Secure Multi-Robot Adaptive Information Sampling

Tamim Samman, James Spearman, Ayan Dutta,
O. Patrick Kreidl, Swapnoneel Roy, Ladislau Bölöni

***Abstract*— In a coordinated multi-robot information sampling scenario, robots often share their collected information with others for a better prediction. As with any other online data sharing technique, data integrity is a concern, but it has not yet been addressed in the multi-robot information sampling literature. In this paper, we study *how* to secure the information being shared among the robots in such a multi-robot network against integrity attacks and *what* is the cost of integrating such security techniques. To this end, we propose a Blockchain-based information sharing protocol that helps the robots reject fake data injection by a malicious entity. On the other hand, optimal information sampling is a compute-intensive technique and so are the popular Blockchain-based consensus protocols. Therefore, we also study the impact of adding such a security protocol on the execution time of the sampling algorithm, which in turn effects the energy spent by the robots. Results show that our proposed technique is effective against such data tampering attempts while the effect of the added computation varies largely on the consensus protocol used.**

I. Introduction

In today's era of automation, mobile robots are being deployed for collecting meaningful information from an environment. This has high practical relevance in precision agriculture, search and rescue, monitoring, among others [6], [10]. Such information collection helps human users to make more informed decisions/actions. A single robot usually does not have enough capabilities to complete all the relevant tasks and, therefore, multiple low-cost robots are being used. This poses additional challenges of coordination and communication. In this paper, we study such a multi-robot coordination problem, namely multi-robot information sampling where the objective is the following: Given n mobile robots and a budget B, plan n B-length paths for the robots such that the collected information is maximized [6], [7], [18], [24]. Each robot is equipped with an information collection sensor (e.g., camera, radiation detector) and they sense information along their paths. The plan is adaptive in a sense that past observations made by the robot(s) affect the future decision making about where to collect more information. In a multi-robot system, a single robot's future planning is not only affected by its own past sensed data, but also the observations made by the other robots. To plan such optimal paths is proven to be NP-Hard and, therefore, greedy heuristics for navigation are popularly employed [6], [10], [13], [26].

T. Samman, J. Spearman, A. Dutta, O.P. Kreidl, and S. Roy are with the University of North Florida, USA. Emails: {n01379084, n01390826, a.dutta, patrick.kreidl, s.roy}@unf.edu
L. Bölöni is with the University of Central Florida, USA. Email: {ladislau.boloni}@ucf.edu

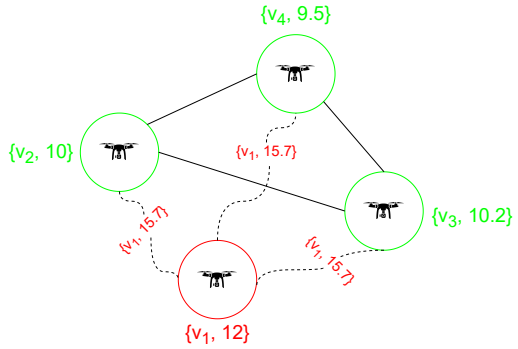

Fig. 1. An instance of a data integrity attack, showing the observed data and the corresponding IDs beside the nodes (with $n = 4$). The red circled robot has been compromised and the malicious actor sends tampered data to its neighbor robots (green circled) – the measured information in location v_1 is 12, but the communicated measurement (displayed on the edges) is 15.7. This, in turn, can lead to degraded future estimates of the underlying information field. The uncompromised green robots broadcast their true collected measurements, but its not shown for brevity.

Navigation planning aside, the multi-robot setting typically assumes the observed data is shared among the robots, which is vulnerable to cyber attacks. Such attacks can have significant financial and ecological impacts [9]. For example, in precision agriculture, farmers use robots' collected data to decide where to spray herbicides in the field to kill weeds. However, if a malicious entity breaches the integrity of the collected data, the farmers can use other physical systems to spray herbicides on good crops and kill them instead of the unwanted vegetation. The problem of multi-robot information collection is under active study but, to the best of our knowledge, how to formally maintain the integrity of the collected information against adversarial influence (see Fig. 1) has not yet been studied.

To this end, we propose a Blockchain-based *secure* multi-robot information sampling framework that is resilient against such data integrity attacks. We employ two popular Blockchain consensus protocols, namely Proof-of-Work (PoW) and Proof-of-Stake (PoS), that help the robots to make decisions based *only* on the untampered data [3], [8], [11], [16]. The tampered data is detected using the above-mentioned consensus protocols and removed from the database. Integrating such Blockchain-based security protocol comes with a cost—the consensus protocols (especially PoW) are known to be compute-intensive and in themselves drain from the robots' on-board power sources. We thus also analyze the additional energy consumption implied by secure sampling, assessing also the cost of our achieved resilience.

For information collection, we employ a popular greedy strategy that is known to yield theoretically-bounded performance [2], [6], [26]. However, it is worth noting that our proposed security technique is generic in nature and can easily be integrated with more sophisticated algorithmic sampling approaches. We test our proposed framework in simulation with up to ten robots and benchmarked the results against an insecure baseline, where data integrity attacks are not prevented. The results show that our proposed technique is always able to detect malicious data received from a compromised robot and then discard it. On the other hand, the added time consumption by our secure variant is significantly higher—up to 46.91 times using PoW.

In summary, this paper's main contributions are:

1) To the best of our knowledge, the first work that studies the secure multi-robot information sampling problem, which is significant due to its sheer practical relevance.
2) A first integration of information sampling and Blockchain-based security techniques, the latter using two consensus protocols (PoW and PoS) and studying via simulations the benefit and cost of each.

II. Background

Mobile robots with on-board sensors (e.g., radiation detectors, hyperspectral cameras) can be deployed to collect information from an unknown environment. The robots visit B "best" information collection locations such that a given optimization criterion, (e.g., entropy or mutual information) is maximized. In this paper, we employ a myopic greedy entropy maximization technique for such sampling, which has been shown in the literature to be efficient [2], [6], [13], [26]. Such models often use a Gaussian Process regressor to integrate the observations made by the different robots into a shared model. In the face of model uncertainty, Gaussian mixture-based coordination strategies have also been proposed [7], [17]. In this paper, we assume complete communication connectivity among the robots in this paper, as illustrated in Fig. 1. Exploration while maintaining such connectivity under limited communication ranges is studied in [6], using a graph theoretic technique. For conflict-free, multi-robot informative path planning, the authors in [18] have used a bipartite matching-based technique while adapting it to handle the spatio-temporal dynamics. However, none of these works study how exchanged data among the robots might withstand data integrity attacks.

Recently, a collective decision making technique for a multi-robot system using blockchains has been studied in [25] albeit not for information sampling. The authors have used Proof-of-Work (PoW) as their implemented consensus protocol. PoW is used by popular crypto-currencies such as Bitcoin and Ethereum [19]. However, it is known to be resource-intensive [4], [5]. A quantification metric to measure the effect on the carbon cost of such crypto-currencies is studied in [14]. Recently, a significantly less resource-intensive consensus protocol, namely Proof-of-Stake (PoS), has been introduced. Ethereum, the second most popular crypto-currency, has announced that its second generation (2.0) will shift from PoW to PoS for verification purposed and, consequently, will consume a fraction (0.0001) of the current energy[1]. While originating for transaction verification in crypto-currencies, blockchain-based security techniques are being employed in many other contexts e.g., vaccine distribution, degree credentials, food trust, supply chain management, among others[2]. A comprehensive survey on consensus protocols for blockchain can be found in [22].

III. Problem Setup

We have a set of n robots $R = r_1, r_2, \cdots r_n$. The robots are homogeneous, localized using a GPS, and move in a shared environment. The environment is discretized into a graph $G_p = \{V, E\}$, where the node set V represents the information collection locations and the connections among them are denoted by the edge set E. Each robot r_i has its unique sub-region for exploration, $\mathcal{V}_i$, and $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$. $\mathcal{V}_i$ can be calculated in a pre-processing stage by applying Voronoi partitioning [27] or K-medoids clustering [12]. W.l.o.g., we assume that $\cup_{i=1}^{n} \mathcal{V}_i = V$. The action set of the robots is denoted by A. For example, in a 8-connected grid G_p, A will hold the motor commands to move to all the eight neighbors. r_i is equipped with an on-board sensor using which it can sense and collect information (e.g., radiation detector). The robots' observations are modeled to be noisy. A robot r_i starts from a node $v_i^0 \in \mathcal{V}_i$. We assume that a robot r_i can communicate with $r_j, \forall r_j \in R \setminus r_i$ after collecting data at any node, i.e., the robots maintain a continuous connectivity throughout the exploration [1]. The algorithmic details of maintaining such a network is out of scope for this work; an example of techniques that can be used are the ones proposed in [6]. The robots are sensing an ambient phenomenon $\mathcal{Z}$ that varies with the location, with $\mathcal{Z}(v_i^0)$ being the (scalar real) value at node v_i^0.

A. Prediction via Gaussian Processes (GPs)

We use a Gaussian Process (GP) to model the uncertain environment. Let $\mathbf{X}$ denote a Gaussian random vector of length $|V|$ with prior mean vector μ and covariance matrix Σ, where μ and Σ represent the prediction in node set V and its corresponding uncertainty, respectively [21]. The volumetric measure of this uncertainty is calculated by an information theoretic metric, (differential) entropy, which is formally defined as $H(\mathbf{X}) = \frac{1}{2}\log|\Sigma| + \frac{|V|}{2}\log(2\pi e)$. Each robot starts with a common initial GP model, GP^0, and then takes measurement $\mathcal{Z}(v_i^0)$ at the start node $v_i^0 \in V$. We assume the measurements are subject to additive white Gaussian noise $\epsilon \in \mathcal{N}(0, \sigma)$. The updated local GP, GP_i, for robot r_i is then given by the posterior statistics:

$$\begin{aligned} \Sigma_i =& \Sigma - \Sigma\mathbf{C}\left(v_i^0\right)' \left(\mathbf{C}\left(v_i^0\right)\Sigma\mathbf{C}\left(v_i^0\right)' + \\ & \sigma_n^2\right)^{-1}\mathbf{C}\left(v_i^0\right)\Sigma \\ \mu_i =& \mu + \Sigma_i^0\mathbf{C}\left(v_i^0\right)' \left(\mathcal{Z}(v_i^0) - \mathbf{C}\left(v_i^0\right)\mu\right)/\epsilon \end{aligned} \tag{1}$$

[1] https://bit.ly/3zlq3aS
[2] www.ibm.com/blockchain

where $\mathbf{C}\left(v_i^0\right)$ denotes the length-$|V|$ row vector of all zeros except for a one in component v_i^0 and $\mathbf{C}\left(v_i^0\right)'$ is its matrix transpose. The reader is referred to [7] for more details.

It is a standard assumption in kernel-based parametrizations of GPs that the the correlation between two nodes are inversely proportional to the distances between them [7], [13], [21]. We exploit this property when computing entropy by approximating the computationally intensive matrix determinant $|\Sigma|$ by the product of the per-node variances (σ_v^2) along the diagonal of Σ. In turn, the associated entropy $H(\mathbf{X})$ decomposes additively across the nodes, each per-node term ($H(X_v)$) given by

$$H(X_v) = \frac{1}{2}\log\left(2\pi e\sigma_v^2\right). \tag{2}$$

The next section utilizes these per-node entropies to drive the robots to opportune locations for information collection.

IV. Algorithms

In the following we present our method that modifies the type of algorithms described in [6], [26] to provide resilience against the insertion of false data into the information model. In algorithms such as these, where the path of the robots is driven by the information model, fake data inserted in the model will send the robots to incorrect paths. For instance, an opponent can insert data to change the GP to high confidence about a given area, preventing the robots to explore it. The protection against fake data must be implemented at data collection time, because even if the fake information is discovered at the post-processing step, the damage had already been done in form of suboptimal paths.

The pseudo-code of our approach is presented in Algorithm 1. Before deployment, each robot is given training data $\mathcal{D}$ to initialize its local GP model GP_i and, consequently, the corresponding prior statistics are calculated. After the measurement is complete in the initial location and GP_i is updated according as described in Section III-A, each robot calculates the per-node entropies using Eq. 2 and updates the rewards. In a greedy and deterministic step, robot r_i will choose to move next to the node v_i^* which maximizes the entropy among its neighboring locations, i.e.,

$$v_i^* = \arg\max_{v\in neigh(v_i^0)} H(v|\mathcal{D}\cup\mathcal{Z}(v_i^0)),\ \text{s.t. } v\in\mathcal{V}_i \tag{3}$$

After a *sense*-and-*move* step, the robots share their observation with all the other robots. Because the robots are exploring their own unique sub-regions in the environment, the observations are non-overlapping. The robots add the newly received data points to their local GP models, evolving the associated statistics accordingly. This <*move-sense-communicate-estimate*> cycle continues until their allocated budget runs out. Such greedy strategies were proven to be effective approximations given the NP-Hardness of the studied problem [2], [13], [24].

When sharing their sensed information, the robots are vulnerable to cyber attacks such as data integrity attacks, denial of service and radio frequency jamming [9], [15]. In this

Algorithm 1: Secure Multi-robot Information Sampling

Input: $\mathcal{V}_i \leftarrow$ A region assigned to $r_i \in R$ to explore;
$v_i^0 \leftarrow$ Starting location of robot r_i s.t. $v_i^0 \in \mathcal{V}_i, \forall r_i \in R$;
$\mathcal{D} \leftarrow$ Initial training data provided to all $r_i \in R$;
$\mathcal{C}_i \leftarrow r_i$'s local blockchain;
/* r_i follows a <*move-sense-communicate-estimate*> cycle */
Each robot 1) begins with the same prior GP learned from $\mathcal{D}$, 2) updates GP_i with v_i^0, and 3) predicts per-node entropies;
$v_i^* \leftarrow$ Choose the next location to move to using Eq. 3;
while $B > 0$ **do**
  Move to the next node v_i^* and *Sense* information in v_i^*;
  Broadcast v_i^* and $\mathcal{Z}(v_i^*)$;
  $\{\tilde{V^*}, \tilde{\mathcal{Z}}\} \leftarrow$ receive similar information from $\forall r_j \in R \setminus r_i$;
  Secure. Decide to add $\{\tilde{V^*}, \tilde{\mathcal{Z}}\}$ to $\mathcal{C}_i$ or not based on either PoW (Algo. 2) or PoS (Algo. 3) technique;
  Estimate. update GP_i with the new data in $\mathcal{C}_i$ (Eq. 1) and update the entropies (Eq. 2);
  Select v_i^* based on the updated entropies;

paper, we focus on data integrity attacks with an attack model that assumes a malicious entity periodically broadcasting falsified measurements along the visited locations. In order to protect the shared data against such malicious attacks, permitting the uncompromised nodes to reject that data in further decision making, we employ a Blockchain-based data security technique. Blockchain is a tamper-resistant digital ledger that the robots maintain in a distributed fashion. In a blockchain the data is stored in discrete units, called blocks, that are linked (chained) to each other by having the hash of one block be part of the data of the next block. What a block contains depends upon the application. In our formulation, each robot maintains a local blockchain $\mathcal{C}_i$. Each block in $\mathcal{C}_i$ hold the following data: $< D, \tau, idx, \mathbf{N}, H_{last} >$, where D denotes the data collected in that round, τ indicates the timestamp, idx is the index of this particular block in $\mathcal{C}_i$, $\mathbf{N}$ is a special number called nonce, and H_{last} represents the previous block's hash. At the beginning, $\mathcal{C}_i$ is initialized with a genesis block (b_g) [20]—the first block—which is not hashed and does not contain any collected information data D. The main advantage of using Blockchain to prevent data integrity attacks is its chained architecture—if an attacker successfully modifies the data in the block b_{idx}, then its corresponding hash will also change and, consequently, it will not match the H_{last} hash stored in the block b_{idx+1}, and so on.

A. Consensus Protocol: Proof-of-Work

After a robot measures information $\mathcal{Z}(v_i^*)$ at location v_i^*, it puts $\mathcal{Z}(v_i^*)$ and v_i^* into a data structure D and creates a block for its local blockchain $\mathcal{C}_i$ along with the other relevant data as mentioned above. Initially, the robot sets the nonce $\mathbf{N}$ to 0 and finds the hash of the entire block, $H(b_{idx})$. One such hash function is SHA256. If the resulting hash does not match the appropriate *difficulty* $d(\cdot)$, i.e., have enough

Algorithm 2: Multi-robot Proof-of-Work (PoW) Consensus Protocol

Input: $\mathcal{C}_i \leftarrow r_i$'s local blockchain;
$\mathcal{C}_j^r \leftarrow$ received blockchain from r_j;
1 **if** $len(\mathcal{C}_j^r) > len(\mathcal{C}_i)$ AND CHECKCHAINVALIDITY() **then**
2 $\mathcal{C}_i \leftarrow \mathcal{C}_j^r$;
3 **Procedure** `checkChainValidity()`
4 $H(b_{idx}) \leftarrow$ hash of block b_{idx};
5 $H_{last}(b_{idx}) \leftarrow$ hash of the previous block b_{idx-1} stored in b_{idx};
6 **for** each block $b \in \mathcal{C}_j^r$ **do**
7 **if** ISVALIDBLOCK(b_{idx}) is false OR $H_{last}(b_{idx}) \neq H(b_{idx-1})$ **then**
8 return false
9 return true
10 **Procedure** `isValidBlock()`
11 $d(b_{idx}) \leftarrow$ difficulty of block;
12 $d_{min} \leftarrow$ minimum valid difficulty;
13 **if** $d(b_{idx}) \leq d_{min}$ **then**
14 return true
15 **else**
16 return false

Algorithm 3: Multi-robot Proof-of-Stake (PoS) Consensus Protocol

Input: $\mathcal{C}_i \leftarrow r_i$'s local blockchain;
$\mathcal{K} \leftarrow$ A set of n public keys for n robots;
$m_i \leftarrow \emptyset$;
1 $e \leftarrow e + 1$;
2 $\mathcal{C}_i^l \leftarrow$ Subset of $\mathcal{C}_i$ for committee selection;
3 $m_i \leftarrow$ Decide committee members from $\mathcal{C}_i^l$;
4 Broadcast m_i;
5 Receive $m_j,\ \forall r_j \in R \setminus r_i$;
6 Verify that received m_j's match m_i;
7 **if** $r_i \in m_i$ **then**
8 $p_i \leftarrow$ Decide the block proposer for this epoch;
9 Broadcast p_i;
10 Receive $p_j,\ \forall r_j \in R \setminus r_i$;
11 Verify that p_j's match p_i;
12 **else**
13 wait;
14 **if** r_i *is the block proposer* **then**
15 Create a new block c and append it to $\mathcal{C}_i$;
16 Broadcast $\mathcal{C}_i$;
17 **else**
18 Receive $\mathcal{C}$ from p;
19 **if** CHECKCHAINVALIDITY() *(Algo. 2)* **then**
20 Accept the received $\mathcal{C}$ and overwrite $\mathcal{C}_i$;
21 **else**
22 Reject it and keep $\mathcal{C}_i$;

leading zeros, the nonce is increased by one and a new hash is calculated. This continues until the resulting hash has the appropriate number (selected by the user) of leading zeros. This process is known as *mining*. The coordination and the corresponding block verification happens in a sequence. We generate a random order of the robots to emulate measurement and/or communication lag. Next, the robots, in the generated order, create a block with the newly observed data, mine it, and place it in its local blockchain. Thereafter the robot sends the block to its neighbors. The process repeats until each robot has had a turn, and the end result is that each robot is left with a blockchain that contains every robot's new data, plus the original data that existed before the robots began communicating (lines $1-3$ in Algorithm 2).

While finding the proper nonce is time-consuming, verification is quick and easy, with robots simply taking the given nonce and calculating the hash for themselves. Because our adversary is assumed to be unaware of the blockchain, its false data is detected simply via the efficient check that the received block shows no signs of proof-of-work i.e., the difficulty of the hash is lower than the minimum target difficulty d_{min} and so the block is rejected (lines $11-16$ in Algorithm 2).[3] The pseudocode for this consensus protocol is presented in Algorithm 2.

Difficulty. One of the biggest challenges one faces in implementing PoW is deciding what the difficulty should be. In a hexadecimal hash there are, naturally, 16 possible options per digit, with only one of them being 0. An implementation of PoW with a difficulty of 1 runs the possibility that an attacker who does not implement PoW still has a probability of $\frac{1}{16}$ of ending up with a hash that satisfies the difficulty condition, which is considerably high when security is concerned. Increasing the difficulty increases security, but at the cost of increasing the amount of time and energy robots spend attempting PoW and, in turn, reducing efficiency. Most crypto-currencies have a set time frame that they want mining to take (known as the *block time*) and automatically alter the difficulty level over time to match this. For example, Bitcoin has a block time of roughly 10 minutes, while Ethereum has a block time of around 13 seconds[4]. As one might expect, Bitcoin is known as being more secure than Ethereum, but also less efficient and more harmful to the environment due to it being so resource-intensive [4], [5], [22].

B. Consensus Protocol: Proof-of-Stake

Next, we discuss the second Blockchain-based consensus protocol that we have employed, namely Proof-of-Stake (PoS). The complexity of PoS is relatively lower than PoW and known to be more energy-efficient than PoW in large networked systems – this is particularly motivating for us due to the limited on-board power supply of the robots [23]. The pseudocode of the PoS protocol used in this paper is inspired by the Snow White PoS, a provably secure consensus protocol [3] and is shown in Algorithm 3.

[3]One could also consider an advanced adversary who does understand that there is a blockchain and, knowing also the difficulty level, performs the time-consuming proof-of-work step at the subverted node just as the non-subverted nodes must do. Preserving data integrity in this case would require additional security measures and likely a trusted hardware assumption for all nodes e.g., a Public-Key-Infrastructure (PKI) in which private key signatures cannot be neither falsified or spoofed—such an adversary is beyond the scope of this paper.

[4]https://econ.st/3zfzx7o

Similar to PoW, each robot starts with its local blockchain $\mathcal{C}_i$. The robots are given a common private seed $\mathcal{P}$. The consensus happens by means of selecting a committee after every round of information collection. These rounds are called epochs and denoted by e. Each block in $\mathcal{C}_i$ holds the following information $< D, \tau, idx, H_{last}, pk_i >$, where $D, \tau, idx,$ and H_{last} are defined as before, and $pk_i \in \mathcal{K}$ is the public key of the robot that added the particular block to $\mathcal{C}_i$. First, the robots select a committee to decide which robot will become the next block proposer, i.e., the only robot capable of adding new blocks to the blockchain. Ideally, this would be the robot that has the highest *'reputation'*, i.e., the one that contributed the most blocks to the robots' current blockchains. To decide on the committee, r_i first considers the last $2l$ blocks in $\mathcal{C}_i$, where $l = length(\mathcal{C}_i)/k$, where k is an integer. Let $\mathcal{C}_i^l$ denote this Blockchain-subset. The robots that contributed more blocks within $\mathcal{C}_i^l$, have a higher probability of being selected as part of the committee. For example, if there are three robots and they have contributed $15, 5,$ and 10 blocks respectively to $\mathcal{C}_i^l$, then their probabilities of being part of the committee in that epoch would be $0.5, 0.167,$ and 0.333 respectively. To ensure that a robot r_i has some chance to be part of a committee even if it has not contributed any block to $\mathcal{C}_i^l$ so far, we propose a technique based on Laplace smoothing: we consider that every robot has virtually contributed one block to the blockchain, and thus, in the previous example, the modified block contributions would be $16, 6,$ and 11. r_i selects the committee members this way while seeding the random number generator with $(\mathcal{P}+e)$ to have a commonality across the robots. Once the committee is decided, r_i broadcasts it to the network and receives similar committee selections from $\forall r_j \in R$. The sent and the received committee memberships are verified to ensure that no 'man-in-the-middle' adversary has tampered with this data.

If r_i is selected to be a part of the committee, it randomly selects $p_i \in m_i$ to be the block proposer. It is the only robot that can add new blocks to the current blockchains. Like earlier, the random selection is seeded with $(\mathcal{P} + e)$ to have a commonality among the robots. Next, r_i broadcasts the block proposer's identity. On the other hand, if r_i is not part of the committee, it waits during this period. Once the block proposer p is selected, it creates a new block c and appends it to $\mathcal{C}_i$. Note that the block proposer will add all the observed data since the last time it has been selected as a block proposer to the D field of the block. Thereafter, the block proposer robot broadcasts $\mathcal{C}_i$. The receiving robots $(R \backslash p)$ use the CHECKCHAINVALIDITY() function to check whether the received blockchain is valid or not. If the function returns true, $r_i \neq p$ overwrites $\mathcal{C}_i$ with the received Blockchain; otherwise rejects it and keeps its own intact.

V. EXPERIMENTS

A. Settings

We have implemented the proposed secure multi-robot information sampling approaches in simulation using MATLAB. We have tested the algorithms with up to 10 robots. The robots locations are drawn from a uniform random distribution representing an 8-connected 14×14 unit-length square grid environment. A robot only explores within its allocated region $\mathcal{V}_i$. The budget is set to 20 node visits. We have sampled our underlying ground truth information for 196 grid locations from a zero-mean Gaussian random vector, where the covariance matrix represents an exponential kernel function: specifically, for any pair of nodes v_s and v_t, the covariance between them is represented by $\beta^2 \exp(-||v_s - v_t||/\ell)$, where hyperparameters $\beta > 0$ is the local standard deviation and ℓ is the exponential rate of diminishing covariance between increasingly distant nodes. In our experiments, β and ℓ are set to 1 and 25 respectively. The additive white Gaussian noise $\epsilon \in \mathcal{N}(0, 0.25)$.

The adversarial influence is modeled as commanding the same subverted node to falsify its measurement only periodically, leaving that node's measurements and behavior unaltered otherwise. More specifically, all experiments with adversarial influence assume one specific node is subverted, falsifying its measurement only every four rounds. The falsification process itself is also simplistic; specifically, the adversary first chooses a magnitude uniformly from the range $[1, 10]$, then chooses its sign as positive or negative with equal probability. Observe that this falsification scheme is consistent in the mean with the untampered measurement process, but of course results in potential degradation of prediction due to inconsistency in the covariance. Moreover, a more damaging (but also, in the long run, more detectable) adversary is one who has subverted multiple nodes, attacks with shorter periods or falsifies using larger magnitudes; while interesting, this paper's brevity must leave experimental studies as a function of adversary to future work.

We have compared our proposed PoW and PoS-based algorithms against two benchmarks: 1) *No Attack*. in this baseline, no data integrity attack happens, and therefore, the robots share their true collected data; and 2) *Insecure*. in this baseline, data tampering happens similarly to our proposed secure methods, however, there is no security in place to save the robots from such data integrity attacks. Each test is run five times.

B. Results

First, we are interested in investigating the effect of the security attacks on the quality of the information model estimated by the robots. For this purpose, we have calculated the Mean Square Error (MSE) between the ground truth information model and the average predicted model by the robots. The results are presented in Fig. 2. The shaded regions indicate the standard deviations. For PoW, we have varied the difficulty level between $[1, 5]$, i.e., between one and five leading zeros in the hash. We see that with more robots, the average effect of the data integrity attacks on MSE, usually minimizes. As we only have one robot that sends tampered data to the other robots, with $n = 2$, that accounts for 50% robots being malicious, whereas with $n = 10$, it only accounts for 10%. For example, the final MSE with these two values of n are 0.78 and 0.19 respectively.

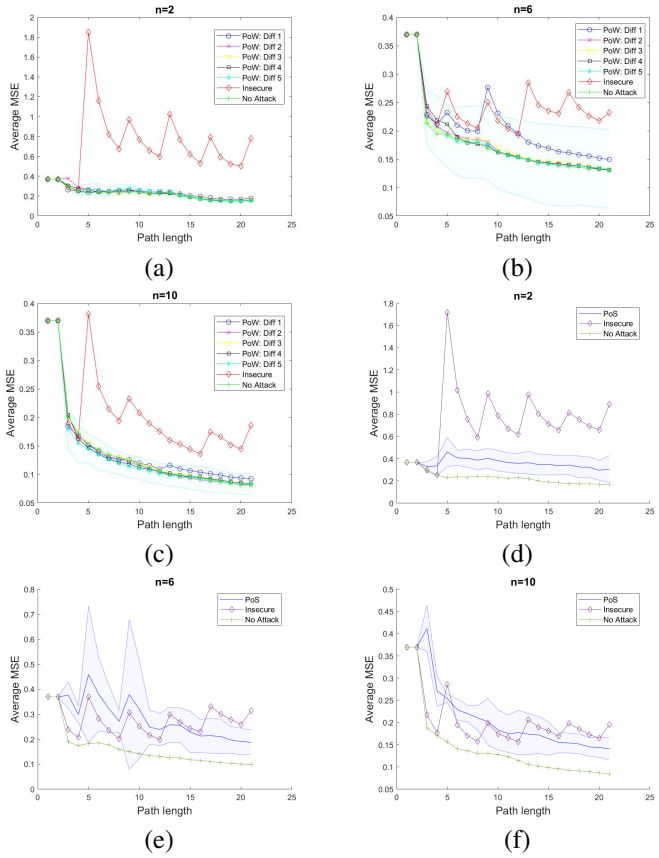

Fig. 2. MSE comparison (lower the better) between our proposed secure techniques and the implemented benchmark algorithms.

In any case, the MSE goes down as the robots make more observations [2], [13], [6]. However, if we have an insecure system and there is such an integrity attack, we see the MSE values jump to higher numbers any time there is an injection of tampered data. As the attack happens periodically, in every fourth iteration, therefore, the spikes in the plots are also periodic. We also observe that if the difficulty in PoW is low, e.g., 1, the attacker might get 'lucky' due to the high probability of the hash being found, and consequently, the security breaking down (Section IV-B). Since there are 16 possible hash values per digit and only one digit is an acceptable value for the prefix (0), the probability that the hash satisfies the difficulty 1 condition is $\frac{1}{16}$. For difficulty 2, this would probability is $(\frac{1}{16})^2 = \frac{1}{256}$; for difficulty 3, $(\frac{1}{16})^3 = \frac{1}{4096}$, and so on. In our experiments, there were a total of 375 attacks with difficulty level 1; 28 of those attacks succeeded in successfully tampering the data and letting other robots use that for estimation, which has a probability of $\frac{28}{375} \approx \frac{1}{13.4}$. This is evident from the periodic spikes with $n = 6$ in Fig. 2.(b). However, when we increase the level of the difficulty, e.g., 5, the MSE values coincide with the MSE values in the *no attack* scenario.

On the other hand, when we have used PoS to secure the shared information, we see the MSE values are higher than the *no attack* scenario as well as the PoW technique. The main reason for that is the shared blockchain in case of PoS is dominated by the robot(s) that has contributed more to the chain thus far. This potentially starves some robots, and consequently, the estimation of the sensor measurements in the unvisited locations might heavily depend on a few robots while the others might have very low contributions. For example, with $n = 10$, the final average MSE values using PoW (with difficulty level 5) and PoS are 0.08 and 0.14 respectively. This "rich gets richer" is a known property of PoS, which has a significant effect on the final information model estimation [3]. When investigated, we found that in our experiments, a robot has contributed 3.33 blocks to the shared blockchain on average with the maximum, minimum, and the standard deviation being $4.2, 2.8,$ and 0.47 respectively. We can also observe the effect of this on the MSE metric in regards to the *no attack* scenario. As there is no security protocol in place, the average MSE value with *no attack* is noticeably lower than that with the PoS mechanism. This is unlike PoW as all the robots equally contribute to the shared blockchain, and therefore, the average MSE with PoW, difficulty level 5, coincide with the *no attack* model.

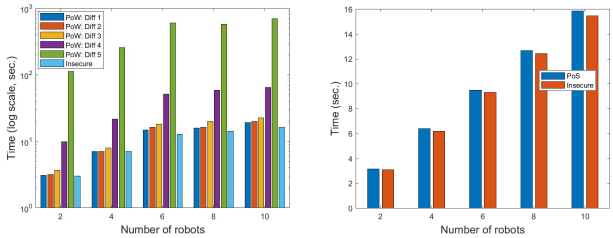

Fig. 3. Run time comparison between our proposed secure techniques and the implemented benchmark algorithms.

As discussed earlier, although the PoW and PoS-based consensus protocols allow us to achieve secure information collection, they also consume a significant amount of resources. To measure such effects, we investigate the run time metric next. We see that with a higher difficulty level in PoW, the run time significantly increases. For example, with difficulty 1, the run time is 19.18 sec. whereas with difficulty 5, the run time is 694.74 sec. with $n = 10$. This is due to the fact that each attempt at satisfying a difficulty 1 hash has a $\frac{1}{16}$ chance of succeeding, while satisfying a difficulty 5 hash has a $(\frac{1}{16})^5 = \frac{1}{1{,}048{,}576} \approx 9.54 \times 10^{-7}$ chance indicating more attempts must be made. On the other hand, in the *insecure* scenario, the corresponding run time is 15.02 sec. We can see that there is an increment of 46.25 times in the execution times to protect the information against data tampering attempts with difficulty 5. As found in the literature, PoS incurs a significantly lower time-cost than PoW. For the same setting above, PoS incurs a time cost of 15.88 sec., which is 43.75 times lower than PoW (difficulty 5), but 1.03 times higher than the corresponding *insecure* version.

VI. Conclusion and Future Work

In this paper we considered the task of multi-robot adaptive information sampling, in a setting where the robots

are collaborating to obtain a high quality global model by updating a Gaussian Process-based estimate of the investigated phenomena. We have shown that such systems are vulnerable to opponents inserting fake information into the model, because the path planning decisions of the robots depend on the current model. We proposed a secure multi-robot information sampling algorithm where the robots rely on a blockchain-based technique to accept or reject incoming observations. We proposed and implemented two variations of the blockchain technique based on the Proof-of-Work and Proof-of-Stake approaches respectively. Experiments show that both algorithms can significantly improve the quality of the information model in the presence of a persistent attacker. However, there is a tradeoff between the number of zeros requested in the hash for the PoW algorithm and thus implicitly the quality of the model and the computational cost. Overall, we found that the proposed algorithms while effective against the considered attack, add a significant computational overhead to the robot.

Our results suggest several future directions of research. Depending on the particular scenario it might be possible to determine in real time the optimal choice of the difficulty in the blockchain algorithm for a specific balance of the computational cost and model accuracy. A better understanding of the relationship between model and the path chosen by the robots could also allow for a partial offloading of the blockchain computations to the cloud, after the data sampling had been completed.

ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation (NSF) Cyber-Physical Systems Grant #1932300.

REFERENCES

[1] F. Amigoni, J. Banfi, and N. Basilico. Multirobot exploration of communication-restricted environments: a survey. *IEEE Intelligent Systems*, 32(6):48–57, 2017.

[2] N. Cao, K. H. Low, and J. M. Dolan. Multi-robot informative path planning for active sensing of environmental phenomena: a tale of two algorithms. In M. L. Gini, O. Shehory, T. Ito, and C. M. Jonker, editors, *International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013*, pages 7–14. IFAAMAS, 2013.

[3] P. Daian, R. Pass, and E. Shi. Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In *International Conference on Financial Cryptography and Data Security*, pages 23–41. Springer, 2019.

[4] A. De Vries. Bitcoin's growing energy problem. *Joule*, 2(5):801–805, 2018.

[5] L. Dittmar and A. Praktiknjo. Could bitcoin emissions push global warming above 2° c? *Nature Climate Change*, 9(9):656–657, 2019.

[6] A. Dutta, A. Ghosh, and O. P. Kreidl. Multi-robot informative path planning with continuous connectivity constraints. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3245–3251. IEEE, 2019.

[7] A. Dutta, O. P. Kreidl, and J. O'Kane. Opportunistic multi-robot environmental sampling via decentralized markov decision processes. In *Proc. International Symposium on Distributed Autonomous Robotic Systems*, 2021.

[8] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *Annual international cryptology conference*, pages 139–147. Springer, 1992.

[9] M. Gupta, M. Abdelsalam, S. Khorsandroo, and S. Mittal. Security and privacy in smart farming: Challenges and opportunities. *IEEE Access*, 8:34564–34584, 2020.

[10] G. A. Hollinger and S. Singh. Multirobot coordination with periodic connectivity: Theory and experiments. *IEEE Transactions on Robotics*, 28(4):967–973, 2012.

[11] M. Jakobsson and A. Juels. Proofs of work and bread pudding protocols. In *Secure information networks*, pages 258–272. Springer, 1999.

[12] L. Kaufmann. Clustering by means of medoids. In *Proc. Statistical Data Analysis Based on the L1 Norm Conference, Neuchatel, 1987*, pages 405–416, 1987.

[13] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

[14] M. J. Krause and T. Tolaymat. Quantification of energy and carbon costs for mining cryptocurrencies. *Nature Sustainability*, 1(11):711–718, 2018.

[15] C. L. Krishna and R. R. Murphy. A review on cybersecurity vulnerabilities for unmanned aerial vehicles. In *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pages 194–199. IEEE, 2017.

[16] W. Li, S. Andreina, J.-M. Bohli, and G. Karame. Securing proof-of-stake blockchain protocols. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, pages 297–315. Springer, 2017.

[17] W. Luo and K. Sycara. Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6359–6364. IEEE, 2018.

[18] K.-C. Ma, Z. Ma, L. Liu, and G. S. Sukhatme. Multi-robot informative and adaptive planning for persistent environmental monitoring. In *Distributed Autonomous Robotic Systems*, pages 285–298. Springer, 2018.

[19] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[20] O. Novo. Blockchain meets iot: An architecture for scalable access management in iot. *IEEE Internet of Things Journal*, 5(2):1184–1195, 2018.

[21] C. E. Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[22] M. Salimitari, M. Chatterjee, and Y. P. Fallah. A survey on consensus methods in blockchain for resource-constrained iot networks. *Internet of Things*, page 100212, 2020.

[23] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller. The energy consumption of blockchain technology: beyond myth. *Business & Information Systems Engineering*, 62(6):599–608, 2020.

[24] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[25] V. Strobel, E. Castelló Ferrer, and M. Dorigo. Blockchain technology secures robot swarms: a comparison of consensus protocols and their resilience to byzantine robots. *Frontiers in Robotics and AI*, 7:54, 2020.

[26] A. Viseras, T. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino. Decentralized multi-agent exploration with online-learning of gaussian processes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4222–4229. IEEE, 2016.

[27] G. Voronoi. New applications of continuous parameters ‘a to the theory of quadratic forms. second memory. research on primitive parallels. *Journal f ”u r die Reine und angewandte Mathematik (Crelles Journal)*, 1908(134):198–287, 1908.