International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 4982-4988

Common Security Attacks on Drones

Carlos Augusto Tovar Bonilla[1], Octavio José Salcedo Parra[1, 2], Jhon Hernán Díaz Forero[2]

[1]*Faculty of Engineering - Universidad Nacional de Colombia, Bogotá D.C., Colombia.*

[2]*Faculty of Engineering - Universidad Distrital Francisco José de Caldas, Bogotá D.C., Colombia.*

Abstract

In this paper we postulate the vulnerability of the security component of the commercial dron model based on signals via WiFi, to attacks generated by hackers. This is to demonstrate that the standard ARDiscovery connection process and the WiFi access point, used by this commercials dron are usable and easily exploitable to the point of disabling the flight by means of a remote attack.

Keywords: ARDiscovery, Dron, Flight, Vulnerability.

INTRODUCTIÓN

There is a growing need today to record all our outlets and plans, capturing spectacular and unexpected scenes. This is evident in all kinds of events, whether you are practicing extreme sports or attending outdoor concerts or exhibitions, every detail is important and we need to record them from different angles, creating an unforgettable memory.

Thanks to the drones and their ability to move quickly this is possible, it will only be necessary to add an action camera and all these moments will be stored. The practice of this system generates a need on the part of consumers filling the air space.

In the boom of unmanned vehicles and the increasing technological advance it is normal to see one or more of these drones flying over an area, causing in turn inconvenience to residents and even damage to property. As a fashion item, manufacturers bring to market models that are increasingly cheaper without paying attention to important details such as device safety.

This paper will analyze the security component of some of the most common models, in order to know what kind of security are handled by signals sent via WiFi and how they can be exploited

BACKGROUND

Although there are few related works that exactly use open data sources to generate heat maps that show the occurrence of diseases in a given area, heat maps have been used to determine the occurrence of news generated in South Korea and are reported by Newscasts or portals worldwide, by Chen [1]. Keneshloo [2] uses GDELT (The Global Data on Events, Location and Tone) to predict an internal political crisis in a country of interest, is a very useful tool for social scientists and policy makers. It has a large amount of event data for historical analysis and thus generate a predictive utility.

The following are vulnerabilities found in commercial drones.

Drone Vulnerabilities: The author in [4] shows how the DEFCON 23 hacking group managed through Telnet to force some drones into disarray. In this work the Telnet application will be used for the fuzzing method (example: connect to port 44444 and send a JSON with erroneous information).

Security researcher Samy Kamkar used a Raspberry Pi, a Perl script, and freeware tools for drone deauthentication and control. This allowed his script to take control over the dron while in flight [6, 7]. This attack is called Skyjack.

The author in [8] demonstrated that drones are vulnerable to ARP (Address Resolution Protocol) poisoning, DHCP poisoning, and MAC cloning. In this paper we will demonstrate the vulnerability in one of the models that use the ARDiscovery component, shown by [18].

The previous hacks demonstrate the vulnerability of the drones based on WiFi communications, Most of these vulnerabilities were demonstrated on the old platform ARDrone. In this paper the hacks will be made on the new platform. It was confirmed that the ARP Cache poisoning vulnerability mentioned in [8] on the old drones platform is still present on the new platform. A new area of unreported vulnerabilities was also found on the other platforms, the ARDiscovery process. These previous works show us that commercial drones do not give importance to security and also the users are at risk of possible attacks when using them.

Security Frameworks: Until now, there are no related jobs that offer security on these drones. Some authors address basic safety issues in drones [4, 6-8] and some address the facets of these and the lack of security in them [9], but no solution is proposed to this problem.

A security framework on commercial drones is needed, but so far it does not exist. For example, in [10] the authors focus on securing the access point of the drones. While in [11] the authors focus on GPS security and its access point. These works go in the right direction, where the author in [9] identifies the safety faults in each component of the drone. Although the problems have been identified, a solution is not yet proposed.

WiFi based drones: In 2010 the development of a consumer multi-rotor aircraft was started to fill the growing hobby of aviation [12]. One of the devices was an aircraft

powered by a smartphone using 802.11 wireless technology to provide real-time video and easy-to-use on-screen controls with stable flight characteristics. Since this initial development there have been multiple iterations of the framework.

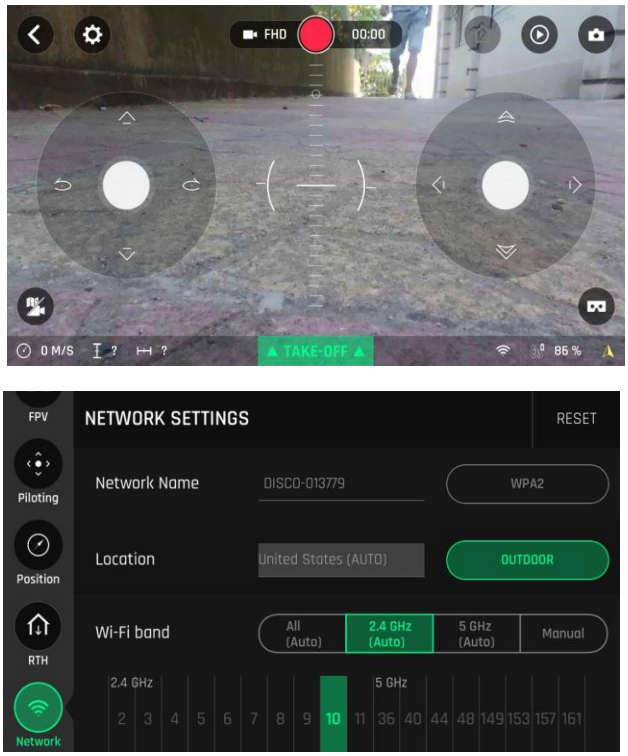

Figure 1. Application used to control dron. Source: Authors

The drones have 802.11 modules on board that allow the user to manage them through a smartphone, a tablet, a PC or in some cases a dedicated control, the devices can be linked either directly to the dron or to the control thanks to an open wireless connection through an access point, this connection keeps it open and active. The communication between the smartphone and the drone is done through an application available in the virtual stores of Android and iOS [13].

For its utility, price and easy to use, in addition to being controlled by mobile devices is that its security will be evaluated in this paper.

METHODOLOGY

ARDiscovery

The process by which network devices identify and connect to nearby nodes, particularly in an ad-hoc configuration, is known as a discovery protocol. Discovery protocols vary in implementation and operation details, but their purpose is the same: identify and allow the connection of a wireless device to an existing network.

Figure 2. Dron used for the realization of this document. Source: Authors

A discovery method known as ARDiscovery is used to establish a connection between the drone running the AR.UAV 2.0+ software and the control. The discovery protocol is limited to devices connected to its WiFi access point, and works on a combination of TCP and UDP channels, establishing a relationship over a known TCP port.

The control initiates a TCP relationship from dron-control (d2c) and control-drone (c2d) in order to establish the communication channels necessary for a successful flight. Data sent between connected devices is made via JSON sent via UDP. Once started, the application starts the ARDiscovery process to achieve a communication between the dron and its control.

Multi-layer security framework

One of the most practical security frameworks to date is defense-in-depth [16]. It is a multi-layer approach to network security. It is believed that a similar approach should be taken when securing commercial drones. In particular, we focused on Wi-Fi based drones. The challenge for safety in these types of unmanned aerial vehicles is the fact that they are wireless equipment that can fly. Therefore, security should be considered for a computer, a navigation system and a wireless network.

1. Security attack model: The threat model used in this paper is based on three basic attacks: (1) Denial-of-Service (DoS), (2) BufferOverflow and (3) ARP Cache Poisoning. This model was developed by conducting penetration tests in the dron and discovering three vulnerabilities. These three security attacks are capable of disrupting the in-flight behavior of the drone. The dron used and other similar drones are probably vulnerable to other security attacks; however, this was not evident in this analysis. In this document, the multi-layer security framework for the defense against these security attacks will be limited.

Table 1. Controller-to-UAV JSON record

<From the controller to the UAV>
{"d2c_port":54321,"controller_name":"HTC One M9","controller_type":"htc_himaulatt","device_id":"PI040338AA5B037455" }

Table 2. UAV-to-controller JSON record, controller accepted

<From the UAV to the controller>
{ "status": 0, "c2d_port": 54321, "arstream_fragment_size": 65000,"arstream_fragment_maximum_number":4,"arstream_max_ack_interval":-1,"c2d_update_port": 51, "c2d_user_port": 21 }

Table 3. UAV-to-controller JSON record, controller rejected

<From the UAV to the controller>
{ "status": -3999, "c2d_port": 0, "arstream_fragment_size": 0, "arstream_fragment_maximum_number": 0, "arstream_max_ack_interval": -1, "c2d_update_port": 51, "c2d_user_port": 21 }

Figure 3. Different JSON obtained controller-dron. Source: [18]

2. Problem in control-dron communication: The multi-layer security framework is a direct consequence of penetration testing. Based on the safety assessment of the drone, the problem is the interconnection of control to the dron. In other words, it was demonstrated that the processes and protocols used to interconnect the control and the dron are unsafe. This problem can be mitigated by adding: (1) watchdog timer to limit the time the CPU is used in non-browsing process, (2) filtering of all data sent to the dron system, and (3) anti-spoofing mechanisms to the dron access point. The proposed watchdog timer is designed to protect against DoS attacks and will work in the operating system domain. This will ensure that each non-navigation process is low priority in the tasks (Algorithm 1: line 2) and only allows access to the CPU for a specified period of time τ within a given time interval γ (Algorithm 1: lines 3-8). The functionality of the proposed watchdog timer is illustrated in Algorithm 1. Both τ and γ must be established based on experimental tests to ensure that these parameters prohibit non-navigational processes from opening the CPU, thereby stopping DoS attacks.

The proposed input data filtering security mechanism will operate in the network domain, but will be able to interrupt and terminate non-compliant processes. In other words, no process running on the embedded dron system, which receives input data remotely, can accept load data with a length greater than αA. The functionality of the proposed data filtering mechanism is illustrated in algorithm 2. Additional tests must be performed on the dron to determine the length of characters to be specified in αA.

Finally, the anti-spoofing security mechanism will be implemented in the OSI Network and Data Link layers on the access point and will be able to eliminate ARP responses with counterfeit IP or MAC addresses. This will ensure that potential controllers do not corrupt the network with misinformation through an ARP Cache Poison Attack. Once the access point detects the JSON (that is, Table 2) sent by the dron to allow the controller to access its control ports, it copies the dron's ARP cache and uses the inputs as a template for the IP address and address MAC of the dron. The IP address and MAC address of the controller validated (Algorithm 3: lines 2-4). The access point can now remove any ARP response packets containing the IP or MAC address of the controller or dron (Algorithm 3: lines 5-17). This will ensure the integrity of the dron and controller interconnect. In other words, layer 2 and 3 of OSI warns the access point to record the IP address and MAC of the dron and the controller, and ensures that no device in the network can falsify them. The proposed functionality of the anti-spoofing mechanism is described in algorithm 3.

Test Bench

The dron used has a dual-core P7 CPU with a quad-core embedded GPU running on a Linux-based system with WiFi, AP, GPS and camera modules. This dron can be controlled by a smartphone.

Experimental Procedure

In this work, we work on three common security attacks. A pentest was performed on the dron and exploits were developed that allow DoS, Buffer-Overflow, and ARP Cache Poisoning to launch attacks on the dron. The steps to follow are:

1. Using NMAP, look for the open ports in the dron
2. (Fig 2).
3. Perform routine flights with the dron and capture network traffic from the attack laptop using wireshark (Fig. 3).
4. Analyze the capture of network traffic to develop an initial fuzzing strategy.
5. Perform the fuzzing on the controller-dron interconnect.

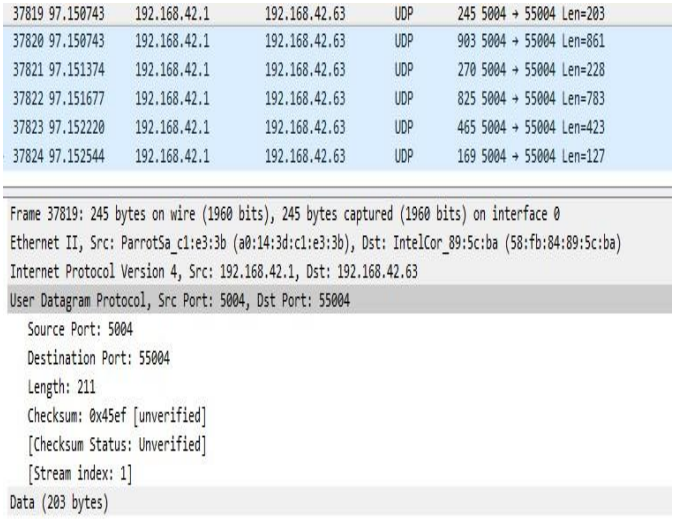

Figure 4. Running nmap under a Linux environment. Source: Authors

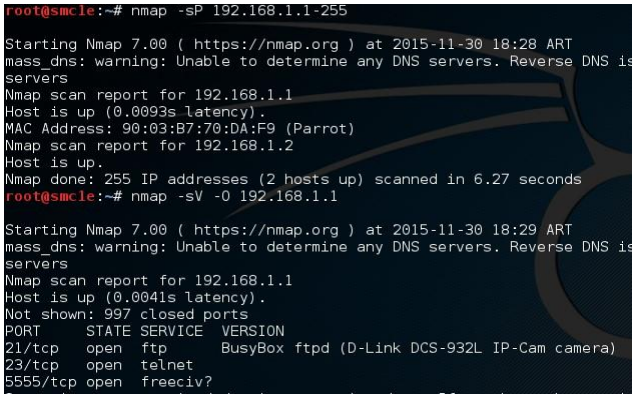

Figure 5. Network traffic monitoring in Wireshark. Source: Authors

Algorithm 1: Watch dog timer

Input: P_i - Non-navigational processes
Output: t - Runtime per P_i in timeframe γ
01: t=0
02: P_i -> low priority
03: **for** each P_i in timeframe γ **do**
04: **set timer t**
05: **if**($t > \tau$)
06: **interrupt** P_i
07: **end if**
08: **end while**
09: **for** each interrupted P_i **do**
10: **sleep** $2*\gamma$
11: **restore** P_i
12: **end while**
13: **return** t

Figure 6. Algorithm 1. Source: [18]

Algorithm 2: Hardline input filtering

Input: F_i - Field from application, packet or frame.
Output: f - Number of characters per field
01: f=0
02: **for** each F_i **do**
03: **count each character** in F_i
04: f++
05: **if**($f > \lambda_A$)
06: **interrupt** P_i
07: **end if**
08: **end while**
09: **return** f

Figure 7. Algorithm 2. Source: [18]

Algorithm 3: Access point anti-spoofing

Input: I_U & M_U – IP and correlated MAC address for UAV
I_C & M_C – IP and MAC of current controller
Output: 1 if I_i or M_i are spoofed
01: e=0
02: **if JSON c2d_port record** contains (status == 0)
03: **copy** UAV ARP cache
04: **end if**
05: **for** each I_i **do**
06 **if ARP Reply** contains ($I_i == I_U$ Or $I_i == I_C$)
07: **drop** corresponding packet
08: e=1
09: **end if**
10: **end for**
11:
12: **for** each M_i **do**
13 **if ARP Reply** contains ($M_i == M_U$ Or $M_i == M_C$)
14: **drop** corresponding packet
15: e=1
16: **end if**
17: **end for**
18: **return e**

Figure 8. Algorithm 3. Source: [18]

In the first pentron of the dron, it was discovered that the smartphone performs an ARP search for the MAC of the device with the IP address 192.168.42.1. Then the smartphone sends a JSON to the IP address 192.168.42.1 via port 44444 in the dron, this JSON contains data about the smartphone and configuration parameters. After that, the dron responds with a JSON pointing the controller to port 54321 or denying the request. (The sent JSON records are those shown in the methodology.

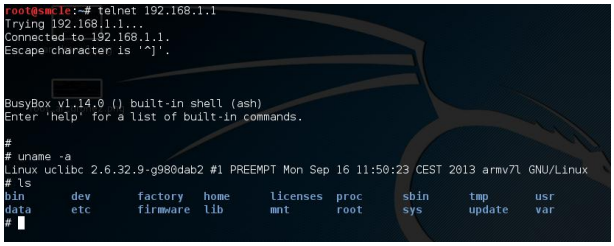

Figure 9. Connection via Telnet with dron. Source: Authors

1. Exploit # 1: Buffer Overflow Attack: The fuzzing method was launched on the dron using the smartphone as controller and also using a script on the attacking computer. A JSON with up to 1000 characters was sent in the first field using the telnet command 192.168.42.1 44444 [variable size JSON]. During the course of the procedure, the attacking PC captured network traffic and embedded system statistics from the / proc / stats folder. It increased from 1 to 1 the length of the entry in the first field of the JSON, starting with 940 until the dron began to lose control (after several tests it was concluded that the dron began to fail near 1000 characters) .
2. Exploit # 1: Denial of Service (DoS): Using the previous method the number of requests sent to the dron through the Telnet tool was increased, the number of requests increased from a 1 (starting at 1) the dron will lock. As in the previous method, network traffic was tracked through Wireshark as illustrated in Figure 3.
3. Exploit # 3: ARP Cache Poison: This method was done with the same base of the previous ones, in this case a script was used using a Python library called Scapy, the idea was to send data to the dron by passing the attacking PC with the IP address 192.168.42.1, the script was executed until the smartphone was disconnected from the dron.

RESULTS AND DISCUSSION

The Pentest made to the controller-dron connection showed that there are at least 3 vulnerabilities present in the dron model used which allowed to develop 3 types of security attacks [17]. These results are an indicator of the vulnerability present in the commercial drones and that an improvement to this problem is required. It is proposed to develop a framework that addresses these security issues.

Exploit #1: Buffer-Overflow

The results of fuzzing in the process of ARDiscovery, increasing the number of characters in the first field of the

JSON sent by a possible controller, shows us that the developers of the dron never considered the possibility of this case.

Now, if there is a driver validated by flying the dron, it is not able to give a good handle to the JSON sent by possible drivers with the first field greater than 950 characters. By capturing the system statistics of the / proc / stats directory on the dron, it was possible to calculate CPU load and memory while the experiments were performed. In figure 8, the performance of the dron CPU in normal use is illustrated. the performance samples are subjected to fuzzing attack. The behavior in both cases is similar, until the dron accepts the large JSON and causes it to crash. At that point the CPU usage decreases to 10%, it may be because the navigation system stopped working. Figure 9 illustrates the performance of the memory samples is under attack and under normal conditions, as in the graph of the CPU we can assume that the performance of this low when the navigation system fails. It is believed that with Algorithm 2 proposed by the authors in [18] can defend the system against these attacks of BufferOverflow.

Exploit #2: Denail of Service (DoS)

The fuzzing attack on the ARDiscovery process with simultaneous requests from potential controllers reveals that the developers did not have this scenario and so the dron is unable to handle 1000 concurrent requests when it already has a valid controller associated and is in flight. Once again the statistics were revised and although the experiment was different, the behavior of the CPU and memory was very similar to that illustrated in Figure 8 and 9. It is observed that the behavior is the same (seen from the perspective of the CPU and memory) until the tests are done, these cause it to deviate from normal use and causes a memory failure, causing the dron to crash and memory and CPU usage drops dramatically to 10%. It can be inferred that exploits cause a bug in the drone navigation application. It is believed that Algorithm 1 (watchdog timer) proposed by the authors in [18] can defend the dron from the DoS attacks.

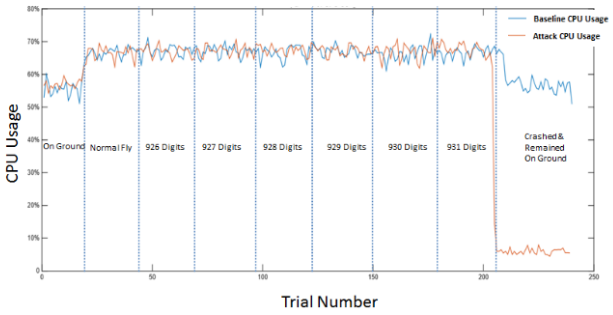

Figure 10. CPU usage. Source: [18]

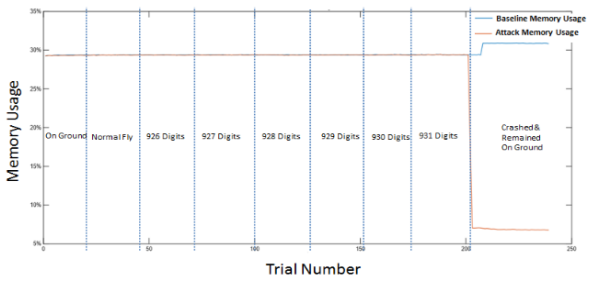

Figure 11. Memory usage. Source: [18]

Exploit #3: ARP Cache Poison

The result of this experiment was a complete untying of the primary controller. When the attacker passes another entity with the dron's IP, it causes a conflict in the dron's wireless network, preventing the primary controller and dron from communicating. After one minute of disconnection between the dron and the controller, the dron takes the behavior described in the manual [15]. It is believed that Algorithm 3 (Anti-spoofing) proposed by the authors in [18] can protect the dron from these attacks of ARP Cache Poisoning.

Bonus

When running the NMAP commands on the dron and see that between the open ports is the Telnet connection and the FTP leaves the user information exposed. Through FTP it is possible to log in as an anonymous user and have access to the images and videos captured by the dron. And the telnet connection leaves us as root users in something that the dron calls BusyBox, as root users we can modify any important files in the system or worse, run the command rm -rf / and leave the dron unusable.

PROPOSED SECURITY FRAMEWORK

Model

Figure 12 shows a 3-part network model that addresses the problems mentioned above in drones. This model can deal with physical and network security issues. In a network environment, the controller can execute motion actions on the dron and obtain images and navigation data through both channels. The Middleware (Application Interchange Information Logic) connects directly to the dron and performs an authentication process with the controller. The Middleware manages two communication channels and detects intrusions of potential attackers through the primary channel. When an intrusion is detected, an attack will be sent to the primary channel to cause the attacker to lose control of the intruder.

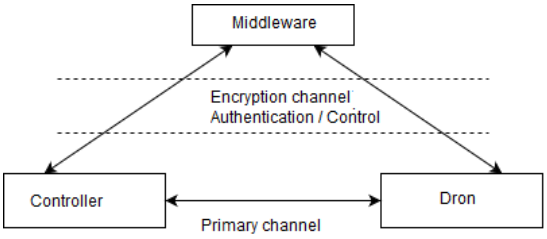

Figure 12. Proposed model of the system. Source: Authors

In Figure 13, before the dron starts its flight, the controller calculates values using the time zones and creates a public key with AES encryption. Then the controller sends to the dron an array with random values and the public key, the dron is ready to fly once the controller verifies that the dron received the information correctly.

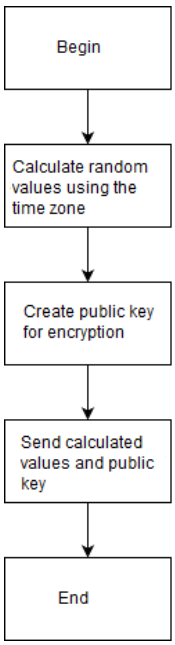

Figure 13. Synchronization process for pre-flight authentication. Source: Authors

Figure 14 shows the algorithm proposed in this paper. When the flight starts, the dron periodically starts to authenticate with the controller. The primary channel of the dron is monitored whether it is attacked or not during communication with the controller. The controller will detect if the dron has been attacked by any device in the dron's network. The controller then sends the dron a signal to deactivate the primary channel in order to prevent a leak of information through the hacked channel. The dron lands and terminates communication with the controller, after which, it initiates the authentication process between the dron and the controller again.

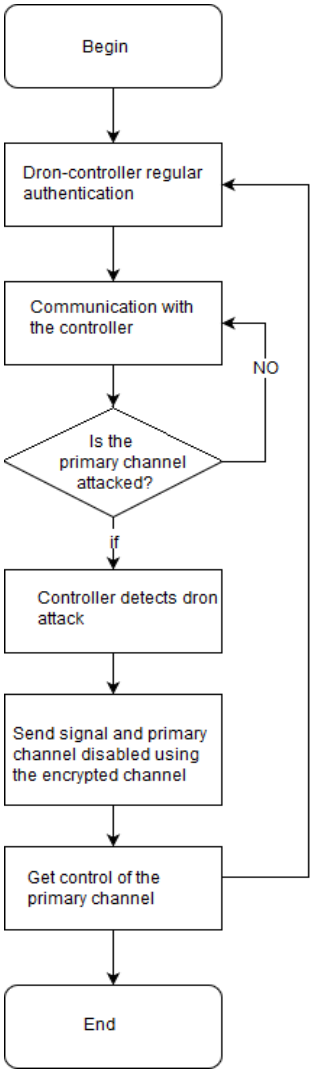

Figure 14. Proposed system algorithm. Source: Authors

Figure 15 shows an authentication algorithm when the dron wants to initiate or maintain a communication channel with other devices during flight. In many cases, the dron collects information that it later sends to the controller. Consequently, the transmitter is the dron and the receiver is the controller. First of all, the sender generates an encrypted random array index to send the information to the receiver. When the receiver receives the information decrypts the index using the public key.

The receiver then encrypts the value in the array index that was synchronized before the flight started and then sends it to the sender. The sender receives the information sent by the receiver and decrypts it, once decrypted it compares it with its value in the array index. If both values are correct, authentication completes successfully and the authentication process continues as long as the dron and the controller maintain communication. However, if both values are incorrect, authentication fails. When authentication fails, the sender disconnects this channel and returns to the first step. In this process, the proposed algorithm can solve a possible Reflection attack caused in a bidirectional way by a challenge-response protocol. The attacker easily obtains the public authenticator

key by reflection. In contrast, this algorithm can prevent attacker intrusion by processing authentication with the encrypted index value internally.

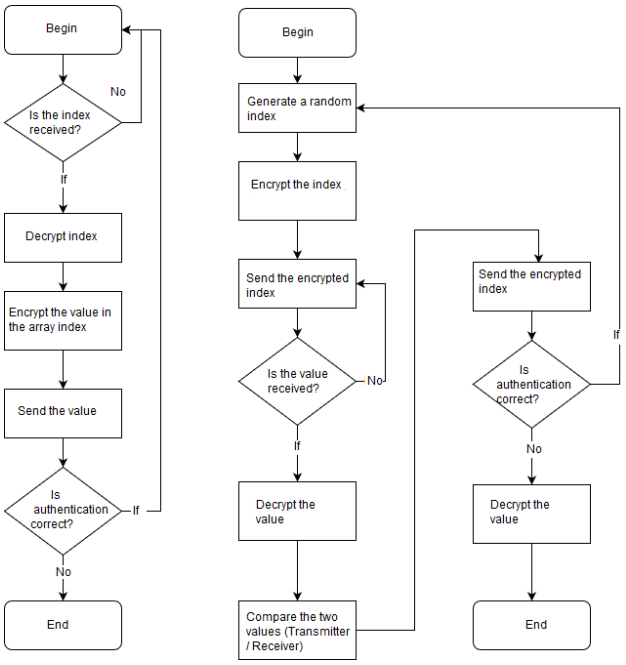

Figure 15. Proposed system algorithm. Source: Authors

CONCLUSIONS

A framework should be developed with the algorithms proposed above and that also involves security as to the active and open services in the dron. It is possible that the authentication system fails and if so the services would be exposed. This is why, in addition to the authentication algorithms, the algorithms responsible for protecting against Buffer Overflow, DoS and ARP Cache Poison attacks must be implemented. With this will be possible to mitigate the possible damage that can cause some people taking advantage of these failures. Thanks to the analysis made and the tests executed it was possible to confirm that these failures are still in force and apparently are not the priority of the developers, being a commercial type dron is not supposed to give a special use as possibly is given to a military type dron.

In any case, the development of the framework will defend the drones and in turn the users of possible threats and will make the use of these devices become even more popular. In a future work, it will be possible to take statistics of the operation of the framework proposed in this paper and verify that it was possible to mitigate these attacks by limiting access to force to the device. As discussed in section 2, Related Jobs, there is currently no development involving the safety of these drones and it is not possible for us to take related data. It is expected that in the near future the producers of the different drones of the market take into account these security flaws and develop the appropriate software to avoid them.

REFERENCES

[1] C. Anderson. (2012) ”How I Accidentally Kickstarted the Domestic UAV Boom”. Wired.com. Available: http://www.wired.com/2012/06/ff UAVs/

[2] J. Booton, ”UAV demand sizzles heading into the holidays,” in MarketWatch, ed. [Online], 2014.

[3] ”FAA needs to make thoughtful safety rules before UAVs deliver our packages or pizza,” in Los Angeles Times, ed. [Online], 2015.

[4] S. Gallagher, ”Parrot UAVs easily taken down or hijacked, researchers demonstrate,” Ars Technica, 2015.

[5] D. Crockford, ”The application/json media type for javascript object notation (json),” 2006.

[6] S. Kamkar. samyk/skyjack [Online]. Available: https://github.com/samyk/ skyjack

[7] K. Moskvitch, ”Are UAVs the next target for hackers?”, in BBC, ed, 2014.

[8] J. Rinke, ”Take Control of Paired ARUAV”, ed. YouTube.com, 2013.

[9] E. Deligne, ”ARDrone corruption,” Journal in Computer Virology, vol. 8, pp. 15-27, 2012.

[10] J. Pleban, R. Band, and R. Creutzburg, ”Hacking and securing the AR.UAV 2.0 quadcopter - Investigations for improving the security of toy”, in The International Society For Optical Engineering, 2014.

[11] S. M. Giray, ”Anatomy of unmanned aerial vehicle hijacking with signal spoofing”, in Recent Advances in Space Technologies (RAST), 2013 6 th International Conference on, 2013, pp. 795-800.

[12] B. Brock and K. Rajamani, ”Dynamic power management for embedded systems”, in IEEE International Systems-on-Chip (SOC) Conference, 2003, pp. 416-419.

[13] ”FreeFlight 3”, ed: Parrot S.A., 2015.

[14] H. Zimmermann, ”OSI reference model–The ISO model of architecture for open systems interconnection”, Communications, IEEE Transactions on, vol. 28, pp. 425-432, 1980.

[15] Parrot Bebop User Guide. (2016) Available: http://www.parrot.com/usa/ support/parrot-bebop-drone/.

[16] S. Institute, ”Defense In Depth: SANS Institute Reading Room”, SANS Institute, [Online] 2015.

[17] J. Burns. (2016) Johns Hopkins Team Hacks, Crashes Hobby Drones To Expose Security Flaws Forbes.com. Available: https://goo.gl/cpm2aW.

[18] M. Hooper, Y. Tian, R. Zhou, B. Cao, A. Lauf, L. Watkins, W. Robinson and W. Alexis, ”Securing Commercial WiFi-Based UAVs From Common Security Attacks”, MILCOM, 2016.