2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Privacy Assessment for Linear Consensus using Constrained Convex Generators

Daniel Silvestre

Abstract— **The problem of designing privacy-preserving algorithms for multi-agent systems running distributed algorithms has attracted the attention of the Control community, especially for maintaining the privacy of the initial state. In this paper, we tackle the problem of checking the privacy of the algorithm itself in terms of the linear parameters used by each agent. We first start by introducing a metric of privacy that translates the uncertainty that an attacker has related to the parameters for the case that it can eavesdrop the state from other agents given the public nature of the network. We then propose to resort to techniques in the literature to compute such metric and show how these can be used by: i) the attacker to estimate the parameters from successive runs of the algorithm; ii) by a defender that can decide when to trigger a negotiation of new parameters to ensuse privacy of the overall system. The technique is illustrated in simulations for the specific example of a consensus protocol. The tools developed herein can complement resilient consensus algorithms based on reputation metrics in the sense that the defender triggering changes to the dynamics while maintaining the overall convergence value can render the calculations of the optimal attacks rather troublesome.**

Index Terms— **Privacy; Linear Systems; Multi-agent Systems**

I. Introduction

In the realm of automatic control, many distributed algorithms incorporate in some form a consensus step like: formation control [1], [2], distributed Kalman filters [3], PageRank computation [4], distributed optimization such as the DEXTRA algorithm [5], desynchronization of transmitter in a sensor network [6], [7], solvers for large MPC problems [8], distributed fault detection [9], among many others. Given the widespread importance of linear consensus algorithms, there have been extensive research in attempting to deal with attackers and to maintain privacy. However, keeping sensitive information is not only beneficial for privacy concerns but is also tightly connected with the ability of attackers to be successful and change the steady state.

In the literature of resilient consensus, the work in [10] considered the continuous case of a consensus problem in the presence of potential attackers and presented a graph-theoretic metric to assess network topologies that can be robust and lead normal agents to consensus. In [11], a protocol called ARC-P is presented that uses a parameters f to decide which values to maintain in the computations. In another direction, the authors of [12] tackled the general problem of reaching resilient consensus among a set of agents in the presence of faulty nodes. The presented method is an extension of [13] and is suitable for both discrete-time and continuous-time consensus and identifies the normal agents based on the assumed dynamics. The aforementioned works all share a common assumption for the design of their protocols, namely that the matrix governing the nodes updates is known by the entire network. If such assumption is not met, it becomes rather troublesome to check the graph conditions or estimate how closely each agent is following the algorithm.

In another direction, the work in [14] proposes resilience by computing the variance of each node in the network followed by a voting mechanism to identify the agent with the largest variance. Once again, such approach is only possible because the dynamics is known and a key property is proved that the variances are sorted by the distance in hops to the attacker.

Fault detection and isolation has also been proposed for consensus systems leveraging the concept of distinguishability [15] between the nominal dynamics and those with each subset of the node set being controlled by an attacker as the example in [16]. By resorting to guaranteed state estimation using solely the network in the vicinity of each node, the technique is capable of generating sets that contain all possible trajectories and combine the estimates in a consensus-like update [17] to improve the speed of convergence to a single point and offer theoretical guarantees of bounded effect for all possible attacker strategies. Overall, the same key observation still applies. The mentioned methods all resort to the knowledge of the dynamics to build the defensive mechanisms. From the attacker angle, [18] showed that the zero dynamics of the system (using the knowledge of the dynamics) can be exploited to design attack sequences that cannot be detected.

In this paper, we are interested in considering the case where the network designer does not disclose the dynamics but the distributed algorithm is still being run in a public network. In this setup, a defender must assume that all communicated state values could be subject of eavesdropping by the attacker and wants to maintain the privacy of the algorithm itself. The main contributions in this paper can be summarized as follows:

- We introduce a strategy based on guaranteed state es-

D. Silvestre is with School of Science and Technology from the NOVA University of Lisbon (FCT/UNL), 2829-516 Caparica, Portugal, with COPELABS from the Lusófona University, and also with the Institute for Systems and Robotics, Instituto Superior Técnico, University of Lisbon. dsilvestre@isr.tecnico.ulisboa.pt

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) through Institute for Systems and Robotics (ISR), under Laboratory for Robotics and Engineering Systems (LARSyS) project UIDB/50009/2020, through project PCIF/MPG/0156/2019 FirePuma and through COPELABS, University Lusófona project UIDB/04111/2020.

timation using Constrained Convex Generators (CCGs) [19] that the attacker can use to estimate the parameters used in the algorithm;
- Using the previous method, we propose a defensive mechanism where an entity maintains the estimation procedure and triggers a change in the dynamics whenever the knowledge regarding the current values is compromised.

A related topic to the work being presented herein is the concept of state estimation for Linear-Time Invariant (LTI) systems, which can be accomplished for various set representation such as using intervals [20], zonotopes [21] and ellipsoids [22] which are not accurate since intersections cannot be expressed in closed-form. Techniques resorting to polytopes [23] and in the format of constrained zonotopes [24] are the most accurate given that the operations can be performed in closed-form. However, in this paper we will be using CCGs that allows to represent a larger class of bounds.

The remainder of the paper is organized as follows. Section II formalizes the state estimation problem associated with identifying the dynamics of the system. We review in Section III the definition and main set operations for CCGs, while Section IV is dedicated to presenting estimation procedure that can be used by an attacker. Section V then proposes a triggering technique to maintain a desired level of privacy for the algorithm, which is validated in simulations presented in Section VI. Conclusions and directions of future work are given in Section VII.

Notation : We let 0_n denote the n-dimensional vector of zeros and I_n the identity matrix of size n. The operator $\mathrm{diag}(v)$ creates a diagonal matrix with v in the diagonal or extracts the diagonal if the argument is a matrix. The transpose of a vector v is denoted by $v^\intercal$, while the Euclidean norm for vector x is represented as $\|x\|_2 := \sqrt{x^\intercal x}$. On the other hand, $\|x\|_\infty := \max_i |x_i|$. The cartesian product is denoted by $\times$, the Minkowski sum of two sets by $\oplus$ and the intersection after applying a matrix R to the first set by $\cap_R$.

II. Problem Statement

We consider a set of n agents with scalar state $x_i(k)$, $1 \leq i \leq n$, whose objective is to compute some weighted average of their initial values, i.e.,

$$\lim_{k\to\infty} x_i(k) = x_{\text{av}} := \frac{1}{n}\sum_{i=1}^{n} a_i x_i(0).$$

The n nodes use a public network such that they can possibly communicate with all the remaining nodes. However, in order to reduce the communication overhead, the nodes do not use all possible links. A communication topology can be represented by a directed graph $G = (\mathcal{V}, E)$, where $\mathcal{V}$ represents the set of n agents and $E \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. If a node i uses the value of node j, then $(i, j) \in E$. Moreover, there is a *weighted adjacency matrix* W associated with the graph G with entries:

$$W_{ij} := \begin{cases} w_{ij}, \text{ if } (i,j) \in E, w_{ij} \in [0,1] \\ 0, \text{ otherwise} \end{cases}$$

In order to increase privacy, it is assumed that nodes communicate in a multicast fashion and append their state value to the received message along with some cryptography token that ensures data has not been compromised from the previous node. In doing so, an attacker cannot infer which edges are in E since a node receiving a message from j might be retrieving a value from node κ that is present in the message. Therefore, the full iteration can be written in matrix form as:

$$x(k+1) = Wx(k), \tag{1}$$

where matrix W must satisfy the following properties: $W \geq 0$ (non-negative entries), $W\mathbf{1} = \mathbf{1}$ (the vector of ones is a right eigenvector associated with the unity eigenvalue), and, the underlying graph is strongly connected (implies that the remaining eigenvalues have magnitude strictly smaller than one).

The problem tackled in this paper can be formalized as follows.

Problem 1: Attacker: Given information of sequences of state values $x(k)$, how to estimate the matrix W used by the distributed system. *Defender:* Assuming that the attacker is running the previous mechanism, design a method to guarantee that an attacker cannot retrieve W.

III. Constrained Convex Generators overview

In this section, we first review the definition for a CCG in Definition 2 and the main operations in Definition 3.

Definition 2 (Constrained Convex Generators): A Constrained Convex Generator (CCG) $\mathcal{Z} \subset \mathbb{R}^n$ is defined by the tuple $(G, c, A, b, \mathfrak{C})$ with $G \in \mathbb{R}^{n\times n_g}$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{n_c\times n_g}$, $b \in \mathbb{R}^{n_c}$, and $\mathfrak{C} := \{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_{n_p}\}$ such that:

$$\mathcal{Z} = \{G\xi + c : A\xi = b, \xi \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_{n_p}\}.$$

Definition 3: Consider three Constrained Convex Generators (CCGs) as in Definition 2:

- $Z = (G_z, c_z, A_z, b_z, \mathfrak{C}_z) \subset \mathbb{R}^n$;
- $W = (G_w, c_w, A_w, b_w, \mathfrak{C}_w) \subset \mathbb{R}^n$;
- $Y = (G_y, c_y, A_y, b_y, \mathfrak{C}_y) \subset \mathbb{R}^m$;

and a matrix $R \in \mathbb{R}^{m\times n}$ and a vector $t \in \mathbb{R}^m$. The three set operations are defined as:

$$RZ + t = (RG_z, Rc_z + t, A_z, b_z, \mathfrak{C}_z)$$

$$Z \oplus W = \left(\begin{bmatrix} G_z & G_w \end{bmatrix}, c_z + c_w, \begin{bmatrix} A_z & 0 \\ 0 & A_w \end{bmatrix}, \begin{bmatrix} b_z \\ b_w \end{bmatrix}, \{\mathfrak{C}_z, \mathfrak{C}_w\}\right)$$

$$Z \cap_R Y = \left(\begin{bmatrix} G_z & 0 \end{bmatrix}, c_z, \begin{bmatrix} A_z & 0 \\ 0 & A_y \\ RG_z & -G_y \end{bmatrix}, \begin{bmatrix} b_z \\ b_y \\ c_y - Rc_z \end{bmatrix}, \{\mathfrak{C}_z, \mathfrak{C}_y\}\right).$$

The above operations have been implemented in Matlab code and can be retrieved as a toolbox that allows comparison of different guaranteed state estimation techniques in ReachTool from the Github repository in `https://github.com/danielmsilvestre/ReachTool`. We point out that CCGs encompass a large family of other set representations, namely:

- an interval corresponds to $(G, c, [\,], [\,], \|\xi\|_\infty \le 1)$, for a diagonal matrix G;
- a zonotope is given by $(G, c, [\,], [\,], \|\xi\|_\infty \le 1)$;
- an ellipsoid is defined by $(G, c, [\,], [\,], \|\xi\|_2 \le 1)$, for a square matrix G;
- a constrained zonotope or polytope is $(G, c, A, b, \|\xi\|_\infty \le 1)$;
- a convex cone in $\mathbb{R}^n$ is $(G, c, [\,], [\,], \xi \ge 0)$;
- ellipsotopes [25] are given by $(G, c, A, b, \|\xi\|_{p_1} \le 1, \cdots, \|\xi\|_{p_m} \le 1)$, for some $p_i > 0$, $1 \le i \le m$;
- AH-polytopes [26] are given by $(G, c, [\,], [\,], A\xi \le b)$.

IV. Estimation of W using Constrained Convex Generators (CCGs)

In this section, we aim to produce set-valued estimates for the matrix W that is being used in the distributed consensus algorithm. From the required characteristics found in the problem definition in Section II, we can introduce a set $\mathcal{W} \in \mathbb{R}^{n_w}$ ($n_w = n^2$) that will represent all the entries of W following a vectorization operation:

$$\mathcal{W}(0) = \mathcal{H} \cap_{1_n^\intercal \otimes I_n} (0_n, 1_n, [\,], [\,], \|\xi\|_\infty \le 1) \tag{2}$$

where

$$\mathcal{H} = \left(\frac{1}{2} I_{n_w}, \frac{1}{2} 1_{n_w}, [\,], [\,], \|\xi\|_\infty \le 1 \right),$$

which already satisfies the condition that all $W_{ij} \in [0, 1]$ and the eigenvector constraint.

The iterative procedure to obtain a better estimate $\mathcal{W}(k)$ from $\mathcal{W}(k-1)$ after using another state value information can be obtained with:

$$\mathcal{W}(k) = \mathcal{W}(k-1) \cap_{x(k-1)^\intercal \otimes I_n} (0_n, x(k), [\,], [\,], \|\xi\|_\infty \le 1) \,. \tag{3}$$

Following iteration (3) results in a polytope of all possible values for the entries of matrix W. Therefore, a suitable metric for privacy can be defined as:

$$m(k) = \max_i y_i - z_i, \forall y, z \in \mathcal{W}(k) \tag{4}$$

which translates the maximum uncertainty interval for any entry of two points in $\mathcal{W}(k)$. The metric in (4) can be computed by solving $2n_w$ optimization problems of the form:

$$\begin{aligned} \min_{\eta} \quad & v^\intercal \eta \\ \text{s.t.} \quad & \eta \in \mathcal{W}(k) \end{aligned} \tag{5}$$

for all canonical vectors e_i and $-\mathrm{e}_i$, with $1 \le i \le n_w$.

The attacker strategy of following Algorithm 1 can be made to always stop providing that the sequence of initial conditions for the consensus algorithm produce linearly independent hyperplanes in step 5, which most likely in general will happen. In the next section, we take advantage of the same algorithm to introduce a possible defense strategy that is guaranteed to result in a private dynamics in the sense that $m(k) > c$ for some specific desired level of privacy.

Algorithm 1 Attacker Strategy.

```
Require: Set $\mathcal{W}(0) \subseteq \mathbb{R}^{n_w}$ using (2) and threshold $\epsilon > 0$.
Ensure: Calculation of $W$ and a uncertainty metric $0 \le m(k) \le 1$.
1: for $k > 0$ do
2:   /* Construct measurement set $Y(k)$ */
3:   $Y(k) = (0_n, x(k), [\,], [\,], \|\xi\|_\infty \le 1)$
4:   /* Compute new estimate $\mathcal{W}(k)$ */
5:   $\mathcal{W}(k) = \mathcal{W}(k-1) \cap_{x(k-1)^\intercal \otimes I_n} Y(k)$
6:   for each $1 \le i \le n_w$ do
7:     /* Calculate maximum for each variable */
8:     $\bar{p_i}$ = solving (5) with $v = -\mathrm{e}_i$
9:     /* Calculate minimum for each variable */
10:    $\underline{p_i}$ = solving (5) with $v = \mathrm{e}_i$
11:    /* Calculate metric $m(k)$ */
12:    $m(k) = \max(m(k), \bar{p_i} - \underline{p_i})$
13:    if $m(k) < \epsilon$ then
14:      return $W = \mathrm{reshape}(p, [n, n])$
15:    end if
16:  end for
17: end for
```

V. Defense Strategy

From the previous section, given that CCGs are exact for all the required operations, the estimation translates precisely the uncertainty associated with the attacker possibilities. For that reason, if the defender appropriately runs the estimation assuming full knowledge of the messages from the attacker and measures the privacy metric, that value is guarantee in the worst case. The only concern should be that the estimation procedure is executed prior to sending the data of the resulting state. That can be achieved if the agents perform the estimation, decide whether to abort this computing step and only then send the new data. We summarize the method in Algorithm 2.

The correctness of Algorithm 2 in the following lemma.

Lemma 1: Let us assume a n-node network using a matrix W and dynamics (1) and a defense mechanism as in Algorithm 2. Then, $\forall k > 0$ if $m(k-1) > \epsilon$, for some $0 < \epsilon < 1$, we have that $m(k) > \epsilon$.

Proof: We start by analyzing the base case when $k = 1$. Given that, at $k = 0$, any p_i entry of the polytope $\mathcal{W}(0)$ not belonging to the attacker will have uncertainty interval equal to $[0, 1]$ and $m(0) = 1 > \epsilon$. In line 13, Algorithm 2 will test whether $m(1) < \epsilon$. If that is not the case, the result stands. If $m(1) < \epsilon$, a new W will be selected and $m(1) = 1$ since $\mathcal{W}(1) = \mathcal{W}(0)$. For the induction step, if the test in line 13 is false, the conclusion is trivial. If the test yields true, $m(k) = 1$ since $\mathcal{W}(k) = \mathcal{W}(0)$ as $x(k)$ has not been sent and the attacker using Algorithm 1 cannot perform the $k - th$ iteration and will get the maximum uncertainty corresponding to $m(k-1)$, and the conclusion follows. ■

Algorithm 2 Defender Strategy.

Require: Set $\mathcal{W}(0) \subseteq \mathbb{R}^{n_w}$ using (2) and threshold $\epsilon > 0$.
Ensure: Decision to trigger a change for a different W.

```
1:  for $k > 0$ do
2:    /* Construct measurement set $Y(k)$ with the undisclosed point $x(k)$ */
3:    $Y(k) = (0_n, x(k), [\,], [\,], \|\xi\|_\infty \leq 1)$
4:    /* Compute new estimate $\mathcal{W}(k)$ */
5:    $\mathcal{W}(k) = \mathcal{W}(k-1) \cap_{x(k-1)^\intercal \otimes I_n} Y(k)$
6:    for each $1 \leq i \leq n_w$ do
7:      /* Calculate maximum for each variable */
8:      $\bar{p_i}$ = solving (5) with $v = -\mathrm{e}_i$
9:      /* Calculate minimum for each variable */
10:     $\underline{p_i}$ = solving (5) with $v = \mathrm{e}_i$
11:     /* Calculate metric $m(k)$ */
12:     $m(k) = \max(m(k), \bar{p_i} - \underline{p_i})$
13:     if $m(k) < \epsilon$ then
14:       /* A change in $W$ is required */
15:       return True
16:     else
17:       /* The algorithm can proceed*/
18:       $\mathrm{send}(x(k))$
19:       return False
20:     end if
21:   end for
22: end for
```

VI. Simulations

In this section, we aim at presenting simulation results that can highlight both the successful and eventual detection of the attacker in the absence of a defense mechanism but also that the proposed strategy for the defender is able to maintain a given level of privacy. Simulations were run in Matlab R2018a running on a HP machine with a Intel Core i7-8550U CPU @ 1.80GHz and 12 GB of memory resorting to Yalmip as the language to model optimization problems and Mosek as the underlying solver for the optimizations in (5).

The weighted adjacency matrix was randomly selected by adding each possible link in the graph with probability 0.2 and then testing if it produced a strongly connected graph. If that was not the case, the procedure was repeated for the non-existent links until the property held. We then converted the adjacency through a normalization of each row by the number of in-neighbor agents. We remark that other methods could be used to produce different dynamics such as selecting a random value for the self-loop entry and normalizing the remaining entries such that the row adds up to one.

In the scenario with no defensive mechanism, the simulation halts when matrix W is known by the attacker. In order to make the simulations more challenging for the defensive strategy, we also included some nodes to be controlled by the attacker meaning that those rows of W are known *a priori*. The distributed consensus process will be halted by the agents whenever the values are within a 10^{-4} tolerance. The

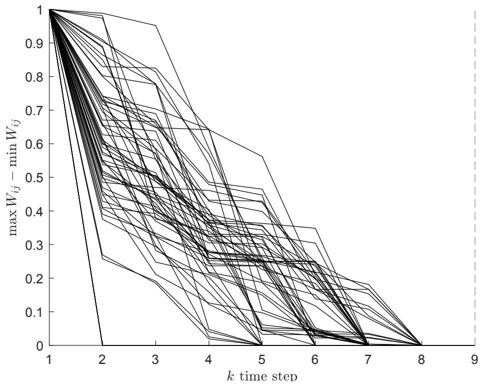

Fig. 1: Uncertainty for each entry of the matrix $W \in \mathbb{R}^{8\times 8}$ as more data is gathered.

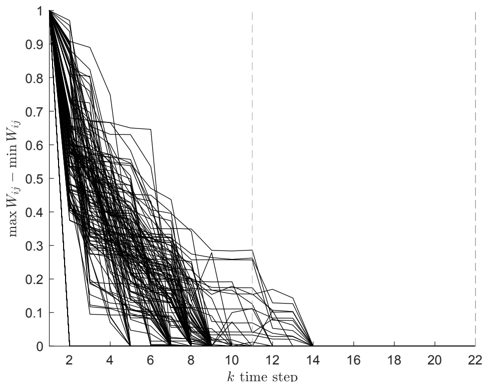

Fig. 2: Uncertainty for each entry of the matrix $W \in \mathbb{R}^{12\times 12}$ as more data is gathered.

dashed lines represent how many consensus computations needed to be gathered before the attacker could retrieve the entire matrix.

Figure 1 presents the evolution of the uncertainty for each of the parameters of the dynamics matrix when there are 8 nodes and a single agent controlled by the attacker. As observed, matrix W can be retrieve in 8 iteration, which is fewer iterations than the consensus algorithm would perform to achieve a steady state within the desired accuracy.

In contrast, the case of a 12×12 matrix is harder to obtain as shown in Figure 2. For this size, the attacker would have to gather all the state values transmitted during two calculation cycles. A similar pattern was observed for other simulations with a network of 15 and 20 nodes. A key point to consider is whether the amount of nodes being controlled by the attacker (hence the prior knowledge of W) has a meaningful impact on the number of iterations required to retrieve the remaining. In Figure 3, it is depicted the uncertainty evolution for 3 different values of number of controlled agents. Interestingly, there is very little difference and the small change can be attributed to numerical issues in the solver when solving the optimizations in (5) (both the case of $n_y = 1$ and $n_y = 3$ overlap). The three simulations used exactly the same unknown matrix W, revealing that the uncertainty is caused by the size of the network and the

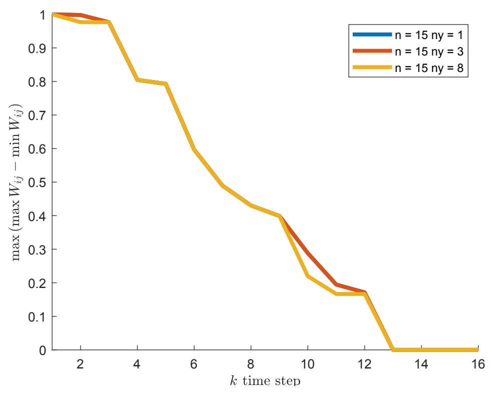

Fig. 3: Uncertainty for each entry of the matrix $W \in \mathbb{R}^{15\times 15}$ as more data is gathered.

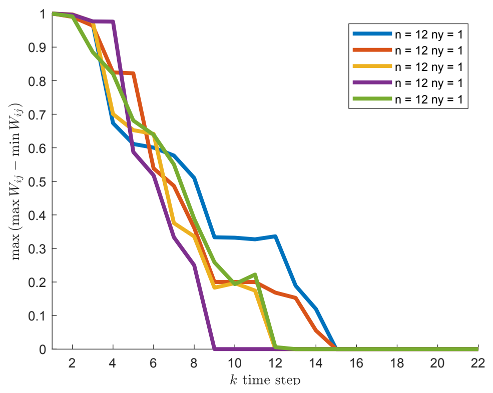

Fig. 4: Uncertainty for each entry of the matrix $W \in \mathbb{R}^{12\times 12}$ as more data is gathered using different topologies.

initial values for the consensus state rather than the number of controlled nodes.

A final interesting point is the variability of the uncertainty evolution depending on the topology that the algorithm is using. Figure 4 showcases a 12-node network simulation but using 5 different topology graphs. The uncertainty of the estimation follows a similar path in all cases albeit with different iteration times when the attacker retrieved the entire matrix of 9, 12 and 15 time steps. A possible avenue of future research is understanding whether carefully selected initial conditions can have a significant impact on the estimation.

Having illustrated the behavior of the estimation if no countermeasure is implemented by the system designer, the next simulation aims at showcasing the effectiveness of the proposed method. As presented in Section V, as the defender is running the same observer and no operations are conservative, it is expected that the mechanism maintains the desired level of privacy. In this setup, we assume a value of 0.1 uncertainty with the results depicted in Figure 5. The dashed lines represent the times where either the consensus algorithm halted naturally or the dynamics were changed prematurely and the process continued with a new dynamics matrix. As expected, the metric always stays above the designed value. Given that the attacker could retrieve in most networks the entire matrix within 1 or 2 rounds, it is

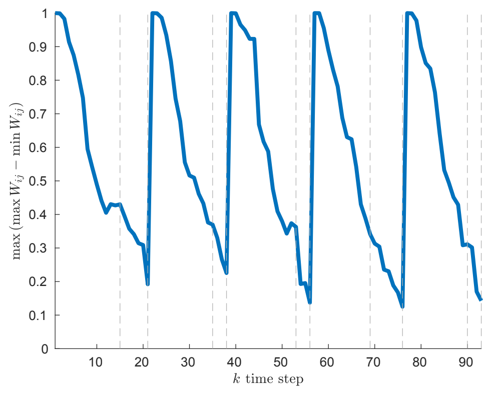

Fig. 5: Uncertainty for each entry of the matrix $W \in \mathbb{R}^{30\times 30}$ as more data is gathered when the defensive mechanism is used.

not surprising that the simulation shows large rounds (that completed) followed by shorter ones where a new matrix had to be used in order to continue the computation while preserving the privacy of which links are being used and its corresponding weights.

VII. Conclusions and Future Work

In this paper, we have address the problem of identifying the set of parameters being used by a distributed consensus algorithm in order to assess its privacy. It is assumed full knowledge on the attacker side related to the sent state values but not on which values are used by the agents. By framing the problem as a set-valued estimation, we resort to Constrained Convex Generators (CCGs) to build a description of the dynamics matrix. We then leverage the estimation tool as a defensive mechanism where the nodes can compute the same estimation that a fully-informed attacker could and decide whether to send the new state value or redefine the dynamics matrix to increase the uncertainty on the attacker estimation.

In simulation, we first tested the effectiveness of the estimation without any defensive mechanism and showed that the attacker can retrieve the full matrix by eavesdropping a very small number of communication times. The evolution of the uncertainty was mainly due to the size of the network for a random initialization of the consensus process. With the countermeasures in place, the simulation ran until the end with the metric for privacy satisfying the desired lower bound. The current work opens the possibility to define other metrics of privacy other than the maximum length of a side of the overbounding hyper-rectangle. In a future work, we aim at considering the case when the attacker cannot know all the state values because the network is divided into a public and private subnetworks interconnected by a gateway server. Such a case could be quite beneficial if we are considering the setup of a sensor network running the consensus and bridged to the public network by a more expensive and robust node.

References

[1] R. Ribeiro, D. Silvestre, and C. Silvestre, "A rendezvous algorithm for multi-agent systems in disconnected network topologies," in *28th Mediterranean Conference on Control and Automation (MED)*, 2020, pp. 592–597.

[2] ——, "Decentralized control for multi-agent missions based on flocking rules," in *CONTROLO 2020*, J. A. Gonçalves, M. Braz-César, and J. P. Coelho, Eds. Cham: Springer International Publishing, 2021, pp. 445–454.

[3] R. Olfati-Saber, "Distributed kalman filter with embedded consensus filters," in *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005, pp. 8179–8184.

[4] D. Silvestre, J. Hespanha, and C. Silvestre, "A pagerank algorithm based on asynchronous gauss-seidel iterations," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 484–489.

[5] C. Xi and U. A. Khan, "Dextra: A fast algorithm for optimization over directed graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 4980–4993, 2017.

[6] D. Silvestre, J. Hespanha, and C. Silvestre, "Desynchronization for decentralized medium access control based on gauss-seidel iterations," in *2019 American Control Conference (ACC)*, 2019, pp. 4049–4054.

[7] ——, "Fast desynchronization algorithms for decentralized medium access control based on iterative linear equation solvers," *IEEE Transactions on Automatic Control*, vol. 67, no. 11, pp. 6219–6226, 2022.

[8] E. Camponogara, D. Jia, B. Krogh, and S. Talukdar, "Distributed model predictive control," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 44–52, 2002.

[9] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre, "Distributed fault detection using relative information in linear multi-agent networks," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 446–451, 2015, 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315016961

[10] H. J. LeBlanc, H. Zhang, S. Sundaram, and X. Koutsoukos, "Resilient continuous-time consensus in fractional robust networks," in *2013 American Control Conference*. IEEE, 2013, pp. 1237–1242.

[11] H. J. LeBlanc and X. Koutsoukos, "Resilient first-order consensus and weakly stable, higher order synchronization of continuous-time networked multiagent systems," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1219–1231, 2017.

[12] G. Ramos, D. Silvestre, and C. Silvestre, "General resilient consensus algorithms," *International Journal of Control*, vol. 0, no. 0, pp. 1–15, 2020.

[13] ——, "A general discrete-time method to achieve resilience in consensus algorithms," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 2702–2707.

[14] D. Silvestre, J. P. Hespanha, and C. Silvestre, "Resilient desynchronization for decentralized medium access control," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 803–808, 2021.

[15] D. Silvestre, P. Rosa, and C. Silvestre, "Distinguishability of discrete-time linear systems," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 5, pp. 1452–1478, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.5367

[16] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre, "Stochastic and deterministic fault detection for randomized gossip algorithms," *Automatica*, vol. 78, pp. 46–60, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109816305192

[17] ——, "Finite-time average consensus in a byzantine environment using set-valued observers," in *2014 American Control Conference*, 2014, pp. 3023–3028.

[18] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[19] D. Silvestre, "Constrained convex generators: A tool suitable for set-based estimation with range and bearing measurements," *IEEE Control Systems Letters*, vol. 6, pp. 1610–1615, 2022.

[20] R. E. H. Thabet, T. Raïssi, C. Combastel, D. Efimov, and A. Zolghadri, "An effective method to interval observer design for time-varying systems," *Automatica*, vol. 50, no. 10, pp. 2677 – 2684, 2014.

[21] C. Combastel, "A state bounding observer based on zonotopes," in *European Control Conference (ECC)*, 2003, pp. 2589–2594.

[22] F. Chernousko, "Ellipsoidal state estimation for dynamical systems," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 63, no. 5, pp. 872 – 879, 2005, invited Talks from the Fourth World Congress of Nonlinear Analysts (WCNA 2004).

[23] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre, "Set-based fault detection and isolation for detectable linear parameter-varying systems," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 18, pp. 4381–4397, 2017.

[24] J. K. Scott, D. M. Raimondo, G. R. Marseglia, and R. D. Braatz, "Constrained zonotopes: A new tool for set-based estimation and fault detection," *Automatica*, vol. 69, pp. 126 – 136, 2016.

[25] S. Kousik, A. Dai, and G. X. Gao, "Ellipsotopes: Uniting ellipsoids and zonotopes for reachability analysis and fault detection," *IEEE Transactions on Automatic Control*, pp. 1–13, 2022.

[26] S. Sadraddini and R. Tedrake, "Linear encodings for polytope containment problems," in *IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 4367–4372.