2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 1-5, 2023. Detroit, USA

BRNES: Enabling Security and Privacy-aware Experience Sharing in Multiagent Robotic and Autonomous Systems

Md Tamjid Hossain, Hung Manh La, Shahriar Badsha, and Anton Netchaev

***Abstract*— Although experience sharing (ES) accelerates multiagent reinforcement learning (MARL) in an advisor-advisee framework, attempts to apply ES to decentralized multiagent systems have so far relied on trusted environments and overlooked the possibility of adversarial manipulation and inference. Nevertheless, in a real-world setting, some Byzantine attackers, disguised as advisors, may provide false advice to the advisee and catastrophically degrade the overall learning performance. Also, an inference attacker, disguised as an advisee, may conduct several queries to infer the advisors' private information and make the entire ES process questionable in terms of privacy leakage. To address and tackle these issues, we propose a novel MARL framework (BRNES) that heuristically selects a dynamic neighbor zone for each advisee at each learning step and adopts a weighted experience aggregation technique to reduce Byzantine attack impact. Furthermore, to keep the agent's private information safe from adversarial inference attacks, we leverage the local differential privacy (LDP)-induced noise during the ES process. Our experiments show that our framework outperforms the state-of-the-art in terms of the steps to goal, obtained reward, and time to goal metrics. Particularly, our evaluation shows that the proposed framework is 8.32x faster than the current non-private frameworks and 1.41x faster than the private frameworks in an adversarial setting.**

I. INTRODUCTION

Experience sharing (ES) [1] has become increasingly significant in the multiagent reinforcement learning (MARL) [2] paradigm due to its efficacy in accelerating learning performance. As the popularity of ES processes increases, so do concerns about their security and privacy. Namely, advisors' shared experience shapes the learning behavior and outcomes of an advisee [1]. A shared but malicious experience could mislead an advisee to take incorrect measures during the *experience harvesting (EH)* phase of ES [3], [4]. Likewise, as the shared experience is computed based on the inputs (e.g., reward signal) that commonly rely on advisors' data, an inference attack on those may leak advisors' private information during the *experience giving (EG)* phase of ES [5], [6]. These security (adversarial manipulation) and privacy (adversarial inference) threats, unfortunately, overlooked by many related studies [1], [7]–[10]; can bring down catastrophic consequences on MARL-based safety-critical applications in domains such as robotics [2], cyber-physical systems [11], automotive industries [12], etc. For example, false advising from an advisor car in autonomous driving may make lane-changing ambiguous and lead to severe road accidents for an advisee car [13], whereas an inference attack from an advisee may reveal sensitive data of the advisors [5], [14]. *Therefore, to facilitate a secure and private MARL for next-generation robotic and autonomous systems, a study of the adversarial manipulation and inference threats posed by the current ES process is non-trivial.*

Particularly, from a security perspective, false advising threat is prominent in the decentralized MARL settings, where there is no central authority to ensure the consensus on advice integrity and agents' authenticity, and thus susceptible to Byzantine general problems [15]. Researchers in [3] address this false advising threat from Byzantine advisors in a MARL platform by adopting differential privacy (DP) [16] at the advisee's end. However, a strategic attacker can exploit the DP-noise to conduct optimal false data injection (or simply false advising) attacks and hamper the learning outcomes significantly [17], [18]. *To tackle this, we propose to incorporate the experience, whether it is differentially private or not, into the advisee's learning through a weighted experience aggregation technique.*

From a privacy perspective, we argue that inference attackers, disguised as advisees, could try to infer advisors' sensitive information by recursively querying their experience for every state-action pair. For example, the advisors' experience in Q-value sharing frameworks (e.g., [1], [19], [20]) can reflect their *rewarding strategy* that builds their decision-making criteria, and *movement trajectory* that carries important contextual information, such as users' preference, next course of actions, etc. [5]. *To protect such sensitive information in untrusted environments, unlike [3], we propose to adopt local differential privacy (LDP) [21]* during ES. LDP perturbs advisors' experience before sharing it, making it harder for inference attackers to obtain sensitive information.

Different from the above-mentioned works, our paper presents a novel Byzantine Robust Neighbor Experience Sharing (BRNES) framework that addresses the security and privacy threats in the ES process from two adversarial perspectives: (1) false advising during the advisee's EH, and (2) inference attack during the advisor's EG. Therefore, our contribution in this paper is twofold: in decentralized EH, we address the security attacks from *Byzantine attackers*,

Md Tamjid Hossain and Hung La are with the Advanced Robotics and Automation Lab, Computer Science and Engineering, University of Nevada, Reno, Nevada, USA. Emails: mdtamjidh@nevada.unr.edu, hla@unr.edu

Shahriar Badsha is with Bosch Engineering, North America. Email: shahriar.badsha@us.bosch.com

Anton Netchaev is with the U.S. Army Corps. Email: anton.netchaev@erdc.dren.mil

This work was partially funded by the U.S. National Science Foundation (NSF) under grants: NSF-CAREER: 1846513, and NSF-PFI-TT: 1919127. The views, opinions, findings, and conclusions reflected in this publication are solely those of the authors and do not represent the official policy or position of the NSF.

*Source code of the task and the computational model behind the setup available at https://github.com/aralab-unr/BRNES

and in EG, we address the privacy attacks from inference attackers. To ensure the framework is Byzantine robust, we develop an *adaptive heuristic neighbor zone selection* process for each advisee that limits the possibility of a Byzantine advisor deterministically appearing in the vicinity of any targeted advisee significantly due to the inherent randomness in the process. Additionally, to further limit the false advising impacts from Byzantine advisors, we leverage a *weighted experience aggregation* technique that prevents the direct integration of advisors' experience. To prevent inference attackers from inferring advisors' sensitive data, we leverage the provable privacy guarantee offered by the LDP mechanism. In summary, our contributions are:

- to enable security and privacy-aware ES in MARL, we propose a novel framework (BRNES) that addresses adversarial manipulation and inference problems in existing multiagent robotic and autonomous systems and fills two important research gaps in the literature- (1) *the absence of a Byzantine robust decentralized EH mechanism*, and (2) *the lack of a private EG process*.
- to achieve Byzantine robustness, we formulate a novel adaptive heuristic neighbor zone selection strategy and leverage the weighted experience aggregation technique.
- to make EG privacy-protected, we leverage the provable privacy guarantee offered by the LDP technique.
- comparing to the state-of-the-art (SOTA), we show that our framework is 1.41x faster than DA-RL [20] and 8.32x faster than AdhocTD [1] under adversarial presence.

II. Related Works

ES strategies have been studied extensively to enhance the learning performance of MARL agents [1], [3], [4], [7], [8], [10], [19], [20], [22]–[26]. For instance, the problem of slow convergence of MARL policy is addressed in [7], where, to tackle the slow learning, the authors propose central knowledge transfer units for the participating agents. Similarly, to mitigate the curse of dimensionality in conventional ES-driven MARL platforms, several novel MARL algorithms based on mixed Q-networks [26], simultaneous learning [1], [10], and differential advising [3], [20] have been developed.

Specifically, [1] reduces the number of inter-agent communications by (1) limiting the students to seek advice from the teachers only when their confidence is low for a given state, and (2) limiting the teachers to respond only when they believe they have much knowledge for that state. *Nonetheless, [1] overlooks the possibility of adversarial manipulation and inference, which may impede the success of ES processes in real-life MARL applications.*

An alternative approach to simultaneous learning, the iteration-based Q-learning is proposed in [10], where a *centralized aggregator* forms a swarm matrix containing the extremes of Q-values from all agents. *Nonetheless, centralized aggregation may possess various drawbacks (e.g., single-point-of-failure) despite its fast convergence.*

Intuitively, a decentralized mechanism is more effective in an environment with resource-constrained edge devices than a centralized mechanism. Moreover, decentralization alleviates the single-point-of-failure problem. Considering this, [19] introduces a decentralized and heuristic Q-value advising method called PSAF that addresses *when to ask for the advice, when to give the advice, and how to use the advice* in a teacher-student framework. *However, decentralization may create opportunities for the Byzantine and inference attackers [15].* Field research and experience of MARL application's post-deployment [4], [22] show that any malicious agent, in general, may conduct eavesdropping, inference attacks, Byzantine attacks, etc., creating significant security and privacy challenges for multiagent systems (MAS).

Researchers partially solve the false advising in MARL [3]. They design the adviser selection problem as a Multi-armed bandit and solve it using the DP technique. However, their assumption of eliminating probabilistic false advice by malicious agents through direct DP integration does not hold in the presence of any strategic attacker. The extension of their work involves accommodating the advice from a slightly different state [20]. *Yet, they adopt the DP mechanism for learning performance improvement only, but not to protect the privacy and security of the agents.*

Privacy and security concerns in MAS are addressed in [4], [27]. From the privacy perspective, [27] emphasizes preserving agents' privacy against inference attackers by proposing a DP-MAS framework. From the security perspective, [4] shows that an adversary can mislead honest agents to attain its malicious objectives in a consensus-based MARL platform. *However, both [4], [27] are limited to centralized environments, and thus, cannot apply to decentralized MARL applications.* We summarize major contrasting points between literature and this work in Table I.

TABLE I

MARL FRAMEWORK COMPARISON. SYMBOL: ADDRESSED (✓), NOT ADDRESSED (□). "L"EARNING TYPE ("C"ENTRAL OR "D"ECENTRAL). "H"EURISTIC ADVISING . "A"DVISING CONFIDENCE. "B"UDGET CONSTRAINTS. "F"ALSE ADVISING. "P"RIVACY ATTACKS. "N"EIGHBOR ZONE. "**W**"EIGHTED ADVICE AGGREGATION.

	L		H	A	B	F	P	N	W
	C	D							
Silva et al., 2017 [1]	□	✓	✓	✓	✓	□	□	□	□
Matta et al., 2019 [10]	✓	□	□	□	□	□	□	□	✓
Ye et al., 2020 [3]	□	✓	✓	✓	✓	✓	□	□	□
Figura et al., 2021 [4]	✓	□	□	□	□	✓	✓	□	✓
Zhu et al., 2021 [19]	□	✓	✓	✓	□	□	□	□	□
Li et al., 2021 [27]	✓	□	□	□	□	□	✓	□	✓
Ye et al., 2022 [20]	□	✓	✓	✓	✓	□	□	□	□
This work	□	✓	✓	✓	✓	✓	✓	✓	✓

III. Problem Formulation and Threat Modelling

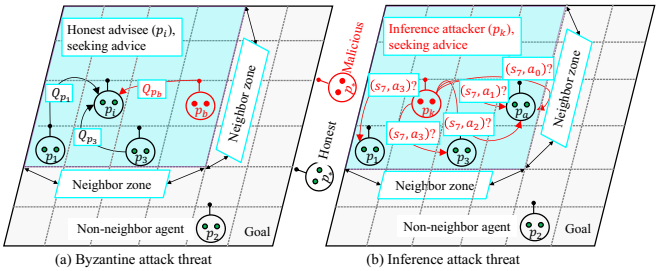
Let us consider $\mathcal{N}$ robotic agents ($\mathcal{N} = \{p_1, ..., p_n\}$), which are learning cooperatively to achieve an objective in environment $\mathbb{E}$ of $\mathcal{H} \times \mathcal{W}$ dimension following a Markov game. The game is represented as a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, \Phi, \gamma, \mathcal{T})$ having state-space $\mathcal{S} = \mathcal{S}_1 \times ... \times \mathcal{S}_n$, joint action space $\mathcal{A} := \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_n$, transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}$,

reward function $\Phi : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, and discount factor $\gamma \in [0, 1]$ for all future rewards. The goal point is $\mathcal{G}$ and the action space is $\mathcal{A} = \{Left, Right, Up, Down\}$. Consider several obstacles $\mathcal{O}_{x=\{0,1,...\}}$ in the environment. If any agent hits the environment boundaries or the obstacles, it would get a penalty, $\phi_O \in \Phi$. However, assume one freeway $\mathcal{F}$ in $\mathbb{E}$, which could be used by any agent to earn a reward before reaching $\mathcal{G}$. Incorporating this freeway structure within the grid-based environmental model enhances the opportunities for agents to accrue supplementary rewards, and invariably introduces additional dimensions of complexity to the task landscape. After reaching $\mathcal{G}$, the agents are rewarded with $\phi_\mathcal{G} \in \Phi$ s.t. $\phi_\mathcal{G} > \phi_\mathcal{F}$ (if $\phi_\mathcal{G} \leq \phi_\mathcal{F}$, the agents would not be motivated to move to the goal). Note that, $|\mathcal{N}|+|\mathcal{O}_\chi|+|\mathcal{F}|+|\mathcal{G}| < \mathcal{H} \times \mathcal{W}$, otherwise the agents cannot move smoothly through the empty spaces of the grid.

The position of the agents, obstacles, and freeway are randomly initialized in each episode. Assume the obstacles randomly change positions at each step, thus making it harder for the agents to learn. The objective completes when all the agents reach $\mathcal{G}$. Any individual agent p_i is spatially aware of the position of $\mathcal{G}$. **p_i's objective is to take the lowest possible steps to goal (SG_{min}) for collecting freeway reward ϕ_f and reaching goal $\mathcal{G}$ without hitting environment boundary or any obstacles $\mathcal{O}_x$ while, also, earning maximum rewards** ($\phi_{max} = \phi_\mathcal{F} + \phi_\mathcal{G} + [\phi_O = 0]$). Thus, p_i's objective can be formalized as (a) $SG_{p_i} = SG_{min}$, (b) $\phi_{p_i} = \phi_{max}$, and (c) $\|(x_{p_i}, y_{p_i}) - (x_\mathcal{G}, y_\mathcal{G})\| = 0$, where $(x_{p_i}, y_{p_i}) \in [(0, 0), (\mathcal{H} \times \mathcal{W})]$ and $(x_\mathcal{G}, y_\mathcal{G}) \in [(0, 0), (\mathcal{H} \times \mathcal{W})]$ are p_i's and $\mathcal{G}$'s position, respectively.

A. Byzantine Attacks during EH

During EH, a Byzantine advisor $p_b \in \mathcal{N}$ may send false information to p_i, with a *malicious objective to impede p_i's convergence* as depicted in Figure 1a. We assume that p_b has the knowledge of $\mathcal{A}, \mathcal{S}, \Phi, (x_\mathcal{G}, y_\mathcal{G})$ and p_i's current state s. Particularly, p_b could promote a larger Q-value for a misleading action $a_m \in A$ than the rest of the actions $\mathcal{A}\backslash\{a_m\}$, i.e., $Q_{p_b}(s^t, a_m) > Q_{p_b}(s^t, a_h) \forall a_h \in \mathcal{A}\backslash\{a_m\}$, thus continuously drive p_i towards a desired malicious point. However, if p_b always shares a set of large Q-values to attain a large incentive, it might be identified easily by any anomaly detector at the advisee's end. On the contrary, if it shares a set of small Q-values, the attack impact might be negligible. This fundamental adversarial tradeoff problem can be tackled in several ways. One approach is to shuffle the Q-values for all actions corresponding to the requested state and inject false noise that is similar to the maximum reward using reward poisoning methods [28]. Another approach is to draw the false noise from an adversarial distribution that has similar statistical properties to a benign noise distribution used for achieving DP [17]. For simplicity, but without losing generality, we choose the former method to generate false advice in this study since any optimal false advising attack method would always involve false data that is difficult to distinguish from benign data.



(a) Byzantine attack threat (b) Inference attack threat

Fig. 1. Threats in MARL: (a) A Byzantine advisor (p_b) providing false information (Q_{p_b}) to the honest advisee (p_i); (b) An inference attacker (p_k) performing multiple queries ($s_7, a_{0,1,...}$) to an advisor (p_a).

B. Inference Attacker during EG

The advisee itself could be an adversary, whose *malicious objective is to infer the private information of an honest advisor $p_a \in \mathcal{N}$ by analyzing p_a's experience* (Figure 1b). We assume that the advisee has the knowledge of $\mathcal{A}, \mathcal{S}, \Phi, (x_\mathcal{G}, y_\mathcal{G})$, but does not know p_a's current state. Specifically, a malicious advisee $p_k \in \mathcal{N}$ could perform multiple queries to p_a's Q-tables for each and every state and action in order to reconstruct p_a's entire Q-table and infer sensitive information related to p_a's residing states, next actions, rewards, and adopted strategies.

IV. BRNES Framework

We model our BRNES framework (Figure 2) for robotic agents which share their experience under the adversarial presence and budget constraints. We use a model-free and off-policy MARL approach, Q-learning, to develop and test our framework in a stochastic environment. We formulate the experience as Q-values instead of the recommended actions since the Q-value advising, unlike the action advising, does not impair the performance of the agent's learning directly [19]. The framework mimics an advisee-advisor network where agents are homogeneous and interchangeable. They have identical strategies, however, they maintain their own Q-table to store their local knowledge. Algorithm 1 presents

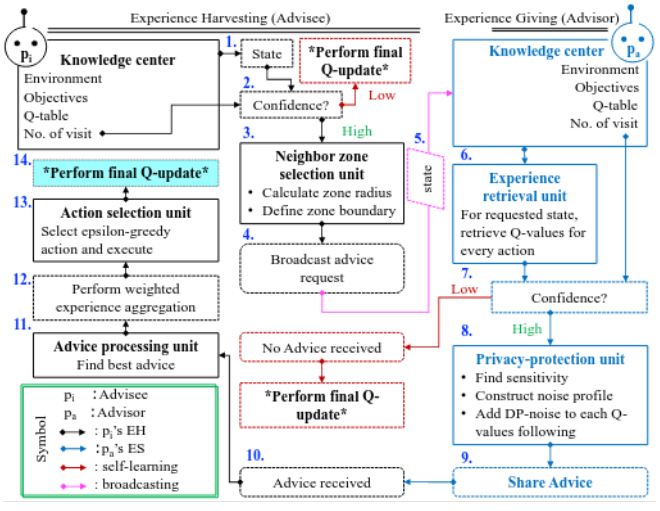


Fig. 2. BRNES framework: Advisee p_i is harvesting the experience while advisor p_a is sharing experience to p_i.

Algorithm 1: Experience harvesting (EH) by advisee p_i. $\mathcal{G}$: goal, p_i: advisee, $\mathbb{E}$: environment, n^v: number of visit, ϵ: probability, α: learning rate, s: state, a: action, Φ: reward set, γ: discount factor, $\mathcal{Z}$: neighbor zone, $\mathcal{N}$: agent set, ξ: best advice, (x, y): position coordinate, w: aggregation factor (weight), B: advice seeking budget, τ, τ', κ: predefined threshold

Require: Environment, $\mathbb{E}$
1 **Initialize** Q-table and **set** ϵ, α, γ
2 **for** *each* $t = 1, 2, ..., T$ *episodes* **do**
3 Observe $s^t_{p_i}$, find $n^v_{p_i} \leftarrow s^t_{p_i}$, and compute $P^a_{p_i} = f(n^v_{p_i}, B_{p_i}, B^{tot}_{p_i}, \tau, \tau')$ from *Algorithm* 2
4 **if** $0 < P^a_{p_i} < \kappa$ **then**
5 Find $\mathcal{Z}^t_{p_i} = NZ(|\mathcal{N}|, s^t_{p_i}, \mathbb{E})$ from *Algorithm* 2
6 Send advise request to neighbors within $\mathcal{Z}^t_{p_i}$
7 **if** *No Advice* **then**
8 Perform final Q-update
9 **else**
10 Receive advice from all k advisors as $\left[Q'_{p_a}(s^t_{p_i})\right]^k_{p_a=1}$ (refer to *Algorithm 3*)
11 Find best advice ($\xi^t_{p_i}$) by grouping & averaging Q-values for each action
12 Perform weighted aggregation $Q_{p_i}(s^t_{p_i}) = w \times Q_{p_i}(s^t_{p_i}) + (1 - w) \times \xi^t_{p_i}$
13 Find ϵ-greedy action & observe $s^{t+1}_{p_i}$, $\Phi^t_{p_i}$
14 Perform final Q-update for selected action $Q^t_{p_i}(s^t_{p_i}, a^t) = (1 - \alpha) Q^t_i(s^t_{p_i}, a^t) + \alpha(\Phi^t_{p_i} + \gamma * \underset{a^{t+1}}{max}\, Q^t_{p_i}(s^{t+1}_{p_i}, a^{t+1}))$
15 **end**
16 **else**
17 Take action and perform final Q-update
18 **end**
19 Set $s^t_{p_i} = s^{t+1}_{p_i}$ `// update to next state`
20 **if** $\|(x_{p_i}, y_{p_i}) - (x_{\mathcal{G}}, y_{\mathcal{G}})\| > 0$ **then** Continue
21 **else** End episode and reset environment
22 **end**

Algorithm 2: Sub-functions.

1 **Function** $f\,(n^v_{p_i}, B_{p_i}, B^{tot}_{p_i}, \tau, \tau')$ **:**
2 **return** $P^a_{p_i} = \begin{cases} \frac{1}{\sqrt{n^v_{p_i}}} \cdot \sqrt{\frac{B_{p_i}}{B^{tot}_{p_i}}}, & \tau \le n^v_{p_i} \le \tau' \\ 0, & Otherwise \end{cases}$
3 **Function** $NZ\,(|\mathcal{N}|, s^t_{p_i}, \mathbb{E})$ **:**
4 **Find** p_i's position at t, i.e., $(x^t_{p_i}, y^t_{p_i}) \leftarrow s^t_{p_i}$
5 **Find** height($\mathcal{H}$), width($\mathcal{W}$) $\leftarrow \mathbb{E}$
6 **Calculate** zone radius, $r^t_{p_i} = \sqrt{\frac{\mathcal{H} \times \mathcal{W}}{|\mathcal{N}|}}$
7 **Define** zonal boundary lines, $\mathcal{Z}^t_{p_i} = \left[x^t_{p_i} \pm r^t_{p_i}, y^t_{p_i} \pm r^t_{p_i}\right]$ $\forall 0 \le (x^t_{p_i} \pm r^t_{p_i}) \le \mathcal{W}$ and $0 \le (y^t_{p_i} \pm r^t_{p_i}) \le \mathcal{H}$
8 **return** $\mathcal{Z}^t_{p_i}$

the pseudocode for the EH phase. Algorithm 2 outlines the required sub-functions and Algorithm 3 shows the LDP adaptation technique during the EG phase.

A. Experience Harvesting (EH) Process

To tackle the adversarial manipulation, it is necessary to ensure that no particular advisor frequently appears in the close vicinity of advisee p_i for multiple episodes and that their advice is not directly integrated into p_i's Q-learning. Considering this, at timestamp t, p_i first observes its current state initialized by a stochastic initialization process (i.e., at every episode, all agents appear in random states, thus limiting the consecutive attack opportunity over a targeted agent) (Figure 2, step 1). Then, p_i computes its experience harvesting confidence (EHC), $P^a_{p_i}$ (Algorithm 1, line 3). The advisee seeks advice from the experienced advisors in its neighborhood only when- (1) its knowledge of that particular state is low, and (2) it has the budget to seek advice.

1) Computing Experience Harvesting Confidence (EHC): p_i's EHC can be calculated as Algorithm 2, line $1 - 2$ [20], where p_i's current and total communication budget are B_{p_i} and $B^{tot}_{p_i}$, respectively. The user-defined threshold τ prevents p_i to avoid spending all of its budgets in the early episodes and τ' prevents p_i to avoid seeking advice for the highly-visited states. Function, f provides a higher probability for the states that the advisee visits rarely and vice versa. p_i performs final Q-update if $P^a_{p_i}$ is zero. Otherwise (i.e., $0 < P^a_{p_i} < \kappa$ where κ is a predefined threshold), it proceeds to the next steps as shown in line $5 - 14$ of Algorithm 1.

2) Selecting Adaptive Heuristic Neighbor Zone: To avoid any specific agent from frequently appearing in the neighbor zone, p_i computes the radius of the neighbor zone based on the environment's dimensions and the total number of agents (Algorithm 2, line $4 - 7$). Since we use a 2D grid space, we only consider the x and y coordinates of the environment when calculating the neighbor zone. However, in more complex environments with multiple dimensions, the neighbor zone to those dimensions could be extended. If there are few agents in a large grid space, the zone radius would be large, but if the agent number increases or the grid space gets smaller, the zone radius would become smaller. The boundary of the zone is calculated at each timestamp, and it is adjusted as p_i moves to a new state in each episode. Since the zone size is dynamic and shifts with p_i's movement, the chance of the same manipulative advisor repeatedly appearing in p_i's neighbor zone is reduced.

3) Performing Weighted Experience Aggregation: Advisee p_i seeks advice from the agents residing in its neighbor zone. If no advice is received, the EH process is terminated, and the final Q-update is computed. Nonetheless, if p_i receives advice, then it computes the best advice set of Q-values ($\xi^t_{p_i}$) by grouping and averaging Q-values (i.e., $\xi^t_{p_i} \leftarrow \frac{1}{n} \sum^k_{p_a=0} Q_{p_a}$) for every action (Algorithm 1, line $10 - 11$). After that, p_i incorporates the best advice into its Q-table following a weighted linear combination process (Algorithm 1, line 12). The degree of advice is controlled by a user-defined weight factor $w \in [0, 1]$. This ensures that even if any Byzantine advisor p_b provides false information with the highest Q-value, it should not affect p_i's learning significantly. Next, p_i performs the conventional ϵ-greedy action and observes the next state and reward. Finally, p_i performs the final Q-update and update its state (Algorithm

1, line $13-14$, and 19).

B. Experience Giving (EG) Process

When advisor p_a receives an advice request from an advisee p_i for any state, it has to solve the following problems: (1) whether it is confident enough to provide the advice, and (2) if it is safe to provide the advice.

1) Computing Experience Giving Confidence (EGC): To tackle the first problem, we use the experience giving confidence (EGC) process described in Algorithm 2, line 2. Specifically, p_a computes a probability of giving advice, $P^g_{p_a}$ based on its knowledge about that state (i.e., visit time, $n^v_{p_a}$ and advice giving budget, B_{p_a}. If $P^g_{p_a}$ is zero, p_a does not provide any advice to p_i.

2) Incorporating Local Differential Privacy (LDP): To solve the adversarial inference problem, advisor, p_a uses the DP technique that ensures that the output of an algorithm is not affected by small changes in input data from individual users. DP is typically set up in a way that involves a trusted third party, who collects data, adds noise to the query results in a way that meets the DP requirements, and then releases the noisy results. Nonetheless, in practice, finding a trusted third party could be difficult [21]. For example, in our threat model, the advisee itself could be an untrusted party. To address this issue, the ε-LDP mechanism [21], a variant of the basic DP technique [16], emerges. ε-LDP applies the DP property locally to each user's data following a predefined privacy budget (ε) without the need for a trusted third party, rather than to the data as a whole. The formal definition of the ε-LDP mechanism can be given as [21]:

Definition 1: A randomized mechanism $\mathcal{M}$ satisfies ε-LDP if for any pairs of input values x and x' in the domain of $\mathcal{M}$, and for any possible output $y \in \mathcal{Y}$, it holds

$$\mathbb{P}\left[\mathcal{M}(x) = y\right] \leq e^{\varepsilon} \cdot \mathbb{P}\left[\mathcal{M}(x') = y\right], \tag{1}$$

where $\mathbb{P}[\cdot]$ denotes probability, $\mathcal{Y}$ denotes output domain, and ε is the privacy budget. The smaller the ε, the stronger the privacy protection, but the weaker the data utility, and vice versa. ε-LDP allows advisors to have *plausible deniability* whether or not the advisee is compromised. It satisfies the sequential property that facilitates the development of complex LDP algorithms from simpler subroutines and can be described as [21]:

Theorem 1: If $\mathcal{M}_i(x)$ is an ε_i-LDP algorithm for x and $\mathcal{M}(x)$ is the sequential composition of $\mathcal{M}_1(x), ..., \mathcal{M}_n(x)$, then $\mathcal{M}(x)$ satisfies ε-LDP for $\varepsilon = \sum_{i=1}^{n} \varepsilon_i$.

Further details and the proof of Theorem 1 can be found in [21]. The fundamental mechanism to achieve ε-LDP is the randomized response (RR) [29], a generalized version of which is *Generalized Randomized Response (GRR)*, [29]. GRR is also described as a special Direct Encoding (DE) method and a generalization of k-randomized response [29]. In GRR, given the domain size $d = |\mathcal{D}|$ and privacy budget, ε, the following perturbation probability ensures ε-LDP [29].

$$Pr\left[\mathcal{M}_{GRR}(x) = y\right] = \begin{cases} \frac{e^{\varepsilon}}{d+e^{\varepsilon}-1} & \text{if } y = x \\ \frac{1}{d+e^{\varepsilon}-1} & \text{otherwise} \end{cases} \tag{2}$$

Algorithm 3: Experience giving (EG) by advisor, p_a.
n^v: no. of visits, ε: privacy budget, s: state, a: action, B: advice budget, η: LDP-noise, d: domain size

Require: $s^t_{p_i}, n^v_{p_a}, \varepsilon$
Receive advice request for state $s^t_{p_i}$ from advisee p_i
$P^g_{p_a} = \begin{cases} 1 - \frac{1}{\sqrt{n^v_{p_a}}} \cdot \sqrt{\frac{B_{p_a}}{B^{tot}_{p_a}}}, & n^v_{p_a} > n^v_{p_i} \\ 0, & \text{Otherwise} \end{cases}$
if $P^g_{p_a} > 0$ **then**
  for *each Q-value, x in set* $Q_{p_a}(s^t_{p_i})$ **do**
    b = random.random()
    if $b \leq e^{\frac{\varepsilon}{n}}/(d + e^{\frac{\varepsilon}{n}} - 1)$ **then** $Q'_{p_a}(s^t_{p_i}) \leftarrow x$
    else $Q'_{p_a}(s^t_{p_i}) \leftarrow$ Uniform$(Q_{p_a}(s^t_{p_i})/x)$
  end
  return $Q'_{p_a}(s^t_{p_i})$
end
else return No Advice

Theorem 2: GRR satisfies ε-LDP.

Proof: To satisfy ε-LDP, the ratio of the probabilities for $x, x' \in \mathcal{D}$ needs to be equal to e^{ε}. Here, we have

$$\frac{Pr\left[\mathcal{M}_{GRR}(x) = y\right]}{Pr\left[\mathcal{M}_{GRR}(x') = y\right]} = \frac{\frac{e^{\varepsilon}}{d+e^{\varepsilon}-1}}{\frac{1}{d+e^{\varepsilon}-1}} = e^{\varepsilon} \tag{3}$$

which satisfies the condition of ε-LDP. ■

In our setting, the advisors follow the GRR-based perturbation mechanism to achieve ε-LDP since GRR directly takes the original value as input into the perturbing step without the need for the encoding process. Following Definition 1, if we assume the set of all Q-values for every action in a particular state, s_{p_i} as a dataset, $\mathcal{D}_a = \{Q_{a_1}, ..., Q_{a_n}\}$, then the corresponding private (perturbed) dataset, $\mathcal{D}'_a = \{Q'_{a_1}, ..., Q'_{a_n}\}$. The original Q-values $\{Q_{a_x}\}^n_{x=1}$ and private Q-values $\{Q'_{a_x}\}^n_{x=1}$ are linked by the privacy preservation mechanism $\mathcal{M}_{GRR}$. Here, Q'_{a_x} depends only on Q_{a_x}; and not on any other Q-values Q_{a_y} or Q'_{a_y} for $y \neq x$. Therefore, this noninteractive framework can be given as

$$Q'_{a_x} \leftarrow Q_{a_x} \text{ and } Q'_{a_x} \perp \{Q_{a_y}, Q'_{a_y}, y \neq x\} | Q_{a_x}, \tag{4}$$

where $\perp$ denotes the symbol of noninteractive relation. Algorithm 3 shows the pseudocodes (line 4 to 7) of the GRR mechanism for Q-value sharing. Given a privacy budget, ε, an original Q-value set D_a, the algorithm returns a perturbed Q-value set D'_a. Nonetheless, for any two neighboring Q-value sets of equal length (e.g., $\mathcal{D}_a = \{Q_{a_1}, ..., Q_{a_n}\}$, $\mathcal{D}_b = \{Q_{b_1}, ..., Q_{b_n}\}$, and $|\mathcal{D}_a| = |\mathcal{D}_b| = n$), the changes can occur for maximum n positions. Therefore, the sensitivity of the mechanism is n here. Specifically, the mechanism keeps a particular Q-value unchanged (i.e., $Q'_{a_x} \leftarrow Q_{a_x}$) with a probability, $p = \frac{e^{\varepsilon/n}}{d+e^{\varepsilon/n}-1}$ and perturbs it to a different random Q-value (i.e., $Q'_{a_x} \leftarrow$ Uniform$(\mathcal{D}_a/Q_{a_x})$) with probability $q = \frac{1}{d+e^{\varepsilon/n}-1}$.

Proposition 1: The proposed EG method satisfies ε-LDP.

Proof: The algorithm applies the GRR mechanism separately to each Q-value of a state learned by an advisor. If $\mathcal{M}_i(.)$ is applied on a particular Q-value, $x \in$

$Q_{p_a}(s^t_{p_i})$, where $|Q_{p_a}(s^t_{p_i})| = n$ and the output is y, then

$$\frac{Pr\left[\mathcal{M}_i(x) = y\right]}{Pr\left[\mathcal{M}_i(x') = y\right]} = \frac{\frac{e^{\varepsilon/n}}{d+e^{\varepsilon/n}-1}}{\frac{1}{d+e^{\varepsilon/n}-1}} = e^{\varepsilon/n} \tag{5}$$

Therefore, following Theorem 2, $\mathcal{M}_i(.)$ satisfies $\frac{\varepsilon}{n}$-LDP. Now, if we consider $\varepsilon_i = \frac{\varepsilon}{n}$, then we can combine n subroutines (each satisfying ε_i-LDP independently) for n number of Q-values by following the sequential property of ε-LDP given in Theorem 1 for our EG algorithm and show that EG satisfies $\sum_{i=1}^{n} \varepsilon_i = n \cdot \varepsilon_i = n \cdot \frac{\varepsilon}{n} = \varepsilon$-LDP. ■

Remark. In our experimental setting, there are four Q-values for four corresponding actions (*Left, Right, Up, Down*). Thus, $|\mathcal{D}_a| = 4$. Also, the maximum difference between two adjacent Q-value sets would be 4. Hence, the applied GRR mechanism for each Q-value in a set satisfies $\frac{\varepsilon}{4}$-LDP, ensuring the overall EG method satisfies $4\text{x}\frac{\varepsilon}{4} = \varepsilon$-LDP.

V. Experimental Analysis

We implement our framework following a modified predator-prey domain [24]. Next, we compare our results with two SOTA approaches: **AdhocTD** [1], *which proposes visit-based advising, but neither adopts DP nor incorporates weighted experience aggregation during ES*; and **DA-RL** [20], *which proposes a differential advising method but does not incorporate any neighbor zone concept and/or weighted experience aggregation technique to enable security and privacy-aware ES*.

Our environment is a $\mathcal{H} \times \mathcal{W}$ grid world with multiple agents and one goal. Agents have four actions to choose, *Left, Right, Up, Down* to move from one cell to another. They can collect additional rewards upon visiting a freeway on the path to the goal. Nonetheless, grid obstacles can cause penalties upon encounter. Moreover, the agents get penalties if they hit any grid boundary. The positions of the agents, obstacles, and freeway are initialized randomly at the beginning of every episode. The game ends when all the agents reach the goal. Nonetheless, if the agents do not reach the goal within a predefined (grid size $\times$ 100) steps, the environment is reset. While we demonstrate our work in a grid world, it is also extendable to real-world domains with sensitive data. Table II lists the parameters we have used during our experiment.

We investigate the impact of the environment in three scenarios of different scales: (1) **small-scale:** 5×5 grid with 5 agents, 1 obstacle, and 1 freeway, (2) **medium-scale:** 10×10 grid with 10 agents, 3 obstacles, and 1 freeway, and (3) **large-scale:** 30×30 grid with 20 agents, 5 obstacles, and 1 freeway. We also consider varying percentages of attackers in each environment (e.g., no attacker, 20% attackers, etc.). To evaluate and compare our results with AdhocTD [1] and DA-RL [20], we use three popular metrics [19]: **Steps to goal (SG)**, **Reward**, and **Time to goal (TG)**. SG is the average number of steps needs to reach the goal, Reward is the total average incentive earned, and TG is the total average learning time (in seconds) before reaching the goal. The experiments were conducted on a Lambda Tensorbook equipped with an 11th Gen Intel(R) Core(TM) i7-11800H @2.30GHz CPU, RTX 3080 Max-Q GPU, 64 GB RAM, 2 TB storage, Windows 10 pro (64-bit) OS, Python 3.9.7, and PyTorch 1.10.0+cpu.

TABLE II

Parameter value. α: learning rate, ϵ: exploration-exploitation probability, γ: discount factor, B: communication budget, w: aggregation factor, τ, τ', κ: predefined threshold, ϕ: reward, ε: privacy budget.

Parameter	α	ϵ	γ	$B^{tot}_{p_i}$	$B^{tot}_{p_a}$	w	τ
Value	0.10	0.08	0.80	100,000	10,000	0.85	100
Parameter	$\phi_{\mathcal{G}}$	$\phi_{\mathcal{F}}$	$\phi_{\mathcal{O}}$	$\phi_{\mathcal{W}}$	ε	κ	τ'
Value	10.0	0.50	-1.50	-0.50	1.0	0.1	100,000

A. Trajectory Analysis

We perform a trajectory analysis of the agents. The result is illustrated in Figure 3. The cells with darker colors have been visited more frequently than the cells with lighter colors. It can be inferred that all of the agents have visited the cells that are closer to the goal more frequently as compared to the cells that are far distant from the goal. Another interesting fact is that in most cases, the agents have a lower tendency to visit the boundary cells, which in turn provides evidence that the agents have learned to avoid hitting the grid boundaries and getting penalties.

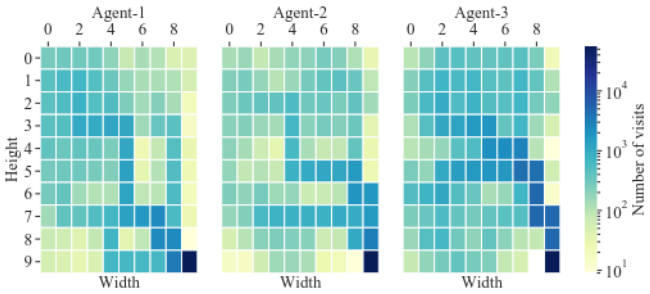


Fig. 3. Visiting trajectory of the agents.

B. Steps to Goal (SG) and Reward Analysis

Figure 4a-4d reflects the average SG values and corresponding rewards of our framework (BRNES), AdhocTD [1], and DA-RL [20] in a medium-scale environment under *no attacker* and *multiple attackers* scenarios. Lower SG values indicate that the agents reach the goal more quickly, and vice versa. When there is no attacker (Figure 4a), all of the frameworks have lower SG values from early episodes (i.e., < 200 episodes). Particularly, AdhocTD [1] exhibits the most stable performance in *no attacker* cases (Figure 4a). This is mostly because it does not incorporate any DP noise and thus, incurs zero privacy cost. Nonetheless, despite having some privacy overhead, BRNES continues to closely follow AdhocTD [1] and outperforms DA-RL [20] for *no attacker* case. In contrast, as soon as Byzantine advisors appear, the SG values of AdhocTD [1] rapidly grow. The more the concentration of the attacker among the agents, the more the SG values. This can be observed in Figure 4b-4d, which illustrates that *BRNES outperforms both AdhocTD [1] and DA-RL [20] in multiple attacker scenarios*. Reward graphs,

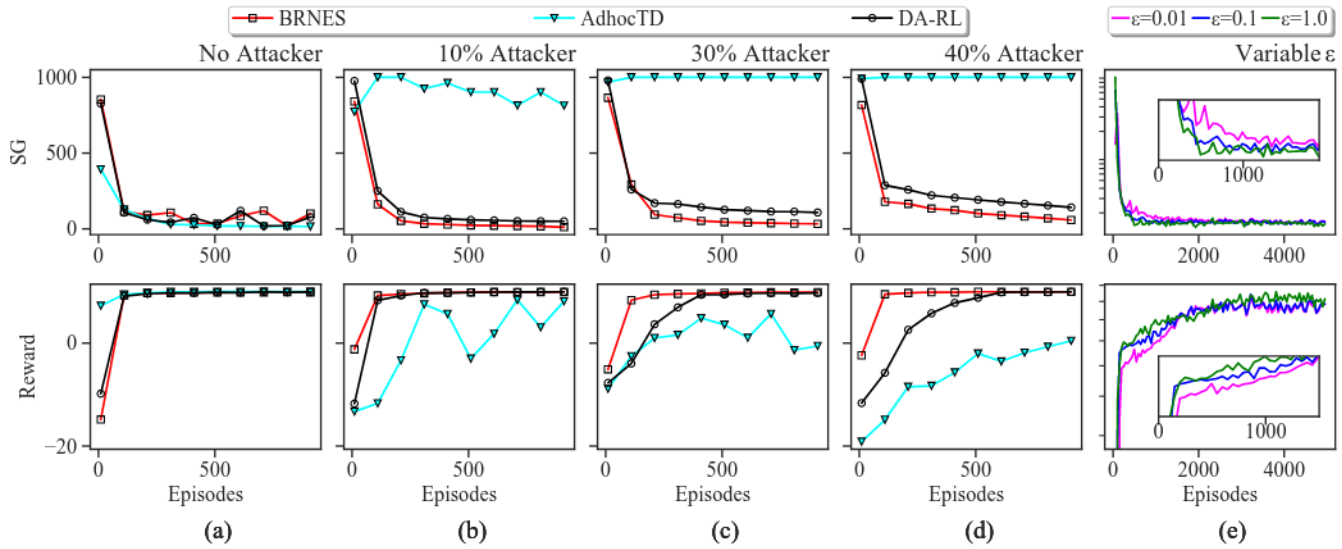


Fig. 4. Steps to goal (SG) and Reward comparison for 1000 episodes among AdhocTD [1], DA-RL [20] and Our (BRNES) framework. Environment ($\mathbb{E}$) feature: $[(\mathcal{H} \times \mathcal{W}) : (10 \times 10), \mathcal{G} : 1, \mathcal{F} : 1, \mathcal{O} : 3, |\mathcal{X}| = 10]$. (a) Baseline scenario (No Attacker), (b)-(d) Multiple Attackers, (e) Variable privacy budget.

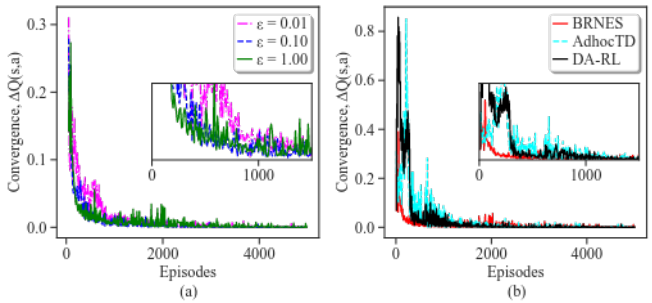


Fig. 5. (a) Convergence is faster when privacy is low (i.e., large ε), (a) BRNES converges faster than AdhocTD [1] and DA-RL [20]. Both (a) and (b) are in a medium-scale environment with 30% attackers.

underneath the corresponding SG graphs, exhibit similar results. Specifically, in Figure 4b-4d where AdhocTD [1] and DA-RL [20] obtained optimal reward after approximately $300, 400,$ and 600 episodes, BRNES continues to indicate significant improvement in learning by obtaining optimal rewards in earlier episodes.

C. Impact of Privacy Budget

We evaluate our framework for multiple values of privacy budget, ε. As shown in Figure 4e, BRNES performs better for higher ε (i.e., low privacy regime) in terms of both SG and Reward. Also, Figure 5a depicts that the convergence happens faster for $\varepsilon = 1.00$ compared to $\varepsilon = 0.01$, which also supports the privacy-utility tradeoff scenario of DP, i.e., higher privacy, lower utility, and vice versa.

D. Convergence Analysis

Convergence analysis under adversarial presence is depicted in Figure 5b. It is evaluated based on the average values of the $\Delta Q(s^t, a^t)$ for all $\Delta Q = Q(s^{t+1}, a^{t+1}) - Q(s^t, a^t)$. The key idea is to show the Q-values are converging into the optimal Q value (Q^*). For simplicity, we only present the deterministic case, in which $Q(s^{t+1}, a^{t+1})$ converges to $Q^*(s, a)$. Therefore, if the average of $\Delta Q(s, a)$ goes to zero, BRNES can be considered stable. From Figure 5b, it can be seen that $\Delta Q(s, a)$ gradually goes to zero. Nonetheless, while AdhocTD [1] and DA-RL [20] are converging after around 900 and 400 episodes respectively, BRNES converges faster (i.e., in < 200 episodes).

E. Time to Goal (TG) Analysis

TG value comparison is presented in Figure 6a and Table III. BRNES requires the lowest time for the agents to reach the goal, except for 0% attackers cases since it deploys LDP-noise to enable private experience sharing, which leads to noisy Q-values. In addition to this privacy cost, the neighbor zone selection and weighted aggregation technique also incur some computational overhead. Nonetheless, this overhead becomes insignificant for BRNES as compared to other frameworks under adversarial presence (Figure 6a and Table III). Particularly, for 40% attacker case in a medium scale-environment, BRNES is $(15640.1/1877.8) \approx 8.32$x faster than AdhocTD [1], and $(2660.2/1877.8) \approx 1.41$x faster than DA-RL [20] in terms of TG value metric.

F. Protection from Inference Attacks

To empirically evaluate the effectiveness of our LDP-driven BRNES framework against inference attacks, we compare multiple ε scenarios with a baseline Non-LDP scenario We observe how accurately and quickly an attacker

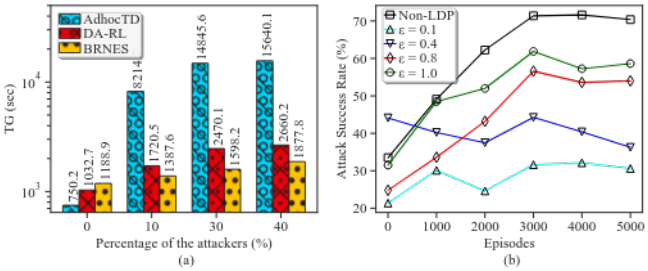


Fig. 6. (a) TG comparison under adversarial presence, (b) Inference attack success rate (%) is the lowest when privacy is the highest (i.e., $\varepsilon = 0.1$).

TABLE III
EXPERIMENTAL RESULT FOR TIME TO GOAL (TG).

Environment Type	Attacker (%Agent)	AdhocTD [1] (TG (sec)	DA-RL [20] (TG (sec)	BRNES (TG (sec)
small-scale	0%	**91.3**	699.7	552.4
	20%	3125.1	776.5	**584.9**
	40%	4170.2	970.1	**754.2**
medium-scale	0%	**750.2**	1032.7	1188.9
	30%	14845.5	2470.1	**1598.2**
	40%	15640.1	2660.2	**1877.8**
large-scale	0%	**45487.5**	71479.4	61693.8
	30%	164852.8	95172.9	**73740.8**
	40%	245425.7	146295.7	**103787.3**

could infer the movement of an advisor by performing repeated advising requests. The results, as shown in Figure 6b, demonstrate that the Non-LDP baseline scenario allows an attacker to achieve a success rate of approximately over 70% within 3000 episodes. However, as we adopt LDP through our proposed framework and increase privacy protection (i.e., decrease ε), the attack success rate decreases significantly.

VI. CONCLUSION

In this study, to mitigate the adversarial impact during experience sharing in CMARL, we propose a novel framework, BRNES, that strategically incorporates neighbors' experiences for effective and faster convergence. Our framework outperforms the SOTA approaches in terms of steps to goal (SG), reward, and time to goal (TG) while achieving ε-LDP to mitigate inference attacks. **Specifically, our framework achieves 8.32x faster TG than a non-private framework, AdhocTD [1], and 1.41x faster TG than a private framework, DA-RL [20] in a medium-scale environment under adversarial presence.**

Several interesting extensions emerge for future privacy and security research in MARL, including analyzing adversarial activity in fully cooperative or competitive and mixed cooperative-competitive environments. Our framework could be extended to a more dynamic environment, where agents receive new tasks when they complete their current tasks.

REFERENCES

[1] F. L. Da Silva, R. Glatt, and A. H. R. Costa, "Simultaneously learning and advising in multiagent reinforcement learning," in *Proceedings of the 16th conference on autonomous agents and multiagent systems*, ser. AAMAS '17, 2017, pp. 1100–1108.

[2] H. M. La, R. Lim, and W. Sheng, "Multirobot cooperative learning for predator avoidance," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 52–63, 2014.

[3] D. Ye, T. Zhu, W. Zhou, and P. S. Yu, "Differentially private malicious agent avoidance in multiagent advising learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 10, pp. 4214–4227, 2020.

[4] M. Figura, K. C. Kosaraju, and V. Gupta, "Adversarial attacks in consensus-based multi-agent reinforcement learning," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3050–3055.

[5] B. Wang and N. Hegde, "Privacy-preserving q-learning with functional noise in continuous spaces," vol. 32, 2019.

[6] K. Prakash, F. Husain, P. Paruchuri, and S. Gujar, "How private is your rl policy? an inverse rl based analysis framework," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 8009–8016.

[7] M. Mahdavimoghadam, A. Nikanjam, and M. Abdoos, "Improved reinforcement learning in cooperative multi-agent environments using knowledge transfer," *The Journal of Supercomputing*, vol. 78, no. 8, pp. 10 455–10 479, 2022.

[8] Y. Li, Y. Zheng, and Q. Yang, "Cooperative multi-agent reinforcement learning in express system," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 805–814.

[9] A. Hussein, E. Petraki, S. Elsawah, and H. A. Abbass, "Autonomous swarm shepherding using curriculum-based reinforcement learning." in *AAMAS*, 2022, pp. 633–641.

[10] M. Matta, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Re, F. Silvestri, and S. Spanò, "Q-rts: a real-time swarm intelligence based on multi-agent q-learning," *Electronics Letters*, vol. 55, no. 10, pp. 589–591, 2019.

[11] A. Prasad and I. Dusparic, "Multi-agent deep reinforcement learning for zero energy communities," in *2019 IEEE PES innovative smart grid technologies Europe (ISGT-Europe)*. IEEE, 2019, pp. 1–5.

[12] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2022.

[13] W. Zhou, D. Chen, J. Yan, Z. Li, H. Yin, and W. Ge, "Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic," *Autonomous Intelligent Systems*, vol. 2, 12 2022.

[14] M. Farhan, A. Y. Rahman, and N. McFarlane, "A ph sensing system with security enhanced cryptographic system," in *Proceedings of the 66th IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2023.

[15] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the works of leslie lamport*, 2019.

[16] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12.

[17] J. Giraldo, A. Cardenas, M. Kantarcioglu, and J. Katz, "Adversarial classification under differential privacy," in *Network and Distributed Systems Security (NDSS) Symposium 2020*, 01 2020.

[18] M. T. Hossain, S. Badsha, and H. Shen, "Privacy, Security, and Utility Analysis of Differentially Private CPES Data," in *2021 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2021, pp. 65–73.

[19] C. Zhu, H.-F. Leung, S. Hu, and Y. Cai, "A q-values sharing framework for multi-agent reinforcement learning under budget constraint," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 15, no. 2, pp. 1–28, 2021.

[20] D. Ye, T. Zhu, Z. Cheng, W. Zhou, and P. S. Yu, "Differential advising in multiagent reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 5508–5521, 2022.

[21] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" in *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008, pp. 531–540.

[22] M. Cheng, C. Yin, J. Zhang, S. Nazarian, J. Deshmukh, and P. Bogdan, "A general trust framework for multi-agent systems," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '21, 2021, pp. 332–340.

[23] F. L. Da Silva, M. E. Taylor, and A. H. R. Costa, "Autonomously reusing knowledge in multiagent reinforcement learning." in *IJCAI*, ser. AAMAS '18, 2018, pp. 5487–5493.

[24] H. M. Le, Y. Yue, P. Carr, and P. Lucey, "Coordinated multi-agent imitation learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1995–2003.

[25] H. Nguyen, H. M. La, and M. Deans, "Hindsight experience replay with experience ranking," in *2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2019, pp. 1–6.

[26] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2018, pp. 4295–4304.

[27] Q. Li, B. Guo, and Z. Wang, "A privacy-preserving multi-agent updating framework for self-adaptive tree model," *Peer-to-Peer Networking and Applications*, vol. 15, no. 2, pp. 921–933, 2022.

[28] X. Zhang, Y. Ma, A. Singla, and X. Zhu, "Adaptive reward-poisoning attacks against reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 11 225–11 234.

[29] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 127–143.