Preparation Phase Report

Obfuscated Location Disclosure for RemoteID-enabled UAVs

Supervisor:
Assistant Professor, dr. S. Sciancalepore
Security (SEC) Cluster, Department of
Mathematics and Computer Science
Eindhoven University of Technology

Written by:
M. Schotsman
MSc. Information Security Technology
Eindhoven University of Technology

Contents

List of Acronyms

CI	Critical Infrastructure
CS	Control Station
DRIP	Drone Remote ID Protocol
FAA	Federal Aviation Administration
GRR	Generalized Randomized Responses
LSB	Location Based Services
NFZ	No-Fly Zone
PIM	Planar Isotropic Mechanism
RemoteID	Remote Identification of Unmanned Aircraft
UAV	Unmanned Aerial Vehicle
ZSP	Zone Service Provider

1 Introduction

The use of Unmanned Aerial Vehicles (UAVs), better known as drones, is increasing in many different application areas. They can be used for search and rescue, surveillance, delivery of goods, agriculture, and many more fields of work [1]. But UAVs are not only being used to better society. They can also be used in malicious ways, such as dropping harmful items [2], transporting forbidden goods [3], and invading/disrupting Critical Infrastructures (CI) such as airfields [4].

To regulate UAV traffic, the Federal Aviation Administration (FAA) has published a regulation named Remote Identification of Unmanned Aircraft (RemoteID) [5]. This regulation applies to almost all UAVs except for some very specific cases, such as UAVs specifically designed for aeronautical research. RemoteID states that a UAV should broadcast a message at least once every second containing the following parameters: a unique identifier, UAV's location, UAV's altitude, UAV's velocity, Control Station's (CS) location, CS's altitude, time mark, and emergency status indication. All of these values are sent in plain text and the messages are broadcasted such that anyone could receive and read them.

Because anyone could read these messages, malicious adversaries might use the information contained in RemoteID messages against the UAV, the CS, and the goods that the UAV is carrying. To protect the UAV and its package, instead of sending the plain location information, the UAV could disclose an obfuscated location, such that an adversary could no longer track the UAV. This is a similar countermeasure to a kind of related problem present for Location Based Services (LSB), which focuses on obfuscating locations so users cannot be tracked. The LSB here can be an app that shows a user the list of restaurants based on the user's location. The user does not want to reveal their exact location but the obfuscated location must still be close enough to retrieve a meaningful list of restaurants. Such conflicting requirements call for a trade-off between the security that the user can get and the utility of the location.

In the literature, there exists a paper [6] tackling a similar problem in our context, in which an obfuscation technique was created based on Geo-Indistinguishability [7]. However, the method which was chosen was not the best fit for this scenario. This is because the privacy gets greatly diminished when reporting the location frequently [8], while also being vulnerable to several interference attacks that could reveal the actual location [9]. That is why this report first dives into the literature to find the most suitable method to obfuscate the location of a UAV while taking many requirements into account. Also, the technique from [6] was only implemented on a computer and it was not shown if their solution could run on a UAV and fulfil the requirements of Remoteid, e.g. send out a message each second. Instead, we plan our solution to be implemented on a Raspberry PI [10], after evaluating it could also be implemented in a UAV if the overhead is deemed small enough. We will also provide an extensive performance assessment of the solution, in terms of required time, memory, and energy on the selected hardware device. This is to show that the chosen method works for the actual use case. All the challenges discussed above are non-trivial to be solved and have not yet been explored.

This report is organized as follows. Section 2 gives more context about the RemoteID specification and about the Drone Remote ID Protocol (DRIP) working group. Section 3 describes the scenario including all the actors, the requirements for the obfuscation method, and the adversarial model. Section 4 dives deep into the literature to find the most suitable method to obfuscate the UAV's location. Section 5 explains the methodology used for the rest of the project. Finally, Section 6 illustrates the planning for the rest of the Graduation Phase.

2 Preliminaries

This section explains in more detail the RemoteID regulation and the working group that is creating standards for the RemoteID.

2.1 RemoteID

RemoteID [5] is a regulation that was published in 2021 by the FAA. This regulation forces almost all UAVs, except for some specific cases such as UAVs that are specifically created for aeronautical research, to send out a RemoteID message at least once each second. A RemoteID message consists of a unique identifier, UAV's location, UAV's altitude, UAV's velocity, Control Station's (CS) location, CS's altitude, time mark, and emergency status indication. There are two different types of RemoteID namely Network RemoteID and Broadcast RemoteID. In the Network RemoteID, the UAV or CS sends the RemoteID message to a third-party service provider over the Internet. While in the Broadcast RemoteID version, the UAV will broadcast RemoteID using Wi-Fi or Bluetooth, such that anyone in the near area could receive the RemoteID messages. This report

using WiFi or Bluetooth, such that anyone in the near area could receive the RemoteID messages. This report

will only apply to the Broadcast RemoteID version.
If the UAV has the required hardware it can broadcast the RemoteID message itself, otherwise, a RemoteID Broadcast Module will have to be used. This broadcast module does all the computation and communication. In a special area called FAA-recognized identification areas, there is no requirement to send out RemoteID messages. Starting from the 16th of September 2023, all UAVs must comply with the RemoteID ruling. The goal of RemoteID is to combat UAVs that are flying in areas they should not be flying, such as near a CI or invading an NFZ. The RemoteID message is completely plaintext, and the regulation contains no plans to encrypt or obfuscate private information, such as the UAV's or CS's location.

2.2 DRIP

The RemoteID regulation did not mandate specific techniques that should be used for deploying RemoteID in practice. The Drone Remote ID Protocol (DRIP) [11] working group works on defining specific techniques that should be used when working with RemoteID. This consists of specifying which specific protocols should be used for RemoteID and also creating documents that list all requirements to use the protocols in combination with RemoteID. Because the UAV research area is still young and this regulation has been created very recently the working group also tries to create new protocols specifically for RemoteID-enabled UAVs. Because DRIP defines the specifics when working with RemoteID, all the actors and the architecture assumed in this project are based on the ones under the standardization of the DRIP working group.

3 Scenario Description

This section will describe the scenario including all the actors, actors' capabilities, communication capabilities, and their constraints. There is also a section about the adversary model which goes over different attacker models to get a broad overview of all possible attacks.

3.1 Scenario

An overview of the scenario description can be found in Figure 3.1. This overview shows a UAV, which is being controlled by an operator, near a No-Fly Zone (NFZ). The UAV operator is located at the Control Station (CS). A UAV is a constrained device, with constraints in multiple different aspects such as computational power, memory, battery life, weight, and cost [12]. From takeoff to shut down, the UAV has to broadcast RemoteID messages containing a unique identifier, UAV's location, UAV's altitude, UAV's velocity, CS' location, CS' altitude, time mark, and emergency status indication. The UAV broadcasted location must be within 100 feet, approximately 30 meters, of the current UAV location. A RemoteID message can be broadcasted using either Wi-Fi or BlueTooth. For this work, we will assume the use of Wi-Fi, and possibly, BlueTooth for broadcasting RemoteID messages.
Completely separated from the UAV and the NFZ there is also a Trusted Third Party (TTP). This TTP could store the UAV's unique identifier but could also store cryptographic material to secure RemoteID messages. This storing and retrieving of information can be done with so-called registries. In [11], there are different kinds of registries, most importantly the public registry and the private registry. The public registry can be used by anyone to retrieve any public information from the unique identifier. While the private registry can only be used by certain parties to retrieve private information.
An NFZ can be around a Critical Infrastructure (CI), such as an airport. This NFZ does not have to be a particular shape, however, it is important that one or more receivers of the CI cover the entire NFZ area. Depending on the size of the NFZ, this could be achieved by one or many receivers. These receivers should support Wi-Fi and BlueTooth to ensure that all the broadcasted RemoteID messages will be received. Once a RemoteID message has been received, it is the job of a public safety observer [11] to check whether a UAV has invaded an NFZ or not. We assume that a CI has an Observer, which we will call the CI Observer. If a UAV is still outside of the NFZ perimeter then no action has to be taken. However, once the UAV has invaded the NFZ, appropriate action should be taken to ensure the privacy and safety of the CI.

3.2 Adversary Model

In this report, we consider only a single adversary, namely the Eavesdropper. The eavesdropper's goal is to track the UAV based on RemoteID messages. This can be achieved by receiving the broadcasted RemoteID messages

the UAV based on RemoteID messages. This can be achieved by receiving the broadcasted RemoteID messages

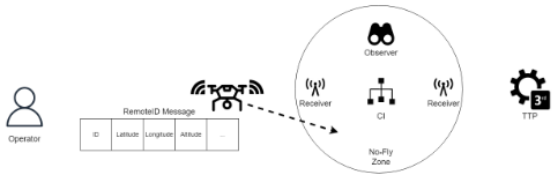

Figure 1: Overview of the scenario

via a receiver and storing the locations. From these stored locations the attacker could even predict where the UAV is going to fly. All of this can be exploited to capture a UAV and steal anything it's carrying. This could lead to damages to the UAV and to the package. Because the UAV is tracked, the attacker could find out at which location a package is dropped off and steal it from there. But not only the UAV location is broadcasted the CS location is also contained in RemoteID. This could lead to harm being done to the UAV operator. Note that we assume that the UAV will always have RemoteID messages enabled, and thus will always broadcast its location. If RemoteID messages are disabled then this report will no longer be applicable.
At the same time, the CI Observer should still be able to identify when a UAV has accidentally invaded the NFZ. We do not consider this an adversary because this was done without any malicious purpose. The reason this is explicitly mentioned is that lots of incidents happen by users that accidentally invade an NFZ [13] and we want to prevent this.

3.3 Solution in a Nutshell

To combat the malicious actor of tracking and predicting the UAV's flight path, the UAV might obfuscate its location. With obfuscation, it is meant that instead of the real location being broadcasted a fake location is broadcasted instead. This obfuscation could lead to creating a large distance between two consecutive RemoteID messages. An unobfuscated velocity could give information about the possible places the UAV could be in. To combat any attack which uses the plain velocity, the velocity also needs to be obfuscated while taking the previous and the following location into account. The altitude could also be obfuscated. If this is done then the obfuscated velocity should take this into account. However, obfuscating the altitude will add extra overhead to the already constraint UAV. All these obfuscated positions should be calculated such that an attacker is unable to accurately predict the UAV's exact location.
By obfuscating the location and velocity, the CI might not be fully sure if a UAV has invaded their NFZ. The TTP that is already storing unique identifiers, such as the FAA, can help by confirming whether a UAV has invaded an NFZ or not. The UAV will encrypt the actual location and velocity with the public key of a TTP and added at the end of the RemoteID message. This TTP could for example be the FAA, which created the regulation [5]. The TTP can decrypt the location and report back to the CI Observer whether a UAV has invaded an NFZ.

3.4 Requirements

From the above scenario description, a list of the 8 most applicable requirements for our case is created. These requirements will serve as a baseline to rate different State-Of-The-Art Techniques.

- R1 The latitude and longitude must be obfuscated, from takeoff to shut down, when sending a RemoteID message. This will ensure that the location of a UAV is always obfuscated such that an eavesdropper cannot know where the UAV exactly is.
- R2 Velocity must be obfuscated from takeoff to shut down. This is to combat information leakage about the possible UAV location.
- R3 Altitude should be obfuscated from takeoff to shut down. This will prevent an eavesdropper to know when a UAV is ascending and descending. This could potentially reveal when a package has been dropped for a package delivery UAV.

R4 An adversary must not be able to calculate the real location from the obfuscated location with respect to temporal correlations. If an adversary were able to calculate the real location then all the location privacy would be lost.

R5 The obfuscated location might be bounded (by 30 meters). This will help with increasing the utility of the UAV, such that the CI Observer can still identify when a UAV has entered an NFZ.

R6 The obfuscated velocity should take the previous and the current location into account. This will make it such that obfuscated velocity will be in line with the obfuscated locations. Otherwise, some information might be leaked.

R7 The method to obfuscate the location should work independently for each UAV. Because this system should work for all UAVs, no matter the circumstances, it should also work in scenarios where there is only a standalone UAV.

R8 Each obfuscated location must be calculated at most within 1 second. Because RemoteID messages have to be sent at least once each second, the obfuscated location must be calculated within the same period for all the locations to be obfuscated.

The above requirements are used to evaluate the suitability of the related work to our scenario in Section 4.

4 Related Work

This section will explain the current state-of-the-art techniques to obfuscate the UAV and its location. Section 4.1 explains the techniques to obfuscate the location. While Section 4.2 will explain the techniques for making the UAV more anonymous. Finally, Section 4.3 will weigh the found techniques against the requirements from 3.4.

4.1 Obfuscation

This section will be a summary of state-of-the-art techniques that could apply to finding the best-suiting location obfuscated system that can be used in RemoteID Messages. These techniques will be compared against the State-Of-The-Art Requirements in Section 3.4 and the results will be filled in Table 4.3.

When researching what obfuscation methods are already available for UAVs two papers jump out. Namely from Brighente, A. et al. [6] and from Svaigen, A. et al. [14].
In the paper from Brighente, A. et al. [6] the obfuscation method was created based on Geo-Indistinguishability [7]. Geo-Indistinguishability protects the user's exact location while still being able to use location-based services. But these services will be used by an approximate location, in some radius, instead of the exact location. Note that these locations are only in latitude and longitude. This means the latitude and longitude will be obfuscated but there is no implementation with altitude or velocity. Geo-Indistinguishability does support sending multiple locations such that they all are still obfuscated. It also works on standalone systems, thus when there is only 1 UAV.
Geo-Indistinguishability does not preserve a lot of privacy with only a small radius and the privacy gets even more diminished by sending messages frequently [8]. This was also stated in [6], where it states that the amount of privacy goes down the more messages are received. The authors of [9] also show that the actual location could be revealed by various interference attacks.
Geo-Indistinguishability was implemented such that it would work for RemoteID-enabled UAVs in the paper from Brighente, A. et al. [6]. Because the obscured locations could be far away from the real location the authors provided a detection mechanism named ICARUS. In short, when a reported location is in the no-fly zone it will immediately report it as an invasion, but if there is no reported location in a specific time frame, namely 15 seconds, then it will report that there is no invasion. The reason for this interval is to be more confident that there was no actual invasion. The paper does indicate that it could be improved by considering temporal correlations [9], which will be covered later in this section.

The second obfuscation method already in the UAV area is from Svaigen, A. et al. [14]. They created multiple design guidelines, a few of which are important for obfuscating the location and some are useful for anonymity. For the points about anonymity see Section 4.2
One of their suggestions was about Spatio-Temporal Obfuscation, which means that a UAV only reports an area and not its exact location in that area. When multiple UAVs are in the same area, they all report the same area, which makes it harder to track a specific UAV and to know its exact location. Do note that with their implementation of spatial-temporal correlation, it might still be possible to track a UAV. For example, if a UAV flies straight into a new area at a certain speed an attacker could deduce its location. Also in our case, the velocity should be reported. If the real velocity is reported it makes it easier for the attacker to track the UAV. If a fake velocity is given the privacy remains the same.
Another suggestion was to use a privacy protocol that works on a node-to-node basis. Such protocols are often computationally expensive and might drastically limit the operational time of a UAV, which is why this suggestion has been committed in Table 4.3.

Instead of looking only at the obfuscation method in the UAV space, we also looked at similar areas, such as aerospace. In the aerospace world, they also thought about some techniques regarding obfuscated locations. In [15] they created multiple suggestions to increase privacy, most of them were about increasing anonymity such that planes are harder to track, this was already mentioned in Section 4.2.
They did come up with a technique to obfuscate the location of all but one aeroplane. This only works if multiple aeroplanes fly in the same direction and with the same velocity. Then each aeroplane sends only data to each other but one aeroplane still broadcasts its location and shows that it is the leader of the group. This idea might in theory work for UAVs. But due to UAVs being agile, in regards to their turns and velocity, these groups will not exist for an extended period. Also, the leader of the group does not have any obfuscation regarding the location.
Because there is only obfuscation when multiple aeroplanes fly in the same direction at the same speed. This obfuscation method can not be guaranteed to completely work from startup to shutdown. For the UAV that is in the group, its location will be obfuscated and the real location cannot be found out by an eavesdropper.

The rest of this section is based on techniques in the LSB, which focuses on obfuscating locations so users cannot be tracked. While still allowing location-based queries such as finding a list of restaurants based on the user's location.

An extension to the Geo-Indistinguishability technique was covered in [16]. They made it work on location traces and not just a single location disclosure. It tries to lower the computational cost by testing whether new noise has to be generated or that old noise can be reused. The system was created such that it could give enough privacy for an entire day, which they deemed as 30 queries in a day. The problem with querying more often is that the privacy budget will run out and it has to be reset. But this can create a correlation between the data from the old privacy budget and the data from the new privacy budget. So this system will work for frequent intervals in traces, but the problem is that these traces cannot disclose too many locations, otherwise correlations will be found.

Instead of just focusing on obfuscating only a location, the authors of [17] not only obfuscate locations but also events, such as "often moving between two places" or "the user went to the dentist last week". The goal is to hide specific locations/events, with higher privacy, that the user deems sensitive such that an adversary could not tell if an event has happened or not. All the other locations are obfuscated with Geo-Indistinguishability or with the "δ-location set" Privacy. We cannot simply deem one of the trajectories completely sensitive, this will make it such that the adversary cannot distinguish between that trajectory and not that trajectory. But when looking at two arbitrary trajectories the adversary can distinguish them.
Such a system could be implemented in our scenario as long as some specific events/patterns happen. For example, if the UAV often moves packages from location A to location B, we could deem location A and location B sensitive. But if the UAV is flying around and no specific location is sensitive then this system does not add anything on top of the normal Geo-Indistinguishability or the "δ-location set".

In [9] the authors do not only focus on just increasing the location privacy but also on the temporal correlations between the locations. Temporal correlation is the correlation over time. In [18] the system was fully implemented in an open-source repository.

The authors provided a technique called "δ-location set" that aims to provide location privacy for each query. This technique uses a two-coordinate system and creates the area around the user as a giant grid. A user's location can be represented by the x and y coordinates. For each, timestamp we have a vector with all the probabilities for the user being in a cell. A "δ-location" is a subset of the vector with only areas where the user is likely to be. If the user is in an area that was filtered out, there is some 'drift' and this is fixed by choosing a 'surrogate' location that is the closest location, to the real location, which is not filtered out.
From the earlier mentioned "δ-location", we can eventually calculate an obfuscated location. However, the exact details of this calculation have been omitted. Most location releases took less than 0.3 seconds to be computed. A problem with this system is that you need a transition matrix, which was calculated offline. This transition matrix shows how often the user has gone from a cell in the grid to another cell. This requires that there are already some existing traces. Because based on the transition matrix the obfuscated locations are calculated.

A very similar work to [9] was presented in [19]. Instead of finding the lower bound of error of the Planar Isotropic Mechanism (PIM), the authors created a method to find the upper bound of error. This was done by using Generalized Randomized Responses (GRR) instead of Markov Chains. The new system does give more utility than the system from [9]. The randomized distance of GRR is closer to the real location than that from PIM. In our case, it is not a problem that the randomized distances are closer because we would like the obscured location to be somewhat close to the real location. This is to not have too many false positives of a UAV being in an NFZ. This system once again needs a transition matrix, just as in [9]. The main fallback with this system is that there is a chance for a non-obfuscated location to be revealed. Because the system reveals the real location every so often this will not be suitable for our scenario.

In [20] they provide a system using Markov's chain that can obscure the location in real-time while also taking the temporal correlations into account. The authors created a system that can efficiently compute new obscured locations from the current real location and a few previous obscured and real locations. This is done by not finding the exact optimal, but by setting upper and lower bounds to speed up the process significantly. The main drawback of this system is that it requires a transition matrix, just as in [9], but it also requires an offline pre-computation. This pre-computation needs to be done only once as long as the user does not change their mobility profile. This pre-computation took less than 7 hours on average, which could be done at night, but makes it harder to deploy the UAV sporadically or in new locations.

An improved version on [7] is [21]. In the paper, they introduce the concept of geo-indistinguishability but for frequent queries. A cell division technique is used, but unlike in [9, 18, 20] the cells are split up into hexagons instead of square cells. Computing these cells might be costly and in the paper, they assume a location-based service will compute them. These cells also have to change shapes when the user is close to the edge to reduce other computation costs. An obscured location is computed from the user's actual location and location in a cell. On the reported time efficiency, the basic mechanism they provided takes often less than a second and their advanced systems take around 2 seconds. It was not stated if this includes creating the cells and with what kind of system these mechanisms were run. Even though the mechanisms provided support for multiple queries they did not keep temporal correlation in mind. The Table 4.3 is filled in based on the use of the basic system because the advanced system already takes too long to compute for our case.

4.2 Anonymity

This section will summarise the state-of-the-art techniques to anonymise the UAV and its location. These techniques do not obfuscate the location on their own. But they make it harder to track the same UAV or distinguish a real UAV from a fake one. Even though anonymity does not solve our problem it can be added on top of a location obfuscation technique to increase privacy further.

A way to add anonymity is by using pseudonyms [6, 14, 15], this has already been shown in multiple areas including UAVs. Pseudonyms are short-term identifiers. This makes it harder for anyone to follow the same object because its pseudonym is changed constantly. However, if you know the location and time of the object before the pseudonym change then you might still be able to connect the old pseudonym with the new one.

Other anonymity techniques in the UAV area are ARID [22] and A2RID [23]. ARID uses ephemeral pseudonyms, this pseudonym changes for each message and can only be linked to the long-time ID by a TTP such as the FAA, this also provides unlinkability such that an adversary cannot distinguish between two messages from the same UAV or from two different UAVs. On top of this the UAV messages are authenticated such that they cannot be spoofed or replayed. All of these techniques were implemented on a real UAV and a performance evaluation was performed. ARID only takes around 11ms to generate a message while being energy efficient.
A2RID focuses on anonymous direct authentication and remote identification of UAVs specifically designed for the commercial sector. To makes sure A2RID works on a wide variety of UAVs three designs were implemented, the first where the computations are heavy but the memory requirement is low, the second requires a lot of memory while being computationally inexpensive, and the last is designed for severely constrained UAVs. The main benefit of A2RID instead of ARID is that it does not require communication with a TTP in order to verify message authenticity. Just as ARID, A2RID provides anonymity, message authenticity, and protection against replay and spoofing. This has once again been formally verified. A2RID does increase the overhead, both computationally and energy cost, but the authors state that this is a worthwhile trade-off to achieve anonymous direct authentication.

A similar area to UAVs is that of aerospace, in [15] multiple suggestions are made for anonymity and obfuscating. The obfuscating techniques can be found in Section 4.1.
The first suggestion is if 2 aeroplanes fly in a specific zone (MIX-Airspaces) they can stop sending messages such that it will be harder to know which entree to the zone corresponds to which exit of the zone. For our requirements, we need the UAV to send a message each second, so this method cannot be applied.
The second suggested option is to have a silent period in which the aeroplane changes pseudonym when it is close to another aeroplane. This suggestion makes it harder to track a specific aeroplane if multiple have a similar trajectory. However, this once again has the issue of our UAV having to send a RemoteID Message each second.

In [24] a technique using a so-called 'dummy' UAV was implemented. This was done by having a single UAV create multiple dummy UAVs with different locations and flight paths. This accomplishes the goal of making it harder to know which UAV is real and which is fake, thus making the UAV more anonymous.
This method was created such that it will work even when there is a low density of UAVs. This uses the principle of k-anonymity but with k-1 dummies. In their approach, a so-called Zone Service Provider (ZSP) would create the dummy data. To make sure that the dummy UAVs do not go too far from the actual UAV location they are bound by a radius R. The paper states that the anonymity of the UAV was increased by reducing the radius R and increasing the number of dummies. With an R=50 and k=10, the attacker could find out the real location in less than 25% of the cases.
Do note that in our case all the calculations of creating dummies and moving these dummies will be performed by the UAV itself and the UAV will have to send k messages each second which can greatly lessen the battery life of the UAV. There still needs to be a way for the TTP to determine whether a real YAV is in an NFZ or if it is a dummy.

There are already some design guidelines regarding location privacy in the UAV literature. In [14] there are multiple design guidelines, a few of which are important for obfuscating the location and some are useful for anonymity. For the points about location obfuscation see Section 4.1.
Just as in [15] they also suggest pseudonyms to increase anonymity. But a new suggestion was about dummy data. The application sends multiple messages/queries of which only one is the real data. Because of these fake messages, it becomes harder to follow the real UAV. In our case, we need a TTP to still find out which UAV is real, such that it can be dealt with once it has entered a no-fly zone.

4.3 Discussion

In Table 4.3 the papers from Section 4.1 will be put against the State-Of-The-Art Requirements created in Section 3.4. This table shows which methods apply to our application. Each cell can have three different values namely ✓, ×, and nothing. This means that a requirement is met, a requirement is explicitly not met, and there is no info in the paper about the requirement respectively. Some cells require more explanation, those are marked with a *, the explanation of those cells can be found below the table.

	R1	R2	R3	R4	R5	R6	R7	R8
[7]	✓			×	✓		✓	
[14]	✓			×	✓		✓	
[15]	×			✓	✓		×	
[16]	✓			×	✓		✓	
[17]	✓			✓			✓	
[9]	✓			✓			✓	✓
[19]				✓			✓	
[20]	✓			✓			✓	
[21]	✓						✓	✓

Table 1: Requirements comparison of state-of-the-art techniques

Looking at the table, a few columns are completely empty. These are 2,3, and 6, which are respectively about velocity obfuscation, altitude obfuscation, and velocity obfuscation between 2 obfuscated points. All the systems that could somewhat be applicable for our use case protected only the longitude and latitude, while completely disregarding the altitude and velocity of the subject. This can be explained because for LSB the user, in general, only has to send the location data and not the speed it is traveling or on which floor of a building they are. However, the velocity and altitude are important for RemoteID-enabled UAVs, the velocity is important because some information about the UAV's actual location could be derived from the previous locations and the velocity, while the altitude could tell if a UAV is near its destination and is already slowly lowering to the ground. The velocity could be obfuscated if we take the difference between the previous and the current obfuscated location and calculate the velocity from that. This would require some extra calculations but could further increase privacy. Obfuscating altitude is harder because there is no simple calculation to obfuscate it that does not add a lot of overhead. Because we are working on a constrained device with limited time to calculate the obfuscation location we will omit the altitude obfuscation altogether.
Looking at the rest of the columns, the one system that implements the most requirements is [9], with missing requirements 2,3,5, and 6. We have already discussed about 2,3, and 6, requirement 5 is that the obfuscation location might be bounded (by 30 meters). This is a requirement to ensure that the utility of the obfuscation locations is high, and thus a CI Observer will not have too many false negatives and has a high amount of true positives. However, there are also some other ways to lower the false negatives and increase the true positives as stated in [6]. Even though [9] is missing a few requirements, we still think it is the most suitable state-of-the-art method based on the requirements from Section 3.4.

5 Research Question and Methodology

This section will explain the methodology for the rest of the project. It starts with defining a research question and the corresponding sub-questions. After that, the future process will be explained and finally, the methodology of evaluating the final result is made clear.

5.1 Research Question

When determining the research question there are many aspects to take into account. We have to think about the security-utility tradeoff and specifically the utility of RemoteID messages for the CI. If we ignore the CI utility it will be difficult to determine if a UAV has invaded an NFZ. To find a suitable obfuscation method we first have to determine the requirements of such a method. Also, the obfuscation method has to be implemented on

a UAV, which is a constrained device. This means that the obfuscation method has to be lightweight otherwise it will not finish the calculation within one second. This must be the case because RemoteID should be sent at least once every second. All of this together leads to the following research question and its sub-questions.
How can we obfuscate the location of the RemoteID-enabled UAV in a lightweight way while allowing remote observers to detect UAV misbehaviour?

Sub-questions
1. What are the requirements for a lightweight obfuscation method that maintains enough utility for a remote observer to detect UAV misbehaviour?
2. What is the most suitable obfuscation method in the literature based on the requirements of sub-question 1?
3. What is the overhead of running the obfuscation method from sub-question 2 in a Remote-enabled UAV?

5.2 Expected Final Result

To answer the research question, we will implement an obfuscation method on a RemoteID-enabled UAV. To find the obfuscation method, we first have to define the requirements for this obfuscation method, this was already done in Section 3.4. The requirements together with literature research on state-of-the-art obfuscation methods lead us to the most suitable obfuscation method for our scenario. This method will eventually be implemented on a Raspberry PI, emulating a RemoteID-enabled UAV. This final implementation will be evaluated based on the utility of the CI Observer, based on the True Positive Rate and the False Negative Rate, and the computational speed of the obfuscation method in a RemoteID-enabled UAV.

5.3 Development Approach

We start by creating the architecture design, this will show all the main software components that need to be implemented to make the obfuscation method in a RemoteID-enabled UAV work. The protocol flow is created to show the communication between different parties, such as the UAV and CI Observer. There is already an implementation of the chosen obfuscation method in Matlab, but we would like it to be in C such that it will work on most devices and specifically a UAV. There already exists a toolbox that can change Matlab to C [25], this will greatly help. However, this might introduce some bugs or inefficiencies that should be fixed before continuing. Once the C code runs on a laptop we will implement it on a Raspberry PI and test the code once again. If this all went as expected it will be implemented in a UAV to show that the whole system works together. If at some point the computation time for a location is more than one second, we will attempt to speed up the process while still maintaining the same output.

6 Planning

In Table 2 the planning for the preparation and the graduation phases are shown. The preparation phase took roughly 2 academic quartiles while the graduation phase will take roughly 6 months. The preparation phase makes sure that a sufficient amount of time is spent on the literature research and that the research question has been properly defined. The architecture design is meant to define all the building blocks and to already find possible problems between connecting the building blocks. After that, the complete implementation of the state-of-the-art method and the communication between the UAV and server, which both run on the laptop, will be implemented. After this is done the UAV part will be implemented on a Raspberry PI. This will work as a proof of concept step. If after evaluating the overhead of implementing the obfuscation method then the system could also be implemented on an actual UAV. From the results, we can finalize the report and finish it with a presentation for the graduation defense.

6.1 Risk Analysis

There already exists an implementation of the state-of-the-art method from [9] that is written in MATLAB. To save time the decision has been made to automatically change the code from MATLAB to C with the help of toolboxes [25]. This could introduce bugs, these should be found as early as possible such that they do not exist anymore when implementing it in the Raspberry PI.

ID	Name	Duration	Start	End
1	**Preparation Phase**	97 days	01-12-2022	19-04-2023
2	Literature Research	54 days	01-12-2022	15-02-2023
3	Preparation Report	84 days	01-12-2022	29-03-2023
4	Preparation Defense Presentation	14 days	29-03-2023	19-04-2023
5	Preparation Defense	1 day	19-04-2023	19-04-2023
6	**Architecture Design**	2 days	20-04-2023	21-04-2023
7	**Laptop Implementation**	50 days	24-04-2023	30-06-2023
8	MATLAB to C	5 days	24-04-2023	28-04-2023
9	Complete Client	20 days	01-05-2023	26-05-2023
10	Listening Server	20 days	29-05-2023	23-06-2023
11	Laptop Testing	5 days	26-06-2023	30-06-2023
12	**Raspberry PI**	35 days	03-07-2023	18-08-2023
13	Raspberry PI Implementation	15 days	03-07-2023	21-07-2023
14	Raspberry PI Communication	15 days	24-07-2023	11-08-2023
15	Raspberry PI Testing	5 days	14-08-2023	18-08-2023
16	**Backup time**	20 days	21-08-2023	15-09-2023
17	**Finalize Report**	15 days	18-09-2023	06-10-2023
18	**Presentation**	10 days	09-10-2023	20-10-2023

Table 2: The scheduled planning for the preparation and graduation phases.

Communication between the laptop and the Raspberry PI might be hard to implement. However, there already exists an open-source repository [26] which should simplify this problem.
The computational system, which is first a laptop and eventually becomes less powerful when it is implemented in the Raspberry PI, could become a bottleneck for computing the RemoteID message in time. If this happens at any time, attempts should be made to improve the code such that it will run faster. If, after many attempts, the computation still takes more than 1 second for a single RemoteID message then it will be hard to recommend obfuscating locations for now. However, in the future UAVs will most have more computational power, and this system could then be used widely. Another option would be to use a so-called broadcast module, which does all the broadcasting and computations. This broadcast module, which was already a suggestion in [5], could be upgraded more easily while still using the exact same UAV.

References

[1] Hazim Shakhatreh et al. “Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges”. In: *IEEE Access* 7 (2019), pp. 48572–48634. DOI: 10.1109/ACCESS.2019.2909530.

[2] Christoph Koettl and Barbara Marcolini. “A Closer Look at the Drone Attack on Maduro in Venezuela”. In: *New York Times* (Aug. 2018). URL: https://www.nytimes.com/2018/08/10/world/americas/venezuela-video-analysis.html (visited on 02/15/2023).

[3] Loch K. Johnson et al. “An INS Special Forum: intelligence and drones/Eyes in the sky for peacekeeping: the emergence of UAVs in UN operations/The democratic deficit on drones/The German Approach to Drone Warfare/Pursuing peace: the strategic limits of drone warfare/Seeing but unseen: intelligence drones in Israel/Drone paramilitary operations against suspected global terrorists: US and Australian perspectives/The ‘Terminator Conundrum’ and the future of drone warfare”. In: *Intelligence and National Security* 32.4 (2017), pp. 411–440. DOI: 10.1080/02684527.2017.1303127. URL: https://doi.org/10.1080/02684527.2017.1303127

[4] "Gatwick Airport: Drones ground flights". In: *BBC* (Dec. 2018). URL: https://www.bbc.com/news/uk-england-sussex-46623754 (visited on 02/15/2023).

[5] FAA. *Remote Identification of Unmanned Aircraft*. Last visited on 15-11-2022. 2021. URL: https://www.faa.gov/news/media/attachments/RemoteID_Final_Rule.pdf.

[6] Alessandro Brighente, Mauro Conti, and Savio Sciancalepore. "Hide and Seek: Privacy-Preserving and FAA-Compliant Drones Location Tracing". In: ARES '22. Vienna, Austria: Association for Computing Machinery, 2022. ISBN: 9781450396707. DOI: 10.1145/3538969.3543784. URL: https://doi.org/10.1145/3538969.3543784.

[7] Miguel E. Andrés et al. "Geo-Indistinguishability: Differential Privacy for Location-Based Systems". In: *CoRR* abs/1212.1984 (2012). arXiv: 1212.1984. URL: http://arxiv.org/abs/1212.1984.

[8] Ricardo Mendes, Mariana Cunha, and Joao Vilela. "Impact of Frequency of Location Reports on the Privacy Level of Geo-indistinguishability". In: *Proceedings on Privacy Enhancing Technologies* 2020 (Apr. 2020), pp. 379–396. DOI: 10.2478/popets-2020-0032.

[9] Yonghui Xiao and Li Xiong. "Protecting Locations with Differential Privacy under Temporal Correlations". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Denver, Colorado, USA: Association for Computing Machinery, 2015, pp. 1298–1309. ISBN: 9781450338325. DOI: 10.1145/2810103.2813640. URL: https://doi.org/10.1145/2810103.2813640.

[10] Last visited on 20-03-2023. URL: https://www.raspberrypi.com/.

[11] *Drone Remote ID Protocol (drip)*. Last visited on 22-03-2023. URL: https://datatracker.ietf.org/wg/drip/about/.

[12] Stuart W. Card et al. "Drone Remote Identification Protocol (DRIP) Requirements and Terminology". In: (2022).

[13] *Worldwide Drone Incidents*. Last visited on 20-03-2023. URL: https://www.dedrone.com/resources/incidents-new/all.

[14] Alisson R. Svaigen et al. "Design Guidelines of the Internet of Drones Location Privacy Protocols". In: *IEEE Internet of Things Magazine* 5.2 (2022), pp. 175–180. DOI: 10.1109/IOTM.008.2100131.

[15] Krishna Sampigethaya and Radha Poovendran. "Privacy of future air traffic management broadcasts". In: *2009 IEEE/AIAA 28th Digital Avionics Systems Conference*. 2009, 6.A.1-1-6.A.1–11. DOI: 10.1109/DASC.2009.5347456.

[16] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. "A Predictive Differentially-Private Mechanism for Location Privacy". In: *CoRR* abs/1311.4008 (2013). arXiv: 1311.4008. URL: http://arxiv.org/abs/1311.4008.

[17] Yang Cao et al. "PriSTE: From Location Privacy to Spatiotemporal Event Privacy". In: *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, Apr. 2019. DOI: 10.1109/icde.2019.00153. URL: https://doi.org/10.1109%2Ficde.2019.00153.

[18] Yonghui Xiao et al. "LocLok: Location Cloaking with Differential Privacy via Hidden Markov Model". In: *Proc. VLDB Endow.* 10.12 (Aug. 2017), pp. 1901–1904. ISSN: 2150-8097. DOI: 10.14778/3137765.3137804. URL: https://doi.org/10.14778/3137765.3137804.

[19] Xingxing Xiong et al. "Locally differentially private continuous location sharing with randomized response". In: *International Journal of Distributed Sensor Networks* 15 (2019).

[20] Wenjing Zhang et al. "Online Location Trace Privacy: An Information Theoretic Approach". In: *IEEE Transactions on Information Forensics and Security* 14 (2019), pp. 235–250.

[21] Jingyu Hua, Fengyuan Xu, and Sheng Zhong. "A Geo-Indistinguishable Location Perturbation Mechanism for Location-Based Services Supporting Frequent Queries". In: *IEEE Transactions on Information Forensics and Security* 13 (2017), pp. 1155–1168.

[22] Pietro Tedeschi, Savio Sciancalepore, and Roberto Di Pietro. "ARID: Anonymous Remote IDentification of Unmanned Aerial Vehicles". In: *Annual Computer Security Applications Conference*. ACSAC '21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 207–218. ISBN: 9781450385794. DOI: 10.1145/3485832.3485834. URL: https://doi.org/10.1145/3485832.3485834.

[23] Eva Wisse et al. "A2RID -Anonymous Direct Authentication and Remote Identification of Commercial Drones". In: *IEEE Internet of Things Journal* (2023), pp. 1–1. DOI: 10.1109/JIOT.2023.3240477.

[24] Alisson Renan Svaigen et al. "A Topological Dummy-based Location Privacy Protection Mechanism for the Internet of Drones". In: *ICC 2022 - IEEE International Conference on Communications*. 2022, pp. 1–6. DOI: 10.1109/ICC45855.2022.9838525.

[25] *MATLAB Coder*. Last visited on 13-03-2023. 2023. URL: https://www.mathworks.com/products/matlab-coder.html.

[26] *Welcome to opendroneid.org*. Last visited on 20-03-2023. 2019. URL: https://www.opendroneid.org/.