GENERIC AND AUTONOMOUS SYSTEM FOR AIRBORNE NETWORKS CYBER-THREAT DETECTION

Silvia GIL CASALS, LAAS-CNRS, Univ. de Toulouse and THALES Avionics, France
Philippe OWEZARSKI, LAAS-CNRS and Univ. de Toulouse, France
Gilles DESCARGUES, THALES Avionics, Toulouse, France

Abstract

Cyber-security on airborne systems is becoming an industrial major concern that arises many challenges. In this paper, we introduce a generic security monitoring framework for autonomous detection of cyber-attacks on airborne networks based on unsupervised machine learning algorithm. The main challenge of anomaly detection with unsupervised techniques is to have an accurate detection since they tend to produce false alarms. After evaluating the suitability of the One Class SVM, we propose some hints to improve detection accuracy of the monitoring framework by collecting information from the airborne architecture.

Introduction

Cyber-security on airplanes is becoming a novel issue to be considered by airframers and manufacturers. As a matter of fact, air traffic management facilities and on-board devices have already been targeted: viruses spreading on airline Electronic Flight Bags [1], fighter planes [2] or drones [3-4] grounded by computer viruses, incidents due to misused maintenance laptop tools [5], potential backdoors on chips [6], etc. Sooner this year, Hugo Teso created a high expectancy at the Hack in the Box conference [1] pretending that hijacking an aircraft from ground through a simple smartphone application is possible. Both EU and US certification authorities EASA [2] and FAA [3] address Certification Review Items and Special Conditions so security issues are appropriately considered before delivering the Type Certificate. Also, standardization organizations such as EUROCAE's[4] working group WG-72 [5] and RTCA's [6] special committee SC-216 [7] have started to write the new airworthiness security standards: ED-202/DO-326 [7] and ED-203/DO-xxx [8] that will provide specifications and guidelines for the development and certification of safe and secure airborne systems. The International Federation of Air Line Pilot's Associations (IFALPA) laid the stress in a report [9] on the fact that data could be corrupted during transfer from a network to another, as for instance it is provided on a non-secure form via ACARS. It underlines the need for a list of security countermeasures and among them for monitoring systems to "look for things that don't belong".

Contrary to the safety domain, there is no experience feedback on aircraft security attacks and their consequences unless by extrapolation from the IT domain. Indeed, observing cyber-threats is thus necessary to acquire a basic knowledge on their nature, conditions and frequency of occurrence, justify the most suitable countermeasures while avoiding over-design, evaluate their effectiveness and determine on a less subjective manner, what is called the "attack likelihood" in our risk assessment methodology [10]. Considering the perennial design constraints of airplanes, it implies having "long-term security solutions" [11], in this case, for anomaly detection. It seems paradoxical to design a perennial security audit solution within the permanent evolution of cyber-attacks in a stable, safe and deterministic context such as aeronautics.

As flight crews are not meant to be security specialists, threats detection must be done on real-time and as autonomously as possible. Machine Learning (ML) techniques appeared to be the most suitable for the security audit system as they can handle huge quantities of data autonomously without needing any signature or update and, contrary to

[1] Available at http://youtu.be/wk1jIKQvMx8
[2] European Aviation Safety Agency
[3] Federal Aviation Administration
[4] European Organization for Civil Aviation Equipment
[5] http://www.eurocae.net/working-groups/wg-list/41-wg-72.html
[6] Radio Technical Commission for Aeronautics
[7] http://www.rtca.org/comm/Committee.cfm?id=76

other Intrusion Detection Systems such as anti-virus, ML-based ones are able to detect novel attacks. This paper introduces a generic security monitoring framework for autonomous detection of cyber-attacks on airborne networks based on One Class SVM, an unsupervised ML technique. However, ML is still a research topic where main concern is detection accuracy, due to the risk of false alarms.

The paper is structured as follows: Chapter 1 describes the feared security threats among new aircraft architecture, Chapter 2 defines Intrusion Detection Systems and machine learning algorithms. Then, Chapter 3 explains the basic framework of the security audit function. Chapter 4 shows the detection results obtained. Chapter 5 gives some hints of for a more accurate detection integrated in the airborne environment. Finally, Chapter 6 concludes by opening to other perspectives.

Architecture Evolutions and Threats

Actual and Upcoming Innovations

In new aircraft, three networks can mainly be considered: the Aircraft Control Domain (ACD) network, the Airline Information Services Domain (AISD) network and the Passenger Information and Entertainment Service Domain (PIESD) network. A simplified architecture is shown at figure 1.

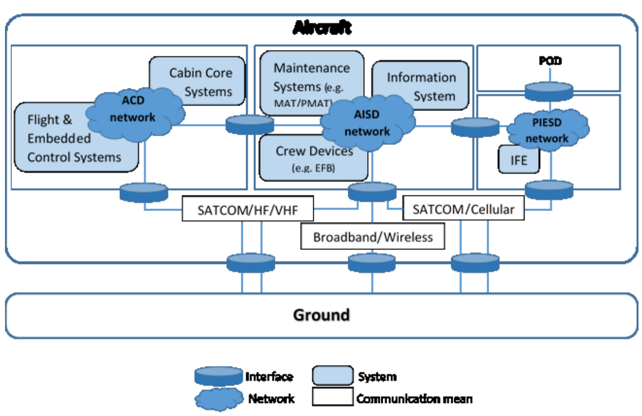

Figure 1. Simplified Generic Architecture of On-board Networks

Originally, these networks used to be strictly segregated and had their own link to ground facilities but if we take a closer look to the evolution of some airborne systems we can notice they tend to be increasingly interconnected. To improve and extend airline services while reducing costs, airborne systems tend to become generic, to share resources or air-ground communication channels and to require more interactions with off-board systems:

- Offering new communication services to passengers, allowing media broadcast through WiFi and Internet access from In-Flight Entertainment (IFE) units or Passenger Owned Devices (POD),
- Using general purpose devices such as iPads to replace paper flight manuals (approved in cockpits by the FAA),
- Replacing Line Replaceable Units by Integrated Modular Avionics with field loadable software to optimize maintenance dispatch,
- Increasing the use of Commercial Of The Shelf (COTS) equipment to reduce development time and costs,
- Homogenizing communication protocols to allow inter-operability between on-board networks, COTS and ground systems (e.g. Newsky project [12] and the evolution from ATN/OSI to ATN/IPS),
- Migrating from ground-based voice to satellite-based data exchange to avoid radio voice communications drawbacks (e.g. frequency saturation, sectors coordination),
- Migrating from FAA's National Airspace System to the Next Generation Air Transportation System (NextGen) for Air Traffic Management with massive information gathering,
- In the future, airplanes are meant to become intelligent communication nodes to ease Air Traffic Control tasks and increase aircraft autonomy so they can engage in free flight in remote areas, self-optimizing their trajectory and choosing their own route.

Feared Attacks

By using COTS, potential known vulnerabilities exploitable by an attacker can be introduced. Depending on network and interfaces design, the attacker could gain access to any other equipment of the network and then cause interference on inter-systems interfaces or air/ground communications if segregation with safety critical networks is not ensured or critical data corruption. We can separate threats into two groups: host-level threats and network-level threats.

Host-Level Threats

Host-level threats affect a device but might not be detectable from the network. If human-machine interfaces of airborne devices have not enough authentication means (at user, software and hardware level), examples of threats[8] are:

- Access gain by unauthorized person (e.g. password replay or cracking) either to enter the system (by exploiting host vulnerabilities) or to get sensitive information about the device or its configuration,
- Denial (e.g. disable / block access to human-machine interface of the device),
- Install corrupted software, viruses, worms, etc.

Host-level attacks are out of the scope of this paper, taking them into consideration would require having a monitoring agent on each host of the network, and then generate traffic to centralize all the monitored logs. In this proof of concept study, generating traffic must be avoided in order not to be intrusive in the airborne devices nor in the networks. We will thus concentrate on packet captures analysis.

Network-Level Threats

If the network does not have any countermeasure such as firewalls or filters correctly implemented, some of the network-level attacks likely to occur in such a complex system are:

- Footprinting: accumulate sensitive information about the network and its vulnerabilities (scans),
- Denial of Service attacks: either send huge quantities of data to saturate routers/switches or sending malformed packets to see network elements reaction (e.g. flooding, teardrop, fuzzing),
- Spoofing attacks, i.e. present altered content as legitimate for example by re-injecting previously captured packets with alterations,
- Introducing unauthorized devices in the network,

[8] A more extended list is given by the ARINC 811 [13].

Assuming that the three networks are connected to a same Ethernet switch and separated using VLANs, [14] considers that Denial of Service (DoS) and hacking are plausible on aircraft as VLANs are vulnerable to attacks such as MAC flooding, frame tagging, ARP poisoning, multicast brute force, VLAN hopping due to spanning-tree protocol, random frame stress, etc. It proposes to use IPsec encryption to secure air-ground and inter-networks communications within the aircraft, as well as an anomaly detection engine to monitor network traffic looking for "any deviation from the normal behaviour". Every monitored packet is given an anomaly score: the fewer times a type of packet has occurred, the higher is the anomaly score, all scores help upgrading an occurrence probability table. However, the monitoring system still raises a considerable number of false alerts.

Background Theory

Intrusion Detection Systems (IDS)

There are two types of IDS: on the one hand, "misuse detectors" are based on the study of the abnormal behaviour of the network, i.e. threats are detected by comparison to a given signature pattern established manually by a security expert (for instance anti-virus). On the other hand, "anomaly detectors" are usually based on measures of deviation from the normal behaviour of the network. Misuse detectors do not fully suit to the aeronautics context because they fail in detecting novel attacks, they require frequent update of threat signatures databases which takes time and is costly. Machine Learning is an Artificial Intelligence domain studying the algorithms that allow the adaptation of the behavior of a machine after a learning phase. For the learning phase, it uses data mining techniques such as statistics and clustering for the analysis of a huge quantity of data in order to extract knowledge and eventually a model. Learning is said to be effective if it is descriptive (captures learning data), predictable (generalizes the model for its application on unknown data), and explicative (describes on an understandable way the learned concepts). Anomaly-based network IDS can use two ML techniques to classify traffic, either through supervised or unsupervised learning. The latter has no assumptions on the input data set and performs automatic classification to extract useful information, whereas the former requires a training step with a labelled set of data to establish a model before starting to predict labels for new data.

It has to be noticed that ML techniques are under research to be embedded on airplanes for other goals such as performance monitoring [15], fuel consumption prediction [16] or intelligent processing of sensors data in unmanned autonomous systems in the airspace [17], etc.

Some Clustering Algorithms Principles

Clustering is an unsupervised exploration data technique that consists in gathering similar samples together into subsets or clusters. Here is a short explanation of the main algorithms we tested for a preliminary evaluation.

K-means

K-means clustering aims at classifying the samples into K clusters, K being an input parameter. First, K samples are arbitrarily chosen as seeds, and the rest of samples are assigned to the nearest seed (by calculating Euclidean distance for instance), obtaining K initial clusters. Then, the centroid of each cluster is calculated and the samples are reassigned to the K centroids. The algorithm iterates until the centroids converge to a local optimum position.

DBSCAN

Density-Based Spatial Clustering of Applications with Noise [18] is a density-based clustering algorithm. It requires two parameters: ε, the reachable neighborhood and *MinPts* the minimum quantity of points to be in a cluster. It starts with an arbitrary point and looks for all other ε-reachable points from it. If the point has enough neighbors, then the cluster expands, but if not, the point is considered as noise. DBSCAN finds clusters of any shape or size, but has much better performances in low-space dimensions, that is why it is suggested to make subspace clustering [19]. However, its main drawback is that it does not cluster with the same accuracy whenever there are significant density variations in the feature space.

Mean Shift

Mean Shift [20] is another density-based iterative clustering algorithm that looks for the densest regions of the feature space. It starts with an arbitrary initial centroid y_0 and a region of interest of a given radius x. Then, the center of mass of the interest region is calculated by simply adding all its point's coordinates and dividing it by its number of points. Then, the interest region shifts on the center of mass and so on until reaching the maximum density in the local neighborhood. Its clear advantage compared to other algorithms is that it is non-parametric: contrary to K-means, it does not require the number of clusters to be extracted, nor other parameters such as DBSCAN. Also, Mean Shift does not constrain clusters' shape contrary to radial-search based algorithms where shapes are rather elliptical. However, Mean Shift is not always able to correctly classify all kind of cluster shapes, particularly when the density in the feature space is too high.

Support Vector Machines (SVM)

Original SVM Principles

SVM are supervised learning models to recognize patterns for classification. They require two steps: training and testing. Each sample of the training set contains features (observed variables that characterize the samples) and a class label (e.g. "normal" vs. "anomalous"). From the training data set, the goal is to produce a model based on training data that predicts the class labels of the test data given only its features. The algorithm looks for the optimal hyperplane that maximizes the separation margin between the 2 classes and minimizes the number of errors. The advantages are that they work with high dimensions and have same or better performances than neural networks or Gaussian Mixture Model.

Given a data set $S=\{(x_1,y_1), (x_2,y_2), ..., (x_n, y_n)\}$ where $x_i \in R^d$ is a feature vector and $y_i \in \{-1,1\}$ the label. The goal is to find a linear function[9], where the decision boundary: $h(x) = w^T + w_0 = 0$ gives the equation of the separation hyperplane. The distance between this hyperplane and the closest points in the data set (also called support vectors) is maximized,

[9] where w^T is the weight vector and w_0 the offset.

and labels are given by the sign of the function: h(x) ≥ 0 ⇔ class 1 and h(x) ≤ 0 ⇔ class -1.

If the problem cannot be solved linearly then data is transformed into a higher dimensional one where the linear classification is possible. SVM usually works with four basic kernels: linear, polynomial, radial basis function (RBF) and sigmoid, but many others are under research.

One Class SVM

One Class SVM is the unsupervised version of the original SVM algorithm for novelty detection. Its main advantage is that it does not require the class labels of the samples nor a training step. Tax and Duin [21] presented the Support Vector Data Description (SVDD) that consists in looking for the minimized circumscribing hypersphere around the data in the feature space. The hypersurface is characterized by a center **a** and a radius **R**>0 being the distance from the center to any point on the boundary (support vector). The algorithm returns 1 if data is inside the hypersurface and -1 elsewhere.

Algorithm Choice Justification

Given that the anomalous traffic is statistically different from normal one [22] and that the majority of traffic is supposed to be normal, we assume that normal traffic is supposed to be included in clusters containing the majority of samples. The anomalies are supposed to be outliers, i.e. single samples with no immediate neighborhood, or included in small clusters.

After testing the three previous clustering algorithms on flight simulators traffic captures, we have noticed that none of them was able to detect outliers as all events (even anomalous ones) were included in clusters of at least 10 samples, which is an unacceptable detection rate. Indeed, figure 2 shows Mean Shift clustering results using two dimensions where an anomaly was clearly observable (fuzzing). We notice the anomaly is a false negative (i.e. undetected threat) as it is included in the upper cluster containing 19 samples (please refer to "proof of concept" chapter for more details on data sets and anomalies).

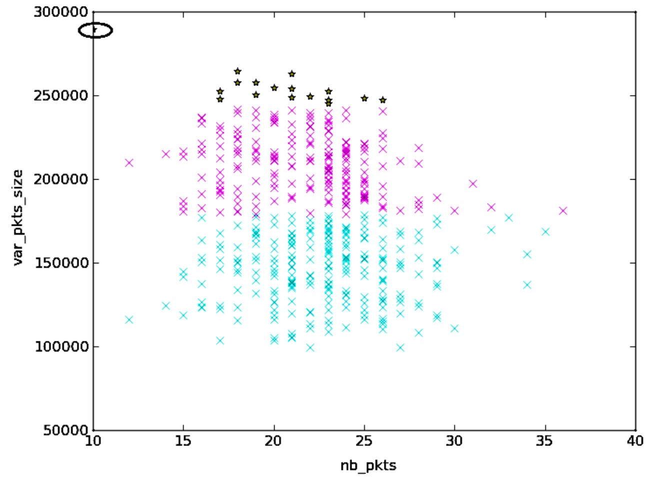

Figure 2. Mean Shift Clustering Results

But figure 2 also shows that our feature space is not very sparse with a quite dense region easy to delimit. That is the reason that made us look for a supervised-like solution with no need for initial training step with labels such as One Class SVM.

Basic Audit Function Framework

In this chapter, we detail the different steps of a framework to detect network anomalies from packet captures using the OCSVM algorithm summarized in figure 3.

1. Data Acquisition Step

Traffic Capture

Packets are captured continuously on a network node, in our case, on the gateway interface between the maintenance systems of AISD network and the ACD network. This function as well as packet analysis for step 2 are performed using the Python tool Scapy (www.secdev.org/projects/scapy) which allows either to directly capture the traffic from the network (like Wireshark) or to import traffic traces from previously captured .pcap files.

The Observation Window

The characteristics of a network attack are observable given a set of packets rather than based on single-packet observation. Thus, we need to determine the observation window ΔT on which the attributes will be computed. In Avionics Full DupleX switched Ethernet (AFDX) used in ACD networks, most of aircraft messages are periodical, we simply need to find the smaller period between synchronous messages Tm and we apply Nyquist-Shannon's sampling theorem: $\Delta T \leq T_m / 2$. The minimal period between periodic AFDX frames being of T_{mAFDX}=50ms, we chose an observation window of ΔT_{AFDX}=20ms so packets contained in this time slot can be considered as representative enough of the AFDX traffic. The same has been done for the Ethernet side where T_{mEth}=100ms and we have chosen ΔT_{Eth}=50ms.

Logs and Identification

The packets of each observation window are kept in memory (RAM) and are identified by the timestamp of its first packet. Timestamp is crucial as it is the only basis for future correlation with other detected events in the aircraft. To optimize the use of memory resources, packets are kept until the third step (detection) determines whether:

- an anomalous event occurred during ΔT: in which case the set of packets will be stored in a Non Volatile Memory, as well as the previous and the next ΔT for further analysis (step 4 and forensic)
- ΔT traffic is normal: in which case the set of packets will be removed from the RAM

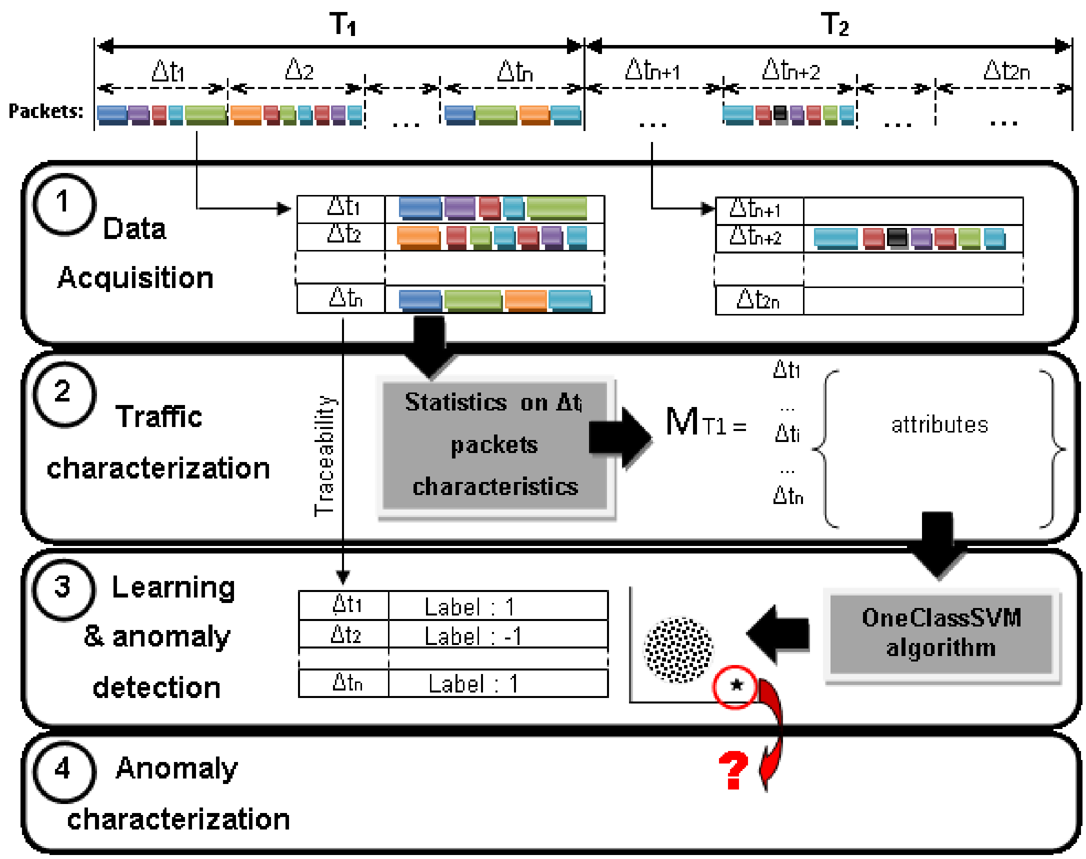

Figure 3. Audit Function Framework

2. Traffic Characterization Step

One of the most crucial steps when using machine learning techniques is the attributes or features definition. Attributes are network traffic characteristics to feed a machine learning algorithm so that common samples are grouped by similarity. They are set upon statistics based on packets properties for every observation window ΔT. Every ΔT corresponds to a sample, i.e. a vector of attributes. Table 1 gives a list of standard attributes commonly used for well-known cyber-attacks detection:

Table 1. List of Traffic Attributes

Name	Description	ACD	AISD
nb_pkts	Total number of packets to detect high variations in traffic volume	X	X
nb_ip_src	Number of different IP source addresses (hosts) exchanging packets to detect the omnipresence of a single host sending traffic to the network (DoS)	X	X
nb_ip_dst	Number of different IP destination addresses (hosts) exchanging to detect network scans if single source sending to several destinations or DDoS[10] if several sources sending to a single destination.	X	X
nb_mac_src	Number of different MAC source addresses (same as IP src but on another layer)	X	X
nb_mac_dst	Number of different MAC destination addresses (same as IP dst but on another layer)	X	X
nb_arp	Number ARP packets to detect arp-based attacks		X
nb_icmp	Number ICMP packets to detect icmp-based attacks		X
nb_tftp	Number TFTP packets to detect tftp-based attacks		X
nb_snmp	Number SNMP packets to detect snmp-based attacks	X	X
nb_frag	Number of fragmented packets to detect teardrop attacks	X	X
nb_pkts_max_per_sport	Maximal number of packets sent by every source IP address' ports to detect DoS	X	X
nb_pkts_max_per_dport	Maximal number of packets sent by every destination IP address' ports to detect DoS	X	X
nb_ports_max_per_IP_dst	Maximal number of destination ports used for each IP destination address to detect port scans	X	X
nb_ports_max_per_IP_src	Maximal number of source ports used for each IP source address to detect DoS or virus output from single port	X	X
min_size	Minimum packet size to detect anomalous-sized packets	X	X
max_size	Maximum packet size to detect anomalous-sized packets		X
avg_size	Average packet size to detect anomalous-sized packets	X	X
var_size	Variance of packets size to detect scans or other attacks that have no significant payload variations	X	X

[10] Distributed Denial of Service: attacks aiming at rendering a network resource unavailable.

Note that some of the attributes only apply to the Ethernet side. For instance, in AFDX the *max_size* of a packet is a fixed value of 1492 bytes, also some protocols are not supported.

The set of attributes constitutes the feature space of the model. It is known that the bigger the feature space, the more computing resources are required for data processing. The amount of input attributes for the machine learning algorithm can be reduced by performing a preprocessing feature selection step to discard non-significant attributes. However, the techniques used, filters (based on information gain calculation) and wrappers (guided by accuracy measures), require the previous knowledge of labels. As we have not tested a large amount of attacks and that the goal of our audit system is to find the anomaly signature, we have decided to keep all the attributes until proving that some of them are useless for classification.

3. Learning and Anomaly Detection Step

We have chosen the Sklearn[11] Python machine learning library that offers an important quantity of both supervised and unsupervised algorithms, as well the previously mentioned tools for feature selection. More precisely, we use the OneClassSVM (OCSVM) algorithm for the learning step.

Learning Period

Main challenges in all IDS are accuracy and real-time detection. Machine Learning algorithms require a certain amount of samples to correctly perform clustering and accurately detect deviations. However, we already need to gather packets during ΔT to build one single sample! Once again, we performed grid search to determine the minimum number of samples required to launch OCSVM without errors while having the minimum false positive rate on clean traffic. We found out that the algorithm requires at least n=50 samples (or ΔT slots), which sets the minimum learning period to T_{learn}=1s for AFDX and 2,5s for Ethernet networks.

OCSVM Parameters Determination

Before starting to use the OCSVM for anomaly detection, we have performed a grid search to determine the algorithm parameters that minimize the number of false positives by applying it on a normal traffic capture.

In practice, the OCSVM algorithm takes a matrix of dimensions [number of attributes × number of ΔT slots] as input and returns an array of classification labels y for each ΔT (y=1 if normal, y=-1 if anomaly). The time-stamp of each ΔT is kept as a unique identifier for traceability between the anomalous slots and their labels.

4. Anomaly Characterization Step

Once the anomalous observation window has been identified and isolated, the anomaly characterization step consists in extracting the attribute(s) that significantly separate the sample considered as anomalous from the boundary support vectors of normal traffic. This is done by using the feature importance calculation tool of the ExtraTreesClassifier algorithm of Sklearn that takes as inputs the features matrix and the classification labels and that returns importance values for each one of the features. The resulting important attributes above a given threshold will be the signature of the anomaly.

[11] http://scikit-learn.org/stable/

In Practice: Proof of Concept

The Testing Data Set

To make the previous evaluations (OCSVM parameters determination), we have used an extended Wireshark traffic capture of maintenance operations on ground on the AISD side of the gateway between AISD and ACD networks. This traffic is assumed to be clean, i.e. with no anomaly. On the other hand, we copied some packets from the clean set that we forged into attack packets with Scapy and we re-injected them into the clean data set. We assume the threat scenario consists in getting access to the Maintenance Access Terminal and performing the following attacks on the AISD/ACD gateway:

- ARP scan: the Address Resolution Protocol (ARP) is a protocol that translates IP into MAC addresses. The attack consists in broadcasting "ARP-who-has" requests asking for a list of IP addresses to map all devices of a network by getting both their IP and MAC addresses. For instance, to map some devices from the ACD: Flight & Embedded Control Functions, we have scanned a portion of IP addresses from 10.0.0.0 to 10.127.255.255, then, the devices having one of these IP addresses answer back by giving their MAC address.
- Port scan: probing attack to search for open ports on devices. It consists in sending many times a same legitimate packet to a given device using each time a different destination port.
- Fuzzing: consists in sending huge quantities of all kind of invalid packets to see if it has an impact on any device, especially to saturate routers, switches or gateways.
- Teardrop: consists in sending overlapping fragmented packets to a targeted device to induce it to an anomalous behavior or to crash it.

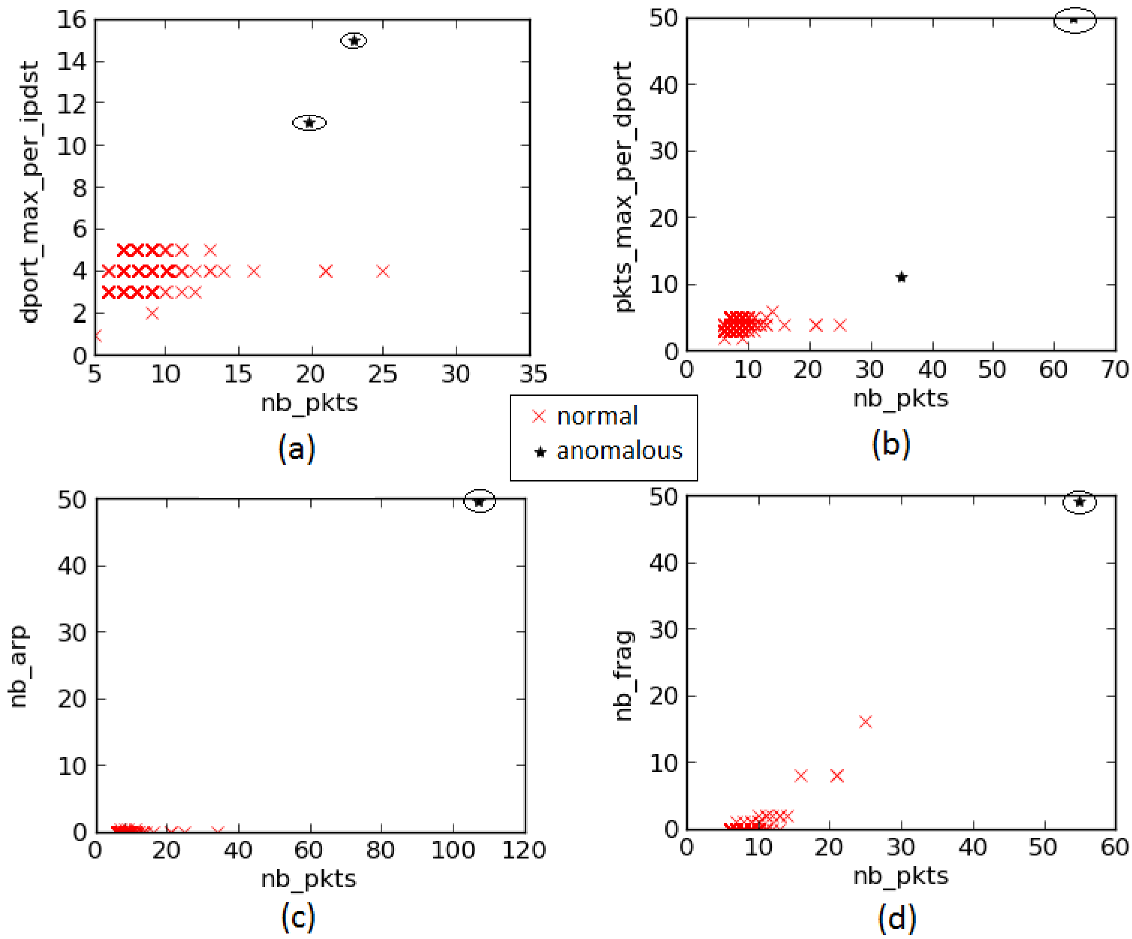

Figure 4. OCSVM Results Respectively for Portscan Attack (a), Fuzzing Attack (b), Arpscan Attack (c) and Teardrop Attack (d)

Anomaly Detection Results

Contrary to our tests with clustering algorithms where we barely detected the real anomaly in the middle of false positives, here, in each of the four cases, the anomaly is clearly detected. To ease visualization, we traced 2D plots with the Python library matplotlib (figure 4) of the OCSVM algorithm results taking as dimensions the two most significant attributes (those having the higher importance scores at step 4). In (a) there are not two anomalies, it is the same one being detected on two successive ΔT time slices. As it can be observed in (b) there is one false positive, corresponding to the Maintenance Access Terminal legitimate ARP requests when starting the maintenance operations.

Some Hints for an Audit Integrated Architecture

As we have seen, the main problem of machine learning algorithms is the risk of false alarms. They can be originated by:

- the occurrence of rare events such as the one we have noticed in the previous chapter,
- the traffic nature variations during transitions between flight phases,
- eventual safety failures than can cause unusual emergency traffic in the network,

Hereafter, we give some hints to improve detection accuracy.

Setting Normal Flight Phase Traffic Profiles

Traffic varies slightly depending on the flight phase (e.g. maintenance operations are only allowed on ground). This could bring a lot of false positives during the transition from one phase to another. It is thus necessary and possible to determine the normal traffic behavior profile for each flight phase because our feature space is not especially sparse as shown in figures 2 and 4. It consists simply in applying the three previous steps of the framework offline, on clean traffic that does not present any anomaly (typically test bench traffic captures) for a very long period. This way, we can be sure to observe rare but legitimate events. Thanks to the OCSVM algorithm, we get the support vectors (i.e. the boundaries) as well as the centers of the "macro-cluster" of normal traffic for each flight phase. Whenever an anomaly is detected during a flight phase transition, it must be verified whether this anomaly is included in the next flight phase "macro-cluster" shape.

Redundant and Dissimilar Combination of Machine Learning Algorithms

To improve the detection accuracy, we take inspiration from safety avionic architectures to propose the redundant and dissimilar framework shown in figure 5. It consists in capturing traffic and building the correspondent attributes adapted to ACD, AISD and PIESD network traffic. The dissimilar concept consists in detecting anomalies by running independently different machine learning algorithms or at least with different configurations on these attributes. Also, logs from each network's devices can be added to trace the attack path from the initial attacked device to network consequences. Finally, the anomaly detection results of the different algorithms are correlated to obtain a more accurate anomaly detection. Whenever an attack is detected simultaneously on two or more of the networks, it provides more assurance of the detection consistency and would directly set the rank of the security alarm at a critical rate.

Concerning the introduction of devices access logs, N.K. Rao [23] set in 1989 the specifications of an audit function for embedded avionics systems, in order to detect and deter penetration of security controls by unauthorized subjects. It uses a Trusted Kernel or Trusted Computing Base that keeps traces of access and operations performed on embedded devices by subjects (e.g. subject ID, action performed, parameters used, status before and after the operation, resource usage, etc.). Whenever the frequency of a certain type of operations is above a given threshold, it is able to detect security measures bypassing or escalate privileges. These thresholds were determined by statistical observation of "normal" operations frequency.

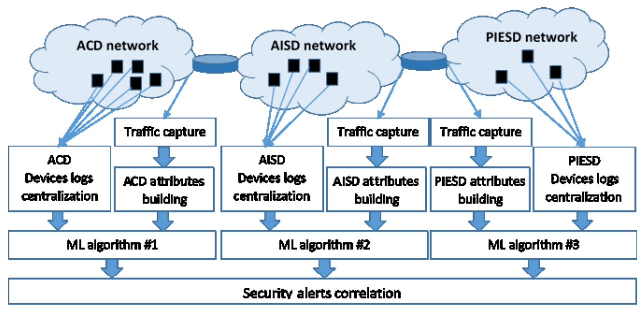

Figure 5. Redundant and Dissimilar Framework

Correlation with Safety Events

Establishing a correlation between security anomalies detection and safety failure events has a dual interest:

- On the one hand, if a safety failure occurs and shortly after a security anomaly is detected, it is very likely that the traffic considered as anomalous has been triggered by the failure and that it is legitimate emergency traffic.
- On the other hand, to go deep into the potential attack consequences, whenever a security anomaly is detected, attention must be paid to posterior failure events in order to determine whether the attack caused a safety impact.

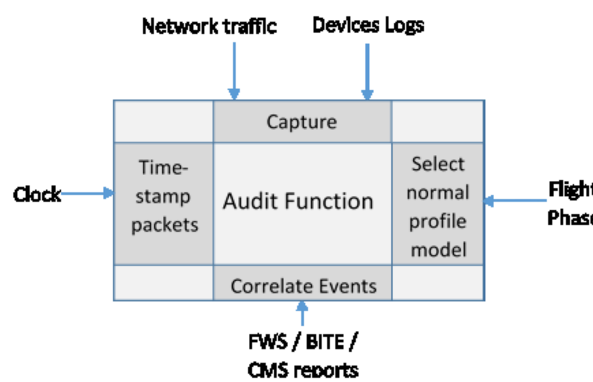

Figure 6. Audit Function Interfaces

To do so, the security audit system should have an interface with the Flight Warning System, with the BITEs (Built-In Test Equipment) or receive reports from the Centralized Maintenance System. Another crucial point is to share a common and rigorous time-stamping of captured packets with the safety events monitoring (see figure 6 audit function interfaces).

Conclusion

Airworthiness security is a brand new domain arising challenges such as real-time constraints for attack detection, designing security countermeasures that do not compromise safety, among other performances. In this paper, we have shown a generic anomaly detection framework based on the One Class SVM unsupervised machine learning algorithm that could be integrated in airplanes for autonomous cyber-attack detection. Some hints have been given on how to improve its accuracy by setting normal traffic profiles for each flight phase, using a redundant and dissimilar architecture and correlating security alerts to detected safety failures. As far as aeronautics will be threatened by cyber-attacks, having a monitoring system to detect them will obviously not avoid them, but at first, it can be useful to security specialists to automatize their forensic analysis. If security countermeasures are embedded on an aircraft, the monitoring system will help evaluating their performances, or even help noticing if countermeasures are over-designed given the threat environment. The goal is to have a security monitoring system accurate enough to lay the foundations of a future embedded proactive security system.

References

[1] R. D. Cerchio, C. Riley, Oct. 2011, "Aircraft systems cyber security," the 30th Digital Avionics Systems Conference, Seattle, WA.

[2] K.Willsher, Feb 2009, "French fighter planes grounded by computer virus," The Telegraph.

[3] P. Passeri, Oct. 2011, "Oops, my drone was infected!," http://hackmageddon.com/2011/10/08/oops-my-drone-was-infected/.

[4] N. Shachtman, Jul. 2011, "Exclusive: Computer virus hits U.S. drone fleet," http://www.wired.com/dqngerroom/2011/10/virus-hits-drone-fleet/.

[5] Bureau d'Enquêtes et d'Analyses (BEA), Jan. 2007, "Rapport sur l'incident survenu le 10 décembre 2006 sur l'aérodrome de ParisOrly au Boeing 747-400 immatriculé F-HLOV exploité par Corsair," http://www.bea-fr.org/docspa/2006/f-ov061210/pdf/f-ov061210.pdf (available in french only).

[6] C. Arthur, May 2012, "Cyber-attack concerns raised over Boeing 787 chip's 'back door'," http://www.guardian.co.uk/technology/2012/may/29/cyber-attack-concerns-boeing-chip.

[7] EUROCAE WG-72 and RTCA SC-216, 2010, “ED-202: Airworthiness Security Process Specification”.

[8] RTCA SC-216 and EUROCAE WG-72, 2011, "ED-203: Airworthiness Security Methods and Considerations" (Working draft version).

[9] The IFALPA (International Federation of Air Line Pilot’s Associations) Security Committee, Jun. 2013, "Cyber treats: who controls your aicraft?", http://www.ifalpa.org/store/14POS03 - Cyber threats.pdf.

[10] S. Gil Casals, P. Owezarski and G. Descargues, Sep. 2012, "Risk assessment for airworthiness security", in 31st International Conference on Computer Safety, Reliability, and Security SAFECOMP 2012, Magdeburg, Germany.

[11] K. Sampigethaya, R. Poovendran and L. Bushnell, Dec. 2008, "Secure Operation, control, and Maintenance of Future E-Enabled Airplanes", IEEE - PIEEE, vol. vol.96, no. no.12, pp. pp.1992-2007.

[12] F. Schreckenbach, Mar. 2009, "NEWSKY Project - Mobile communication network based on IPv6 to integrate satellite and air-ground links", ACGFG/5 and NexSAT/10 Meeting, Brussels.

[13] Aeronautical Radio Inc. (ARINC), Dec. 2005, "ARINC report 811: Commercial aircraft information security concepts of operation and process framework".

[14] M.S Ali, R. Bhagavathula and R. Pendse, Oct. 2004, "Airplane data networks and security issues", 23rd Digital Avionics Systems Conference, Salt Lake City, UT.

[15] F. Famili, Nov. 1999, "Monitoring of aircraft operation using statistics and machine learning", 11th IEEE International Conference on Tools with Artificial Intelligence, Chicago, IL.

[16] G. Li, Jul. 2010, "Machine learning in fuel consumption prediction of aircraft", in 9th IEEE International Conference on Cognitive Informatics (ICCI), Beijing.

[17] S.H. Rubin and G. Lee, 2011, "Human-machine learning for intelligent aircraft systems," 2nd International Conference on Autonomous and Intelligent Systems.

[18] M. Ester, K. H.-P., J. Sander and X. Xu, 1996, "A density-based algorithm for discovering clusters in large spatial databases with noise", 2nd International Conference on Knowledge and Data Mining, KDD'96.

[19] J. Mazel, P. Casas and P. Owezarski, 2011, "Sub-space clustering and evidence accumulation for unsupervised network anomaly detection", 3rd international conference on traffic monitoring and analysis TMA'11.

[20] K. Fukunaga and L. D. Hostetler, Jan. 1975, "The estimation of the gradient of a density function, with applications in pattern recognition," in IEEE Transactions on Information Theory.

[21] D.M.J. Tax and R.P.W. Duin, Jan. 2001, "Support vector data description", Machine Learning Journal, vol. 54, pp.45-66, Kluwer Academic Publishers-Plenum Publishers.

[22] D. D. E., Feb. 1987, "An intrusion-detection model", IEEE Transactions on Software Engineering.

[23] K.N. Rao, Dec. 1989, "Security audit for embedded avionic systems", 5th Annual Computer Security Applications Conference, Tucson, AZ.

32nd Digital Avionics Systems Conference
October 6-10, 2013