Detecting Ghost Aircraft Flooding in the Surveillance of Low-Flying Civil and Military Aircraft

1st Pirolley Melvyn*
Université de Franche-Comté
FEMTO-ST Institute, CNRS
Belfort, France
melvyn.pirolley@univ-fcomte.fr
*Corresponding author

2nd Raphaël Couturier
Université de Franche-Comté
FEMTO-ST Institute, CNRS
Belfort, France
raphael.couturier@univ-fcomte.fr

3rd Fabrice Ambert
Université de Franche-Comté
FEMTO-ST Institute, CNRS
Belfort, France
fabrice.ambert@univ-fcomte.fr

4rd Michel Salomon
Université de Franche-Comté
FEMTO-ST Institute, CNRS
Belfort, France
michel.salomon@univ-fcomte.fr

***Abstract*—In the past few years, the increasing number of aircraft in low-altitude traffic has brought new challenges for air traffic controllers. As the ADS-B protocol takes an important part in the air surveillance task, it is important to be prepared to face attacks related to it. Moreover, as the ADS-B is open and uses broadcasts, it is easy for an attacker to emit fake ADS-B data to flood the air traffic controller's screens. In this work, we will present two algorithms, the first one is based on hashing techniques, invariant to geometric transformations to detect trajectory replays. The second one uses a deep-learning model to detect irrelevant ghost trajectories. Our proposed approach can automatically filter a majority of ghost aircraft without risking to filter real aircraft.**

***Index Terms*—Flooding attacks, ADS-B, Low-altitude air traffic, Machine learning, Hash functions**

I. Introduction

In the past few years, the fast increase in air traffic load has brought new challenges for air traffic controllers. Moreover, many air traffic monitoring tools depend on the Automatic Dependent Surveillance-Broadcast (ADS-B) protocol. Indeed, thanks to onboard transponders, aircraft broadcast important information such as their identification (or call sign), position, altitude, and velocity in the form of ADS-B messages. Unfortunately, these messages are all vulnerable to false data injection attacks because the ADS-B protocol is open and lacks encryption and authentication. One of the most classical attacks consists of injecting ghost aircraft into the network. This attack could be used to flood air traffic controller's screens with a lot of ghost aircraft. Such an attack would make it impossible to know which aircraft are real and which ones are not, hence air traffic surveillance would become impossible. One solution do deal with these kinds of attacks consists in using machine learning techniques and hash tables to identify ghost aircraft.

Our study focuses on low-altitude traffic because the upcoming arrival of electric vertical take-off and landing (eVTOL) aircraft, drones, and others will strongly increase the low-altitude traffic load. The low-altitude traffic will be even more exposed to attacks, as it concerns a large diversity of aircraft that can sometimes fly in sensitive areas such as urban zones. Our project is a continuation of previous works carried out as part of the GeLeaD research project (see Acknowledgements on the bottom left part) [1, 2, 3]. Those three projects worked on ADS-B anomaly detection but their main limitation is that they focused on commercial traffic, whereas this study focuses on low-altitude traffic.

The main contributions of this study are as follows:

- Recognising ghost trajectories that replay ADS-B messages by using a hash function for trajectories, specifically designed to be invariant to geometric transformations.
- A residual LSTM model trained to distinguish ghosts in divergent saturation where ghosts spread around a real aircraft.

The remainder of the paper is structured as follows. Section II presents an overview of related studies. Section III explains the format of our dataset. Section IV presents the

This work was supported by the DGA (French defence procurement agency) in the context of the DApIA project (project number ANR-22-ASM2-0001) related to the ANR ASTRID Maturation program (specific support for follow-up research works of projects that have received a grant from the French Ministry of Armed Forces - GeLeaD project number ANR-18-ASTR-0011). It was also partially supported by the EIPHI Graduate School (contract ANR-17-EURE-0002). Computations have been performed on the supercomputer facilities of the "Mésocentre de Franche-Comté".

attacks we wanted to counteract and the techniques used to solve them. Section V presents some implementation choices that were made and shows the results of the study. Finally Section VI will conclude this article and open on our potential future works.

II. Related works

The problems involved in our study are similar to more general problems sych as how to evaluate distances between two time-series or how to predict future locations based on their trajectories.

In [4] the authors have developed an approach to evaluate precisely the distance between two time-series as handwritten digits. What makes this publication interesting for our work is that the metric has to be invariant to transformations such as translation, scaling, and rotation. Moreover, it uses a very similar method to ours to achieve invariance. However, as the objectives of this study are different to ours, the authors do not use hashing but dynamic time warping (DTW) to categorize similar time-series. This allows for a greater accuracy than the present approach but this option is impossible, as a fast pair-wise comparison with millions of entries is not feasible.

This study also deals with the problem of how to predict the future position of aircraft based on their trajectory. This is an important problem which has multiple applications. In [5], the authors have developed a method for continuing to predict aircraft positions in case of transponder failures, lack of ADS-B coverage, or local signal interference. This method is similar to ours but applies to commercial aviation. However, it could also be used in our domain to make a high-altitude flooding detection system.

III. Dataset

Some historical ADS-B records from the OpenSky database were collected to benefit from a large diversity of aircraft, flying in the same area. The dataset filters the flights under 10,000 ft, in the area of Toulouse and its surroundings. It focuses on recent ADS-B data, from 2022, and each message separated in trajectories is saved in a ".csv" file including classical ADS-B features such as Latitude, Longitude, Altitude, Velocity, Track... The dataset is accessible here: https://mega.nz/folder/R1MHER6a#uRYHrlwbAb14JCqoHDlfwg

IV. Method

As it is possible to use ghost aircraft attacks in many different ways, two main categories of attacks were distinguished.

Before presenting this method, it is important to mention that in both attacks, we consider that the saturation happens on the latitudinal and longitudinal axis. Such attacks are less dangerous and easier to counteract even if, it is also possible to make a saturation on the altitude axis. Modifying slightly this algorithm would allow us to detect ghosts based on altitude. However, this method takes into account the fact that the attacker could also modify the velocity or the track of the aircraft to maintain a certain coherence.

A. Ghost aircraft flooding

The first variant is the ghost aircraft flooding attack, which consists in generating multiple ghost aircraft surrounding a real aircraft. Ghost aircraft will have the same icao24 and callsign as the attacked one, making them impossible to distinguish. Figure 1 shows two ghost aircraft flooding attacks.

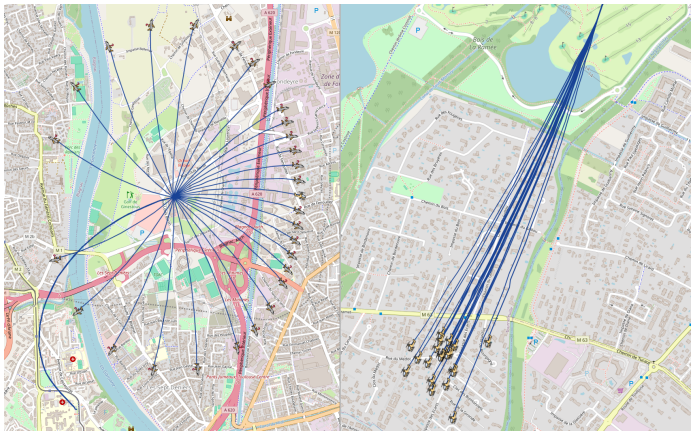


Fig. 1: Two examples of ghost aircraft flooding attacks

To detect which aircraft are ghosts, an artificial neural network model was trained to predict the next geographic coordinates of the aircraft. For each flight, when receiving new messages, the artificial neural network predicts the next latitude and longitude of the aircraft based on a window of the last 30 ADS-B messages. At the same time, the algorithm computes the distance (in meters) between the previous coordinates predicted by the model and the ones in the new ADS-B message. This distance metric gives the error of the model. When this distance increases too much, one can admit that the trajectory is abnormal.

In practice, when a ghost aircraft appears next to a real one, the ghost aircraft takes a slight turn to separate from the original aircraft. This causes the distance metrics to spike up because the model is "surprised" by this unexpected turn. Theoretically, among ghost aircraft, the one with the smallest error is the right one because it has the less aberrant trajectory.

B. Ghost aircraft injection

The second attack, which is called "ghost aircraft injection" consists of the regular re-transmission of ADS-B messages from a recorded flight of the past. For example, an attacker could replay a take-off in an airport, every minute of it, with the icao24 code of an aircraft about to take off (Figure 2). Such kind of replay would make the air traffic controller of the airport unable to know if there is a real plane taking off, or if it is just a ghost.

To counteract this type of attack, the goal is to identify, as fast as possible, if a trajectory has already taken place. To this aim, a hashing algorithm, specifically designed for trajectories, has been developed. It regularly checks if a window of the last 32 seconds (32 messages) matches any other one in a database. Of course, pair-wise comparisons of trajectories are

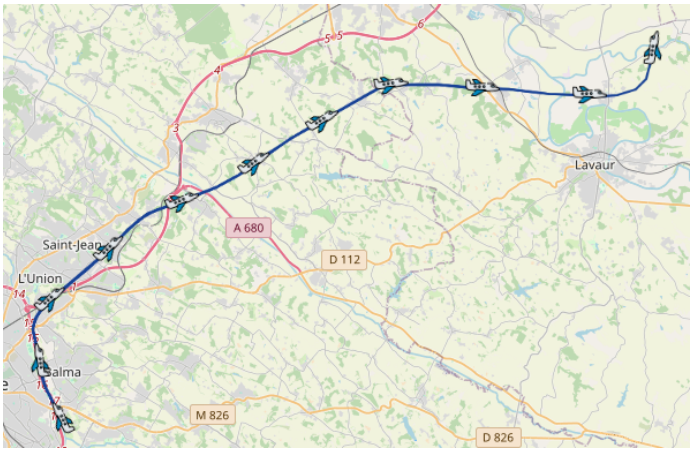


Fig. 2: Example of ghost aircraft injection attacks,

impossible because it would take too long, instead, a hash table is generated from the flights of our database. Then, all that is left to do is to hash the 32 time-steps window and check if there is another match. This task is extremely fast and solved in O(1) even with a very big hash table. Generating the hash table can, on the other hand, be slower, depending on the size of the database, but it is not a problem as it has been done once, before using our system.

Then, to have the most efficient replay detector, our hash method has to be precisely adjusted. It has to tolerate small modifications of the latitude and longitude values otherwise replays will generate new hashes and will not match. On the other hand, this hash method should not be too permissive or every flight will be detected as a replay. Moreover, attackers could use a flight from another location and translate the replay wherever they want. So a perfect hash function has to be invariant to the maximum possible transformations such as translation, rotation, scaling, flips, and others. To deal with these constraints, our algorithm converts trajectories into a series of right, left, or flat turns and then hashes them.

The methodology adopted in this study is the following. The raw trajectory is defined as T, a series of n pairs of latitude and longitude (lat, lon). First, the latitude and longitude are normalized to 0°N 0°E with rotations and projections to reduce distortions due to the earth's curvature. The newly generated coordinates are now "flat" and the new series N (Normalized) can be represented with rectangular coordinates x and y (x related to the longitude and y to the latitude)

$$\begin{aligned} N_i = &\begin{pmatrix} cos(T_i[lon]) \times cos(T_i[lat]) \\ sin(T_i[lon]) \times cos(T_i[lat]) \\ sin(T_i[lat]) \end{pmatrix} \times \\ &Qrot_z(-Tlon_{i-1}) \times Qrot_y(Tlat_{i-1}) \\ &N_i = N_i[yz] \text{ //Only keep y and z coordinates} \\ &Qrot_a = \text{3D rotation matrix on axis } a \end{aligned} \tag{1}$$

Then, the series D of "direction" and the series V of "speed" are calculated by computing the angular direction between each message representing at each timestamp the direction of the aircraft.

$$\begin{aligned} D_i &= arctan2(N_i[y], N_i[x]) \\ V_i &= \sqrt{N_i[y]^2 + N_i[x]^2} \end{aligned} \tag{2}$$

Next, the series S is calculated, the serialized representation of our original series T.

$$S_i = \begin{cases} 0 & \text{if } V_i < 2e-6 \text{ or } V_i > 1e-4 \\ -1 & \text{if } D_(i-1) \text{ to } D_i \text{ is a left turn} \\ 1 & \text{if } D_(i-1) \text{ to } D_i \text{ is a right turn} \end{cases} \tag{3}$$

Finally, the hash of the time series can be computed. Thanks to the serialization step, S is now invariant to translation, rotation, and mostly all other simple transformations. However, it is still affected by vertical or horizontal flips of the trajectory. Reversing the trajectory reverses all left and right turns. This last transformation is handled directly into the math of the hash function, representing left turns by negative values and right turns by positive values. Thus, if the hash of the original time server is positive, the hash of the inverted server will be negative, and vice versa. Consequently, the absolute value of the hash is invariant to reversal transformation

$$H_i = \mid \sum_{n=0}^{W} S_{i-n} * 3^i \mid \tag{4}$$

W is the window size (32 in our implementation).

V. Results

This section will present the choices that have been made to improve the capabilities of our method. We will explain some implementation details and try to reach the limits of our system. The code used to perform these experiments is available on our GitHub at https://github.com/DApIA-Project/Anomaly-Detection/tree/main.

A. Ghost aircraft flooding

For the resolution of ghost aircraft flooding attacks, it was determined that using a residual LSTM model was one of the best approaches. Our model (Figure 3) takes as input the classical ADS-B field and tries to output the future latitude and longitude. The model is very light, with only 265,880 parameters allowing it to be used in real-time. The training is achieved in 128 epochs of 32 batches of 64 samples.

Our model can predict the future position of the aircraft with a precision of **29.3m**. When a ghost aircraft spreads around the real one, one can easily notice that the turn made by ghosts surprises the model. The sharper the turn is, the higher the model errors are. In Figure 4, different examples of ghost aircraft can be seen. The first one takes a sharp turn of 30 degrees, the second one a slight turn of 10 degrees, and the third figure is the trajectory of the real aircraft. The green line is the path of the aircraft and the blue cross is the location where the model thought the aircraft would be.

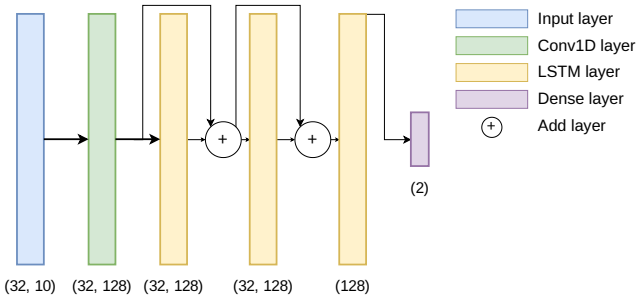
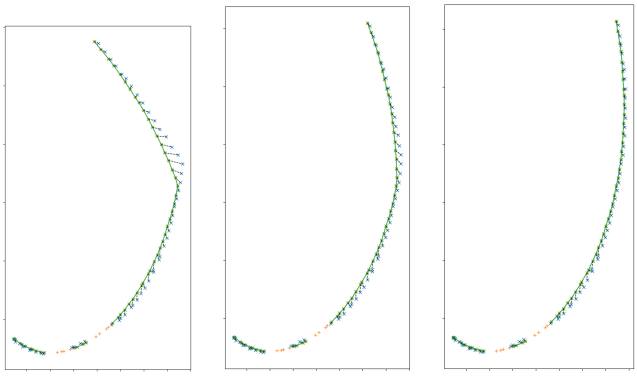


Fig. 3: Res-LSTM model architecture used to predict the coordinates of the aircraft

The prediction and the target position were connected with black dashed lines to highlight the error of the model at each timestamp. The longer the black line is, the more unexpected the turn, indicating the likelihood that the aircraft is a ghost. Finally, the orange cross is the location where the model was unable to make any predictions because of the poor quality of the trajectory.



(a) Divergence=30° (b) Divergence=10° (c) Real aircraft

Fig. 4: Comparison between model predictions and truth values

To analyze the model's predictions, we can then plot the evolution of the error as a function of time, and see where the error reaches its maximum. For example, in Figure 5, the error spikes up near time-step 60 for the first and the second flight as they are ghosts but it does not for the third one as it is a real aircraft.

For the prediction to be relevant, a very important parameter is the distance in time of the model's prediction. A model predicting coordinates with too short a time horizon (e.g. T+1) will have a very low loss. However, it will adapt too fast to unexpected turns and will not detect an anomaly (Figure 6a). On the other hand, a model that predicts too far ahead will not be accurate enough under normal conditions to tell whether a message is abnormal or not. In Figure 6b one can see that the model error is too high before the attack for a difference to be noticeable at the moment of attack. That is why it is so important to adapt this parameter to achieve a

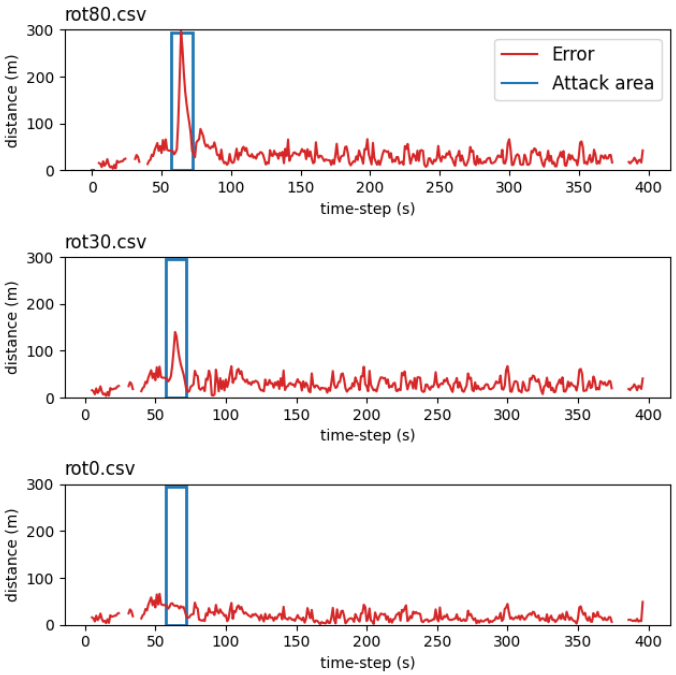
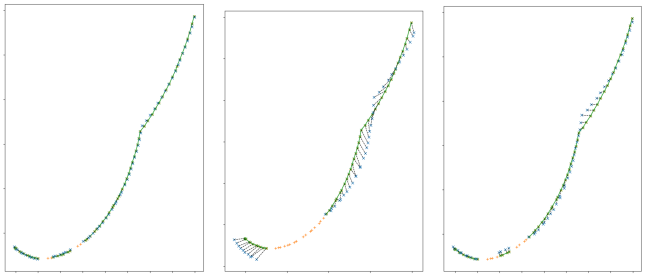


Fig. 5: Comparison of the error profile for the prediction on two ghost aircraft first, and then the real aircraft.

compromise between accurate predictions and late predictions to increase the unexpectedness of an attack. Figure 6c shows the best balance. By testing different parameters, it has been determined that predicting the aircraft's position in 4 seconds gives the best contrast.



(a) Horizon=T+1 (b) Horizon=T+7 (c) Horizon=T+4

Fig. 6: Comparison of the effect of setting prediction horizon too low, too high, or correctly

Finally, we tried to reach the limits of our model by testing very small divergences, with angles of less than 10 degrees, and under difficult conditions: the real aircraft also took a turn while the attack was taking place. This experiment shows that if a ghost aircraft takes a slight turn, diverging by less than 7 degrees from the real aircraft, our method starts to become uncertain in distinguishing the real plane. This result is encouraging, because even if, under difficult conditions, the algorithm is unable to distinguish the real aircraft, it is still able to accurately filter out a large number of ghost aircraft

deviating by more than 7 degrees.

B. Ghost aircraft injection

To evaluate the detection capacities of replays with our algorithm, we filled the hash table with more than 10,000 flights. Hashing all the trajectories generates, 10,801,293 hashes, and 260,716 hashes are associated with at least two flights. We will call them collisions, but they are not truly real hash collisions, it only happens because two flights can have the same series of 32 right and left turns. Moreover, the effect of those collisions on the final prediction is so low that their impact can be ignored.

Next, we ran our algorithm through one hundred flights in the database to check whether they would be detected as replays. Each of these hundred flights has been randomly translated, rotated, scaled ... Moreover, we also submitted to our algorithm, a hundred flights that are not in the database to check if the algorithm would not generate false positive alerts.

As a result, **96%** of the 200 flights were correctly classified as replay or not. The algorithm has never generated any false positives which means that the algorithm is very reliable and will never label a real aircraft as a ghost. On the other hand, the algorithm failed to detect 8 flights as replays because in some situations the trajectory shapes combined with transformations are too straight to correctly identify micro-turns as right or left. If we repeat this experiment without applying any transformations to the trajectories tested, the system's accuracy is perfect, with 100

VI. Conclusion

To conclude, this paper demonstrates that it is possible to automatically detect ghost aircraft by using machine learning techniques or specific hash functions. Even if this study shows that under some conditions it is not always possible to tell each aircraft if they are real or a ghost, we show that we can precisely filter almost every ghost. The results are encouraging because they prove that there are solutions to face flooding attacks and even if we can not always filter every ghost, it will in any case greatly simplify the air traffic control task in case the system is targeted by a flooding attack.

In our future work, we will try to improve our replay detector by improving the robustness of our hash function to manage the problem of micro right and left turns better. Moreover, there are still a lot of experiments that need to be conducted to improve our artificial neural network for solving divergent flooding attacks. One idea would be to combine multiple predictions with different time horizons to improve the accuracy of the method. Other models such as a recent transformer model adapted for regression tasks could also be used [6].

References

[1] Ralph Karam, Michel Salomon, and Raphaël Couturier. "A comparative study of deep learning architectures for detection of anomalous ADS-B messages". In: *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*. Vol. 1. IEEE. 2020, pp. 241–246.

[2] Ralph Karam, Michel Salomon, and Raphaël Couturier. "Supervised ADS-B Anomaly Detection Using a False Data Generator". In: *2022 2nd International Conference on Computer, Control and Robotics (ICCCR)*. 2022, pp. 218–223. DOI: 10.1109/ICCCR54399.2022.9790149.

[3] Antoine Chevrot, Alexandre Vernotte, and Bruno Legeard. "CAE: Contextual auto-encoder for multivariate time-series anomaly detection in air transportation". In: *Computers & Security* 116 (2022), p. 102652.

[4] Michail Vlachos, D. Gunopulos, and Gautam Das. "Rotation invariant distance measures for trajectories". In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. Seattle, WA, USA: Association for Computing Machinery, 2004, pp. 707–712. ISBN: 1581138881. DOI: 10.1145/1014052.1014144.

[5] Gong Y. et al. Yang Z. Kang X. "Aircraft trajectory prediction and aviation safety in ADS-B failure conditions based on neural network". In: 2023. DOI: 10.1038/s41598-023-46914-2.

[6] Haixu Wu et al. *Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting*. 2022. arXiv: 2106.13008 [cs.LG].