Effective UAV and Ground Sensor Authentication

Xuening Xu, Chenglong Fu, Xiaojiang Du
Department of Computer & Information Sciences
Temple University
Philadelphia, PA, USA
{xuening.xu, chenglong.fu, dux}@temple.edu

E. Paul Ratazzi
Air Force Research Laboratory, Information Directorate
Rome, NY, USA
edward.ratazzi@us.af.mil

***Abstract*—Nowadays, The Internet of Things (IoT) has been widely used in various fields due to its smart sensing and communication capabilities. IoT devices serve as bridges for the cyber system to interact with the physical environment by providing various useful sensing capabilities such as battlefield surveillance, home monitoring, traffic control, etc. These capabilities also make IoT an important role in tactical missions in the military, including Reconnaissance, Intelligence, Surveillance, and Target Acquisition (RISTA). Nevertheless, IoT devices are known to have critical issues on security due to constraints on cost and resources. Most existing researches are based on smart sensors that have comparatively more computing and communication resources, while security solutions for *dumb* sensors are still lacking. Some IoT sensors that are deployed in a hostile environment are dumb due to limitations on cost and power supply, making them more vulnerable to attacks. In this work, we try to tackle this problem by proposing effective authentication solutions between a UAV and dumb IoT devices (also referred to as dumb sensors) within an example application of a UAV-sensor collaborative RISTA mission. We present two different schemes for two-way mutual authentication between the UAV and dumb sensors which utilize non-cryptographic physical layer cover channel and neighboring devices' signal sensing correlations respectively. We demonstrate the feasibility and effectiveness of our schemes with extensive real-world experiments on our prototype deployment.**

***Index Terms*—IoT, Authentication, UAV**

I. Introduction

As the trend of Internet of Things (IoT) continues to thrive, embedded smart devices with network connections are pervasively used in many fields. With these low-cost IoT devices being deployed at a large scale, a lot of human work can be replaced by smart devices. Especially for work in a hostile environment, using IoT not only avoid risks on human's life, but also achieves higher efficiency.

IoT devices can be roughly categorized into two classes: 1) *smart* IoT devices that have comparatively higher computational and communication capabilities, such as smart speakers, smart TVs, and smart doorbells; 2) *dumb* IoT devices that have very limited computational and communication resources due to constraints on cost, energy consumption, and/or device size. An example of dumb devices are light sensors and contact sensors with Zigbee [1] or ZWave wireless connections.

As a typical use case of dumb IoT devices in the battlefield and tactical environment, we consider the unmanned aerial vehicles (UAVs) centric battlefield surveillance system as the study case. In this case, UAVs are used to collect data from a large number of dumb sensors which are deployed on hostile territory for the mission of reconnaissance and target acquisition. The dumb sensors run on sensing mode by caching all collected data in its memory while keeping radio silent until activated by certain beacon messages broadcast by flying over UAVs. After that, dumb sensors exchange data with the UAV through wireless connections to export cached data and get reconfigured.

As a matter of fact, these dumb sensors face a lot of security threats because they are deployed in the battlefield or hostile territory that are physically accessible by adversaries. As the connection between the UAV and dumb sensor is intermittent, secure mutual authentication before the data exchange is critical to prevent spoofing and impersonating attacks. However, traditional crypto-based authentication solutions usually involve with high computing overhead that makes it unsuitable for dumb devices [2]. In addition, the large number of dumb devices impose significant difficulties on secure key management.

Several papers (e.g., [3]–[8]) have studied related IoT security and wireless issues. In this paper, we propose a practical non-crypto authentication scheme for UAVs and dumb sensors, which can satisfy both the requirements of high reliability and low overhead. In our work, we utilize the physical covert channel and the neighbor nodes of dumb sensors. For the authentication of the UAV, we revise the μTESLA protocol [9] to get part of the secret sent via the physical covert channel, which achieves enhanced security. For the authentication of dumb sensors, a dumb sensor's signal strength pattern is measured by multiple nearby devices and is used as the unique feature of its location. We evaluate our proposed authentication schemes regarding the effectiveness and overhead by a prototype implementation in real IoT devices. The results show that our schemes achieve satisfying security with very small overhead.

Our contributions are summarized as follows:

- We enhance the μTESLA protocol by adding the physical covert channel for a secure and low-overhead one-to-many authentication, which suits the need of the authentication of UAVs.
- We propose a novel signal strength pattern matching method to authenticate dumb sensors by taking advantage of smart devices that distribute close by.
- We evaluate our proposed schemes based on real-world experiments with prototype implementation.

The rest of the paper is organized as follows. In Section II, a brief literature review of related works on the physical covert channel, neighboring proximity-based authentication, and μTESLA is given. In Section III, system models, threat models, and assumptions are discussed. In Section IV, we introduce the design of our schemes. In Section V, we present our prototype system that is used to emulate the considered use scenario and the evaluation results of the authentication schemes. In Section VI we conclude the paper.

II. Related Works

One of our authentication schemes utilizes covert channel to send secret information. The other scheme is a neighboring-based device authentication scheme. In this section, we review the works in these two research areas. At the end, we discuss a protocol call μTESLA.

A. Using Covert Channel for Authentication

A covert channel is defined as one kind of channel that is not intended for information transfer at all. However, it can be any entity which can be utilized by a process to transfer information. Actually, covert channels raise challenges in information security [10]. To effectively detect covert channels, they should be exploited as often as possible. Many covert channels have been proposed. Most covert channels can be classified into two categories: storage covert channels [11] and timing covert channels [12]. A storage covert channel transfers information through the settings of bits by one program and the reading of these bits by another. A timing covert channel conveys information through the arrival pattern of packets. Some recently proposed covert channels have explored other aspects of communications, such as power [13], sound [14] and operating systems [15].

B. Neighboring-based Device Authentication

When a device has quite limited capabilities like dumb sensors, we may use other more powerful devices to help authenticate the dumb device. An in-band proximity authentication scheme using radio signal strength (RSS) level is proposed in [16], and this scheme utilizes radio wave transmission: RSS variations of transmitters that are in close proximity are highly correlated, whereas they are randomly diverse when two transmitters are slightly separated. We also utilize this fact in our authentication scheme. Another work [17] also exploits RSS to classify devices, but simplifies the process by limiting the scenario to body area networks.

Next, we describe a protocol called μTESLA, which is designed for authenticating broadcasts in wireless sensor networks. In μTESLA, a key chain is generated using a one-way function (e.g., SHA), and all receiver nodes loosely synchronize with the sender nodes. The sender broadcasts a message with a MAC to all nodes, where MAC is generated by the current key. The receivers buffer the received message and the MAC. Then after some delay, the sender node sends the key to all nodes. The receiving nodes can easily verify the key using the one-way function, and if valid, the MAC is also verified.

The difference between our work and μTESLA is given below:

- We simplify μTESLA by removing the MAC, and the key is directly delivered to all receiving nodes.
- Each key is split into two parts. We send two parts of the key via the normal and covert channel. Because part of the key is sent via the covert channel, the security of the scheme is enhanced.

III. Models and Assumptions

A. System Model

In this paper, we consider the scenario of bi-directional communications between UAVs and dumb sensors in the missions of reconnaissance, intelligence, surveillance, and target acquisitions (RISTA). In the scenario, three different types of devices are considered:

1) **Dumb sensors:** Small, low-cost IoT devices deployed in large scale in the battlefield or hostile territory. They have simple sensing and communication capabilities to sense the environment and exchange data with the UAV flying over. Their computational capability is too limited to perform any kind of encryption or other cryptographic operations.
2) **Smart Devices:** IoT devices deployed along with the dumb devices in the same area. They have comparatively small quantities but stronger computational capability to perform common cryptographic operations (such as encryption) for secure communications with the UAV.
3) **UAV:** The UAV functions like a coordinating base station and it is used to collect data from dumb sensors deployed on the ground. It flies over the deployed sensors intermittently to collect sensing data.

During the operation, two-way authentications are conducted. First, the flying UAV broadcasts an activation message along with a secret key (partially via a covert channel) to wake up dumb sensors from the sensing mode. Dumb sensors receive the message and they verify the secret key. If the key is valid, they accept the activation message and switch to transmission mode to exchange data with the UAV. Otherwise, they discard the message and go back the sensing mode. Second, the UAV also maintains secure connections with the smart devices. The UAV asks all smart devices to observe the signal strength of nearby dumb devices and report the results back to the UAV. Then, a dumb sensor's authenticity is determined by comparing the signal strength data with the ground truth collected during the device deployment.

B. Threat Model & Assumptions

The use of RISTA usually requires the dumb sensors to be deployed in a battlefield or a hostile territory. Adversaries may launch cyber attacks to thwart the normal functioning of the IoT network. Here, we consider two categories of attacks:

a) UAV Impersonation: An adversary may fly her own UAV, which impersonates the good UAV and tries to communicate with the dumb sensors, and then the adversary's UAV can launch various attacks.

b) Dumb Sensor Impersonation: An adversary may place her own sensors in the same field, which pretend to be benign dumb sensors and report false sensory data to the UAV. This would cause inaccurate or incorrect battlefield analysis.

To achieve these attacks, adversaries need to break the authentication procedure between the UAV and dumb sensors. We assume a common adversary with the following capabilities:

- The adversary can eavesdrop on communication between the UAV and dumb sensors since no cryptographic encryption is used.
- The adversary can use counterfeit UAV or dumb sensors to send arbitrary content via the wireless link.
- The adversary can store, analyze, and replay the history communication content between the UAV and dumb sensors.

Also, we make some reasonable assumptions about the limitations of the adversaries' capabilities:

- The adversary cannot break one-way secure hash functions such as SHA-1 and SHA-256.
- The adversary cannot sniff the content sent through the physical covert channel.
- The adversary cannot obtain the exact location of each dumb sensor.

IV. System Design

In this section, we present our non-crypto mutual authentication mechanism between the UAV and dumb sensors. Our mechanism includes different authentication methods. For the broadcast authentication of the UAV, we adapt the μTESLA protocol where we have part of the secret key sent through the physical covert channel to achieve enhanced security. For the authentication of dumb sensors, we propose smart device-assisted proximity-based authentication to differentiate benign sensors from fake sensors deployed by the adversary. Both two authentication methods utilize the covert channel encoded by some wireless physical layer parameters.

A. Physical Layer Parameters

a) Received Signal Strength Indication: In wireless communications, the received signal strength indication *(RSSI)* is the signal strength that a receiver detects, with a unit of dBm. The RSSI is mainly affected by the signal source's transmission power and signal channel characteristics such as distance, channel medium, multipath effect, and shadowing. During the very short dumb sensor activation broadcast time slot, we can reasonably assume channel characteristics remain constant, which means the RSSI would be significantly correlated to the transmission power. As a result, the sender can utilize the RSSI as a communication channel by encoding bits stream into the sequence of transmission power changes.

b) Packet Interval: In our use scenario, an UAV sends packets to dumb sensors. After the dumb sensors receive the first packet from the UAV, they keep recording the following packets and measuring each time interval between two consecutive packets. The time difference between any two consecutive packets is defined as the packet interval. With a pre-shared threshold, each interval can be decoded as a bit '1' or '0'. From the sender side, packets are sent with intentionally added delay to encode secret binary streams in the form of packet interval variations. In our scheme, the intervals longer than the threshold are decoded as 1 while shorter ones are decoded as 0. According to our study and prototype implementation, packet intervals can be detected by dumb sensors without adding any special external devices. What's more, one cannot discover the covert channel easily without knowing the pre-defined threshold.

B. UAV-to-sensor Authentication

For the case of using dumb sensors to authenticate the UAV, one-to-one authentication is not effective considering the large quantity of dumb sensors and the short communication time window. Here we adopt the μTESLA protocol [9], which utilizes the aforementioned packet interval based covert channel to send part of the secret key.

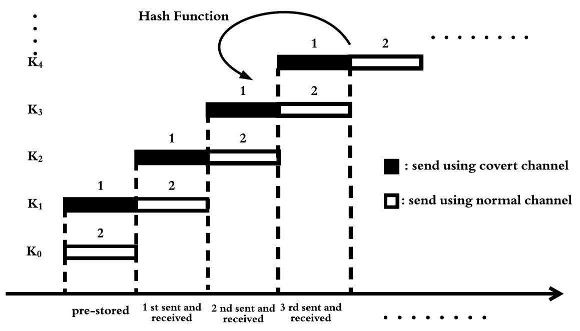


Fig. 1. Each time send and receive two part of keys. Assemble into an entire key to obtain the previous key by applying hash function, and compare the second part of computing result with stored second part of the key.

We first use the secure hash function (SHA-128) to generate a chain of hash values: K_0, K_1, K_2, K_3, and *so on*. This key chain is used in reverse order. We split each key K_i into two parts: K_i^1 and K_i^2. Before dumb sensors are deployed, the second part of K_0 (K_0^2) and the first part of K_1 (K_1^1) should be pre-stored in dumb sensors. Let's take the very first authentication as an example. UAV sends K_1^2 through normal channel and K_2^1 through covert channel to dumb sensors. Once dumb sensors receive K_1^2, they assemble the K_1^2 with pre-stored K_1^1 to get K_1, and apply the same one-way function to obtain K_0 (Fig. 1). Then they compare the pre-stored K_0^2 with the second part of the computing result K_0. If it is valid, that means UAV can be trusted and dumb sensors will make an attempt to retrieve K_2^1 from the covert channel, then dumb sensors discard stored K_0^2 and K_1^1 and store the newly received parts of the keys K_1^2 and K_2^1, which will be used for the next authentication. If invalid, dumb sensors will not perform the above and just discard received messages while keeping the stored data unchanged. Theoretically, the more bits that are sent using the covert channel, the more secure the scheme is. However, we also need to consider the overhead because we are using the covert channel. Thus, we need to figure out how many bits of the key sent using the covert channel are reasonable while not increasing too much overhead. We discuss this trade-off in the experiment part.

C. Sensors-to-UAV Authentication

The other authentication scheme we proposed is to authenticating dumb sensors using neighboring smart devices. Smart devices can detect the signal strengths of dumb sensors and use these to determine whether the sensor is in the place it should be [18]. This approach can detect a impersonating sensor that is close to a legitimate dumb sensor, which is demonstrated by our experiments.

The RISTA field typically has both smart devices and dumb sensors deployed. Smart devices have more computing and communication resources to deploy crypto-based mutual authentication with the UAV and maintain secure communications with it. In our method, these smart devices are utilized as assisting observers to measure the transmission signal strength of nearby dumb devices.

We use RSSIs measured by neighboring smart devices as patterns (*rssi_1*, textitrssi_2, *rssi_3*, *rssi_4*, ...) to represent the identity of a dumb sensor. First of all, the pattern of each dumb sensor's correct position is pre-stored in the UAV as the ground truth, e.g., during the dubm sensor deployment. When a dumb sensor sends wireless signals and tries to connect to the UAV, the smart devices around it observe RSSIs of the dumb sensor. They then send the values to the UAV, which show a pattern of the dumb sensor's current position. The UAV may compute the *Pearson Correlation Coefficient* (Equation 1) of the pre-stored pattern and the current pattern, and then decide if this is a legitimate sensor.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

where X and Y are random variables, cov is the covariance, and σ is the standard deviation.

Here, we need to set a threshold of the correlation coefficient; if the correlation coefficient is larger than the threshold, it is considered as the legitimate dumb sensor. Otherwise, the sensor is not accepted. In the experiment part, we evaluate the performance of this scheme and discuss the range in which it can be considered as the correct position.

V. Prototype Implementation & Evaluation

A. Prototype System Setup

a) Authentication Using Covert Channel: We emulated the use scenario by building a prototype system consisting of two Raspberry Pi 3 Model B Boards. One of them was configured to a WiFi (802.11b) access point, named *Pi_UAV* to emulate the UAV; the other one was normally configured to emulate the dumb sensor, named *Pi_dumb*. The access point functionality can be realized by using the *hostapd* tool with *dnsmasq*. *Dnsmasq* is an easy-to-configure DNS forwarder, designed to provide DNS services to a small-scale network. It can serve the names of local machines which are not in the global DNS.

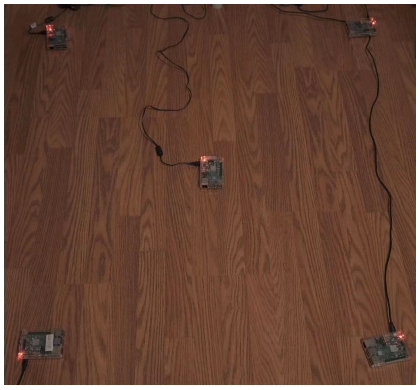


Fig. 2. *Pi_smarts* (corner) and *Pi_dumb* (center)

b) Authentication Based on Neighboring Devices: Here we use six Raspberry Pi 3 Model B Boards to emulate the use scenario. Two of them were named *Pi_UAV* and *Pi_dumb* as mentioned before. The other four were configured in monitor mode, named *Pi_smart1*, *Pi_smart2*, *Pi_smart3*, and *Pi_smart4*. *Pi_dumb* and four *Pi_smarts* are shown in Fig. 2. *Aircrack-ng* is used to monitor the *Pi_dumb*. We run *sudo iwconfig wlan1 mode monitor* command in terminal on four *Pi_smarts* to activate monitor mode and run *sudo airodump-ng wlan1* to observe the signal strength of *Pi_dumb*.

B. Packets Interval Based Covert Channel

In our experiment, we take an interval time delay of two packets as the covert channel. Before modifying the delay, it is necessary to figure out the normal interval time of sending packets. To obtain this, we create a Python script in *Pi_UAV* to send 2,000 packets and another Python script in *Pi_dumb* to receive these packets to get a normal interval time. We set a threshold to see the distribution of the interval times. The results are shown in Fig. 3.

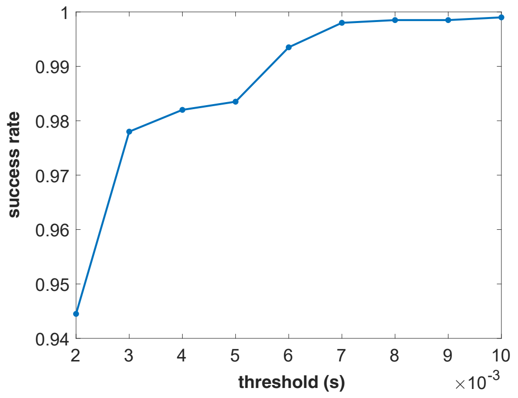
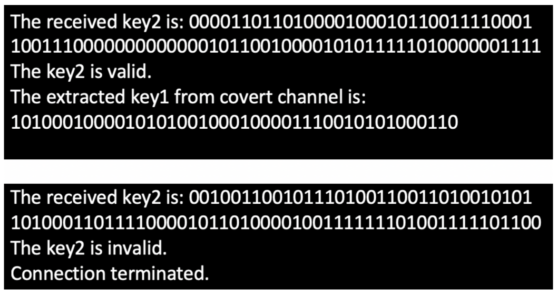


Fig. 3. For normal continuous communication, more than 98% of packets have inter-packet intervals shorter than 0.004s, indicating the interval threshold of 0.004s is enough to prevent bit error of the physical covert channel.

Success rate means the proportion of time below the threshold. Since most of intervals are larger than 0.003s, a 0.003s delay can be used to represent *0* digits. And since we want to send packets as quickly as possible, we can use a 0.005s delay as *1* digit and make the threshold 0.004s to distinguish between these two delays.

However, on Raspberry Pi 3, timing loops in software are generally not that useful because the program can be preemptively interrupted by Linux doing something else. That means when using *time.sleep()* to set up the delay, it may not "sleep" precisely the time we want. To reduce this error in the experiment, we emulated by using 10-times delay, that is, 0.03s and 0.05s with a threshold of 0.04s.

Once *Pi_dumb* is connected to the access point *Pi_UAV*, *Pi_UAV* can send packets to *Pi_dumb*. Assume that UAV has *m* data packets to send to the dumb sensor after valid authentication by sending the second part of key through normal channel, and we want to make use of the interval of two data packets as the covert channel to send the first part of the key (hidden key). Let *n* be the number of bits of the hidden key using the covert channel, and if it is no more than *m*, the intervals of *m* data packets are sufficient to send hidden key; otherwise, UAV needs to send extra packets to satisfy the usage of the hidden key. Here, we compute the growth rate of overhead in the cover channel due to modifying delay. Each time, the number of data packets is fixed (Fig. 5). From this figure, it suggests that we can make the number of bits of hidden key no more than the data packet number needed to send without suffering too much extra overhead. Also the results of *Pi_dumb* in one specific authentication are shown in Fig. 4.



```
The received key2 is: 000011011010000100010110011110001
1001110000000000000010110010000101011111010000001111
The key2 is valid.
The extracted key1 from covert channel is:
1010001000010101001000100001110010101000110
```

```
The received key2 is: 001001100101110100110011010010101
1010001101111000010110100001001111111010011111101100
The key2 is invalid.
Connection terminated.
```

Fig. 4. The picture above is the result when receiving the correct second part key; The picture below is when the received second part key is incorrect.

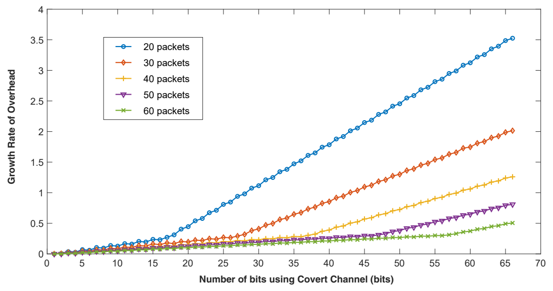


Fig. 5. When the number of bits exceeds the number of data packets, overhead increases faster. The fewer data packets to send, the more significant the growth rate of overhead when extra packets need to be sent.

C. Correlation Coefficient of Different Positions

In the use scenario, we assume each dumb sensor has more than four smart devices around it, which can be used to help UAV verify the dumb sensor. With this assumption, we do not need to emulate the scenario in which dumb sensor is on the margin or outside of the square. We placed four *Pi_smarts* in a square with an edge length of one meter and put *Pi_dumb* as the center of the square (Fig. 6). Firstly, each *Pi_smart* collected 5 signal strengths of *Pi_dumb* and computed the mean values to generate the pattern of the right position (center of the square) and stored it in the *Pi_UAV*. The pattern was stored as a list: *(rssi_1, rssi_2, rssi_r3, rssi_4)*. The mean received signal strengths were generated by relative *Pi_smart*.

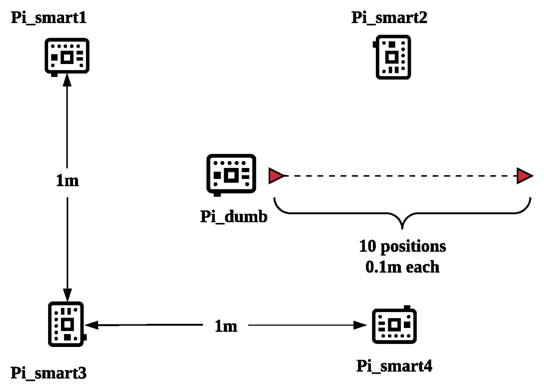
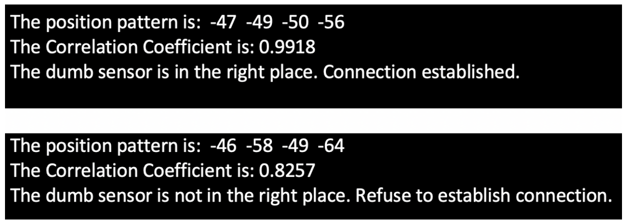


Fig. 6. There are total of ten positions between two red triangles shown in the figure to emulate malicious sensors. Every two positions has a distance of 0.1 meters.



```
The position pattern is:  -47  -49  -50  -56
The Correlation Coefficient is: 0.9918
The dumb sensor is in the right place. Connection established.
```

```
The position pattern is:  -46  -58  -49  -64
The Correlation Coefficient is: 0.8257
The dumb sensor is not in the right place. Refuse to establish connection.
```

Fig. 7. The picture above is the signal strength of the benign dumb sensor measured four neighboring smart devices. The correlation coefficient between the measured pattern and the ground truth pattern is higher than 99% to indicate the device's genuineness. In contrast, the result of a fake sensor which is shown in the picture below presents a correlation coefficient far lower than the first one.

Then, we change the position of *Pi_dumb* to emulate a malicious sensor (Fig. 6). The first position is 0.1 meters away from the center, the second position is 0.2 meters away from the center, and so on. At each position, we run the *sudo airodump-ng wlan1* command in the terminal on four *Pi_smarts* to observe the signal strength of the malicious sensor. After they have RSSIs, they send them to *Pi_UAV* to generate the pattern of the specific position. *Pi_UAV* computes the *Pearson Correlation Coefficient* of these two patterns. Some of the outputs on *Pi_UAV* are shown in Fig. 7. Fig. 8 shows the curves of different patterns, which makes it more intuitive to see the trend of the pattern as it changes with position. The relationship between distance and the correlation coefficient is shown in Fig. 9. Since we set 0.95 as the threshold of the correlation coefficient, a little move of the sensor can be detected. Maybe only within 0.05 meters, it can be considered as in the right position. Due to the fluctuation of the RSSI, this result is acceptable.

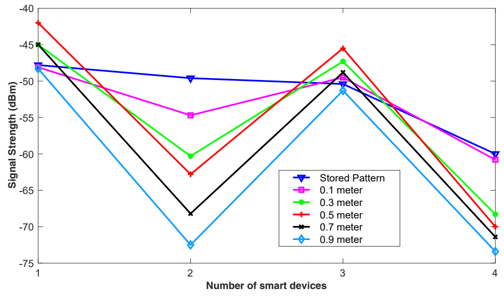


Fig. 8. As distance increases, the curve gradually deviates from the reference.

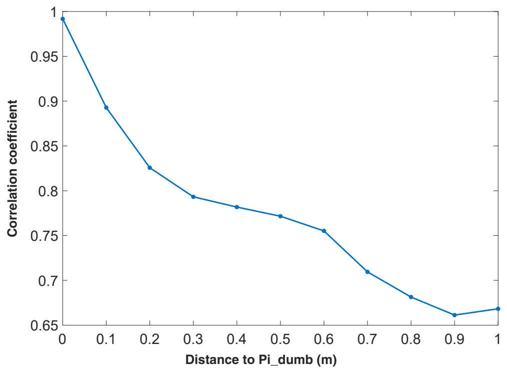


Fig. 9. Relation between the fake-to-benign distance and the signal strength pattern correlation coefficient. The measured coefficient drops drastically when the distance gets larger. Even 0.1 meters distance results the correlation coefficient of 0.8928, which is below the threshold of 0.95.

In our use scenario, the distance between devices may be much larger, but military standard devices may be used. The power and antenna sensitivity of military-grade devices should be remarkably higher than Raspberry Pi. Thus, it is reasonable to emulate the use scenario in such a small area using Raspberry Pi.

VI. Conclusions

In this paper, we proposed novel mutual authentication schemes for an UAV and dumb sensors without using crypto. The schemes can detect impersonation attacks in IoT use scenarios of air-ground collaborating with RISTA operations using UAVs and mission-specific (dumb) sensors. Our authentication schemes include the physical layer covert channel based scheme and the neighboring device based scheme. Using a covert channel enhances the security of the authentication procedure and only slightly increases the overhead. A neighboring-based authentication scheme can accurately identify sensors at different locations. We built prototype systems using real IoT devices, demonstrating that our schemes are feasible and effective.

Acknowledgement

This work was supported in part by the US National Science Foundation (NSF) under grants CNS-1828363 and CNS-1824440.

DISTRIBUTION A. Approved for public release: distribution unlimited. Case Number 88ABW-2019-2452, 20 May 2019.

References

[1] B. E. Bilgin and V. Gungor, "Performance evaluations of zigbee in different smart grid environments," *Computer Networks*, vol. 56, no. 8, pp. 2196–2205, 2012.

[2] A. Mukherjee, "Physical-layer security in the internet of things: Sensing and communication confidentiality under resource constraints," *Proceedings of the IEEE*, vol. 103, no. 10, pp. 1747–1761, 2015.

[3] L. Xiao, Y. Li, X. Huang, and X. Du, "Cloud-based malware detection game for mobile devices with offloading," *IEEE Transactions on Mobile Computing*, vol. 16, no. 10, pp. 2742–2750, 2017.

[4] X. Du, Y. Xiao, S. Ci, M. Guizani, and H.-H. Chen, "A routing-driven key management scheme for heterogeneous sensor networks," in *2007 IEEE International Conference on Communications*. IEEE, 2007, pp. 3407–3412.

[5] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway, "A survey of key management schemes in wireless sensor networks," *Computer communications*, vol. 30, no. 11-12, pp. 2314–2341, 2007.

[6] X. Du, Y. Xiao, M. Guizani, and H.-H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 24–34, 2007.

[7] X. Du and H.-H. Chen, "Security in wireless sensor networks," *IEEE Wireless Communications*, vol. 15, no. 4, pp. 60–66, 2008.

[8] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen, "A routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks," *Trans. Wireless. Comm.*, vol. 8, no. 3, pp. 1223–1229, Mar. 2009. [Online]. Available: http://dx.doi.org/10.1109/TWC.2009.060598

[9] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks," *Wireless networks*, vol. 8, no. 5, pp. 521–534, 2002.

[10] I. S. Moskowitz and M. H. Kang, "Covert channels-here to stay?" in *Proceedings of COMPASS'94-1994 IEEE 9th Annual Conference on Computer Assurance*. IEEE, 1994, pp. 235–243.

[11] K. Denney, A. S. Uluagac, K. Akkaya, and S. Bhansali, "A novel storage covert channel on wearable devices using status bar notifications," in *Consumer Communications & Networking Conference (CCNC), 2016 13th IEEE Annual*. IEEE, 2016, pp. 845–848.

[12] S. Cabuk, C. E. Brodley, and C. Shields, "Ip covert timing channels: design and detection," in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 2004, pp. 178–187.

[13] Z. Wang, L. Huang, W. Yang, and Z. He, "A classifier method for detection of covert channels over lte," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2017 International Conference on*. IEEE, 2017, pp. 454–460.

[14] L. Deshotels, "Inaudible sound as a covert channel in mobile devices," in *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14)*, 2014.

[15] K. Block, S. Narain, and G. Noubir, "An autonomic and permissionless android covert channel," in *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2017, pp. 184–194.

[16] A. Kalamandeen, A. Scannell, E. de Lara, A. Sheth, and A. LaMarca, "Ensemble: cooperative proximity-based authentication," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 331–344.

[17] L. Shi, M. Li, S. Yu, and J. Yuan, "Bana: body area network authentication exploiting channel characteristics," *IEEE Journal on selected Areas in Communications*, vol. 31, no. 9, pp. 1803–1816, 2013.

[18] P. Barsocchi, S. Lenzi, S. Chessa, and G. Giunta, "A novel approach to indoor rssi localization by automatic calibration of the wireless propagation model," in *VTC Spring 2009*. IEEE, 2009, pp. 1–5.