Trust-Awareness to Secure Swarm Intelligence from Data Injection Attack

Bin Han*, Dennis Krummacker†, Qiuheng Zhou†, and Hans D. Schotten*†

*Division of Wireless Communications
and Radio Positioning (WICON)
Technische Universität Kaiserslautern
Kaiserslautern, Germany
{bin.han | schotten}@eit.uni-kl.de

†Research Department Intelligent Networks
German Research Center of Artificial Intelligence (DFKI)
Kaiserslautern, Germany
{dennis.krummacker | qiuheng.zhou |
hans_dieter.schotten}@dfki.de

***Abstract*—Enabled by the emerging industrial agent (IA) technology, swarm intelligence (SI) is envisaged to play an important role in future industrial Internet of Things (IIoT) that is shaped by Sixth Generation (6G) mobile communications and digital twin (DT). However, its fragility against data injection attack may halt it from practical deployment. In this paper we propose an efficient trust approach to address this security concern for SI.**

***Index Terms*—Trust, security, multi-agent, swarm intelligence**

I. Introduction

Over the past years, the technological trends of cyber-physical system (CPS) and industrial Internet of Things (IIoT) have raised a tide of Industry 4.0 (I4.0) [1], which has swept the world with its revolutionary use cases such as smart manufacturing, asset tracking, predictive maintenance, among others. After one decade, a variety of emerging technologies have arisen into the view and shown great potentials to push the current I4.0 one step further. The most significant ones among them are including the Sixth Generation (6G) mobile network [2], Machine Learning (ML) [3], artificial intelligence (AI) [4], and digital twin (DT) [5]. As key technical enablers, they are leading us into the hallway towards the next era of industry, where numerous intelligent devices and services shall be pervasively deployed and interconnected, with humans also seamlessly included and organically integrated [6].

Such a vision is advocating to deploy multi-agent systems (MAS) for a new paradigm of future smart industry: the industrial agents (IAs), which is believed to be powerful against an emerging set of industrial challenges [7]. It is worth remarking that IA provides a solid support to one promising approach of collaborative distributed intelligence: the emergent intelligence (EI), especially the swarm intelligence (SI) [8]. Compared to classical centralized ML solutions, SI exhibits several unique advantages such as privacy, robustness, and scalability [5], and therefore becomes a potential complimentary to the emerging technology of Federated Learning (FL) technology [9].

B. Han is the corresponding author.

Nevertheless, relying on information exchange among massive agents and the agents' reactions to perceived information, SI can be fragile against data injection attacks from insiders, i.e. from the involved agents. Unfortunately, to the best of our knowledge, there has been little research effort made on the trust and security measures to enhance the robustness of SI against such threat, except for a few highly use-case-specific studies [10]. To close this gap, in this paper we propose an efficient trust-aware approach to secure SI from data injection attack. We choose a simple use case of the classical particle swarm optimization (PSO) algorithm for our study, for the reason that it is generic enough so that our contribution can be straightforwardly extended, and adopted by more complex and specific SI techniques.

The remainder of this paper is organized as follows: We begin with Sec. II to present the problem under our investigation, and the conventional PSO algorithm to solve it. Then in Sec. III we set up various models of the data injection attack, and assess their threat to the studied SI use case. Afterwards, we introduce our main contributions, i.e. the trust score regression mechanism and the trust-aware PSO approach, in Sec. IV and Sec. V, respectively. Both proposals are validated by numerical simulations. To the end, we close this paper with our conclusion and some outlooks.

II. Problem Setup

As justified earlier, in this study we focus on one of the most typical and generic SI methods: the PSO algorithm. A use case of PSO-based multi-agent joint localization, as illustrated in Fig. 1, is investigated. Multiple mobilized agents, denoted as a *swarm* $\mathcal{I} = \{1, 2, \dots I\}$, are distributed across an open area with a target at unknown position. Each agent is equipped with a positioning module, a communication device, and a non-directive sensor to measure its distance to the target. Iteratively exchanging the position and distance information with each other, agents are supposed to jointly localize the target and move towards it. To simplify the discussion we consider a two-dimensional geometric model, as an extension to the three-dimensional case will be straightforward.

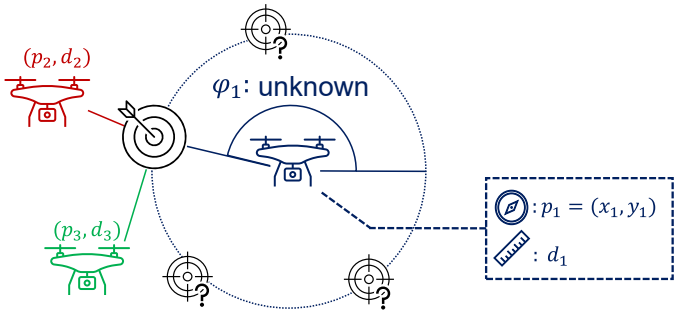


Fig. 1: The investigated use case

In each iteration t, every agent i obtains its accurate position p_i^t, but only an inaccurate distance d_i^t, which is the true distance D_i^t modulated by a log-normal random noise:

$$d_i^t = D_i^t \times 10^{\frac{n_i^t}{10}}, \tag{1}$$

where $n_i \sim \mathcal{N}(0, \sigma^2)$. Due to the noise, the accurate target location cannot be uniquely determined within one iteration. A conventional PSO solution to this problem is summarized by Alg. 1. For each (i, t), after measuring the data, the agent compares its current distance to target d_i^t with its own record of lowest historical distance d_i^{best}, and eventually update the latter if the former is even lower. In case of updating d_i^{best}, the associated position p_i^{best} is also updated by p_i^t. The swarm-best record among all agents, $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}\right]$, is also therewith checked and eventually updated. Afterwards, every i corrects its previous velocity v_i^{t-1} regarding its spatial offsets to both p_i^{best} and $p_{\mathcal{I}}^{\text{best}}$. Two constant coefficients c_1 and c_2 are specified to adjust the long-term impact weights of p_i^{best} and $p_{\mathcal{I}}^{\text{best}}$, respectively. Additionally, two random coefficients r_1 and r_2 are used to introduce short-term randomness for mitigating premature convergence. The agent speed is constrained by an upper bound $s_{\max}$.

Algorithm 1: Conventional PSO algorithm

1 Input: $\mathcal{I}, s_{\max}, c_1, c_2, T, \{p_i^0 : \forall i \in \mathcal{I}\}$
2 Initialize: $d_{\mathcal{I}}^{\text{best}} = +\infty,\ \forall i \in \mathcal{I} : v_i^0 = [0, 0],\ d_i^{\text{best}} = +\infty$
3 **for** $t = 1 : T$ **do**
4 **for** $i \in \mathcal{I}$ **do**
5 Update: d_i^t
6 **if** $d_i^t < d_i^{\text{best}}$ **then**
7 $\left[d_i^{\text{best}}, p_i^{\text{best}}\right] \leftarrow [d_i^t, p_i^t]$
8 **if** $d_i^{\text{best}} < d_{\mathcal{I}}^{\text{best}}$ **then**
9 $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}\right] \leftarrow \left[d_i^{\text{best}}, p_i^{\text{best}}\right]$
10 **end**
11 **end**
12 **end**
13 **for** $i \in \mathcal{I}$ **do**
14 Generate: $[r_1, r_2] \sim \mathcal{U}^2(0, 1)$
15 $v_i^t \leftarrow v_i^{t-1} + c_1 r_1 \left(p_i^{\text{best}} - p_i^t\right) + c_2 r_2 \left(p_{\mathcal{I}}^{\text{best}} - p_i^t\right)$
16 **if** $\|v_i^t\|_2 > s_{\max}$ **then**
17 $v_i^t \leftarrow s_{\max} v_i^t \,/\, \|v_i^t\|_2$
18 **end**
19 **end**
20 $p_i^{t+1} \leftarrow p_i^t + v_i^t$
21 **end**

While this solution has been demonstrated in [11] as effective, it obviously relies on the trustworthiness of information shared among agents, and can be fragile against data injection attacks. By manipulating a minority of the involved agents to maliciously report incorrect information, an attacker is capable of misdirecting other agents' decision so that the system performance is compromised. In addition, even benevolent agents without such intention may also behave similarly and cause the same effects in unawareness, for instance, when one is equipped with a distance sensor of poor quality or in malfunction. Regardless the origin or hostility, such data reported by untrustworthy agents are threatening the system in the same way. Therefore, in this paper we do not explicitly distinguish them from each other, but generally refer to them as data injection attacks. The target of our study is to design a trust approach, which allows the PSO algorithm to efficiently and correctly converge in presence of such attacks.

III. Data Injection Attacks: Threat Assessment

A. Attack Models

Instead of consistently honestly updating d_i^t to the system as supposed in Line 5 of Alg. 1, a manipulated agent i may commit a data injection attack by a certain chance:

$$\alpha_i^t \sim \mathcal{B}(1, r_{\text{atk}}), \tag{2}$$

where $\alpha_i^t \in \{\mathit{True}, \mathit{False}\}$ is the indicator of i committing attack in iteration t, and $r_{\text{atk}} \in (0, 1]$ the attack rate. In case of attacking, i replaces its raw measurement with a modified value $\tilde{d}_i^t$ regarding its attack model. Here we define four attack models: *i)* random distance, *ii)* biased distance, *iii)* extra distance error, and *iv)* zero distance:

$$d_i^t \leftarrow \tilde{d}_i^t = \begin{cases} d_{\text{rand},i}^t, & \text{random distance;} \\ \max\{0, d_i^t + \Delta d_i^t\}, & \text{biased distance;} \\ d_i^t \Big/ \left(10^{a_i^t/10}\right), & \text{extra distance error;} \\ 0, & \text{zero distance,} \end{cases} \tag{3}$$

where $d_{\text{rand},i} \sim \mathcal{U}_{[0, \Theta^{-1}]}$, $\Delta d_i \sim \mathcal{U}_{[-10\Theta, 0]}$, $a_i \sim \mathcal{N}(0, \Theta)$, and Θ is the attacking parameter.

B. Numerical Results

To assess the threat of data injection attack on the conventional PSO algorithm, we conducted numerical simulations. In every individual test, $I = 100$ agents were independently located regarding a uniform random distribution over a 60×60 m^2 rectangular region, while the target is fixed at the middle of the region (but unknown to any agent). The log-normal measurement error power is set to $\sigma^2 = 0.1$ and the maximal agent speed $s_{\max} = 5$ m/round. In every test, a random subset of agents $\mathcal{I}_{\text{atk}} \subset \mathcal{I}$ is selected as attackers to commit a data injection attack, where $\|\mathcal{I}_{\text{atk}}\|_0 \sim \mathcal{U}(3, 10)$. For the PSO algorithm we set $c_1 = c_2 = 0.5$. The same attack model is shared by all attacking agents and remains consistent throughout each individual test. To evaluate the converging performance of the PSO algorithm, we recorded after each iteration the average agent-to-target distance among

all trustworthy agents that do belong to $\mathcal{I}_{\mathrm{atk}}$. We carried out this test under different specifications of attack models and attack rates, while fixing the attack parameter $\Theta = 1$. We repeated it 1000 independent runs for every specification, with each run simulating $T = 50$ iterations of the PSO algorithm.

The average result of the Monte-Carlo tests are comparatively illustrated in Fig. 2. While the PSO solution is rapidly converging to an average agent-target distance around $5\,\mathrm{m}$ (as the curves "None" are showing), all kinds of data injection attacks can significantly degrade the performance at convergence, especially by reporting random or zero distance, which are capable of sufficiently failing the PSO solution. It deserves to be noted in particular that even under a low attack rate of 10%, an effective attack can be accomplished. This observation amplifies our concern to the threat of data injection attack, since the attackers can usually, if not always, effectively hide themselves from being identified by lowering the attack rate, as we will see in Sec. IV.

IV. Trust Score Regression

To protect a multi-agent system from data injection attacks committed by insiders, there are generally two key detection challenges: *i)* the detection of data anomaly; and *ii)* the detection of untrustworthy agents. To address the former one, it generally relies on system-specific knowledge, which can be either model-based or empirical. The latter one, in contrary, outlines a more generic problem of evaluating the trustworthiness of an individual agent upon its historical behavior. While the two approaches are non-exclusive to each other, in this work we are focusing on the second. To omit detailed discussions about data anomaly detection while avoiding a loss of generality, we conceive a data anomaly detector with certain error rates:

$$r_{\mathrm{md}} = P(\zeta_i^t \mid i \in \mathcal{I}_{\mathrm{atk}} \ \wedge \ \alpha_i^t), \tag{4}$$

$$r_{\mathrm{fa}} = P(\zeta_i^t \mid i \notin \mathcal{I}_{\mathrm{atk}} \ \vee \ \neg\alpha_i^t), \tag{5}$$

where $\zeta_i^t \in \{\mathit{True}, \mathit{False}\}$ indicates if d_i^t is an anomaly, r_{md} and r_{fa} are the misdetection and false alarm rates, respectively.

A. Update Models and Regression Principles

To realize a trust mechanism with memory on the historical behavior of agents, we define a trust score for every individual agent, notified as $\rho_i^t \in [0,1]$ for agent i in iteration t. Every agent is initialized with a certain trust ρ_i^0. Upon the classification of updated distance d_i^t by the data anomaly detector, ρ_i^t is updated in every iteration $t \in \mathbb{N}^+$. We propose three models of trust score update, namely *i)* binary, *ii)* linear, and *iii)* exponential, respectively. More specifically, if d_i^t is classified as normal, i is rewarded in its trust score:

$$\rho_i^t\big|_{\neg\zeta_i^t} = \begin{cases} 1, & \text{binary reward;} \\ \min\{\rho_i^t + 0.05, 1\}, & \text{linear reward;} \\ \min\{2\rho_i^t, 1\}, & \text{exponential reward.} \end{cases} \tag{6}$$

Similarly, with d_i^t classified as anomaly, i is punished:

$$\rho_i^t\big|_{\zeta_i^t} = \begin{cases} 0, & \text{binary penalty;} \\ \max\{\rho_i^t - 0.05, 0\}, & \text{linear penalty;} \\ \frac{\rho_i^t}{2}, & \text{exponential penalty.} \end{cases} \tag{7}$$

Furthermore, we establish five different trust score regression strategies by flexibly combining these rewarding and punishing models: *i)* binary/binary, *ii)* linear/linear, *iii)* exponential/exponential, *iv)* exponential/linear, and *v)* linear/exponential, where the former mode in each combination stands for the reward, and the later for penalty. Especially, note that the binary/binary strategy does nothing but simply taking the raw output of data anomaly detection as the result of attacker detection, which is used only as the baseline.

The trust score regression approach can be straightforwardly combined with a threshold-based attacker detector. After each iteration, every agent is labeled upon its instantaneous trust score w.r.t. a pre-defined threshold ρ_{th}:

$$\xi_i^t = \begin{cases} \mathit{True}, & \rho_i^t < \rho_{\mathrm{th}}; \\ \mathit{False}, & \text{otherwise,} \end{cases} \tag{8}$$

where ξ_i^t indicates if i is identified as attacker at t.

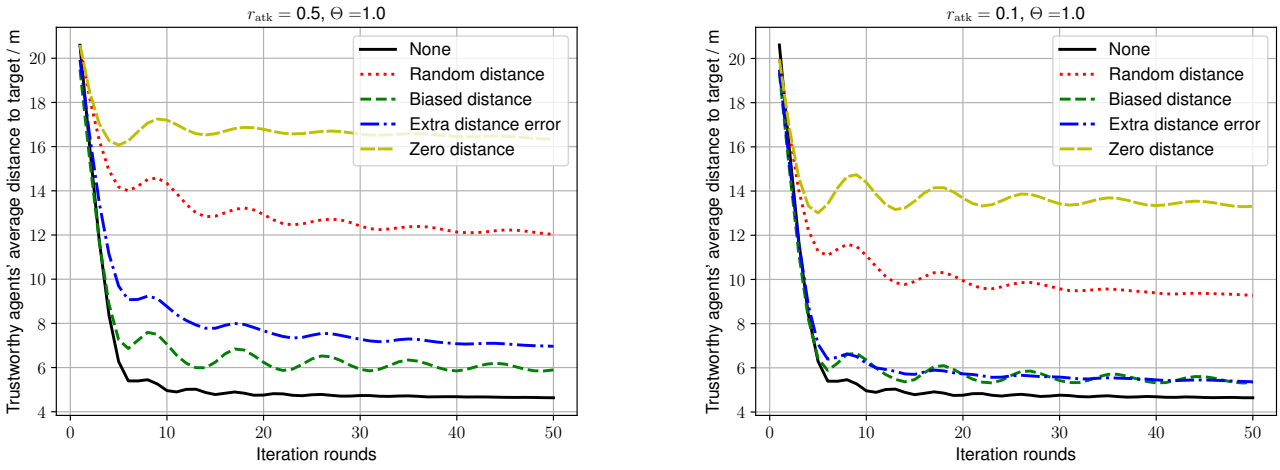
B. Numerical Results

To evaluate the performance of our proposed trust score regression strategies, we applied each of them to the system we considered in Sec. III. We executed the test with zero distance attack[1] at two different rates: 10% and 50%. For every individual specification, we repeated 1000 independent runs of Monte-Carlo test, each lasting $T = 50$ iterations of the PSO algorithm. We set the initial trust score to $\rho_i^0 = 0.5$ for all $i \in \mathcal{I}$ and the threshold $\rho_{\mathrm{th}} = 0.382$.

Therewith we have obtained under each specification the misdetection rate r_{md} and false alarm rate r_{fa} of the attacker detection. Remark that they shall be distinguished from those rates of the anomaly detection, i.e. p_{md} and p_{fa}. As we can observe from the results shown in Fig. 3, while the binary/binary baseline is exhibiting a consistent $r_{\mathrm{fa}} = p_{\mathrm{fa}}$ for an obvious reason, all four other regression strategies are capable of rapidly reducing the false alarm rate within 10 iterations. However, even at a high attack rate of 50%, only the linear/exponential strategy is efficient in improving r_{md} w.r.t. the binary/binary baseline, while the other three are showing only limited or even negative gains. When it comes to a low attack rate of 10%, none of the proposed strategies can outperform the binary/binary baseline, resulting in a high $r_{\mathrm{md}} > 0.85$ (or even up to 1).

In summary, a properly designed trust score regression can help us better exploit the entire history of data anomaly detection result, and therewith improve the accuracy of attacker detection. Nevertheless, even against a detector with reasonable regression strategy, by lowing the attack rate, attackers can still

[1]Indeed, since the values of p_{md} and p_{fa} are fixed regardless of the attack model, the attack model is irrelevant here and can be arbitrarily configured.



(a) (b)

Fig. 2: Different data injection attacks on conventional PSO, at attack rates of (a) 50% and (b) 10%, respectively.

manage to evade much of the detection while keeping threat to the system. Though a higher trust score threshold ρ_{th} will certainly help reduce r_{md}, it essentially leads to a raised r_{fa} as its price. Therefore, playing around with ρ_{th} is no promising solution to this problem, especially since the attack rate is unknown to the system. Instead, it is rational to seek for a more advanced trust-aware mechanism.

V. Trust-Aware Particle Swarm Optimization

A. Mechanism Design

To enhance the PSO algorithm with a trust-awareness, the procedures of data anomaly detection, trust score regression, and trust-based attacker detection must be sequentially executed in every iteration, right after updating d_i^t. Therewith, the agent-reported distances can be selectively exploited regarding the attacker detection results.

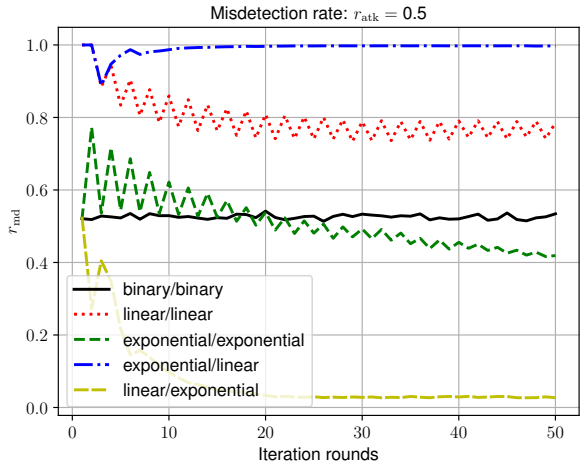
First, for all i that $\xi_i^t = \mathit{True}$, d_i^{best} shall be rejected from updating $d_{\mathcal{I}}^{\text{best}}$. Second, since it takes time for the trust score of every agent to regress, there is a risk that an attacker $i \in \mathcal{I}_{\text{atk}}$ successfully evades the trust-based attacker detection and injects a fake $\tilde{d}_i^t$ into the PSO process as $d_{\mathcal{I}}^{\text{best}}$ before eventually being detected. Therefore, when updating the swarm-best record $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}\right]$ with a certain entry $\left[d_i^{\text{best}}, p_i^{\text{best}}\right]$ (Alg. 1, Line 9), the index i of the associated agent must also be recorded as $i_{\mathcal{I}}^{\text{best}}$ to keep track of the data source's trustworthiness, and be examined every iteration. If the source agent of the current swarm-best record is identified as attacker, the current record must be invalidated.

Moreover, regardless the specific value of ρ_{th}, there is a fundamental and intrinsic drawback of the threshold-based attacker detector: it is incapable of dealing with ambiguous trust scores that are not significantly higher than the threshold. Data reported by an agent with such a trust score shall not be fully trusted, since there is a considerable risk that the agent is an attacker; nor shall it be simply rejected, since it still has a good chance to represent true and valuable information. Therefore, a solution beyond binary rejection is required to discriminate against data, based on the trust score of their sources.

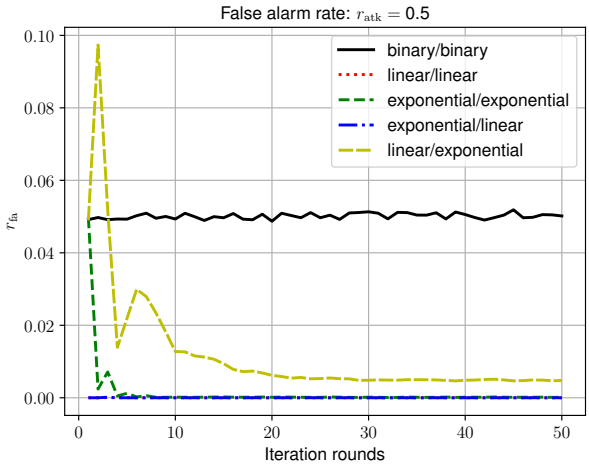
Thus, we propose a generic trust-aware PSO framework as in Algorithm 2, where the function `GenBest` in Line 20 updates the swarm-best record regarding agent trust scores.

Algorithm 2: Trust-aware PSO algorithm

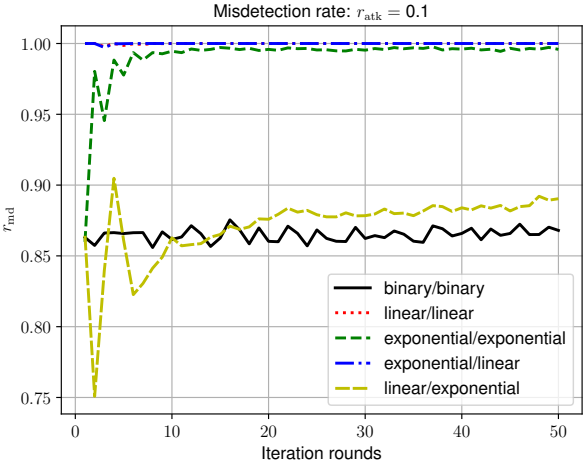
1 Input: $\mathcal{I},\ s_{\max},\ c_1,\ c_2,\ T,\ \left\{p_i^0 : \forall i \in \mathcal{I}\right\},\ \rho_{\text{th}}$
2 Initialize: $d_{\mathcal{I}}^{\text{best}} = +\infty,\ i_{\mathcal{I}}^{\text{best}} = 1,\ \forall i \in \mathcal{I} : v_i^0 = [0,0],\ \rho_i^0 = 0.5,\ d_i^{\text{best}} = +\infty$
3 **for** $t = 1 : T$ **do**
4 **for** $i \in \mathcal{I}$ **do**
5 Update: d_i^t
6 **if** $i \in \mathcal{I}_{\text{atk}} \ \wedge\ \alpha_i^t$ **then**
7 $d_i^t \leftarrow \tilde{d}_i^t(\texttt{AttackModel}, d_i^t)$
8 **end**
9 **if** $d_i^t < d_i^{\text{best}}$ **then**
10 $\left[d_i^{\text{best}}, p_i^{\text{best}}\right] \leftarrow \left[d_i^t, p_i^t\right]$
11 **end**
12 $\zeta_i^t \leftarrow \texttt{AnomalyDetection}(d_i^t)$
13 $\rho_i^t \leftarrow \texttt{TrustScoreRegression}(\zeta_i^t, \rho_i^{t-1})$
14 $\xi_i^t \leftarrow (\rho_i^t < \rho_{\text{th}})$
15 **end**
16 **if** $\xi_{i_{\mathcal{I}}^{\text{best}}}$ **then**
17 $d_{\mathcal{I}}^{\text{best}} \leftarrow +\infty$ `// Reject untrustworthy swarm-best`
18 **end**
19 $\mathcal{I}_{\text{tw}}^t \leftarrow \{i \in \mathcal{I} : \xi_i^t = \mathit{False}\}$ `// Trustworthy agents`
20 $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right] \leftarrow$ $\texttt{GenBest}\left(\left\{d_i^t, p_i^t, \rho_i^t, i : \forall i \in \mathcal{I}_{\text{tw}}^t\right\}, d_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right)$
21 **for** $i \in \mathcal{I}$ **do**
22 Generate: $[r_1, r_2] \sim \mathcal{U}^2(0,1)$
23 $v_i^t \leftarrow v_i^{t-1} + c_1 r_1 \left(p_i^{\text{best}} - p_i^t\right) + c_2 r_2 \left(p_{\mathcal{I}}^{\text{best}} - p_i^t\right)$
24 **if** $\left\|v_i^t\right\|_2 > s_{\max}$ **then**
25 $v_i^t \leftarrow s_{\max} v_i^t \Big/ \left\|v_i^t\right\|_2$
26 **end**
27 **end**
28 $p_i^{t+1} \leftarrow p_i^t + v_i^t$
29 **end**



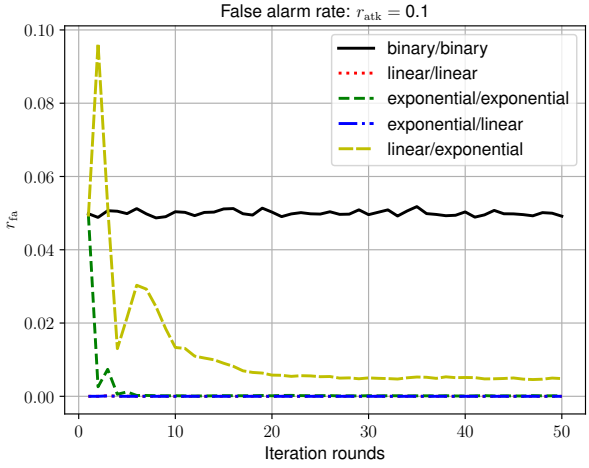
(a) r_{md} at 50% attack rate



(b) r_{fa} at 50% attack rate



(c) r_{md} at 10% attack rate



(d) r_{fa} at 10% attack rate

Fig. 3: Accuracy of trust-based attacker detection with different trust score regression strategies

B. Swarm-Best Record Update Policies

Regarding the specific use case under our study, we propose two policies of this update, namely *i)* hyperbolic scaling and *ii)* stochastic filtering, respectively.

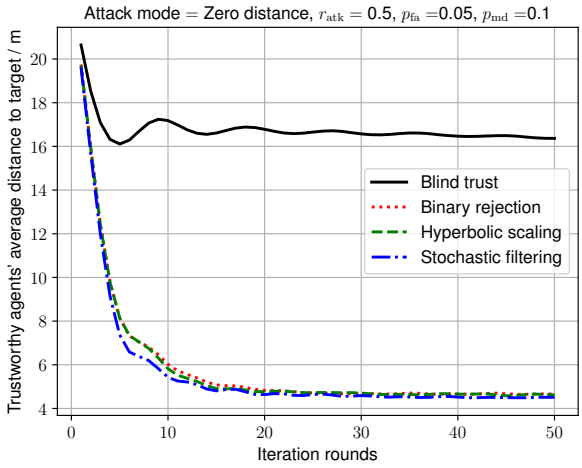
The policy of hyperbolic scaling takes an algebraic approach: it leverages the instantaneous trust score of agents to scale their reported distances, as described in Alg. 3. Thus, an agent with lower trust scores generally has lower chance to update the swarm-best record, even if it claims to be close to the destination. In contrary, the policy of stochastic filtering discriminates against agents regarding their trust score in a stochastic manner, as described in Alg. 4. In every iteration, a random array $\mathcal{I}_{\text{filtered}}$ of agent indices is generated from the set $\mathcal{I}_{\text{tw}}^t$ of agents that are currently classified as no attacker. The size of $\mathcal{I}_{\text{filtered}}^t$ equals that of $\mathcal{I}_{\text{tw}}^t$, while the probability mass function (PMF) that an element in $\mathcal{I}_{\text{filtered}}^t$ takes $i \in \mathcal{I}_{\text{tw}}^t$ is proportional to ρ_i^t. The update to swarm-best record is thereafter executed by agents in $\mathcal{I}_{\text{filtered}}^t$ instead of by those in $\mathcal{I}_{\text{tw}}^t$. Thus, an agent with lower trust score has less chance to be selected into $\mathcal{I}_{\text{filtered}}^t$. Moreover, as a baseline, the simple binary rejection without additional trust-aware agent discrimination can also be represented as a specific implementation of `GenBest`, as shown in Alg. 5.

Algorithm 3: `GenBest` with hyperbolic scaling

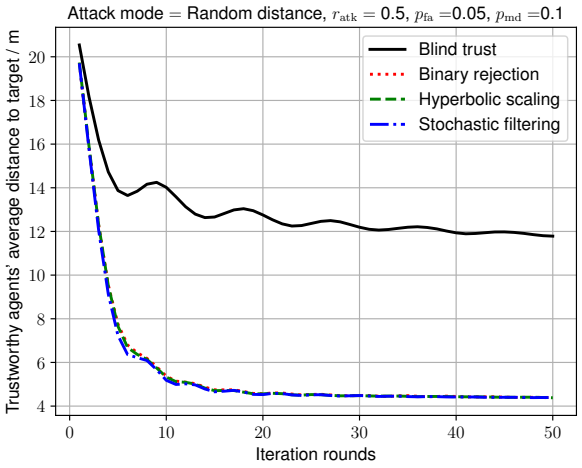
Input: $\left\{d_i^t, p_i^t, \rho_i^t, i : \forall i \in \mathcal{I}_{\text{tw}}^t\right\}, d_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}$
for $i \in \mathcal{I}_{\text{tw}}^t$ **do**
  if $\left(d_i^t / \rho_i^t\right) < \left(d_{\mathcal{I}}^{\text{best}} \Big/ \rho_{i_{\mathcal{I}}^{\text{best}}}^t\right)$ **then**
    $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right] \leftarrow \left[d_i^t, p_i^t, i\right]$
  end
end
return $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right]$

C. Numerical Results

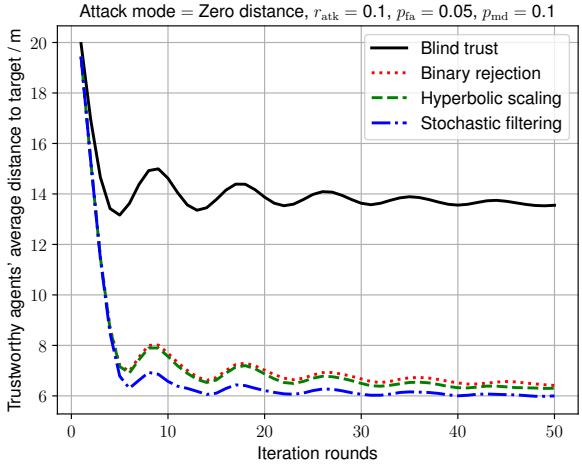
To evaluate the performance of our proposed trust-aware PSO, we tested it under the same system specifications as



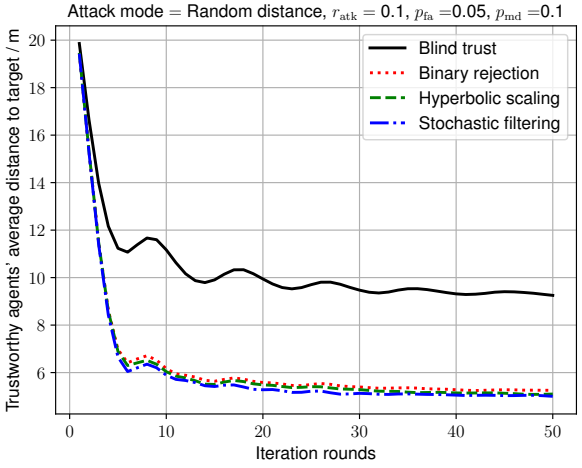
(a) Zero-distance attacks by 50%



(b) Random-distance attacks by 50%



(c) Zero-distance attacks by 10%



(d) Random-distance attacks by 10%

Fig. 4: Convergence performance of trust-aware PSO with different swarm-best update policies under data injection attacks

Algorithm 4: `GenBest` with stochastic filtering

Input: $\left\{d_i^t, p_i^t, \rho_i^t, i : \forall i \in \mathcal{I}_{\text{tw}}^t\right\}, d_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}$

Construct PMF: $P_K(k) = \begin{cases} \frac{\rho_k^t}{\sum_{i \in \mathcal{I}_{\text{tw}}^t} \rho_i}, & k \in \mathcal{I}_{\text{tw}}^t; \\ 0, & \text{otherwise} \end{cases}$

Generate: $\mathcal{I}_{\text{filtered}}^t \leftarrow \texttt{Random}\left(\text{PMF} = P_K, \text{size} = \left\|\mathcal{I}_{\text{tw}}^t\right\|_0\right)$

for $i \in \mathcal{I}_{\text{filtered}}^t$ **do**

  if $d_i^t < d_{\mathcal{I}}^{\text{best}}$ **then**

    $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right] \leftarrow \left[d_i^t, p_i^t, i\right]$

  end

end

return $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right]$

Algorithm 5: `GenBest` with binary rejection

Input: $\left\{d_i^t, p_i^t, \rho_i^t, i : \forall i \in \mathcal{I}_{\text{tw}}^t\right\}, d_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}$

for $i \in \mathcal{I}_{\text{tw}}^t$ **do**

  if $d_i^t < d_{\mathcal{I}}^{\text{best}}$ **then**

    $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right] \leftarrow \left[d_i^t, p_i^t, i\right]$

  end

end

return $\left[d_{\mathcal{I}}^{\text{best}}, p_{\mathcal{I}}^{\text{best}}, i_{\mathcal{I}}^{\text{best}}\right]$

we did with the conventional PSO in Sec. III. We considered the same anomaly detector as in Sec. IV, an attacker detector with the regression strategy of linear reward and exponential penalty, and initial trust score of 0.5 for all agents. We considered the attack models of random distance and zero distance, and the attack rates of 10% and 50%. We tested the trust-aware PSO with both proposed policies of hyperbolic scaling and stochastic filtering, as well as the simple binary rejection. The conventional PSO with blind trust to all agents is compared as a baseline. For every individual specification, we repeated 1000 independent runs of Monte-Carlo test, each lasting $T = 50$ iterations.

As the results in Fig. 4 are showing, compared to the conventional PSO, our proposed trust-aware PSO algorithm

significantly improves the system robustness against data-injection attacks in general. Specifically, at a high attack rate where the trust score regression is more effective in detecting attackers, all swarm-best update policies, including the simple binary rejection, are performing similarly well and sufficiently blocking fake data injections. At a low attack rate where p_{md} is high, the policy of stochastic filtering outperforms hyperbolic scaling, while the latter is performing only mildly better against the benchmark of binary rejection. We comprehend this performance gap between the policies as a result of their different exploitation of the trust information: by separately exploiting ρ_i^t and d_i^t in the processes of selective rejection and distance comparison, respectively, stochastic filtering can make better utility of the information than hyperbolic scaling, which mixes the two together and therewith lose the ability to distinguish a low distance with low trust score from a high distance with high trust score.

VI. Conclusion and Outlooks

So far, we have discussed the threat of data injection attack in SI systems, and demonstrated our proposal to address it. According to our experiment, data injection attacks, even with a low attack rate, can significantly disrupt SI with blind trust among agents. As a solution, our proposed trust-aware approach is proven efficient to counter such attacks.

As for the next step, it is interesting to design and implement an efficient solution of data anomaly detection for the studied problem, and evaluate our proposed approach with the specific detector. Furthermore, analytical efforts are also required in addition to enable parameter optimization of our proposed methods.

Acknowledgment

This work is supported partly by the European Commission through the H2020 project Hexa-X (GA no. 101015956), and partly by the German Federal Ministry of Education and Research (BMBF) through the project Open6GHub (GA no. 16KISK003K, 16KISK004).

References

[1] E. Sisinni, A. Saifullah, S. Han *et al.*, "Industrial Internet of Things: Challenges, opportunities, and directions," *IEEE Trans. Industr. Inform.*, vol. 14, no. 11, pp. 4724–4734, July 2018.

[2] W. Jiang, B. Han, M. A. Habibi *et al.*, "The road towards 6g: A comprehensive survey," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 334–366, February 2021.

[3] J. P. Usuga Cadavid, S. Lamouri, B. Grabot *et al.*, "Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0," *J. Intell. Manuf.*, vol. 31, no. 6, pp. 1531–1558, January 2020.

[4] R. S. Peres, X. Jia, J. Lee *et al.*, "Industrial artificial intelligence in Industry 4.0 - Systematic review, challenges and outlook," *IEEE Access*, vol. 8, pp. 220 121–220 139, December 2020.

[5] B. Han, B. Richerzhagen, H. Schotten *et al.*, "Impact of AI and digital twins on IIoT," in *Intelligent Edge-Embedded Technologies for Digitising Industry*, M. Coppola and O. Vermesan, Eds. Denmark: River Publishing, 2022.

[6] M. Maier, "6G as if people mattered: From Industry 4.0 toward Society 5.0," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1–10.

[7] S. Karnouskos, P. Leitao, L. Ribeiro *et al.*, "Industrial agents as a key enabler for realizing industrial cyber-physical systems: Multiagent systems entering Industry 4.0," *IEEE Ind. Electron. Mag.*, vol. 14, no. 3, pp. 18–32, September 2020.

[8] B. M. Khorsandi, M. Hoffmann, M. Uusitalo *et al.*, "Hexa-X Deliverable D1.3 - Targets and requirements for 6G - initial E2E architecture," April 2022, Available: https://hexa-x.eu/wp-content/uploads/2022/05/Hexa-X_D1.3.pdf.

[9] B. Han, B. Richerzhagen, L. Scheuvens *et al.*, "Hexa-X Deliverable D7.2 - Special-purpose functionalities: Intermediate solutions," April 2022, Available: https://hexa-x.eu/wp-content/uploads/2022/05/Hexa-X_D7.2_v1.0.pdf.

[10] S. Fu, Z. Jiang, S. Zhang *et al.*, "Data-injection-proof predictive vehicle platooning: Performance analysis with cellular-V2X sidelink communications," *IEEE Internet Things J.*, 10 2021.

[11] S. Yuan, B. Han, D. Krummacker *et al.*, "Massive twinning to enhance emergent intelligence," *arXiv preprint arXiv:2204.09316*, May 2022.