Vulnerability and Protection for Distributed Consensus-based Spectrum Sensing in Cognitive Radio Networks

Qiben Yan* Ming Li† Tingting Jiang* Wenjing Lou* Y. Thomas Hou*
* Virginia Polytechnic Institute and State University, VA, USA
† Utah State University, Logan, Utah, USA

***Abstract*—Cooperative spectrum sensing is key to the success of cognitive radio networks. Recently, fully distributed cooperative spectrum sensing has been proposed for its high performance benefits particularly in cognitive radio ad hoc networks. However, the cooperative and fully distributed natures of such protocol make it highly vulnerable to malicious attacks, and make the defense very difficult. In this paper, we analyze the vulnerabilities of distributed sensing architecture based on a representative distributed consensus-based spectrum sensing algorithm. We find that such distributed algorithm is particularly vulnerable to a novel form of attack called covert adaptive data injection attack. The vulnerabilities are even magnified under multiple colluding attackers. We further propose effective protection mechanisms, which include a robust distributed outlier detection scheme with adaptive local threshold to thwart the covert adaptive data injection attack, and a hash-based computation verification approach to cope with collusion attacks. Through simulation and analysis, we demonstrate the destructive power of the attacks, and validate the efficacy and efficiency of our proposed protection mechanisms.**

I. Introduction

Cognitive radio (CR) [1] has emerged as a key technology to enabling the use of licensed spectrum bands from incumbents, also known as primary users (PUs), when they are idle. An important challenge in CR technology is *reliable spectrum sensing* [2], by which cognitive radio devices, also known as secondary users (SUs), detect and exploit a spectrum band when it is unused, but vacate the channel immediately upon detecting the presence of primary users. Cooperative spectrum sensing, which exploits the cooperation of multiple SUs and leverages the spatial diversity among those location-dispersed SUs, has shown significant advantages in achieving reliable spectrum sensing results [3].

Cooperation in spectrum sensing can be achieved in two models: centralized or distributed. The former uses a common receiver (i.e., fusion center) to collect sensing results from all SUs and to make final spectrum sensing decision. It relies on a centralized infrastructure which may be unavailable in ad-hoc CR networks. In contrast, a distributed approach allows SUs to share individual sensing results with their neighbors, and to make their own sensing decisions. Therefore, distributed spectrum sensing model is more suitable for cognitive radio ad-hoc networks (CRAHN) [4].

Despite the many benefits cooperative spectrum sensing entitles, it is vulnerable to many potential attacks. Attackers may generate a false primary user signal to launch a *primary user emulation (PUE) attack* [5] in order to gain unfair share of the spectrum usage, or they can manipulate SUs' sensing reports by various means in order to invert the detection results (i.e., presence or absence of PUs), which is often termed as *data falsification attack* [6]. Current research in securing cooperative spectrum sensing have been focusing on addressing these attacks under the centralized model [7], [8]. Similar security threats exist in the distributed schemes but are left under addressed thus far. In fact, a distributed scheme is even more vulnerable to such attacks due to its distributed and cooperative natures. As an example, recently, a bio-inspired consensus-based distributed spectrum sensing algorithm has been proposed [9], [10]. It is merely based on localized sensing status measuring and exchanging, thus is very efficient and scalable. However, due to the distributed and cooperative natures of the protocol, the impact of malicious behaviors of an attacker, if not defended properly, would propagate through the whole network [11], causing long-term widespread impacts. More involved attacks which would undermine the distributed spectrum sensing mechanisms without being detected are also possible.

In this paper, we focus on the protection of distributed spectrum sensing in CR ad hoc networks. We first identify various forms of attacks that can subvert distributed consensus-based spectrum sensing, and then propose corresponding protection mechanisms. To the authors' best knowledge, this is the first paper to address the security issues in distributed spectrum sensing in CR networks. This paper makes three major contributions as follows.

(1) We analyze the vulnerabilities of distributed consensus-based spectrum sensing by proposing several novel attacks. They include naive ones that aim at causing disruption to sensing operations, and more sophisticated *covert adaptive data injection attack*, which is capable of adjusting attack strategies via learning through perceived environments and stealthily manipulating the sensing results without being caught by traditional detection schemes. The

This work was supported in part by the US National Science Foundation under grants CNS-1117111, CNS-0831628, and CNS-0910531. T. Jiang's work has been supported by an NSF Graduate Research Fellowship.

978-1-4673-0775-8/12/$31.00 ©2012 IEEE

latter is the first adaptive attack with learning capability in the area of secure spectrum sensing. We also discuss advanced collusion attacks that are hard to defend against.

(2) We present several protection mechanisms corresponding to the various attacks we have identified. In particular, we propose a novel robust distributed outlier detection algorithm with adaptive local threshold to defend against covert adaptive data injection attack, and a hash-based computation verification scheme to defend against colluding attackers.

(3) Through extensive simulation and analysis, we show the severe impacts of covert adaptive attacks to distributed spectrum sensing. We also present the effectiveness of our detection mechanisms under various detection parameters, network topologies and sensing data variances. Moreover, the costs of proposed protection mechanisms are shown to be low.

II. Related Work

The existing work on secure cooperative spectrum sensing mainly focused on centralized secondary network model. The vulnerabilities in the centralized model lead to two types of attacks. The spectrum sensing data falsification attack is considered in [6], [7], [12], [13]. Li et al. [12] proposed *dependent attack* to deviate the OR rule at fusion center, in which case the attackers know the reports from other secondary users. In [13], Fatemieh et al. presented *omniscient attack* to stealthily manipulate the average power, by which coordinated attackers have the measurements of the whole network. However, these attackers are incapable of adapting their attack strategies for each sensing period. In contrast, our proposed attacks have the following differences: 1) our attacks employ adaptive attack strategies with learning capability, by which the attackers can adjust their strategies according to their perceived local environments and sensing algorithm; 2) we consider distributed model, with only local information available; 3) our attacks focus on consensus-based spectrum sensing algorithm [9] to disrupt the consensus operation or covertly deviate the sensing result. Another existing attack termed as primary user emulation attack is proposed in [5], in which an attacker may mimic a primary user's signal to evict secondary users, which is complementary to our attacks.

For defense mechanisms, outlier detection is widely used, either by statistics-based methods [6] or signal propagation-based methods [7], [13]. Our outlier detection mechanism depends on signal propagation model for classifying original measurements from different SUs. However, our approach further introduces an adaptive detection algorithm with varying parameters, which differs from all existing work based on fixed defense strategies. Furthermore, oppose to current detection mechanisms, ours will ensure the correctness of consensus operations by integrating hash-based computation verification at each node that is able to testify the integrity of data involved in its neighbors' computations. The verification process is different from existing hash-based scheme [14] that only attests the data from the node itself.

III. System Model

In this section, we describe the network model considered in this paper, and we briefly review the distributed consensus-based spectrum sensing algorithm.

A. Network Model

We consider a CR network where PUs and SUs coexist. PUs are located far away from the secondary network, but usually have high transmission power, so we assume the whole secondary network is within the PUs' transmission range. Different PUs working under the same spectrum are separated far away enough to reduce interference. As a result, in one secondary network, the entire spectrum can be regarded as multiple disjoint primary channels and each SU needs to sense all of them. Without loss of generality, we consider the scenario that all the SUs in a secondary network are detecting the incumbent existence in one primary channel.

SUs form a CRAHN. We assume the collective coverage range of the SUs that form the CRAHN is small compared with the coverage range of a PU, while the distance between two SUs is long enough for spatial diversity exploitation. We assume the primary and secondary network topologies remain unchanged during one sensing period, which starts from each SU measuring its local received PU power level and ends upon achieving a unanimous decision by all SUs. We also assume the communication links between SUs employ some reliable communication protocol so the communications are error-free.

We adopt energy detection spectrum sensing method. The sensing output of each SU n_i is the received power of the PU, P_i, which can be expressed by following the signal propagation model [15] as follows:

$$P_i = P_0 - (10\alpha log_{10}(d_i/d_0) + S_i + M_i)(dB), \qquad (1)$$

where P_0 is the transmit power of PU, α is the path-loss exponent, d_0 is the reference distance (in this paper, $d_0 = 1m$), d_i denotes the distance from SU to the measured PU, S_i represents power loss due to shadowing fading, and M_i represents the multi-path fading effect. We adopt the widely used log-normal shadowing model [15], by which S_i is modeled as a Gaussian random variable with $S_i \sim N(0, \sigma^2)$. We consider M_i as negligible. Therefore, P_i can be modeled as a Gaussian distribution $P_i \sim N(\mu_i, \sigma^2)$, in which $\mu_i = P_0 - 10\alpha log_{10}(d_i)$. For simplicity, we assume σ is distance-independent to PU, so that each SU experiences i.i.d. Gaussian shadowing and fading.

B. Distributed Consensus-based Spectrum Sensing

In a nutshell, a distributed consensus-based spectrum sensing algorithm works as follows. It starts a sensing period by each SU taking the local measurement of the received PU's signal using an energy detection mechanism. The SUs then exchange their local sensing measurements with their direct neighbors. Each SU, upon receiving the updates from all its neighbors, updates its sensing state following a state update algorithm; if the differences among these new sensing states are above a certain threshold, an update is necessary and the

SU will send a state update message to all its neighbors. This process continues iteratively till no more update is triggered and the sensing state at every node in the network has reached the consensus. In what follows, we briefly review a distributed consensus-based spectrum sensing algorithm focusing on the state update algorithm and distributed state update protocol.

The secondary network can be modeled as an undirected graph, denoted by a pair $(\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{n_1, n_2, \ldots, n_m\}$ denoting a set of secondary nodes, $\mathcal{E} \in \mathcal{N}^2$ denoting a set of undirected edges. We will use secondary nodes and SUs interchangeably in the following sections. The performance of consensus algorithm is associated with the connectivity of secondary network, which can be represented by an adjacency matrix of the network graph. The consensus-based spectrum sensing algorithm can be expressed using a discrete-time state equation:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k)), \tag{2}$$

where the initial state $x_i(0)$ is the original sensing measurement of node n_i, and $x_i(k)$ is the updated state at time step k, $\mathcal{N}_i = \{j | (j,i) \in \mathcal{E}\} \in \mathcal{N}$ denotes the neighbor set of node n_i, and ϵ is a consensus parameter. State update occurs at discrete time $k = 0, 1, 2, \ldots$ for each node locally. With some constraints on network connectivity and parameter ϵ [16], the final average consensus result $\alpha = [\sum_{i=1}^{m} x_i(0)]/m$ is asymptotically reached for all nodes. The final sensing decision at each node is made by comparing the consensus result with a primary detection threshold γ as follows:

$$\alpha = [\sum_{i=1}^{m} x_i(0)]/m \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma, \tag{3}$$

where $\mathcal{H}_1$ and $\mathcal{H}_0$ denote the hypotheses corresponding to the presence or the absence of PU. The primary detection threshold γ is determined by performance requirements. For instance, if we want to keep the primary miss detection rate $P_{MD} = P(\alpha < \gamma | H_1)$ below a threshold, while minimizing the primary false alarm rate $P_{FA} = P(\alpha > \gamma | H_0)$, the threshold γ is given as follows [17]:

$$\gamma = \frac{\sigma}{\sqrt{m}} Q^{-1}(1 - P_{MD}) + \mu_\alpha, \tag{4}$$

where $\alpha \sim N(\mu_\alpha, \sigma^2/m)$, in which $\mu_\alpha = (\sum_{i=1}^{m} \mu_i)/m$, $Q^{-1}(.)$ is the inverse of well-known Q-function.

Compared to the centralized cooperative spectrum sensing schemes, distributed consensus protocols are fully distributed, scalable, and with exponential convergence rate. It well suits the CRAHN.

IV. Vulnerability Analysis of Distributed Consensus-based Spectrum Sensing

Although the distributed nature of the consensus-based protocols entitles such protocols significant performance benefits, it also exposes such protocols to a number of security threats. In this section, we identify and evaluate the vulnerabilities of consensus-based protocols under various potential attacks. We consider both passive and active attackers. We also consider both insider and outsider attackers. An outsider attacker is one who may intercept other nodes' states, inject false states, perform replay attack, camouflage other honest nodes with their captured identities, etc., but who does not possess valid security keying material. An insider attacker is a compromised SU, who has knowledge of all the keying material stored in the SU node if any, capable of manipulating its sensing measurements/states, and then disseminating the spoofed information, etc. Note that another type of attackers is faulty nodes who may output measurements/states with a large deviation due to hardware or software failure. We do not differentiate intentional attackers from faulty nodes as the consequence caused is the same.

To facilitate our analysis, we classify the attacks into two categories based on the intended objectives and consequences: *disruption of sensing operation* and *stealthy manipulation of sensing results*. Attackers in the former category have limited information and capabilities, and typically launch arbitrary attacks with the objective to disrupt the sensing operation. In contrast, attackers in the latter category are more capable. To avoid being detected, they can adapt their attack strategies to the perceived environment, and can collude with each other.

Note that, we do not consider sybil attack in which a node fabricates multiple identities, or radio-jamming attack. Those are considered outside the scope of this paper and could be addressed in separate publications.

A. Disruption of Sensing Operation

We identify two types of attacks that can lead to disruption of sensing operation. The attacks in this category may come from insider attackers, outsider attackers or faulty nodes. We also analyze their harmful impacts on consensus-based spectrum sensing algorithm.

1) Blocking attack: Blocking attack refers to unexpected cease of information transmission from a SU. This is the weakest attack in the sense that the only induced damage is the isolation of the SU and possible partition of the network graph. Intuitively, if the network graph is divided into several subgraphs, the consensus can only be reached for each isolated subgraph. Its impacts are stated in the following theorem:

Theorem 1. *let $A \in M_{n \times n}$ be the adjacency matrix of a secondary network. After blocking several secondary users by the attackers, if the adjacency matrix of the remaining network $\widetilde{A} \in M_{\widetilde{n} \times \widetilde{n}}$ satisfies*

$$(I + \widetilde{A})^{\widetilde{n}-1} > 0,$$

the attackers achieve no more benefit than defeating several secondary users. Otherwise, the whole secondary network is partitioned so that a global decision can never be attained.

The proof of the theorem is straightforward based on graph theory, thus is omitted here.

2) Arbitrary False Data Injection Attack: Here, the attacker injects forged data during each consensus iteration. Two forms of such attack, differing in their ways of data injection, are presented as follows.

• **Constant data injection:** the attacker ignores the state update algorithm and keeps transmitting a constant value in each state update message. We have the following theorems to illustrate the impacts of single attacker scenario and multiple attackers' scenario separately:

Theorem 2. *In a connected undirected graph with one arbitrary false data injection attacker sending constant data, a consensus can be asymptotically reached which equals to the constant value injected by the attacker, for any set of initial states.*

Theorem 3. *In a connected undirected graph with more than one arbitrary false data injection attackers sending constant data at different values, a consensus cannot be reached for the whole network.*

We omit the proof here. Intuitively, the consensus algorithm with a single attacker will take a much longer time to converge than the normal case, because the information flow is sourced from a single node, while in the normal case information are more evenly distributed across the whole network and all the nodes cooperatively propagate the information flow. Therefore, such attack also delays the consensus reaching time.

• **Random data injection:** the attacker injects random values into its neighborhood at each iteration. The impacts of random data injection attack are hard to analyze, but we can conclude that unstopped random value injection will disrupt the consensus algorithm by causing network divergence in most cases or converging to an arbitrary value.

B. Stealthy Manipulation of Sensing Results

The goal of stealthy manipulation of sensing results is to reverse the consensus result of a PU's status, either from absence to presence, i.e. *exploitation objective*, or from presence to absence, i.e. *vandalism objective* [13]. With the exploitation objective, attackers can evacuate other SUs to obtain exclusive usage of the available spectrum, while with vandalism objective, attackers will cause severe interference to the primary network. We propose two novel attacks: (1) *covert adaptive data injection attack* that achieves stealthy manipulation of sensing results with independent attacker(s), and (2) *covert adaptive data injection attack with node collusion*.

1) Covert Adaptive Data Injection Attack: This attack has two major features: "adaptive" means the attacker can adapt its strategy based on neighbors' state update information, with prior knowledge about the detection algorithm; "covert" reflects the attacker's goal of covertly manipulating the sensing results, without being detected by the detection mechanism. Outsider attackers can be effectively expelled from the network with an authentication mechanism. In this paper, we focus on insider attackers that reside in legitimate nodes.

Outlier detection algorithms are commonly used to defend against insider attackers performing data injection attacks. Generally, the current outlier detection algorithms all rely on a static attack detection threshold[1] λ to classify honest SUs and attackers. Suppose we directly apply these detection algorithms into the distributed system, i.e. at each iteration, a detection algorithm automatically expels the abnormal SUs. However, according to Kerckhoffs's principle, if the attacker knows the detection algorithm and detection threshold λ, a covert adaptive data injection attacker defined below is capable of bypassing the traditional detection algorithms.

Attack strategy. (1) At each iteration, the attacker first collects all its neighbors' states normally.

(2) The attacker then computes a maximal acceptable deviated state based on all its neighbors' submitted states and the threshold in the detection algorithm. The difference between the deviated state and genuine state indicates the *attack strength* $a(\hat{k})$, where $\hat{k} \in [0, k_{stop}]$ is the attack time.

(3) Finally, it injects the forged state into its neighborhood.

To illustrate the attacker strategy of computing a maximal acceptable deviated state, we give an example of a simple detection scheme with a threshold λ_d, by which one node n_a is flagged as attacker whenever any of its neighbors detect its abnormality. Assume the attacker has vandalism objective with its neighbors' states as $\{st_1, ..., st_{|\mathcal{N}_a|}\}$, then the maximal acceptable deviated state can be $\{\max_{i=1\to|\mathcal{N}_a|}(st_i) - \lambda_d\}$ to avoid being detected.

In practice, if the attacker is unable to collect all its neighbors' states before sending its own states, at current iteration, it replaces its previous state with a maximal acceptable deviated state calculated by a collection of neighbors' previous states. And then, it updates its current state with the spoofed previous state and sends it to the neighbors. In this case, the attack strength enforced at the current state will directly affect the next state of the network.

For an attacker, the knowledge of attack stop time k_{stop} is crucial for launching a successful attack. If $k_{stop} \to \infty$, the consensus protocol will not converge. The following inequalities show the basic principle in terms of the amount of changes an attacker has to inject in order to fulfill exploitation objective and vandalism objective respectively:

$$\bar{x} + \frac{\sum_{i=0}^{k_{stop}} a(i)}{m} > \gamma, a(i) \geq 0, \text{exploitation objective}, \quad (5)$$

$$\bar{x} + \frac{\sum_{i=0}^{k_{stop}} a(i)}{m} < \gamma, a(i) \leq 0, \text{vandalism objective}, \quad (6)$$

where $\bar{x}$ is the average value of the original measurements of the whole network, which is also the legitimate consensus result, and m is the number of nodes in the network. Because

[1] This threshold is for the purpose of detecting the presence of attacks, which is different from primary detection threshold γ mentioned before. In the later section, unless otherwise noted, the detection threshold denotes attack detection threshold.

the consensus algorithm has invariant average quantity [16], the impact of each attack strength $a(i)$ can be quantified as augmenting the final consensus result by $a(i)/m$. However, the attackers have no way of knowing $\bar{x}$ without the global knowledge. Therefore, we propose an iterative stop strategy. Let $\tilde{x}_{min}(k)$ and $\tilde{x}_{max}(k)$ be the minimal and maximal state from neighbors of the attacker at k-th iteration, the attacker injects forged states only when $\tilde{x}_{min}(k) < \gamma$ or $\tilde{x}_{max}(k) > \gamma$ for exploitation or vandalism objectives respectively. Otherwise, the attacker follows the consensus protocol. The proposed stop strategy guarantees the attacker's neighborhood to achieve the objective first, whose deviated states will then spread through the whole network for reversing the consensus result.

Multiple attackers with the covert adaptive attack strategy can jointly set their attack strengths, so that the protocol converges faster to their desired consensus objectives. Covert adaptive data injection attack is effective in evading traditional outlier detection. In section V-A, we will present a novel detection mechanism to invalidate such attack.

2) Covert Adaptive Data Injection Attack with Node Collusion: The covert adaptive data injection attack becomes even more powerful and harder to defend against when nodes start to collude. Not only can such attack obtain a faster convergence rate to their desired objectives, but it can also evade the computation verification scheme proposed in section V-B. Both the insider attacker and outsider attacker can perform such collusion attack.

When we involve the protection mechanism with computation verification to check the legitimacy of consensus operation, collusion attackers will avoid being caught by sending forged verification to cover up each other's false data. Moreover, stronger collusion attackers are capable of employing more malicious neighbors for false validations. In section V-B, we will present a hash-based computation verification mechanism to defend against collusion attacks.

V. Protection of Distributed Consensus-based Spectrum Sensing

The vulnerabilities of distributed consensus-based spectrum sensing algorithm demand for effective protection mechanisms. In this section, we first present a robust distributed outlier detection mechanism with adaptive local threshold to counter covert adaptive data injection attack. Then, we put forward a hash-based computation verification mechanism that ensures the correctness of a neighbor's state update process to thwart collusion attacks by common neighbor cross-validation.

A. Robust Distributed Outlier Detection with Adaptive Local Threshold

The goal of this protection mechanism is to detect and eliminate the abnormal states injected by attackers. As described in section IV-B1, the traditional outlier detection mechanisms used in the existing spectrum sensing rely on a fixed and known global detection threshold, which enables an attacker to have strengthened attacking capabilities throughout the whole consensus process.

According to the consensus algorithm, the maximum state of the network is monotonically decreasing, while the minimum state of the network is monotonically increasing until reaching consensus [18]. The updated states of each node are bounded by the maximum and minimum states, which gradually converge on the same value, while the differences among updated states of various SUs are bounded by the differences between the maximum and minimum states, which gradually diminish until reaching zero.[2] So the main idea of our detection algorithm is to use localized detection threshold at each node, and adapt the threshold with the diminishing behavior of state differences. The benefits of adaptive local threshold are twofold: (1) it becomes more difficult for attackers to guess all the instant detection thresholds of its neighbors without two-hop information; (2) the detection thresholds drop with the shrinking of variances among different states. Especially at the final consensus state, the detection thresholds reach zero which gives zero-tolerance to the attackers.[3]

To illustrate the detection mechanism, we assume a common communication range for each SU d_{cr}. Consider to compute the threshold at honest SU n_h in its neighborhood, we have its two honest neighbors n_i and n_j with distances d_i and d_j from the PU with $d_j = d_i + \Delta d_{ij}$, $0 < \Delta d_{ij} \leq 2d_{cr}$. We use the method in [7] to find a detection threshold λ_{h0} for SU n_h at starting time, such that with high probability (e.g. 0.99), $x_i(0) - x_j(0) \leq \lambda_{h0}$ satisfies. According to Eq. (1), the distribution of difference is as follows:

$$x_i(0) - x_j(0) = N(10\alpha log_{10}\frac{d_i + \Delta d_{ij}}{d_i}, 2\sigma^2). \qquad (7)$$

For a fixed d_i, $\Delta d_{ij} = 2d_{cr}$ will maximize the mean of the distribution. We assume a large attenuation factor α to reduce the influence from uncertainty of α. Then, d_i is estimated through a robust statistic estimation. *Median estimate* is used in [7], while *biweight estimate* [19] is another good candidate, which has a higher efficiency in terms of the variance of estimation. To trade off the overhead and performance, we use median and biweight estimation comparatively to determine the estimation of $x_i(0)$: x_{est} =median$(x_k(0))$ or biweight$(x_k(0))$ for $k \in \mathcal{N}_h$. Thus, $d_i \approx d_{est} = 10^{\frac{P_0 - x_{est}}{10\alpha}}$. Now we have the following distribution of difference:

$$x_i(0) - x_j(0) = N(10\alpha log_{10}\frac{d_{est} + 2d_{cr}}{d_{est}}, 2\sigma^2). \qquad (8)$$

Assume $\mathbf{Pr}(x_i(0) - x_j(0) < \lambda_{h0}) > 1 - \mu$, where μ is *detection parameter* (typically $\mu = 0.01$), we can calculate λ_{h0} using Eq. (8) as $\lambda_{h0} = \sqrt{2}Q^{-1}(\mu) + 10\alpha log_{10}(\frac{d_{est}+2d_{cr}}{d_{est}})$.

Up to now, we obtain the detection threshold at starting time. Then the updating threshold of node n_h at k-th iteration is denoted as λ_{hk}. From the above deduction, we notice the

[2] Although the differences between the updated states of any two SUs are not monotonically decreasing, the diminishing trends are guaranteed.

[3] Even if the attacker has global knowledge to guess all the instant detection thresholds, and compute a deviated state to bypass the detection at each iteration, the influence to the final results will be limited due to the shrinking thresholds.

Assume: every SU in the network owns a unique ID and shares a unique key with every neighbor; every pair of neighbor nodes has at least one common neighbor; there exists a secure neighborhood discovery mechanism, with which each node can obtain two-hop neighborhood information. In the k-th iteration ($k > 0$) (when $k = 0$, every node first submits its measurement to its neighbors using authenticated broadcast.)

- Update Commit: (First round) after one node collects all its neighbors' submissions, it computes and disseminates an updated submission using authenticated broadcast containing a hash commitment of its inputs together with its own data. Therefore, node n_v receives a collection of updated submissions from its neighbors.
- Distributed Verification: (Second round) every node disseminates all its neighbors' data collected at the beginning of first round using authenticated broadcast, so node n_v receives a collection of data from the two-hop neighbors. Then node n_v performs the following verification: (1) it checks the IDs in the collection are consistent with its stored neighbor IDs; (2) it checks whether its own data and the common neighbors' data are incorporated in each updated state by recomputing each updated state and hash commitment; (3) whenever one of the verification for node n_p fails, multiple $MAC_{K_{vi}}(ERR, ID_p)$ will spread through the whole network to stop the state update process, where ERR is a unique message identifier, K_{vi} is the shared key between n_v and its neighbor n_i.

Fig. 1: The Distributed Hash-based Verification of Neighbor State Update.

implicit meaning of detection threshold is the maximal acceptable difference between two honest SUs in the neighborhood. Then in order to adapt the detection threshold according to the shrinking difference, we calculate the robust statistic estimate for differences, $estdif_{hk}$ =median($|x_{j1}(k) - x_{j2}(k)|$) or biweight($|x_{j1}(k) - x_{j2}(k)|$), where $n_{j1},\ n_{j2} \in \mathcal{N}_h$. Therefore, we propose the following updating algorithm:

$$\lambda_{h(k+1)} = \frac{estdif_{h(k+1)}}{estdif_{hk}} \lambda_{hk}. \tag{9}$$

As $estdif_{hk} \xrightarrow{k \to k_{final}} 0$, we can guarantee $\lambda_{hk} \xrightarrow{k \to k_{final}} 0$, finally revealing zero-tolerance to attackers. To prevent attackers from forging an alarm to exclude legitimate nodes, we adopt majority rule to dispute any suspicious attacker. The whole protocol is described as follows:

- Every node computes its detection threshold at each iteration according to Eqs. (8) and (9), and then identifies suspicious attackers in its neighborhood.
- Once a node discovers a suspicious attacker, it broadcasts a *primitive alarm* to its neighbors which will not be forwarded.
- Assume the number of common neighbors between a node and the suspicious attacker is B. If the node collects no less than $\lceil B/2 \rceil$ primitive alarms from the common neighbors, it will broadcast a *confirmed alarm* and forward it to the remaining network.
- Finally, once the presence of covert adaptive data injection attackers is disclosed, it is straightforward to handle them or eliminate their impacts.

B. Hash-based Computation Verification of Neighbor State Update

The above outlier detection mechanism detects abnormal node measurements/states in the statistical sense. It is only effective when the majority of nodes in a neighborhood are honest. When malicious colluding nodes exist, the statistical outlier detection methods become less effective. In order to defend against collusion attacks, each node must ensure: (1) the authenticity and integrity of the updated states sent by neighbor nodes; (2) the state update algorithm has been followed truthfully at a neighbor node n_v, i.e. it has correctly executed the update algorithm using all n_v's neighbors' data.

To realize the above additional goals, we propose a hash-based approach with common neighbor cross-validation to counter *collusion attacks with honest common neighbors* and provide computation verification. To provide sensing data legitimacy check, data authenticity/integrity and computation verification simultaneously, we can combine our outlier detection with the hash-based verification mechanism. Next we will focus on the hash-based verification mechanism.

We assume each node has a unique identifier and shares a unique secret symmetric key with every neighbor. In addition, every node uses authenticated broadcast such as μTESLA [20] to send messages to its neighbors to enforce message authenticity and integrity. In the fully distributed CRAHN, every node should keep a unique one-way key chain and send the key chain commitments to every neighbor.

The main goal of this approach is to ensure each neighbor node perform trustworthy state updates. We adapt the idea of common neighbor cross-validation in traditional secure data aggregation techniques [14] to counter collusion attacks. In our scheme, each SU is responsible for checking not only its own contributions, but also the common neighbors' contributions incorporated into the updated states. The details of our proposed scheme are illustrated in Fig. 1.

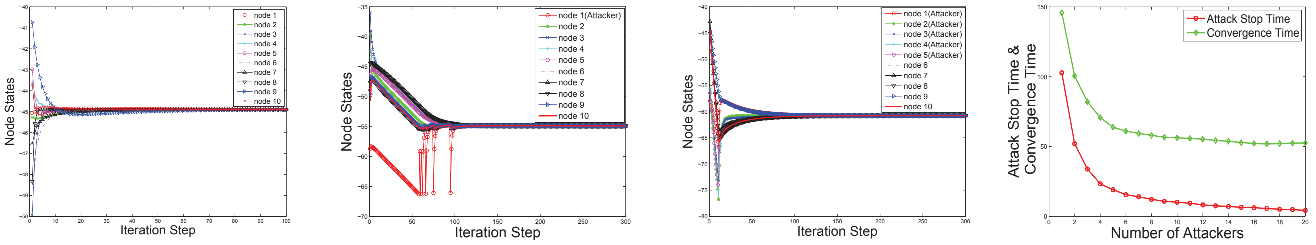
We assume there exists a secure neighbor discovery mechanism [21], by which each node can securely discover its two hop neighbors during the network initialization process. Each iteration contains two phases: update commit and distributed verification. We give an example to illustrate how the scheme works. In the k-th iteration, the initial submission of node n_h has the format: $\langle k, ID_h, value \rangle$, where $value$ is the power measurement. In the update commit phase, node n_h collects the following data from its neighbors: $d_1^{(k-1)}, d_2^{(k-1)}, \ldots, d_q^{(k-1)}$, and its updated submission $s_h^{(k)}$ has the format:

$$\langle k, ID_h, state^{(k)},$$

$$H(k \| ID_h \| state^{(k)} \| ID_1 \| d_1^{(k-1)} \| ID_2 \| d_2^{(k-1)} \| \ldots$$

$$| ID_q \| d_q^{(k-1)} \|) \rangle, \quad k > 0.$$

The hash digest in each submission is called *hash commitment* used for computation integrity check. In the distributed verification phase, n_h disseminates its neighbors' data $\langle ID_1, d_1^{(k-1)}, ID_2, d_2^{(k-1)}, \ldots, ID_q, d_q^{(k-1)} \rangle$ using authenticated broadcast. Each node in its neighborhood



(a) No attacker case

(b) One covert adaptive attacker case

(c) Ten covert adaptive attackers case

(d) Impact of attacker population

Fig. 2: Performance of covert adaptive data injection attack to distributed consensus-based spectrum sensing protected by traditional detection scheme with detection threshold -56dB.

TABLE I: Simulation Parameters

Parameter	Value	Description
N	50	Number of secondary users
R_s	1km	Length and Width of secondary network
d_{sp}	5km	Distance between primary user and center of secondary network
P_0	66dBm	Transmission power in dBm
d_{cr}	300m	Communication range of secondary user
α	3	Path-loss exponent
σ	3dB	Standard variance for fading and shadowing
N_0	-80dB	Noise power
ϵ	0.05	Consensus parameter

can then recompute the updated states based on Eq. (2) and regenerate the hash commitments, and compare the updated states and hash commitments with the received ones to verify the computation. This approach enables each honest node to check whether its neighbor nodes have performed the consensus-based state update algorithm correctly. In addition, as long as each pair of colluding attackers shares one honest common neighbor, this scheme can also detect colluding attacks. We provide a detailed security analysis in section VI-C.

VI. Evaluation

In this section, we first evaluate the vulnerabilities of distributed consensus-based spectrum sensing. We then demonstrate the effectiveness and present the security analysis of our proposed protection mechanisms, followed by a numerical analysis on their efficiency. Table I lists the system parameters used in our simulation with MATLAB. The performance results are averaged over 1000 simulation runs.

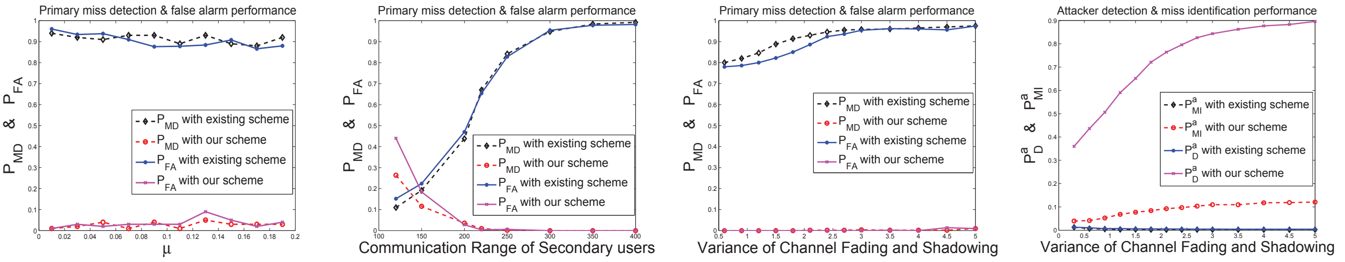
A. Impact of Covert Adaptive Data Injection Attacker

Fig. 2(a)-Fig. 2(d) show how vulnerable the consensus-based spectrum sensing with traditional outlier detection scheme [7] is under covert adaptive data injection attacks. Fig. 2(a) depicts the normal behavior in terms of protocol convergence without attackers. In less than 10 iterations, the difference among all the nodes becomes less than 1dB, which means a consensus has been reached. Fig. 2(b) shows the effect of a single attacker launching the attack. It stealthily deviates its states in order to subvert the consensus results for vandalism objective. In around 60 iterations, the nodes' states in the attacker's neighborhood has been dragged lower than the threshold so the attacker temporarily stops injecting false states and starts to follow the algorithm properly. After that, the attacker repeatedly enforce attack strength for several iterations whenever it finds the maximal state in its neighborhood stays higher than γ, until the consensus of the whole network. At around the $100th$ iteration, the network reaches a consensus but it is a wrong one (i.e., the opposite one). When there are multiple attackers working for the same objectives, the consensus can be reached much faster as shown in Fig. 2(c). Finally, Fig. 2(d) demonstrates the connection between the attack stop time and the convergence time[4] with respect to the attacker population. We observe that with the growing of attacker population, both times will decrease gradually, indicating an increasing attack power. In general, we observe that the proposed attack can be very effective in bypassing the traditional outlier detection mechanism and manipulating the final spectrum sensing result.

B. Effectiveness of Robust Distributed Outlier Detection with Adaptive Local Threshold

We evaluate our proposed outlier detection scheme by comparing with existing outlier detection scheme [7]. We study the impacts of detection parameter, network topology and sensing data variance to two primary detection performance criteria, primary miss detection rate P_{MD} and primary false alarm rate P_{FA}, determined by attack capabilities under the protection mechanisms. Fig. 3(a)-Fig. 3(c) show the effectiveness of our detection scheme, which can successfully eliminate the impacts of attackers. We observe from Fig. 3(a), the detection performance is insensitive to μ, because the attackers know the values of detection threshold and μ, and employ them to bypass the detection scheme. Fig. 3(b) shows that when the SU's communication range increases, correspondingly the density of the neighborhood increases, both P_{MD} and P_{FA} of existing scheme increase.However, with our detection scheme, P_{MD} and P_{FA} both decrease, owning to the increasing attacker detection rate with more neighbors. We notice that when

[4] The convergence time is defined as the number of iterations before the network-wide consensus is reached. Reaching a consensus means the difference among all the node states falls below 0.5dB.



(a) Impact of detection parameter μ (b) Impact of communication range (c) Impact of sensing data variance (d) Attacker detection performance

Fig. 3: Performance of robust distributed outlier detection with adaptive local threshold

communication range is extremely small, existing scheme is shown to outperform our scheme. This is because when neighborhood has a small population, cross-validation is less effective. Some honest nodes may be mistaken as attackers, which potentially amplifies the impacts of uncovered attackers to the remaining network. Fig. 3(c) indicates with the increase of data variance, the attackers have a much larger influence to the existing detection method, but our method effectively impedes the influence. To further evaluate attacker detection performance, we involve two more criteria: P^a_D as the attacker detection rate[5] and P^a_{MI} as the attacker miss identification rate[6]. Fig.3(d) shows both P^a_D and P^a_{MI} are steadily growing with the increasing of data variance with our scheme, while the increasing of P^a_{MI} is a side effect of our scheme caused by adaptive threshold, but it will not degrade the primary detection performance.

C. Security analysis of hash-based computation verification approach

We first consider the case where there is a single attacker A in the network. The security of the hash-based computation verification scheme is based on the following. (1) The secure neighborhood discovery scheme utilized in the network initialization process ensures each node learn two-hop neighborhood information securely, so that the attacker can neither discard the state value of a legitimate neighbor node nor include a forged state value while updating its state. (2) The message authenticity and integrity are guaranteed by the broadcast authentication. (3) Whether the attacker A uses a different state value $d_B^{'(k-1)}$ from what submitted by a neighbor node B for state update computation, and includes $d_B^{'(k-1)}$ in the message sent to B in the verification phase; or computes a wrong $state'^{(k)}$ using the correct neighbor states, the inconsistency will be detected by B. (4) Otherwise, based on the collision resistant property of hash function, it is computationally infeasible to generate a valid hash commitment $H(k||A||state'^{(k)}||ID_1||d_1^{'(k-1)}||\cdots||ID_n||d_n^{'(k-1)}) = H(k||A||state^{(k)}||ID_1||d_1^{(k-1)}||\cdots||ID_n||d_n^{(k-1)})$, where at least one of the primed state values does not equal to the authentic ones.

Next, we discuss the security of state update algorithm with colluding attackers. In Fig. 4, we identify four types of collusion attacks based on their increasing colluding capabilities:

- **Pairwise collusion:** this collusion (in Fig. 4(a)) emphasizes the collusion by two neighboring attackers.
- **Collusion attack with honest common neighbors:** this collusion (in Fig. 4(b)) involves more neighbors of two pairwise attackers as their collusion companions, but every pair of nodes has at least one honest common neighbor.
- **Collusion attack without honest common neighbor:** this collusion (in Fig. 4(c)) involves all the common neighbors of two pairwise attackers as their collusion companions.
- **Neighborhood Collusion**: If all the neighbors of an attacker are also malicious attackers, they form a neighborhood collusion (in Fig. 4(d)).

We show that the hash-based computation verification scheme is able to deal with the former two types of collusion attacks, but not the latter two. In cases that pairwise attackers exist (including Fig. 4(a) and Fig. 4(b)), a malicious neighbor node can cover up the forgery of the central node. However, inconsistency will still be discovered by a honest common neighbor of these colluding nodes, because the honest neighbor overhears colluding attacker's input state value.

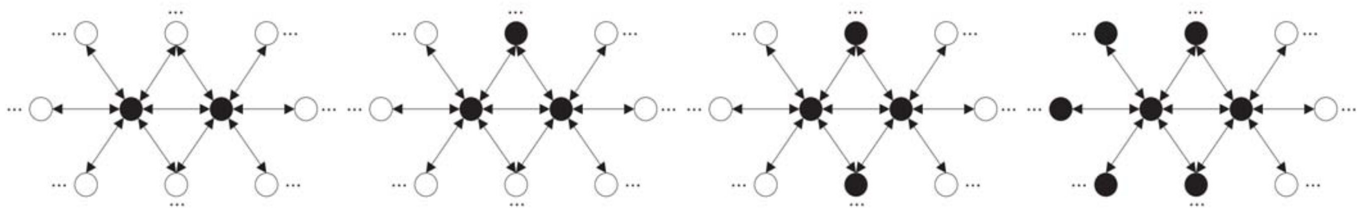
However, when there exists collusion attackers without honest common neighbor (Fig. 4(c)), one node can arbitrarily deviate its malicious neighbor's states without being detected. Note that neighborhood collusion in Fig. 4(d) cannot be addressed by distributed computation verification schemes. The central attacker in the colluded neighborhood is regarded as a hidden attacker, whose malicious behavior is most difficult to detect, because none of its neighbors is honest to follow the verification mechanism. Thus, the misbehavior of a hidden node will continue and eventually, the entire network will be inevitably controlled by the hidden attacker.

D. Cost evaluation of hash-based computation verification approach

Finally, we evaluate the efficiency of our proposed hash-based computation verification scheme through its computational and communication costs. We skip the discussion of the overhead for authenticated broadcast. We measure the computational cost by the number of hash operations at one node in each iteration, while assessing the communication

[5] This rate is defined as the probability of detecting one attacker.

[6] This rate is defined as the probability of mistakenly identifying a legitimate node as an attacker.



(a) Pairwise collusion attack

(b) Collusion attack with honest common neighbors

(c) Collusion attack without honest common neighbor

(d) Neighborhood collusion attack

Fig. 4: Different collusion styles (solid points represent attackers, hollow points represent honest SUs)

TABLE II: Costs of Hash-based Computation Verification Scheme at one node for each iteration (N is the number of neighbors, P is the length of state in bytes, h is the length of hash in bytes.)

Computation	$O(N+1)$
Communication	$N(NP+N+h+1)$
Key Storage	N

cost in terms of number of transmitted/received bytes. Another important metric for computational cost is key storage, which is defined as number of keys stored by each node. The costs are listed in Table II, where we estimate both the iteration number and node ID by one byte. The computational costs of the hash operation depend on the number of input, which relies on the number of neighbors. Therefore, the computational cost of this approach is determined by the number of neighbors. The table shows that the computational and communication costs are both acceptable.

VII. Conclusion

In this paper, for the first time, we investigated the vulnerability and protection of consensus-based spectrum sensing. We proposed various attacks that can disrupt the consensus algorithm or stealthily subvert the sensing results, especially the covert adaptive attacks with learning capability. We also developed a robust distributed outlier detection scheme with adaptive local threshold to counter covert adaptive attacks by exploiting the state convergence property. In addition, a hash-based computation verification scheme is presented to effectively defend against colluding attackers. Our simulation results demonstrated the severe vulnerabilities of distributed spectrum sensing, and also showed that our protection mechanisms are secure, robust, and efficient.

References

[1] S. Haykin, “Cognitive radio: brained-empowered wireless communications,” *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 2, pp. 201–220, 2005.

[2] S. Haykin, D. J. Thomson, and J. H. Reed, “Spectrum sensing for cognitive radio,” *Proceedings of the IEEE*, vol. 97, no. 5, pp. 849–877, May 2009.

[3] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, “Cooperative spectrum sensing in cognitive radio networks: a survey,” *Physical Communication*, vol. 4, pp. 40–62, 2011.

[4] I. F. Akyildiz, W.-Y. Lee, and K. R. Chowdhury, “CRAHN: cognitive radio ad hoc networks,” *Ad Hoc Networks*, vol. 7, no. 5, pp. 810–836, July 2009.

[5] R. Chen, J.-M. Park, and J. H. Reed, “Defense against primary user emulation attacks in cognitive radio networks,” *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 1, pp. 25–37, 2008.

[6] R. Chen, J.-M. Park, and B. Kaigui, “Robust distributed spectrum sensing in cognitive radio networks,” in *INFOCOM 2008, IEEE*, April 2008, pp. 1876–1884.

[7] O. Fatemieh, R. Chandra, and C. A. Gunter, “Secure collaborative sensing for crowdsourcing spectrum data in white space networks,” in *New Frontiers in Dynamic Spectrum, 2010. (DySPAN ’2010) IEEE Symposium on*, April 2010, pp. 1–12.

[8] A. W. Min, K.-H. Kim, and K. G. Shin, “Robust cooperative sensing via state estimation in cognitive radio networks,” in *New Frontiers in Dynamic Spectrum, 2011. (DySPAN ’2011) IEEE Symposium on*, 2011.

[9] Z. Li, F. R. Yu, and M. Huang, “A distributed consensus-based cooperative spectrum-sensing scheme in cognitive radios,” *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 1, pp. 383–393, 2010.

[10] F. R. Yu, M. Huang, and H. Tang, “Biologically inspired consensus-based spectrum sensing in mobile ad hoc networks with cognitive radios,” *Network, IEEE*, vol. 24, no. 3, pp. 26–30, May 2010.

[11] J. L. Burbank, “Security in cognitive radio networks: the required evolution in approaches to wireless network security,” in *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, May 2008, pp. 1–7.

[12] H. Li and Z. Han, “Catch me if you can: an abnomality detection approach for collaborative spectrum sensing in cognitive radio networks,” *Wireless Communications, IEEE Transactions on*, vol. 9, no. 11, pp. 3554–3565, 2010.

[13] O. Fatemieh, A. Farhadi, R. Chandra, and C. A. Gunter, “Using classification to protect the integrity of spectrum measurements in white space networks,” in *Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS ’11)*, February 2011.

[14] H. Chan, A. Perrig, and D. Song, “Secure hierarchical in-network aggregation in sensor networks,” in *CCS ’06 Proceedings of the 13th ACM conference on Computer and communications security*, October 2006.

[15] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[16] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[17] R. Tandra, S. M. Mishra, and A. Sahai, “What is a spectrum hole and what does it take to recognize one?” *Proceedings of the IEEE*, vol. 97, no. 5, pp. 824–848, May 2009.

[18] V. Yadav and M. V. Salapaka, “Distributed protocol for determining when averaging consensus is reached,” in *Proceedings of 2007 Allerton Conference on Communication, Control, and Computing*, 2007.

[19] F. Mosteller and J. W. Tukey, *Data analysis and regression: A second course in statistics*. Addison-Wesley Publishing Company, 1977.

[20] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. E. Culler, “SPINS: Security protocols for sensor networks,” *Wireless Networks*, vol. 8, pp. 521–534, 2002.

[21] I. Khalil, S. Bagchi, and N. B. Shroff, “LITEWORP: a lightweight countermeasure for the wormhole attack in multihop wireless networks,” in *DSN 2005, Dependable Systems and Networks*, 2005.