The cloud service discussed in this section, see also [52], is an alternative to the distributed trust

management scheme analyzed in Section 11.10. mobile devices are ubiquitous nowadays and their use

will continue to increase. Clouds are emerging as the computing and storage engines of the future for

a wide range of applications. There is a symbiotic relationship between the two; mobile devices can

consume as well as produce very large amounts of data, whereas computer clouds have the capacity

to store and deliver such data to the user of a mobile device. To exploit the potential of this symbiotic

relationship, we propose a new cloud service for the management of wireless networks.

Mobile devices have limited resources; new generations of smartphones and tablet computers are

likely to use multicore processors and have a fair amount of memory, but power consumption is still and

will continue to be a major concern; thus, it seems reasonable to delegate and data-intensive tasks to the

cloud. The motivation for this application is to reduce the power consumption of the mobile devices.

Transferring computations related to CRN management to a cloud supports the development of

new, possibly more accurate, resource management algorithms. For example, algorithms to discover communication channels currently in use by a primary transmitter could be based on past history but are

not feasible when the trust is computed by the mobile device. Such algorithms require massive amounts

of data and can also identify malicious nodes with high probability.

Mobile devices such as smartphones and tablets are able to communicate using two networks: (i)

a cellular wireless network; and (ii) a Wi-Fi network. The service we propose assumes that a mobile

device uses the cellular wireless network to access the cloud, whereas the communication over the

Wi-Fi channel is based on cognitive radio (CR). The amount of data transferred using the cellular

network is limited by the subscriber’s data plan, but no such limitation exists for the Wi-Fi network.

The cloud service, discussed next, will allow mobile devices to use theWi-Fi communication channels

in a cognitive radio network environment and will reduce the operating costs for end users.

Although the focus of our discussion is on trust management for CRNs, the cloud service we propose

can be used for tasks other than bandwidth management; for example, routing in mobile ad hoc networks,

detection and isolation of noncooperative nodes, and other network management and monitoring

functions could benefit from the identification of malicious nodes.

Model Assumptions. The cognitive radio literature typically analyzes networks with a relatively small

number of nodes and active in a limited geographic area; thus, all nodes in the network sense the same

information on channel occupancy. Channel impairments, such as signal fading, noise, and so on cause

errors and lead trustworthy nodes to report false information. We consider networks with a much

larger number of nodes distributed over a large geographic area; as the signal strength decays with the

distance, we consider several rings around a primary tower. We assume a generic fading model given

by the following expression:

γ i

k

= Tk   A2

sα

ik

(11.10)

where γ i

k is the received signal strength on channel k at location of node i , A is the frequency constant,

2   α   6 is path loss factor, sα

ik is the distance between primary tower Pk and node i , and Tk is the

transition power of primary tower Pk transmitting on channel k.

In our discussion we assume that there are K channels labeled 1, 2, . . . , K and a primary transmitter

Pk transmits on channel k. The algorithm is based on several assumptions regarding the secondary

nodes, the behavior of malicious nodes, and the geometry of the system. First, we assume that the

secondary nodes:

• Are mobile devices; some are slow-moving, others are fast-moving.

• Cannot report their position because they are not equipped with a global positioning system (GPS).

• The clocks of the mobile devices are not synchronized.

• The transmission and reception range of a mobile device can be different.

• The transmission range depends on the residual power of each mobile device.

We assume that the malicious nodes in the network are a minority and their behavior is captured by the

following assumptions:

• The misbehaving nodes are malicious rather than selfish; their only objective is to hinder the activity

of other nodes whenever possible, a behavior distinct from the one of selfish nodes motivated to gain

some advantage.

• The malicious nodes are uniformly distributed in the area we investigate.

• The malicious nodes do not collaborate in their attack strategies.

• The malicious nodes change the intensity of their Byzantine attack in successive time slots. Similar

patterns of malicious behavior are easy to detect, and an intelligent attacker is motivated to avoid

detection.

The geometry of the system is captured by Figure 11.13. We distinguish primary and secondary nodes

and the cell towers used by the secondary nodes to communicate with service running on the cloud.

We use a majority voting rule for a particular ring around a primary transmitter. The global decision

regarding the occupancy of a channel requires a majority of the votes. Since the malicious nodes are a

minority and they are uniformly distributed, the malicious nodes in any ring are also a minority; thus,

a ring-based majority fusion is a representative of accurate occupancy for the channel associated with

the ring.

All secondary nodes are required to register first and then to transmit periodically their current power

level, as well as their occupancy report for each one of the K channels. As mentioned in the introductory

discussion, the secondary nodes connect to the cloud using the cellular network. After a mobile device

is registered, the cloud application requests the cellular network to detect its location. The towers of

the cellular network detect the location of a mobile device by triangulation with an accuracy that is

a function of the environment and is of the order of 10 meters. The location of the mobile device is

reported to the cloud application every time it provides an occupancy report.

The nodes that do not participate in the trust computation will not register in this cloud-based version

of the resource management algorithm; thus, they do not get the occupancy report and cannot use it to

identify free channels. Obviously, if a secondary node does not register, it cannot influence other nodes

and prevent them from using free channels, or tempt them to use busy channels.

In the registration phase a secondary node transmits its MAC address and the cloud responds with

the tuple ( , δs ). Here,   is the time interval between two consecutive reports, chosen to minimize

the communication as well as the overhead for sensing the status of each channel. To reduce the

communication overhead, secondary nodes should only transmit the changes from the previous status

report. δs <   is the time interval to the first report expected from the secondary node. This scheme

provides a pseudo-synchronization so that the data collected by the cloud, and used to determine the

trust is, based on observations made by the secondary nodes at about the same time.

An Algorithm for Trust Evaluation Based on Historical Information. The cloud computes the

probable distance dk

i of each secondary node i from the known location of a primary transmitter, Pk .

Based on signal attenuation properties we conceptualize N circular rings centered at the primary, where

each ring is denoted by Rk

r , with 1   r   N the ring number. The radius of a ring is based on the

distance dk

r to the primary transmitter Pk . A node at a distance dk

i   dk

1 is included in the ring Rk

1,

nodes at distance dk

1 < dk

i   dk

2 are included in the ring Rk

2, and so on. The closer to the primary, the

more accurate the channel occupancy report of the nodes in the ring should be. Call nk

r the number of

nodes in ring Rk

r .

At each report cycle at time tq , the cloud computes the occupancy report for channel 1   k   Kused

by primary transmitter Pk . The status of channel k reported by node i ∈ Rk

r is denoted as sk

i (tq ).

Call σk

one(tq ) the count of the nodes in the ring Rk

r reporting that the channel k is not free (reporting

sk

i (tq ) = 1) and σk

zero(tq ) the count of those reporting that the channel is free (reporting sk

i (tq ) = 0):

σk

one(tq ) =

nk

r

i=1sk

i (tq ) and σk

zero(tq ) = nk

r

− σk

one(tq ). (11.11)

Then, the status of channel k reported by the nodes in the ring Rk

r is determined by majority voting as

σkR

r (tq )

= 1 when σk

one(tq )   σk

zero(tq ).

= 0 otherwise.

(11.12)

To determine the trust in node i we compare sk

i (tq ) with σkR

r

(tq ); call αk

i ,r (tq ) and βk

i ,r (tq ) the number

of matches and, respectively, mismatches in this comparison for each node in the ring Rk

r . We repeat

this procedure for all rings around Pk and construct

αk

i (tq ) =

nk

r

r=1αk

i ,r (tq ) and βk

i (tq ) =

nk

r

r=1βk

i ,r (tq ). (11.13)

Node i will report the status of the channels in the set Ci (tq ), the channels with index k ∈ Ci (tq ); then,

the quantities αi (tq ) and βi (tq ) with αi (tq ) + βi (tq ) = |Ci (tq )| are

αi (tq ) =  k∈Ci αk

i (tq ) and βi (tq ) =  k∈Ci βk

i (tq ). (11.14)

Finally, the global trust in node i is

ζi (tq ) = αi (tq )

αi (tq ) + βi (tq )

. (11.15)

The trust in each node at each iteration is determined using a similar strategy to the one discussed

earlier. Its status report, Sj (t), contains only information about the channels it can report on, and only

if the information has changed from the previous reporting cycle.

Then, a statistical analysis of the random variables for a window of time W, ζj (tq ), tq ∈ W allows us

to compute the moments as well as a 95% confidence interval. Based on these results we assess whether

node j is trustworthy and eliminate the untrustworthy nodes when we evaluate the occupancy map at

the next cycle. We continue to assess the trustworthiness of all nodes and may accept the information

from node j when its behavior changes.

Let’s now discuss the use of historical information to evaluate trust. We assume a sliding window

W(tq ) consists of nw time slots. Given two decay constants k1and k2, with k1 + k2 = 1, we use an

exponential averaging that gives decreasing weight to old observations. We choose k1   k2 to give

more weight to the past actions of a malicious node. Such nodes attack only intermittently and try to

disguise their presence with occasional good reports; the misbehavior should affect the trust more than

the good actions. The history-based trust requires the determination of two quantities:

αH

i (tq ) =

nw−1

i=0 αi (tq − iτ)ki1

and βH

i (tq ) =

nw−1

i=0 βi (tq − iτ)ki2

. (11.16)

Then, the history-based trust for node i valid only at times tq   nwτ is:

ζ H

i (tq ) = αH

i (tq )

αH

i (tq ) + βH

i (tq )

. (11.17)

For times tq < nwτ the trust will be based only on a subset of observations rather than a full window

on nw observations.

This algorithm can also be used in regions in which the cellular infrastructure is missing. An ad

hoc network could allow the nodes that cannot connect directly to the cellular network to forward their

information to nodes closer to the towers and then to the cloud-based service.

Simulation of the History-Based Algorithm for Trust Management. The aim of the history-based

trust evaluation is to distinguish between trustworthy and malicious nodes. We expect the ratio of

malicious to trustworthy nodes as well as node density to play an important role in this decision. The

node density ρ is the number of nodes per unit of the area. In our simulation experiments the size of the

area is constant but the number of nodes increases from 500 to 2,000; thus, the node density increases

by a factor of four. The ratio of the number of malicious to the total number of nodes varies between

α = 0.2 and a worst case of α = 0.6.

The performance metrics we consider are as follows: the average trust for all nodes, the average

trust of individual nodes, and the error of honest/trustworthy nodes. We want to see how the algorithm

behaves when the density of the nodes increases, so we consider four cases with 500, 1,000, 1,500,

and 2,000 nodes on the same area. Thus, we allow the density to increase by a factor of four. We also

investigate the average trust when α, the ratio of malicious nodes to the total number of nodes, increases

from α = 0.2 to α = 0.4 and, finally, to α = 0.6.

This straightforward data-partitioning strategy for the distributed trust management algorithm is

not a reasonable one for the centralized algorithm, because it would lead to excessive communication

among the cloud instances. Individual nodes may contribute data regarding primary transmitters in a

different subarea; to evaluate the trust of each node, the cloud instances would have to exchange a

fair amount of information. This data partitioning would also complicate our algorithm, which groups

together secondary nodes based on their distance from the primary one.

Instead, we allocate to each instance a number of channels, and all instances share the information

about the geographic position of each node. The distance of a secondary node to any primary one can

then be easily computed. This data-partitioning strategy scales well in the number of primaries. Thus, it

is suitable for simulation in large metropolitan areas, but may not be able to accommodate cases when

the number of secondaries is on the order of 108–109.

The objective of our studies is to understand the limitations of the algorithm; the aim of the algorithm

is to distinguish between trustworthy and malicious nodes. We expect that the ratio of malicious to

trustworthy nodes, as well as the node density should play an important role in this decision. The

measures we examine are the average trust for all nodes, as well as the average trust of individual nodes.

The effect of the malicious versus trustworthy node ratio on the average trust. We report the effect

of the malicious versus trustworthy node ratio on the average trust when the number of nodes increases.

The average trust is computed separately for the two classes of nodes and allows us to determine whether

the algorithm is able to clearly separate them.

Recall that the area is constant; thus, when the number of nodes increases, so does the node density.

First we consider two extreme cases: the malicious nodes represent only 20% of the total number of

nodes and an unrealistically high presence, 60%. Then we report on the average trust when the number

of nodes is fixed and the malicious nodes represent an increasing fraction of the total number of nodes.

Results reported in [52] show that when the malicious nodes represent only 20% of all nodes, there

is a clear distinction between the two groups. The malicious nodes have an average trust of 0.28 and

trustworthy nodes have an average trust index of 0.91, regardless of the number of nodes.

When the malicious nodes represent 60% of all the nodes, the number of nodes plays a significant

role; when the number of nodes is small, the two groups cannot be distinguished, so their average trust

index is almost equal, 0.55, although the honest nodes have a slightly larger average trust value. When

the number of nodes increases to 2,000 and node density increases fourfold, the average trust of the

malicious group decreases to 0.45 and for the honest group it increases to about 0.68.

This result is not unexpected; it only shows that the history-based algorithm is able to classify the

nodes properly, even when the malicious nodes are a majority, a situation we do not expect to encounter

in practice. This effect is somewhat surprising; we did not expect that under these extreme conditions

the average of the trust of all nodes would be so different for the two groups. A possible explanation

is that our strategy to reward constant good behavior rather than occasional good behavior, designed to

mask the true intentions of a malicious node, works well.

Figures 11.14(a) and (b) shows the average trust function of α, the ratio of malicious versus total

number of nodes. The results confirm the behavior discussed earlier. We see a clear separation of the

two classes only when the malicious nodes are in the minority. When the density of malicious nodes

approaches a high value so that they are in the majority, the algorithm still performs, as is evident from

the figures. The average trust for honest nodes even at high value of α is larger than the trust of malicious

nodes. Thus, the trusts allows the identification of malicious nodes.We also observe that the distinction

between the two classes of nodes is more clear when the number of nodes in the network increases.

The benefits of a cloud-based service for trust management. A cloud service for trust management

in cognitive networks can have multiple technical as well as economic benefits [74]. The service is

likely to have a broader impact than the one discussed here, and it could be used to support a range of

important policies in a wireless network where many decisions require the cooperation of all nodes.

A history-based algorithm to evaluate the trust and detect malicious nodes with high probability is at

the center of the solution we have proposed [52].

A centralized, history-based algorithm for bandwidth management in CRNs has several advantages

over the distributed algorithms discussed in the literature:

• Drastically reduces the computations a mobile device is required to carry out to identify free channels

and avoid penalties associated with interference with primary transmitters.

• Allows a secondary node to get information about channel occupancy as soon as it joins the system,

and later on demand. This information is available even when a secondary node is unable to receive

reports from its neighbors, or when it is isolated.

• Does not require the large number of assumptions critical to the distributed algorithms.

• The dishonest nodes can be detected with high probability and their reports can be ignored; thus,

over time the accuracy of the results increases. Moreover, historic data could help detect a range of

Byzantine attacks orchestrated by a group of malicious nodes.

• Is very likely to produce more accurate results than the distributed algorithm because the reports are

based on information from all secondary nodes reporting on a communication channel used by a

primary, not only those in its vicinity; a higher node density increases the accuracy of the predictions.

The accuracy of the algorithm is a function of the frequency of the occupancy reports provided by

the secondary nodes.

The centralized trust management scheme has several other advantages. First, it can be used not only

to identify malicious nodes and provide channel occupancy reports, but also to manage the allocation

of free channels. In the distributed case, two nodes may attempt to use a free channel and collide; this

situation is avoided in the centralized case. At the same time, malicious nodes can be identified with

high probability and be denied access to the occupancy report.

The server could also collect historic data regarding the pattern of behavior of the primary nodes

and use this information for the management of free channels. For example, when a secondary requests

access for a specific length of time, the service may attempt to identify a free channel likely to be

available for that time.

The trust management may also be extended to other network operations such as routing in a mobile

ad hoc network; the strategy in this case would be to avoid routing through malicious nodes.