Sacrificial Node-Assisted Defense against

Search-based Physical Attacks in Sensor Networks

Xun Wang, Wenjun Gu, Sriram Chellappan, Dong Xuan and Ten H. Lai

Abstract

In this paper we study the defense of sensor networks against Search-based Physical Attacks. In search-based physical attacks, the attacker walks through the sensor network using appropriate signal detecting equipment to locate active sensors, and then physically destroys them. We design an effective defense approach to defend sensor networks against such attacks. The core principle of our defense is to trade short term local coverage for long term global coverage through appropriate attack notification and state switching of sensors. The effectiveness of attack notification and state switching relies on the existence of Sacrificial Nodes. Sacrificial nodes are sensors which detect the attacker in order to protect other sensors at the risk of themselves being detected and destroyed by the attacker. To study the effects of both search-based physical attacks and our defense approach, we define a novel performance metric called Accumulative Coverage (AC), which effectively captures coverage and lifetime of the sensor network. Our performance data clearly demonstrate that search-based physical attacks cause a significant deterioration in AC. However, our performance data show that our defense approach can effectively maintain AC even under intense search-based physical attacks. To the best of our knowledge, ours is the first work that identifies the problem of search-based physical attacks in sensor networks and proposes defenses against them. We strongly believe that the viability of sensor networks in the future is contingent on their ability to resist physical attacks, which is the core of our work here.
I. INTRODUCTION

Recently, wireless sensor networks have received significant attention from the research community. An important aspect of research in this area is the security of sensor networks. In this realm, many possible attacks to sensor networks, and defense schemes against such attacks have been proposed [1], [2], [3], [4], [5], [6], [7], [8]. With many new applications, including defense related, mission critical, emergency services envisaged in the near future, sensor network security is becoming very important.

The small form factor of sensors, coupled with the unattended and distributed nature of their deployment expose sensor networks to a special class of attacks that could result in the physical destruction of sensors. We define Physical Attacks as those that result in the physical destruction of sensors and render them permanently non-functional. The significance of studying physical attacks comes from the following factors. Physical attacks are inevitable threats in sensor networks. They are relatively simple to launch for the attacker and extremely destructive to the sensor networks. In the simplest case, the attacker can just drive a vehicle in the sensor field or hurl grenades/bombs in the field and destroy sensors. A smarter attacker can stealthily move in the network to detect sensors and destroy only the sensors. In any case, the end result of physical attacks can be quite fatal. The backbone of the network (the sensors themselves) is destroyed. Destruction of sensors may also result in the violation of the network paradigms. This could be the topology, routing structure, power management etc. As such, a wide spectrum of impacts may result due to physical attacks and when left unaddressed, physical attacks have the potential to render the entire sensor network mission useless.

In this paper, we first define a class of physical attacks called Search-based Physical Attacks. In our search-based physical attack model, a mobile attacker walks in the network to search for sensors and then destroys them. The searching process is executed by means of detecting electronic, magnetic, heat signals emitted by the sensors. Once sensors are identified, the attacker physically destroys the sensors. Another type of physical attacks that we have modeled in [9] is called Blind physical attacks, where the attacker blindly attacks the sensor network using brute-force methods like grenades/bombs, tanks, etc. However, blind physical attacks will result not only in destruction of sensors, but will also cause casualties...
to the deployment field. In cases where the attacker cannot tolerate casualties to the deployment field (like airports, oil fields, battlefields etc. of the attacker side), search-based physical attacks will be the preferred method.

We then propose a distributed Sacrificial Node-assisted defense protocol to defend sensor networks against search-based physical attacks. The core principle of our defense is to trade short term local coverage for long term global coverage through the sacrificial node-assisted attack notification and state switching of sensors. A sacrificial node is one that detects the attacker in order to protect other sensors at the risk of itself being detected and destroyed by the attacker. Our rationale is to select certain nodes in the network as sacrificial nodes, in order to compensate for the weakness of sensors’ ability in detecting the attacker and protecting themselves. Intuitively, a sensor is preferable to be chosen as a sacrificial node if it can protect more sensors from being detected by the attacker. We propose a utility function based scheme to determine sacrificial nodes. The utility quantifies the preference of each sensor to be a sacrificial node. Our performance metric in this paper is Accumulative Coverage ($AC$) of the network. $AC$ captures both the lifetime and coverage, and as such is an effective metric to measure performance. Our performance data clearly show that search-based physical attacks dramatically reduce $AC$. However, with our defense mechanism in place, the accumulative coverage can be improved significantly even under intense attacks.

Physical attacks are patent and potent threats to sensor networks. We believe that the viability of future sensor networks is contingent on their ability to resist physical attacks. As such, our work is an important step in this regard. The defense strategy we propose can effectively defend sensor networks against search-based physical attacks. The rest of the paper is organized as follows. In Section II, we discuss the attack model. Section III describes our defense protocol and Section IV reports the performance evaluation. We present related work in Section V, and conclude our work in Section VI.
II. Search-based Physical Attacks

A. Sensor Detection Mechanism

The attacker in search-based physical attacks can be a human being, a remote controlled mobile agent, or a programmed robot. The attacker is equipped with signal detection equipment to locate sensors. Once a sensor location is identified, the attacker can move to the particular location and destroy the sensor. The destruction can be accomplished using circuit/hardware tampering, heat/radiation emission or simply applying physical force to the sensors. In search-based physical attacks, the basic method used by the attacker to identify sensors is to detect signals emitted by the sensors. We classify signals emitted by sensors into two types. Passive signals include heat, vibration, magnetic signals that are part of the physical characteristics of the sensors. In our model, the attacker cannot visually detect sensors. Active signals on the other hand include communication messages, beacons, query messages that are part of normal network communications. These two signal types are quite different from the perspective of attacker’s detection. Passive signals propagate small ranges, and the detection of them enables the attacker to accurately detect the location of their source (the sensor emitting the passive signals). Active signals can propagate longer distance, but the attacker can only isolate the location of the source of an active signal within an area. We denote this area as the sweeping area. Obviously, when a sensor is closer to the attacker, a stronger active signal is detected by the attacker. Consequently, the isolation is more accurate, and the sweeping area is smaller for the sensor. We use $R_{ps}$ and $R_{as}$ to denote the maximal distances within which the attacker can detect passive and active signals respectively. Thus, $R_{ps} < R_{as}$. In our model, if the attacker detects multiple sensors, it can store their locations in memory. While the attacker can detect multiple sensors, it can only choose and proceed to destroy one sensor at a time. We call the detected sensor that the attacker currently chooses to destroy as target.

When the attacker isolates one area for the source sensor of a received active signal, there is the issue of isolation accuracy. Once a signal from a sensor (say $s_i$) is detected, the attacker will attempt to locate the sensor ($s_i$). To do this, the attacker needs to estimate the distance from its current position to sensor $s_i$, referred to as $d_i$, and the orientation or the arrival angle of the detected signal [10], [11]. Approaches
discussed in [10], [11] can be used for this purpose. The attacker can only isolate the sensor's location within a certain area because its estimation of the sensor's location is not accurate. We define $r_i$ as the maximum distance between actual location of sensor $s_i$ and its location estimated by the attacker. The area isolated is a circle with radius $r_i$ and the center of it is the estimated location of the detected sensor as shown in Fig. 1. In order to determine $r_i$, the attacker will make use of the detection error, $\theta$, in the detection of the orientation of the received signal [10], [11] and the estimated distance of the signal source (a sensor) as follows,

$$r_i = d_i \cdot \sin(\theta).$$  \hspace{1cm} (1)

We can see that the accuracy of the attacker in determining the sensor's location is inversely proportional to $\theta$. That is, if $\theta$ is small, the accuracy is higher ($r_i$ is small). Similarly, a large $\theta$ means larger $r_i$, which consequently means less accurate isolation. The term $\theta$ measures the detection accuracy of the attacker. We call the area ($= \pi r_i^2$) as the sweeping area for sensor $S_i$. The attacker will proceed to sweep this area if it wants to destroy this sensor. Since the maximal distance that the attacker can detect the active signal of the sensor is $R_{as}$, the maximal $d_i$ is $R_{as}$. Thus the maximal radius of isolation area of one active signal is $R_{as} \cdot \sin(\theta)$.

Fig. 1. The sweeping area of the detected sensor.

The ability of the attacker to detect a sensor also depends on the state of the sensor. A sensor is Destroyed if it has been physically destroyed by an attacker. Otherwise it is Alive. In our model, a sensor that is Alive can be in one of the three states: sleeping, sensing or sending state. A sensor can voluntarily turn itself off and be in the sleeping state. In this state, the sensor emits no signal and hence cannot be detected.
by the attacker (even if minute signals are emitted while sleeping, we assume they are imperceptible to
the attacker). A sensor in the sensing state carries out only sensing tasks, without sending out any active
signal. The signals emitted during sensing are just passive signals. A sensor in the sending state emits
both passive and active signals. We call a sensor Active if it is in sensing or sending state. An active
sensor can be detected by the attacker via the signals emitted by the sensor. A sensor can instantaneously
switch among these three states at will as long as it is alive.

B. The Search-based Physical Attack Model

Model 1 describes our search-based physical attack model. This model describes the attacker’s response
to different events taking place during the attack process. Initially, the attacker does not have any sensor to
destroy. Thus, the attacker performs a random straight line walk in the network field and keeps detecting
passive or active signals. We use $v$ to denote the moving speed of the attacker. In our attack model, if
the attacker reaches the boundary of the network, it is aware of the fact and turns in a suitable direction
in order to once again walk into the network.

\begin{verbatim}
Model 1 Search-based physical attack model
1: Initialization: Target ← ∅; Mem ← ∅;
2: while the attacker is alive do
3:   switch type of event
4:     case detect a sensor $s$ through passive signal:
5:         Target = $s$; Target.type ← passive; Target.location ← Location of $s$;
6:         Target ≠ ∅ AND Target.type = passive:
7:             Add $s$ to Mem;
8:     case detect a sensor $s$ through active signal:
9:         Target = $s$; Target.type ← active; Target.location ← Sweeping area of $s$;
10:        Target ≠ ∅ AND Target.type = active:
11:            Add Target to Mem: Target ← $s$;Target.type ← passive;Target.location ← Location of $s$;
12:        case reach Target.location:
13:            Target.type = passive; Directly destroy Target; Target ← Remove(Mem);
14:            Target.type = active; Sweep the sweeping area of Target; Target ← Remove(Mem);
15:        default:
16:            Whenever Target ≠ ∅, walk towards Target.location, otherwise perform a random straight line walk;
17:   endswitch
18: end while
\end{verbatim}

Once the attacker detects a signal from a sensor, it first checks the type of the signal. Case 1: If it is a
passive signal, the attacker first estimates the location of the source of the signal. If the attacker currently has no target, it then sets the sensor that emitted this signal as the target and walks towards it. Otherwise, if the attacker already has a target which was detected through a passive signal, it immediately puts the source of this signal into memory. If the attacker has a current target detected through an active signal, the attacker puts the current target into memory and sets the newly detected sensor (through a passive signal) to be the target. Case 2: If it is an active signal, the attacker identifies the sweeping area and put it into memory. If the attacker has no target when this active signal is detected, the attacker sets the sensor that emitted this signal to be the current target and walks towards it. If the attacker already has a target, it will put this newly detected sensor and the corresponding sweeping area into memory. In our model, the attacker at any point in time can have only one sensor as a target to destroy. Multiple detected sensors/sweeping areas can be put into memory for future targets. We denote the memory size as $M$.

Once the attacker reaches the target, the attacker will destroy it. If the target was detected by a passive signal, the attacker will destroy the sensor directly. If only an active signal was used for detecting the target, the attacker sweeps the sweeping area, thereby destroying all alive sensors in that area including sensors in sending, sensing and sleeping states. In our model, the attacker keeps detecting sensor signals during sweeping. After destroying the target, if the attacker has sensors in memory, it will pick the closest sensor detected by a passive signal as the next target if there is one. Otherwise, it chooses the closest sensor detected by active signals as the next target. If the memory content is empty, the attacker does a random straight line walk to search for sensors.

C. Discussions

Our search-based attack model presented here is quite representative. The philosophy of the attacker is to reach and destroy closest sensors first. Our model can be extended to represent a wide spectrum of physical attacks. If there is no search process, the attacker can just use random sweeping to destroy sensors. This is similar to brute force blind attacks that we modeled in [9]. Another special case is for the attacker to destroy only specific sensors among many detected sensors. Such sensors can be cluster
heads, data aggregators etc. In some cases, the attacker can aim to destroy the functionality of the sensor network by partitioning the sensor network. Such an attack may be hard to conduct, as the attacker needs a priori knowledge of the topology, communication pattern and range of sensors. In our current model, we assume there is only one attacker. However, it can be easily extended to multiple attackers if there is no cooperation among the attackers. Modeling multiple attackers with cooperation among them is part of our ongoing work. In this paper, we assume the attacker does not have the capability to reach or destroy the base station.

III. Defending against Search-based Physical Attacks

A. Design Rationale

The primary success criterion of the attacker in conducting search-based physical attacks is the coverage loss as a result of the destroyed sensors. Thus, the goal of our defense is to maintain network performance in terms of coverage under attacks. In our defense protocol, we assume that the active sensors can detect the attacker and notify its neighbors before being destroyed (We will discuss the case where these assumptions do not hold in Section III-F). The detection can be achieved by detecting signals emitted by the attacker (motion, electromagnetic etc) [12]. However, the detecting ability of the sensors is less powerful than that of the attacker. Therefore, the active sensors may only detect the attacker after they have been detected by the attacker. In this paper, we do not assume that the sensors have any knowledge about the attacker, including its speed $v$, memory size $M$, signal detection ranges $R_{as}$ and $R_{ps}$.

Our defense objective is to maximize network performance in terms of coverage under search-based physical attacks. We propose a sacrificial node-assisted approach, in which some sensors in the detection range of the attacker choose to stay in the sending state even if they are aware of the proximity of the attacker. These sensors could have switched to sensing/sleeping states to protect themselves from being detected, but they stay in the sending state to sacrifice themselves and protect other sensors.

Fig. 2 illustrates the basic rationale of our defense protocol. Sensor $s_1$ detects the attacker and notifies its neighbors, including sensors $s_2$, $s_3$ and $s_4$, but sensors $s_5$, $s_6$ and $s_7$ are not aware of the proximity of
the attacker. If sensors $s_2$, $s_3$ and $s_4$ switch to sensing/sleeping states and the attacker chooses to move to the right after it destroys sensor $s_1$, sensors $s_5$, $s_6$ and $s_7$ are at risk of being detected. In this situation, it is important for sensor $s_2$ to stay in the sending state so that it can notify sensors $s_5$, $s_6$ and $s_7$ before they are detected. Sensor $s_2$ could have protected itself by switching to the sensing or sleeping state, but its sacrifice helps to protect many other sensors, especially when the density in the shaded area is relatively high. We call sensor $s_2$ a \textit{sacrificial node}. We define a sacrificial node as one that stays in sending state and detects the attacker in order to protect other sensors at the risk of itself being detected and destroyed. The challenge is how sensor $s_2$ can decide whether it should be a \textit{sacrificial node} or not. This will be described in detail in Section III-C.

Our \textit{sacrificial node}-assisted approach helps to improve the performance of the network under search-based physical attacks by extending the area in which sensors are aware of the proximity of the attacker. The existence of \textit{sacrificial nodes} compensates for the weakness of sensors in being able to detect the attacker and protect themselves. We \textit{sacrifice} a few sensors to protect more, which is the core principle of our defense protocol. Besides, our approach is localized and distributed in that the defense only involves the sensors in the local area around the attacker and each sensor’s behavior is based on its local information, which makes our approach scalable and efficient.

\textbf{B. Defense Protocol}

In this section, we give a formal description of our defense protocol, followed by an example. The main notations used in this paper are listed in Table I.
TABLE I
NOTATIONS AND DEFINITIONS

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulative Coverage</td>
<td>Accumulative Coverage</td>
<td>$\nu$</td>
<td>Attacker moving speed</td>
</tr>
<tr>
<td>Effective Lifetime</td>
<td>Effective Lifetime</td>
<td>$M$</td>
<td>Attacker memory size</td>
</tr>
<tr>
<td>Coverage($t$)</td>
<td>Network coverage at time $t$</td>
<td>$k_i$</td>
<td>Number of sensors in sensor $s_i$’s protection area</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Network coverage threshold $t$</td>
<td>$k_i$</td>
<td>Number of unprotected sensors among $k_i$</td>
</tr>
<tr>
<td>$N$</td>
<td>Number of sensors in the network</td>
<td>$d(i,j)$</td>
<td>Distance between sensor $s_i$ and sensor $s_j$</td>
</tr>
<tr>
<td>$S$</td>
<td>Area of the sensor field</td>
<td>$u(i)$</td>
<td>Utility value of sensor $s_i$</td>
</tr>
<tr>
<td>$f$</td>
<td>Active signal frequency</td>
<td>$u_f(i)$</td>
<td>Contribution of sensor $s_i$ to $u(i)$</td>
</tr>
<tr>
<td>$R_{nt}$</td>
<td>Notification range</td>
<td>$U_{th}$</td>
<td>Utility threshold</td>
</tr>
<tr>
<td>$R_{as}$</td>
<td>Active signal detection range</td>
<td>$U_{ref}$</td>
<td>Reference utility value</td>
</tr>
<tr>
<td>$R_{ps}$</td>
<td>Passive signal detection range</td>
<td>$D(i)$</td>
<td>Timer for SN message of sensor $s_i$</td>
</tr>
<tr>
<td>$R_a$</td>
<td>Sensor’s detection range for attacker</td>
<td>$T_1(i)$</td>
<td>Timer from sleeping to sensing for sensor $s_i$</td>
</tr>
<tr>
<td>$R_s$</td>
<td>Sensor’s sensing range for coverage</td>
<td>$T_2(i)$</td>
<td>Timer from sleeping to sending for sensor $s_i$</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Active signal direction detection error</td>
<td>$T_3(i)$</td>
<td>Timer from sensing to sending for sensor $s_i$</td>
</tr>
</tbody>
</table>

1) Protocol Description: The protocol is executed by individual sensors switching among different states triggered by events, as shown in Procedure 2. In the defense protocol, we separate the sending state into SENDING, SENDING_AN_SENT, SENDING_SN_SENT states, and separate the sensing state into SENSING, SENSING_SN RECEIVED states.

Let us first define each of the above states. SENDING_AN_SENT is a type of sending state, in which the sensor has sent out one or more attack notification (AN) messages to announce the approaching of the attacker. SENDING_SN_SENT is a type of sending state, in which the sensor is a sacrificial node and it has sent out a sacrificial node notification (SN) message to inform nearby sensors of itself being a sacrificial node. For a sensor that is not in any of above two specific types of sending state, we simply denote it as being in SENDING state. For a sensor that is in sensing state, if it has received a SN message but has not received the corresponding AN message, it is in SENDING_SN RECEIVED state, otherwise it is in SENSING state. We define DOWN state as one in which the sensor has been destroyed. There is one other issue that needs to be discussed prior to the defense protocol procedure. Each sensor in our model has a unique ID. The AN message contains the ID of the sender. The SN message contains the IDs of both the sender of the SN message and the sender of the corresponding AN message, which triggers the sending of the SN message.

At the beginning, an active sensor detects the attacker (event 1). It sends out an AN message and stays in
Procedure 2: Defense Protocol Procedure

1: Initialization: \( \text{state} \leftarrow \text{SENDING or SENSING or SLEEPING}; \text{last\_AN\_sender\_JD} \leftarrow \text{NULL}; \)
2: while \( \text{state} \neq \text{DOWN} \) do
3: switch type of event
4: case detect the attacker: // event 1
5: Broadcast an AN message; \( \text{state} \leftarrow \text{SENDING\_AN\_SENT}; \)
6: case receive an AN message: // event 2
7: if \( \text{state} = \text{SENDING or SENDING\_SN\_SENT or SENSING or SENDING\_SN\_RECEIVED} \) then
8: \( \text{last\_AN\_sender\_JD} \leftarrow \text{the ID of the AN message’s sender}; \) Make decision whether to be a sacrificial node;
9: if decide to be sacrificial node then
10: // event 2.1
Create a SN message;
SN message’s \( \text{AN\_sender\_JD} \leftarrow \text{last\_AN\_sender\_JD}; \) // in the SN message, record the ID of the corresponding
// AN message’s sender
Broadcast the SN message; \( \text{state} \leftarrow \text{SENDING\_SN\_SENT}; \)
11: else
// event 2.2
Start \( T1 \) timer; Start \( T2 \) timer; \( \text{state} \leftarrow \text{SLEEPING}; \)
12: end if
13: end if
14: case receive an SN message: // event 3
15: if \( \text{state} = \text{SENDING or SENDING\_SN\_SENT or SENSING or SENDING\_SN\_RECEIVED} \) then
16: // check whether received the corresponding AN message
17: if \( \text{last\_AN\_sender\_JD} \neq \text{the\_AN\_sender\_JD} \) in the SN message then
18: Start \( T3 \) timer; \( \text{state} \leftarrow \text{SENSING\_SN\_RECEIVED}; \)
19: end if
20: end if
21: if \( T1 \) expires: \( \text{state} \leftarrow \text{SENSING}; \) // event 4
22: if \( T2 \) or \( T3 \) expires: \( \text{state} \leftarrow \text{SENDING}; \) // event 5
23: case destroyed by the attacker: \( \text{state} \leftarrow \text{DOWN}; \) // event 6
24: end switch
25: end while

the \( \text{SENDING\_AN\_SENT} \) state. The active sensors receiving the AN message will decide whether to be sacrificial nodes or not based on a sacrificial nodes determination scheme (discussed later in Section III-C). For the recipient sensors of the AN message that decide to be sacrificial nodes, they will send out sacrificial node notification messages (SN messages) and stay in the \( \text{SENDING\_SN\_SENT} \) state (event 2.1). For other recipient sensors of the AN message that decide not to be sacrificial nodes (called nonsacrificial nodes) they will set two timers, \( T1 \) and \( T2 \), and switch to the sleeping state immediately (event 2.2). These sensors will switch back to \( \text{SENSING} \) and \( \text{SENDING} \) states after \( T1 \) and \( T2 \) expire respectively (event 4 and 5).

For the active sensors that receive a SN message but do not receive the corresponding AN message, they will set a timer \( T3 \) and switch to the \( \text{SENSING\_SN\_RECEIVED} \) state immediately (event 3). In this case, the recipient sensors of this SN message know the approaching of the attacker. However, since
they do not receive the corresponding AN message, the attacker is not very close to them. Hence, they can switch to `SENSING_SN_RECEIVED` state instead of `sleeping` state to avoid un-necessary coverage loss due to early sleeping. While staying in `SENSING_SN_RECEIVED` state, the sensors can avoid being detected by the attacker through active signals. In order to check the corresponding relation between SN and AN messages, the IDs in the messages are checked. If the `AN_sender_ID` in the received SN message is same as the sender ID of the last received AN message (last `AN_sender_ID`), then the last received AN message is the corresponding AN message of the received SN message. The sensors in `SENSING_SN_RECEIVED` state will switch back to the `SENDING` state after `T_3` expires (event 5). Note that, the event of active sensors receive a SN message and the corresponding AN message is already contained in event 2. The sensors that are destroyed by the attacker will be in the `DOWN` state (event 6).

We assume the attacker cannot create AN/SN messages, which could be due to an existing pairwise key scheme [13] among the sensors that the attacker will not be aware of. The description of *sacrificial nodes* determination scheme and the discussion of the timers, `T_1`, `T_2` and `T_3` will be detailed in Section III-C and Section III-D respectively.

2) Example: In the following, we use an example in Fig. 2 to further explain our defense protocol. In the beginning, all sensors are in the sending state as *nonsacrificial nodes*. Suppose at some time, sensor `s_1` in Fig. 2 detects the attacker and sends an AN message to all other sensors in its notification area. The notification message contains the global ID of `s_1` and the notification area is a circle of radius `R_{noti}` centered at `s_1`. We let `R_{noti}` be the same as the sensor communication range. Recall that the attacker is more powerful than sensors in terms of sensing ability. As such, sensor `s_1` will be detected by the attacker before `s_1` has detected the attacker. Thus it is better for `s_1` to send out an AN message instead of switching to the sensing or sleeping state. After sending out the AN message, `s_1` will stay in the sending state.

For the recipients of the AN message sent by `s_1`, which are `s_2`, `s_3` and `s_4`, we assume `s_2` and `s_3` decide to be *sacrificial nodes* while `s_4` does not. Sensors `s_2` and `s_3` will each send out an SN message at different time. In our protocol, we apply a randomized algorithm to let different *sacrificial nodes* send
out SN messages at different time, thus alleviating the problem of message collision, the detail of which is discussed in Section III-C. After $s_2$ and $s_3$ send out the SN messages, they will stay in the sending state as sacrificial nodes. The SN message of $s_2$ contains the global IDs of $s_2$ and $s_1$. The usage of this message is two-fold. First, it is used to update its state information stored in its neighbors, the usage of which will be described in Section III-C. Second, it is used by the sensors in its protection area for state switching, which will be described below. The protection area of a sensor is a circle centered at the sensor with radius $R_{noti}$. On the other hand, $s_4$ will calculate two timers, $T_1$ and $T_2$, and switch to the sleeping state immediately. After $T_1$ and $T_2$ expire, $s_4$ will switch back to sensing and sending (as nonsacrificial node) states respectively.

In Fig. 2, $s_5$, $s_6$ and $s_7$ receive the SN message sent by $s_2$ and $s_3$, but they did not receive the corresponding AN message sent by $s_1$. Each of them will independently set a timer $T_3$ and switch to the sensing state immediately. By doing so, they are protected from being detected via active signals since the attacker may approach them in the near future. However, it may not be preferable for them to switch to the sleeping state for two reasons. First, this will result in a large coverage loss, which is an overkill since the attacker will only choose to move in one direction. Second, they are already in the protection area of $s_2$. They may be notified of the approaching attacker by $s_2$ before their own passive signals are detected. They can then switch to the sleeping state before the attacker can detect them.

C. Sacrificial Nodes Determination

Sacrificial nodes are selected to protect other sensors. Thus, a sensor is more preferable to become a sacrificial node if it can protect more sensors. We use a utility function $u(i)$ to measure the preference for a sensor $s_i$ to be a sacrificial node. Our sacrificial nodes determination is based on this utility function calculated by each sensor.

1) Derivation of Utility Function $u(i)$: Intuitively, a sensor can protect more sensors when there exist more sensors in its protection area, and thus it should have more chance to be a sacrificial node. In Fig. 2, $s_2$ is more preferable than $s_4$ because it can potentially protect more sensors. Thus, a simple utility
function is given by

\[ u(i) = |k_i|, \]  

(2)

where \( k_i \) denotes the set of sensors in the protection area of sensor \( s_i \) and \(|k_i|\) denotes its size.

It may seem obvious that a sensor with high utility value is always preferable to be a sacrificial node. However, if two sensors both have high utility values and they are close to each other, it is not deserved for both of them to be sacrificial nodes. The reason is that the protection areas of both sensors have much overlap. Selecting the second one as a sacrificial node brings little extra benefit. Besides, it incurs more risk since both of them become potential targets now. A reasonable modification is given by

\[ u(i) = |k'_i|, \]  

(3)

where \( k'_i \) denotes the set of sensors that are in the protection area of sensor \( s_i \), but not in the protection area of any other sacrificial node known by \( s_i \) and \(|k'_i|\) denotes its size. If \( s_3 \) in Fig. 2 is a sacrificial node known to sensor \( s_2 \) via an SN message, then \( s_7 \) will not be counted in \( s_2 \)'s utility function.

Furthermore, we observe that in calculating the utility function of a sensor \( s_i \), the relative distances of the sensors in \( k'_i \) from sensor \( s_i \) also make a difference. For instance, In Fig. 2, we assume the attacker moves to the right after it has destroyed \( s_1 \). Compared with \( s_6 \), \( s_5 \) is closer to \( s_2 \) and is more likely to be detected by the attacker before \( s_2 \) sends out an AN message. In this case, the contribution of \( s_5 \) to the utility function of \( s_2 \) should be smaller than that of \( s_6 \). Let us denote \( u_j(i) \) as the contribution of \( s_j \) to \( u(i) \). Therefore, when evaluating \( u_j(i) \), we make \( u_j(i) \) proportional to the distance between \( s_i \) and \( s_j \) (denoted by \( d(i,j) \)) \(^1\). Thus, \( u_j(i) = \frac{d(i,j)}{R_{noti}} \). Then we obtain our final utility function as follows,

\[ u(i) = \sum_{s_j \in k'_i} u_j(i) = \sum_{s_j \in k'_i} \frac{d(i,j)}{R_{noti}}. \]  

(4)

In the ideal situation, where all sensors have full knowledge about the attacker, a sensor \( s_i \) can calculate which of its \(|k'_i|\) neighbors are already detected by the attacker and should not be considered in \( u(i) \). We

\(^1\)The distance between neighboring sensors can be obtained by sensor localization schemes [14], [15], [16].
denote the ideal \( u(i) \) assuming full knowledge of the attacker as \( u^{\text{opt}}(i) \). Theorem 1 states that the utility function in (4) is optimal in terms of minimizing the expected mean square error between \( u(i) \) and \( u^{\text{opt}}(i) \) under the assumption that the sensors have no a priori knowledge of the sensing ability of the attacker.

**Theorem 1:** If \( u(i) \) can be any continuous and differentiable function of \( d(i, j) \), the function \( u(i) = \sum_{s_j \in k'_i} \frac{d(i, j)}{R_{\text{noti}}} \) is optimal in terms of minimizing the expected mean square error between \( u(i) \) and \( u^{\text{opt}}(i) \) under the assumption that the sensors have no a priori knowledge of the sensing ability of the attacker.

**Proof:** We assume the utility function of node \( s_i \) is the sum of \( u_j(i) \) for all \( j \in k'_i \). Therefore, the utility value for \( s_i \) is \( u(i) = \sum_{s_j \in k'_i} u_j(i) \) and the ideal utility value is \( u^{\text{opt}}(i) = u(i|R_{\text{as}}, R_{ps}) = \sum_{s_j \in k'_i} u_j(i|R_{\text{as}}, R_{ps}) \). Denote the sensors in the set \( k'_i \) by \( s_{j_1}, s_{j_2}, \ldots, s_{|k'_i|} \). Without loss of generality, we assume \( 0 \leq d(i, j_1) \leq d(i, j_2) \leq \ldots \leq d(i, j_{|k'_i|}) \leq R_{\text{noti}} \). The expected Mean Square Error (MSE) between \( u(i) \) and \( u^{\text{opt}}(i) \) is,

\[
\text{MSE}(u(i)) = E[(u(i) - u^{\text{opt}}(i))^2] = \frac{1}{R_{\text{noti}}} \int_{R_{ps}=0}^{R_{noti}} [u(i) - u(i|R_{\text{as}}, R_{ps})]^2 dR_{ps}
\]

\[
= \frac{1}{R_{\text{noti}}} \left( \int_{R_{ps}=0}^{d(i, j_1)} \sum_{s_j \in k'_i} u_j(i) - \sum_{s_j \in k'_i} u_j(i|R_{\text{as}}, R_{ps})]^2 dR_{ps} + \int_{R_{ps}=d(i, j_1)}^{d(i, j_2)} \sum_{s_j \in k'_i} u_j(i) - (|k'_i| - 1)]^2 dR_{ps} + \ldots
\]

\[
= \frac{1}{R_{\text{noti}}} \left( \sum_{s_j \in k'_i} u_j(i) - 2 \frac{\alpha}{R_{\text{noti}}} \sum_{s_j \in k'_i} u_j(i) + \frac{\beta}{R_{\text{noti}}} \right).
\]

In the above, \( \alpha \) is given by \( \sum_{s_j \in k'_i} d(i, j) \) and \( \beta \) is given by \( d(i, j_1)(|k'_i| - 1) + d(i, j_2)(2|k'_i| - 3) + \ldots + d(i, j_{|k'_i|}) \cdot 1 \). By deriving the first and second derivatives, the above expected mean square error is minimized when \( u(i) = \sum_{s_j \in k'_i} u_j(i) = \sum_{s_j \in k'_i} \frac{d(i, j)}{R_{\text{noti}}} \). Thus, the theorem holds. \( \blacksquare \)

Theorem 1 shows that the utility function \( u(i) \) is optimal when sensors have no a priori knowledge of the sensing ability of the attacker. This utility function will be used in our defense protocol. In the
following, we introduce the term $U_{ref}$, which is used to obtain the threshold of the utility function in Formula (4) for determining sacrificial nodes and setting the timers of the defense protocol. In (4), if we replace $k_i'$ by a set with size being the average number of neighbors for a sensor and replace the weight $\frac{d(i,j)}{R_{not}}$ by the maximum weight 1, we obtain an approximate upper bound for $u(i)$, which is denoted by $U_{ref}$. The expression of $U_{ref}$ is given by,

$$U_{ref} = \frac{N\pi R^2}{S},$$  

(5)

in which $N$ is the number of sensors in the network and $S$ is the area of network. Since $k_i'$ is a subset of the set of all neighbors of sensor $s_i$, the value of $|k_i'|$ is usually smaller than the average number of neighbors for a sensor. Besides, the weight is no more than 1. Thus, the utility value of a sensor is generally smaller than $U_{ref}$.

2) Sacrificial Nodes Determination Scheme: We now describe the criterion used by a sensor to decide whether it should be a sacrificial node based on its utility value. Intuitively, a sensor that has a high utility value should become a sacrificial node. Thus an empirical threshold $U_{th}$ is necessary here. The sensors whose utility values are above $U_{th}$ will become sacrificial nodes. The value of $U_{th}$ lies in the interval $[0, U_{ref}]$. Similar to the utility function, an ideal utility threshold is impossible to obtain without the knowledge of the attacker information. We will investigate the issue of choosing a reasonable $U_{th}$ in Section IV.

We first discuss sacrificial node determination criterion, and then describe the scheme used by recipient sensors of an AN message for sacrificial nodes determination. Since it is possible that multiple sensors have initial utility values larger than $U_{th}$, we introduce a randomized algorithm here to prevent the collision of SN messages and deal with the problem of protection area overlap. After first calculating the utility function, the sensors whose utility values are smaller than $U_{th}$ will switch to the sleeping state. Other sensors, called candidate sacrificial nodes, will calculate a random delay and set a timer (denoted by
\( D(i) \). It is given by,

\[
D(i) = \begin{cases} 
\epsilon \cdot \Delta t, & u(i) \geq U_{\text{ref}} \\
\Delta t + (1 - \frac{u(i)}{U_{\text{ref}}}) \cdot \Delta t, & U_{\text{th}} \leq u(i) < U_{\text{ref}}
\end{cases}
\]  

(6)

where \( \epsilon \) is a random number uniformly distributed in \([0,1]\) and \( \Delta t \) is an adjustable parameter. Ideally, \( \Delta t \) should be as small as possible to avoid a large delay of SN messages. However it should be comparable to the transmission time of an SN message to avoid collision among different SN messages. A candidate sacrificial node will send out an SN message after its timer expires and then become a sacrificial node. Thus, the sensor with higher utility value generally will send out SN message earlier. After receiving an SN message, a candidate sacrificial node who has not sent out its SN message will cancel its timer and adjust its utility value accordingly by (4). If the new value is less than \( U_{\text{th}} \), it will switch to the sleeping state. Otherwise, it will calculate a new delay and set a timer as above. This process iterates until each recipient sensor of the AN message either becomes a sacrificial node or switches to the sleeping state.

D. State Switching Timers

Recall that the attacker will proceed to destroy other detected sensors in its memory or choose a random direction to move if its memory is empty. The sensors that receive the AN/SN messages cannot accurately predict the movement of the attacker. In the protocol described above, we let the sensors triggered by events 2 and event 3 in Procedure 2 immediately switch to sleeping and sensing states respectively. This could be a conservative scheme. The sensors may switch to sleeping/sensing state too early or even unnecessarily if the attacker never approaches them, but this guarantees they will not be detected by the attacker. Any delay in state switching will definitely incur a risk. On the principle of being conservative, we determine the timers \( T_1(i) \), \( T_2(i) \) and \( T_3(i) \) by,

\[
T_1(i) = T_3(i) = max \{ T, T + (1 - \frac{u(i)}{U_{\text{ref}}}) \cdot T \},
\]

(7)

\[
T_2(i) = 2T_1(i),
\]

(8)
in which, $T$ is an adjustable parameter. We let the sensors switch back to the sensing/sending states at different time. Otherwise, it will incur more risk if the attacker is still nearby when the sensors switch back to the sensing/sending state altogether. Ideally, the value of $T$ depends on the attacker information such as speed, memory content and sensing ability. However, the sensors have no knowledge about this, so they need to be conservative in estimating the value of $T$, which can be based on the knowledge of maximum speed and sensing ability of the attacker.

E. Defense Protocol for Hierarchical Sensor Networks

The above protocol assumes that the sensor network is flat. We now discuss how to extend it in the case of hierarchical sensor networks, where the networks are organized as multiple clusters and there is a cluster-head in each cluster. In hierarchical sensor networks, a cluster-head is responsible for the connectivity of its cluster to other clusters and all the clusters build up the communication backbone of the sensor network [17]. In some cases, the cluster-heads can even be data aggregators for their clusters. In general, the cluster-head is the most important sensor in any cluster, and it is preferable that it is not chosen as a sacrificial node. However, in some specific sensor networks such as those that have multiple cluster-heads per cluster, or where the cluster-head rotation/replacement is very fast, cluster-heads can also be chosen as sacrificial nodes. Hence, in order to make the defense protocol flexible and consider the requirement of different hierarchical sensor networks, we give the cluster-head a degree of priority during the selection of sacrificial nodes. Different sensor networks can assign different priorities to their cluster-heads while determining sacrificial nodes. Correspondingly, Formula (4) need to be modified as,

$$u(i) = \sum_{s_j \in k_i} u_j(i) \cdot p_i = \sum_{s_j \in k_i'} \frac{d(i,j)}{R_{noti}} \cdot p_i,$$

(9)

where $p_i$ is the priority assigned to sensor $s_i$. Normal sensors (that are not cluster heads) and cluster-heads are assigned different priority values. In a hierarchical sensor network, cluster-heads can have the same or different priority values depending on the design of the network. We will investigate the impacts of the priority of cluster-heads on the defense performance in Section IV.
F. Discussions

In our defense protocol, we assume the sensors can detect the attacker. We would like to point out that even if the sensor does not have the ability to detect the attacker remotely, it may still be able to send an AN message just before being destroyed via some hardware triggering mechanism. In the case when the destroyed sensor is not able to send out AN message before destruction, its neighbors can use existing sensor fault detection methods [18], [19] to detect the destroyed sensor and send out AN message for it.

In our protocol, we do not assume that the sensors have a priori knowledge about the attacker information. However, some attacker information such as $v$, $R_{as}$ and $R_{pa}$ may be obtained either by run-time measurements or off-line knowledge. In case this information is known or a good estimation like upper bound is available for the sensors, we can obtain optimal utility threshold $U_{th}$ and optimal timers parameter $T$, which is one of our future work.

IV. Performance Evaluations

A. Performance Metric and Simulation Settings

In order to evaluate the performance of sensor networks under search-based physical attacks, we define a novel metric, namely Accumulative Coverage ($AC$). $AC$ is defined as the integration of the network coverage over the effective lifetime of the sensor network. Network coverage is defined as the percentage of the area in the sensor field that is in the sensing range of at least one active sensor, and effective lifetime is the time period until when the sensor network becomes nonfunctional because the coverage falls below a system required threshold $\alpha$. Denoting $\text{coverage}(t)$ as the network coverage at time $t$, and $EL$, as the effective lifetime, we have,

$$AC = \int_{t=0}^{EL} \text{coverage}(t)dt.$$  \hfill (10)

We believe that $AC$ is an effective metric to measure the performance of a sensor network in many situations since it effectively combines both coverage and lifetime, two of the most important performance metrics in sensor networks. A general metric commonly used in the literature is effective lifetime, which
is defined as the maximum time period during which the coverage is above a certain threshold and thus considers both coverage and lifetime. However, it is not representative enough for situations where for the same effective lifetime, a sensor network with a high coverage can provide more accurate information than one with a lower coverage. Our metric, $AC$ not only considers coverage threshold and lifetime, but is also more representative of real life situations. Thus $AC$ is the basic metric we use to evaluate the performance of a sensor network under search-based physical attacks.

In our simulation, the sensor network area is a $500 \text{ meters} \times 500 \text{ meters}$ square, in which 2000 sensors are randomly uniformly distributed. The active signals are generated following a constant frequency $f$. The signals may collide with the AN/SN messages generated by sensors. If a collision happens, all packets involved are lost and no lost packet will be retransmitted. The following are the default values of specific parameters used in the simulations, unless otherwise stated. Coverage threshold $\alpha = 0.5$; $f = \frac{1}{60 \text{ seconds}}$; $R_{\text{noti}} = 20 \text{ meters}$; active and passive signal detection range $R_{a} = 20 \text{ meters}$, $R_{ps} = 5 \text{ meters}$; sensor’s detection range for attacker $R_{a} = 0.1 \text{ meter}$; sensor’s sensing range $R_{s} = 10 \text{ meters}$; attacker’s detection error of the active signal direction $\theta = \frac{1}{20} \text{ radian}$; attacker moving speed $v = 1 \text{ meter/second}$; attacker memory size $M = 2000$, which is the maximum number of sensors whose locations can be stored by the attacker; utility threshold used in sacrificial node selection $U_{th} = 0.7 \ast U_{ref}$; delay parameter used in sending SN message $\Delta t = 0.01 \text{ second}$; and parameter in timing of state switching $T = 20 \text{ seconds}$. Each point of data in the figures is the average value of the results from multiple simulations with different randomly generated network topologies.

B. Performance Comparison with and without Defense

In Fig. 3, we show the instant coverage loss of the sensor network during attack in order to demonstrate how our defense protocol improves $AC$ over time. As discussed in Section III, $AC$ is the integration of coverage($t$), the instant coverage of the sensor network. We protect the sensor networks against search-based physical attacks by balancing short term performance (coverage($t$)), and the long term performance ($AC$). Fig. 3 includes the scenarios with and without our defense protocol. In Fig. 3 (a), coverage($t$)
in the $Y$ – axis is in the time domain under search-based physical attacks. The data show that only at the very beginning (about 100 seconds), the coverage without defense is slightly larger than that with defense. However, when time goes on, the coverage value of the former drops much faster than that of the latter. The reason is, the defense mechanism forces some sensors to sleep temporarily, which decreases the short term temporary coverage. But this prevents these sensors from being detected and hence destroyed. Consequently, these sensors can have longer lifetimes and provide higher sensing coverage for longer time, and as a result long term coverage is increased.

The improvement in terms of $AC$ is more clearly shown in Fig. 3 (b), in which we compare the $AC$ with and without defense protocol under various network coverage threshold $\alpha$, ranging from 0.1 to 0.8. Recall that the sensor network is considered nonfunctional when the coverage falls below $\alpha$. We can see that $AC$ decreases when $\alpha$ increases in both scenarios, following an almost linear pattern. However, the $AC$ with our defense protocol consistently outperforms that without defense, with performance improvements ranging from 100% to 150%. In the following subsections, we will further investigate the performance improvement of our defense protocol under various network parameters, attack parameters and defense parameters.

C. Sensitivity of Defense Effectiveness to Sensor Network Parameters

In the following, we investigate the sensitivity of the defense in terms of performance improvement to two key network parameters, namely number of sensors $N$ and active signal frequency. We vary $N$ from 1000 to 5000. As we fix the size of the sensing field, different numbers of sensors in fact corresponds to
different sensor densities. The corresponding average numbers of neighbors of a sensor vary from 5 to 25, which corresponds to variation from a relatively sparse network to a relatively dense network respectively. The active signal frequency $f$ ranges from one per 100 seconds to one per 10 seconds, which captures the sampling rates of most sensor network applications. We do not show the simulation results with other values of the network parameters due to space limitations. Interested readers can refer to [20]. However, the data we report here are representative.

Fig. 4 shows that $AC$ decreases when $f$ increases or when $N$ decreases. When $f$ is large, more sensors are detected, hence $AC$ is smaller. When $N$ is large, the coverage is large due to large sensor density and more redundancy in coverage. An interesting observation is that, there exists a threshold for $f$, beyond which $AC$ decreases rapidly. When $f$ is smaller than the threshold, $AC$ decreases slowly with the increase of $f$. However, when $f$ is larger than the threshold, $AC$ decreases sharply with the increase of $f$. The reason why there exists a threshold of $f$ is; when $f$ is small, sensors send out active signals infrequently, so most sensors are detected through passive signals. In this case, increasing $f$ does not change the fact that few sensors are detected through active signals. Contrarily, when $f$ is above the threshold, most sensors are detected by active signals due to the high frequency of active signals and the fact that $R_{as}$ is larger than $R_{ps}$. In this case, active signals dominates the effectiveness of attack and increasing $f$ will significantly increase the attack effectiveness. The existence of a threshold for $f$ can help the network designer to choose a reasonable $f$ to make a good tradeoff between $AC$ and the throughput/delay of the network. While a small $f$ helps to improve the resilience of the network and $AC$, it may decrease the network performance by reducing the throughput and increasing the communication delay of the network.
A reasonable $f$ should be smaller than, but close to the threshold, which can achieve reasonable level of $AC$ while introducing little compromise to the network throughput/delay.

threshold depends on the attacker information. In case the attacker information is known or a good estimation like upper bound is available for the sensors, we may obtain the optimal threshold, which is one of our future work.

D. Sensitivity of Defense Effectiveness to Attack Parameters

In the following, we investigate the sensitivity of the defense to four key attack parameters, which are attacker moving speed, $v$, the size of the attacker memory, $M$, detection range of active signal emitted by sensors, $R_{as}$, and detection range of passive signal emitted by sensors, $R_{ps}$. We vary $v$ from 0 and 2 meters/second, which covers the range of the moving speed of most robots and human beings. For $M$, we consider two extreme cases, 0 and 2000. $M = 2000$ is one extreme where every detected sensor will be stored in memory. We vary $R_{as}$ and $R_{ps}$ between 0 and maximum communication range of sensors $R_{comm}$.

Fig. 5 shows that, $AC$ decreases with the increases of both $v$ and $M$. A large $v$ can significantly decrease $AC$ because a fast attacker can visit a larger area within a certain amount of time, and thus detect and destroy more sensors. The second observation is; the trend of the decrease of $AC$ over the increase of $v$ follows an almost linear pattern when defense is there, but $AC$ decreases much more sharply with the increase of $v$ when there is no defense. This confirms the effectiveness of our defense protocol under intense attacks. The third observation is that the improvement of $AC$ provided by our defense
protocol is more significant for large $v$. This is because, when $v$ is small, some sensors may switch to sleeping state much earlier before the attacker comes and switch back to sensing/sending state before the attacker leaves, which incurs a relatively low $AC$ improvement. As mentioned before, this problem can be alleviated if the sensors are able to detect the speed of the attacker and adjust the states switching timers accordingly. Sensors can detect attacker speed via multiple samplings at different time. When $v$ increases, our conservative state switching ensures that the sensors will not switch back too early, which increases the $AC$ improvement. The fourth observation is, $AC$ is not so sensitive to $M$ as it is to $v$. We observe that only the initial increase of $M$ helps the attacker to decrease $AC$ significantly. When $M$ is larger than that, there is little extra help for the attacker. This is because, in most situations, the number of active sensors in the detection range of the attacker is limited due to the sensor density not being very high. Therefore, the attacker cannot detect many sensors most of the time. Thus larger memory is not so useful.

In Fig. 6, we show the $AC$ with our defense protocol under different combinations of $R_{as}$ and $R_{ps}$. We can see that $AC$ decreases with the increase of either $R_{as}$ or $R_{ps}$ due to the increasing detecting ability of the attacker. Note that the curve for $R_{as} = 5 \text{ meters}$ and $R_{as} = 10 \text{ meters}$ merges when $R_{ps}$ is larger than 10 $\text{meters}$. This is because in our attack model, the passive signals have higher priority than active signals because of the higher detection accuracy. When $R_{ps}$ becomes larger than $R_{as}$, all sensors are detected by passive signals and the change of $R_{as}$ has no impact on $AC$. For similar reasons, the curves for $R_{as} = 5, 10, 15 \text{meters}$ merge when $R_{ps}$ is larger than 15 $\text{meters}$.

E. Sensitivity of Defense Effectiveness to Defense Parameters

In the following, we investigate the sensitivity of performance improvement under two key defense parameters, namely utility threshold $U_{th}$ and state switching timer parameter $T$.

In Fig. 7, we can see that the $AC$ is constant when there is no defense, because the defense parameters are not applicable for this case. In the presence of defense, when $U_{th}$ increases from 0 to $U_{ref}$, $AC$ increases first and then decreases when $U_{th}$ approaches $U_{ref}$. This is because, when $U_{th}$ is small, there are
more than necessary *sacrificial nodes* and these *sacrificial nodes* render themselves to be potential targets for the attacker. In this case, the $AC$ improvement comes mainly from the state switching corresponding to SN messages. On the other hand, when $U_{th}$ is close to $U_{ref}$, too few sensors become *sacrificial nodes*, which makes many sensors outside the attack notification range at the risk of being detected. However, $AC$ in this case is still much better than that when $U_{th}$ is extremely small, because the improvement of $AC$ in this case comes from the state switching corresponding to both AN messages and SN messages. While it is hard to derive the optimal $U_{th}$ without the knowledge of the attacker, we observe that $AC$ stays at a relatively high value for a wide range of $U_{th}$. In this range, the benefit of fewer number of *sacrificial nodes* being potential targets due to large $U_{th}$, is compensated by the loss due to more sensors unprotected by the SN messages. In practice, the sensor network designer can first choose a reasonable value for $U_{th}$ (say $U_{ref}/2$), and adjust this value after obtaining some attacker information when time goes on.

We also observe that $AC$ increases with the increase of $T$ at first, then later decreases after $T$ is beyond a certain threshold. The reason is; when $T$ is small, many sensors switch back to sensing/sending states before the attacker leaves the area. Thus the increase of $T$ in this case helps to improve $AC$ significantly. However, a very large $T$ is unnecessary because the attacker leaves the area long time before the timers expire, which reduces the network performance in terms of coverage. Similar to what we have discussed above about optimal $U_{th}$, an optimal $T$ is hard to obtain without the attacker information. In practice, we can apply a conservative (relatively large) $T$ at the beginning and adjust it when more attacker information is obtained.
F. The Effectiveness of Defense in Hierarchical Sensor Networks

In the following simulation, we keep the same size of the sensor network but separate the network into 25 clusters. Each cluster has only one cluster-head and re-selects cluster head very 600 seconds in order to balance the power consumption among different sensors. We set the priority in Formula (9) to 1 for non-cluster-head sensors and use different priorities for cluster-heads.

Fig. 8 shows the sensitivity of the defense performance to the cluster-head priority in sacrificial node selection, referred as $p_c$ in the figure. We set all the cluster-heads with the same priority value $p_c$ and all the non-cluster-head sensors with priority value 1. Thus, $p_i = p_c$ if sensor $s_i$ is a cluster-head, otherwise $p_i = 1$. In our simulations, we set $p_c = 0, 1, 1000$ as three extreme cases. When $p_c = 0$, no cluster-head will be chosen as sacrificial node if there is any alive non-cluster-head sensor in the cluster. When $p_c = 1$, there is no difference between cluster-head and non-cluster-head sensors. When $p_c = 1000$, the cluster-head is always choose as sacrificial node when it receives the attack notification. Fig. 8 shows that giving lower priority to cluster-heads produces better performance in terms of coverage. The reason is because, in our simulation, there is only one cluster-head in a cluster, and it is responsible for communication with other clusters. Thus, the cluster-head should be protected and should not be the sacrificial node.

V. RELATED WORK

Security in WSNs is a broad area. A good overview of security in WSNs is presented in [1]. In [4], a survey on sensor network routing protocol vulnerabilities and defense schemes against several electronic attacks are explored. Our work is different from the above in that physical attacks destroy
sensors completely, unlike many other attacks, where the sensors are only affected partially in terms of functionality. In a prior work, we have identified and modeled blind physical attacks [9]. In [9], we studied the issue of deployment of sensors in a sensor network to meet lifetime requirement under blind physical attacks. Our focus in this paper is search-based physical attacks, which is quite different from blind physical attacks.

A type of attack related to physical attacks is jamming attacks [8], [21], where the attacker jams or interferes with the radio frequencies that sensor(s) are using. Physical attacks are quite different from jamming attacks in that jamming only causes a loss of operation for the attack duration, while physical attacks result in irreversible sensor destructions.

In some cases, attackers can compromise sensors with malicious intent. For instance, attackers can extract cryptographic secrets, replace them with malicious sensors under the control of the attacker etc. To protect against sensor tampering, one defense involves tamper-proofing the node’s physical package [8]. Another class of work like [22] focuses on building tamper-resistant hardware to make the memory contents inaccessible to attackers. While the above work tries to protect sensors’ physical security via improved hardware, which may not be always achievable under powerful physical attacks, we propose a defense protocol that does not assume any indestructible hardware and can alleviate the destruction of physical attacks significantly.

He et al. [23] propose to use *sentries* to save the power of sensors in event-tracking sensor networks. Sentries are a subset of sensors which keep awake in order to monitor events when other sensors are sleeping or in low-power consuming state. When events happen, sentries awaken some of the other sensors near them in order to increase the quality of surveillance or event-tracking. In [23] the sentries are used to save power of sensors, but in our work, the sacrificial nodes are used to protect sensors from being physically destroyed by the attacker.

Li et al. [24] propose a distributed algorithm for guiding a user across a sensor network. The user can communicate with the sensors and thus avoid some danger areas in the network. Corke et al. [25] study a deployment problem in sensor network in which autonomous aerial vehicles communicate with sensors
deployed and determine the gaps in connectivity, which is used for a later repair process. In these works, sensors help users, which could be human beings, robots or autonomous aerial vehicles, achieve certain goal via communication. In this paper, we are addressing a different problem, in which sensors cooperate with each other to defend against attacks via local communication.

Gui et al. [26] study the trade-off between power consumption and quality of surveillance in event tracking. In [26], sensors around a moving event notify other sensors in the neighborhood such that the sensors nearby, which are in low power surveillance state, can switch to high power tracking state in time to achieve good quality of surveillance with minimum power consumption. Constant event speed is assumed in [26]. A similar work is [27], in which sensors cooperate with each other via messages so that only the sensors around a moving event are in tracking state. In [27], the speed and moving direction of the event are assumed to be known/measurable by the sensors. While the attack notification and states switching in our defense protocol bear some similarities with the above work, we have one extra constraint, i.e., minimizing the number of messages for notification that can be detected by the attacker. The fundamental tradeoff between cooperation via messages and minimizing number of messages detected, and the trade-off between providing sensing coverage, while avoiding being detected makes our problem more challenging. Besides in our model, sensors do not possess any attacker knowledge, like attacker speed, moving direction etc.

VI. CONCLUSIONS

Physical attacks are patent, potent and inevitable threats in sensor networks. In this paper we addressed the issue of search-based physical attacks in sensor networks and their defense. Specifically, we defined a representative model of search-based physical attacks. We then proposed a sacrificial node-assisted defense protocol to defend sensor networks against search-based physical attacks. The core principle of our defense is to trade short term local coverage for long term global coverage through the sacrificial node-assisted attack notification and state switching of sensors. We studied performance impacts based on a novel metric that we defined, namely Accumulative Coverage (AC). Our performance data clearly demonstrated that search-based physical attacks cause a significant deterioration in AC. However, our
performance data showed that our defense approach can effectively maintain AC even under intense search-based physical attacks. To the best of our knowledge, ours is the first work that identifies the problem, defines an attack model, and proposes a mechanism for defending against search-based physical attacks. We however believe that this is just an important first step in this regard. Our current ongoing work is focusing on studying multiple and cooperating physical attackers.

REFERENCES


