2018 IEEE 16th Int. Conf. on Dependable, Autonomic & Secure Comp., 16th Int. Conf. on Pervasive Intelligence & Comp., 4th Int. Conf. on Big Data Intelligence & Comp., and 3rd Cyber Sci. & Tech. Cong.

Countering Real-Time Stream Poisoning: An architecture for detecting vessel spoofing in streams of AIS data

Ioannis Kontopoulos[ǂ*+], *Giannis Spiliopoulos*[*], *Dimitrios Zissis*[*†], *Konstantinos Chatzikokolakis*[*] *and Alexander Artikis*[ǂ]

[ǂ] *Institute of Informatics & Telecommunications,*
National Centre for Scientific Research (NCSR) Demokritos, Athens, Greece
Email: {ikon, a.artikis} @iit.demokritos.gr
** MarineTraffic, London, United Kingdom*
Email:{giannis.spiliopoulos, konstantinos.chatzikokolakis} @marinetraffic.com
† Department of Product and Systems Design Engineering, University of the Aegean, Syros, Greece
Email: dzissis@aegean.gr
+Department of Informatics and Telematics, Harokopio University, Athens, Greece

***Abstract*— "Well poisoning" is an ancient war stratagem which was frequently used as a "scorched earth tactic". Today this tactic has been adapted by malicious attackers to the digital world and evolved into "stream poisoning", in which corrupt or fallacious data is injected into a data lake, so as to corrupt the integrity of the information stored there. Numerous maritime surveillance systems nowadays rely on the Automatic Identification System (AIS), which is compulsory for vessels over 299 Gross Tones, for vessel tracking purposes. Ship AIS spoofing involves creating a nonexistent vessel or masquerading a vessel's true identity, resulting in hiding or transmitting false positional data, so that a vessel appears to behave legitimately, thus deceiving stakeholders and authorities. Due to the volume and velocity of data received traditional approaches fail to automatically detect these spoofing events in real time. We focus on an industrial use case of detecting spoofing events in AIS streams and validate our approach in real world conditions.**

Keywords*— distributed stream processing; big data; AIS vessel monitoring; anomaly detection

I. INTRODUCTION

Unlike in the past, when maritime surveillance had suffered from a lack of data, systems are now producing unprecedented amounts of data. These systems can be classified into two broad categories: i) cooperative self-reporting systems where vessels broadcast information regarding their identity and location [these include Automatic Identification System (AIS), Long Range Identification and Tracking (LRIT) and Vessel Monitoring System (VMS)] and ii) non-cooperative surveillance systems which detect, track and/or identify vessels without any cooperation from the vessels itself e.g., radar [1]. The most commonly used is AIS, a collaborative, self-reporting system which allows vessels to periodically broadcast their identification information, characteristics, navigation, and locational data, which is then received by other vessels and coastal or satellite receiving stations. In 2002, the IMO SOLAS agreement made it compulsory for all vessels over 299 Gross Tones to carry an AIS transponder on board for safety reasons. Soon AIS was used for global scale vessel tracking, with many much smaller vessels optionally reporting their positions.

Today AIS is considered a reliable and trustworthy source of information by numerous maritime stakeholders, including port authorities and coast guard agencies across the world. Using this data, several websites provide information free of cost to the general public (e.g. marinetraffic.com). Such AIS based systems provide an almost global depiction of maritime transportation (often referred to as white shipping) in real time. These systems are constantly flooded by endless streaming data generated from millions of distributed onboard sensors at rates of numerous gigabytes per day.

Unfortunately, as the AIS was not designed with security as a primary requirement, it is subject to malicious attacks, which attempt to falsify positional or identification data. These attacks aim at the integrity and reliability of the AIS data, so that it is modified, fabricated, or interrupted (loss). As a result of these attacks, stakeholders have often been deceived to believe that vessels may be conducting illegal activity when they are not, or illegal activity has been made to appear legitimate or been concealed altogether. This evolution of the ancient stratagem "poisoning the well" involves corrupting stored information by maliciously injecting false data.

On the occasion of a serious incident, popular Big Data approaches are used to analyze petabytes of historical data and shed light on what took place at a specific time and location [2][3]. While for many other domains retrospective batch processing may be sufficient, for a wide range of real world applications, this is simply too late. For applications such as navigation, surveillance etc., timeliness is a top priority; making the right decision regarding steering a vessel away from danger, or deploying a search and rescue mission, is only useful if it is a decision made in due time. For these fast changing, massive, potentially infinite sequences of real-time data, batch processing for proactive and real-time decision making is not really an option. Unfortunately, current state of the art techniques and technologies are incapable of dealing with these growing volumes of high-speed, loosely structured,

978-1-5386-7518-2/18/$31.00 ©2018 IEEE
DOI 10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00139

spatiotemporal data streams, that require real-time analysis in order to achieve rapid response times. For this reason, data streams research and their correlation is gaining attention as a subset of the more generic "Big Data" research field.

In this work we focus on detecting data spoofing and falsification attacks in real-time. We propose a distributed architecture capable of handling firehoses of data and countering "Stream Poisoning" in real time. We focus on the industrial use case of detecting spoofing events in stream of AIS and validate our approach under real world conditions. The rest of the paper is organized as follows: Section 2 shortly presents previous work in this domain, while Section 3 describes our approach. Section 4 presents the preliminary results, while section 5 concludes this paper by briefly outlining the main contributions of this work and suggesting future improvements.

II. Related Work

A spoofing attack is a situation in which an attacker falsely impersonates a valid entity (masquerading), tricking the system into trusting the data provided. Ship AIS spoofing often involves creating a nonexistent vessel or masquerading vessel's identity with a false one, hiding or transmitting false positional data, so that a vessel appears or behaves legitimately (Fig). The rise of applications relying on AIS data for vessel tracking has increased the interest of researchers in detecting AIS vessel spoofing. In this context, several studies have focused on the problem [4]-[12].

In [4], Balduzzi presents a method of spoofing a vessel by injecting invalid data into AIS gateways triggering fake collision warning alerts and potentially making surrounding vessels alter their course. Similarly in [4], a method of AIS hijacking is described, by which attackers can maliciously modify information provided by the AIS base station, eavesdropping on all transmissions (i.e. man-in-the-middle) or modifying data. This method makes it possible for attackers to alter any information broadcast by existing AIS stations regarding real vessel data (e.g., cargo, speed, location, and country). Iphar, Napoli, and Ray propose an architecture capable of identifying spoofing messages in AIS after it has entered the database, by using a methodology based on integrity and quality assessment of the AIS messages [5]. In Papi et al. [6], authors combine a classic radio-localization method based on time difference of arrival with an extended Kalman filter designed to track vessels in geodetic coordinates. Katsilieris, Braca, and Coraluppi focus on the problem of

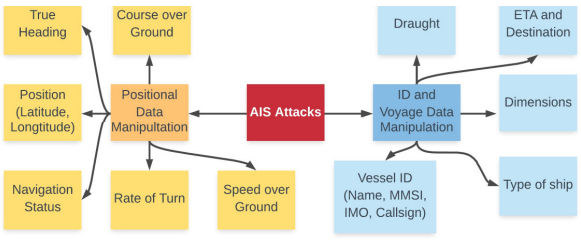

Fig. 1. AIS Attack tree by message type and parameter

whether the received AIS data are trustworthy with the help of radar measurements and information from the tracking system [7]. Guerriero, Coraluppi, Carthel, and Willett (2010) attempt to identify the intentional on-off switching, through the introduction of the notion of channel memory and the application of Hidden Markov Models (HMMs) [8][9]. Similarly focusing also on the intentional on-off switching of AIS transponders, Mazzarella et. al. [10] explore a solution exploiting the Received Signal Strength Indicator available at the AIS Base Stations (BS). In designing such an anomaly detector, the electromagnetic propagation conditions that characterize the channel between ship AIS transponders and BS have to be taken into consideration.

Fewer works focus on real time spoofing detection. In Zissis [11], Support Vector Machines were employed on the edge of the cloud to perform low footprint unsupervised learning and analysis of sensor data, so as to detect spoofing attempts in streams of AIS data. In Salmon [12], authors propose an architecture which combines archived data analysis with real-time streams to process mobility data as they arrive, tackling the limitations introduced by either of the approaches. Moreover, they compare the results to purely offline or online approaches in terms of efficiency. Our approach is differentiated in a sense that it is purely online and has great scaling capabilities, relying only on AIS and the timestamp given by the BS in order to identify spoofs/duplicate MMSI transmissions on a global scale.

III. APPROACH

The main objective of our work is to explore the possibility of detecting spoofing events and trajectories with a distributed method on high velocity streams of data. In short, our approach needs to:

- be able to handle large volumes of data with high velocity, or bursty event rates by distributing the load across several machines, and
- maintain high accuracy rate and detection performance in real world conditions, such as detecting spoofing events from an incoming stream of AIS messages received by a transceiver.

In the following section we describe our approach, which is designed for maritime data and architecture set up, before evaluating our results.

A. *SPOOFING DETECTION ALGORITHM*

To tackle the problem described in Section II we employ an approach that calculates the average speed required to move, following the shortest path, between two consecutive positions reported through AIS. This is achieved by calculating first the Haversine distance for each pair of positions and then and the time needed to cover that distance based on the timestamps (ingestion time) recorded by the AIS receivers. If the calculated speed is within a feasible range (i.e., the vessel's average speed cannot exceed 50 knots), then the succeeding message becomes the new last valid position for that vessel. Otherwise, the message is flagged as possible spoofing. This is independent of the vessel type, or the area in which the vessel is travelling.

TABLE I below shows the algorithm that decides and flags accordingly each message in real time, either as spoofing or valid, based on the transition feasibility criterion shown in TABLE II. We keep track of last valid position per MMSI in a hash map structure, where the MMSI is the hash key. TABLE II shows the implementation of the transition feasibility criterion, which calculates the average speed needed for a vessel to move from the position recorded in M_1 to the position recorded in M_2.

Even though we can detect possible trajectory outliers, we have no prior knowledge of what a valid trajectory is. This is often referred to as an algorithmic cold start problem, where initially, the system has no messages and no trajectories recorded. As the messages are consumed, the system waits until it detects two consecutive valid transitions (i.e., the system buffers up to two positions) before assigning the first valid last known position which will be used to detect possible spoofing events. All positions that fail to meet the transition feasibility criterion (i.e., the calculated average speed threshold), before the first last known position for a specific vessel is assigned, are marked as spoofs.

B. ARCHITECTURE FOR DATA PROCESSING

The system needs to be capable of handling large volumes of data with high velocity or bursty event rates. To tackle these demands, we use an asynchronous, message passing framework, called the Akka framework [13], which makes the distribution of workload possible.

TABLE I. SPOOFING DETECTION ALGORITHM

Step No.	SpoofingDetection (lastValidPositions, buffer, M[a])
1	F = M
2	**IF** buffer.size < 3 **THEN**
3	**IF** isSpoofing(buffer.lastElement, M) **THEN**
4	buffer.append(M)
5	**FOR** N in buffer
6	N.spoof = TRUE
7	**END FOR**
8	F = buffer
9	buffer = List()
10	**RETURN F**[b]
11	**ELSE**
12	buffer += M
13	**RETURN NULL**
14	**ENDIF**
15	**ELSE IF** M.mmsi in lastValidPositions **THEN**
16	lastValidPosition = lastValidPositions(M.mmsi)
17	**IF** isSpoofing(lastValidPosition,M) **THEN**
18	F.spoof = TRUE
19	**ELSE**
20	lastValidPositions += (M.mmsi → M)
21	**ENDIF**
22	**ELSE**
23	lastValidPositions += (M.mmsi → M)
24	**ENDIF**
25	**RETURN List(F**[b]**)**

a *Single AIS message*

b *Flagged List of AIS messages*

TABLE II. TRANSITION FEASIBILITY CRITERION

Step No.	isSpoofing (M_1,M_2)
1	distance = haversineDistance(M_1,M_2)
2	duration = M_2.timestamp – M_1.timestamp
3	speed = distance / duration
3	**IF** speed > 50.0 **THEN**
4	**RETURN** true
10	**ELSE**
11	**RETURN** false
12	**ENDIF**

The system architecture has been implemented with Akka [13], a framework for highly concurrent and distributed systems based on the Actor model. This framework is preferred over other popular distributed systems. While Apache Flink [14] is such an alternative option, with built-in protocols for load balancing and sanity checks, that has been used for real-time trajectory and mobility analysis [15], Akka offers a low-level, message-exchange interface (a less restrictive programming model) making it ideal for flexible and customizable approaches. Our proposed distributed architecture consists of three types of nodes: one Master node, n Worker nodes and a Consensus node. Every AIS message is received from the Master node and is distributed to the worker nodes using a load balancer solution.

Master node. The Master is responsible for receiving timestamped AIS messages in the form of <timestamp, AIS message> tuples (as recorded by AIS stations), decoding them, and passing them to the Worker nodes. The default format of the AIS messages is the NMEA message format. An example of the AIS message format can be seen below:

!AIVDM,1,1,,A,14eG;o@034o8sd<L9i:a;WF>062D,0*7D

In the NMEA encoding each character corresponds to 6 binary bits, unlike ASCII, which uses 8 bits. This encoded string is converted to a sequence of ones and zeros. The sequence can then be divided into sets of strings, each set corresponding to a different piece of information, such as vessel MMSI, recorded speed or heading. At first, the master node receives a stream of raw messages with <timestamp, AIS message> tuples from a source (e.g., AIS transceiver) in the NMEA standard format and decodes them. Then, it must select a worker node to forward the message to. This is accomplished through a routing table maintained in the Master node that contains state information about the MMSIs associated with the Worker nodes and a Hash Map – hereafter 'Balancer' – that contains processing status information for each Worker (i.e., number of trajectories processed). In case the MMSI has already been routed to a Worker based on the routing table, the Master will immediately send the message to the corresponding Worker node. Otherwise, the Balancer lets the Master know which Worker has processed the least number of trajectories, defined by the number of MMSIs processed (each MMSI corresponds to a trajectory). The Master sends the message to that node and updates the routing table. Finally, the Master node periodically sends messages to the Consensus node, the

role of which will be explained in the following paragraphs. These messages contain information about how many messages have been decoded and routed by the Master.

Worker node. Each Worker is independent and responsible for a distinct set of trajectories. To detect possible spoofing, it employs the detection algorithm described in Section III.A. The output of each worker is a stream of decoded AIS messages containing a flag indicating whether a message has been spoofed or not. Finally, Worker nodes, similarly to the Master node, send periodically messages to the Consensus node which contain the number of messages processed by each worker.

Consensus node. This node is responsible for collecting information from the other nodes and measure the reliability and the performance of the system. In a distributed system it is often hard to evaluate its reliability; therefore, we have adopted a naïve approach with checkpoints to perform sanity checks periodically (i.e., once every *x* messages). At each checkpoint, the Consensus node receives information from the Master node and the Worker nodes about the total number of messages sent from the Master to the Workers and the number of messages processed cumulatively by the workers. If the two numbers are equal, we consider our system to be reliable. Furthermore, the Consensus node collects data about spoofing events and message consumption rates from the Worker nodes and evaluates our system in terms of performance, by comparing the number of messages processed between each pair of successive checkpoints and calculating the system's throughput.

Fig. 2 illustrates the overview of the system architecture with four worker nodes. The master receives raw AIS messages from the transceiver and sends decoded messages to the workers. Every *x* messages, both the workers and the master send the number of events processed to the Consensus node for sanity check and throughput calculation.

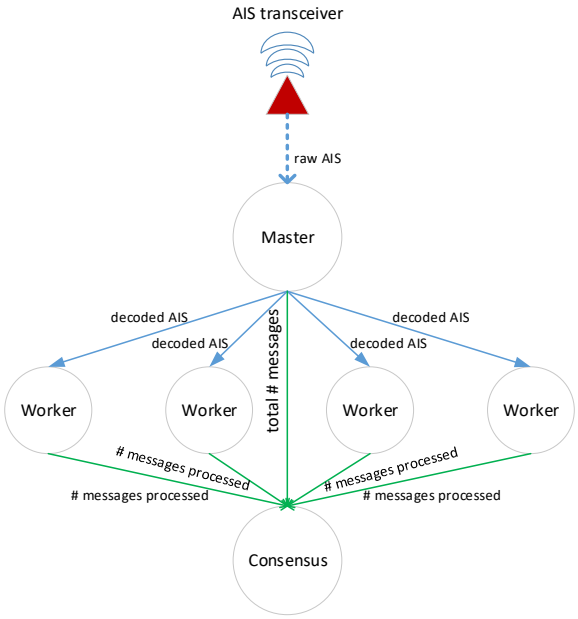

Fig. 2. System architecture overview

IV. EXPERIMENTS

A. DATASET DESCRIPTION

The dataset that was used was provided by MarineTraffic and contains 43,912,236 AIS messages received from 28,961 vessels sailing in the Mediterranean in May 2016. The total size of the dataset is 2.4 GB. At this point it should be noted that the Search and Rescue (SAR) aircrafts, which can have speeds greater than the max speed of a vessel, are considered outliers and thus have been removed from the dataset. To evaluate our approach, we artificially induced noise into the dataset in two different ways. First, we defined a random selection function that decides randomly in real time if a message will be replaced by noise or not. This was achieved by picking numbers out of a normal distribution and comparing them to a threshold that corresponds to the noise level that we target at each iteration (we demonstrate results for five different noise levels in the following section). In case that the corresponding message is selected to be replaced, we sample the pair of artificial coordinates (i.e. longitude and latitude) out of their corresponding normal distributions that are projected to cover the Mediterranean area. In addition, we allow a cold start for our spoofing detection algorithm by replacing messages only after our algorithm has detected the minimum number of consecutive valid messages for each MMSI (i.e., three messages corresponding to two valid transitions). Secondly, we aimed to synthetically create a worst-case scenario dataset, by introducing entire spoofing trajectories into valid ones. Such spoofing is far more complex to detect compared to sole signals. To generate such a dataset, we initially filtered from the dataset all the positional data of MMSIs with less than 30 messages transmitted during the whole interval (i.e., 1 month). Then, we transformed the filtered data into noise, by changing the MMSI of each entry with an MMSI coming from the rest of the data (i.e. MMSIs that have more than 30 transmissions during the considered time interval). Finally, we merged the noise data to the rest of the dataset. In both cases the noise ratio is approximately close to noise observed in real systems (e.g. marinetraffic.com).

Fig. 3 and Fig. 4 illustrate the artificially induced noise in the area of Sicily, Italy. Red dots denote AIS messages which correspond to spoofing events. Fig. 3 illustrates the noise induced using the first method, which most resembles the spoofing events occurring in a real-world setting, while Fig. 4 demonstrates the noise induced by changing the MMSIs, which is a worst-case scenario, not so commonly seen in real world cases. It is observed from those figures that the second scenario contains less spoofing events, which is normal considering the AIS transmission frequencies and the strict threshold of 30 messages for a whole month.

B. EXPERIMENTAL EVALUATION

The experimental evaluation presented below was performed on a machine with 2 cores (4 logical processors), an AMD A10-7870K Radeon R7 CPU, and 8 GB of RAM, running Windows 10 with Java OpenJDK 1.8, Scala 2.11.7 and Akka 2.5.1.

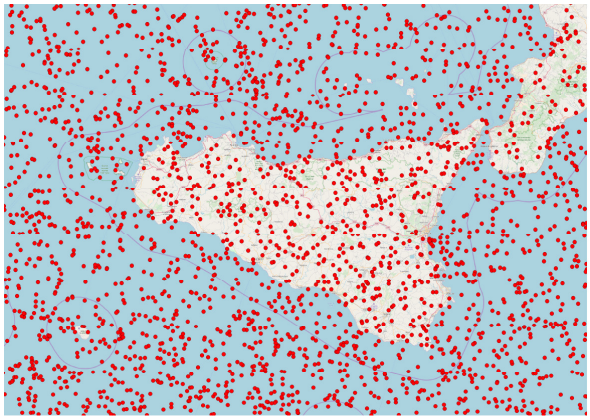

Fig. 3. Induced noise by randomly changing coordinates of AIS messages

When evaluating the first scenario, we performed five rounds of experiments, each one introducing a different level of noise to the dataset. To evaluate our methodology, we measured the number of spoofing messages the system was able to detect (true positives), the number of messages that were falsely detected as spoofing (false positives) and the number of spoofing messages that were falsely ignored and classified as valid (false negatives) for each dataset using two Worker nodes. Each round of experiments was executed multiple times (i.e., 5 times) and the average values of those measurements were calculated. TABLE III shows the results from these experiments. The first column highlights the average noise to total messages ratio of each round. The noise ratio ranges from 0.22% to 6.81%. The rest of the table shows the minimum, maximum and average values of f1-score the algorithm managed to achieve in each case.

Fig. 5 is a visual illustration of TABLE III. Moreover, it shows the precision and the recall of the algorithm. From Fig. 5 we can see that as the noise increases the f1-score of our solution decreases. In cases where there is increased noise in the stream, and since in the beginning the system had no prior knowledge about a valid trajectory, it was difficult to distinguish the real trajectories from the fake ones, and thus the false positive/negative rate increased. A visual illustration on

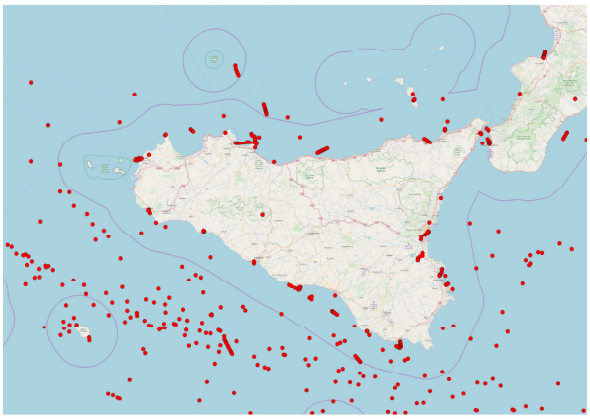

Fig. 4. Induced noise by changing MMSIs

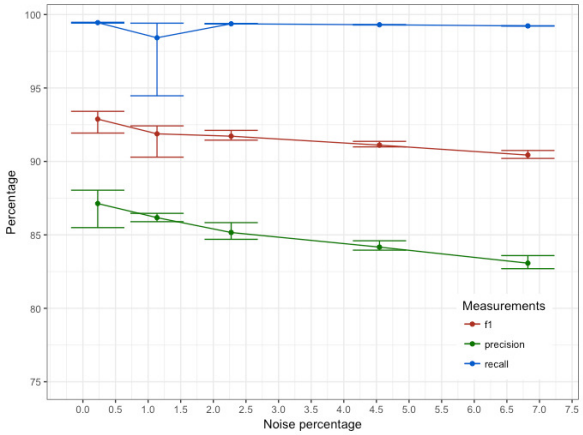

Fig. 5. Average performance, measuring precision, recall and f1-score for the spoofing algorithm

the map is given by Fig. 6. It was observed that several spoofing events were not detected (false negatives, blue dots), while others were falsely detected as spoofing events (false positives). False positives can easily be seen by comparing Fig. 3 and Fig. 6. Several trajectories (dense red dots that form trajectories around Sicily) can be seen in Fig. 6, while in Fig. 3 they are missing. This means that an entire trajectory was falsely identified as spoofing.

In extreme cases, such as when two vessels use the same MMSI, either for a long period of time, or share close geographic proximity, a fake trajectory can be falsely classified as valid, which leads to an increase of false positive/negative detections. This is mainly because the system has no knowledge about a valid trajectory in the beginning, and thus the first seen trajectory is classified as valid. This case is represented by the second type of noise that was created (i.e., the second scenario). The f1-score achieved by the algorithm in the second scenario was 22.43%, while the precision was 41% and the recall was 15.3%. Fig. 7 visualizes the second scenario, which is an extreme case, and compares the true positives with false positives/negatives.

Comparing Fig. 4 and Fig. 7, we can see that the algorithm detected as spoofing events trajectories that were not. Also, in the second scenario we had an increased number of false negatives (messages that were falsely classified as valid), which is why the recall was so low.

TABLE III. PERFORMANCE EVALUATION

AVG. NOISE RATIO	F1 Statistics		
	Min	*Average*	*Max*
0.2272597	91.93045	92.88076	93.40959
1.1362555	90.29008	91.87921	92.42006
2.2732047	91.44832	91.71793	92.11491
4.5471772	90.99060	91.11057	91.36623
6.8167651	90.2043	90.46079	90.7442

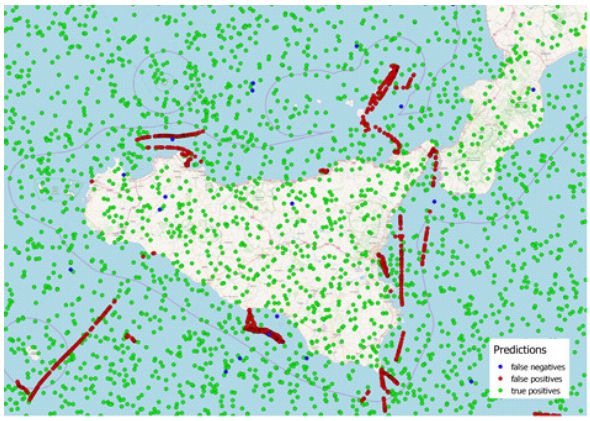

Fig. 6. Spoofing events vs Detected events in the first scenario. Green dots are the correctly identied noise messages, blue dots are not detected noise (false negatives) and red dots are missclassified valid trajectories (false positives).

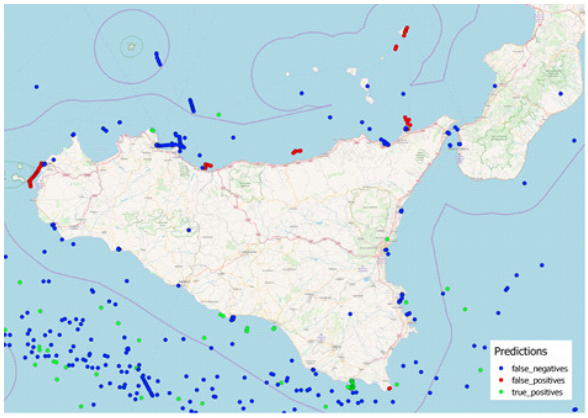

Fig. 7. Spoofing events vs Detected events in the second scenario. Green dots are the correctly identied noise messages, blue dots are not detected noise (false negatives) and red dots are missclassified valid trajectories (false positives).

V. CONCLUSIONS AND FUTURE WORK

In this work, we presented a distributed architecture for real-time spoofing detection. We demonstrated its use in a real-world application and showed its increased performance in terms of accuracy, using a varying number of noise levels injected artificially in the stream. As future work, we aim to improve the spoofing detection algorithm to handle more use cases of real world noise in the streams of AIS messages. Furthermore, we plan to enhance the system architecture with better fault tolerance techniques in case of node failures and test the reliability of the system in more advanced ways.

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 732310 and supported by the Program of Industrial Scholarships of Stavros Niarchos Foundation.

REFERENCES

[1] Carlos Santamaria, Virginia Fernandez Arguedas, Pietro Argentieri, Marlene Alvarez, Harm Greidanus, and Mattia Stasolla, "Sentinel-1 maritime surveillance-Testing and Experiences with Long-term Monitoring," European Commission / Joint Research Centre Ispra, Italy, Feb. 2016.

[2] L. M. Millefiori, D. Zissis, L. Cazzanti, and G. Arcieri, "A distributed approach to estimating sea port operational regions from lots of AIS data," in 2016 IEEE International Conference on Big Data (Big Data), 2016, pp. 1627–1632.

[3] G. Spiliopoulos, D. Zissis, and K. Chatzikokolakis, "A Big Data Driven Approach to Extracting Global Trade Patterns," in Mobility Analytics for Spatio-Temporal and Social Data, 2017, pp. 109–121.

[4] M. Balduzzi, "AIS Exposed Understanding Vulnerabilities & Attacks 2.0," in BlackHat Asia, 2014.

[5] C. Ray, C. Iphar, and A. Napoli, "Methodology for Real-Time Detection of AIS Falsification," 2016.

[6] F. Papi et al., "Radiolocation and tracking of automatic identification system signals for maritime situational awareness," Sonar Navig. IET Radar, vol. 9, no. 5, pp. 568–580, 2015.

[7] F. Katsilieris, P. Braca, and S. Coraluppi, "Detection of malicious AIS position spoofing by exploiting radar information," in Proceedings of the 16th International Conference on Information Fusion, 2013, pp. 1196–1203.

[8] M. Guerriero, P. Willett, S. Coraluppi, and C. Carthel, "Radar/AIS data fusion and SAR tasking for Maritime Surveillance," in 2008 11th International Conference on Information Fusion, 2008, pp. 1–5.

[9] M. Guerriero, S. Coraluppi, C. Carthel, and P. Willett, "Analysis of AIS Intermittency and Vessel Characterization using a Hidden Markov Model.," 2010.

[10] F. Mazzarella, M. Vespe, A. Alessandrini, D. Tarchi, G. Aulicino, and A. Vollero, "A novel anomaly detection approach to identify intentional AIS on-off switching," Expert Syst. Appl., vol. 78, pp. 110–123, Jul. 2017.

[11] D. Zissis, "Intelligent security on the edge of the cloud," in 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC), 2017, pp. 1066–1070.

[12] L. Salmon and C. Ray, "Design Principles of a Stream-based Framework for Mobility Analysis," Geoinformatica, vol. 21, no. 2, pp. 237–261, Apr. 2017.

[13] https://akka.io/

[14] https://flink.apache.org/

[15] Loic Salmon and Cyril Ray. 2017. Design principles of a stream-based framework for mobility analysis. Geoinformatica 21, 2 (April 2017), 237-261.