On the vulnerabilities of Voronoi-based approaches to mobile sensor deployment

N. Bartolini, S. Ciavarella, S. Silvestri, and T. La Porta

Abstract—Mobile sensor networks are the most promising solution to cover an Area of Interest (AoI) in safety critical scenarios. Mobile devices can coordinate with each other according to a distributed deployment algorithm, without resorting to human supervision for device positioning and network configuration. In this paper, we focus on the vulnerabilities of the deployment algorithms based on Voronoi diagrams to coordinate mobile sensors and guide their movements. We give a geometric characterization of possible attack configurations, proving that a simple attack consisting of a barrier of few compromised sensors can severely reduce network coverage. On the basis of the above characterization, we propose two new secure deployment algorithms, named SecureVor and SSD (Secure Swap Deployment). These algorithms allow a sensor to detect compromised nodes by analyzing their movements, under different and complementary operative settings. We show that the proposed algorithms are effective in defeating a barrier attack, and both have guaranteed termination. We perform extensive simulations to study the performance of the two algorithms and compare them with the original approach. Results show that SecureVor and SSD have better robustness and flexibility, excellent coverage capabilities and deployment time, even in the presence of an attack.

Index Terms—Mobile sensors, self-deployment, Voronoi approach.

1 Introduction

Mobile wireless sensors are the most suitable technology for monitoring inaccessible or hostile environments, where manual positioning of static sensors is not feasible [23]. These devices can autonomously deploy over an Area of Interest (AoI). Coordination among the sensors is obtained by means of a deployment algorithm that determines the device movement and positioning rules.

Besides the security problems typical of ad hoc networks, such as communication issues [9], [11], [25], false position claims [18], [8], Sybil [22] and node replication [29] attacks, mobile sensor networks suffer from other vulnerabilities. Mobile sensors usually lack tamper-proof hardware, thus an adversary may capture several nodes, extract their cryptographic material and reprogram them according to its malicious goal. The reprogrammed sensors, hereafter called *malicious* sensors, may perform several attacks to damage the network, exploiting the specific vulnerabilities of the *deployment algorithm* in use.

Previous solutions for deploying mobile sensors fall in to one of three major families: approaches based on virtual force models [15], [19], [30], [14], [17], on the formation of patterns [3], [27], or on computational geometry techniques [26], [20], [4]. Only recently, the vulnerabilities of the virtual force approach for sensor deployment have been considered [2]. This work introduced a simple attack tailored for mobile sensor deployment algorithms, called the *Opportunistic Movement* (OM) attack. Using a small set of malicious sensors, the attacker can influence the deployment of legitimate sensors by exploiting the coordination mechanism of the self-deployment approach. Malicious nodes may coordinate with each other to reduce the area in which the legitimate sensors are deployed, thus creating a non monitored zone.

While the work in [2] shows the detrimental effects of the OM attack against the virtual force approach, in [5] we provide experimental evidence of similar vulnerabilities in computational geometry approaches, and in particular in the Voronoi approach to mobile sensor deployment [26], [4].

In the present paper we significantly extend our previous results on the vulnerabilities of Voronoi based approaches and we provide, for the first time in the literature, an analytical study of the vulnerabilities of such an approach. In particular, we give a novel geometric characterization and a formal proof of the efficacy of the OM attack [2] against this deployment approach, showing that the attack can seriously compromise coverage.

We show that *during* the deployment of the network the OM attack is more effective against Voronoi based solutions than against the virtual force approach. In particular, with Voronoi based solutions, the efficacy of the attack depends only on the perimeter of the area that the attacker wants to keep uncovered, and there is no gain in increasing the number of legitimate sensors deployed.

By contrast, we show that *after* the network is deployed, Voronoi solutions are more robust than those based on virtual forces. In fact, when the network is deployed according to the Voronoi algorithm, once the area is completely covered, the OM attack no longer has any impact.

On the basis of the geometric characterization described above, we propose two new algorithms, called SecureVor

- *N. Bartolini and S. Ciavarella are with the Department of Computer Science, Sapienza University of Rome, Italy. E-mail: {bartolini,ciavarella}@di.uniroma1.it*
- *S. Silvestri is with the Department of Computer Science Missouri University of Science and Technology, USA. E-mail: silvestris@mst.edu*
- *T. La Porta is with the Department of Computer Science and Engineering, Pennsylvania State University, USA. E-mail: tlp@cse.psu.edu*

This work is partially supported by NATO - North Atlantic Treaty Organization, under the SPS grant G4936 "Hybrid Sensor Network for Emergency Critical Scenarios".

and Secure Swap Deployment (SSD), which are designed to counteract the OM attack. SecureVor works under the assumption that the transmission radius is at least four times larger than the sensing radius. Under this operative setting, which is common to most outdoor application scenarios, a sensor can determine the legitimacy of its neighbors movements and communications. SSD is designed to work in the same operative setting as the original VOR algorithm, i.e., $R_{\mathtt{tx}} > 2R_{\mathtt{s}}$, so that it is complementary to SecureVor. SSD exploits sensor positions swaps to verify the neighbors behavior.

We show that both our algorithms can defeat the OM attack in their respective operative settings, and we formally prove that both terminate in a finite time.

We perform extensive simulations to study the performance of SecureVor and SSD, in comparison with the original solution. The results show that both algorithms are able to successfully neutralize the OM attack and achieve coverage of the AoI at the expense of a small overhead in terms of energy consumption and deployment time. SecureVor is more effective when the transmission radius is sufficiently large with respect to the sensing radius, while SSD is preferable when such an assumption does not hold.

The original contributions of this paper are:

- For the first time, we point out and formally prove the vulnerabilities of Voronoi-based deployment algorithms, giving a geometric characterization of possible attack configurations.
- We propose two new secure deployment algorithms called SecureVor and SSD, which successfully counteract the OM attack, in different and complementary operative settings.
- We show that both algorithms have a guaranteed termination, show through simulation that both defeat the OM attack, and prove that in SecureVor all malicious nodes encountered are detected.
- Through simulations, we highlight the efficacy of our algorithms in providing full coverage, even in the presence of an OM attack, under a wide range of operative conditions, at the expense of a moderate increase in energy consumption and deployment time.

2 Background on the Voronoi approach

The Voronoi approach (VOR) to mobile sensor deployment has been introduced in [26]. It makes use of Voronoi diagrams to guide sensor movements within the AoI. According to [26], sensors communicate within a distance $R_{\mathtt{tx}}$ (*communication radius*), they sense over a circular area of radius $R_{\mathtt{s}}$ (*sensing radius*), with $R_{\mathtt{tx}} > 2R_{\mathtt{s}}$. Nodes can move in any direction inside the AoI, are endowed with low cost GPS, and are loosely synchronized.

VOR is executed in a distributed manner at each node and is round based. At each round t any sensor s broadcasts its position coordinates, and determines its set of neighbors $N_{\mathtt{tx}}^{(\mathtt{t})}(s)$, i.e. the sensors located within its communication radius. It then calculates its Voronoi polygon $V^{(\mathtt{t})}(s)$. Sensor s determines its next destination according to one of two movement criteria: the *Farthest Vertex* (FV) and the *MiniMax* (MM) [26].

According to FV a sensor s moves along the segment connecting its position and the farthest vertex of its polygon. Its destination is a point on this segment at distance $R_{\mathtt{s}}$ from the farthest vertex.

According to MM, the destination of s is the point that minimizes the maximum distance from the vertices of $V^{(\mathtt{t})}(s)$, which is the center of the minimum circle enclosing its polygon.

Regardless of the adopted movement criterion, a sensor s moves to its destination only if its movement provides a better coverage of $V^{(\mathtt{t})}(s)$, otherwise it remains still.

Furthermore, according to [26], s can traverse a maximum distance per round $d_{\mathtt{max}} = R_{\mathtt{tx}}/2 - R_{\mathtt{s}}$, to take into account possible inaccuracies in the distributed construction of Voronoi polygons, which may be due to the limited transmission radius.

3 The opportunistic movement attack

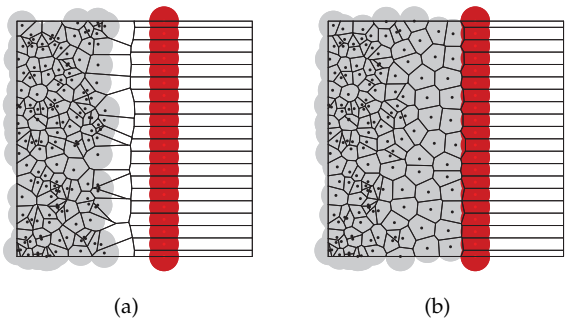
The original work [26] does not address the security vulnerabilities of the VOR approach. Since sensors lack tamper-proof hardware, an adversary may capture some nodes, and extract their cryptographic related information and reprogram them. Such malicious sensors may not be recognized by legitimate sensors as they are able to send valid messages containing a valid ID, and make use of legitimate cryptographic information. The attacker can thus exploit these corrupted nodes to perform malicious attacks to prevent a successful network deployment. For instance, the attacker can be interested in creating a non monitored area around a zone of interest, or isolating a part of the network. To pursue its goal, the attacker utilizes a set of malicious nodes that are able to collude with each other by performing coordinated movements and communications in order to influence the movements of the legitimate sensors.

The OM attack introduced in [2] aims at reducing the network coverage. To this purpose, malicious sensors initially form an *attack configuration* over the AoI. From such a configuration, malicious nodes start the attack by moving according to the adversary strategy, but communicating according to the communication protocol provided by the deployment algorithm.

The OM attack is a general attack which can be performed in different manners, depending on the movement strategy of malicious sensors. A particularly effective strategy is the Barrier Opportunistic Movement (BOM), in which malicious sensors form a linear barrier over the AoI [2], [5].

As provided by the OM attack, malicious sensors periodically communicate their positions at the beginning of each round in a legitimate way. By contrast, they move according to the attacker strategy. In particular, the malicious sensors forming the barrier may move towards legitimate sensors or remain still, in order to prevent legitimate sensors from spreading over uncovered areas.

In Figure 1, we show an example of a BOM attack. The red circular areas represent the sensing disks of the malicious nodes performing the BOM attack. The grey circles are the sensing ranges of the legitimate sensors that are spreading over the AoI according to VOR. The two figures 1(a) and (b) represent the initial and the final deployment, respectively. A barrier of malicious sensors is



(a) (b)

Fig. 1. BOM Attack. Initial deployment (a) and final deployment under VOR (b)

initially deployed over the AoI as in Figure 1(a), limiting the movements of legitimate sensors that will be able to spread only in the area limited by the barrier, as in Figure 1(b). The malicious nodes remain still, forming a barrier that prevents further movements of the legitimate sensors. In fact, the legitimate sensors that come in proximity with the barrier nodes stop moving. They do not move towards and across the barrier because, from the information received by malicious nodes, they derive that there is no way and no necessity to improve their local coverage.

In this paper, we give a geometric characterization of the vulnerabilities of the Voronoi approach to mobile sensor deployment. Moreover, we exploit the information derived from the geometric characterization to design two novel algorithms, SecureVor and SSD, which are able to neutralize the attack. In particular, SecureVor is designed for an operative setting in which the communication radius is at least four times greater than the sensing radius, i.e. $R_{\mathrm{tx}} > 4R_{\mathrm{s}}$. SSD, instead, is designed to work in the same operative settings as the VOR approach, i.e. $R_{\mathrm{tx}} > 2R_{\mathrm{s}}$.

Similarly to [2], [5], in order to highlight the strength of the BOM attack, in this paper we do not consider other attacks which may be performed in conjunction with BOM. Our goal is to show how the BOM attack, alone, can produce detrimental effects in terms of coverage to VOR based solutions.

3.1 Efficacy of the BOM attack against VOR

In this section we formally analyze the vulnerabilities of VOR against the BOM attack. We refer the reader to [10] for a brief survey of the properties of Voronoi tessellations.

We consider the diagram of Figure 2, where a Cartesian reference models the AoI. Malicious sensors are evenly deployed along the axis $x = 0$, with step size d, occupying the positions $(0, d/2 + k \cdot d)$, with $k \in \mathbb{N}$. We hereby call such a configuration a *d-spaced barrier* of malicious sensors.

We define $\Delta(R_{\mathrm{s}}, d) \triangleq \sqrt{R_{\mathrm{s}}^2 - d^2/4}$, also referred to as Δ. Let w be the width of the overlapping region between two adjacent malicious sensors. In Figure 2, $w = d(C, Q) = 2\Delta$, where $d(\cdot, \cdot)$ is the Euclidean distance between two points. Notice that such a width is larger than R_{s} if $d \leq \sqrt{3} \cdot R_{\mathrm{s}}$.

We use the following notation. We denote with $\mathcal{L}(\ell)$ and $\mathcal{R}(\ell)$ the half-planes at the left and right side of the line ℓ, respectively, where ℓ is a generic line of equation $x = x_\ell$. For brevity, we will use the same notation for a point

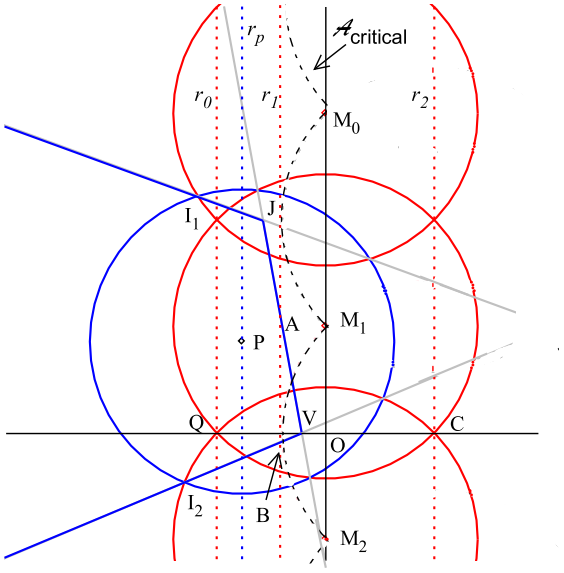


Fig. 2. A legitimate sensor in P approaches a d-spaced barrier (sensors deployed in $(0, d/2 + k \cdot d)$, with $k \in \mathbb{N}$). As P is on the left of line r_1, it does not cross the barrier, as long as $d \leq \sqrt{3}R_{\mathrm{s}}$.

$P = (x_p, y_p)$, denoting with $\mathcal{L}(P)$ and $\mathcal{R}(P)$, the half-planes $\mathcal{L}(x = x_p)$ and $\mathcal{R}(x = x_p)$, respectively.

Let us consider the lines r_0, r_1 and r_2, with equations $x = -\Delta$, $x = -R_{\mathrm{s}} + \Delta$, and $x = \Delta$, respectively. Notice that if $d \leq \sqrt{3} \cdot R_{\mathrm{s}}$, the line r_1 falls between the lines r_0 and r_2. We will prove that the line r_1 acts as a *frontline* of the barrier, precluding legitimate sensors from traversing it, independently of the moving criterion adopted by the Voronoi algorithm VOR.

Given a sensor s positioned in P, we denote with $V(s)$ its Voronoi polygon, and with $\mathcal{C}(s)$ its sensing circle. The following Lemma 1 recalls a general property of Voronoi polygons that is necessary for the following discussion. It is a specific case of a more general theorem given in [4] (Theorem 3.1) and states that if a point of $V(s)$ is covered by any sensor, then it is also covered by s.

Lemma 1 (Theorem 3.1 of [4]). *Let us consider N sensors s_i, $i = 1, \ldots, N$, with positions $P_i = (x_i, y_i)$, sensing circles $\mathcal{C}(s_i)$ and sensing radius R_s. Let $V(s_i)$ be the Voronoi polygon of s_i. For all k and $j = 1, 2, \ldots, N$, $V(s_k) \cap \mathcal{C}(s_j) \subseteq \mathcal{C}(s_k)$.*

Let us denote with $\mathcal{A}_{critical}$ the locus of points determined by this equation:

$$\mathcal{A}_{critical} = \bigcup_{k \in \mathbb{N}} \{(x, y) | (x - \Delta)^2 + (y - k \cdot d)^2 = R_s^2, x \leq 0\}. \quad (1)$$

The shape of this locus is a periodic sequence of circular segments, along the y-axis, as depicted in Figure 2.

Extending the previous notation, we denote with $\mathcal{L}(\mathcal{A}_{critical})$ and with $\mathcal{R}(\mathcal{A}_{critical})$ the regions on the left and right side of $\mathcal{A}_{critical}$, respectively.

Lemma 2 (Frontline). *Let us consider a legitimate sensor s, positioned in $P = (x_p, y_p)$, with $P \in \mathcal{L}(\mathcal{A}_{critical})$, with a d-spaced barrier of malicious sensors, with $d \leq \sqrt{3} \cdot R_s$. It holds that $V(s) \cap \mathcal{R}(r_0) \subseteq \mathcal{C}(s)$.*

In other words, the portion of the Voronoi polygon of s located on the right side of the line r_0 is completely covered, and is covered by s itself.

Proof. Consider the diagram of Figure 2. By contradiction, let us consider a point $Z \in V(s) \cap \mathcal{R}(r_0)$ and assume that Z is not in $\mathcal{C}(s)$. As a consequence of Lemma 1, Z is not covered by any sensor. Since the region between the lines r_0 and r_2 is covered by [at least] the barrier sensors, Z must be in $\mathcal{R}(r_2)$. Therefore, as Voronoi polygons are convex, $V(s)$ must have an uncovered vertex V_z in $\mathcal{R}(r_2)$.

Let us now remove from the diagram every sensor but s itself and the two closest barrier sensors m_1 and m_2, positioned in the points $M_1 = (0, d/2)$ and $M_2 = (0, -d/2)$. We obtain a new Voronoi polygon $V'(s)$ for s, such that $V(s) \subseteq V'(s)$. As s does not cover the vertex V_z, it also does not cover the unique vertex of the bigger enclosing polygon $V'(s)$. Let us denote with V such a vertex, which is the unique intersection of the perpendicular bisectors of the segments $\overline{PM_1}$, $\overline{PM_2}$ and $\overline{M_1M_2}$, as shown in Figure 2.

$$V = (x_v, y_v) = \left(\frac{x_p^2 + y_p^2 - d^2/4}{2 \cdot x_p}, 0 \right).$$

As V is not covered, it must also be located in $\mathcal{R}(r_2)$, therefore $x_v > \Delta$, from which, recalling that $x_p < 0$, we derive: $(x_p - \Delta)^2 + y_p^2 \leq d^2/4 + \Delta^2 = R_s^2$.

Therefore the vertex V, generated when the sensor s is in $\mathcal{L}(\mathcal{A}_{critical})$, would actually be uncovered only if the sensor s were also located in $\mathcal{R}(\mathcal{A}_{critical})$, which is a contradiction. This implies that $Z \notin \mathcal{R}(r_2)$, concluding the proof that if $Z \in V(s) \cap \mathcal{R}(r_0)$ then Z must also belong to $\mathcal{C}(s)$. □

The above Lemma 2 admits the following special case that follows by considering that r_1 is tangential to $\mathcal{A}_{critical}$.

Lemma 3. *Let us consider a legitimate sensor s, positioned in $P = (x_p, y_p)$, with $P \in \mathcal{L}(r_1)$, and a d-spaced barrier of malicious sensors, with $d \leq \sqrt{3} \cdot R_s$. It holds that $V(s) \cap \mathcal{R}(r_1) \subseteq \mathcal{C}(s)$.*

Lemma 3 is necessary to demonstrate that no sensor located in $\mathcal{L}(r_1)$ crosses the barrier with a Voronoi based movement. Therefore a minimum distance threshold

$$d_T(R_s, d) \triangleq R_s - \sqrt{R_s^2 - d^2/4}$$

can be defined, so that no sensor located at a distance higher than d_T from the barrier can cross it by means of a pure Voronoi based movement. We study the two criteria separately.

Theorem 3.1. *Let us consider a network of mobile sensors, with sensing radius R_s, being deployed according to the FV criterion. Let us consider a d-spaced barrier of malicious sensors with step $d \leq \sqrt{3} \cdot R_s$. No legitimate sensor located at a distance longer than $d_T(R_s, d)$ from the barrier, is able to traverse it.*

Proof. Let us consider a Cartesian reference so that the barrier is deployed along the axis $x = 0$, in the positions $(0, d/2 + k \cdot d)$, with $k \in \mathbb{N}$. Due to symmetry we consider the left side only.

Consider a legitimate sensor s, located at a distance higher than $d_T(R_s, d)$ from the barrier, on the left side of it. It follows that $P_s \in \mathcal{L}(r_1)$, where r_1 is the line of equation $x = -d_T(R_s, d)$. Thanks to Lemma 3, we can assert that the Voronoi polygon $V(s)$ of the sensor s does not have any uncovered vertex on the right side of the line r_1. Therefore, given the rules of the FV criterion described in Section 2, either the sensor s does not move, or its destination D is also in the left side of the line r_1, that is $D \in \mathcal{L}(r_1)$. Since any movement of s, even if performed in multiple steps, will carry s from its current position P_s in $\mathcal{L}(r_1)$ to a destination D which is also in $\mathcal{L}(r_1)$, and since the region $\mathcal{L}(r_1)$ is convex, all the paths traversed by s are internal to $\mathcal{L}(r_1)$ and s never crosses the line $x = 0$ at which the barrier is deployed. □

We proceed with the analysis of the MiniMax criterion. We recall that the MiniMax point of a polygon is the center of its smallest enclosing circle.

Lemma 4. *Let $\mathcal{P}$ be a convex polygon with N vertices, and let $\mathcal{E}_\mathcal{P}$ be the minimum enclosing circle of $\mathcal{P}$. Every arc of 180° degrees in $\mathcal{E}_\mathcal{P}$ must traverse at least one vertex of the polygon $\mathcal{P}$.*

Proof. The minimum enclosing circle $\mathcal{E}_\mathcal{P}$ of a polygon $\mathcal{P}$ has at least two vertices of $\mathcal{P}$ on its boundary. As discussed in [10] we have two cases. Case (1): only two vertices of $\mathcal{P}$ are on the boundary of $\mathcal{E}_\mathcal{P}$, and they are antipodal. In such a case the two vertices divide $\mathcal{E}_\mathcal{P}$ into two half-circles. Case (2): more than two vertices of $\mathcal{P}$ are on the boundary of $\mathcal{E}_\mathcal{P}$, and three of these vertices form a non-obtuse triangle (or $\mathcal{E}_\mathcal{P}$ would not be minimal). In this case, the center of $\mathcal{E}_\mathcal{P}$ would coincide with the circumcenter of such a triangle and the angular distance between any two vertices would be less than or equal to 180° degree. It follows that in both cases every arc of the circumference whose length is 180° degree must contain at least one vertex of the polygon $\mathcal{P}$. □

We now give a characterization of the possible positions of the MiniMax point of $V(s)$ on the basis of the position of the sensor s with respect to the barrier.

Lemma 5. *Let us consider a d-spaced barrier of malicious sensors, with $d \leq \sqrt{3}R_s$, along the y-axis of a Cartesian reference and consider a legitimate sensor s positioned in $P \in \mathcal{R}(r_0) \cap \mathcal{L}(\mathcal{A}_{critical})$. If the Voronoi polygon $V(s)$ is not completely covered then its MiniMax point $M \in \mathcal{L}(P)$.*

Proof. Let us refer to Figure 2. As the lines r_0 and r_2 cross the intersection points between pairs of sensing circles of barrier sensors, the region $\mathcal{R}(r_0) \cap \mathcal{L}(r_2)$ is completely covered by the barrier sensors. The width of such a region is $w = 2\Delta$. Since $d \leq \sqrt{3}R_s$ then $w \geq R_s$. For vertical periodicity and horizontal symmetry in the construction, let us only consider the case with $0 \leq y_p \leq \frac{d}{2}$.

We initially neglect the presence of other sensors but s and the barrier sensors located in $M_0 = (0, \frac{3d}{2})$, $M_1 = (0, \frac{d}{2})$, and $M_2 = (0, \frac{-d}{2})$. As sensor s approaches the barrier, $\mathcal{C}(s)$ can have a non null intersection with the barrier sensors, generating two vertices of the Voronoi polygon $V(s)$ (drawn in blue in Figure 2). As the closest barrier sensors are M_1 and M_2, the vertex V generated with these sensors is the closest to the barrier. Due to Lemma 2, as $s \in \mathcal{R}(r_0) \cap \mathcal{L}(\mathcal{A}_{critical})$, the portion of its Voronoi polygon $V(s)$ located in $\mathcal{R}(r_0)$ is completely covered and is covered by s. Therefore the uncovered points of $V(s)$ lie in the region $\mathcal{L}(r_0)$.

Let us denote with I_1 and I_2 the intersection points of $V(s)$ with the boundary of the sensing circle $\mathcal{C}(s)$. The uncovered points of $V(s)$ must be located beyond the arc $\overset{\frown}{I_1 I_2}$, in the left region of the segment $\overline{I_1 I_2}$. P is more distant than R_s from the uncovered points whereas its distance from all the vertices in $\mathcal{R}(P)$ is lower than R_s.

We will now prove that no point of $V''(s) \triangleq V(s) \cap \mathcal{R}(P)$ can be the MiniMax of $V(s)$. We proceed by contradiction. Assume that $X \in V''(s)$ is the MiniMax of $V(s)$, and $V(s)$ has uncovered points. The requirement given by Lemma 4, establishes that X be the center of a circle which crosses at least one vertex of $V(s)$ every 180° degrees.

As the angle formed by I_1 and I_2, with any point of $V''(s)$ at the right hand side of the segment $\overline{I_1 I_2}$ is wider than 180° degrees, Lemma 4 states that an enclosing circle centered in X must cross one of the two vertices V and J, formed with M_1 and M_2 in addition to one or more uncovered vertices at the left side of the arc $\overset{\frown}{I_1 I_2}$, for a total of two or three vertices. Furthermore, the enclosing circle must cross the circumference $\mathcal{C}(s)$ in two points (in order to include an external region), which requires X to be in $\mathcal{L}(P)$[1].

In order to finish the proof we recall that in all this reasoning we neglected the presence of other sensors besides s, and the three barrier sensors located in M_0, M_1 and M_2. The argument remains valid even when considering other sensors, as they would cover additional portions of the AoI and of $V(s)$, and the potentially uncovered portion of the arc $\overset{\frown}{I_1 I_2}$ could only be smaller, leaving even wider angles at its right than we considered in the first part of the proof. Therefore, although having additional sensors may reduce the size of $V(s)$, when coverage of $V(s)$ is incomplete and s is in $P \in \mathcal{R}(r_0) \cap \mathcal{L}(\mathcal{A}_{critical})$, the MiniMax point would be in $\mathcal{L}(P)$. □

Theorem 3.2. *Let us consider a network of mobile sensors, with sensing radius R_s, being deployed according to the MiniMax criterion. Let us consider a d-spaced barrier of malicious sensors, with step $d \leq \sqrt{3} \cdot R_s$. No legitimate sensor located at a distance longer than $d_T(R_s, d)$ from the barrier, is able to traverse it.*

Proof. Thanks to symmetry, we can consider the only left side of our reference plane.

Consider a legitimate sensor s, located at a distance higher than $d_T(R_s, d)$ from the barrier, on the left side of it. Then $P_s \in \mathcal{L}(r_1)$. If P_s is located in $\mathcal{L}(r_0)$, it may have uncovered portions of its polygon $V(s)$ in the half plane $\mathcal{R}(P)$. Therefore its MiniMax point can also be in $\mathcal{R}(P)$. Nevertheless, by analyzing the coordinates of the point V, we derive that $V \in \mathcal{L}(r_1)$. Therefore, the whole polygon $V(s)$ and so its MiniMax point M, are also in $\mathcal{L}(r_1)$.

By contrast, if $P_s \in \mathcal{R}(r_0) \cap \mathcal{L}(r_1)$, Lemma 5 allows us to conclude that the MiniMax point M resides in $\mathcal{L}(r_1)$ or the polygon is completely covered and no movement occurs. Therefore, the destination D is also in the left side of the line r_1. Since any movement of s, even if performed in multiple steps, will carry s from its current position P_s in $\mathcal{L}(r_1)$ to a destination D which is also in $\mathcal{L}(r_1)$, and since the region $\mathcal{L}(r_1)$ is convex, all the path traversed by s must be internal to $\mathcal{L}(r_1)$. Therefore s never crosses the line $x = 0$ at which the barrier is deployed. □

The above theorems show that the number of malicious sensors necessary to impede complete coverage of an area only depends on the perimeter of the area, regardless of the number of legitimate sensors deployed.

Notice also that the OM attack has no impact on an already deployed network which provides full coverage of the AoI.

Theorem 3.3. *Under VOR, once legitimate sensors have achieved full coverage of the AoI, the OM attack cannot cause the movement of any sensors.*

Proof. Let us consider a legitimate sensor s with neighbors $N(s)$. Since the AoI is completely covered, $V(s)$ is also completely covered, hence s does not move. When the OM attack starts, s has a set of neighbors $\widehat{N}(s)$, which may include some additional malicious sensors, and a polygon $\widehat{V}(s)$. Since $N(s) \subseteq \widehat{N}(s)$ then $\widehat{V}(s) \subseteq V(s)$, thus $\widehat{V}(s)$ is also completely covered, hence s does not move, in agreement with the rules described in Section 2. □

4 THE SECUREVOR ALGORITHM

In this Section we introduce SecureVor, a secure Voronoi-based deployment algorithm.

SecureVor is designed on the basis of the adversary model introduced in Section 3. It assumes a signature protocol to verify the exchanged messages, and an algorithm to verify position claims of nodes within the communication range R_{tx} [12], [28][2]. SecureVor assumes that $R_{tx} > 4R_s$ and sets $d_{max} = R_{tx}/4 - R_s$. We relax this assumption with the algorithm SSD, discussed in Section 5. Notice that, we do not require the communication range of a sensor to be a perfect disk. Indeed, there can be anisotropies provided that a sensor is able to communicate with all sensors located at a distance up to $4R_s$ from itself. Finally, similar to previous works [26], [4] on mobile sensor deployment, we assume that nodes are endowed with consumer grade GPS[3] and that they are loosely synchronized.

SecureVor provides a method to recognize malicious sensors and detect malicious movements when the deployment is based on VOR. It can be applied to both moving

1. This is because: 1) an enclosing circle bigger than $\mathcal{C}(s)$ and centered in $\mathcal{R}(P)$ would not touch any vertex in an angle wider than 180° degrees, contradicting Lemma 4. Therefore if X had a radius $R_X > R_s$, it would be in $\mathcal{L}(P)$; 2) an enclosing circle of the same size as $\mathcal{C}(s)$ or even smaller is also possible, but in order for it to include points that are external to $\mathcal{C}(s)$ on the left side of the arc $\overset{\frown}{I_1 I_2}$, in addition to both vertices in $\mathcal{R}(P)$ it must be centered in $X \in \mathcal{L}(P)$.

2. Location verification can be achieved by using dedicated hardware and/or previously deployed anchor nodes. Sensors can autonomously verify position claims if they are equipped with a radar system [12], [28]. These radars conform to our requirements as they are inexpensive, low power and provide object detection up to 20m distance. Alternatively, Ultra Wide Band systems [13] and anchor nodes can be used for location verification through Verifiable Multilateration (VM) [7]. In this case, anchor nodes are responsible for the location verification and advertise false location claims when detected. Using VM, a sensor incurs in a constant communication overhead for each anchor it communicates with.

3. Low-cost, consumer grade GPS currently available provide accuracy in the orders of few decimeters [6] and have a cost around 200$ per unit [21].

strategies FV and MiniMax. The idea of SecureVor is to detect malicious nodes by verifying the compliance of their movements to the rules of the deployment algorithm in use. This verification activity allows each sensor to formulate its own list of *trusted* and *untrusted* sensors. Each sensor will ignore untrusted neighbors and use only the information exchanged with trusted ones to determine future movements.

In order to let sensors reciprocally verify each other's movement, at the beginning of each round every sensor s is required to declare the set of its trusted neighbors, namely the set of sensors that it will use to determine its polygon. Notice that, a sensor determines this set only on the basis of its local observation, since SecureVor does not require transitive trust among sensors. The neighbor sensors of s locally calculate the polygon of s, based on its stated set, and verify whether its movement is in compliance with the deployment algorithm or not. If a malicious movement is detected, s is marked as untrusted and ignored by its neighbors thereafter. SecureVor, and similarly SSD, could be extended with reputation-based techniques [24], [16].

Let N be the set of sensors to be deployed. We recall from Section 2 that we denote by $N_{tx}^{(t)}(s)$ the neighbors of s, that is the set of sensors that are, at round t, at a distance less then the communication radius R_{tx} from s. The sets $N_{trusted}^{(t)}(s)$ and $N_{untrusted}^{(t)}(s)$ keep track, for a sensor s, of the set of sensors that s considers as *trusted* and *untrusted*, respectively, until round t. These sets are updated at each round. According to SecureVor, a sensor s only considers neighbors at a distance less than $R_{tx}/2$ as potential neighbors to calculate its own polygon. We refer to such neighbors at a round t as $Q^{(t)}(s)$. This choice enables s to be in communication with the sensors considered by its neighbors in $Q^{(t)}(s)$ to determine their polygon. Among the nodes in $Q^{(t)}(s)$, s takes into account only the sensors that it considers as trusted in order to determine its polygon. We define the set of sensors that s actually considers at round t as $N_{SV}^{(t)}(s) = Q^{(t)}(s) \bigcap N_{trusted}^{(t)}(s)$. $N_{SV}^{(t)}(s)$ may be empty if s has no trusted neighbor in its proximity at round t. In such a case, $V^{(t)}(s)$ is the whole AoI. Finally, the position of sensor s at the current round is denoted with $pos^{(t)}(s)$. Table 1 summarizes the adopted notation.

Notation	Description
$V^{(t)}(s)$	Polygon of s
$N_{tx}^{(t)}(s)$	Neighbors of s (distance $\leq R_{tx}$)
$Q_{tx}^{(t)}(s)$	Neighbors of s (distance $\leq R_{tx}/2$)
$N_{trusted}^{(t)}(s)$	Sensor s trusted neighbors until round t
$N_{untrusted}^{(t)}(s)$	Sensor s untrusted neighbors until round t
$pos^{(t)}(s)$	Position of s
$\widehat{pos}^{t}(s)$	Expected position of s
$N_{SV}^{(t)}(s)$	Sensors considered by s to build $V^{(t)}(s)$

TABLE 1
Summary of adopted notation. All notations refer to round t.

4.1 SecureVor in detail

SecureVor is round based similar to VOR. In particular, it comprises four phases, namely: *Position communication*, *Movement verification*, *Trusted neighbors communication* and *Coverage evaluation and movement*. Notice that we do not consider localization errors of the GPS positioning system or of the location verification algorithm. SecureVor can be extended to take these aspects into account with the same approach described in Section 8.4 of [2].

The pseudo-code is shown as Algorithm SecureVor.

Algorithm SecureVor, node s at round t.

```
   // Position communication:
1  Broadcast pos^(t)(s);
2  Receive and verify neighbor positions;
3  Determine the sets N_tx^(t)(s) and Q^(t)(s);
   // Movement verification:
4  if t = 0 then
5      N_untrusted^(t)(s) = ∅;
6      N_trusted^(t)(s) = N;
7  else
8      N_untrusted^(t)(s) = N_untrusted^(t-1)(s) ∪ (Q^(t-1)(s) \ N_tx^(t)(s));
9      for q ∈ Q^(t)(s) s.t. q ∉ N_untrusted^(t)(s) do
10         if (s ∉ N_trusted^(t-1)(q) ∨ N_trusted^(t-1)(q) ⊈ N_tx^(t-1)(s)) then
11             N_untrusted^(t)(s) ← q;
12         Calculate V^(t-1)(q);
13         Calculate pos^t(q);
14         if pos^t(q) ≠ pos^t(q) then N_untrusted^(t)(s) ← q;
15     N_trusted^(t)(s) = N \ N_untrusted^(t)(s);
16     N_SV^(t)(s) = Q^(t)(s) ⋂ N_trusted^(t)(s);
   // Trusted neighbors communication:
17 Broadcast the list of nodes in N_SV^(t)(s);
18 Receive N_SV^(t)(z) from any z ∈ Q^(t)(s);
   // Coverage evaluation and movement:
19 Calculate V^(t)(s) on the basis of N_SV^(t)(s);
20 if V^(t)(s) is completely covered then do not move;
21 else Determine destination point and move accordingly.
```

Position communication (lines 1-3)
At the beginning of a round each sensor communicates its position to the neighbors through a signed message and determines the sets $N_{tx}^{(t)}(s)$ and $Q^{(t)}(s)$, which are the set of communication neighbors of s and the set of nodes located at less than $R_{tx}/2$ from s, respectively.

Movement verification (lines 4-16)
In this phase, a sensor s verifies the movements of its neighbors to determine $N_{trusted}^{(t)}(s)$, $N_{untrusted}^{(t)}(s)$ and ultimately $N_{SV}^{(t)}(s)$. At the first round, $N_{trusted}^{(t)}(s) = N$ and $N_{untrusted}^{(t)}(s) = \emptyset$ **(lines 4-6)**.

The set of untrusted neighbors at round $t > 1$, $N_{untrusted}^{(t)}(s)$, contains all the sensors that were determined as untrusted in any of the previous rounds $N_{untrusted}^{(t-1)}(s)$ plus the sensors that were in $Q^{(t-1)}(s)$ and that are no longer in communication with s at the current round **(line 8)**[4]. Other sensors that are detected as malicious in the current round are added to $N_{untrusted}^{(t)}(s)$ **(lines 9-16)** as explained in the following.

A sensor s verifies, for each sensor q in $Q^{(t-1)}(s)$, not yet in $N_{untrusted}^{(t)}(s)$, the correctness of its movement in the previous round[5]. The first check that s performs for a sensor q, in order to verify the correctness of its movement, is on the truthfulness of the set $N_{SV}^{(t-1)}(q)$ **(lines 12-13)**. Two inconsistencies can be detected by s.

First inconsistency: the sensor q may have maliciously omitted s itself in the set of its trusted neighbors. Since s

4. SecureVor imposes that a sensor travels a maximum distance $d_{max} = R_{tx}/4 - R_s$. Hence even if two sensors, at a distance at most $R_{tx}/2$, move in opposite directions, they will stop at a distance from each other less than $R_{tx}/2 + 2(R_{tx}/4 - R_s)$ which is less than R_{tx}. This means that $Q^{(t-1)}(s) \subseteq N_{tx}^{(t)}(s)$, so if a sensor in $Q^{(t-1)}(s)$ is not in $N_{tx}^{(t)}(s)$, s can mark it as untrusted.

5. Notice that, the trustworthiness of the sensors belonging to $Q^{(t)}(s) \setminus Q^{(t-1)}(s)$ will be evaluated at the next round.

knows that it has behaved correctly according to the moving strategy, q must include s in its trusted set.

Second inconsistency: the sensor q may have fabricated the presence of some sensors in $N_{SV}^{(t-1)}(q)$ which are not physically located in its proximity to justify its movement. Sensor s can detect such malicious behavior because, according to SecureVor, a sensor q must select the sensors in $N_{SV}^{(t-1)}(q)$ among those in $Q^{(t-1)}(q)$. In order to be in $N_{SV}^{(t-1)}(q)$, a sensor must be at a distance at most $R_{tx}/2$ from q which implies that it is at a distance at most R_{tx} from s, being q at a distance at most $R_{tx}/2$ from s ($q \in Q^{(t-1)}(s)$). More formally $N_{SV}^{(t-1)}(q) \subseteq Q^{(t-1)}(q) \subseteq N^{(t-1)}(s)$.

If an inconsistency is detected, q is marked as untrusted and will be ignored by s hereafter. If no inconsistency is detected, the sensor s verifies whether q has moved according to the nodes belonging to $N_{SV}^{(t-1)}(q)$ **(lines 14-16)**. To this aim, s calculates the polygon of q at the previous round $V^{(t-1)}(q)$ on the basis of $N_{SV}^{(t-1)}(q)$ and $pos^{(t-1)}(q)$. Sensor s then compares the current position $pos^{(t)}(q)$, which q has just broadcast in the previous phase, with the expected position of q at the current round, $\widehat{pos}^{(t)}(q)$, calculated considering the polygon $V^{(t-1)}(q)$ and $pos^{(t-1)}(q)$. If $pos^{(t)}(q)$ is different from $\widehat{pos}^{(t)}(q)$, sensor s marks q as untrusted.

Trusted neighbors communication (lines 19-20)
In this phase each sensor s broadcasts a signed message containing the IDs of the nodes belonging to the set $N_{SV}^{(t)}(s)$ calculated in the previous phase. This information enables the neighbors of s to verify its movement at the next round.

Coverage evaluation and movement (lines 21-23)
This phase is the same as the original VOR approach described in Section 3, except that each sensor s calculates its Voronoi polygon $V^{(t)}(s)$ on the basis of the sensors in $N_{SV}^{(t)}(s)$. Furthermore s looks for a destination point p within a distance $d_{max} = R_{tx}/4 - R_s$ instead of $d_{max} = R_{tx}/2 - R_s$.

5 THE SSD ALGORITHM

In this section we describe the SSD algorithm, designed to work in scenarios for which the hardware available at the sensor nodes does not satisfy the requirement on the transmission radius of SecureVor. In particular, unlike SecureVor which requires $R_{tx} > 4R_s$, SSD works under the same assumption of the original VOR algorithm, i.e. $R_{tx} > 2R_s$. Except for the transmission radius, SSD adopts the same assumptions of SecureVor discussed in Section 2.

The algorithm SSD explicitly aims at solving the blocked movement situation geometrically characterized in Section 3.1, in which a legitimate sensor does not move towards uncovered regions because it is in front of a barrier of malicious sensors.

Because this algorithm works under the relaxed assumption $R_{tx} > 2R_s$, sensors are not able to verify the movement of their neighbors only on the basis of message exchanges. This is because the communication range is too small to let a sensor verify whether its neighbors are behaving consistently with what should be their Voronoi polygon. Hence a sensor is not able, on the basis of messages alone, to distinguish a blocked movement situation (under attack) from a normal condition in which it cannot contribute a better coverage. In both cases the polygon of the sensor is completely covered and the sensor is not required to move to increase coverage of its polygon.

For these reasons, SSD provides temporary position swaps among pairs of neighbors to be performed when sensors are stationary and potentially in a blocked movement situation. We show a high level pseudocode in Algorithm SSD. As in the case of SecureVor, SSD requires each sensor s to maintain a list of trusted neighbors at time t: $N_{trusted}^{(t)}(s)$. In the next section we describe SSD in detail, making use of a similar nomenclature to the one introduced for SecureVor in Section 4.

5.1 SSD in detail

As in the case of SecureVor, according to SSD, each sensor s updates the list of trusted neighbors $N_{trusted}^{(t)}(s)$ at each round t. Such a set initially includes all the network nodes N **(line 2)**. At round t, s calculates its Voronoi polygon $V^{(t)}(s)$ by taking account only of the sensors in the set $N_{SSD}^{(t)}(s)$, which is defined as the set of sensors in its radio proximity that s considers as trusted, i.e. $N_{SSD}^{(t)}(s) = N_{trusted}^{(t)}(s) \bigcap N_{tx}^{(t)}(s)$ **(lines 6-7)**.

If $V^{(t)}(s)$ is completely covered, s should remain still. Nevertheless this situation may occur in the presence of an attack. Therefore SSD provides the following mechanism to perform legitimacy checks of the behavior of its Voronoi neighbors. In order to determine the presence of malicious sensors, the sensor s selects one of its Voronoi neighbors and temporarily swaps its position with it. In order to prevent conflicting requests, these are generated at random times in a given time interval and served according to a FIFO discipline.

We now describe the process and conditions that result from a legitimate sensor being bound by a malicious barrier. When sensors are spread from a safe location, a legitimate sensor encounters a barrier that was initially far from it. When a sensor is blocked by some malicious sensors of the barrier, its Voronoi polygon is determined by new neighbors which were not previously observed, or by neighbors with which s forms a vertex that was uncovered in any previous round. If the sensor s moves towards a steady barrier, it converges to a position in which its polygon has vertices at the boundary of the sensing regions of a barrier sensor and therefore of the sensor s itself; this occurs because initially the sensor s forms uncovered vertices with barrier sensors, and it performs additional movements of smaller and smaller size, until it stops due to complete coverage. Figure 3 shows a legitimate sensor at the left of a malicious barrier, which is in a blocked movement situation, forming two vertices V and W which are both newly covered and located at the boundary of the sensing region.

We call any of the Voronoi neighbors of s resulting from this scenario the *vertex neighbor* of s[6].

Sensor s invites one of its new vertex neighbors, let it be j, to perform a swap of positions **(lines 8-10)**. The purpose of this swap is to let s perform a legitimacy check of j in order to calculate and verify its expected future movement.

6. Notice that in order to provide convergence in a finite number of steps both VOR and SSD provide a movement threshold which prevents infinitesimal movements. Such a threshold is also kept into account in the definition of a vertex neighbor.

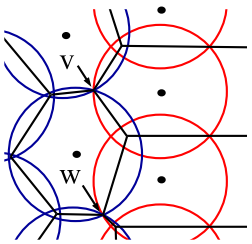


Fig. 3. Boundary vertices V and W, between a legitimate sensor and two barrier sensors.

As j itself calculates its Voronoi polygon on the basis of its set of neighbors $N_{\text{SSD}}^{(t)}(j)$, the position swap requires also j to send this set to s in order to let s be able to properly calculate the expected future movement of sensor j **(line 11)**. After this information exchange, s moves to the position currently held by j **(line 12)**, while j is required to move towards the position previously held by s. This last movement of j is required to ensure that the position of s is continuously covered and that the movements of other neighbors of s can be correctly verified. Sensor s exploits its location verification capabilities to verify if j honored the position swap protocol, otherwise it removes j from its local list of trusted sensors, therefore $N_{trusted}^{(t)}(s) \leftarrow N_{trusted}^{(t)}(s) \backslash \{j\}$ **(lines 14-15)**.

Algorithm SSD, executed by node s at round t.

```
1  if t=0 then
2      N_trusted^(t)(s) ← N;
3  Exchange position msgs, determine N_tx^(t)(s);
   // Movement verification:
4  if (swapped with j at time (t−1)) ∧ (pos^(t)(j) ≠ p̂os^(t−1)(j)) then
5      N_trusted^(t)(s) ← N_trusted^(t)(s)\{j} ;
6  Let N_SSD^(t)(s) ← N_trusted^(t)(s) ⋂ N_tx^(t)(s);
7  Update V^(t)(s) based on N_SSD^(t)(s);
   // Coverage evaluation and Swap Agreements:
8  if V^(t)(s) is covered ∧ # of new vertex neighbors ≥ 2 then
9      Select a Vertex Neighbor j;
10     Send swap_request to j;
11     Receive N_SSD^(t)(j) from j and send N_SSD^(t)(s) ;
12     Move to pos^(t)(j) and send neighbor discovery msg;
13     Receive position msgs and determine N̂_tx^(t)(j);
14     if (j did not reach pos^(t)(s)) then
15         N_trusted^(t)(s) ← N_trusted^(t)(s)\{j} ;
16     else
17         if (N_SSD^(t)(j) ⊆ N̂_tx^(t)(j)) then
18             Calculate V^(t)(j) on the basis of N_SSD^(t)(j);
19             Calculate p̂os^(t)(j) ;
20             move to pos^(t)(s);
   // Voronoi's Movement Phase
21 else
22     Move according to VOR criterion;
```

Once in the position previously held by sensor j, the sensor s sends a neighbor discovery message. The discovered list of communication neighbors of j is hereby denoted with $\widehat{N}_{\text{tx}}^{(t)}(j)$. Sensor s and j send the list of communication trusted neighbors to each other. After this message exchange, thanks to its location verification capabilities, s verifies the consistency of the list of neighbors received by j, namely $N_{\text{SSD}}^{(t)}(j)$ **(line 17)**, with the list of neighbors it observed while in the place of j, namely $\widehat{N}_{\text{tx}}^{(t)}(j)$ **(line 13)**. If such consistency check fails, or if the trusted set of j does not include s, s does not return to its original position and continues the algorithm execution from the former position of sensor j. If otherwise the consistency check succeeds, sensor s is now able to calculate the Voronoi polygon of sensor j and the expected movement that j should perform **(lines 18-19)**. Sensors s and j can now return to their original positions **(line 20)**.

After this temporary swap activity, the algorithm SSD proceeds with the execution of the regular activities provided by the Voronoi approach **(lines 21-22)**. During the next movement phase, s verifies the movement of j using the location verification capabilities. If, during the next movement phase, sensor j fails to perform the expected movement calculated by s, it is removed from the trusted list of sensor s to be used at the next round **(lines 4-5)**. From now on, the sensor s will consider j as untrusted and will ignore it and adapt its Voronoi polygon and coverage as if j did not exist.

SSD provides some additional mechanisms to prevent more complex behaviors of malicious sensors. For example, malicious sensors could refuse to fulfill swap requests pretending to be involved in other position swap activities (with other malicious sensors). In order to prevent this behavior, first, SSD requires that a sensor which refuses a swap request provide a proof of the previous swap agreement (signed messages of both involved parties). Second, according to SSD, a random permutation $P^{(t)}$ of the sensor IDs is generated at each round, using the round counter t as a seed. This permutation is common to all sensors, and it establishes a priority in the position swap activities. In particular, sensors with higher priority at the current round have precedence in swapping, thus preventing two malicious sensors to continuously swap only between themselves. In addition, SSD allows the same pair of sensors to swap positions only once. Although a higher number of swaps per pair would increase the accuracy of detection of malicious sensors, this would be at the expense of energy for movements. Furthermore, by limiting the number of swaps, we prevent malicious sensors from extinguishing the batteries of legitimate neighbor sensors demanding unnecessary swaps. Note that, for the sake of simplicity the pseudo code in Algorithm SSD does not address the additional mechanisms described above, nor does it cover possible synchronization issues, and the case of s receiving swap requests from other sensors, which is treated according to the permutation priority described above.

6 Algorithm Properties

In this section we provide a theoretical analysis of SecureVor and SSD. We hereafter denote with L and M the set of legitimate and malicious sensors, respectively. Hence, the total number of sensors deployed over the AoI is $|N| = |L| + |M|$.

6.1 Properties of SecureVor

We first study the capability of SecureVor to counteract the OM attack. Notice that, if a malicious node m moves in compliance to VOR it cannot be detected, since it is actually

behaving as a legitimate sensor. Nevertheless, such movements are unlikely to meet the attacker goals. We define a *malicious movement* of a malicious sensor as a movement which is not in compliance with the deployment rules. Furthermore, given a malicious sensor $m \in M$ performing a malicious movement at round t, we define the set L_m^t as the set of legitimate sensors whose movement can be influenced by the malicious movement of m.

Lemma 6. *Given a malicious sensor $m \in M$ performing a malicious movement at round t, if $L_m^t \neq \emptyset$ then m is marked as untrusted by at least one sensor in L_m^t at round $t+1$.*

Proof. Since m can influence the movement of the sensors in L_m^t, such sensors consider m as trusted at the current round. Furthermore, since we assume that a node considers only sensors at a distance $R_{tx}/2$ to determine its polygon, $\forall s \in L_m^t\ d(s,m) < R_{tx}/2$ thus s is able to verify if $N_{SV}^{(t)}(m)$ is inconsistent. As a result, according to the assumptions made in Section 3, the only degree of freedom that m has in order to try to justify its malicious movement without being detected lies in the selection of the nodes to be advertised in $N_{SV}^{(t)}(m)$. Notice that all nodes in L_m^t are legitimate and are at a distance less than $R_{tx}/2$ from m, thus such sensors should be included in the trusted set of m. If m does not include one or more of them in $N_{SV}^{(t)}(m)$, such sensors mark m as untrusted at round $t+1$ and the assertion is valid.

If, on the contrary, m includes all sensors in L_m^t in $N_{SV}^{(t)}(m)$, such sensors are in communication range with m at round $t+1$ since $\frac{R_{tx}}{2} + 2d_{max} < R_{tx}$. As a result, sensors in L_m^t are able to verify the correctness of the current movement of m at the next round. Since m is performing the OM attack, its malicious movement is detected and thus all sensors in L_m^t mark m as untrusted at round $t+1$. □

We now prove that SecureVor terminates in a finite time. To this purpose, we show that at each round, either at least a malicious sensor is detected, or the overall coverage provided by legitimate sensors increases. We define a network state as follows.

Definition 6.1. *A* network state *under SecureVor is a vector $S_{SV} =< c_1, \ldots, c_{|M|}, s_1, \ldots, s_{|L|}, m_1, \ldots, m_{|M|} >$ where c_j is the number of legitimate sensors which consider the malicious sensor $m_j \in M$ as untrusted, $s_i \in L$ for $i = 1, \ldots, |L|$ and $m_j \in M$ for $j = 1, \ldots, |M|$.*

We define a function $f_{\text{SV}} : \mathbb{N}^{|M|} \times L^{|L|} \times M^{|M|} \rightarrow \mathbb{N} \times \mathbb{R}_+$ such that given a network state S_{SV}, $f_{\text{SV}}(S_{\text{SV}}) = (\sum_{j=0}^{|M|} c_j, A_{total})$, where A_{total} is the size of the area covered by legitimate sensors in S_{SV}. Given two network states $S_{\text{SV}}^1, S_{\text{SV}}^2$ we say that $f_{\text{SV}}(S_{\text{SV}}^1) \prec f_{\text{SV}}(S_{\text{SV}}^2)$ according to the lexicographic order. Notice that, the function f_{SV} is upper-bounded by the pair $(|L||M|, AoI)$. In the following, in order to prove the convergence of SecureVor, we show that at each round the value of such function increases.

Theorem 6.1. *The algorithm SecureVor converges.*

Proof. Let us consider a generic state change from round t to round $t+1$. We want to show that $f_{\text{SV}}(S_{\text{SV}}^{(t)}) \prec f_{\text{SV}}(S_{\text{SV}}^{(t+1)})$. We recall that, for a malicious sensor $m \in M$ performing a malicious movement at round t, L_m^t is the set of legitimate nodes whose movement can be influenced by the malicious movement of m. We consider two cases:

Case 1: $\exists m_j \in M$ s.t. $L_{m_j}^t \neq \emptyset$.
Thanks to Lemma 6 we know that there exist at least one legitimate sensor at round $t+1$ that marks m_j as untrusted. As a result, $c_j[S_{\text{SV}}^{(t)}] < c_j[S_{\text{SV}}^{(t+1)}]$, hence $f(S_{\text{SV}}^{(t)}) \prec f(S_{\text{SV}}^{(t+1)})$.

Case 2: $\forall m_j \in M$, $L_{m_j}^t = \emptyset$.
In this case no malicious movement influences the movement of legitimate sensors. As a result no malicious sensor is detected at round $t+1$, hence $\forall\ j = 1, \ldots, |M|$, $c_j[S_{\text{SV}}^{(t+1)}] = c_j[S_{\text{SV}}^{(t)}]$. Notice that, if no malicious sensor is detected SecureVor lets sensors deploy according to the rules of VOR. Under VOR, if in a specific round at least one sensor moves, the provided coverage increases (as shown in the proof of Theorem 4.1 of [4]), so also in this case it holds that $f_{\text{SV}}(S_{\text{SV}}^{(t)}) \prec f_{\text{SV}}(S_{\text{SV}}^{(t+1)})$. As the function $f_{\text{SV}}()$ is upper-bounded and it increases at each round of the algorithm execution, we can conclude that SecureVor converges.

□

The above theorem proves that SecureVor converges, nevertheless the increase in coverage may be infinitesimal and the algorithm may require an infinite number of rounds to terminate.

Corollary 1. *The algorithm SecureVor terminates if movements are allowed only if they provide a coverage increase which exceeds a positive minimum threshold ϵ.*

The introduction of ϵ ensures fast termination and power saving, at the expense of a small loss in the coverage extension.

6.2 Properties of SSD

Similarly to SecureVor, to prove the termination of SSD we first show that it converges. We consider a static barrier of malicious sensors performing the BOM attack.

Definition 6.2. *A* network state *under SSD is a vector $S_{SSD} =< a_1, \ldots, a_{|L|}, s_1, \ldots, s_{|L|}, m_1, \ldots, m_{|M|} >$ where a_j is the number of swaps performed by the legitimate sensors s_j, $s_i \in L$ for $i = 1, \ldots, |L|$ and $m_j \in M$ for $j = 1, \ldots, |M|$.*

We define a function $f_{\text{SSD}} : \mathbb{N}^{|L|} \times L^{|L|} \times M^{|M|} \rightarrow \mathbb{N} \times \mathbb{R}_+$ such that given a network state S_{SSD}, $f_{\text{SSD}}(S_{\text{SSD}}) = (\sum_{j=0}^{|L|} a_j, A_{total})$, where A_{total} is the size of the area covered by legitimate sensors in S. Given two network states $S_{\text{SSD}}^1, S_{\text{SSD}}^2$ we say that $f_{\text{SSD}}(S_{\text{SSD}}^1) \prec f_{\text{SSD}}(S_{\text{SSD}}^2)$ according to the lexicographic order.

Notice that, the function $f_{\text{SSD}}()$ is upper-bounded by the pair $(|L| \times (|L| - 1 + |M|), AoI)$, since legitimate sensors are allowed to swap at most once with another sensor, and the maximum area that can be covered is the whole AoI. Similarly to the case of SecureVor, we show that during the unfolding of SSD the value of such function increases.

Theorem 6.2. *The algorithm SSD converges.*

Proof. Let us consider a generic network state $S_{\text{SSD}}^{(t)}$ at round t. We want to show that either the algorithm has terminated at round t, or there exists $k \in \mathbb{N}$ s.t. $f_{\text{SSD}}(S_{\text{SSD}}^{(t)}) \prec f(S_{\text{SSD}}^{(t+k)})$. We consider two cases:
Case 1: $\exists s_i \in L$ that performs a movement at round t.
Legitimate sensors deploy according to the rules of the VOR

approach. Since under VOR if a sensor moves then the overall coverage increases [4], this holds also under SSD. As a result, if at least one sensor moves then $f_{\text{SSD}}(S_{\text{SSD}}^{(\text{t})}) \prec f_{\text{SSD}}(S_{\text{SSD}}^{(\text{t+1})})$.

Case 2: $\nexists s_i \in L$ that performs a movement at round t.
This case occurs if no sensor can move and increase the coverage of its polygon, hence sensors will also not move at subsequent rounds. As a result, either the algorithm has terminated, or the network state may change as a consequence of a swap. Let us consider a sensor s_i which wants to exchange with s_j. s_i may not be able to exchange with s_j at round t, due to their priority in $P^{(\text{t})}$. However, the random generation of permutations ensures that there eventually exists $k \in \mathbb{N}$ s.t. in $P^{(\text{t+k})}$, s_i has higher priority than s_j, and the swap can be performed. In this case, the number of swaps increases from state $S_{\text{SSD}}^{(\text{t})}$ to $S_{\text{SSD}}^{(\text{t+k})}$, and in particular $a_i^{(\text{t+k})} = a_i^{(\text{t})} + 1$. As a result, $f_{\text{SSD}}(S_{\text{SSD}}^{(\text{t})}) \prec f_{\text{SSD}}(S_{\text{SSD}}^{(\text{t+k})})$.

As $f_{\text{SSD}}()$ is upper-bounded and it increases at each round, SSD converges. □

Similarly to SecureVor, to prove the termination of SSD we include a positive threshold $\epsilon > 0$, which prevents infinitesimal increase in coverage, as stated by the following corollary.

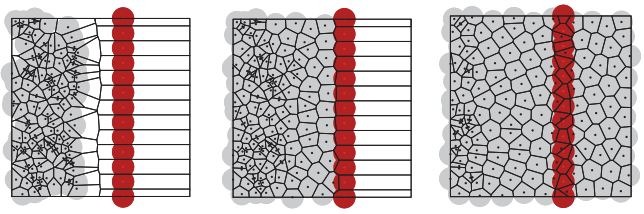
Corollary 2. *The algorithm SSD terminates provided that movements are allowed only if they enable a coverage increase greater than a threshold $\epsilon > 0$.*

Unlike SecureVor, we cannot formally prove that every time a malicious node is encountered in SSD it is detected, due to the limited information available at each sensor as a consequence of the smaller transmission radius than with SecureVor. In particular, we cannot exclude that a malicious sensor m is not detected during a swap, because its polygon is actually fully covered by some legitimate sensors that crossed the barrier in a previous round. Nevertheless, the experiments show that overall, SSD thwarts the OM attack. Hence, in Section 7 the effectiveness of SSD in defeating the OM attack is shown through extensive experiments, which also demonstrate the capability of SSD to achieve full coverage of the AoI.

7 Experimental results

In this section we provide an analysis of the performance of SecureVor and SSD. To this purpose, we developed a simulator on the basis of the Wireless module of the Riverbed Opnet simulation environment [31]. In the simulations we considered a squared AoI of size 80m×80m. Sensors can move at a maximum speed of 1m/s. We set the threshold ϵ for minimum coverage increase to 0.001, for both SecureVor and SSD. We investigated several scenarios which consider different settings of R_{tx} and R_s.

In the first scenario (Scenario A), we consider a setting favorable to SecureVor, i.e. such that $R_{\text{tx}} > 4R_{\text{s}}$. In the second scenario (Scenario B), we consider instead a setting for which SSD is designed, i.e. $4R_{\text{s}} > R_{\text{tx}} > 2R_{\text{s}}$. The third scenario (Scenario C) is devoted to a sensitivity analysis of both algorithms to the setting of the transmission radius. While all these scenarios consider a static BOM attack, the last experimental scenario (Scenario D) considers a BOM attack with a mobile barrier.



(a) (b) (c)

Fig. 4. Scenario A: Initial deployment of 150 legitimate sensors and 13 malicious sensors (a), final deployment of VOR (b), and SecureVor (c).

7.1 Scenario A

In this scenario we set $R_{tx} = 30$m and $R_s = 5$m, and investigate the performance of SecureVor. Under this setting, the maximum moving distance d_{max} is 2.5m. Malicious sensors perform the BOM attack by periodically advertising their position during the Position communication phase while remaining still. In order to avoid being easily detected by the surrounding legitimate sensors, each malicious sensor m, advertises a trusted set $N_{trusted}^{(\text{t})}(m) = Q^{(\text{t})}(m)$. Legitimate sensors are randomly deployed on the left side of the AoI.

We compare the performance of SecureVor with respect to the results obtained by the original VOR algorithm in the same setting. In order to evaluate the overhead introduced by SecureVor, we also show the behavior of VOR when all sensors are legitimate and expand freely (VOR-Free in the figures) without a barrier.

Before showing the results, we provide an example of the detrimental effect of the BOM attack in this scenario with 150 legitimate sensors and 13 malicious sensors. Figure 4(a) shows the initial deployment, while Figures 4 (b) and (c), show the final deployments achieved by VOR and SecureVor, respectively. Under VOR legitimate sensors are not able to cross the barrier, resulting in a significant loss of coverage. On the contrary, under SecureVor legitimate sensors detect malicious sensors, and are able to cross the barrier and achieve full coverage of the AoI.

In the experiments we set the number of malicious sensors to 13 and we increase the number of legitimate sensors from 60 to 240. Figure 5(a) shows the coverage of the AoI achieved by the considered algorithms. Legitimate sensors under VOR are not able to cross the barrier of malicious sensors, no matter how many legitimate sensors are deployed. Therefore the coverage is at most 60%.

On the contrary SecureVor, thanks to its security policy, detects and ignores malicious sensors and successfully covers the AoI. Note that, SecureVor achieves the same coverage of VOR-Free, that is the original Voronoi algorithm with no attack. This shows that SecureVor completely defeats the attack and maximizes the coverage.

Since under VOR sensors are not able to spread over the AoI when the attack is in place, this algorithm achieves lower values of all the considered performance metrics, such as traversed distance and consumed energy, with respect to the other algorithms. This does not imply superior performance of this algorithm, but just the inability to cover the AoI. For this reason, in the following we do not discuss its results although we show them in the figures.

Figure 5(b) shows the average distance traversed by sensors. SecureVor introduces a very small overhead in terms of traversed distance with respect to VOR-Free. The peak in the

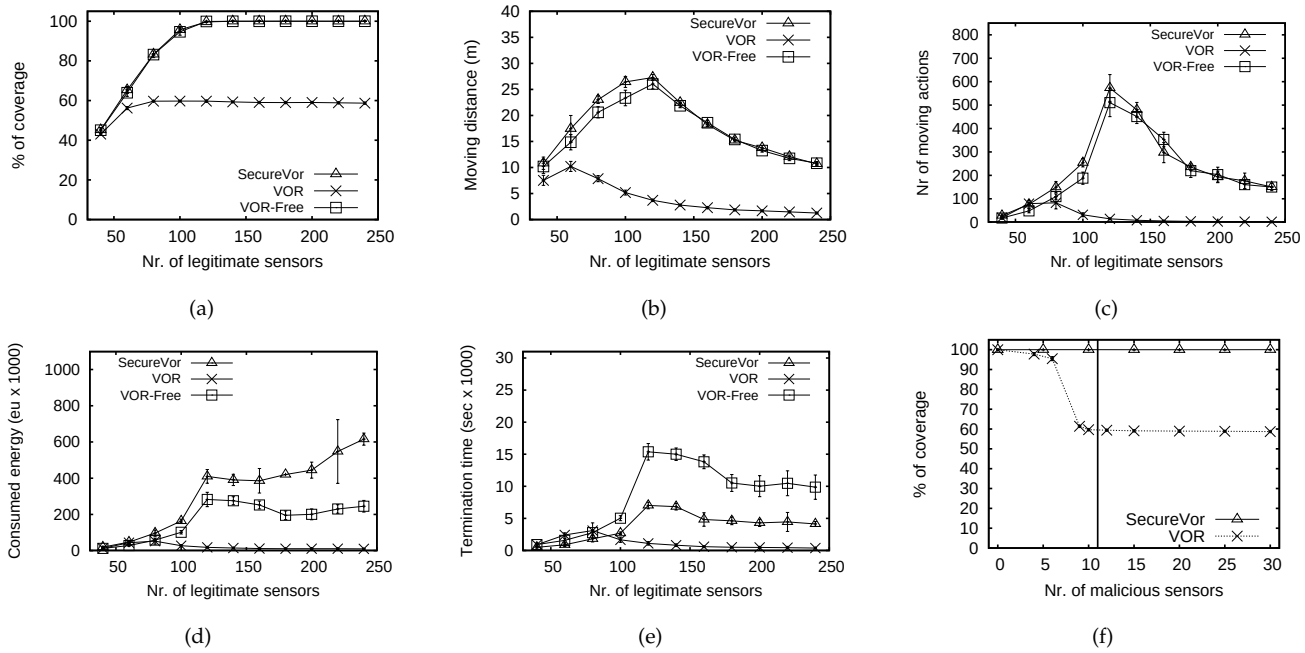


Fig. 5. Scenario A: coverage of the AoI (a), traversed distance (b), number of movements (c), consumed energy (d), termination time (e). Coverage achieved with 140 legitimate sensors (f).

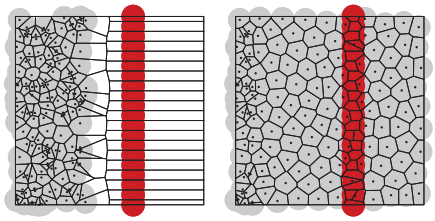
traversed distance of all approaches is a common behavior of mobile sensors deployment algorithms since, when few sensors are available, all sensors move in order to contribute to the achievement of the final coverage. Instead, when more sensors are available, the average traversed distance decreases, since only sensors detecting a coverage hole are allowed to move.

Figure 5(c) shows the average number of moving actions. This is an important metric to evaluate mobile sensor deployment algorithms, since a sensor consumes a high amount of energy to start and stop a movement. Similar considerations with respect to the traversed distance and the peaks in the Figures discussed above can be made. SecureVor introduces a small overhead in terms of number of movements due to the reduced traversed distance per round which results in an higher number of movements to traverse the same distance.

We now show results related to sensor energy consumption. We adopt the energy cost model commonly used in the literature for mobile sensors [3], [26], [1]. In particular, receiving a message costs 1 energy units (eu), sending a message 1.125eu, traversing one meter costs 300eu and starting/stopping a movement costs as one meter of movement. We consider a cumulative energy consumption metric which takes into account all the above contributions.

Figure 5(d) shows the obtained results. All algorithms incur in a higher communication cost as the sensor density increases. Such an overhead is higher under SecureVor because of the additional messages required to communicate the trusted neighbor set. The energy consumption under VOR-Free is 43% less energy with respect to SecureVor.

The termination time is shown in 5(e). SecureVor shows a shorter termination time with respect to VOR-Free. This is due to the shorter maximum traversed distance of SecureVor which allows shorter movements that are forbidden by VOR. As a result, under VOR sensors move only when a long movement is possible, thus resulting in cascade movements which lengthens the termination time. On the contrary, shorter movements enable sensors to move more in parallel, resulting in a lower termination time for SecureVor.



(a) (b)

Fig. 6. Scenario B: Initial deployment (a), and final deployment under SSD (b).

In order to further study the performance of the considered algorithms, we performed some experiments by setting the number of legitimate sensors to 140 and by increasing the number of malicious sensors from 0 to 30. Figure 5(f) shows the achieved coverage. The vertical line represents the minimum number of malicious sensors for which the distance d between them is less than $\sqrt{3}R_s$. As proven in Theorems 3.1 and 3.2, legitimate sensors are not able to cross the barrier if d is less than or equal to such a value. These experiments show that legitimate sensors do not cross the barrier even when a small number of malicious sensors is present. SecureVor is not affected by the number of malicious sensors deployed, since legitimate sensors are able to detect malicious sensors and cover the AoI.

7.2 Scenario B

In this section we consider a setting for which SSD is designed, that is where $4R_{\mathrm{s}} > R_{\mathrm{tx}} > 2R_{\mathrm{s}}$. In particular, we set $R_{\mathrm{tx}} = 12\mathrm{m}$, and $R_{\mathrm{s}} = 5\mathrm{m}$. Therefore, the maximum moving distance d_{max} is 1m.

Similar to the previous scenario, we study the performance of SSD in presence of the BOM attack performed by 20 malicious sensors and by increasing the number of legitimate sensors deployed. We compare the performance of SSD to the original VOR algorithm and with the same algorithm in absence of the attack (VOR-Free).

Figure 6(a) show an instance of this scenario with 150 legitimate sensors. Figure 6(b) shows the final deployment achieved by SSD. Even with limited transmission radius, legitimate sensors are able to detect malicious nodes thanks to position swaps, and ultimately achieve full coverage.

Figure 7(a) shows the coverage of the AoI achieved by the considered approaches. VOR achieves similar results as in the previous scenario, with legitimate sensors unable to cross the barrier. On the contrary, SSD successfully defeats the attack and enables legitimate sensors to cover the AoI, achieving the same coverage of VOR-Free. Similarly to the previous scenario, we do not discuss the performance of VOR in the following.

Figure 7(b) shows the average distance traversed by sensors. The figure evidences the additional traversed distance of SSD with respect to VOR-Free, due to the position swaps necessary to detect malicious sensors. As mentioned for Scenario A, the peak in the traversed distance occurs in correspondence to the minimum number of legitimate sensor necessary to achieve full coverage.

Figure 7(c) shows the average number of start and stop actions. SSD shows a lower number of starts and stops with respect to VOR-Free. This apparently surprising result is due to the swap activity. In particular, when a legitimate sensor swaps with a malicious sensor, the malicious sensor is detected and the legitimate sensor does not move back to its original position. As a result, such sensor performed a longer movement, whose length is not limited by the parameter $d_{\max}$, nor it is affected by small local position adjustments.

Figure 7(d) shows the overall consumed energy. Such a measure includes both the communication and the movement costs. SSD performs better than VOR-Free in this case, thanks to the fewer number of start and stop actions, that dominate the energy consumption.

The termination time is shown in Figure 7(e). The results detailed in this figure reveal a longer termination time of SSD compared to VOR-Free. This is due to the longer round length required to include swap activities during every iteration of the algorithm. Nevertheless, this increased termination time allows SSD to defeat the attack, even in the restricted case of the communication radius.

We finally show in Figure 7(f) the average number of swaps per sensor under SSD. The number is relatively low, with a peak of eight swaps when the number of sensors deployed is close to the minimum to achieve full coverage. As the number of sensor increases, the number of swaps rapidly decreases.

We performed additional experiments varying the number of malicious sensors. We do not show them in the paper as they look very similar to those obtained for SecureVor and detailed in Figure 5 (d). These results confirm that the performance of SSD also, is not significantly affected by the number of malicious sensors.

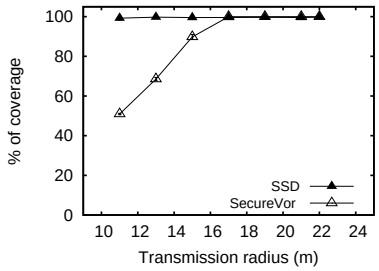


Fig. 8. Scenario C: coverage of the AoI (a).

7.3 Scenario C

In this section we perform a sensitivity analysis to compare SecureVor and SSD under various settings of the transmission radius. We recall that SecureVor assumes that $R_{\mathtt{tx}} > 4R_{\mathtt{s}}$, while SSD is designed for the more restricted scenario in which $4R_{\mathtt{s}} > R_{\mathtt{tx}} > 2R_{\mathtt{s}}$. In these experiments we increase $R_{\mathtt{tx}}$ from the setting of SSD to the setting of SecureVor, and compare the performance of the algorithms.

To enable SecureVor to work even when $R_{\mathtt{tx}} < 4R_{\mathtt{s}}$, we define a *virtual sensing radius* $R_{\mathtt{vs}}$, for which $R_{\mathtt{tx}} > 4R_{\mathtt{vs}}$. Using this modification, legitimate sensors deploy as if the sensing radius were the virtual radius $R_{\mathtt{vs}}$. This allows legitimate sensor to detect malicious sensors. However the drawback of this setting is a denser deployment, so more sensors are needed to achieve full coverage.

In this experimental scenario, we consider 200 sensors with sensing radius $R_{\mathtt{s}} = 5$m, while we let the transmission radius $R_{\mathtt{tx}}$ vary from 11 meters up to 22 meters. As the maximum allowed distance for SecureVor depends on the virtual radius according to the equation $d_{\max} = \frac{R_{\mathtt{tx}}}{4} - R_{\mathtt{vs}}$, we fix the maximum moving distance $d_{\max}$ for SecureVor to 0.5m, and let the value of $R_{\mathtt{vs}}$ grow according to the equation $d_{\max} = \frac{R_{\mathtt{tx}}}{4} - R_{\mathtt{vs}}$. Therefore $R_{\mathtt{vs}} = \min\left\{R_{\mathtt{s}}, (\frac{R_{\mathtt{tx}}}{4} - d_{\max})\right\}$. Under such a setting, when $R_{\mathtt{tx}}$ spans from 11 meters to 22 meters, $R_{\mathtt{vs}}$ correspondently grows from 2.25 meters to 5 meters. A further increase in $R_{\mathtt{tx}}$ would not cause any increase in the virtual radius, which would be the same as the real sensing radius. This last setting is what SecureVor requires to work at its best, deploying sensors at the density required by VOR.

Figure 8, shows the coverage achieved by the two algorithms when the transmission radius $R_{\mathtt{tx}}$ increases. As we can see, SSD always reaches full coverage of the AoI, independently of the setting of $R_{\mathtt{tx}}$. By contrast, SecureVor is unable to complete the coverage when working with transmission radius lower than 17m, because the corresponding virtual sensing radius is too short to cover the area with only 200 sensors. These results highlight the benefit of using SSD when the assumptions required of SecureVor are not met by the available hardware. Although SecureVor can be used with minor modifications, its performance can be significantly penalized. Due to space limitations, we omit the results of the other performance metrics, which however show similar trends to those shown in Figure 5.

7.4 Scenario D

The last set of experiments introduces a more complex attack, in which malicious sensors initially form a barrier, and then start moving towards legitimate sensors. Malicious

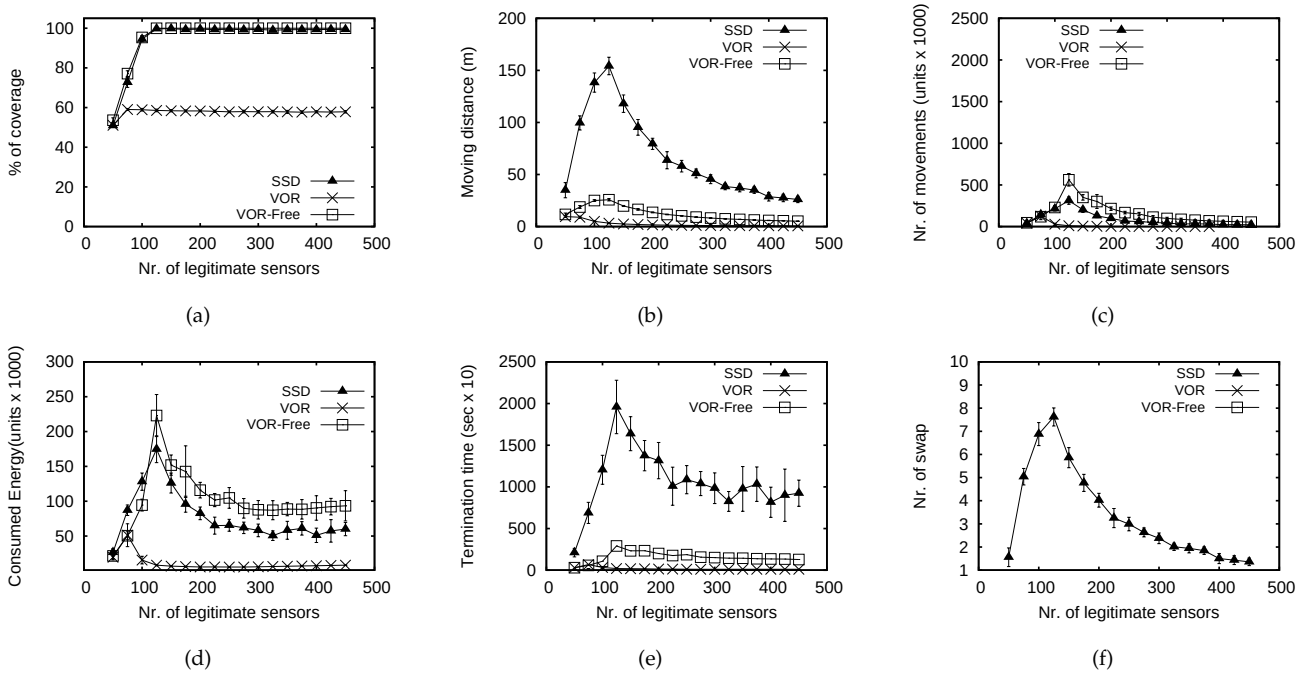


Fig. 7. Scenario B: coverage of the AoI (a), traversed distance (b), number of movements (c), consumed energy (d), termination time (e), average number of swaps (f).

movements are perpendicular to the barrier and are of length $d_{\max}$, according to the algorithm rules. A malicious sensor never breaks the barrier when moving, therefore it only moves if it can maintain a distance lower than $\sqrt{3}R_s$ with neighbor malicious sensors. A malicious sensor stops moving as soon as it reaches a distance lower than $2R_s$ from at least one legitimate sensor. In this setting we use a setting of transmission and sensing radius suitable for both SecureVor and SSD.

Figure 9(a) shows the initial deployment, with 150 legitimate sensors and a moving barrier of 20 malicious sensors. This attack can severely compromise the coverage provided by legitimate sensors under VOR, which in fact terminates the execution as shown in Figure 9(b). Malicious sensors successfully confine legitimate sensors in a small portion of the AoI. Legitimate sensors do not cross the barrier because the mutual distance between malicious sensors is always lower than $\sqrt{3}R_s$, which also confirms the theoretical results described in Section 3.1.

Under SecureVor legitimate sensors discover the malicious movements resulting from the barrier movement. Consequently, legitimate sensors are able to detect and ignore malicious sensors, and achieve full coverage as shown in Figure 9(c). SSD requires a minor modification to work under the attack of a dynamic barrier, with particular focus on the concept of *vertex neighbor*. When a legitimate sensor approaches the barrier, the new vertex can be located in a circular corona of the boundary, and not exactly on the boundary, to take into account possible movements of the barrier sensors. With such a modification, SSD successfully lets legitimate sensors discover barrier sensors and ignore them, achieving full coverage as shown in Figure 9(d).

We conducted additional experiments with more complex configurations, such as multiple barriers or barriers of various irregular shapes. Results show that both SecureVor and SSD are able to defeat such attacks.

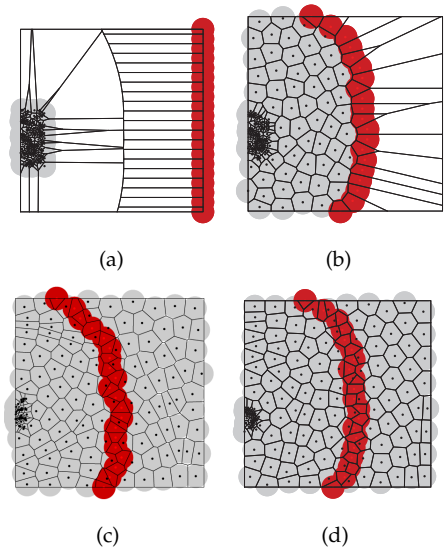


Fig. 9. Scenario D. Dynamic barrier: initial deployment (a), final deployment with VOR (b), SecureVor (c) and SSD (d).

8 Conclusions

We addressed the vulnerabilities of one of the most acknowledged approaches to mobile sensor deployment: the Voronoi based approach. We consider a recently proposed attack to mobile sensor networks, the OM attack, and characterize the geometric conditions under which such an attack is effective when the network adopts the Voronoi approach to deployment.

We propose two algorithms called SecureVor and Secure Swap Deployment (SSD) to counteract the OM attack. The algorithms work in complementary operative settings. Both allow legitimate sensors to determine the malicious nature

of their neighbors by observing their movements. We formally prove that SecureVor is able to defeat the OM attack, and that both SecureVor and SSD have a guaranteed termination. Additionally, we performed an extensive experimental analysis that confirmed that with these algorithms the network achieves its monitoring goals even in the presence of an attack, at the expense of a small overhead in terms of movements and deployment time.

REFERENCES

[1] N. Bartolini, G. Bongiovanni, T. La Porta, and S. Silvestri. On the security vulnerabilities of the virtual force approach to mobile sensor deployment. *IEEE INFOCOM*, 2013.

[2] N. Bartolini, G. Bongiovanni, T. La Porta, and S. Silvestri. On the vulnerabilities of the virtual force approach to mobile sensor deployment. *IEEE Trans. on Mobile Computing*, 13(11):2592–2605, 2014.

[3] N. Bartolini, T. Calamoneri, E. G. Fusco, A. Massini, and S. Silvestri. Push & pull: autonomous deployment of mobile sensors for a complete coverage. *Wireless Networks*, 16(3):607–625, 2010.

[4] N. Bartolini, T. Calamoneri, T. La Porta, and S. Silvestri. Autonomous deployment of heterogeneous mobile sensors. *IEEE Trans. on Mobile Computing*, 10(6):753 –766, 2011.

[5] N. Bartolini, T. La Porta, S. Silvestri, and F. Vincenti. Voronoi-based deployment of mobile sensors in the face of adversaries. *IEEE ICC*, 2014.

[6] T. Beran, R. B. Langley, S. B. Bisnath, and L. Serrano. High-accuracy point positioning with low-cost gps receivers. *Navigation*, 54(1):53–63, 2007.

[7] S. Capkun and J.-P. Hubaux. Secure positioning of wireless devices with application to sensor networks. *IEEE INFOCOM 2005*, 3:1917–1928, 2005.

[8] S. Capkun, K. Rasmussen, M. Cagalj, and M. Srivastava. Secure location verification with hidden and mobile base stations. *IEEE Trans. on Mobile Computing*, 7(4):470–483, 2008.

[9] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. *IEEE Symposium on Security and Privacy*, 2003.

[10] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. Computational geometry: Algorithms and applications. *Springer-Verlag Berlin Heidelberg*, 2008.

[11] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. *IEEE INFOCOM*, 2004.

[12] P. K. Dutta, A. K. Arora, and S. B. Bibyk. Towards radar-enabled sensor networks. *ACM IPSN*, pages 467–474, 2006.

[13] R. J. Fontana, E. Richley, and J. Barney. Commercialization of an ultra wideband precision asset location system. *IEEE UWST*, pages 369–373, 2003.

[14] M. Garetto, M. Gribaudo, C.-F. Chiasserini, and E. Leonardi. A distributed sensor relocation scheme for environmental control. *IEEE MASS*, 2007.

[15] N. Heo and P. Varshney. Energy-efficient deployment of intelligent mobile sensor networks. *IEEE Trans. on Syst., Man and Cyb.*, 35(1):78–92, 2005.

[16] B. Lagesse, M. Kumar, and M. Wright. Arex: An adaptive system for secure resource access in mobile p2p systems. In *IEEE P2P*, pages 43–52, 2008.

[17] M. Lam and Y. Liu. Two distributed algorithms for heterogeneous sensor network deployment towards maximum coverage. *IEEE ICRA*, pages 3296–3301, 2008.

[18] K. Liu, N. Abu-Ghazaleh, and K. Kang. Location verification and trust management for resilient geographic routing. *Elsevier JPDC*, 67(2):215–228, 2007.

[19] K. Ma, Y. Zhang, and W. Trappe. Managing the mobility of a mobile sensor network using network dynamics. *IEEE Trans. on Paral. and Distr. Syst.*, 19(1):106–120, 2008.

[20] M. Ma and Y. Yang. Adaptive triangular deployment algorithm for unattended mobile sensor networks. *IEEE Trans. on Computers*, 56(7):946–847, 2007.

[21] Memsic. Mts420/400 datasheet.

[22] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. *ACM IPSN*, 2004.

[23] G. Sibley, M. Rahimi, and G. Sukhatme. Mobile robot platform for large-scale sensor networks. *IEEE ICRA*, pages 1143–1148, 2002.

[24] K. Walsh and E. G. Sirer. Experience with an object reputation system for peer-to-peer filesharing. In *USENIX NSDI*.

[25] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. Shantz. Energy analysis of public-key cryptography for wireless sensor networks. *IEEE PerCom*, 2005.

[26] G. Wang, G. Cao, and T. La Porta. Movement-assisted sensor deployment. *IEEE Trans. on Mobile Computing*, 5(6):640–652, 2006.

[27] Y.-C. Wang, C.-C. Hu, and Y.-C. Tseng. Efficient placement and dispatch of sensors in a wireless sensor network. *IEEE Trans. on Mobile Computing*, 7(2):262–274, 2008.

[28] G. Yan, S. Olariu, and M. C. Weigle. Providing vanet security through active position detection. *Elsevier Computer Communications*, 31(12):2883–2897, 2008.

[29] Y. Zeng, J. Cao, S. Zhang, S. Guo, and L. Xie. Random-walk based approach to detect clone attacks in wireless sensor networks. *IEEE Selected Areas in Communications*, 28(5):677–691, 2010.

[30] Y. Zou and K. Chakrabarty. Sensor deployment and target localization in distributed sensor networks. *ACM Trans. on Emb. Comp. Syst.*, 3(1):61–91, 2003.

[31] Opnet technologies inc. *http://www.opnet.com*.

Novella Bartolini graduated with honors in 1997 and received her PhD in computer engineering in 2001 from the University of Rome, Italy. She is now Associate Professor at the University of Rome and visiting professor at Penn State University, since 2014. She was chaired several international conferences, and has served on the editorial board of Elsevier Computer Networks and ACM/Springer Wireless Networks. Her research interests lie in the area of wireless networks and network management.

Stefano Ciavarella graduated with honors in computer science at Sapienza University of Rome, Italy. He is a Phd Student in Computer Science at Sapienza University of Rome. He is visiting scholar at the Computer Science Department of Missoury University of Science and Technology. His research interests lie in the area of network management, wireless sensor networks, emergency scenarios in cyber-physical systems, and smart grid security.

Simone Silvestri graduated with honors and received his PhD in computer science at Sapienza University of Rome, Italy. He is now an Assistant Professor at the Computer Science Department of Missouri University of Science and Technology. He serves as program committee member of several international conferences. His research interests lie in the area of network management, hybrid wireless sensor networks, interdependent cyber-physical systems, and smart grid security.

Thomas F. La Porta is the WIliam E Leonhard Chair Professor in the department of computer science and engineering at Penn State University. He joined Penn State in 2002. Dr. La Porta is the Director of the Institute for Networking and Security Research. He is an IEEE Fellow, Bell Labs Fellow and he also won Thomas Alva Edison Patent Awards in 2005 and 2009. His research interests include mobility management, wireless networks, mobile data and sensor systems, and network security.