of injected offsets is canceled out.
2) *The second algorithm injects offsets only during the initialization procedure.* Specifically, each node injects a quantized offset to the states of its out-neighboring nodes only during the initialization procedure. Then, the node injects to its own state a (possibly) nonzero offset (in addition to any offsets injected by its in-neighbors) such that the accumulated sum of the injected offsets is equal to zero and proceeds with executing a finite-time event-triggered quantized average consensus protocol.

We show that both algorithms converge after a finite number of time steps. We also present the topological conditions that ensure privacy for the nodes that follow the proposed protocols. Note that the operation of our algorithms relies on the zero-sum offset strategy, but the main challenges in this article are the following: 1) communication-efficient operation; 2) finite-time convergence; 3) enhancing the speed of convergence; and 4) improving the topological conditions for privacy preservation. In order to address these challenges, our algorithms need to operate with quantized information, while they converge in finite time with no quantization error.

Unlike other privacy-preserving protocols proposed in the literature (see, e.g., [11], [15], and [16]), the proposed algorithms take advantage of their finite-time operation since the added offsets are integers. As a result, consensus to the *exact* average of the initial states is achieved after a finite number of steps, while the error, introduced from the offset, vanishes completely.

A motivating application for our proposed algorithms is discussed in Appendix A. In this application, a neighborhood of interconnected households is able to request the *total* demanded power from a smart meter over a network of next-generation power systems [22] in a privacy-preserving manner.

### B. Organization

The rest of this article is organized as follows. In Section II, we review necessary notation and background, while in Section III, we provide the problem formulation. In Sections IV and V, we present our first and second privacy strategies, respectively, along with the corresponding distributed algorithms. Furthermore, we analyze their convergence and present sufficient topological conditions that ensure privacy preservation. In Section V, we present our second privacy strategy and the corresponding distributed algorithm, analyze its convergence, and present sufficient topological conditions that ensure privacy preservation. In Section VI, we demonstrate our strategies via illustrative examples and compare their operation against other algorithms in the current literature. Finally, Section VII conclude this article.

## II. Notation and Preliminaries

*Notation:* The sets of real and integer numbers are denoted by $\mathbb{R}$ and $\mathbb{Z}$, respectively. The symbols $\mathbb{Z}_{\geq 0}$ ($\mathbb{Z}_{>0}$) and $\mathbb{Z}_{\leq 0}$ ($\mathbb{Z}_{<0}$) denote the sets of non-negative (positive) and nonpositive (negative) integers, respectively. Vectors are denoted by small letters, whereas matrices are denoted by capital letters. The transpose of a matrix $A$ is denoted by $A^T$. For $A \in \mathbb{R}^{n\times n}$, $A_{ij}$ denotes the entry at row $i$ and column $j$. By $\mathbf{1}$, we denote the all-ones vector, and by $I$, we denote the identity matrix (of appropriate dimensions).

### A. Graph Theory

Consider a network of $n$ ($n \geq 2$) nodes communicating only with their immediate neighbors. The communication topology can be captured by a directed graph (digraph), called *communication digraph*. A digraph is defined as $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ with cardinality $n = |\mathcal{V}| \geq 2$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \setminus \{(v_j, v_j) \mid v_j \in \mathcal{V}\}$ is the set of edges (self-edges excluded) whose cardinality is denoted by $m = |\mathcal{E}|$. A directed edge from node $v_i$ to node $v_j$ is denoted by $m_{ji} \triangleq (v_j, v_i) \in \mathcal{E}$ and captures the fact that node $v_j$ can receive information from node $v_i$ (but not the other way around). We assume that the given digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ is *strongly connected* (i.e., for each pair of nodes $v_j, v_i \in \mathcal{V}$, $v_j \neq v_i$, there exists a directed *path*[1] from $v_i$ to $v_j$). The subset of nodes that can directly transmit information to node $v_j$ is called the set of in-neighbors of $v_j$ and is represented by $\mathcal{N}_j^- = \{v_i \in \mathcal{V} \mid (v_j, v_i) \in \mathcal{E}\}$, while the subset of nodes that can directly receive information from node $v_j$ is called the set of out-neighbors of $v_j$ and is represented by $\mathcal{N}_j^+ = \{v_l \in \mathcal{V} \mid (v_l, v_j) \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^-$ is called the *in-degree* of $v_j$ and is denoted by $\mathcal{D}_j^-$ (i.e., $\mathcal{D}_j^- = |\mathcal{N}_j^-|$), while the cardinality of $\mathcal{N}_j^+$ is called the *out-degree* of $v_j$ and is denoted by $\mathcal{D}_j^+$ (i.e., $\mathcal{D}_j^+ = |\mathcal{N}_j^+|$).

### B. Node Operation

With respect to the quantization of information flow, we have that at time step $k \in \mathbb{Z}_{\geq 0}$, each node $v_j \in \mathcal{V}$ maintains the state variables $y_j^s[k]$, $z_j^s[k]$, and $q_j^s[k]$, where $y_j^s[k] \in \mathbb{Z}$, $z_j^s[k] \in \mathbb{Z}_{>0}$, and $q_j^s[k] = y_j^s[k]/z_j^s[k]$, and the mass variables $y_j[k]$ and $z_j[k]$, where $y_j[k] \in \mathbb{Z}$ and $z_j[k] \in \mathbb{Z}_{\geq 0}$. Note here that $q^s$ is a memoryless function of the states $y^s$ and $z^s$ (which would make it an output and not a state). We assume that each node is aware of its out-neighbors and can directly transmit messages to each of them. However, it cannot necessarily receive messages (at least not directly) from them. In the proposed distributed protocols, each node $v_j$ assigns a *unique order* in the set $\{0, 1, \ldots, \mathcal{D}_j^+ - 1\}$ to each of its outgoing edges $m_{lj}$, where $v_l \in \mathcal{N}_j^+$. More specifically, the order of link $(v_l, v_j)$ for node $v_j$ is denoted by $P_{lj}$ (such that $\{P_{lj} \mid v_l \in \mathcal{N}_j^+\} = \{0, 1, \ldots, \mathcal{D}_j^+ - 1\}$). This unique predetermined order is used during the execution of the proposed distributed algorithm as a way of allowing node $v_j$ to transmit messages to its out-neighbors in a *round-robin*[2] fashion.

### C. Quantized Averaging via Deterministic Mass Summation

Quantized average consensus algorithms allow nodes to process and transmit quantized information, so that they have short communication packages and eventually obtain a fraction $q^s$ that is equal to the *exact* average of the initial quantized states of the nodes, after a finite number of steps. Note that the calculation of the *exact* average of the initial values without any error is due to the quantized nature of the initial states.

Following [10], we assume that each node $v_j$ in the network has a quantized initial state $y_j[0] \in \mathbb{Z}$. At each time step $k$, each node $v_j \in \mathcal{V}$ maintains its mass variables $y_j[k] \in \mathbb{Z}$ and $z_j[k] \in \mathbb{Z}_{\geq 0}$ and its state variables $y_j^s[k] \in \mathbb{Z}$, $z_j^s[k] \in \mathbb{Z}_{>0}$, and $q_j^s[k] = y_j^s[k]/z_j^s[k]$. It updates the values of the mass variables as

$$y_j[k+1] = y_j[k] + \sum_{v_i \in \mathcal{N}_j^-} \mathbb{1}_{ji}[k] y_i[k] \tag{1a}$$

$$z_j[k+1] = z_j[k] + \sum_{v_i \in \mathcal{N}_j^-} \mathbb{1}_{ji}[k] z_i[k] \tag{1b}$$

where $\mathbb{1}_{ji}[k] = 0$ if no message is received at node $v_j$ from its in neighbor $v_i$ at iteration $k$ (else $\mathbb{1}_{ji}[k] = 1$). If *either* of the following event-triggered conditions:

(C1): $z_j[k+1] > z_j^s[k]$
(C2): $z_j[k+1] = z_j^s[k]$ and $y_j[k+1] \geq y_j^s[k]$

is satisfied, node $v_j$ updates its state variables as follows:

$$z_j^s[k+1] = z_j[k+1] \tag{2a}$$

$$y_j^s[k+1] = y_j[k+1] \tag{2b}$$

$$q_j^s[k+1] = \frac{y_j^s[k+1]}{z_j^s[k+1]}. \tag{2c}$$

Then, it transmits $y_j[k+1]$ and $z_j[k+1]$ to an out-neighbor $v_l \in \mathcal{N}_j^+$ (chosen according to the unique order in Section II-B) and sets $y_j[k+1] = 0$ and $z_j[k+1] = 0$.

[1] A directed *path* from $v_i$ to $v_j$ exists if we can find a sequence of nodes $v_i \equiv v_{l_0}, v_{l_1}, \ldots, v_{l_t} \equiv v_j$ such that $(v_{l_{\tau+1}}, v_{l_\tau}) \in \mathcal{E}$ for $\tau = 0, 1, \ldots, t-1$.

[2] When executing the protocol, each node $v_j$ transmits to its out-neighbors, one at a time, by following the predetermined order. The next time it transmits to an out-neighbor, it continues from the outgoing edge it stopped the previous time and cycles through the edges in a round-robin fashion.

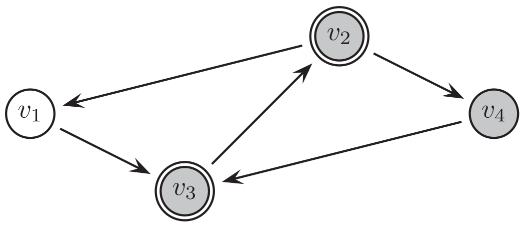

Fig. 1. Example of a digraph with different types of nodes in the network, where $\mathcal{V}_c = \{v_2, v_3\}$ and $\mathcal{V}_p = \{v_2, v_3, v_4\}$.

***Definition 1:*** Quantized average consensus is achieved if there exists $k_0$ so that for every $k \geq k_0$, we have

$$q_j^s[k] = \frac{\sum_{l=1}^{n} y_l[0]}{n} =: \overline{y} \tag{3}$$

for every $v_j \in \mathcal{V}$.

The following result from [10] provides a worst-case upper bound regarding the number of time steps required for quantized averaging to be achieved. Note here that the operation of the algorithm in [10] guarantees finite-time convergence, but due to the equalities in the event-triggered conditions in (C1) and (C2), some nodes may keep communicating even after consensus has been achieved [10].

***Theorem 1 (see [10]):*** The system (1), (2) reaches quantized average consensus after a finite number of steps $\mathcal{S}_t \leq nm^2$, where $n$ is the number of nodes and $m$ is the number of edges in the network.

## III. Problem Formulation

Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$, where $|\mathcal{V}| \geq 3$. Each node $v_j \in \mathcal{V}$ has an initial quantized state $y_j[0]$ (for simplicity, we take $y_j[0] \in \mathbb{Z}$). Nodes require to calculate

$$\overline{y} = \frac{\sum_{l=1}^{n} y_l[0]}{n} \tag{4}$$

in a distributed way, exclusively through local exchange of information. The information exchange takes place only between nodes that are neighbors with respect to $\mathcal{G}_d$, which represents the system communication topology. Any node in the set $\mathcal{V}$ could be curious, which means that it tries to identify the initial states $y[0]$ of all or a subset of nodes in the network. The set $\mathcal{V}$ is partitioned in two ways: 1) a subset of nodes $\mathcal{V}_p \subseteq \mathcal{V}$ that wish to preserve their privacy, i.e., node $v_j \in \mathcal{V}_p$, does not want to reveal its initial state $y_j[0]$ to other nodes (the remaining nodes in the set $\mathcal{V}_n = \mathcal{V} \setminus \mathcal{V}_p$ are indifferent about their privacy); and 2) a subset of nodes $\mathcal{V}_c \subset \mathcal{V}$ are curious and collude among themselves in order to identify the initial values of other nodes (nodes in $\mathcal{V} \setminus \mathcal{V}_c$ could also be curious but do not collude among themselves). Note here that nodes in $\mathcal{V}_c$ are assumed not to care about their privacy (since they may have to share their initial states with other nodes in $\mathcal{V}_c$). An example is shown in Fig. 1.

The concept of privacy is typically defined as the ability of an individual node to seclude itself or hide information about itself and thereby express itself selectively. In our case, we consider that the information of interest for each node is its initial state $y_j[0]$. The notion of privacy that we adopt aims to ensure that the state $y_j[0]$ cannot be inferred exactly by curious nodes and relates to notions of possible innocence in theoretical computer science [23], [24] in the sense that there is some uncertainty about $y_j[0]$.

***Definition 2:*** A node $v_j \in \mathcal{V}_p \setminus \mathcal{V}_c$ is said to preserve the privacy of its initial state $y_j[0] \in \mathbb{Z}$ if the value $y_j[0]$ cannot be inferred by any curious node in $\mathcal{V} \setminus \{v_j\}$ (or even by the colluding nodes in $\mathcal{V}_c$) at any point during the operation of the protocol, i.e., the curious nodes can only determine a range $[\alpha, \beta]$, $\alpha < \beta$, for which they can ensure that the values $y_j[0]$ lie in, and $v_j$ can make $\alpha \in \mathbb{R}$ arbitrarily small and/or $\beta \in \mathbb{R}$ arbitrarily large.

The problem we consider in this article is to develop a strategy for nodes $v_j \in \mathcal{V}_p \setminus \mathcal{V}_c$ that wish to preserve their privacy (i.e., not reveal their initial states $y_j[0]$ to other nodes) when they exchange quantized information with neighboring nodes while calculating $\overline{y}$ in (4). Note that this strategy should allow nodes to operate seamlessly along with those that use the underlying average consensus algorithm described in Section II-B that is not privacy preserving. As aforementioned, any node could be curious and try to identify the initial states of other nodes, but *no node interferes* in the computation in any other way (i.e., each node either executes a privacy-preserving strategy or the underlying (plain) average consensus algorithm in Section II-B). Curious nodes are aware of the predefined algorithm followed by nodes that would like to preserve their privacy, and the topology of the network, but not the actual parameters chosen by the nodes $v_j \in \mathcal{V}_p \setminus \mathcal{V}_c$ that want to preserve their privacy. We also assume that curious nodes in the subset $\mathcal{V}_c$ may collaborate arbitrarily. Finally, we assume that nodes in the set $\mathcal{V} \setminus \mathcal{V}_p$ simply execute the underlying average consensus algorithm described in [10] and reviewed in Section II-B.

***Remark 1:*** Definition 2 implies that if one looks at the set of variables that are *a priori* unknown to the curious nodes (e.g., the initial states of nodes, offsets chosen by other nodes, etc.), then one can find different sets of values for these variables that match the observations that become available to the curious nodes (including the eventual knowledge of the average of the initial states of the nodes), such that the node that wants to preserve its privacy exhibits different initial states in these two sets.

## IV. Event-Based Privacy-Preserving Strategy

### A. Initialization for Quantized Privacy-Preserving Strategy

The primary objective in our system is to calculate $\overline{y}$ in (4) while preserving the privacy of at least the nodes following the protocol. Our strategy is based on the event-triggered deterministic algorithm (1), (2) with some modifications (since the event-triggered deterministic algorithm (1), (2) is not privacy preserving). The main difference is that a mechanism is deployed that incorporates an offset to the mass variable of each node $v_j \in \mathcal{V}_p$, effectively preserving the privacy of its initial state $y_j[0]$.

In previous works (see, e.g., [11], [15], [16], and references therein), node $v_j$ sets its initial state to $\widetilde{y}_j[0] = y_j[0] + u_j$, where $u_j \in \mathbb{R}$. However, in this case, we require that the initial offset $u_j$ is a negative integer number, i.e., $u_j \in \mathbb{Z}_{<0}$, so that the event-triggered conditions (C1) and (C2) are guaranteed to lead to the calculation of the initial average after a finite number of time steps. Furthermore, each node $v_j$ maintains the privacy values $u_j[k] \in \mathbb{Z}_{\geq 0}$, the offset adding steps $L_j \in \mathbb{Z}_{>0}$, the offset adding counter $l_j \in \mathbb{Z}_{>0}$, and its transmission counter $c_j \in \mathbb{Z}_{>0}$. The absolute value of the initial offset $u_j$ and the number of offset adding steps $L_j$ need to be chosen to be greater than the number of out-neighbors $\mathcal{D}_j^+$ of node $v_j$. Specifically, at initialization, each node $v_j$ chooses the number of steps $L_j$ and the offsets $u_j \in \mathbb{Z}_{<0}$, and $u_j[l_j] \in \mathbb{Z}_{\geq 0}$ for all $l_j \in \{0, 1, 2, \ldots, L_j\}$, to satisfy

$$L_j \geq \mathcal{D}_j^+ \tag{5a}$$

$$u_j = -\sum_{l_j=0}^{L_j} u_j[l_j] \tag{5b}$$

$$u_j[l_j] \geq 0 \quad \forall\, l_j \in [0, L_j] \tag{5c}$$

$$u_j[l_j] = 0 \quad \forall\, l_j \notin [0, L_j]. \tag{5d}$$

Constraints (5a)–(5d) are explicitly analyzed below.

1) In (5a), the offset adding steps $L_j$ of every node $v_j$ need to be greater than or equal to node $v_j$'s out-degree so that every out-neighbor $v_i \in \mathcal{N}_j^+$ will receive at least one value of $u_j[l_j]$ from node $v_j$. As discussed in Section IV-D, this is motivated by the privacy preservation guarantees.
2) Equation (5b) means that the accumulated offset infused in the computation by node $v_j$ is equal to zero, and the exact quantized average of the nodes' initial states can be calculated eventually without any error.
3) In (5c), the offset $u_j[l_j]$ that is injected to the network by each node $v_j$ each time its event-triggered conditions hold (i.e., for events $l_j \in [0, L_j]$) needs to be non-negative so that: a) the event-triggered conditions (C1) and (C2) hold for every node after a finite number of steps and b) the exact quantized average of the initial states can be eventually calculated.
4) Equation (5d) means that node $v_j$ stops injecting nonzero offsets in the network so that the exact quantized average of the initial states can be calculated without any error.

The above choices imply that the initial offset $u_j$ every node $v_j$ injects in the network needs to be chosen so that it is negative and satisfies $u_j \leq -\mathcal{D}_j^+$. This is important to ensure that, during the operation of the proposed algorithm, the event-triggered conditions (C1) and (C2) hold for every node after a finite number of steps. If $u_j \geq 0$, the event-triggered conditions (C1) and (C2) may not hold, and the proposed protocol may fail to calculate the average of the initial states.

## B. Algorithm Description

The proposed algorithm is a quantized value transfer process, in which every node in a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ performs operations and transmissions according to a set of event-triggered conditions. The intuition behind the algorithm is as follows. Each node $v_j \in \mathcal{V}_p$ that would like to preserve its privacy performs the following steps.

1) It initializes a counter $l_j$ to zero (i.e., $l_j = 0$) and chooses the total number of offset adding steps $L_j$ such that $L_j \geq \mathcal{D}_j^+$ and the set of $(L_j + 1)$ positive offsets $u_j[l_j] > 0$, where $l_j \in \{0, 1, \ldots, L_j\}$. Finally, it sets the initial negative offset $u_j$ that it injects to its initial state $y_j[0]$ to $u_j = -\sum_{l_j=0}^{L_j} u_j[l_j]$. For example, suppose that node $v_j$ has four out-neighbors. This means that it can choose $L_j = 6$ and then (randomly) set $u_j[0] = 1$, $u_j[1] = 3$, $u_j[2] = 2$, $u_j[3] = 4$, $u_j[4] = 1$, and $u_j[5] = 2$, $u_j[6] = 5$; finally, it sets $u_j = -18$.
2) It chooses an out-neighbor $v_l \in \mathcal{N}_j^+$ according to the unique order $P_{lj}$ (initially, it chooses $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = 0$) and transmits $z_j[0]$ and $\widetilde{y}_j[0] = y_j[0] + u_j + u_j[0]$ to this out-neighbor. Then, it sets $\widetilde{y}_j[0] = 0$, $z_j[0] = 0$, and $l_j = l_j + 1$.
3) During the execution of the algorithm, at every step $k$, node $v_j$ may receive a set of mass variables $\widetilde{y}_i[k]$ and $z_i[k]$ from each in-neighbor $v_i \in \mathcal{N}_j^-$. Then, node $v_j$ updates its mass variables according to (1a) and (1b) (where in the sum of (1a), we use $\widetilde{y}_j[k]$) and checks if either of its event-triggered conditions (C1) and (C2) holds. If so, it injects an offset $u_j[l_j]$ to $y_j[k+1]$ and increases its offset increasing counter $l_j$ by 1. Then, a) it sets its state variables $y_j^s[k+1]$ and $z_j^s[k+1]$ equal respectively to $\widetilde{y}_j[k+1]$ and $z_j[k+1]$ [i.e., it updates them according to (2)]; and b) it transmits $\widetilde{y}_j[k+1]$ and $z_j[k+1]$ to an out-neighbor according to the predetermined order. If none of the conditions (C1) and (C2) holds, node $v_j$ stores $y_j[k+1]$ and $z_j[k+1]$. If no message is received from any of the in-neighbors and no transmission takes place, the mass variables remain the same.

***Remark 2:*** It is important to note that during the operation of Algorithm 1, each node $v_j$ chooses the values $u_j[k]$, in a way that ensures that curious nodes can only determine a range $[\alpha, \beta]$, $\alpha < \beta$, for which they can ensure that the values $y_j[0]$ lie in, and $v_j$ can make $\alpha \in \mathbb{R}$ arbitrarily small and/or $\beta \in \mathbb{R}$ arbitrarily large.

The proposed algorithm is summarized in Algorithm 1. Note here that curious nodes that try to identify the initial states of other nodes follow either Algorithm 1 or the underlying (plain) average consensus algorithm in Section II-B.

***Remark 3:*** Unlike other privacy-preserving protocols proposed in the literature (see, e.g., [11], [15], and [16]), the proposed strategy takes full advantage of the algorithm's finite-time nature, which means that consensus to the average of the initial states is reached after a finite number of iterations, while the error, introduced via the offset initially infused in the network by the nodes following the protocol, vanishes.

## C. Convergence Analysis

For the development of the necessary results regarding the operation of Algorithm 1, let us consider the following setup, the analysis of which for the non-privacy-preserving case can be found in [10].

*Setup:* Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. During the execution of Algorithm 1, at time step $k_0$, there is at least one node $v_{j'} \in \mathcal{V}$, for which

$$z_{j'}[k_0] \geq z_i[k_0] \quad \forall v_i \in \mathcal{V}. \tag{6}$$

**Algorithm 1:** Privacy-Preserving Event-Triggered Quantized Average Consensus With Event-Based Offset.

```
Input: A strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with
  $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$
  has an initial state $y_j[0] \in \mathbb{Z}$.
Initialization: Each node $v_j \in \mathcal{V}_p$ does the following:
1) It assigns a unique order $P_{lj}$ in the set
  $\{0, 1, \ldots, \mathcal{D}_j^+ - 1\}$ to each of its out-neighbors
  $v_l \in \mathcal{N}_j^+$.
2) It sets counter $c_j$ to 0 and priority index $e_j$ to $c_j$.
3) It sets counter $l_j$ to 0, chooses $L_j \in \mathbb{Z}_{>0}$, where
  $L_j \geq \mathcal{D}_j^+$, and $u_j[k] \geq 0$ for $k \in \{0, 1, \ldots, L_j\}$, and
  $u_j[k'] = 0$ for $k' > L_j$. It also sets $u_j = -\sum_{l_j=0}^{L_j} u_j[l_j]$.
4) It sets $\widetilde{y}_j[0] = y_j[0] + u_j$, $z_j[0] = 1$, $z_j^s[0] = 1$ and
  $y_j^s[0] = \widetilde{y}_j[0]$ (which means that $q_j^s[0] = \widetilde{y}_j[0]/1$).
5) It selects out-neighbor $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = e_j$
  and transmits $z_j[0]$ and $\widetilde{y}_j[0] + u_j[0]$ to this
  out-neighbor. Then, it sets $\widetilde{y}_j[0] = 0$, $z_j[0] = 0$,
  $l_j = l_j + 1$.
6) It sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
Iteration: For $k = 0, 1, 2, \ldots$, each node $v_j \in \mathcal{V}_p$ does
  the following:
• if it receives $\widetilde{y}_i[k]$, $z_i[k]$ from at least one in-neighbor
  $v_i \in \mathcal{N}_j^-$ then it updates its values according to (1a) and
  (1b).
  ∘ if either of the conditions (C1) and (C2) hold then
    - it sets $\widetilde{y}_j[k+1] = u_j[l_j] + y_j[k+1]$ and $l_j \leftarrow l_j + 1$;
    - it sets $z_j^s[k+1] = z_j[k+1]$, $y_j^s[k+1] = \widetilde{y}_j[k+1]$ and
      $q_j^s[k+1] = \widetilde{y}_j[k+1]/z_j^s[k+1]$;
    - it transmits $z_j[k+1]$ and $\widetilde{y}_j[k+1]$ to out-neighbor
      $v_\lambda \in \mathcal{N}_j^+$ for which $P_{\lambda j} = e_j$ and it sets $\widetilde{y}_j[k+1] = 0$,
      $y_j[k+1] = 0$ and $z_j[k+1] = 0$;
    - it sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
  ∘ else it stores $y_j[k+1]$ and $z_j[k+1]$.
Output: $q_j^s[k]$, for every $v_j \in \mathcal{V}$.
```

Then, among the nodes $v_{j'}$ for which (6) holds, there is at least one node $v_j$ for which

$$\widetilde{y}_j[k_0] \geq \widetilde{y}_{j'}[k_0],\ v_j, v_{j'} \in \{v_i \in \mathcal{V} \mid (6) \text{ holds}\}. \quad (7)$$

For notational convenience, we will call the mass variables of node $v_j$ for which (6) and (7) hold as the "leading mass" (or "leading masses"). Now, we present the following two lemmas, which are helpful in the development of our results.

***Lemma 1 (see [10]):*** The "leading mass" or "leading masses" at time step $k$ will always fulfill the "event-triggered conditions" (C1) and (C2). This means that the mass variables of node $v_j$ for which (6) and (7) hold at time step $k_0$ will be transmitted (at time step $k_0$) by $v_j$ to an out-neighbor $v_l \in \mathcal{N}_j^+$.

***Lemma 2:*** If the event-triggered conditions of node $v_j \in \mathcal{V}_p$ are fulfilled in at least $(L_j + 1)$ instances, then, from (5a)–(5d), we have that the accumulated amount of offset injected by node $v_j$ to the network becomes equal to zero.

***Proof:*** Each node $v_j \in \mathcal{V}_p$ at time step $k$ adds the offset $u_j[l_j]$ to its mass variable $y_j[k]$ if and only if either of the event-triggered conditions (C1) and (C2) holds. As a result, if the event-triggered conditions of $v_j$ are fulfilled for at least $(L_j + 1)$ instances, then the accumulated amount of offset node $v_j$ has injected in the computation becomes equal to zero. ■

The following theorem states that the proposed algorithm allows all the nodes to reach quantized average consensus after a finite number of steps, for which we provide an upper bound.

***Theorem 2:*** Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. The execution of Algorithm 1 allows each node $v_j \in \mathcal{V}$ to reach quantized average consensus after a finite number of steps, $\mathcal{S}_t$, bounded by $\mathcal{S}_t \leq m^2(L_{\max} + 1 + n)$, where $n$ is the number of nodes, $m$ is the number of edges in the network, and $L_{\max} = \max_{v_j \in \mathcal{V}} L_j$ is the maximum value of offset adding steps chosen by nodes in the network.

***Proof:*** See the proof of [1, Th. 2]. ■

***Remark 4:*** Theorem 2 relies on the fact that if the proposed distributed protocol is executed for a finite number of time steps equal to $m^2 L_{\max}$, then every node in the network will receive a set of nonzero mass variables that are equal to the leading mass for at least $L_{\max}$ instances. This means that the event-triggered conditions will be fulfilled for each node in the network for at least $L_{\max}$ instances, and from Lemma 2, the accumulated amount of offset injected in the network from each node is equal to zero. As a result, by executing the proposed protocol for an additional number of time steps equal to $nm^2$, we have that every nonzero mass in the network will merge to one leading mass (or multiple equally valued leading masses) that is (are) equal to the average of the initial states, and subsequently, this (these) leading mass (masses) will update the state variables of each node in the network, setting them equal to the average of the initial states.

## D. Topological Conditions for Privacy Preservation

***Proposition 1:*** Consider a fixed strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes, where $|\mathcal{V}| \geq 3$. Assume that a subset of nodes $\mathcal{V}_p$ follows the predefined privacy-preserving protocol, as described in Algorithm 1, with node $v_j \in \mathcal{V}_p$ wishing to preserve its privacy by choosing offsets as in (5a)–(5d). Any arbitrary subset of colluding curious nodes $\mathcal{V}_c \subseteq \mathcal{V} \setminus \{v_j\}$ (including any individual curious node $v_c \in \mathcal{V} \setminus \{v_j\}$), will not be able to identify the initial state $y_j[0]$ of $v_j$, as long as $v_j$ has:

1) at least one in- or out-neighbor $v_\ell \in \mathcal{V}_p \setminus (\mathcal{V}_c \cup \{v_j\})$ connected to it, and there is a message exchange between $v_j$ and $v_\ell$, while both are implementing the privacy-preserving mechanism (i.e., before $v_j$ and $v_\ell$ have injected all the strictly positive values $u_j[l_j], u_\ell[l_\ell]$ in the network);
2) has an in-neighbor $v_i \in \mathcal{V} \setminus (\mathcal{V}_c \cup \{v_j\})$, which first transmits to node $v_j$ at initialization.

***Proof:*** The proof follows the proof of [1, Proposition 1]. However, the main difference with [1] is the following: In Algorithm 1, let us assume that at time step $k'$, nodes $v_j$ and $v_\ell$ are executing the privacy mechanism (i.e., $\sum_{l_j=k'}^{\infty} u_j[l_j] > 0$ and $\sum_{l_\ell=k'}^{\infty} u_\ell[l_\ell] > 0$) and node $v_j$ transmits a message to its out-neighbor $v_\ell$ (the case where $v_\ell \in \mathcal{N}_j^-$ can be proven identically). Node $v_j$ will inject $u_j[l_j]$ to the value sent to $v_\ell$, and $u_j[l_j]$ is only known to $v_j$ and (perhaps) $v_\ell$. Then, $v_\ell$ will inject an offset

$u_\ell[l_\ell]$ to the received message at time step $k'+1$ (or at a future time step) and will transmit it. The transmitted message depends on the sum of offsets $u_j[l_j]+u_\ell[l_\ell]$. Since both $v_j, v_\ell \in \mathcal{V}_p \setminus \mathcal{V}_c$, the curious nodes may be able to determine $u_j[l_j]+u_\ell[l_\ell]$, but not the individual values $u_j[l_j]$ and $u_\ell[l_\ell]$. As a result, the privacy of both node $v_j$ and node $v_\ell$ is preserved. ■

Note here that a set of curious nodes could also attempt to "estimate" the initial states of some other nodes (e.g., by taking into account any available statistics about the initial states, $u_j$ and $L_j$). However, in our analysis in this article, we are interested in whether the curious nodes can exactly infer the state of another node (see Definition 2).

## V. Initial Offset Privacy-Preserving Strategy

In this section, we present and analyze another privacy-preserving algorithm, in which offsets are introduced only at the initialization stage. The main difference with the approach in Section IV is that the proposed mechanism incorporates an offset to the mass variable of each node only during the initialization steps, in order to effectively preserve the privacy of its initial state.

### A. Initialization for Quantized Privacy Strategy

We have that each node maintains the variables $u_j \in \mathbb{Z}$ and $u_j^{(l)} \in \mathbb{Z}$ for every $v_l \in \mathcal{N}_j^+$ and its transmission counter $c_j \in \mathbb{Z}_{>0}$. Then, during initialization, it chooses the variables $u_j^{(l)}$ and $u_j$ to satisfy the following constraints:

$$u_j^{(l)} \in \mathbb{Z},\ \forall v_l \in \mathcal{N}_j^+ \tag{8a}$$

$$u_j = -\sum_{v_l \in \mathcal{N}_j^+} u_j^{(l)}. \tag{8b}$$

Constraints (8a) and (8b) are explicitly analyzed as follows.

1) In (8a), an integer offset for out-neighbor $v_l \in \mathcal{N}_j^+$ is selected in order to be infused to the initial state that node $v_j$ transmits to that out-neighbor $v_l \in \mathcal{N}_j^+$. The selection of a set of integer offsets, each one corresponding to an out-neighbor $v_l \in \mathcal{N}_j^+$, has to do with privacy preservation guarantees, as discussed in Section V-D.
2) Equation (8b) means that the accumulated offset infused in the network during the initialization steps by node $v_j$ is equal to zero, and the exact quantized average of the initial states can be calculated without any error.

### B. Algorithm Description

The proposed algorithm is a quantized value transfer process, in which each node in a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ performs operations and transmissions according to a set of event-triggered conditions. The intuition behind the algorithm is as follows. Each node $v_j \in \mathcal{V}_p$ that would like to preserve its privacy performs the following steps.

1) It selects a set of integer offsets $u_j^{(l)}$, one for each out-neighbor $v_l \in \mathcal{N}_j^+$. It transmits the values $u_j^{(l)}$ to every out-neighbor $v_l \in \mathcal{N}_j^+$, while it receives the values $u_i^{(j)}$ from its in-neighbors $v_i \in \mathcal{N}_j^-$. Then, it sets its initial state

$$\widetilde{y}_j[0] = y_j[0] + u_j + \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)} \tag{9}$$

where the initial offset $u_j$ is equal to $u_j = -\sum_{v_l \in \mathcal{N}_j^+} u_j^{(l)}$. For example, suppose that $v_j$ has four out-neighbors ($\mathcal{D}_j^+ = 4$), three in-neighbors ($\mathcal{D}_j^- = 3$), and initial state $y_j[0] = 6$. This means that it can choose $u_j^{(0)} = 3$, $u_j^{(1)} = -2$, $u_j^{(2)} = 5$, and $u_j^{(3)} = -3$, while it sets $u_j = -3$. It transmits each of the values $u_j^{(0)}, u_j^{(1)}, u_j^{(2)}$, and $u_j^{(3)}$ to the corresponding out-neighbor, while it receives the values, say, 8, 3, and 6, from its in-neighbors. Then, from (9), it sets its initial state equal to $\widetilde{y}_j[0] = 20$. Note that (9) is essential not only for preserving the privacy of every node's initial quantized state but also for preserving the sum of the initial states, i.e., $\sum_{v_j \in \mathcal{V}} y_j[0] = \sum_{v_j \in \mathcal{V}} \widetilde{y}_j[0]$, as it will be seen later.
2) It chooses an out-neighbor $v_l \in \mathcal{N}_j^+$ according to the unique order $P_{lj}$ (initially, it chooses $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = 0$) and transmits $\widetilde{y}_j[0]$ and $z_j[0]$ to this out-neighbor. Then, it sets $\widetilde{y}_j[0] = 0$ and $z_j[0] = 0$.
3) During the execution of the algorithm, at every step $k$, node $v_j$ may receive a set of mass variables $\widetilde{y}_i[k]$ and $z_i[k]$ from each in-neighbor $v_i \in \mathcal{N}_j^-$. Then, node $v_j$ updates its mass variables according to (1a) and (1b) (where in (1a), we use $\widetilde{y}_j[k]$) and checks if either of its event-triggered conditions (C1) and (C2) holds. If so, a) it sets its state variables $y_j^s[k+1]$ and $z_j^s[k+1]$ equal, respectively, to $\widetilde{y}_j[k+1]$ and $z_j[k+1]$ [i.e., it updates them according to (2)]; and b) it transmits $\widetilde{y}_j[k+1]$ and $z_j[k+1]$ to an out-neighbor according to the predetermined order. If none of the conditions (C1) and (C2) holds, then node $v_j$ stores $\widetilde{y}_j[k+1]$ and $z_j[k+1]$. Note that if no message is received from any of the in-neighbors, the mass variables remain the same.

***Remark 5:*** During the operation of Algorithm 2, each node $v_j$ is able to choose the values of $u_j^{(l)}$, for every $v_l \in \mathcal{N}_j^+$, in a way that ensures that each curious node in $\mathcal{V} \setminus \{v_j\}$ (or a set of colluding nodes in $\mathcal{V}_c \subseteq \mathcal{V}$) can only determine a range $[\alpha, \beta]$, $\alpha < \beta$, for which they can ensure that the values $y_j[0]$ lie in, and $v_j$ can make $\alpha \in \mathbb{R}$ arbitrarily small and/or $\beta \in \mathbb{R}$ arbitrarily large.

The proposed algorithm is summarized in Algorithm 2. Curious nodes that try to identify the initial states of other nodes follow either Algorithm 2 or the underlying average consensus algorithm in Section II-B.

### C. Deterministic Convergence Analysis

***Theorem 3:*** Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. The execution of Algorithm 2 allows each node $v_j \in \mathcal{V}$ to reach quantized average consensus after a finite number of steps, $\mathcal{S}_t$, bounded by $\mathcal{S}_t \le m^2 n$, where $n$ is the number of nodes and $m$ is the number of edges in the network.

***Proof:*** See Appendix B. ■

**Algorithm 2:** Privacy-Preserving Event-Triggered Quantized Average Consensus With Zero-Sum Initial Offsets.

```
Input: A strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with
  $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$
  has an initial state $y_j[0] \in \mathbb{Z}$.
Initialization: Each node $v_j \in \mathcal{V}_p$ does the following:
1) It assigns a unique order $P_{lj}$ in the set
  $\{0, 1, \ldots, \mathcal{D}_j^+ - 1\}$ to each of its out-neighbors
  $v_l \in \mathcal{N}_j^+$.
2) It sets counter $c_j$ to 0 and priority index $e_j$ to $c_j$.
3) It chooses $u_j^{(l)} \in \mathbb{Z}$, for every $v_l \in \mathcal{N}_j^+$. Then, it sets
  $u_j = -\sum_{v_l \in \mathcal{N}_j^+} u_j^{(l)}$.
4) It transmits $u_j^{(l)}$ to each $v_l \in \mathcal{N}_j^+$.
5) It sets $\widetilde{y}_j[0] = y_j[0] + u_j + \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)}$, $z_j[0] = 1$,
  $y_j^s[0] = \widetilde{y}_j[0]$, and $z_j^s[0] = z_j[0]$ (which means that
  $q_j^s[0] = \widetilde{y}_j[0]/z_j^s[0]$).
6) It selects out-neighbor $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = e_j$
  and transmits $\widetilde{y}_j[0]$ and $z_j[0]$ to this out-neighbor. Then,
  it sets $\widetilde{y}_j[0] = 0$ and $z_j[0] = 0$.
7) It sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
Iteration: For $k = 0, 1, 2, \ldots$, each node $v_j \in \mathcal{V}_p$ does
  the following:
• if it receives $\widetilde{y}_i[k]$ and $z_i[k]$ from at least one
  in-neighbor $v_i \in \mathcal{N}_j^-$ then it updates $\widetilde{y}_j[k+1]$ and
  $z_j[k+1]$ according to (1a) and (1b).
○ if either of the conditions (C1) and (C2) hold then
- it sets $y_j^s[k+1] = \widetilde{y}_j[k+1]$, $z_j^s[k+1] = z_j[k+1]$,
  and $q_j^s[k+1] = y_j^s[k+1]/z_j^s[k+1]$;
- it transmits $\widetilde{y}_j[k+1]$ and $z_j[k+1]$ to out-neighbor
  $v_\lambda \in \mathcal{N}_j^+$ for which $P_{\lambda j} = e_j$ and it sets $\widetilde{y}_j[k+1] = 0$
  and $z_j[k+1] = 0$;
- it sets $c_j = c_j + 1$ and $e_j = c_j \mod \mathcal{D}_j^+$.
○ else it stores $\widetilde{y}_j[k+1]$ and $z_j[k+1]$.
Output: $q_j^s[k]$, for every $v_j \in \mathcal{V}$.
```

## D. Topological Conditions for Privacy Preservation

***Proposition 2:*** Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes, where $|\mathcal{V}| \geq 3$. Assume that a subset of nodes $\mathcal{V}_p$ follows the predefined privacy-preserving protocol, as described in Algorithm 2, with offsets chosen as in (8a) and (8b). Any arbitrary subset $\mathcal{V}_c$ of curious nodes $\mathcal{V}_c \subseteq \mathcal{V} \setminus \{v_j\}$ including individually operating curious nodes (i.e., $|\mathcal{V}_c| = 1$) will not be able to identify the initial state $y_j[0]$ of $v_j$, $v_j \in \mathcal{V}_p \setminus \mathcal{V}_c$, as long as $v_j$ has at least two out-neighbors $v_l, v_{l'} \in \mathcal{N}_j^+$ and at least one out-neighbor $v_l \in \mathcal{N}_j^+$ does not collude with nodes in $\mathcal{V}_c$ (i.e., $v_l \notin \mathcal{V}_c$). In particular, any single curious node will not be able to identify the initial state $y_j[0]$ of node $v_j$ as long as it is not the sole out-neighbor of node $v_j$.

***Proof:*** See Appendix C. ■

***Remark 6:*** Note that the topological requirements for Algorithm 2 are relaxed with respect to those of Algorithm 1. While the topological conditions of Algorithm 1 require two connected nodes to follow the privacy-preserving protocol or an in-neighbor to explicitly transmit to the privacy-preserving node during its initialization, the topological conditions of Algorithm 2 do not necessarily need a node that follows the protocol or has side information. In addition, apart from improving the topological conditions that ensure privacy for the nodes following the proposed protocol, Algorithm 2 also exhibits increased convergence speed (see the subsequent discussion in Section VI), since the injection of the zero-valued offset requires only one time step during the initialization procedure.

## VI. Simulation Results

In this section, we present simulation results to illustrate the behavior of our proposed distributed protocols. Specifically, we analyze the cases of:

1) a randomly generated digraph of 20 nodes with the average of the initial states of the nodes turning out to be equal to $q = 181/20 = 9.05$;
2) 1000 randomly generated digraphs of 20 nodes each, where, for convenience in plotting the results, the initial quantized state of each node remained the same (for each one of the 1000 randomly generated digraphs); this means that the average of the nodes' initial quantized states also remained equal to $q = 185/20 = 9.25$.

For each of the above cases, we analyze the scenarios where each node $v_j \in \mathcal{V}$ does the following: Case (i) executes the privacy protocol described in Algorithm 1 and initially infuses in the network a randomly chosen offset $u_j \in [-100, -50]$ with randomly chosen offset adding steps $L_j \in [20, 40]$, Case (ii) executes the privacy protocol described in Algorithm 2 and initially infuses in the network the randomly chosen offset $u_j \in [-100, 100]$, the randomly chosen offsets $u_j^{(l)} \in [-20, 20]$, for every $v_l \in \mathcal{N}_j^+$, and $u_j$ chosen according to (8b), and Case (iii) initially does not infuse any offset in the network, i.e., $u_j = 0$ and $u_j^{(l)} = 0$ for every $v_l \in \mathcal{N}_j^+$, which means that it does not attempt to preserve the privacy of its initial state. Note that the digraphs were randomly generated by creating, independently for each ordered pair $(v_j, v_i)$ of two nodes $v_j$ and $v_i$ ($v_j \neq v_i$), a directed edge from node $v_i$ to node $v_j$ with probability $p = 0.3$.

*Execution over a random digraph of 20 nodes:* In Fig. 2, we illustrate Algorithms 1 and 2 over a random digraph of 20 nodes, where the average of the initial states of the nodes is equal to $q = 181/20 = 9.05$. We analyze the operation of our algorithms for the following scenarios: Case (i) executes Algorithm 1 and initially infuses in the network an offset $u_j \in [-100, -50]$ with offset adding steps $L_j \in [20, 40]$ (see top of Fig. 2); Case (ii) executes Algorithm 2 and initially infuses in the network the randomly chosen offset $u_j \in [-100, 100]$ and the offsets $u_j^{(l)} \in [-20, 20]$, for every $v_l \in \mathcal{N}_j^+$ (see middle of Fig. 2); and Case (iii) initially does not infuse any offset in the network $u_j = 0$, $u_j^{(l)} = 0$, for every $v_l \in \mathcal{N}_j^+$ (see bottom of Fig. 2). We observe that for Case (iii), each node is able to calculate the average of the initial states after 100 time steps. However, for the case where each node $v_j$ wants to preserve the privacy of its initial state, we observe that Algorithm 1 converges after 450 time steps, while Algorithm 2 converges after 105 time steps. Furthermore, we observe that for Case (iii), we have

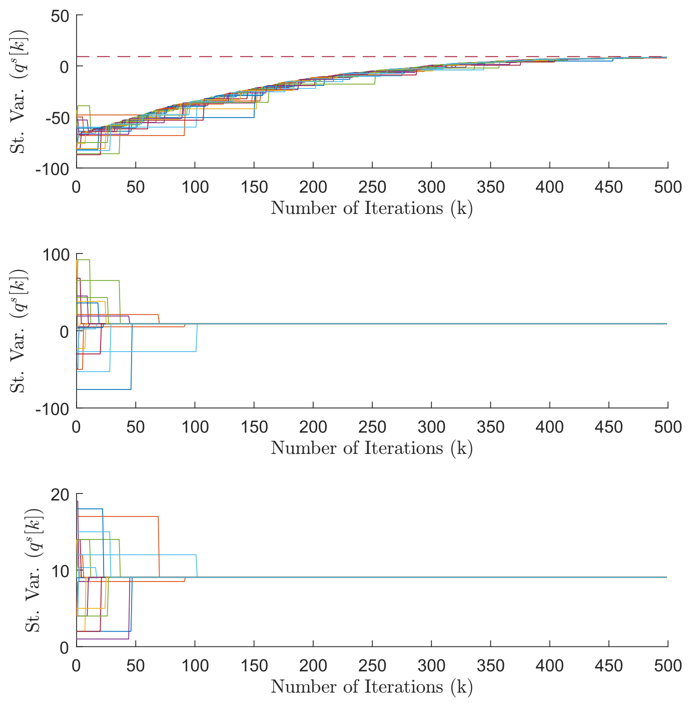

Fig. 2. Execution of Algorithms 1 and 2 for a random digraph of 20 nodes. *Top:* Node state variables with privacy preservation of Algorithm 1 plotted against the number of iterations where the dashed line is the average of the initial states. *Middle:* Node state variables with privacy preservation of Algorithm 2 plotted against the number of iterations. *Bottom:* Node state variables without privacy preservation plotted against the number of iterations.

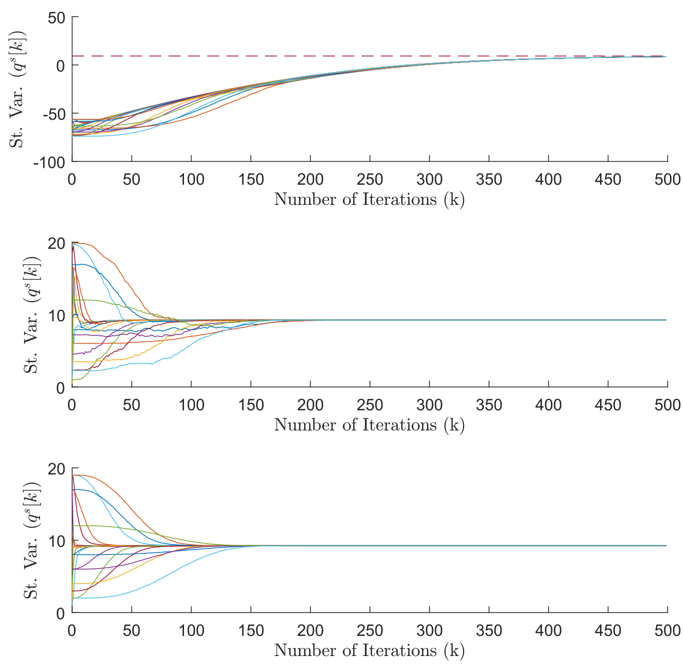

Fig. 3. Execution of Algorithms 1 and 2 averaged over 1000 random digraphs of 20 nodes. *Top:* Average values of node state variables with privacy preservation of Algorithm 1 plotted against the number of iterations (averaged over 1000 random digraphs of 20 nodes), where the dashed line is the average of the initial states. *Middle:* Average values of node state variables with privacy preservation of Algorithm 2 plotted against the number of iterations (averaged over 1000 random digraphs of 20 nodes). *Bottom:* Average values of node state variables without privacy preservation plotted against the number of iterations (averaged over 1000 random digraphs of 20 nodes).

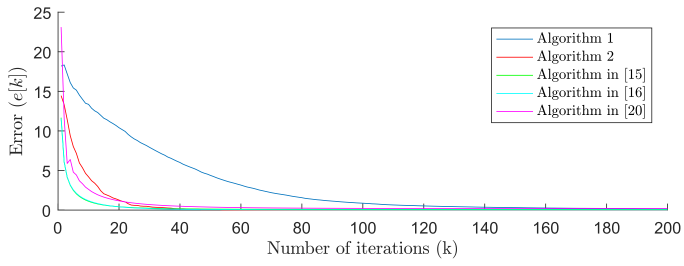

Fig. 4. Comparison between Algorithms 1 and 2, and Algorithms in [15], [16], and [20]. The error ($e[k]$) is plotted against the number of iterations (averaged over 1000 random digraphs of eight nodes).

$y_j[0] \in [3, 19]$ for every $v_j \in \mathcal{V}$; for the case where every node executes Algorithm 1, we have $\widetilde{y}_j[0] \in [-40, -80]$ for every $v_j \in \mathcal{V}$; and for the case where every node executes Algorithm 2, we have $\widetilde{y}_j[0] \in [-80, 90]$ for every $v_j \in \mathcal{V}$. Note here that both the algorithms are able to calculate the *exact* average of the initial states of the nodes without introducing any error due to the utilized privacy-preserving strategy.

*Execution averaged over 1000 random digraphs of 20 nodes:* In Fig. 3, we present the same cases as in Fig. 2 with the difference being that they are averaged over 1000 randomly generated digraphs of 20 nodes. Note that for generating the random digraphs, we used the Erdős–Rényi model. The initial quantized state of each node remained the same for each one of the 1000 randomly generated digraphs (in particular, the average of the initial states of the nodes is equal to $q = 185/20 = 9.25$). For every node $v_j$, the initial offset $u_j \in [-100, -50]$ and the offset adding steps $L_j \in [20, 40]$ during the execution of Algorithm 1, as well as the initial offset $u_j \in [-100, 100]$ and the offsets $u_j^{(l)} \in [-20, 20]$, for every $v_l \in \mathcal{N}_j^+$, during the execution of Algorithm 2, were randomly chosen for each digraph according to a uniform distribution. We can see that the main results resemble those in Fig. 2, and Algorithm 1 converges after 450 time steps, while Algorithm 2 converges after 170 time steps. For Case (iii), we have $y_j[0] \in [4, 19]$ for every $v_j \in \mathcal{V}$, whereas for the case where every node executes Algorithm 1, we have $\widetilde{y}_j[0] \in [-40, -70]$ for every $v_j \in \mathcal{V}$, and for the case where every node executes Algorithm 2, we have $\widetilde{y}_j[0] \in [3, 20]$ for every $v_j \in \mathcal{V}$. This means that, for Case (iii), the states $y_j[0]$ of every $v_j \in \mathcal{V}$ are almost equal to the states $\widetilde{y}_j[0]$ for the case where every node executes Algorithm 2, since the offsets $u_j \in [-100, 100]$ and $u_j^{(l)} \in [-20, 20]$ for every $v_l \in \mathcal{N}_j^+$ were randomly chosen for each digraph according to a uniform distribution.

*Comparison with the existing literature:* In Fig. 4, we compare the performance of Algorithms 1 and 2 against [15], [16], and [20]. We show the error, defined by $e[k] := (\sum_{v_j \in \mathcal{V}} |q_j[k] - \overline{y}|)/|\mathcal{V}|$, plotted against the number of iterations for 1000 random digraphs for the case where the average of the initial states of the nodes is equal to 23.66. Each of the 1000 random digraphs was generated via the Erdős–Rényi model, and the initial state of each node remained the same (thus, the average of the initial states remained the same), but the parameters were randomly

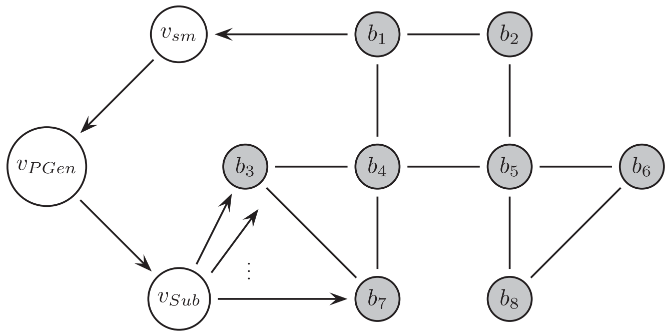

Fig. 5. Example of a digraph representing a smart grid consisting of a neighborhood with eight households $b_1$–$b_8$, a smart meter $v_{\text{sm}}$, a substation $v_{\text{Sub}}$, and a power generator $v_{\text{PGen}}$.

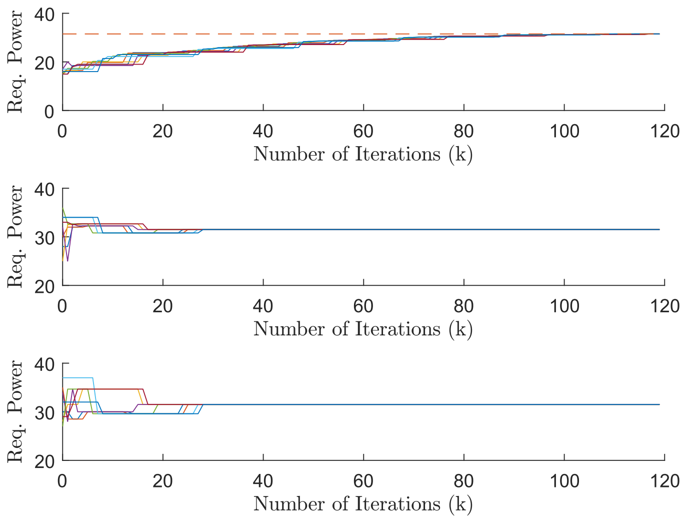

Fig. 6. Execution of Algorithms 1 and 2 for the neighborhood of eight households shown in Fig. 5. *Top:* Requested power per household with privacy preservation of Algorithm 1 plotted against the number of iterations for $\text{day} = 1$, where the dashed line is the average of the initial states. *Middle:* Requested power per household with privacy preservation of Algorithm 2 plotted against the number of iterations for $\text{day} = 1$. *Bottom:* Requested power per household without privacy preservation plotted against the number of iterations for $\text{day} = 1$.

chosen (which is seen from the difference in the initial conditions of every algorithm in Fig. 4). From Fig. 4, we have that 1) algorithms in [15] and [16] have the fastest convergence speed; 2) Algorithm 2 and the algorithm [20] require identical number of time steps for convergence; and 3) Algorithm 1 requires the largest amount of time steps due to its event-triggered conditions. However, note that Algorithms 1 and 2 operate solely with quantized values and converge to the exact solution after a finite number of time steps, while algorithms in [15], [16], and [20] exhibit asymptotic convergence.

***Remark 7:*** In Figs. 2 and 3, we can see that both the algorithms are able to calculate the exact average of the initial states of the nodes without introducing any error due to the utilized privacy-preserving strategies. This makes Algorithms 1 and 2 the first algorithms in the literature that calculate the exact average of the initial states of the nodes in finite time without introducing any error in a privacy-preserving manner. Furthermore, the privacy strategy presented in Algorithm 2 requires fewer time steps for convergence than the strategy in Algorithm 1 since the injection of the zero-valued offset is done during the initialization procedure and requires only one time step. In addition, the topological conditions required by Algorithm 2 are relaxed compared to those required by Algorithm 1. The main operational difference is that in Algorithm 1, the privacy-preserving protocol is executed in the Iteration procedure, while in Algorithm 2, the privacy-preserving protocol is incorporated in the Initialization procedure. However, Algorithm 1 allows for more efficient usage of the available network resources (e.g., communication bandwidth) since each node is required to transmit to at most one out-neighbor at each time step $k$.

## VII. Conclusion

In this article, we proposed two event-triggered quantized privacy-preserving strategies, which allow the nodes of a multiagent system to calculate the average of their initial states using quantized states and after a finite number of time steps without revealing their initial state to other nodes. They take full advantage of the algorithm's finite-time nature, which means that consensus to the *exact* average of the initial states is achieved after a finite number of iterations that we explicitly calculated, while the error, introduced from the offset initially infused in the network by the nodes following the protocol, vanishes completely. The point-to-point communication protocol and the quantized nature of the packets used in the proposed algorithms facilitate the use of cryptographic primitives for setting up secure channels and preventing eavesdropping, while harvesting the benefits of event-triggered and finite-time operation of the distributed privacy-preserving protocol proposed. Finally, we have demonstrated the performance of our proposed protocols via illustrative examples and presented an application in smart grids.

We plan to extend the operation of our algorithms to be hot pluggable (i.e., to be able to operate despite network changes or errors in communication or computation) [21].

## Appendix A
## Application: Power Request in Smart Grids Under Privacy-Preserving Guarantees

In this application, a neighborhood of interconnected households is able to request the *total* demanded (or offered) power from a smart meter in a privacy-preserving manner. One of the main characteristics of smart grids is that the power generator produces electricity based on consumer requests (or offers), which are generated in real time and collected by smart meters. Real-time power demand/offer data may contain patterns from daily/weekly life schedule. Since potential leakage of this sensitive information may lead to malicious situations against the residents of specific households (e.g., thieves may learn time periods when the household is vacant), it is essential to preserve the privacy of the data sent to smart meters (which contains each household's daily requested or offered power).

During the operation, we have the following sequence of actions: 1) the smart meter collects the daily demands (or offers, e.g., due to photovoltaic systems that are installed at the rooftops), from each household and transmits them to the power generator; 2) having received the demands/offers, the power generator produces and delivers the demanded electricity to each substation in the corresponding region; and 3) the electricity is claimed from the substation to the households without any other entities having access to this transaction. The charging of the demanded electricity (or reimbursement for offered electricity) can be communicated at the end of the month from the substations.

As an example, let us consider in Fig. 5 a neighborhood with eight households denoted $\mathcal{B} = \{b_1, b_2, \ldots, b_8\}$, a smart meter $v_{\text{sm}}$, a substation $v_{\text{Sub}}$, and a power generator $v_{\text{PGen}}$. During the operation, $v_{\text{sm}}$ collects (through say $b_1$) the state variable of household $b_1$, which is equal to the average of the daily demanded/offered power from all households in the neighborhood. Then, it multiplies it with the number of houses in the neighborhood in order to calculate the total demanded/offered power and transmit it to the power generator.

By applying either Algorithm 1 or Algorithm 2 in the network shown in Fig. 5, we compute distributively and in a privacy-preserving manner the *total* power requested/offered within a certain time period by a set of interconnected nodes (be it households, electric cars, etc.) in a neighborhood. To formally define the privacy-preserving total power request/offer of a set of interconnected nodes in a neighborhood, let $\mathcal{B} = \{b_1, b_2, \ldots, b_n\}$ denote the set of $n$ interconnected nodes in neighborhood $\mathcal{B}$, and let $\mathcal{C}^{\text{day}} = \{c_1^{\text{day}}, c_2^{\text{day}}, \ldots, c_n^{\text{day}}\}$ denote the set of requested/offered powers per household at each day within a month, where day refers to the day of the month. This means that for node $b_j$, the amount of requested/offered power at the third day of a month is denoted as $c_j^3$. For simplicity of exposition, we show how our algorithms work within a single day, so, hereafter, we drop the index day.

Both the algorithms initially use as input the requested/offered power of each node $b_j \in \mathcal{B}$ for a specific day $c_j$ and create a distorted version $\widetilde{c}_j$. Let $\widetilde{\mathcal{C}} = \{\widetilde{c}_1, \widetilde{c}_2, \ldots, \widetilde{c}_n\}$, be the set of distorted amounts of requested power from every household in the neighborhood at each day within a month. Eventually, the smart meter collects the calculated average demand of the neighborhood, in order to multiply it with the number of participating nodes (there exist algorithms for computing the total number of nodes, in case it can vary, e.g., due to the presence of excess electric vehicles) and calculate the total demanded power. The operation of both the algorithms is as follows.

1) During the operation of Algorithm 1, a set of offset adding steps $L_j$ is chosen from each node $b_j$. Then, each node injects a set of positive offsets for a number of $L_{\max} = \max_{b_j \in \mathcal{B}} L_j$ steps, which will guarantee the preservation of the privacy of the people living in this household.
2) During the operation of Algorithm 2, each node $b_j$ transmits nonzero offsets to its out-neighbors. Then, for each day, the initial states $\widetilde{c}_j$ of every household $b_j$ are calculated (note that it holds $\sum_{j=1}^{n} \widetilde{c}_j = \sum_{j=1}^{n} c_j$). This means that the privacy of the data containing the daily requested/offered power of each household is preserved.

For the households in Fig. 5, let us consider the set $\mathcal{C} = \{30, 35, 28, 34, 27, 37, 29, 32\}$, which denotes the amount of requested power at a certain day from each household (i.e., household $b_1$ requests 30, $b_2$ requests 35, etc.) with the average demand being $\overline{c}^{\text{day}} = 31.5$. Here, we assume that each household 1) has a daily demand above a certain threshold due to daily power consumption, regardless of whether the residents are in the house or not (e.g., fridge consumption and security systems) and 2) may have installed photovoltaic systems, which allow it to produce electricity in order to consume it or sell it (this means that this household may request negative power from the smart meter). During the execution of Algorithm 1, we have $\widetilde{\mathcal{C}} = \{15, 16, 15, 17, 15, 17, 15, 16\}$, while during the execution of Algorithm 2, we have $\widetilde{\mathcal{C}} = \{28, 30, 25, 32, 36, 34, 33, 34\}$. Note here that the daily demands $\widetilde{\mathcal{C}}^{\text{day}}$ for Algorithm 1 are different than the daily demands $\widetilde{\mathcal{C}}^{\text{day}}$ for Algorithm 2, due to the choice of $u_j^{\text{day}}$ and $u_j^l \; \forall v_l \in \mathcal{N}_j^+$, respectively. In Fig. 6, we can see that this choice affects the convergence rate of both the algorithms with Algorithm 2 generally requiring less time steps to converge compared to Algorithm 1 (as already discussed in Section VI).

## Appendix B
## Proof of Theorem 3

In this proof, we focus on showing that the Initialization procedure of Algorithm 2 changes the initial states of each node in such a way that their sum remains the same (which means that the initial average also remains the same).

During the Initialization steps of Algorithm 2, we have that each node $v_j \in \mathcal{V}_p$ sets its initial state as $\widetilde{y}_j[0] = y_j[0] + u_j + \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)}$ and proceeds with executing the protocol described in Section II-C. Focusing on $\widetilde{y}_j[0]$, we have that

$$\begin{aligned}\sum_{v_j \in \mathcal{V}} \widetilde{y}_j[0] &= \sum_{v_j \in \mathcal{V}} \left( y_j[0] + u_j + \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)} \right) \\ &= \sum_{v_j \in \mathcal{V}} y_j[0] + \sum_{v_j \in \mathcal{V}} u_j + \sum_{v_j \in \mathcal{V}} \left( \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)} \right) .\end{aligned} \tag{10}$$

Analyzing the second part of (10), from (8a) and (8b), we have

$$\sum_{v_j \in \mathcal{V}} u_j = - \sum_{v_j \in \mathcal{V}} \left( \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)} \right) \tag{11}$$

so that
$$\sum_{v_j \in \mathcal{V}} \left( y_j[0] + u_j + \sum_{v_i \in \mathcal{N}_j^-} u_i^{(j)} \right) = \sum_{v_j \in \mathcal{V}} y_j[0]. \tag{12}$$

From (10) and (12), we have $\sum_{v_j \in \mathcal{V}} \widetilde{y}_j[0] = \sum_{v_j \in \mathcal{V}} y_j[0]$, which means that the Initialization steps of Algorithm 2 preserves the sum of the initial states. Then, each node executes the algorithm described in Section II-C, whose convergence proof can be seen in [10]. □

## Appendix C
## Proof of Proposition 2

We show that the transmitted values of each $v_j \in \mathcal{V}_p \setminus \mathcal{V}_c$ during the Initialization of Algorithm 2 allow it to preserve the privacy of its initial state for the case where it has at least two out-neighbors, and at least one of its out-neighbors does not collude with the others (regardless of whether the out-neighbor follows the protocol or not).

As was the case with Algorithm 1, the topological conditions will be extracted from simple scenarios, which constitute the building blocks of the directed network.

1) It is easy to observe that if all the in- and out-neighbors of node $v_j$ are curious and they collude with each other, it is not possible for this node to keep its privacy. At initialization, the curious nodes will know the values node $v_j$ transmitted to its out-neighbors, i.e., $u_j^{(l)}$ to every $v_l \in \mathcal{N}_j^+$. In addition, the curious nodes will know the values $v_j$ received during the initialization, i.e., $u_i^{(j)}$ from every $v_i \in \mathcal{N}_j^-$. Hence, after the Initialization of Algorithm 2, the curious nodes will be able to compute the initial offset $u_j$ of node $v_j$, since the initial offset satisfies (8b); hence, privacy of $v_j$'s initial state will not be preserved. As a result, at least one neighbor that is not curious is needed.
2) We consider the case where $v_j$ has at least two out-neighbors $v_l, v_{l'} \in \mathcal{N}_j^+$ and at least one out-neighbor $v_l$ is not colluding with nodes in $\mathcal{V}_c$ (node $v_l$ may or may not follow the privacy-preserving protocol). During the Initialization of Algorithm 2, $v_l$ will receive the value $u_j^{(l)}$ from $v_j$ and will sum it with its own initial state in order to calculate $\widetilde{y}_l[0]$. Since it is not colluding with other curious nodes, the other curious nodes will not be able to directly determine the value of $u_j^{(l)}$. Similarly, node $v_j$ will calculate $\widetilde{y}_j[0]$. We argue below that the privacy of both nodes $v_j$ and $v_l$ is preserved. (Note that a similar argument can be used to establish that node $v_j$ preserves its privacy if one of its in-neighbors $v_i \in \mathcal{V}_p \setminus \mathcal{V}_c$ follows the privacy-preserving protocol and does not collude with other nodes.)

Note that everything that is done after the initialization is a function of $\widetilde{y}_j[0]$ for all $v_j$. Therefore, the curious nodes observe values that are functions of $\widetilde{y}_j[0]$ and can, at best, estimate these values. However, we have that $\widetilde{y}_j[0] = y_j[0] - u_j^{(l)} + \delta$ and $\widetilde{y}_l[0] = y_l[0] + u_j^{(l)} + \theta$, where $\delta, \theta \in \mathbb{Z}$ are some other offsets. Observe that even if $\widetilde{y}_j[0]$, $\widetilde{y}_l[0]$, $\delta$, and $\theta$ become known, one can only estimate the sum $y_j[0] + y_l[0]$ but not the exact values $y_j[0]$ and $y_l[0]$ (which can be arbitrary depending on the value of $u_j^{(l)}$, which is known only to nodes $v_j$ and $v_l$). Note that in such cases, the initial states of both nodes are protected (though the sum of these states may be exposed). As a result, the initial state of the node that follows the protocol can be bounded in an interval $[\alpha, \beta]$, which according to Definition 2 implies that its privacy is preserved. □

## References

[1] A. I. Rikos, T. Charalambous, K. H. Johansson, and C. N. Hadjicostis, "Privacy-preserving event-triggered quantized average consensus," in *Proc. IEEE Conf. Decis. Control*, 2020, pp. 6246–6253.
[2] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.
[3] C. N. Hadjicostis, A. D. Domínguez-García, and T. Charalambous, "Distributed averaging and balancing in network systems, with applications to coordination and control," *Found. Trends Syst. Control*, vol. 5, nos. 3/4, pp. 99–292, 2018.
[4] A. B. Alexandru, M. S. Darup, and G. J. Pappas, "Encrypted cooperative control revisited," in *Proc. IEEE 58th Conf. Decis. Control*, 2019, pp. 7196–7202.
[5] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 1999, pp. 223–238.
[6] C. N. Hadjicostis and A. D. Dominguez-Garcia, "Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus," *IEEE Trans. Autom. Control*, vol. 65, no. 9, pp. 3887–3894, Sep. 2020.
[7] J. Lavaei and R. M. Murray, "Quantized consensus by means of Gossip algorithm," *IEEE Trans. Autom. Control*, vol. 57, no. 1, pp. 19–32, Jan. 2012.
[8] K. Cai and H. Ishii, "Quantized consensus and averaging on gossip digraphs," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2087–2100, Sep. 2011.
[9] M. E. Chamie, J. Liu, and T. Basar, "Design and analysis of distributed averaging with quantized communication," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 3870–3884, Dec. 2016.
[10] A. I. Rikos and C. N. Hadjicostis, "Event-triggered quantized average consensus via ratios of accumulated values," *IEEE Trans. Autom. Control*, vol. 64, no. 10, pp. 4035–4049, Oct. 2020.
[11] M. Kefayati, M. S. Talebi, B. H. Khalaj, and H. R. Rabiee, "Secure consensus averaging in sensor networks using random offsets," in *Proc. IEEE Int. Conf. Telecommun./Malaysia Int. Conf. Commun.*, 2007, pp. 556–560.
[12] S. S. Kia, J. Cortes, and S. Martinez, "Dynamic average consensus under limited control authority and privacy requirements," *Int. J. Robust Nonlinear Control*, vol. 25, no. 13, pp. 1941–1966, 2015.
[13] J. Cortés, G. E. Dullerud, S. Han, J. L. Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *Proc. IEEE Conf. Decis. Control*, 2016, pp. 4252–4272.
[14] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica*, vol. 81, pp. 221–231, 2017.
[15] N. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *Proc. Eur. Control Conf.*, 2013, pp. 760–765.
[16] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 753–765, Feb. 2017.
[17] N. Gupta, J. Katz, and N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine*, vol. 50, pp. 9515–9520, 2017.
[18] N. Rezazadeh and S. S. Kia, "Privacy preservation in a continuous-time static average consensus algorithm over directed graphs," in *Proc. Annu. Amer. Control Conf.*, 2018, pp. 5890–5895.
[19] H. Gao, C. Zhang, M. Ahmad, and Y. Wang, "Privacy-preserving average consensus on directed graphs using push-sum," in *Proc. IEEE Conf. Commun. Netw. Secur.*, 2018, pp. 1–9.
[20] Y. Wang, "Privacy-preserving average consensus via state decomposition," *IEEE Trans. Autom. Control*, vol. 64, no. 11, pp. 4711–4716, Nov. 2019.
[21] I. L. D. Ridgley, R. A. Freeman, and K. M. Lynch, "Private and hot-pluggable distributed averaging," *IEEE Control Syst. Lett.*, vol. 4, no. 4, pp. 988–993, Apr. 2020.
[22] C. Yu, C. Chen, S. Kuo, and H. Chao, "Privacy-preserving power request in smart grid networks," *IEEE Syst. J.*, vol. 8, no. 2, pp. 441–449, Jun. 2014.
[23] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, pp. 66–92, 1998.
[24] K. Chatzikokolakis and C. Palamidessi, "Probable innocence revisited," *Theor. Comput. Sci.*, vol. 367, nos. 1/2, pp. 123–138, 2006.

**Apostolos I. Rikos** (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus, in 2010, 2012, and 2018, respectively.

In 2018, he joined the KIOS Research and Innovation Centre of Excellence in Cyprus, where he was a Special Scientist and an Associate Lecturer. Since 2020, he has been a Postdoctoral Researcher with the Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm, Sweden. His research interests include distributed systems, privacy and security, distributed optimization, and algorithmic analysis.

**Themistoklis Charalambous** (Senior Member, IEEE) received the B.A. degree in electrical and information sciences from Trinity College Dublin, Dublin, Ireland, the M.Eng. degree in electrical and information sciences from the University of Cambridge, Cambridge, U.K., in 2005, and the Ph.D. degree from Control Laboratory, Engineering Department, University of Cambridge, in 2010.

From 2009 to 2010, he was a Research Associate with Human Robotics Group, Imperial College London, London, U.K. From 2010 to 2011, he was a Visiting Lecturer with the Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus. From 2012 to 2015, he was a Postdoctoral Researcher with the Department of Automatic Control, School of Electrical Engineering, Royal Institute of Technology, Stockholm, Sweden. From 2015 to 2016, he was a Postdoctoral Researcher with the Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden. In 2017, he joined the Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, Espoo, Finland, as a tenure-track Assistant Professor. Since 2018, he has been a Research Fellow of the Academy of Finland. From 2020 to 2021, he was a tenured Associate Professor and since then he has been a Visiting Professor. Since 2021, he has been a tenure-track Assistant Professor with the University of Cyprus. His primary research interests include the design and analysis of (wireless) networked control systems that are stable, scalable, and energy efficient.

**Karl Henrik Johansson** (Fellow, IEEE) received the M.Sc. degree in electrical engineering and the Ph.D. degree in automatic control from Lund University, Lund, Sweden.

He is currently a Professor with the School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden, and the Director of Digital Futures. He has held visiting positions with UC Berkeley, Berkeley, CA, USA, Caltech, Pasadena, CA, USA, NTU, HKUST Institute of Advanced Studies, Hong Kong, and NTNU, Trondheim, Norway. His research interests include control systems and cyberphysical systems with applications in transportation, energy, and automation networks.

Dr. Johansson is the President of the European Control Association and a Member of the IFAC Council, and was on the IEEE Control Systems Society Board of Governors and the Swedish Scientific Council for Natural Sciences and Engineering Sciences. He was the recipient of best paper awards and other distinctions from IEEE, IFAC, and ACM. He has been awarded Swedish Research Council Distinguished Professor, Wallenberg Scholar with the Knut and Alice Wallenberg Foundation, Future Research Leader Award from the Swedish Foundation for Strategic Research, triennial IFAC Young Author Prize, and IEEE Control Systems Society Distinguished Lecturer. He is a Fellow of the Royal Swedish Academy of Engineering Sciences.

**Christoforos N. Hadjicostis** (Fellow, IEEE) received the B.S. degrees in electrical engineering, computer science and engineering, and mathematics, in 1995, and the M.Eng. and the Ph.D. degrees in electrical engineering and computer science, in 1995 and 1999, respectively, all from the Massachusetts Institute of Technology, Cambridge, MA, USA,.

In 1999, he joined at the the Faculty University of Illinois at Urbana-Champaign, Champaign, IL, USA, where he was an Assistant Professor and then an Associate Professor with the Department of Electrical and Computer Engineering, Coordinated Science Laboratory, and the Information Trust Institute, respectively. Since 2007, he has been with the Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus, where he is currently a Professor. His research interests include fault diagnosis and tolerance in distributed dynamic systems, error control coding, monitoring, diagnosis and control of large-scale discrete-event systems, and applications to network security, energy distribution systems, anomaly detection, and medical diagnosis.

Dr. Hadjicostis is the Editor-in-Chief of the *Journal of Discrete Event Dynamic Systems*. He was an Associate Editor for *Automatica*, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, *Nonlinear Systems: Hybrid Systems*, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, AND IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I.