University of Amsterdam

MSc System and Network Engineering

Master Thesis

A systematic approach towards GNSS receiver vulnerability analysis on Remotely Piloted Aircraft Systems

Mike Maarse

Supervision at UvA:
Prof. Cees de Laat

Supervision at NLR:
René Wiegers
Judith van Bruggen

Document version 1.1

August 17, 2016

Abstract

Remotely Piloted Aircraft Systems (RPAS) are highly exposed systems due to their wireless nature. Sensory and wireless interfaces aboard the aerial platform have already been proven vulnerable to attack.

In an effort to keep analysis manageable with a growing number of diverse implementations, our research proposes a systematic approach to study and model wireless attacks. The systematic approach builds a theoretical framework which produces an Attack-Defence Tree (ADTree) that visualises and formalises the threat.

Additionally, we implement the intelligence gathered while following the approach by staging and executing a spoofing attack on a representative Global Navigation Satellite System (GNSS) receiver using Software Defined Radio (SDR).

By evaluating the spoofing attack we are able to determine key driving factors that may be used to estimate the likelihood of the attack on a RPAS. Our informal and formal estimations show that spoofing attacks are highly likely, indicating that adversaries are prone to uncover and exploit vulnerable systems.

Acknowledgements

My journey would not have been possible without the assistance of many people along the way.

First and foremost I would like to thank my supervisors at the Netherlands Aerospace Centre (NLR) René Wiegers and Judith van Bruggen, for providing me with the opportunity to work on such an interesting research topic and for their guidance.

Secondly, I would also like to thank Piet Hoogeboom and Hein Zelle for sharing their expertise and enthusiasm on the subject matter and providing me with feedback in times of need.

Next, I wish to extend my gratitude to Jan-Joris van Es for his assistance during the practical experiment. Especially during Friday afternoon troubleshooting.

Although not directly involved, I would also like to thank Nikita Noskov for sharing his insights on the process behind quantitative risk analysis.

I would also like to take the opportunity to thank the NLR in general for facilitating the necessary equipment used to perform the experiment and document my research.

Finally I would also like to thank my family for their continued support and putting up with me working the night shifts.

Mike Maarse
Amsterdam, August 2016

Contents

List of Abbreviations

ADS-B	Automatic Dependent Surveillance Broadcast
AHRS	Attitude Heading Reference System
ARNS	Aeronautical Radio Navigation Service
ATC	Air Traffic Control
BDS	BeiDou Navigation Satellite System
BOC	Binary Offset Carrier
BPSK	Binary Phase Shift Keying
BRLoS	Beyond Radio Line-of-Sight
C/A	Coarse/Acquisition
C2	Command and Control
CDMA	Code Division Multiple Access
CRPA	Controlled Radiation Pattern Antenna
DAG	Directed Acyclic Graph
DGNSS	Differential GNSS
DoS	Denial-of-Service
EKF	Extended Kalman Filter
ESC	Electronic Speed Controller
FDMA	Frequency Division Multiple Access
FHSS	Frequency-Hopping Spread Spectrum
FMS	Flight Management System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
LLA	Latitude, Longitude and Altitude
MiTM	Man-in-The-Middle
NMEA	National Marine Electronics Association
PNT	Positioning, Navigation and Timing
PPD	Personal Privacy Device
PVT	Position, Velocity and Time

RC	Remote Control
RF	Radio Frequency
RINEX	Receiver Independent Exchange Format
RLoS	Radio Line-of-Sight
RPA	Remotely Piloted Aircraft
RPAS	Remotely Piloted Aircraft System
RPS	Remote Pilot Station
SDR	Software Defined Radio
SNR	Signal-to-Noise Ratio
TLS	Transport Layer Security
TTFF	Time To First Fix
UAS	Unmanned Aircraft System

1 | Introduction

Currently, the Remotely Piloted Aircraft System (RPAS) (colloquially known as *drone*) is receiving a lot of attention from researchers and media alike. The market is booming and possible applications [UAV] are being actively researched. Reports indicate that the number of RPAS will only increase in the coming years [Bus].

With the number of implementations and professional applications on the rise, properly securing the operation of RPASs becomes increasingly important as well. Due to these systems being operated remotely, wireless interfaces are continuously exposed while listening for control and navigational inputs including Global Navigation Satellite System (GNSS) signals [HS13].

To prevent adversaries gaining unauthorised control over the RPAS, their wireless communications should be secure by design. Unfortunately, this is not always the case. Research by Rodday and Robinson shows that it is feasible to manipulate RPAS behaviour by attacking wireless communication interfaces [Rod15]; [Rob15].

In a more spectacular but highly controversial incident, Iran captured a Lockheed Martin RQ-170 military Remotely Piloted Aircraft (RPA). The adversary allegedly influenced the on-board navigation system with spoofed GNSS signals [Wikb]. The scientific community has since verified that influencing the trajectory of a RPA is achievable through spoofing, as documented in [Ker+14].

1.1 Contribution

This research proposes a systematic approach to identify possible wireless attacks on RPAS. Consequently, by formalising the threat using visual models, fellow researchers can perform risk assessment and threat mitigation in early stages of system development.

We also implement the systematic approach by staging and executing a spoofing attack on a representative GNSS receiver. It is demonstrated that by performing the practical attack, factors can be gleaned for estimating the attack's likelihood in current RPAS implementations.

1.2 Research question

Our research revolves around two main research questions:

RQ1; *„How can we define a systematic approach to study and model attack paths of wireless attacks on RPAS?"*

This question can be answered by investigating currently applied attack methodologies and available modelling techniques. Additionally, a thorough understanding of how RPASs utilise wireless links will also be required.

RQ2; *„How can we apply the defined approach in a practical experiment involving a GNSS receiver to establish the likelihood of such an attack?"*

Based on the developed approach we should be able to describe an attack against the RPAS. However, we also need to investigate how to exactly leverage GNSS receivers. By performing the practical attack we will be able to ascertain the likelihood.

1.3 Document overview

The remainder of this document is structured as follows. In the next chapter we discuss related efforts conducted by fellow researchers. Chapter 3 provides a basic introduction to RPAS; we will be referring to the system's components throughout this work. In Chapter 4 we describe the systems within scope and the methods used to perform the research.

The findings of our research are revealed in Chapters 5 through 8. First, Chapter 5 provides a detailed target system analysis which is then followed by threat models in Chapter 6. Results of the GNSS spoofing attack are described in Chapter 7. In Chapter 8 we estimate the likelihood of GNSS spoofing attacks occurring in RPAS.

In Chapter 9 we discuss a number of topics related to our results. Subsequently, we conclude our work in Chapter 10. Finally, in Chapter 11 we propose several topics for future work.

Please note that the document is targeted at research in two separate fields; *Aviation* and *Information Technology*. Concepts discussed at length might be familiar to some researchers but might be completely new to others.

Extended threat model The very last page of this document contains the full threat model (ADTree) developed during this research. Due to technical limitations it was not possible to number this page or mention it in the table of contents.

2 | Related work

The amount of research regarding threat modelling (2.1), RPAS security analysis (2.2) and civilian GNSS signal attacks (2.3) is considerable. However, very few efforts seem to combine the topics.

In the following sections we discuss the most relevant efforts that provide the framework of background knowledge used throughout our research.

2.1 Threat modelling

Threat modelling is an important tool to evaluate the security of systems. In 1999, Schneier combined existing efforts into probably the most popular modelling technique; building *attack trees*. In this scheme the root node represents the adversary's goal with subsequent child nodes showing distinct means of achieving that goal. Additional child nodes can then be added in an iterative fashion to model the entire scenario. Another key feature of the technique is the use of **AND** nodes and **OR** nodes to distinguish between combinations and alternatives respectively. A major benefit of attack trees is that the knowledge captured in the model can be re-applied to systems utilising the same components [Sch].

In our research we utilised a recent derivative of attack trees known as *Attack-Defence Trees* (ADTrees) developed at the University of Luxembourg by Kordy et al. The main benefit of ADTrees over attack trees is the ability to include *counter measure* nodes allowing researchers to visualise the "cat-and-mouse game" between adversary and defender. Furthermore, adding child nodes of the same type (i.e. an attack or counter measure node) represents a *refinement*. Nodes without any children of the same type are therefore *non-refined* and represent *basic actions* [Kor+11].

To facilitate the creation of ADTree models, the research group behind ADTrees developed the open source ADTool utility[1]. Moreover, ADTool features several quantitative analysis methods and assists in the assignment of values to individual nodes [Kor+13].

Researchers interested in comparing or reviewing currently available threat modelling techniques may refer to a recent survey of Directed Acyclic Graph (DAG)-based methodologies in [KPS14].

2.2 RPAS security analysis

In 2015, Bruijn and Gratchoff analysed the security of a professional grade RPAS's telemetry link. The investigation revealed multiple security deficiencies in the implementation of the XBee 865/868LP communication modules leaving the system vulnerable to attack. Most notably the configuration used documented defaults and device addresses were not randomised. Moreover, although

[1] The ADTool binary is available through `http://satoss.uni.lu/members/piotr/adtool/`, the source code is available at `https://github.com/tahti/ADTool2`.

available, link encryption had not been enabled. The proof of concept shows that an adversary merely needs to obtain the device address of the XBee module aboard the RPA to establish a rogue telemetry link capable of forwarding Command and Control (C2) traffic [BG15].

Rodday utilised the results from Bruijn and Gratchoff to successfully perform a Man-in-The-Middle (MiTM) attack on the XBee 868LP modules of a RPAS. Subsequently, a number of previously identified *Flight Controller* commands were injected into the channel allowing extensive control over the RPA's behaviour. Furthermore, next to the practical attack, Rodday's work also includes a comprehensive review of several RPAS components and related digital security aspects [Rod15].

During a presentation at DEF CON 23 (2015), Robinson discussed vulnerabilities in the Parrot Bebop consumer grade RPA. Scanning the RPA's IP address revealed open FTP and Telnet ports allowing access to the on-board file and operating system. Moreover, the system did not implement access controls enabling an adversary to manipulate data and execute scripts to control behaviour of the RPA. However, influencing the RPA's in-flight behaviour was also possible without accessing the system. By de-authenticating the genuine operator's controller for 30 seconds adversaries can exploit a safety feature that triggers an automated landing sequence. Additionally, with the genuine operator de-authenticated, control of the RPA can be transferred simply through pairing the RPA to the adversary's controller. Furthermore, Robinson also discusses the use of Global Positioning System (GPS) jammers to prevent the target RPA from calculation its position, this in turn caused the RPA to hover at a fixed point when navigating autonomously (e.g. when executing the return-to-launch function). It was also revealed that GPS jamming affects the DJI Phantom 3 RPA's return-to-launch functionality [Rob15].

2.2.1 Recent exploits

In an effort that focussed on exploiting the unencrypted C2 link of the the Parrot AR.Drone 2.0 RPAS to influence its behaviour, Kamkar developed SkyJack. Through *aircrack-ng* utilities, the SkyJack software scans for the MAC address broadcast by the target RPA and de-authenticates the genuine operator's controller. Next, by means of the *node-ar-drone* library, the software registers the adversary's device as the genuine controller granting full control over the victim's RPA. Moreover, if the adversary also operates a Parrot AR.Drone with SkyJack running on an externally mounted network access point, it is possible to directly relay commands to "slave" RPAs [Kam].

Sasi also investigated the Parrot AR.Drone 2.0 and claims to have developed the first ever back-door payload for RPASs named Maldrone. Once delivered to the target RPA, the software installs itself as a persistent proxy between the *Flight Controller* binary and attached hardware devices through library injection. Following botnet practice, the software automatically connects to the adversary's device to and listens for commands. Supposedly the attack works on every RPA running ARM Linux which also includes the DJI Phantom. However, the attack has only been demonstrated on the Parrot AR.Drone 2.0. At the time of writing, it remains unclear to what extent Maldrone has materialised as no source code has been published [Sas15].

2.2.2 Risk analysis

In addition to the aforementioned practical efforts, several researchers have also taken a more formal approach and based their work around risk analysis.

A commendable effort that combines an RPAS security review and threat modelling can be found in [Hor+15]. In this extensive report, Horowitz et al. analyse RPAS functionality and existing attacks to construct threat models and establish the impact of these attacks on the system. The information subsequently aids in developing the behaviour of a smart *Sentinel* module capable of mitigating potential attacks. To protect the functionality of the RPA, the Sentinel module continuously monitors output from several (navigation and flight critical) components for unexpected values. The action that follows depends on the type of attack and available algorithms but could include strategies such as configuration hopping or simply disabling the compromised component and switching to a backup. Results of testing the *Sentinel* module in simulation and in actual flying conditions aboard a large fixed-wing RPA look promising as the module successfully countered attacks that manipulated GPS data, camera gimbal commands and waypoints.

In [HS13], Hartmann and Steup propose a scheme to assess the risks of RPAS based on the services the system offers (e.g. communication links, sensors, data storage). The result of assessing the risks of each service is a table containing scores on confidentiality, integrity and availability. Another valuable contribution of this paper comes in the form of the abstract model that is used to describe the RPAS components.

2.3 GNSS receiver attacks

Attacks on GNSS receivers appear well researched. Most efforts study the effects of open service (non-military) signal interference and/or suggest counter measures. This is a good thing because millions of applications rely on GNSS signals [Eur]. In general, three attack types are distinguished; *jamming*, *meaconing* and *spoofing* attacks. One research group in particular, the Radionavigation Laboratory at the University of Texas at Austin, appears to be leading in this area of research although their work is primarily focussed on GPS.

2.3.1 Jamming

Borio et al. describe jamming as the deliberate transmission of powerful Radio Frequency (RF) signals which can easily overpower the much weaker GNSS components disturbing and, in some cases, denying GNSS operations. The experiment showed that jamming devices are capable of concurrently influencing signals from GPS and Galileo satellites [BOF13]. However, as mentioned by Bhuiyan et al., true *multi-constellation* receivers (discussed in *subsection 5.2.1*) could provide additional availability and reliability. The suggested jamming detection based GNSS signal selection algorithm could further improve receiver performance [Bhu+15].

In their survey, Mitch et al. investigated the effective range and signal characteristics of 18 commercial GPS jamming devices (commonly marketed as Personal Privacy Devices (PPD)). Results show that the majority of jammers emitted a swept tone (i.e. continuous wave form) to block GPS signals. Furthermore,

their experiment suggests that the actual effective range of jamming devices significantly exceeds the advertised specification. For example, the cigarette-lighter jammer intended for blocking signals aboard a single road vehicle disrupted GPS tracking within 300 m and signal acquisition up to a distance of 1 km [Mit+11]. Leaving consumers in the dark about the effective range can lead to unintentional jamming as was demonstrated by the 2009 incident at Newark International Airport discussed in [PG12, pp. 38–40].

One way of dealing with jammers would be to simply ignore their signal. Such behaviour can be realised through implementing *beam/null-steering* antenna arrays (these solutions are also referred to as Controlled Radiation Pattern Antenna (CRPA)). Brown and Gerein describe *null-steering* as creating an adaptive reception pattern which provides nulls in the direction of a detected jammer (i.e. segments of the sky are blanked out from the receiver's perspective). Instead, *beam-steering* optimises the reception pattern to increase satellite gain. Using a digital *beam-steering* array, in 4 and 16 antenna configurations, the researchers demonstrate reducing the effects of nearby static jamming equipment [BG]. In a more recent article, Curran et al. suggest that the effectiveness of *null-steering* antenna arrays is highly dependent on the quality (i.e. resolution) of signal digitisers [CBF].

Another interesting effort by Yozevitch et al., shows that both presence and position of static jamming equipment can be estimated with reasonable accuracy by combining multiple Signal-to-Noise Ratio (SNR) measurements from one or more receivers. Furthermore, a Bayesian particle filter algorithm is proposed capable of detecting and localising multiple jammers in motion. The algorithm assigns weight to each particle (jammer) based on gathered SNR measurements. During re-sampling, the particles with a lower weight are filtered out [YMS14].

2.3.2 Meaconing

Marnach et al. describe meaconing as the interception and rebroadcasting of navigation signals in order to confuse navigation (based on a definition found in [HJG07]). Obviously similarities exist between meaconing and definitions of traditional (capture and) replay attacks (such as found in [Sta11, p. 319]).

Marnach et al. employed a GPS signal repeater to forward genuine signals from space without modifying their content. Results of verifying the suggested detection algorithm prove the assumption that it is possible to detect meaconing by monitoring the receiver's clock bias over time [Mar+13].

In their work on GNSS spoofing and detection, Psiaki and Humphreys describe meaconing as a method for the adversary to broadcast GNSS signals that might have unpredictable (security) features. Akin to [Mar+13], monitoring the receiver's clock drift is suggested to be especially useful against meaconing as the adversary must maintain a drift rate acceptable by the receiver [PH16].

2.3.3 Spoofing

In 2012, Shepard et al. (of the Radionavigation Laboratory at Austin) performed the first documented spoofing attack on an RPAS using custom built hardware. To generate GPS satellite signals, the hardware implements RX equipment to capture data from genuine signals which can then be altered through a control module. Based on the altered data, multiple satellite signals are simulated and

finally combined into a single bit stream. The TX equipment modulates the bit stream and transmits the counterfeit GPS signal. A major constraint of this set-up is that the adversary's RX antenna has to near the victim's antenna to align the signal. Nevertheless, the spoofing hardware was successful in fooling the receiver aboard the RPA into report drift. Consequently, the RPA's *Flight Controller* started compensating by issuing commands to move in the opposite direction [She+].

A follow-up effort by the same research group, investigated the requirements for gaining control over RPAS navigation systems and covers it in far more detail. Their proof of concept shows that introducing large deviations can force the *Flight Controller* to overcompensate and permanently disable (crash) the RPAS [Ker+14].

In the wake of the successful spoofing attack on the RPAS (of 2012), the Austin research group was involved in spoofing the GPS receiver of the *White Rose of Drachs* super yacht. In the experiment, portable spoofing equipment was placed aboard the vessel. Once again, the attack proved successful in deluding the receiver; in turn causing the vessel's autopilot system and/or crew to navigate along a course laid out by the adversary. As possible counter measure, the research suggests a detection framework that cross-references inputs from multiple sensors [BH15].

While the previously discussed endeavours used custom spoofing hardware, several recent efforts indicate the advent of Software Defined Radio (SDR) based GNSS signal simulators. During DEF CON 23, Huang and Yang discuss synthesising GPS signals and subsequently transmitting samples using relatively cheap SDR hardware such as HackRF, BladeRF and USRP. The gist of the simulator code may be summarised as follows (GNSS terminology will be elaborated upon in *section 5.2*):

1. Load previously gathered *Ephemeris* data
2. Calculate satellites visible to the victim's receiver
3. Generate navigation message for each satellite
4. Convert the bit sequences to wave form
   (a) Calculate transmission times to model signal propagation delay
   (b) Combine satellite signals into single waveform
5. Write signal samples to binary file

Note that step 4a emulates the distance between (each of) the satellites and the receiver, essentially calculating the adversary's intended position for the victim's receiver. By transmitting the generated samples over the air, Huang and Yang claim to have successfully spoofed GPS receivers of smart phones, an in-car navigation system and a DJI Phantom 2 Vision Plus RPAS. In case of the RPAS, by deluding the receiver, the safety measure preventing take-off in a no-fly zone was circumvented [HY15].

Another example of SDR based spoofing was presented during Black Hat Europe 2015 [WCP15], this time using an open source GPS signal simulator. Availability of the simulator significantly reduced the effort (and cost) involved in performing the attack while achieving similar results to Huang and Yang.

Fortunately there are also efforts focussing on counter measures. Recently, Psiaki and Humphreys reviewed state-of-the-art defence strategies against spoofing and meaconing attacks. The research recommends the following practical ways forward to improve receiver design:

1. Check received signal power as spoofed signal are likely to be stronger
2. Monitor clock drift and signal travel time as it propagates from the satellite to the receiver
3. Inspect correlation distortions that cannot be caused by traditional sources of error (requires proper base lining)
4. Use of additional antennas and complementary systems such as Inertial Measurement Unit (IMU)s for cross-referencing
5. Modify the open service signal to accommodate navigation message authentication

Except for modifying the open service signal, all counter measures can be implemented by installing the necessary algorithms on receiver equipment. The included attack/defence matrix shows the probability of detecting a spoofing attack using various counter measures and attack configurations [PH16].

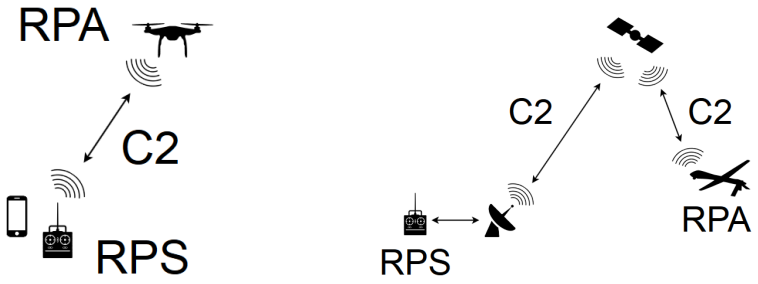
Jovanovic et al. discuss counter measures similar to the ones shown above. Their effort suggests that by combining several spoofing counter measures the number of false alarms can be kept to a minimum [JBF14].

3 | Introducing RPAS

The term Remotely Piloted Aircraft System (RPAS) has been devised to indicate the degree of system autonomy. A RPAS is specific type of Unmanned Aircraft System (UAS) which must involve a human operator at some point during its mission. Hence, systems classified as such cannot be considered fully autonomous. As per ICAO's definition in [ICA15], the RPAS comprises three basic components:

1. a Remotely Piloted Aircraft (RPA);
2. the operator's Remote Pilot Station (RPS);
3. the Command and Control (C2) link between RPA and RPS.

In practice, the aerial platform[1] is usually controlled by a (ground based) RPS through wireless communications. However, this does not imply that the operator is always in the vicinity of the RPA. *Figure 3.1* shows two possible scenarios where the RPS is within (*Figure 3.1a*) and Beyond Radio Line-of-Sight (BRLoS) (*Figure 3.1b*). The latter using satellite communications to transfer C2 messages.



(a) Within Radio Line-of-Sight (RLoS) (b) BRLoS

Figure 3.1: Variable distance operation

Note that ICAO's scheme does not discriminate link users/types or limit the number of C2 links[2]. In our research we distinguish C2 links and data links to on-board payload systems.

The presence of further components within the RPAS is implementation specific and will largely depend on the system's intended purpose. For example, a professional grade surveillance oriented RPAS will encompass a *Flight Controller* capable of navigating waypoints and a camera aboard the RPA. Consequently, RPASs can range from relatively simple to highly complex configurations.

[1] In this document we use the term RPA and *aerial platform* interchangeably.

[2] As was the case in [Rod15], the RPAS used a secondary (telemetry) link with C2 capabilities.

4 | Methodology

Now that we have properly introduced the RPAS we will use this chapter to describe to which implementations our research applies (4.1) and introduce the systematic approach that we will use to study and model attacks (4.2). Additionally, we reveal the details of our practical attack (4.3) and method chosen to estimate its likelihood (4.4).

4.1 Systems in scope

In order to identify RPASs to which this research applies, we need to be clear about the research scope. From the population of RPASs that use GNSS receivers we selected our sample based on application and capabilities of the navigation subsystem.

4.1.1 Application

Our research focusses on systems being used in civilian applications, distinguishing recreational and commercial use. We interpret the listing maintained at [UAV] as a representative overview of commercial applications.

4.1.2 Navigation capabilities

Existing classifications of RPASs are primarily based on performance metrics of the aerial platform. In his thesis, Rodday compares two of such classification schemes [Rod15, pp. 13–16]. However, for the purposes of our research, these schemes are not applicable as they do not include information on the navigation capabilities.

Since autonomous flight is a basic function of RPASs, their navigation capabilities play a central role (e.g. to be able to maintain a current heading). Hence, we developed a classification based on capabilities of the Positioning, Navigation and Timing (PNT) subsystem as shown in *Table 4.1*.

Category	**Sensor type**	**Provides**
I	GNSS receiver	Latitude, longitude, altitude, time
	Pitot-static system	Altitude, airspeed, temperature, pressure
II	Magnetometer	Heading
	Accelerometer	Proper acceleration
	Gyroscope	Pitch, roll, yaw angles
III	Radio altimeter	Altitude
IV	Radio navigation equipment	Position fix
V	RADAR, LiDAR, ground reference	Full situational awareness

Table 4.1: Categorised PNT subsystem capabilities. Note that each subsequent category inherits the capabilities of the preceding category.

The PNT subsystem encompasses all sources of information used to navigate the RPA to its destination. In general the PNT subsystem does not represent a single physical module, but rather a collection of sensory devices. In the PNT based classification scheme, each increment represents more extensive and precise navigation capabilities.

The research focusses on RPASs equipped with category I and II capabilities. Category I sensors represent the bare essentials required for navigating a fixed-wing RPA while category II equipment is required for navigating (and stabilising) single and multi-rotor platforms. Usually combinations of category II sensors are integrated into *IMU* or *Attitude Heading Reference System (AHRS)* solutions [Wika].

Category III and up merely serve as indications that higher-grade sensors exist and in some cases also offer redundancy. Systems equipped as such simply become less reliant on GNSS receiver output under flying conditions which make them less relevant to this research. Nevertheless, it should be considered that high-grade navigation systems might still synchronise the on-board clock using GNSS signals[1].

4.2 Systematic approach

After conducting preliminary research on modelling possible wireless attacks on RPAS, we developed a procedure on how to gather the necessary intelligence. The idea being that fellow researchers may refer to this procedure when performing similar analysis. Additionally, defining the systematic approach is directly related to answering RQ1 (see *section 1.2*).

In the following subsections we describe the specification followed by our implementation.

4.2.1 Specification

The procedure below describes the steps in gathering intelligence on the target system and building a threat model based on the findings. Our goal is to build a theoretical framework that captures the current state-of-the-art. Note that the procedure is intentionally described in abstract form so that it may be applied to investigations regarding either RPASs in general or a specific model.

To keep the model as realistic as possible it is recommended to maintain an adversarial mindset throughout the process.

STEP 1: **Target system analysis**

- In this step we thoroughly investigate the implementation to reveal its components (hardware and software), their interaction and functionality. This can be a highly iterative process in complex implementations where each component comprises several sub-components. Since we are focussing on wireless attacks, topics such as signal properties, remote interfaces, protocols and security mechanisms should receive additional scrutiny.

[1] Clocks are highly relevant to the overall security and robustness of wireless links as their pulse can be used in encryption and frequency hopping schemes.

STEP 2: **Identify attacks and counter measures**

- Now that the system's components have been identified, we can start investigating possible attacks and counter measures. Publications from the scientific and security community are a recommended source of information[2]. Additionally, reviewing the experiments in these publications may also save a lot of time in staging the practical attack later on.

STEP 3: **Formalise threats in visual model**

- The intelligence gathered thus far should suffice to start building an initial version of the threat model (ADTree). In this step we deduce the adversary's actions necessary to perform a specific attack and add them as *attack nodes*. Subsequently, we add available defences as *counter measure nodes*.

STEP 4: **Further refining the model [OPTIONAL]**

- In more complex attack scenarios it is possible the model has not yet been refined to the desired level or the model has become convoluted. A possible solution for further refinement is to focus on developing a specific sub-tree. For example, should a practical experiment follow development of the model, the model could be further refined/improved using the experience gained. Convoluted models may be improved by "pruning" the tree to remove attacks deemed implausible.

4.2.2 Implementation

The subsections below describe our implementation of the systematic approach.

4.2.2.1 Target system analysis

In our research we mainly used abstract functional models of RPAs described in [HS13] and [Hor+15] to identify components that process wireless communication signals and components that use the resulting output to make decisions. Subsequently, to obtain information on a more technical level, we briefly investigated the architecture and source code of open source *Flight Controller* projects including the ArduPilot (APM) [Ardb] and PX4 [PX4] *flight stacks*[3]. Some apparently smaller projects include MultiWii [Mul] and LibrePilot [Lib] which are mainly relevant because of their source code.

Likewise we investigated the GNSS-SDR open source project [Cen] to discover the internals of GNSS receivers. Moreover, the GNSS-SDR documentation nicely complemented the u-blox EVK-6T receiver's operation and protocol specification document [ubl] referenced for the practical experiment.

The results of analysing GNSS receivers and RPAS are discussed in *chapter 5*.

[2]Please be aware that relevant material might be hiding in efforts describing attacks on systems with features similar to RPASs (e.g. computer network equipment, smart phones or modern cars).

[3]Both the APM and PX4 software can be run on readily available *Flight Controller* hardware such as the 3D Robotics Pixhawk [3D].

4.2.2.2 Identify attacks and counter measures

To gather as much intelligence on attacks and counter measures as efficiently as possible, we made extensive use of publications by the scientific and security community. These sources are covered in *section 2.2* and *section 2.3*.

4.2.2.3 Formalise threats in visual model

Based on identified attacks and counter measures we started building the ADTree utilising the ADTool utility [Kor+13]. Unfortunately the utility did not feature numbering capabilities, so we manually added node identifiers for referencing.

4.2.2.4 Further refining the model

Initially we built a generalised model containing possible wireless attacks. Afterwards, we refined the model by developing the C.1 (Alter RPA's calculated position) sub-tree as it relates directly to the practical attack on the GNSS receiver. Note that, in an effort to familiarise ourselves with modelling attacks in more detail, we developed the C.4 (Transmit counterfeit C2 signal) sub-tree based on efforts by Bruijn and Gratchoff and Rodday [BG15]; [Rod15]. Although we do not discuss the C.4 sub-tree in a following chapter we have included it in the extended model (see *Appendix I*).

The resulting threat model can be found in *chapter 6*.

4.3 Mounting the practical attack

Following development of our ADTree we selected the path {A.1, B.2, C.1, D.1, E.1, F.1} (GNSS spoofing) for development into a practical attack. Our target receiver being a u-blox EKT-6T GNSS receiver powered by a LEA-6T-1 chip. Although this chip is only able to process GPS signals, we feel it is representative as the vast majority of currently deployed implementations are relying on GPS rather than other GNSS providers [Eur].

The remainder of the test set-up comprised a Windows 7 workstation running a Ubuntu 16.04 VM in VirtualBox 4.2.12 and a National Instruments USRP-2930 SDR transmitter.

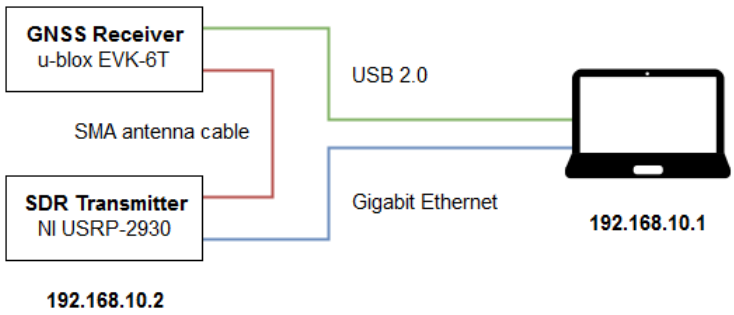


Figure 4.1: Experiment set-up

Due to time and regulatory constraints we were limited to bench-testing using an SMA antenna cable between transmitter and receiver (with a single 30 dB attenuator in place). Note that because of this limitation, we cannot accurately determine boundary conditions for remote attack scenarios.

The set-up is illustrated in *Figure 4.1*. The Ubuntu guest VM was used to control the USRP transmitter while receiver output was processed on the Windows 7 host.

Using a similar approach as Wang et al. [WCP15], we generated counterfeit signals through Ebinuma's GPS-SDR-SIM; an open source SDR based GPS signal simulator [Ebi]. The programme is capable of synthesising baseband GPS signals that simulate receiver motion (dynamic mode) or having the receiver at a fixed position (static mode). In our experiment we performed both static and dynamic simulations.

To generate the signal, GPS-SDR-SIM requires daily broadcast *Ephemeris* data (described in *subsection 5.2.2*) in Receiver Independent Exchange Format (RINEX) format. Although it is possible to obtain current *Ephemeris* data through the receiver [ubl], institutes such as NASA and IGS also collect and publish this data on-line [GNS] [NAS]. After obtaining the *Ephemeris* data, the RINEX file must be passed as a command line argument along with other signal configuration parameters.

By running GPS-SDR-SIM, we generated signal files containing I/Q samples stored in binary format. Subsequently, the *tx_samples_from_file* utility (distributed with the USRP's UHD driver) can be used to transmit the samples.

To configure the receiver and monitor its output, we used version 8.21 of the *u-center* management utility.

Results of the spoofing attack are covered in *chapter 7*.

4.4 Estimating likelihood

By performing the experiment we gained the necessary experience to establish an overall likelihood (see *chapter 8*).

To perform the estimation we used the Risk Rating Methodology described by the Open Web Application Security Project (OWASP) [Ope]. Although OWASP efforts are mainly targeted at web applications, we feel the Risk Rating Methodology can be applied in our research because many aspects of the method are applicable to digital security in general.

5 | Target system analysis

This chapter describes the operation of the aerial platform (5.1) and how its components interact to allow autonomous operation. Subsequently, in similar fashion we describe the basic functionality of GNSS receivers and how they are implemented in RPASs (5.2).

5.1 Aerial platform operation

To fully comprehend the impact of attacks on an RPA we must understand how the system operates. In essence, the aerial platform comprises a set of *controls* (e.g. engine throttle, rudder) that act on *commands* originating from internal or external sources. When a command is executed, the result is measured and corrections are applied if necessary. This kind of mechanism is known as a *control loop* and can be found throughout the system's architecture in varying degrees of complexity. An example two-loop feedback control structure is visualised in *Figure 5.1*. In this diagram, C1 and C2 represent controllers for altitude and pitch respectively.

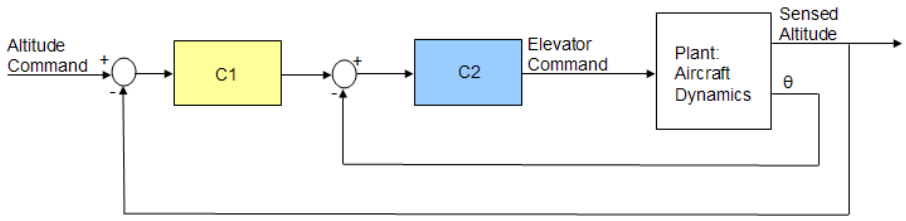


Figure 5.1: Elevator feedback loop [Mat]

The same principle also applies to the aerial platform as a whole. As commands get executed and affect the controls, the *state* of the platform also changes. Sensing the altitude as shown in the figure above represents just one of the many required measurements. Ultimately this results in the system applying continuous corrections to get from a *current state* to a *desired state*.

5.1.1 Core components

Combining the efforts presented in [Bar12], [HS13] and in particular [Hor+15, p. 24], we created a generalised rendition of the core information processing components of the aerial platform. The rendition comprises the *State estimator*, *Flight Management System (FMS)* and *Flight Controller*. The diagram in *Figure 5.2* illustrates an abstract implementation.

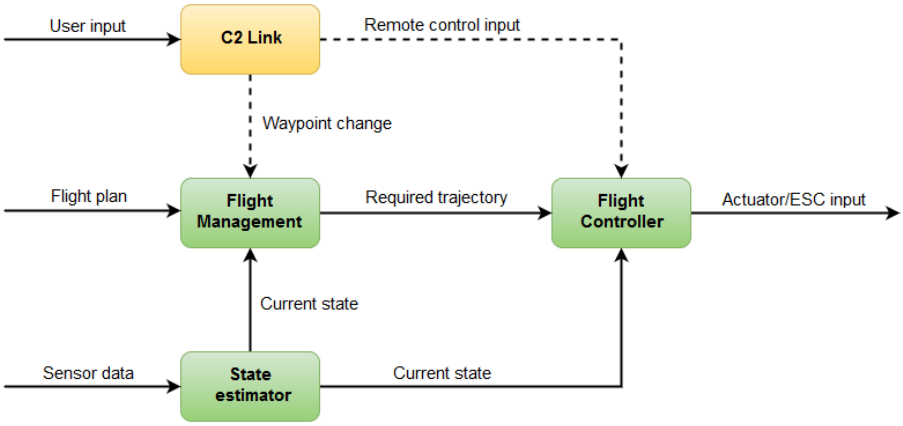


Figure 5.2: Core processing components aboard the RPA (shown in green) and shared components between RPA and RPS (shown in yellow).

The diagram highlights the interaction taking place between components to translate commands into *control surface* or *Electronic Speed Controller (ESC)* inputs. Note that sensing the platform's status results in a closed feedback loop as depicted in *Figure 5.1*.

The paragraphs below describe the on-board components in more detail.

5.1.1.1 State estimator

Estimating the *current state* of the platform is essential to the platform's autonomous operation as it allows the platform to ascertain the corrections required to obtain a *desired state*. Hence, multiple sensors (e.g. accelerometers, gyroscopes) aboard the RPA provide input to a centralised *State estimator*.

State estimators usually implement an Extended Kalman Filter (EKF) or feedback filter to process the output of *sensor fusion* in order to estimate the platform's orientation [Bar12]. Measurements containing significant errors are rejected/filtered during this process. The resulting collection of variables (generally referred to as the *state vector*) represents the platform's current orientation in three-dimensional space (θ, ϕ, ψ), velocities (v_{north}, v_{east}, v_{down}), accelerations (a_x, a_y, a_z) and heading.

Subsequently, state information is propagated to other components for navigation and stabilisation purposes. In addition to on-board use of *State estimator* output, this information can also be relayed to the RPS as telemetry data (this is not shown in the diagram).

5.1.1.2 Flight Management System

The *FMS* maintains the balance between *desired state* and *current state* in navigation terms like trajectory and/or altitude deviations. The *FMS* mainly uses waypoints to determine the desired trajectory and uses deviations between actual position and this trajectory to guide the platform to its destination. It follows that state information is used to determine the current position and to calculate the required change which can include the time to be at a certain location.

Waypoints can be uploaded to the *FMS* prior to flight or modified while the platform is in-flight through the C2 link. The collection of waypoints is generally referred to as the RPA's *flight plan*.

5.1.1.3 Flight Controller

The primary task of the *Flight Controller* is to translate flight commands (e.g. required orientation) into actuator and/or *ESC* input. Actuators and *ESCs* in turn control the platform's *control surfaces* or electric motors respectively.

As shown in *Figure 5.2*, the *Flight Controller* ingests commands from the *FMS* or directly from the operator through the C2 link. The acceptance of commands depends on the selected *flight mode*; autonomous or direct Remote Control (RC).

Additionally the *Flight Controller* uses state information to stabilise the platform. This is most relevant to single and multi-rotor platforms that are required to *loiter* at a fixed position. The effect of winds and rotor vibration continuously produce errors which need to be corrected.

5.2 GNSS receivers

Although there are many different receiver designs, there are a few similarities between them. In the following subsections we describe available signal providers (5.2.1), how the receiver uses these signals to calculate its position (5.2.2) and finally how receiver output is utilised on-board the RPA (5.2.3).

5.2.1 Signal providers

Currently there are 4 providers aiming for global coverage; GPS (United States), Galileo (Europe), GLONASS (Russia) and BeiDou Navigation Satellite System (BDS) (China). All of these providers own a constellation of satellites that orbit the earth and broadcast their messages using electromagnetic (radio) waves.

In general GNSS providers offer two services:

1. open service signals targeted at civilian use;
2. restricted/military signals.

Both signal types are transmitted over carriers in the Aeronautical Radio Navigation Service (ARNS) band and mainly use Binary Phase Shift Keying (BPSK) and Binary Offset Carrier (BOC) modulation techniques (see [Navb]). Typically, centre frequency based identifiers are employed to distinguish available signals. To illustrate, the identifier L1 refers to 1575.42 MHz as used by GPS to transmit the open service signal. Some providers also use an additional suffix to indicate the service type if multiple services are being offered on the same frequency (e.g. GPS offers the open service, denoted by *C/A* and the restricted *precision* service, denoted by *P* on the L1 frequency).

Within each constellation, satellites transmit on the same centre frequency making it a shared resource. Hence, GNSS providers implement *multiple access* methods to allow simultaneous use of the communication channel. Currently all open service signal providers implement Code Division Multiple Access (CDMA)

with GLONASS being the exception by using multiple frequencies (Frequency Division Multiple Access (FDMA)).

Properties of open service GNSS signals are summarised in *Table 5.1* (based on [Navg]).

Provider	Signal	Frequency (MHz)	Channel access	Modulation
BDS (Phase III)*	B1-C	1575.42	CDMA	BOC
Galileo*	E1	1575.42	CDMA	BOC
GLONASS	G1 C/A	1602.00	FDMA	BPSK
GPS	L1 C/A	1575.42	CDMA	BPSK

Table 5.1: Open service GNSS signal properties. Providers marked with an asterisk are still building their constellation and as such are not fully operational at the time of writing. Research suggests that future GLONASS-K2 satellites will also transmit open service signals on the 1575.42 MHz frequency utilising CDMA (see [Nave]).

Hardware manufacturers are already producing *multi-constellation* receivers. These receivers are capable of processing more than one provider's signal for increased precision and service redundancy [Gar+].

5.2.1.1 Security considerations

From a security perspective, there are two fundamental issues with the current implementation of open service GNSS signals [Hum+11]; [PH16].

First, they are unauthenticated making it hard to determine if received signals originated from a legitimate source.

Second, not using an encryption scheme to infuse transmitted signals with entropy only facilitates (re)producing counterfeit signals. Arguably adding encryption to "open service" sounds contradictory and would require a significant effort (i.e. in terms of key distribution and overhead). However, it would add another layer of security.

Adding to the problem is the fact that signals transmitted by GNSS satellites are very weak meaning they are easily overpowered. As an example, [Nat16] mentions -160 dB W for the GPS L1 Coarse/Acquisition (C/A) signal.

We also make the observation that by converging to a single centre frequency, the GNSS jamming counter measure discussed by Bhuiyan et al. [Bhu+15] may no longer be an option in the future since it relies on the same service being offered on multiple frequencies.

5.2.2 Obtaining a position fix

All satellites continuously broadcast *Almanac* and *Ephemeris* data. The *Almanac* data contains coarse orbital parameters for the entire constellation. *Ephemeris* data belongs to the transmitting satellite and contains accurate orbital parameters and clock corrections[1]. Another important distinction is that *Almanac* data remains valid for longer periods of time than the *Ephemeris* data (with exact validity periods differing per provider).

[1]Since *Ephemeris* data is transmitted by all satellites, spoofing this information requires modifying the data on multiple satellites simultaneously for consistency.

When the receiver is powered on it enters an acquisition mode and starts scanning the frequency of the target provider for expected visible satellites (based on *Almanac* data). During this process the receiver attempts to identify individual satellites through correlating known satellite codes and received signals. If there is no *Almanac* data available, the receiver resorts to a brute force method to detect visible satellites that are transmitting navigation signals.

As there are multiple satellites, receivers tend to parallelise the correlation process. Hence, multiple correlators (marketed as *channels*) are implemented to reduce the overall time it takes to identify a satellite's signal. When the receiver positively identifies a satellite it will lock on to its signal and start processing its messages.

For the receiver to calculate its current position relative to the satellite, the receiver must obtain a reference to the satellite's location at a specific point in time. In a mathematical process called trilateration the receiver measures distance between itself and 4 satellites to calculate the time and a position *fix* in three-dimensional space. In an effort to improve the accuracy of the calculation, the receiver utilises the aforementioned *Ephemeris* data to apply corrections. Note that, in practice, the receiver will have to cope with multiple sources of error including clock drift, environmental interference, multipathing and inaccuracies in *Ephemeris* data.

The time elapsed between powering on the receiver and obtaining a *fix* is designated as Time To First Fix (TTFF). To reduce the TTFF, the receiver uses a local clock and stores *Almanac* and *Ephemeris* data in local memory. This allows the receiver quickly lock on to satellite signals [KH05]; [Nat16]; [Navf].

5.2.3 Application in RPAS

Current implementations of GNSS receivers used in RPAS range from single purpose modules to solutions with integrated IMUs and Kalman filtering [Ins]. However complex the implementation may be, in the end the receiver provides the RPAS with three basic measurements; Position, Velocity and Time (PVT) [Navf].

Using PVT information alone the RPAS is already able to establish its current *track* and navigate along a set of waypoints. Arguably, given the fact that GNSS timing information is highly accurate, it makes sense that the RPA also uses this information to synchronise the system clock when navigating waypoints.

In addition, on-board receivers equipped with 2 or more antennas can also be used to determine an RPA's orientation (θ, ϕ, ψ) [Nava].

Apart from the obvious navigation related use cases, GNSS receiver output is also used in Automatic Dependent Surveillance Broadcast (ADS-B) messages [SLM13]. ADS-B messages are used to report position, heading and velocity to Air Traffic Control (ATC) and other aircraft primarily for separation purposes. The recent development of small form factor ADS-B transceivers such as the uAvionix ping [uAv] indicate that RPAS might soon be reporting their position in this manner.

The final use case involves auxiliary systems mounted aboard the RPA that use PVT for geo-referencing. Cameras serve as a good example as they store latitude and longitude coordinates in recorded media.

6 Formalisation

Based on the target system analysis and review of existing attacks and counter measures, we built a threat model utilising the ADTree formalism. In this chapter we elaborate on four sub-trees most relevant to our research.

First, we discuss the top-level nodes, together representing a generalised threat model of remote attacks on RPAS (6.1). Second, we follow along the attack path {A.1, B.2, C.1, D.1, E.1, F.1} in describing GNSS based attacks (6.2).

Please note that the full ADTree model developed during this research is included as the very last page of this thesis. The source XML can be found in *Appendix I.*

6.1 Wireless threats

The model shown in *Figure 6.1* represents a top-level overview of wireless threats against RPASs. Hence, several of the adversary's actions are not fully refined.

Figure 6.1: Sub-tree showing wireless threats. Connecting (arc) lines between refinements depict conjunction (i.e. all sub-goals must be achieved). The presence of a black triangular shape beneath the node indicates the existence of further child nodes.

In the subsections below we briefly discuss the A.1 and A.2 sub-trees shown in *Figure 6.1.*

6.1.1 Eavesdropping

Wireless communication signals emitted by the RPAS are interesting targets to adversaries as they can contain sensitive information. Exactly this type of attack was mentioned in [MQ], where the RPA's on-board camera feed was successfully

captured by adversaries. Another likely application for eavesdropping is to study commands issued by the RPS to *Flight Controller* or *FMS* which may be used later on to stage replay attacks.

6.1.1.1 Determine target frequency

Before being able to listen in, adversaries must first determine the target channel's frequency (ADTree B.3). The centre frequency is likely to be documented or can be detected using a spectrum analyser. A number of common bands used for C2 and payload links are discussed in [HS13] although the 868 MHz band used to perform the MiTM attack in [Rod15] appears to be missing.

6.1.1.2 Sniffing wireless traffic

After locking on to the target frequency, the adversary can proceed to sniff wireless traffic (ADTree B.4). Subsequently, the captured traffic can be either dumped to a file for off-line signal analysis or processed directly to allow real-time eavesdropping. The latter obviously requiring information on how to interpret the traffic.

The SDR receiver mentioned in the model is provided as example equipment.

Reconstruct transmitted data Should the target link prove unencrypted (i.e. messages are sent as *plain text*), the adversary can directly move on to reconstructing data being transmitted over the air (ADTree C.5).

Encrypt C2 messages Encrypting the C2 link has been included in the model (ADTree C.6) since the widespread use of encryption technologies such as Transport Layer Security (TLS) in digital networks makes it a likely counter measure. In this case, the messages will contain *cipher text*. In response to using link encryption, the adversary could assume a more active role and become MiTM (ADTree D.11) between RPS and RPA.

In practice, encryption and authentication schemes are often not implemented or left disabled simply because of the impact on link performance (see [BG15]; [MQ]).

6.1.1.3 Implementing FHSS

Although Frequency-Hopping Spread Spectrum (FHSS) technology (by itself) does not provide any form of confidentiality or integrity [Hav09], it should be considered an important tool for system engineers to make attacking the RPAS more difficult. Hence, it has been added as counter measure B.5 in the model.

However, it should not be ruled out the adversary might be capable of detecting and following the hopping pattern (ADTree C.7). An effort by atlas, Q, cutaway and SoT presented at the SchmooCon 2011 hacker convention describes a possible approach[1] [atl11].

[1] Coincidentally, at the same convention Project Ubertooth was presented by Michael Ossman. The associated hardware platform (Ubertooth One) implements demodulation components handling FHSS in order to monitor BlueTooth traffic [Gre].

6.1.2 Influencing system behaviour

Apart from the previously discussed method of eavesdropping, an adversary may also be interested in actively abusing wireless communications used in the RPAS. These types of attacks might have far-reaching consequences such as loss of control.

6.1.2.1 Blocking C2 signals

In a somewhat rudimentary approach the adversary could attempt to block signals carrying genuine C2 data (ADTree B.1) from ever reaching the RPA basically performing a Denial-of-Service (DoS) attack. As a result the RPA no longer reacts to control inputs from the operator. Moreover, when the RPA is navigating autonomously, it is possible that (as a safety feature) the RPA automatically returns to its launch position as the system detects the loss of the C2 link. Hence, blocking signals could be also interpreted as a means to gaining trajectory control.

6.1.2.2 Gaining trajectory control

Besides blocking the C2 signal, there are more subtle methods for an adversary to gain control over the RPA's trajectory (ADTree B.2). These attacks will result in the RPA deviating from its intended trajectory (*desired state*).

Alter RPA's calculated position In this scenario (ADTree C.1) the adversary targets the *State estimator* and other users of sensory output. The adversary introduces measurement errors by altering inputs of the sensory systems (which includes GNSS receivers) aboard the RPA. As a result, the system will calculate its position based on tainted input. Subsequently, the system assumes it is not at the intended position and starts compensating.

We further elaborate on these type of attacks in *section 6.2*.

Alter RPA's reported position To conceal the effects of a successful attack or force the operator to modify the trajectory, the adversary can attempt to alter the position reported by the RPA (ADTree C.2)[2]. In theory, this can be achieved through modifying downlinked telemetry data (ADTree D.4) given the adversary is a MiTM. In a scenario where the RPA is being monitored by ATC, this action can also be used in an indirect manner by spoofing the RPA's ADS-B signal (ADTree D.5) at an undesirable location (e.g. no-fly zone).

Alter RPA's flight plan Another method that adversaries can use to gain control is altering the RPA's flight plan (ADTree C.3). The adversary could target the *FMS* and upload new waypoints or remove existing ones (ADTree D.6). A critical prerequisite to this attack is control over a mechanism mechanism to reload the waypoints (ADTree D.7). Consequently, performing these attacks will likely require access to the C2 link.

[2]Although the attack could be related to altering the calculated position aboard the RPA, there is a subtle difference in that altering the reported position is focussed on outbound signals.

Alternatively, the adversary could instruct the system to ignore the entire *flight plan* and force the *Flight Controller* to change from autonomous flight mode to direct RC. For instance using the method described in *subsubsection 6.1.2.1*.

Transmit counterfeit C2 signal In the final method to gain control (ADTree C.4), the adversary impersonates the genuine operator and starts transmitting counterfeit C2 signals as a rogue RPS. It follows that the adversary will need a way to reproduce authentic signals. Depending on the implementation; this could involve obtaining access to the channel (ADTree D.8) and either replay or synthesise commands (ADTree D.9) as was demonstrated in [Rod15] and [BG15].

An important consideration for the adversary when transmitting counterfeit C2 signals is that the RPA will continue receive commands from the genuine operator as well. Significant in this respect is the fact that long range RPASs normally use the RLoS link during take-off and landing phases, and the BRLoS link while en-route. The transition between RLoS and BRLoS provides a natural occasion for the adversary to attempt a MiTM attack. The same holds true when using multiple RPSs; the hand-over procedure remains a potential weak spot.

6.2 GNSS receiver threats

In this section we dive into the details of GNSS receiver based threats that can be exploited to ultimately gain trajectory control over the victim's RPA. We also revisit the three attack types discussed in *section 2.3*; *jamming*, *meaconing* and *spoofing* attacks.

Each of the subsections below describe further refinements of our model.

6.2.1 Altering the calculated position

The model shown in *Figure 6.2* contains refinements of altering the RPA's calculated position (ADTree C.1) discussed previously in *subsubsection 6.1.2.2*.

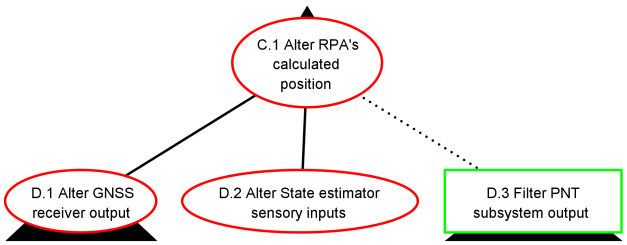


Figure 6.2: Sensor based attacks. As indicated by the black triangle on the top node (C.1), parent nodes are hidden. This sub-tree connects to node B.2.

In discussing the sub-tree we focus on means to affect the output of the PNT subsystem. Recall that we previously distinguished multiple categories of related sensor equipment in *Table 4.1*.

6.2.1.1 Alter GNSS receiver output

One way to influence the calculated position aboard the RPA is to attack the GNSS receiver (ADTree D.1). As previously discussed in *section 2.3* adversaries may perform this attack remotely through the transmission of counterfeit signals or blocking the signal. In turn, this will affect the PVT solution being calculated by the receiver.

If the adversary is successful in deluding the receiver with counterfeit signals, he/she may proceed to modify signal properties to influence the PVT solution. It follows that any components attached to the receiver, including those in charge of navigating the RPA to its destination, will act on falsified information supplied by the adversary.

Blocking the GNSS signal will disrupt the availability of the PVT solution, forcing components to rely on other sensors to provide the same information.

We further refine these types of attacks in *subsection 6.2.2*.

6.2.1.2 Alter State estimator inputs

Although the *State estimator* may also be influenced by altering GNSS receiver output (ADTree D.1), we mention this attack separately (ADTree D.2). Primarily because, as shown by the model in [Hor+15, p. 24], GNSS output does not exclusively pass through the *State estimator*.

In traditional configurations, the *State estimator* gets its data through Category I and II sensory equipment (*Table 4.1*) in the PNT subsystem. Hence, performing a wireless attack besides the aforementioned GNSS attacks might be deemed implausible as it would involve obtaining control over the atmospheric conditions.

However, more exotic configurations should be considered as well. Additional measurement data might be forwarded to the *State estimator* from a ground station across a telemetry link.

Fully compromising the *State estimator* can have serious consequences as the adversary may potentially influence the entire *state vector*. In turn, influencing the autonomous navigation capabilities of the victim's RPA.

6.2.1.3 Filtering PNT output

As perhaps the main defence against manipulated sensor data, many systems implement filters to reject measurements with significant errors (ADTree D.3). As previously mentioned in *subsubsection 5.1.1.1*, *State estimators* usually implement an EKF or feedback filter. Furthermore, it should be considered that GNSS receiver modules or other integrated sensor solutions (such as IMUs) may implement these types of filters in a similar fashion.

During the filtering process, sensor data is gathered from available sources and cross-referenced to detect possible deviations. If the filter detects a significant measurement error, the value will be rejected and will not propagate through the system. Note that this process does not necessarily discriminate sensor types. To illustrate, output from the IMU can be filtered (or corrected) using the GNSS receiver's PVT output which in turn can be filtered using Differential GNSS (DGNSS) measurements (see *subsubsection 6.2.2.2*).

This counter measure is also directly related to the PNT based classification in *Table 4.1*. With each increment in category, more and more sensors

become available for cross-referencing increasing the likelihood that (malicious) measurement errors will be detected.

For the adversary, this translates into keeping the introduced measurement errors within the filter's configured boundaries (ADTree F.9). Hence, this requires high precision estimations of current reference values which could become a major challenge if the RPA is being operated far from the adversary to begin with.

One approach, when performing a GNSS spoofing attack, is to locate a receiver antenna near (or inside) the RPA's area of operation to obtain accurate signal parameters (see *subsection 2.3.3*). This is a realistic scenario if the RPA is flying a prolonged surveillance mission over a geographically small area.

6.2.2 Altering receiver output

The sub-tree in *Figure 6.3* shows the attacks an adversary might employ to alter the GNSS receiver's output. The sub-subsections following the model discuss each of the nodes in more detail.

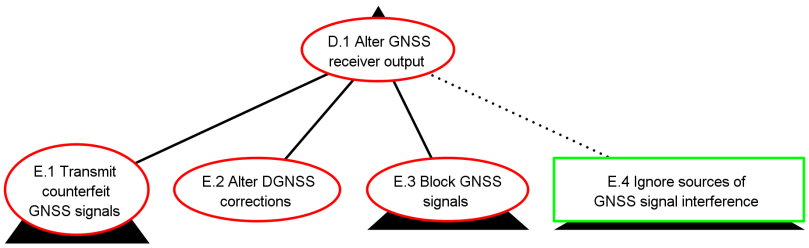


Figure 6.3: Attacks on GNSS signals. This sub-tree connects to node C.1.

6.2.2.1 Transmit counterfeit signals

One way to alter the receiver's output is for the adversary to impersonate a genuine signal source by transmitting counterfeit GNSS signals (ADTree E.1). The aim of this type of attack is to make the victim's receiver use the adversary's signal for calculating PVT information. In turn affecting any component using this information.

This attack is primarily facilitated by the lack of an authentication mechanism in open service signal implementations. Furthermore, although the adversary's transmitting equipment does have to compete with the genuine signals from space, these may be easily overpowered. As mentioned in *subsection 5.2.1*, the GPS L1 C/A signal strength could register as low as -160 dB W.

Typically, transmitting counterfeit signals is achieved through (capture and) replay attacks or spoofing. While replay attacks use existing messages, spoofing attacks generally involve synthesising messages and hiding the fact that they are generated by the adversary. Hence, spoofing attacks allow the adversary further control of message contents.

In *subsection 6.2.3* we elaborate on these types of attacks and their specific counter measures.

6.2.2.2 Alter DGNSS corrections

As described in *subsection 5.2.2*, GNSS signals from space can contain errors which affect the precision of the calculated position. DGNSS is a ground based augmentation solution that uses reference stations at accurately determined positions that track the same satellite(s) as GNSS receivers in its vicinity. Since the reference station's position has been previously determined, it can measure deviations rather accurately and transmit corrections to the receiver. Additionally, the reference stations can also be used for *trilateration* purposes [Navd].

DGNSS is also an interesting attack vector for adversaries. By sending false correction information to the DGNSS interface of the receiver (ADTree E.2), adversaries can introduce errors in PVT information.

It appears the method used to send DGNSS messages to receivers ultimately depends on the implementation. As mentioned in [Navc], protocols even exist to stream GNSS data over the internet. An example implementation in [NTY14] suggests that the attack might involve obtaining access to the C2 link as DGNSS data is propagated through the telemetry link.

6.2.2.3 Block GNSS signals

The fact that GNSS signals are transmitted via radio waves makes them vulnerable to RF interference. Recall that jamming relates to deliberate transmissions of powerful RF signals in order to overpower the signal from space to disturb and/or deny GNSS operations (*subsection 2.3.1*).

By jamming the signal (ADTree E.3), the adversary essentially performs a DoS attack, saturating the target GNSS frequency with unusable signals. This in turn prevents the RPA from receiving genuine GNSS signals and forces the system to rely on other sources (sensors) to provide PVT information. Hence, the consequences can be quite severe when the PNT subsystem only comprises Category I equipment (see *Table 4.1*). Especially when using GNSS for stabilisation purposes.

This attack may be considered highly effective against improperly protected systems using single or *multi-constellation* receivers. Mainly since the majority of open service GNSS signals are transmitted on the same centre frequency (*Table 5.1*).

Unfortunately, from a security perspective, potent GNSS jamming equipment is readily available in the form of PPDs as mentioned by Mitch et al. [Mit+11]. These devices may also be detected using SNR measurements from known locations [YMS14], but how the system responds will depend on the implementation of further counter measures (such as the CRPA solution shown in [BG]).

6.2.2.4 Countering signal interference

Preventing malicious interference types from affecting the receiver (ADTree E.4) could stop adversarial influence right during the signal processing phase (i.e. before propagating a PVT information to other components). Note that we do not refer to interference in the sense of static noise but rather any deliberate method of interfering with GNSS signals.

We recall that Brown and Gerein [BG] discuss *beam-steering* and *null-steering* antenna arrays (CRPA solutions) primarily in the context of jamming (*subsec-*

tion 2.3.1). However, these types of antenna arrays may also be leveraged to reduce interference in general (ADTree F.5). Mainly because jammers, meaconers and spoofers have one common factor; they all emit undesirable signals from one or more locations. Using CRPA solutions the source of interference could simply be "nulled" by the receiver.

Additionally, the receiving equipment could be configured to exclusively use space-based GNSS signals (ADTree F.6). Although this may also be achieved through *null-steering*, there are more straightforward (cheaper) solutions. For example, antennas could be shielded from ground-based transmitters. Or a solution could be implemented that verifies the signal's direction of arrival. The latter method also being mentioned by Psiaki and Humphreys in [PH16].

Carefully monitoring the received signal power per satellite (C/N_0) should also be considered a counter measure (ADTree F.7). This approach exploits another property shared between jammers, meaconers and spoofers; emitted signals by the adversary are stronger than the space-based counterpart. Should the C/N_0 suddenly increase dramatically or attain unrealistic dB-Hz values, the receiver could issue a warning to other components.

Another method to counter signal interference, is to cross-reference the position solution across multiple signal providers (ADTree F.8) and check for deviations. Obviously this approach requires that either *multi-constellation* or multiple distinct *single-constellation* receivers are being implemented. Furthermore, this set-up can also be used to verify the expected satellite reception. Since *Ephemeris* data provides satellite position information, it can serve as a filter on which satellites the receiver should be able to track/emulate. Therefore, combining signals from several GNSS providers, allows detecting interference in the form of illegitimate satellite signals being broadcast by the adversary.

6.2.3 Transmitting counterfeit signals

As discussed in the previous subsection, there are multiple ways for an adversary to transmit counterfeit signals. *Figure 6.4* shows these attacks as child nodes of E.1. In the sub-subsections below we discuss each of these nodes.

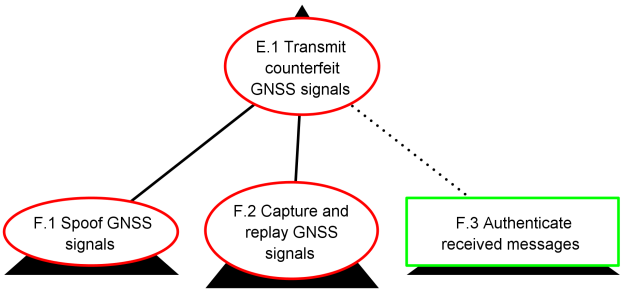


Figure 6.4: Counterfeit signal based attacks. This sub-tree connects to node D.1.

6.2.3.1 Spoofing GNSS signals

When performing a spoofing attack (ADTree F.1), the adversary broadcasts counterfeit GNSS signals with the intent that the receiver aboard the victim's RPA misinterprets the signals as authentic. If the attack is successful, the adversary can directly influence PVT output being calculated by the receiver. In turn, any components attached will have to deal with the tainted input.

Spoofing is achieved by replicating three basic components of GNSS an open service signal; (1) the RF carrier, (2) the *spreading* waveform, and (3) the data bits [PH16]; [KH05]. This can be difficult to maintain in a consistent manner. Hence, executing the attack requires the right combination of software and hardware capable of generating and transmitting the signal (ADTree G.1). Typically, signal simulators are used to this end.

In their work, Jovanovic et al. distinguish *plain signal synthesisers* and *smart spoofers*. A plain signal synthesiser blindly transmits counterfeit signals but does not target a specific receiver. Smart spoofers first estimate the signal parameters currently received by the target receiver and use these parameters as a source of synthesis [JBF14].

Previous efforts such as [CKD12] and [She+] mainly show the use of (expensive) custom built hardware GNSS signal simulators. However, in recent years the advent of SDR based simulators had a profound impact on the required means and expertise to perform the attack [WCP15]; [HY15]. We use an example of the latter in our experiment (see *chapter 7*).

During the attack, the adversary may influence the PVT output by falsifying broadcast satellite data (ADTree G.2). Examples included in the model are:

- **ADTree H.3**: Altering the satellite's clock setting leading to false reports of date and time. This approach was also demonstrated in [HY15]. As a possible counter measure, the receiver could reject jumps in the clock signal (ADTree I.4).
- **ADTree H.4**: Altering the broadcast *Ephemeris* data. In turn, leading to a false position fix. As previously mentioned in *subsection 5.2.2*; spoofing this information requires modifying the data across multiple simulated satellites simultaneously.
- **ADTree H.5**: Setting the satellite's health bit to *false* potentially causing receivers to ignore this satellite long after the initial attack.

6.2.3.2 Capture and replay GNSS signals

In a more straightforward approach, the adversary can simply record genuine GNSS satellite signals and rebroadcast them (ADTree F.2). For instance, using a plain GNSS signal repeater. The attack will confuse any receivers locked on to the same satellites. In *subsection 2.3.2*, we referred to this type of attack as *meaconing*.

During the attack the adversary does not modify the messages but instead aims to delay their arrival. In turn, the delay will cause the victim's receiver to miscalculate its position as the *trilateration* process uses the time-of-arrival to calculate the distance between satellite and receiver [Mar+13]. As a result, PVT output will become tainted.

Research suggests that *meaconing* attacks can potentially spoof any GNSS signal, including the encrypted signals of military/restricted services [PH16].

A possible counter measure against *meaconing* is to monitor the receiver's clock bias over time as it might be challenging for an adversary to maintain a drift rate similar to that of the satellite's atomic clock [Mar+13]; [PH16].

6.2.3.3 Authenticate received messages

The authentication of GNSS messages has been included in the model as a counter measure to the previously mentioned replay and spoofing attacks (ADTree F.3). The lack of an authentication mechanism is an obvious and serious deficiency in current open service GNSS implementations as there is no way to verify the message's origin using just the message. Consequently, the receiver trusts messages originating from illegitimate sources which provides an easy way in for potential adversaries.

Apparently, authenticating open service GNSS messages is not possible by design. Hence, adding such a mechanism will most likely involve modifying satellite signals [PH16, p. 7].

7 | GNSS spoofing results

In our spoofing attack we attempt to delude the GNSS receiver by transmitting synthesised satellite signals using SDR. We simulated both fixed receiver position (7.1) and moving conditions (7.2). During the experiment we also identified factors (7.3) which can be used in estimating the likelihood, these factors will applied in *chapter 8*.

7.1 Simulated static position

Before spoofing the receiver we created a reference data set by recording 5 minutes of receiver output while exposed to genuine L1 C/A signals from GPS satellites. To allow reception, we attached the GPS antenna (u-blox ANN-MS) and mounted at a fixed position. While recording the reference set we obtained a position fix in roughly 25 seconds from a *warm start* condition with the receiver using 8 out of 9 available signals on average. Recall that only 4 of these signals are required to calculate the receiver's position (see *subsection 5.2.2*).

After obtaining our reference data, we disconnected the antenna and created the wired set-up connecting USRP and receiver (see *Figure 4.1*) in order to start transmitting our static position I/Q samples. Initially the UHD driver continuously reported underflow conditions causing transmission delays and leaving the receiver unable to lock on to the spoofed signals. However, after adding a second vCPU to the VM the underflow conditions largely subsided[1].

In a subsequent attempt, the receiver successfully locked on to the counterfeit satellite signals. This was immediately followed by a backward jump in time by the local clock as it synchronised with the signals, showing that the receiver did not reject clock jumps (ADTree I.4). Next, the cached position from the reference test (NLR Marknesse facility) was recalculated to the intended position in Amsterdam. In total, the new position was obtained in roughly 47 seconds.

Description	**Reference**	**Spoofed**	**Unit**
Position accuracy	1.636	17.137	m
Velocity accuracy	0.48	0.37	m/s
Time accuracy	0.002144	0.029557	µs
Latitude deviation	0.00004284	0.00037464	° (degrees)
Longitude deviation	0.00004109	0.00062966	°
Altitude deviation	8.322	22.335	m

Table 7.1: Reported static position accuracy. Measurements represent PVT averages and Latitude, Longitude and Altitude (LLA) standard deviations collected from *u-center* statistics at the end of recorded output.

As shown in *Table 7.1*, the spoofed signals were able to maintain the receiver's position with reasonable accuracy. Further significant observations when

[1]The remaining underflow errors could be related to hardware performance issues, however we did not perform a full root cause analysis.

transmitting the spoofed signals include the signal strength per satellite (C/N_0) and jamming indicator values.

First, the C/N_0 reading was higher when transmitting the spoofed signals (as expected due to using a wired set-up). However, it was not as uniformly consistent as suggested by [WCP15]. Instead of 46 dB-Hz across all satellites we observed values ranging from 43 to 51 dB-Hz.

Second, the receiver's jamming indicator showed an average 57% probability of jamming (and peaked at 100% when signals briefly subsided during an underflow condition). The high probability may be attributed to signal strength crossing the configured threshold [ubl, p. 116]. This finding shows that the receiver implements a combination of the G.7 and F.7 counter measures found in the ADTree.

7.1.1 Time synchronisation

In an experiment conducted separately but in similar fashion to the static position simulation (using the same lab set-up), we targeted the receiver's local clock and thereby its reported time. We were able to successfully delude the receiver by transmitting just a single satellite signal, resulting in a jump in time and date. Again, this shows the receiver did not reject clock jumps or raises an alarm.

Simulating a single satellite was achieved by modifying the source code of GPS-SDR-SIM accordingly.

7.2 Simulated receiver motion

To simulate receiver motion we used GPS-SDR-SIM's capability to process a custom *track* stored as National Marine Electronics Association (NMEA) sentences[2]. Hence, we plotted a path using *Google Earth* and exported it to KML file format and used the freeware *SatGenNMEA* utility (version 4.8i) [Lab] to convert the path to NMEA GPGGA sentences [Dal]. An example GPGGA sentence containing fix information is shown in *Listing 7.1*.

```
$GPGGA,090000.00,5227.43866428,N,00530.86525220,E,...
```

Listing 7.1: Example NMEA position fix sentence

After generating I/Q samples from the NMEA data, we started transmitting them to the receiver. Similar to the static test we successfully obtained a fix in roughly 45 seconds and the receiver's cached position was recalculated to the position of the recorded *track*. Following a brief stationary period, the receiver was reporting it was moving at a steady speed of 50 km/h.

Figure 7.1 shows a plot of the NMEA input file (highlighted in blue) with the receiver's output highlighted in red[3]. Note that, in the plot, the receiver obtained a fix at about 1/5 down the *track* and did not complete the recorded circuit. The latter condition occurred as the transmitter ran out of samples.

[2]The NMEA 0183 standard is commonly used by GNSS receiver implementations to transmit their data in a relatively simple text format [Wikc] [Navh].

[3]A direct comparison between NMEA input and output is also possible as the EVK-6T receiver is capable of generating NMEA messages.

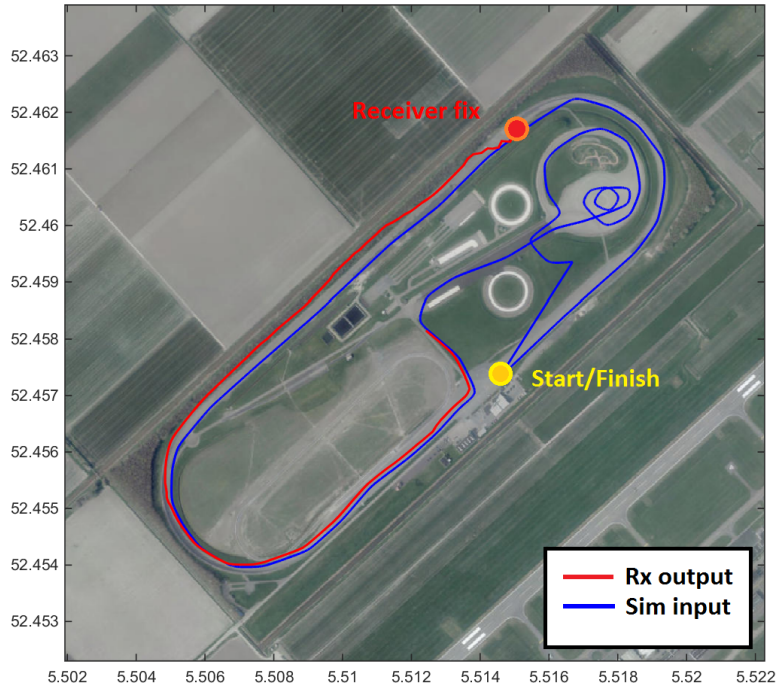


Figure 7.1: Receiver motion

7.3 Evaluation

We were able to successfully delude the u-blox EVK-6T receiver using synthesised open service GPS signals. The attack demonstrates the vulnerability of trusting unauthenticated signals in receiver implementations. Furthermore, the plot in *Figure 7.1* shows the amount of control possible by simulating receiver motion.

The spoofed GPS signals provide realistic receiver output in terms of accuracy and signal strength making it difficult to distinguish spoofed from non-spoofed receiver output.

Given the transmitting equipment is available, the attack is relatively easy to set-up and execute, only little expert knowledge is required. When bench-testing, staging the attack boils down to building the GPS-SDR-SIM main binary and providing a desired location in LLA coordinates.

However, there is a significant difference in using a wired set-up compared to attacking a flying RPA. The adversary will first have to be able to reach the target. This fact alone can increase the cost and complexity of staging the attack considerably. Moreover, if the adversary is aiming for a subtle change of PVT output aboard an RPA at a large distance, he/she will need a highly accurate estimation of the current PVT values.

8 | Estimating likelihood

Through the experiment described in the previous chapter we gathered information on factors that we will now use to estimate the likelihood. In this chapter we demonstrate high-level and numbers driven estimations.

8.1 High-level estimation

GNSS spoofing attacks are public knowledge which reveals a lot of details about how to stage such an attack. It doesn't help that the properties open service GNSS signals have also been disclosed to the public. However, true expert knowledge on the subject might not even be required in the first place. The advent of open source signal simulators such as GPS-SDR-SIM significantly reduces the learning curve [HY15]. Adversaries that do carry expert knowledge can modify the source code and control the content of transmitted messages.

With the simulator capable of generating realistic signals at the push of a button the adversary only has to worry about how to transmit them to the target RPAS. Fortunately for the adversary, SDR transmitters are readily available and affordable. The main challenge of the attack will be how to propagate the signal which will require additional knowledge on antennas.

Adversaries aiming at making more subtle changes to PVT output (to remain undetected) will have to face additional difficulties in obtaining correct reference values.

To conclude, we estimate a **high** likelihood of GNSS spoofing attacks occurring on RPAS. This is primarily due to the ease of staging the attack. We also consider the nature of the spoofing attack as it represents an unauthenticated way of influencing the system's behaviour.

8.2 Quantifying the threat

Although a high-level estimation is valuable as a rough indicator, its form is rather static. The reusable scheme proposed by OWASP allows a more dynamic estimation applicable to different scenarios. The scheme distinguishes factors relating to the *threat agent* and *vulnerability* with each factor being assigned a rating.

To use the scheme we must first specify *who* is attacking the GNSS receiver. For the purpose of our research we will be assuming the adversary is an adept cyber criminal. Key features of this actor are that he/she has basic equipment (such as a laptop) available and has thorough knowledge of systems and digital security.

In the subsections below, we will be assigning ratings[1] to *threat agent* and *vulnerability* factors. Subsequently, these ratings will be used to estimate the overall likelihood.

[1] Assigned ratings are based on those shown in [Ope].

8.2.1 Threat agent factors

The threat agent factors directly relate to the adversary's profile. We consider these ratings more dynamic than vulnerability factors as skill level and motivation are highly subjective. The ratings assigned are shown in *Table 8.1*.

ID	Factor	Rating
T1	Skill level	(6) Network and programming skills
T2	Motivation	(9) High reward
T3	Opportunity	(7) Some access or resources required
T4	Group size	(9) Anonymous users

Table 8.1: Threat agent factors

These ratings were established as follows. The spoofing attack requires programming and radio skills (T1). If successful, the adversary could influence RPA trajectory control (T2). Spoofing equipment needs to be obtained and likely placed in the vicinity of the target RPA (T3). No authentication scheme is employed in current GNSS implementations (T4).

8.2.2 Vulnerability factors

Vulnerability factors are related to the GNSS spoofing attack being discovered and exploited. In determining the ratings for these factors we greatly benefit from gathering intelligence on the RPAS and staging the practical attack. Note that the adept cyber criminal's profile still applies. The ratings assigned are shown in *Table 8.2*.

ID	Factor	Rating
V1	Ease of discovery	(7) Easy
V2	Ease of exploit	(5) Easy
V3	Awareness	(9) Public knowledge
V4	Intrusion detection	(3) Logged and reviewed

Table 8.2: Vulnerability factors

These ratings were established as follows. Well documented example implementations of exploiting this vulnerability exist (V1). Software and equipment readily available (V2). GNSS spoofing attacks are public knowledge through media and scientific publications (V3). Multiple components aboard the RPA and potentially the operator at the RPS monitors receiver output (V4).

8.2.3 Overall likelihood

Using the two sets of ratings shown in *Table 8.1* and *Table 8.2* we can estimate an overall likelihood by calculating the arithmetic mean. By applying $\frac{1}{8}\sum_{i=1}^{8} R_i$ (with R representing a rating) we find an arithmetic mean of 6.875. Subsequently, we can convert this value to the Low-Medium-High (LMH) scheme using *Table 8.3*.

Interval	**Level**
$0 \leq x < 3$	Low
$3 \leq x < 6$	Medium
$6 \leq x \leq 9$	High

Table 8.3: LMH conversion table [Ope]

In conclusion, using the more formal numbers driven method also results in a **high** likelihood. However, the likelihood can now also be related to a specific adversary's profile, that of the adept cyber criminal.

9 | Discussion

9.1 Effects on system capabilities

From the target system analysis in *chapter 5* and discussing the threat model in *chapter 6* it should be apparent that by attacking the GNSS receiver, an adversary affects attached components (e.g. the *State estimator*) by providing a false reference. Hence, if the attack succeeds, the system will process tainted information which can result in undefined behaviour of downstream components.

The *Bowtie* model shown in *Figure 9.1* includes a number of possible consequences that can be caused by a compromised GNSS receiver (i.e. reporting incorrect PVT information).

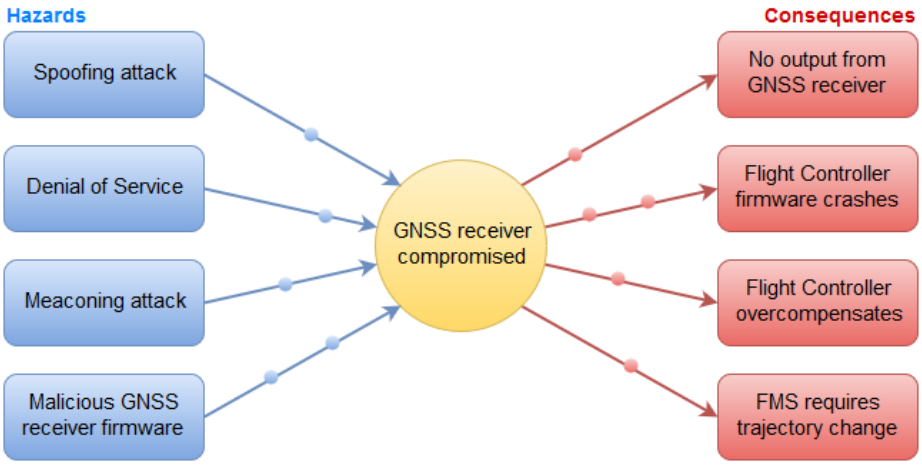


Figure 9.1: Compromised GNSS receiver scenarios. In the Bowtie model, *Hazards* mentioned on the left lead to a *feared event* located at the centre. The feared event in turn leads to possible *Consequences* shown on the right. Blue and red spheres represent *preventative* and *reactive* controls respectively.

Example reasoning based on the model; a possible *preventative control* against spoofing attacks could be the use of a multi-constellation receiver that forces the adversary to concurrently transmit synthesised signals from multiple GNSS providers. Should the GNSS receiver get compromised, a possible *reactive* control might be that the receiver stops generating output when it detects a possible spoofer.

It is also possible the system ignores the spoofer's attempts if the newly calculated (false) position contradicts the predicted position of the EKF. Additionally, downstream components such as the *Flight Controller* could also act on receiver output deviations. For example the ArduPilot software offers GNSS fail-safe and *glitch* protection functionality [Arda].

To conclude, we feel that predicting the exact behaviour under spoofing conditions is only possible through the investigation of a specific implementation,

preferably under realistic (flying) conditions as performed in [Ker+14].

9.2 Modelling technique evaluation

Creating the threat models using the ADTree formalism proved to be a fast and intuitive method of breaking down attack scenarios. However, using tree type graphs to model attacks has one major disadvantage. Different combinations of sub-goals can lead to **more** than one higher level goal being achieved. Although a possible solution would be to simply model these attacks separately, this would lead to a rather large and convoluted model not easily interpretable by humans.

Because of the aforementioned limitation, future analyses on RPAS security might benefit from the use of Bayesian attack graphs as mentioned in [KPS14]. Alternatively neural networks could be considered as well. Mainly due to time constraints we were not able to properly experiment with Bayesian attack graphs or neural networks.

Another, more fundamental limitation of threat models in general is that they are static. Ideally these models should be created dynamically depending on the system configuration. However, this would require a substantial effort as it requires aggregating expert knowledge of system and component security.

Finally, as a recommendation to developers of the ADTool utility we used to build our threat models, we would suggest adding an automatic numbering scheme. This scheme should clearly distinguish attacker and defender goals. Having this feature available would improve the readability of the models significantly.

9.3 Spoofing as an emerging threat

At the time of writing, reports of GNSS spoofing attempts against RPAS outside the scientific community are non-existent. This is perhaps an indication that, although the likelihood of an attack is high, there is nothing to gain for cyber criminals. Arguably the main use for civilian RPASs covered by this research is recreational. However, as the number of commercial RPAS applications increase, we expect that GNSS spoofing attacks will become a very real threat. For example, GNSS spoofing could be used to intercept package delivery RPA's en-route to customers.

10 | Conclusion

This research defines and implements a systematic approach that can be used to analyse wireless attacks on RPASs. In the systematic approach we establish a literary framework comprising a review of existing attacks and a thorough analysis of the target system. Subsequently, we use the gathered intelligence to build visual models highlighting the adversary's goals and probable counter measures. Although the ADTree formalism may not be the most suitable option, it proved very useful in communicating the essence of attack scenarios. A more fundamental issue lies in the fact that the resulting models are rather static in their current form. However, defining the systematic approach can be considered a small step in automating the detection of vulnerabilities.

We also implement the aforementioned systematic approach to stage and execute a spoofing attack against a representative GNSS receiver. In the attack we were successful in controlling the receiver's output. With GNSS receivers being such a fundamental component of current PNT systems this type of attack has a lot of potential.

Actually performing the attack proved to be highly beneficial to accurately modelling the threat as it provides a practical reference and can reveal significant details not available or directly evident from literature. Based on the experience gained we identify relevant factors that can be used to estimate the attack's likelihood. Both informal and formal estimations revealed a high likelihood.

11 | Future work

During the course of this research a number of topics came up that deserve further attention in future efforts. These topics are listed in the paragraphs below.

Wireless GNSS spoofing In our experiment we were limited to using a wired set-up. It would be interesting to see how a receiver behaves under true wireless attacks. This will introduce many new variables such as antenna range, direction and power. Under these conditions the spoofer will have to overpower the genuine signals from space. The primary challenge will probably be to find a controlled environment to conduct the experiment. Note that, although the area is being actively researched, very few works focus on a specific receiver model.

Multi-constellation receivers In anticipation to multiple GNSS signal providers completing their constellation, manufacturers are already producing *multi-constellation* receivers. It will be interesting to evaluate how such a receiver behaves when being spoofed.

Signal variables Although the suggested wireless test can appear captivating, further bench-testing (in a wired set-up) is also possible. Since GPS-SDR-SIM is open source it will be interesting to experiment with the signal's payload and observe receiver behaviour. Additionally, GNSS receiver output parsers can also be tested for their reaction to malformed or tainted inputs.

Risk analysis The estimated likelihood can be used in a risk analysis of RPAS using GNSS receivers. This can be highly relevant to parties defending the operation of these systems.

Bibliography

[3D] 3D Robotics. *Pixhawk Autopilot*. URL: https://pixhawk.org/modules/pixhawk (Retrieved 06/09/2016).

[Arda] ArduPilot Dev Team. *GPS Failsafe and Glitch Protection*. URL: http://ardupilot.org/copter/docs/gps-failsafe-glitch-protection.html (Retrieved 06/09/2016).

[Ardb] ArduPilot Dev Team. *Learning ArduPilot – Introduction*. URL: http://ardupilot.org/dev/docs/learning-ardupilot-introduction.html (Retrieved 06/09/2016).

[atl11] atlas, Q, cutaway and SoT. "Hop Hacking Hedy". In: *ShmooCon* (Nov. 2011), pp. 1–3. URL: https://hedyattack.googlecode.com/files/HopHackingHedy-v1.2.pdf.

[Bar12] J. D. Barton. "Fundamentals of small unmanned aircraft flight". In: *Johns Hopkins APL technical digest* 31.2 (2012), pp. 132–149.

[BG] A. Brown and N. Gerein. "Test results of a digital beamforming GPS receiver in a jamming environment". In: *Proceedings of the 14th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2001)*. (Salt Lake City, UT, United States of America, Sept. 11–14, 2001), pp. 894–903.

[BG15] Y. de Bruijn and J. Gratchoff. "Evaluation of the security of a high class UAV telemetry communication link". MA thesis. Amsterdam, The Netherlands: University of Amsterdam, Jan. 2015. URL: http://rp.delaat.net/2014-2015/p81/report.pdf.

[BH15] J. A. Bhatti and T. E. Humphreys. "Hostile Control of Ships via False GPS Signals: Demonstration and Detection". In: *Journal of the Institute of Navigation (in review)* (2015). URL: https://radionavlab.ae.utexas.edu/images/stories/files/papers/yacht.pdf.

[Bhu+15] M. Z. H. Bhuiyan et al. "Performance analysis of a multi-GNSS receiver in the presence of a commercial jammer". In: *Navigation World Congress (IAIN), 2015 International Association of Institutes of*. Oct. 2015, pp. 1–6.

[BOF13] D. Borio et al. "Jammer impact on Galileo and GPS receivers". In: *Localization and GNSS (ICL-GNSS), 2013 International Conference on*. June 2013, pp. 1–6.

[Bus] Business Insider UK. *THE DRONES REPORT: Market forecasts, regulatory barriers, top vendors, and leading commercial applications*. URL: http://uk.businessinsider.com/uav-or-commercial-drone-market-forecast-2015-2 (Retrieved 06/03/2016).

[CBF] J. T. Curran et al. "Analog and Digital Nulling Techniques for Multi-Element Antennas in GNSS receivers". In: *Proceedings of the 28th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2015)*. (Tampa, FL, United States of America, Sept. 14–18, 2015), pp. 3249–3261.

[Cen] Centre Tecnològic de Telecomunicacions de Catalunya. *GNSS-SDR*. URL: http://gnss-sdr.org/documentation/general-overview (Retrieved 06/01/2016).

[CKD12] M. Cuntz et al. "Future Security: 7th Security Research Conference, Future Security 2012, Bonn, Germany, September 4-6, 2012. Proceedings". In: ed. by N. Aschenbruck et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. Chap. Jamming and Spoofing in GPS/GNSS Based Applications and Services – Threats and Countermeasures, pp. 196–199. ISBN: 978-3-642-33161-9. URL: http://dx.doi.org/10.1007/978-3-642-33161-9_29.

[Dal] Dale DePriest. *NMEA data*. URL: http://www.gpsinformation.org/dale/nmea.htm (Retrieved 06/21/2016).

[Ebi] T. Ebinuma. *GPS-SDR-SIM*. URL: https://github.com/osqzss/gps-sdr-sim (Retrieved 06/01/2016).

[Eur] European GNSS Agency. *GNSS Market Report 2015*. URL: http://www.gsa.europa.eu/system/files/reports/GNSS-Market-Report-2015-issue4_0.pdf (Retrieved 07/01/2016).

[Gar+] F. Garzia et al. *Multi-Constellation. Dual-Frequency. Single-Chip*. URL: http://gpsworld.com/multi-constellation-dual-frequency-single-chip/ (Retrieved 06/22/2016).

[GNS] GNSS Data Center. *GPS - GLONASS - Galileo Tracking Data and Products*. URL: https://igs.bkg.bund.de/ (Retrieved 06/17/2016).

[Gre] Great Scott Gadgets. *Ubertooth One*. URL: http://www.greatscottgadgets.com/ubertoothone/ (Retrieved 08/09/2016).

[Hav09] R. Havelt. "Yes it is too Wi-Fi, and No its not Inherently Secure". In: *Black Hat EU* (2009), pp. 1–13. URL: https://www.blackhat.com/presentations/bh-europe-09/Havelt/Black-Hat-Europe-2009-Havelt-FHSS-Network-Security-whitepaper.pdf.

[HJG07] C. Harpes et al. "Secure localisation with location assurance provider". In: *European Navigation Conference (ENC-GNSS), 2007*. IEEE. 2007, pp. 1–10. URL: https://www.itrust.lu/wp-content/uploads/2009/05/publications_ENC-GNSS_2009_abstract_LAP.pdf.

[Hor+15] B. Horowitz et al. *Security Engineering Project TR-036*. Tech. rep. SERC-2015-TR-036-4. Stevens Institute of Technology: Systems Engineering Research Center (SERC), Jan. 2015, pp. 1–195. URL: http://www.sercuarc.org/wp-content/uploads/2014/11/

SERC-RT-115-Security-Engineering-Pilot-Final-Report-SERC-2013-TR-036-4-Parts-1a-1b-3-4-20150131.pdf.

[HS13] K. Hartmann and C. Steup. "The vulnerability of UAVs to cyber attacks-An approach to the risk assessment". In: *Cyber Conflict (CyCon), 2013 5th International Conference on*. IEEE. 2013, pp. 1–23. URL: https://ccdcoe.org/cycon/2013/proceedings/d3r2s2_hartmann.pdf.

[Hum+11] T. E. Humphreys et al. "A testbed for developing and evaluating GNSS signal authentication techniques". In: *IEEE Transactions on Aerospace and Electronic Systems* (2011).

[HY15] L. Huang and Q. Yang. "GPS Spoofing, Low-cost GPS simulator". In: *DEF CON 23. Hacking Conference*. Aug. 2015. URL: https://media.defcon.org/DEF%20CON%2023/DEF%20CON%2023%20presentations/DEFCON-23-Lin-Huang-Qing-Yang-GPS-Spoofing.pdf.

[ICA15] ICAO. *Manual on Remotely Piloted Aircraft Systems (RPAS)*. Montréal, QC, Canada: International Civil Aviation Organization, 2015. ISBN: 978-92-9249-718-7.

[Ins] Inside GNSS. *The Civilian Battlefield*. URL: http://www.insidegnss.com/node/2509 (Retrieved 06/22/2016).

[JBF14] A. Jovanovic et al. "Multi-test detection and protection algorithm against spoofing attacks on GNSS receivers". In: *2014 IEEE/ION Position, Location and Navigation Symposium - PLANS 2014*. May 2014, pp. 1258–1271.

[Kam] S. Kamkar. *SkyJack*. URL: https://samy.pl/skyjack/ (Retrieved 06/10/2016).

[Ker+14] A. J. Kerns et al. "Unmanned Aircraft Capture and Control Via GPS Spoofing". In: *Journal of Field Robotics* 31.4 (2014), pp. 617–636. ISSN: 1556-4967. URL: http://dx.doi.org/10.1002/rob.21513.

[KH05] E. Kaplan and C. Hegarty. *Understanding GPS: principles and applications*. Norwood, MA, United States of America: Artech house, 2005. ISBN: 1-58053-894-0.

[Kor+11] B. Kordy et al. "Foundations of Attack–Defense Trees". In: *Formal Aspects of Security and Trust: 7th International Workshop, FAST 2010, Pisa, Italy, September 16-17, 2010. Revised Selected Papers*. Ed. by P. Degano et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 80–95. ISBN: 978-3-642-19751-2. URL: http://dx.doi.org/10.1007/978-3-642-19751-2_6.

[Kor+13] B. Kordy et al. "ADTool: Security Analysis with Attack–Defense Trees (Extended Version)". In: *Computing Research Repository* abs/1305.6829 (2013). URL: http://arxiv.org/abs/1305.6829.

[KPS14] B. Kordy et al. “DAG-based attack and defense modeling: Don’t miss the forest for the attack trees”. In: *Computer Science Review* 13–14 (2014), pp. 1–38. ISSN: 1574-0137.

[Lab] Labsat. *SatGenNMEA*. URL: http://www.labsat.co.uk/index.php/en/nmea-download (Retrieved 06/20/2016).

[Lib] LibrePilot Dev Team. *LibrePilot Documentation*. URL: https://librepilot.atlassian.net/wiki/display/LPDOC/Welcome (Retrieved 06/09/2016).

[Mar+13] D. Marnach et al. “Detecting Meaconing Attacks by analysing the Clock Bias of GNSS Receivers”. In: *Artificial Satellites, Journal of Planetary Geodesy* 48.2 (2013), pp. 63–84.

[Mat] Mathworks. *Lightweight Airplane Design*. URL: http://mathworks.com/help/aeroblks/examples/lightweight-airplane-design.html (Retrieved 06/10/2016).

[Mit+11] R. H. Mitch et al. “Signal Characteristics of Civil GPS Jammers”. In: *24th International Technical Meeting, 2011 The Satellite Division of the Institute of Navigation*. Sept. 2011, pp. 1907–1919.

[MQ] M. Mount and E. Quijano. *Iraqi insurgents hacked Predator drone feeds, U.S. official indicates*. URL: http://edition.cnn.com/2009/US/12/17/drone.video.hacked/ (Retrieved 06/03/2016).

[Mul] MultiWii Dev Team. *MultiWii*. URL: http://www.multiwii.com/wiki/index.php?title=Main_Page (Retrieved 06/09/2016).

[NAS] NASA. *Broadcast ephemeris data*. URL: http://cddis.gsfc.nasa.gov/Data_and_Derived_Products/GNSS/broadcast_ephemeris_data.html (Retrieved 06/17/2016).

[Nat16] National Instruments. *GPS Receiver Testing*. May 2016. URL: http://www.ni.com/white-paper/7189/en/ (Retrieved 06/27/2016).

[Nava] Navipedia. *Attitude Determination*. URL: http://www.navipedia.net/index.php/Attitude_Determination (Retrieved 06/21/2016).

[Navb] Navipedia. *Binary Offset Carrier (BOC)*. URL: www.navipedia.net/index.php/Binary_Offset_Carrier_(BOC) (Retrieved 07/05/2016).

[Navc] Navipedia. *DGNSS Standards*. URL: http://www.navipedia.net/index.php/DGNSS_Standards (Retrieved 06/21/2016).

[Navd] Navipedia. *Differential GNSS*. URL: http://www.navipedia.net/index.php/Differential_GNSS (Retrieved 06/21/2016).

[Nave] Navipedia. *GLONASS Future and Evolutions*. URL: http://www.navipedia.net/index.php/GLONASS_Future_and_Evolutions (Retrieved 08/01/2016).

[Navf] Navipedia. *GNSS Receivers General Introduction*. URL: http://www.navipedia.net/index.php/GNSS_Receivers_General_Introduction (Retrieved 06/17/2016).

[Navg] Navipedia. *GNSS signal*. URL: http://www.navipedia.net/index.php/GNSS_signal (Retrieved 06/17/2016).

[Navh] Navipedia. *Interfaces and Protocols*. URL: http://www.navipedia.org/index.php/Interfaces_and_Protocols (Retrieved 06/21/2016).

[NTY14] B. Nizette et al. "Architecture and implementation of an affordable differential GPS system". In: *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*. Oct. 2014, pp. 2321–2326.

[Ope] Open Web Application Security Project (OWASP). URL: https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology (Retrieved 07/04/2016).

[PG12] S. Pullen and G. X. Gao. "GNSS Jamming in the Name of Privacy. Potential Threat to GPS Aviation". In: *Inside GNSS* 7.2 (2012), pp. 34–43.

[PH16] M. L. Psiaki and T. E. Humphreys. "GNSS Spoofing and Detection". In: *Proceedings of the IEEE* 104.6 (June 2016), pp. 1258–1270. ISSN: 0018-9219.

[PX4] PX4 Dev Team. *PX4 Flight Stack*. URL: http://dev.px4.io/concept-flight-stack.html (Retrieved 06/09/2016).

[Rob15] M. Robinson. "Knocking my neighbor's kid's cruddy drone offline". In: *DEF CON 23. Hacking Conference*. Aug. 2015. URL: https://media.defcon.org/DEF%20CON%2023/DEF%20CON%2023%20presentations/DEFCON-23-Michael-Robinson-Knocking-My-Neighbors-Kids-Drone-Offline-UPDATED.pdf.

[Rod15] N. M. Rodday. "Exploring Security Vulnerabilities of Unmanned Aerial Vehicles". MA thesis. Enschede, The Netherlands: University of Twente, July 2015. URL: https://www.jbisa.nl/download/?id=17706129.

[Sas15] R. Sasi. "Drone Attacks: How I hijacked a drone". In: *nullcon Goa'15. Security Conference*. Feb. 2015. URL: http://nullcon.net/website/archives/ppt/goa-15/drone-attacks-how-i-hijacked-a-drone-by-rahul-sasi.pptx.

[Sch] B. Schneier. *Attack trees*. URL: https://www.schneier.com/academic/archives/1999/12/attack_trees.html (Retrieved 06/02/2016).

[She+] D. P. Shepard et al. "Evaluation of smart grid and civilian UAV vulnerability to GPS spoofing attacks". In: *Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012)*. (Nashville, TN, United States of America, Sept. 17–21, 2012), pp. 3591–3605.

[SLM13] M. Strohmeier et al. "Security of ADS-B: State of the Art and Beyond". In: *CoRR* abs/1307.3664 (2013). URL: http://arxiv.org/abs/1307.3664.

[Sta11] M. Stamp. *Information Security: Principles and Practice*. Hoboken, NJ, United States of America: John Wiley & Sons, 2011. ISBN: 978-0-470-62639-9.

[UAV] UAVS Association. *Civil and Commercial UAS Applications*. URL: https://www.uavs.org/commercial (Retrieved 06/07/2016).

[uAv] uAvionix. *ping2020*. URL: http://www.uavionix.com/products/ping2020/ (Retrieved 06/01/2016).

[ubl] u-blox AG. *u-blox 6 Receiver Description and Protocol Specification*. URL: https://www.u-blox.com/sites/default/files/products/documents/u-blox6_ReceiverDescrProtSpec_(GPS.G6-SW-10018)_Public.pdf (Retrieved 06/14/2016).

[WCP15] K. Wang et al. "Time and Position Spoofing with Open Source Projects". In: *Black Hat EU* (2015), pp. 1–8. URL: https://www.blackhat.com/docs/eu-15/materials/eu-15-Kang-Is-Your-Timespace-Safe-Time-And-Position-Spoofing-Opensourcely-wp.pdf.

[Wika] Wikipedia. *Inertial Measurement Unit*. URL: https://en.wikipedia.org/wiki/Inertial_measurement_unit (Retrieved 06/10/2016).

[Wikb] Wikipedia. *Iran–U.S. RQ-170 incident*. URL: https://en.wikipedia.org/wiki/Iran-U.S._RQ-170_incident (Retrieved 06/03/2016).

[Wikc] Wikipedia. *NMEA 0183*. URL: https://en.wikipedia.org/wiki/NMEA_0183 (Retrieved 06/20/2016).

[YMS14] R. Yozevitch et al. "Tackling the GNSS jamming problem using a particle filter algorithm". In: *Electrical Electronics Engineers in Israel (IEEEI), 2014 IEEE 28th Convention of*. Dec. 2014, pp. 1–5.

Appendices

I | Extended threat model

The listing below contains the XML source code of the ADTree developed during this research. It may be imported directly into the ADTool utility (after saving the XML to a separate file) to reproduce the visual model, this has been verified using ADTool version 2.1.1.

```
<?xml version='1.0'?>
<adtree>
<node refinement="disjunctive">
<label>Perform
remote cyber
attack on RPAS</label>
<node refinement="disjunctive">
<label>A.1 Influence
system behaviour</label>
<node refinement="conjunctive">
<label>B.1 Block genuine
C2 signals</label>
</node>
<node refinement="disjunctive">
<label>B.2 Gain
RPA trajectory
control</label>
<node refinement="disjunctive">
<label>C.1 Alter RPA&apos;s
calculated
position</label>
<node refinement="disjunctive">
<label>D.1 Alter GNSS
receiver output</label>
<node refinement="disjunctive">
<label>E.1 Transmit
counterfeit
GNSS signals</label>
<node refinement="conjunctive">
<label>F.1 Spoof GNSS
signals</label>
<node refinement="disjunctive">
<label>G.1 Simulate GNSS
baseband traffic</label>
<node refinement="conjunctive">
<label>H.1 Use SDR based
signal simulator</label>
<node refinement="conjunctive">
<label>I.1 Download broadcast
ephemeris data</label>
```

```
</node>
<node refinement="conjunctive">
<label>I.2 Generate I/Q
samples</label>
</node>
</node>
<node refinement="conjunctive">
<label>H.2 Use hardware based
signal simulator</label>
</node>
</node>
<node refinement="disjunctive">
<label>G.2 Falsify
broadcast
satellite data</label>
<node refinement="disjunctive">
<label>H.3 Alter date/time</label>
<node refinement="disjunctive">
<label>I.3 Alter satellite
clock calibration</label>
</node>
<node refinement="disjunctive" switchRole="yes">
<label>I.4 Reject jumps
in clock signal</label>
</node>
</node>
<node refinement="conjunctive">
<label>H.4 Alter
broadcast
ephemerides</label>
</node>
<node refinement="conjunctive">
<label>H.5 Set satellite
health bit to false</label>
</node>
</node>
</node>
<node refinement="conjunctive">
<label>F.2 Capture and
replay GNSS
signals</label>
<node refinement="conjunctive">
<label>G.3 Use GNSS
signal repeater</label>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>G.4 Monitor clock
bias over time</label>
</node>
</node>
```

```
<node refinement="conjunctive" switchRole="yes">
<label>F.3 Authenticate
received messages</label>
<node refinement="conjunctive">
<label>G.5 Randomise
message IDs</label>
<node refinement="conjunctive" switchRole="yes">
<label>H.6 Mimic
randomisation
algorithm</label>
</node>
</node>
<node refinement="conjunctive">
<label>G.6 Implement message
authentication mechanism</label>
</node>
</node>
</node>
<node refinement="disjunctive">
<label>E.2 Alter DGNSS
corrections</label>
</node>
<node refinement="conjunctive">
<label>E.3 Block GNSS
signals</label>
<node refinement="conjunctive">
<label>F.4 Use
commercial
jammer</label>
<node refinement="disjunctive" switchRole="yes">
<label>G.7 Detect
continuous
wave jamming</label>
</node>
</node>
</node>
<node refinement="disjunctive" switchRole="yes">
<label>E.4 Ignore sources of
GNSS signal interference</label>
<node refinement="disjunctive">
<label>F.5 Implement
beam/null-steering
antenna array</label>
</node>
<node refinement="disjunctive">
<label>F.6 Exclusively
use space-based
GNSS signals</label>
<node refinement="disjunctive">
<label>G.8 Shield antennas
```

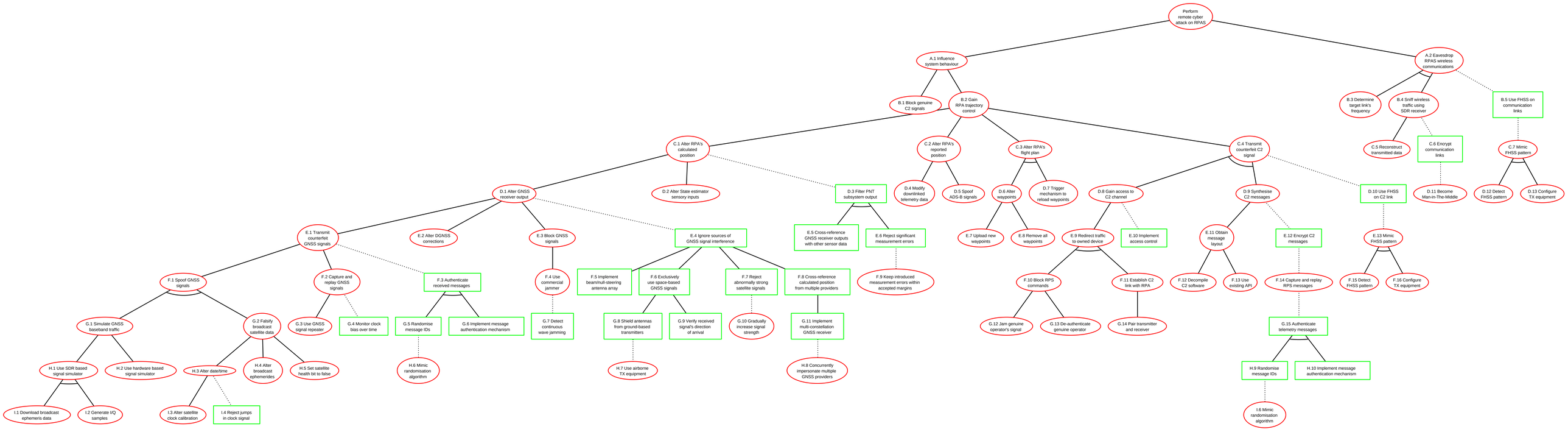
```
from ground-based
transmitters</label>
<node refinement="disjunctive" switchRole="yes">
<label>H.7 Use airborne
TX equipment</label>
</node>
</node>
<node refinement="disjunctive">
<label>G.9 Verify received
signal&amp;apos;s direction
of arrival</label>
</node>
</node>
<node refinement="disjunctive">
<label>F.7 Reject
abnormally strong
satellite signals</label>
<node refinement="disjunctive" switchRole="yes">
<label>G.10 Gradually
increase signal
strength</label>
</node>
</node>
<node refinement="disjunctive">
<label>F.8 Cross-reference
calculated position
from multiple providers</label>
<node refinement="disjunctive">
<label>G.11 Implement
multi-constellation
GNSS receiver</label>
<node refinement="disjunctive" switchRole="yes">
<label>H.8 Concurrently
impersonate multiple
GNSS providers</label>
</node>
</node>
</node>
</node>
</node>
<node refinement="disjunctive">
<label>D.2 Alter State estimator
sensory inputs</label>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>D.3 Filter PNT
subsystem output</label>
<node refinement="disjunctive">
<label>E.5 Cross-reference
GNSS receiver outputs
```

```
with other sensor data</label>
</node>
<node refinement="disjunctive">
<label>E.6 Reject significant
measurement errors</label>
<node refinement="conjunctive" switchRole="yes">
<label>F.9 Keep introduced
measurement errors within
accepted margins</label>
</node>
</node>
</node>
</node>
<node refinement="disjunctive">
<label>C.2 Alter RPA&amp;apos;s
reported
position</label>
<node refinement="conjunctive">
<label>D.4 Modify
downlinked
telemetry data</label>
</node>
<node refinement="conjunctive">
<label>D.5 Spoof
ADS-B signals</label>
</node>
</node>
<node refinement="conjunctive">
<label>C.3 Alter RPA&amp;apos;s
flight plan</label>
<node refinement="disjunctive">
<label>D.6 Alter
waypoints</label>
<node refinement="conjunctive">
<label>E.7 Upload new
waypoints</label>
</node>
<node refinement="conjunctive">
<label>E.8 Remove all
waypoints</label>
</node>
</node>
<node refinement="conjunctive">
<label>D.7 Trigger
mechanism to
reload waypoints</label>
</node>
</node>
<node refinement="conjunctive">
<label>C.4 Transmit
```

```
counterfeit C2
signal</label>
<node refinement="conjunctive">
<label>D.8 Gain access to
C2 channel</label>
<node refinement="conjunctive">
<label>E.9 Redirect traffic
to owned device</label>
<node refinement="disjunctive">
<label>F.10 Block RPS
commands</label>
<node refinement="conjunctive">
<label>G.12 Jam genuine
operator&amp;apos;s signal</label>
</node>
<node refinement="conjunctive">
<label>G.13 De-authenticate
genuine operator</label>
</node>
</node>
<node refinement="conjunctive">
<label>F.11 Establish C2
link with RPA</label>
<node refinement="conjunctive">
<label>G.14 Pair transmitter
and receiver</label>
</node>
</node>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>E.10 Implement
access control</label>
</node>
</node>
<node refinement="conjunctive">
<label>D.9 Synthesise
C2 messages</label>
<node refinement="disjunctive">
<label>E.11 Obtain
message
layout</label>
<node refinement="conjunctive">
<label>F.12 Decompile
C2 software</label>
</node>
<node refinement="conjunctive">
<label>F.13 Use
existing API</label>
</node>
</node>
```

```
<node refinement="conjunctive" switchRole="yes">
<label>E.12 Encrypt C2
messages</label>
<node refinement="conjunctive" switchRole="yes">
<label>F.14 Capture and replay
RPS messages</label>
<node refinement="conjunctive" switchRole="yes">
<label>G.15 Authenticate
telemetry messages</label>
<node refinement="conjunctive">
<label>H.9 Randomise
message IDs</label>
<node refinement="conjunctive" switchRole="yes">
<label>I.6 Mimic
randomisation
algorithm</label>
</node>
</node>
<node refinement="conjunctive">
<label>H.10 Implement message
authentication mechanism</label>
</node>
</node>
</node>
</node>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>D.10 Use FHSS
on C2 link</label>
<node refinement="conjunctive" switchRole="yes">
<label>E.13 Mimic
FHSS pattern</label>
<node refinement="conjunctive">
<label>F.15 Detect
FHSS pattern</label>
</node>
<node refinement="disjunctive">
<label>F.16 Configure
TX equipment</label>
</node>
</node>
</node>
</node>
</node>
</node>
<node refinement="conjunctive">
<label>A.2 Eavesdrop
RPAS wireless
communications</label>
<node refinement="conjunctive">
```

```
<label>B.3 Determine
target link&amp;apos;s
frequency</label>
</node>
<node refinement="conjunctive">
<label>B.4 Sniff wireless
traffic using
SDR receiver</label>
<node refinement="conjunctive">
<label>C.5 Reconstruct
transmitted data</label>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>C.6 Encrypt
communication
links</label>
<node refinement="conjunctive" switchRole="yes">
<label>D.11 Become
Man-in-The-Middle</label>
</node>
</node>
</node>
<node refinement="conjunctive" switchRole="yes">
<label>B.5 Use FHSS on
communication
links</label>
<node refinement="conjunctive" switchRole="yes">
<label>C.7 Mimic
FHSS pattern</label>
<node refinement="conjunctive">
<label>D.12 Detect
FHSS pattern</label>
</node>
<node refinement="disjunctive">
<label>D.13 Configure
TX equipment</label>
</node>
</node>
</node>
</node>
</node>
</adtree>
```


Perform remote cyber attack on RPAS
A.1 Influence system behaviour
A.2 Eavesdrop RPAS wireless communications
B.1 Block genuine C2 signals
B.2 Gain RPA trajectory control
B.3 Determine target link's frequency
B.4 Sniff wireless traffic using SDR receiver
B.5 Use FHSS on communication links
C.1 Alter RPA's calculated position
C.2 Alter RPA's reported position
C.3 Alter RPA's flight plan
C.4 Transmit counterfeit C2 signal
C.5 Reconstruct transmitted data
C.6 Encrypt communication links
C.7 Mimic FHSS pattern
D.1 Alter GNSS receiver output
D.2 Alter State estimator sensory inputs
D.3 Filter PNT subsystem output
D.4 Modify downlinked telemetry data
D.5 Spoof ADS-B signals
D.6 Alter waypoints
D.7 Trigger mechanism to reload waypoints
D.8 Gain access to C2 channel
D.9 Synthesise C2 messages
D.10 Use FHSS on C2 link
D.11 Become Man-in-The-Middle
D.12 Detect FHSS pattern
D.13 Configure TX equipment
E.1 Transmit counterfeit GNSS signals
E.2 Alter DGNSS corrections
E.3 Block GNSS signals
E.4 Ignore sources of GNSS signal interference
E.5 Cross-reference GNSS receiver outputs with other sensor data
E.6 Reject significant measurement errors
E.7 Upload new waypoints
E.8 Remove all waypoints
E.9 Redirect traffic to owned device
E.10 Implement access control
E.11 Obtain message layout
E.12 Encrypt C2 messages
E.13 Mimic FHSS pattern
F.1 Spoof GNSS signals
F.2 Capture and replay GNSS signals
F.3 Authenticate received messages
F.4 Use commercial jammer
F.5 Implement beam/null-steering antenna array
F.6 Exclusively use space-based GNSS signals
F.7 Reject abnormally strong satellite signals
F.8 Cross-reference calculated position from multiple providers
F.9 Keep introduced measurement errors within accepted margins
F.10 Block RPS commands
F.11 Establish C2 link with RPA
F.12 Decompile C2 software
F.13 Use existing API
F.14 Capture and replay RPS messages
F.15 Detect FHSS pattern
F.16 Configure TX equipment
G.1 Simulate GNSS baseband traffic
G.2 Falsify broadcast satellite data
G.3 Use GNSS signal repeater
G.4 Monitor clock bias over time
G.5 Randomise message IDs
G.6 Implement message authentication mechanism
G.7 Detect continuous wave jamming
G.8 Shield antennas from ground-based transmitters
G.9 Verify received signal's direction of arrival
G.10 Gradually increase signal strength
G.11 Implement multi-constellation GNSS receiver
G.12 Jam genuine operator's signal
G.13 De-authenticate genuine operator
G.14 Pair transmitter and receiver
G.15 Authenticate telemetry messages
H.1 Use SDR based signal simulator
H.2 Use hardware based signal simulator
H.3 Alter date/time
H.4 Alter broadcast ephemerides
H.5 Set satellite health bit to false
H.6 Mimic randomisation algorithm
H.7 Use airborne TX equipment
H.8 Concurrently impersonate multiple GNSS providers
H.9 Randomise message IDs
H.10 Implement message authentication mechanism
I.1 Download broadcast ephemeris data
I.2 Generate I/Q samples
I.3 Alter satellite clock calibration
I.4 Reject jumps in clock signal
I.6 Mimic randomisation algorithm