Cyber-Physical Security of Air Traffic Surveillance Systems

Anusha Thudimilla, Bruce Mcmillin

Chapter 1

CYBER PHYSICAL SECURITY OF AIR TRAFFIC SURVEILLANCE SYSTEMS

Anusha Thudimilla and Bruce McMillin.... thanks... This work was sponsored by a grant from the US National Science Foundation under award number CNS-1837472 and with support from the Missouri S&T Intelligent Systems Center.

Abstract Cyber-Physical security is a significant concern for critical infrastructures. The strong inter-dependency between the cyber and physical components introduces integrity issues such as vulnerability to injecting malicious data and projecting fake sensor measurements. Traditional security models partition the CPS from a security perspective into just two domains: high and low. However, this absolute partition is not adequate to address the challenges in the current CPSs as they are composed of multiple overlapping partitions. Information flow properties are one of the significant classes of methods for cyber-physical security, that model how inputs of a system affect its outputs across the security partition. Information flow supports traceability that helps in detecting vulnerabilities as well as the anomalous source, which helps in rendering mitigation measures. In this chapter, an automated model using graph-based information flow traversal has been proposed to identify information flow paths in the Automatic-Dependent Surveillance and Broadcast (ADS-B) system and partition them into security domains. This approach is applied to selected aspects of the ADS-B system to identify the vulnerabilities in case of a system failure or an attack and provide possible mitigation measures.

Keywords: ADS-B, cyber-physical security, critical infrastructure protection, integrity and privacy

1. Introduction

In recent years there has been a significant increase in the development and deployment of smart and mission-critical computing systems that are characterized by tightly coupled embedded software devices and the physical environment [6]. CPSs are composed of physical components,

computational resources, and communication infrastructure [14]. Hence, ensuring both cyber security and physical security are equally important. New forms of risk have been introduced in the CPSs as a result of the tight coupling between the cyber and the physical components. These risks have not been considered adequately in existing CPSs due to the lack of availability of tools to identify vulnerabilities that arise due to the complex interactions between the cyber and physical worlds. These risks can be classified as (1) cyber elements affecting the physical environment (2) physical elements affecting the cyber components. This chapter addresses these risks by identifying failures and attacks in the system using information flow analysis and proposes possible mitigation measures. Information flow analysis is an important paradigm in cyber-physical security established on a set of tools to determine if the input to a process or a system can change its output. This basic principle provides scope to analyze cyber-physical vulnerabilities to detect attacks and failures in the system.

CPSs are ubiquitous in unmanned aerial vehicles, autonomous cars, smart grids, and data centers. With the evolving changes and the introduction of new systems, security in critical infrastructures such as the aviation industry remains a critical area of research to improve security measures. Avionic systems are composed of various cyber-physical components and one of the system with complex cyber-physical interactions are the air traffic surveillance systems. With a significant increase in the number of flights and passengers traveling around the world, the Federal Aviation Administration (FAA) has introduced NextGen technologies to ensure flight predictability, efficiency, and safety. ADS-B is a powerful CPS that integrates computational intelligence with the physical components. These complex integrations involving a multitude of other systems are controlled by advanced computers to provide reliable and efficient communication between the air traffic control (ATC) and aircraft. ADS-B uses a global positioning system (GPS) to determine the aircraft's speed, position, and altitude. As ADS-B broadcasts such sensitive information over unencrypted channels, there is a necessity for employing the right security tools to ensure flight safety as it provides scope for attackers to launch false-data injection attacks, spoofing, flooding, jamming, message modification, and eavesdropping [9]. The security mechanisms of ADS-B have been shown to be not effective enough to defend against these attacks due to the unauthenticated and unencrypted wireless transmissions between the ground stations and aircraft [9, 11, 18].

Organization of this chapter:

This chapter presents an improved cyber-physical vulnerability analysis using information flows as a means to identify the faulty components in the system and detect attacks. The next two sections discuss the ADS-B system in brief and the current state of the art of vulnerability analysis in the ADS-B system. Section 4 presents the threat model. Section 5 presents our proposed mitigation method for airborne surveillance, followed by a section on the graph-based system used to automate the Multiple Security Domain NonDeducibility (MSDND) analysis. Section 7 presents different scenarios to identify the vulnerabilities in the system. Finally, sections 8 and 9 discuss conclusions and future work.

2. ADS-B

An example of a complex CPS is the air traffic surveillance system. ADS-B is an air-borne surveillance system designed to enable seamless surveillance, increase safety, and provide situational awareness for both ground and air communications. ADS-B has been introduced to improve ATC's situational awareness, collision detection, and avoidance, and replace radar-based surveillance. ADS-B broadcasts aircraft position, velocity, identity, and other information over unencrypted data links once per second. An aircraft equipped with ADS-B collects this data from different sources such as GPS, barometric altimeter, and other nearby aircraft and processes the gathered information to determine accuracy and integrity. The processed information is then encoded and broadcast as an ADS-B message to the nearby aircraft and the ground stations. This process ensures the availability of surveillance with no radar coverage. It also enables pilots to have visuals of other aircraft and receive real-time aeronautical information such as flight restrictions, weather, etc.

An ADS-B equipped aircraft broadcasts aircraft's information in an omnidirectional fashion, which can be received by the ground stations as well as other aircraft with a compatible receiving device. These broadcasts differ from other transponder interrogations such as from the Traffic Collision and Avoidance Systems (TCAS). ADS-B has two functional operations: (1) ADS-B OUT and (2) ADS-B IN. ADS-B OUT is a surveillance technology responsible for generating ADS-B broadcasts, thus enabling real-time data transmission to ATC and nearby aircraft. It provides aircraft's velocity, altitude, and position once per second. ADSB-IN is used to receive transmissions from the nearby aircraft (i.e., ADS-B OUT information). ADS-B IN receives information such as

weather updates, conflict detection and de-confliction guidance, graphical display of the position of aircraft, and along-track guidance [10].

3. Related Work In Security For Avionic Systems

CPSs play an important role in controlling and monitoring various control processes in critical infrastructure systems. Due to the complex nature of CPS interactions, there is a necessity to devise mechanisms to protect against attacks and failures. There is an increase in the analysis of vulnerabilities of CPS resulting from malicious, faulty components and system failures. Various security methods have been proposed to address security issues such as denial of service attacks [2], false data injection attacks [8], stealthy deception attacks [19], and replay attacks [12] in critical infrastructures such as avionics, smart-grids, and autonomous vehicles. Information flow security is yet another widely used approach to prevent secret data from leaking to malicious entities. Two primary variants of information flow security are either static or dynamic. The dynamic approach uses labels to describe the security level and propagates these labels to ensure the integrity of the data with respect to invariants or predefined policies whereas the static approach executes information security policies. A vast majority of the research in this field is focused on proving the non-interference property (describes the information flow restrictions) and using a combination of language features and system models to implement information flow security [16]. There is an extensive amount of research done in the area of aviation security to address the security issues by using information flow paths. The real challenge for information flow security is applying the vast theory and language-based designs such as Jif [13], and FlowCaml [15, 17] to real-world problems [23]. Widely known methods to perform security analysis are formal methods that consider the cyber-physical interactions, to identify potential risks and challenges that arise due to the complex transformations in airplane systems.

A time-stamp based method based on signal propagation time to identify and reject spoofed ADS-B messages between senders and receivers has been proposed in [7]. A light-weight security solution to guarantee the privacy and integrity of ADS-B messages has been proposed by [21] by integrating crypto-primitives such as FFX and TESLA. Similar work has been carried out in [22] for congested data links and resource-constrained avionics. In [20], ProVerif has been used to identify the attacks in ADS-B and TCAS systems. This analysis is limited to proving observational equivalence (anonymity property) through the

composition of processes and lacks computational capability. Several works have been done to analyze the security vulnerabilities, attacks, and system failures in ADS-B systems [9, 11, 18] that consider cyber attacks alone. The research in ADS-B security either focuses on cyber attacks or physical component failures but not both.

This work is aimed at addressing the above-mentioned issues by focusing on cyber and physically enabled attacks by using an automated model-based approach to identify security risks and provide mitigation measures.

4. Threat Model

Building a good threat model is crucial to assess the security vulnerabilities in the system and coming up with efficient mitigation measures. To launch an attack an adversary must compromise one or more components in the system or introduce malicious components into the system. Our threat model is composed of the following:

- *Source:* The source of a threat is the entity responsible for initiating a threat which includes system failure, adversarial attacks, and environmental factors.

- *Goal*: The ultimate goal of the threat model is to capture the features of the system that may lead to system failure or identify the features that are modified as a result of an attack.

- *Consequence*: The attack or failure in a CPS results in compromising integrity, safety, and availability of the system. In this threat model, we assume that attacks and failures to exhibit similar behavior.

In case of an attack, the adversary is assumed to have full control of the faulty components: the adversary can eavesdrop, intercept, and modify any message with respect to the faulty component. In this chapter, we assume failures and attacks are arbitrary and unbounded. Our threat model assumes that the adversary cannot exploit certain aspects of the systems such as:

- The adversary cannot corrupt or modify the proposed model.

- The adversary cannot modify more than half of the participating entities while traversing the graph with respect to a certain feature in our proposed model.

5. MSDND

MSDND was introduced in [4] which is based on modal logic [3] to address the shortcomings of traditional security models that partition the security space into two domains: high and low. These models works well only if the boundaries for the security domains are clearly defined. The MSDND model presented in [4] can check for NonDeducibility (deduce the values of two states in a system) with respect to two states at any single point of time. Calculating the security domains for complex infrastructures is difficult, however, our work is aimed at handling this issue as well as automating the MSDND process by relying on modal logic for partitioning the security domains based on the information flow traversal.

Table 1 presents the list of symbols used in this chapter and their description.

Table 1. Model logic symbols and description

Symbol	Description
φ	A boolean statement that can be evaluated to *true* or *false* in $w \in W$
$w \vdash \varphi$	The statement φ is valid in world w ("yields")
$w \vDash \varphi$	Values from world w cause φ to evaluate to *true* ("models")
$\Box\varphi$	The Modal is always such that ("must be so")
$V^i_{S_x}$	Valuation function of entity i w.r.t S_x

In this chapter, $\{V\}$ is used to define a set of valuation functions, such that $V^i_{S_x}(w)$ indicates the value of state variable S_x as observed by an entity i in world w.

Note: If the state variable, say S_i has no valuation function, then MSDND fails to determine the value associated with the state variable nor the value of any logical expression associated with the state variable [4]. State variables are represented by φ_0, φ_1, ...φ_m or by a combination of state variables represented by S_x.

$$S_x = \varphi_0 \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge ....$$

In this chapter, each state variable is associated with a component ID (See Figure 1). For example, Satellite 1 is represented by φ_0, Barometric Altimeter by φ_6 and control panel by φ_9.

Definition: Multiple Security Domain Exclusivity

There exists some world with multiple states in which at any instance the system can be in one true state and the others are false.

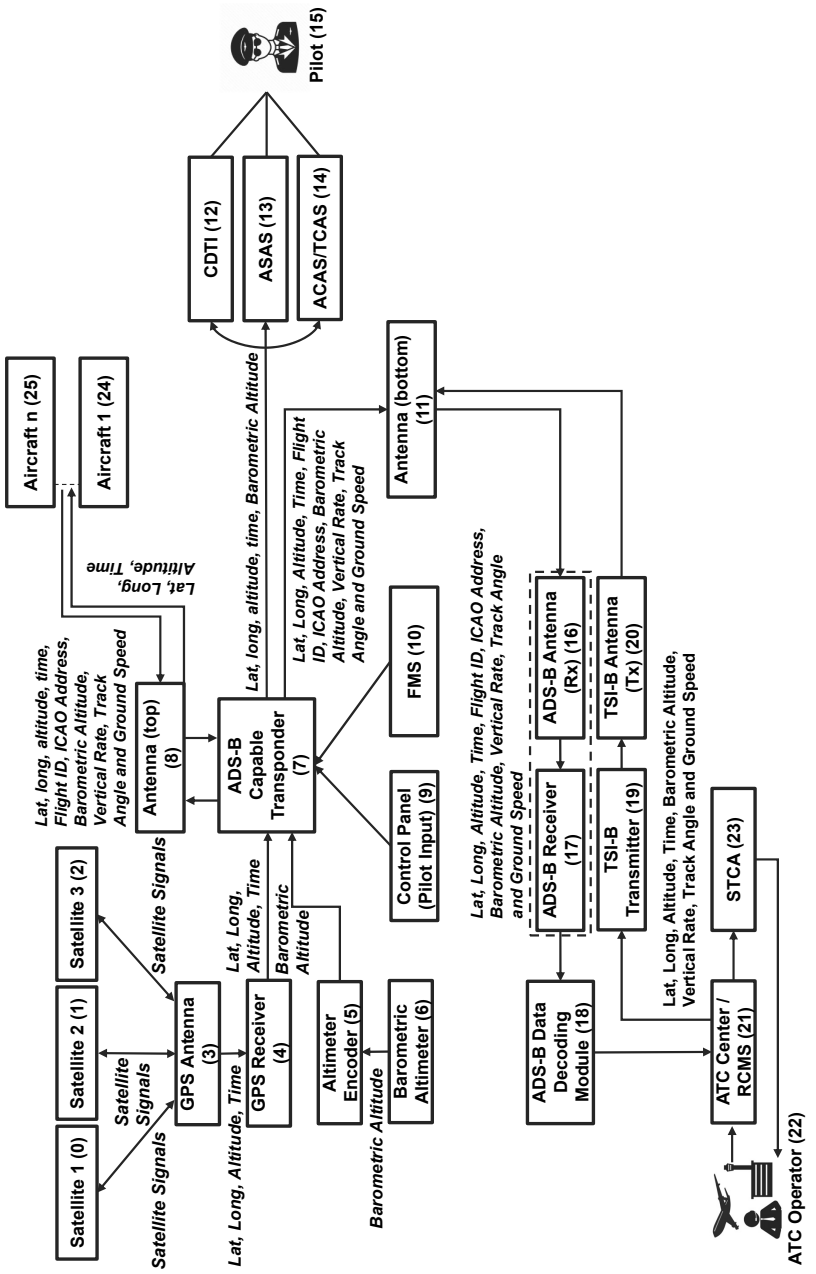

Figure 1. ADS-B Architecture with components represented as nodes and the connections between the components as edges. Every edge is labeled with attributes representing the type of data that is flowing from one component to another [1].

$$f(S_a, S_b, S_c, ...) = \begin{cases} where\ one\ of\ S_a, S_b, S_c, ...\ is\ True \\ otherwise\ False \end{cases} \quad (1)$$

MSDND:

In the MSDND model, an entity i is any part of the system capable of independent observation or action. The Event System, *(ES)*, consists of multiple security domains, SD^i, as viewed by each entity i in the model. These domains may, or may not, overlap depending on the complexity of the *(ES)* [4, 5]. A system is MSDND if

$$\begin{aligned} MSDND(ES) = \exists w \in W : [w \vdash \Box f(S_a, S_b, S_c, ...)] \\ \wedge [w \vDash (\nexists V^i_{S_a} \wedge \nexists V^i_{S_b} \wedge \nexists V^i_{S_c} ...] \end{aligned} \quad (2)$$

$$\cup U_{i \in I} SD^i = (ES) \quad (3)$$

An MSDND proof creates a logical argument of conditions on the observable state of the system under consideration. These conditions are assessed for their valuation from a particular security domain. If no valuation function can be found then the system is MSDND (which is a bad thing as it means an attacker can hide their actions from a particular security domain). Breaking MSDND is a good thing as it means the system can detect the attacker.

6. Graph Based Detection System

The graph-based detection system used in this chapter considers the entire aircraft system as a graph network where each node represents the component in the system and the edges represent the information flow between two nodes. Each edge consists of a set of labels and each label has a value associated with it. This algorithm consists of three steps:

- Identify all the paths in the network using DFS with respect to a label and sort the result set in descending order of subgraph size.
- Identify the subgraphs and eliminate them to get a reduced unique subgraph set.
- In the reduced set, traverse through each edge to check for discrepancies such as inconsistent values associated with a particular label. If there is a discrepancy, find the indegree for each node.
- A node with in-degree ≥ 3 will help in breaking the MSDND, as it contains more than two information flow paths that help in identifying the faulty component or the component under attack during design time and run-time which is sending incorrect data.

A graph $G = (V, E, L)$ consists of a set of nodes V, a set of edges E, and a set of labels L associated with each edge. A graph $S = (V_s, E_s,$

L_s*)* is a subgraph of graph *G = (V, E, L)* iff $V_s \subseteq$ V, $E_s \subseteq$ E and L_s contains l where l is the label under consideration.

Figure 1 represents a single system in an aircraft composed of 26 components referred to as nodes. Each edge has a set of labels and values associated with them which represents the information flow.

Figure 2 represents the ADS-B system in terms of a graph network without edge labels. Each component is identified with the respective node id and the directional edges represent information flow. Refer to Figure 1 for the description of nodes and edge labels.

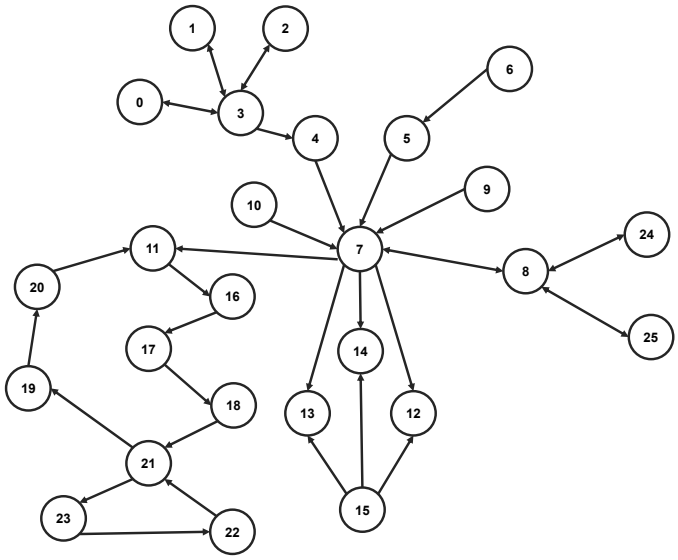

Figure 2. ADS-B Graph

Algorithms 1 and 2 are used to find all the graphs associated with each label and eliminate the subpaths to obtain unique graphs for each label. Source code for these algorithms can be found here: `https://github.com/anushaat/MSDND`

6.1 Finding independent paths:

Algorithm 1 uses DFS to find all the paths with edges labeled with *l*. Theorem 5.1.1 presents the logical arguments that are necessary to prove the correctness of Algorithm 1.

Theorem 5.1.1: In DFS of a directed acyclic graph *G = (V, E, L)*, vertex *s* is a descendant of vertex *d* iff the search discovers *d*, that there is a path from *s* to *d* consisting entirely of edges with label *l*.

Proof: Suppose that DFS is run on a directed acyclic graph $G = (V, E, L)$ to determine the independent paths for each vertex $v_i \in V$. It suffices to show that for any pair of distinct vertices s, $d \in V$, if G contains an edge from s to d. If $s = d$, then the path from s to d contains only s which indicates the source or initial node. If d is an immediate descendant of u, then the path from s to d contains label l. If d is any descendant of s, all edges on the simple path from s to d contains label l.

With a runtime complexity of $\mathcal{O}(V + E)$, where V represents the number of nodes and E represents the number of edges, this algorithm will result in all the subpaths associated with the specified label.

Algorithm 1 Algorithm to find independent paths

Input: Graph dataset G, find all the paths with respect to label l

Output: Result set R which includes all paths associated with each label l

```
1: function DFS(G, u)
2:     visited.add(u)
3:     if onPath.contains(u) then
4:         result.add(Path)
5:     else
6:         for all v ∈ adj[u] do
7:             if v.labels.contains(l) then
8:                 DFS(G, u)
9:     Path.pop()
10:    return result
```

6.2 Eliminate subpaths:

This algorithm is aimed at eliminating the frequent subpaths which occur more than once. This helps in achieving non-redundant which helps in identifying the faulty component or the component under attack. Theorem 5.2.1 provides necessary logical arguments to prove the correctness of algorithm 2.

This algorithm solely operates on the sorted result set obtained from Algorithm 1. Lines 5-7 gets the longest path and stores in k and saves the paths with size k into the final result without any processing. Lines 8-19 uses a variant of sliding window technique to obtain the non-redundant paths and save it to the result. Line 20 returns the non-redundant paths from the list of paths obtained from Algorithm 1. The runtime complexity of this algorithm is $\mathcal{O}(nm)$ where n represents the size of the

Algorithm 2 Algorithm to eliminate the subpaths

Input: Graph dataset G, eliminate frequent subpaths
Output: Set of all graphs associated with each label l

```
1:  function EliminateSubPaths(result)
2:      res ← null
3:      for i ← 0 to N do
4:          S ← result.get(i)
5:          k ← result.get(i).size()
6:          if k = maxSize then
7:              res.add(S)
8:          for all s in res do
9:              count ← 0
10:             N ← s.size()
11:             for i ← 0 to N − k + 1 do
12:                 if s(i) == result.get(i) then
13:                     for j ← 0 to k do
14:                         if s(i + j) == result.get(j) then
15:                             count + +
16:                         else
17:                             break
18:             if count! = k then
19:                 res.add(result.get(i))
20:     return res
```

result set, m represents the size of subpath and k represents the size of the sliding window.

Theorem 5.2.1: Consider any directed acyclic graph *G = (V, E, L)*, and let l_m the label in L based on which the graph traversal is done. The algorithm produces a nonempty subpaths S_p, with edges containing values associated with l_m from result set R_i.

Proof: Let S_p be a maximum-size subset of paths associated with label l_m in R_i. Let $(n_i, ...n_j, ...n_k)$ where $i < j < k$ be the set of nodes connected by edges E_p associated with label l_m in S_p. If the edges connecting $n_i \rightarrow n_j$ and $n_j \rightarrow n_k$ are equal then we are done, $e_{ij} = e_{jk}$, since e_{jk} is a subset of edge set E_p associated with label l_m. If $e_{ij} \neq e_{jk}$, let the edge set $E_p^{'} = E_p - \{e_{jk}\} \cup \{e_{ij}\}$. By substituting e_{ij} for e_{jk} we get $E_p^{'} = E_p$, which shows that e_{ij} and e_{jk} belongs to the same set i.e. having same label. This is true because n_i is a child of n_j and n_k is a child ok n_j. Since $E_p^{'} = E_p$, we conclude that $E_p^{'}$ contains edges with label l_m, and it includes e_{jk}.

7. MSDND Analysis

This section presents an analysis of the vulnerabilities associated with air traffic surveillance systems. To do this, we divided the scenarios into two parts: 1) Identifying the compromised system. 2) Using the graph-based model to identify the information flow paths and use the MSDND model to identifying the faulty paths associated with the ADS-B system. The scenarios presented in this section showcases how MSDND serves as an efficient tool in identifying the vulnerabilities in the system by analyzing the information flow paths. This analysis when performed during design-time provides an in-depth analysis of the various information flow paths present in the system thus, enabling the designers to implement effective security measures in case of a failure or an attack.

7.1 Altimeter Failure

Theorem: *In the case of altimeter compromise, but using automated graph-based analysis, the MSDND model yields deducibility, thereby allowing critical information flow to the pilot and the controller.*

Consider a scenario in which the barometric altimeter is faulty and is sending incorrect altitude data to the pilot. In this case, the altimeter displays incorrect altitude values and thus making them NonDeducible to the pilot.

There are around 10-60 graphs associated with each label. Graphs are generated based on 4 labels namely "bAltitude", "altitude", "mach number", and "airspeed". Figures 3, 4 and 5 represent few graphs generated with respect to the label "altitude". For example, figure 3 presents the information flow w.r.t *altitude* from node 3 ($GPSAntenna$) to node 20 ($TSI-BAntenna$). Once the graphs are generated, a variant of the sliding window technique is used to eliminate the subpaths from the set of graphs. In figure 5, Graph 3 is a subpath of Graph 2. Algorithm 2 aims at eliminating occurrences like Graph 3 from the result set. After eliminating the subpaths, the value associated with each label is evaluated to check for consistency. If there is an inconsistency, the in-degree for each node in the inconsistent set is calculated. If a node has an in-degree ≥ 2 (indicates that there is more than one information flow path carrying similar information), it is considered to have a valuation function and eventually helps in breaking the NonDeducibility. If a valuation function exists for a node, the incoming edges are evaluated to identify the faulty source.

By applying MSDND model yields security domains in this scenario: SD^{BA} {Barometric Altimeter Domain} and SD^{GPS} {GPS Domain}. By combining the valuation functions in SD^{BA} and SD^{GPS} with respect to altitude value from the pilot domain,

$$S_{ba} = \neg\varphi_6 \wedge \neg\varphi_5 \wedge \neg\varphi_7 \wedge \neg\varphi_{15} \Rightarrow \nexists V^P_{\sim a} \tag{4}$$

Since the information received from the barometric altimeter domain is faulty, the pilot cannot valuate the correctness of the altitude data in that domain.

$$S_{gps} = \varphi_3 \wedge \varphi_4 \wedge \varphi_7 \wedge \varphi_{15} \Rightarrow \nexists V^P_a \tag{5}$$

Even though the information received from the GPS domain is not faulty, the pilot cannot valuate the correctness of the altitude data in that domain as they cannot validate the correctness of the data with just two information flow paths.

From Equation 4 and Equation 5, pilot can see two different information flow paths which result in different altitude values.

By combining Equation 4 and Equation 5,

$$\begin{aligned} MSDND(ES) = \exists w \in W : [w \vdash \Box f(S_{ba}, S_{gps})] \\ \wedge [w \vDash (\nexists V^P_{\sim a} \wedge \nexists V^P_a] \end{aligned} \tag{6}$$

Therefore, the pilot cannot deduce that the barometric altimeter is faulty and is sending incorrect altitude data. This situation can be

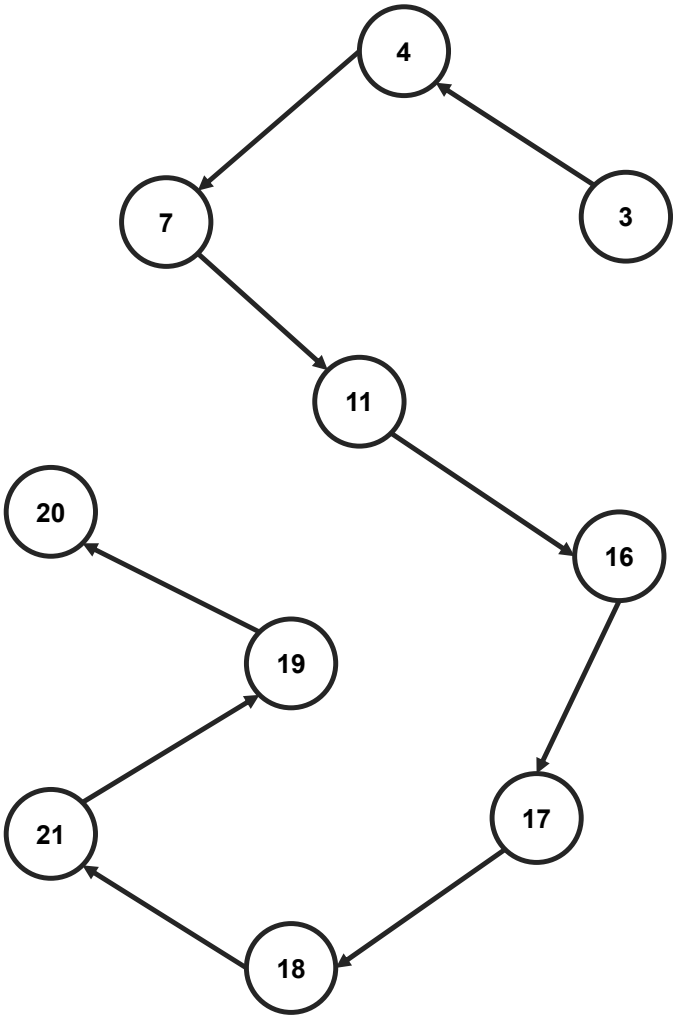

Figure 3. Graph 1

resolved by having an additional information flow path which helps the pilot to resolve the conflict.

$$S_{atc} = \varphi_{21} \wedge \varphi_{19} \wedge \varphi_{20} \wedge \varphi_{11} \wedge \varphi_{7} \wedge \varphi 15 \Rightarrow \exists V_a^P \tag{7}$$

In our system, this additional information flow path is retrieved from the ATC controller which is responsible for sending altitude data.

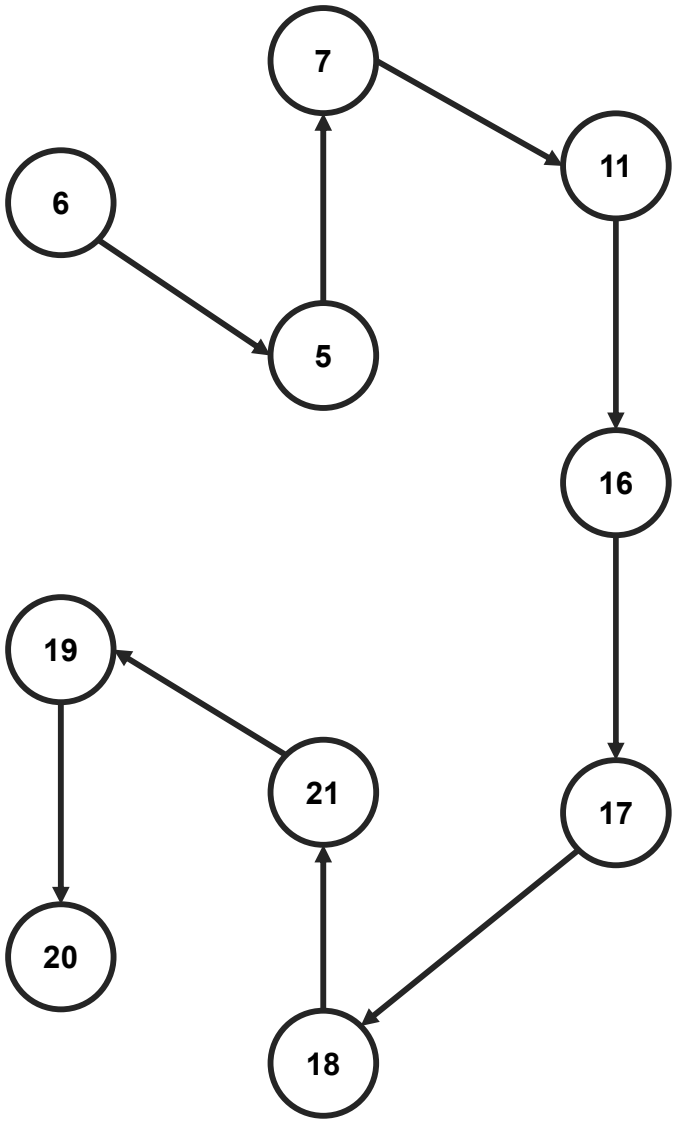

Figure 4. Graph 2

By combining Equation 4, Equation 5 and Equation 7,

$$MSDND(ES) = \exists w \in W : [w \vdash \Box f(S_{ba}, S_{gps}, S_{atc})] \wedge [w \vDash (\nexists V_{\sim a}^{P} \wedge \exists V_{a}^{P} \wedge \exists V_{a}^{P}] \tag{8}$$

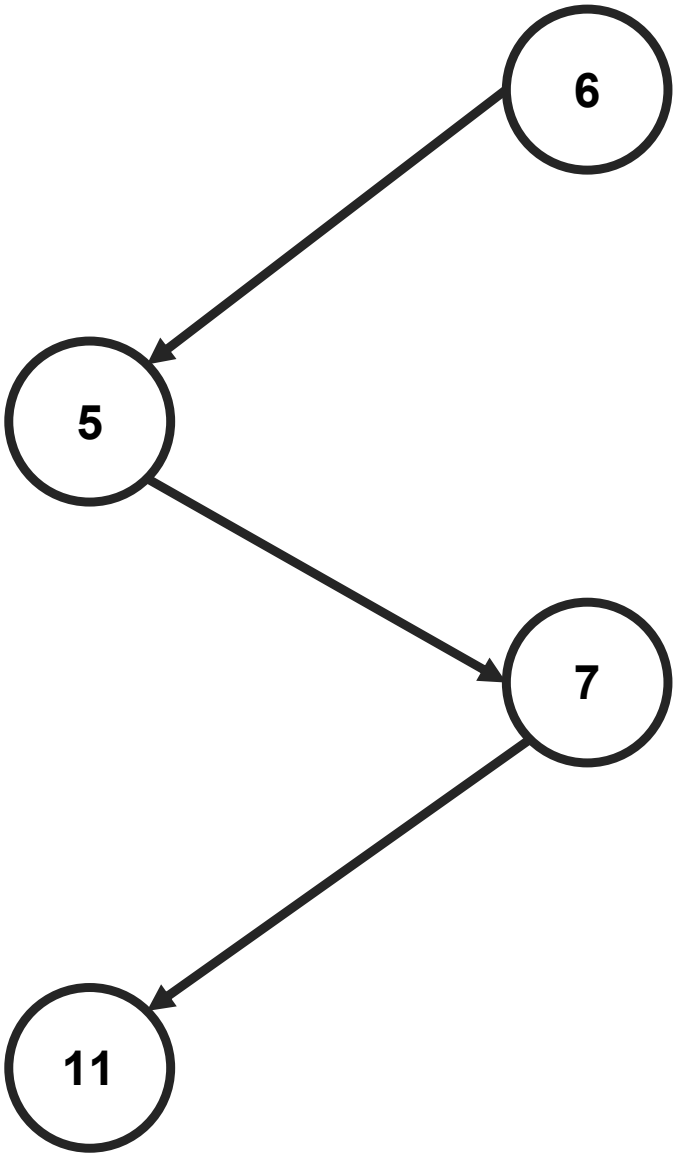

Figure 5. Graph 3

Hence, the system is *not MSDND secure* to pilot as they can deduce the correct altitude value and thereby resolving the conflict by relying on alternate information flow path.

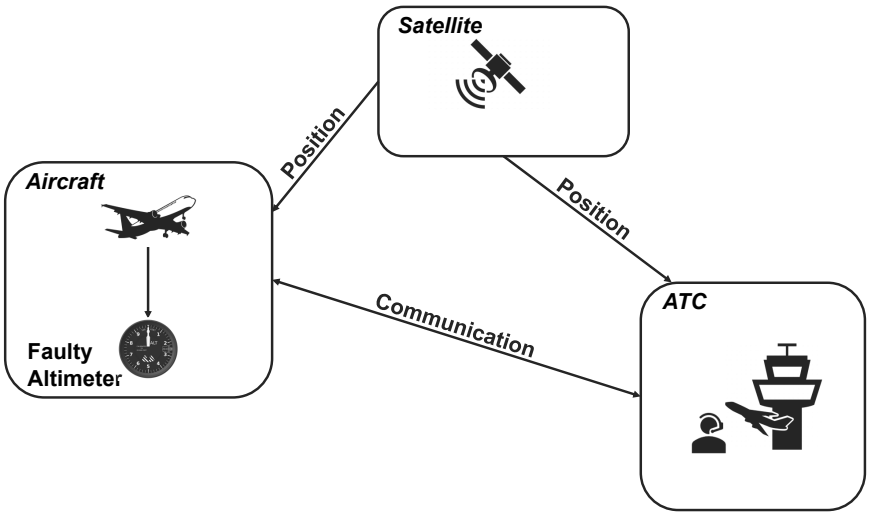

Figure 6. TCAS Helps To Identify The Attacker Plane

7.2 Satellite Failure

Theorem: *In the case of satellite (GNSS) failure, but using automated graph-based analysis the MSDND model yields NonDeducibility, thereby stopping critical information flow to the pilots.*

Proof: Consider a scenario in which the GNSS is faulty and is sending incorrect position data to the aircraft. GNSS is responsible for sending the position data to the planes and the planes communicate with each other. In the case of GNSS failure, the position information retrieved by the aircraft is incorrect and the pilots communicate with each other based on this information. This could lead to potential mid-air collisions and incorrect decisions by the pilots if they couldn't identify the source that is sending incorrect data (See Figure 8).

Once the flight position is retrieved, the pilot trusts the information sent by satellite. If there is another aircraft nearby, both the pilots communicate based on the data transmitted by the satellite. Pilot-1 and pilot-2 cannot identify the problem until they are too close which eventually leads to a breakdown in the separation.

With the automated MSDND analysis, pilots can check for the consistency of information flow paths and find the source of the faulty data.

Figure 7 represents the subgraph generated with respect to the label "pos". Once the graph is generated, a variant of the sliding window technique is used to eliminate the subgraphs from the set of graphs. We apply algorithm 2 to eliminate the subgraphs to avoid redundancy. In

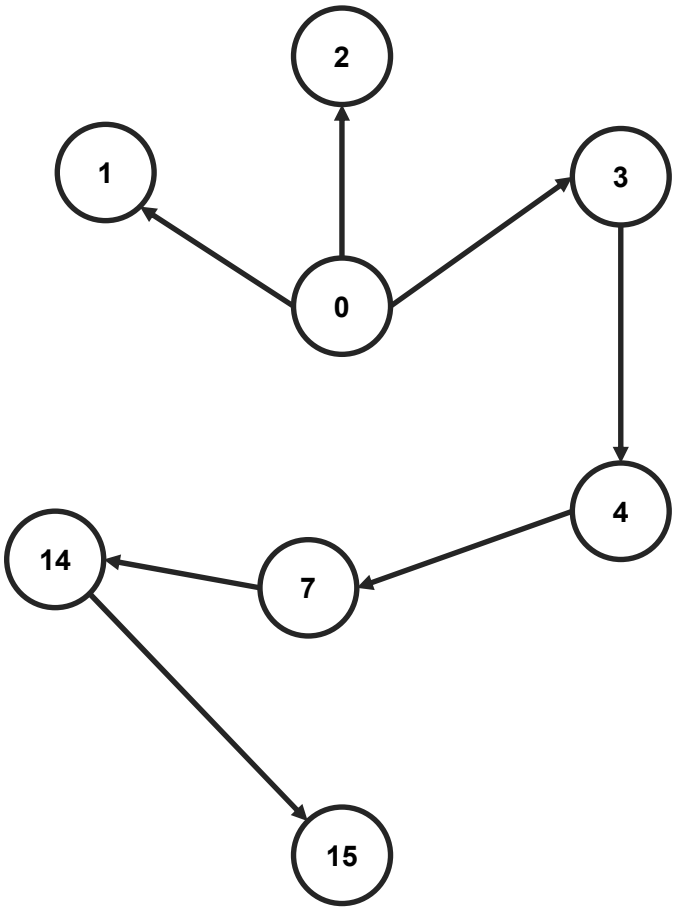

Figure 7. Graph 4

this case, there are no subgraphs and this step doesn't result in a reduced set. After eliminating the subgraphs, the value associated with each label is evaluated to check for consistency. If there is any inconsistency, the in-degree for each node in the inconsistent set is calculated. If a node has an in-degree ≥ 2 (indicates that there is more than one information flow path carrying similar information), it is considered to have a valuation function and eventually helps in breaking the NonDeducibility property. If the valuation function exists for a node, the incoming edges are evaluated to identify the faulty source. In this case, none of the nodes has an in-degree value greater than 2. Therefore, the pilots from both the aircraft cannot deduce that satellite failure is causing the transmission of incorrect information.

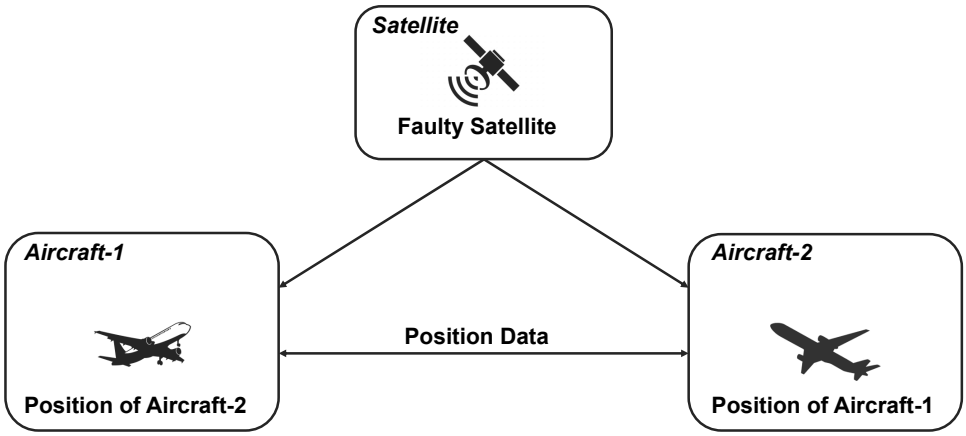

Figure 8. Satellite Failure

Once the flight position is retrieved, pilot-1 trusts the information sent by Plane-2 and vice-versa. Pilot-1 and pilot-2 cannot identify the problem until they are too close which eventually leads to a breakdown in the separation.

The two security domains in this scenario are SD^{P1} {pilot-1 Domain} and SD^{P2} {pilot-2 Domain}. By combining the valuation functions in SD^{P1} and SD^{P2},

$$S_{p1} = \neg\varphi_{a1} \wedge \neg\varphi_{a2} \wedge \neg\varphi_{a3} \wedge \neg\varphi_{a4} \wedge \neg\varphi_{a7} \wedge \neg\varphi_{a15} \Rightarrow \nexists V^{P1}_{\sim pos} \tag{9}$$

Since the information received from the pilot-2 domain is faulty, pilot-1 cannot valuate the correctness of the position data in that domain.

$$S_{p2} = \neg\varphi_{b1} \wedge \neg\varphi_{b2} \wedge \neg\varphi_{b3} \wedge \neg\varphi_{b4} \wedge \neg\varphi_{b7} \wedge \neg\varphi_{b15} \Rightarrow \nexists V^{P2}_{\sim pos} \tag{10}$$

Since the information received from the pilot-1 domain is faulty, pilot-2 cannot valuate for the correctness of the position data in that domain.

By combining Equation 9 and Equation 10,

$$\begin{aligned} MSDND(ES) = \exists w \in W : [w \vdash \Box f(S_{p1}, S_{p2})] \\ \wedge [w \vDash (\nexists V^{P1}_{\sim pos} \wedge \nexists V^{P2}_{\sim pos}] \end{aligned} \tag{11}$$

Therefore, the pilots from both the aircraft cannot deduce that satellite failure is causing the transmission of incorrect information.

Hence, the system is *MSDND secure* to the pilots as they cannot deduce the actual true position of the planes.

7.3 Summary

In this section, we presented two scenarios describing how MSDND can be used to identify the vulnerabilities in the system in case of an attack or failure. We performed this analysis on the various interacting components of the ADS-B system in an aircraft. This analysis, when applied to the entire system, helps the pilots and the manufacturers to identify the vulnerabilities in real-time and design-time respectively.

8. Conclusion

Despite the extensive research in CPS security, there is a need to identify and mitigate the attacks in CPSs. In this chapter, an automated MSDND model is designed and implemented. MSDND is applied to various attack scenarios targeting a specific component failure in an aircraft. MSDND works by identifying independent information flow paths and partitioning them into different security domains based on the consistency of the information flow. The model proposed is suitable for detecting attacks and failures in a CPS with complex state transitions.

9. Future Work

In moving forward, the focus would be on three principal areas. The first is to automatically define the optimal number of security domains for each attack scenario. This is done by clustering the components in the system based on the validation of the information flow paths (the information flow paths qualified as secure will be in *secure* security domain and the other information flow paths will be in $non{-}secure$ security domain). Each of these security domains will have additional security partitions based on the label associated with each information flow path. The second is to extend the MSDND analysis to other infrastructures and monitor if all the attack scenarios are identified. The third is to use MSDND to model confidentiality, integrity and availability vulnerabilities. Future work also includes extending the MSDND model to all the critical infrastructures to identify and evaluate cyber-physical risks.

References

[1] B. S. Ali, W. Ochieng, A. Majumdar, W. Schuster and T. K. Chiew, ADS-B system failure modes and models, *The Journal of Navigation*, vol. 67(6), pp. 995–1017, 2014.

[2] S. Amin, A. A. Cárdenas and S. S. Sastry, Safe and secure networked control systems under denial-of-service attacks, in *Hybrid Systems:*

Computation and Control, R. Majumdar and P. Tabuada (Eds.), Springer, Berlin, Heidelberg, pp. 31–45, 2009.

[3] J. Garson, Modal Logic, The Stanford Encyclopedia of Philosophy Archive, The Metaphysics Research Lab, Center for the Study of Language and Information, Stanford University, Stanford, CA (`plato.stanford.edu/archives/fall2018/entries/logic-modal`), 2018.

[4] G. Howser and B. McMillin, A multiple security domain model of a drive-by-wire system, *Proceedings of the IEEE Thirty-Seventh Annual Computer Software and Applications Conference*, pp. 369–374, 2013.

[5] G. Howser and B. McMillin, Using information-flow methods to analyze the security of cyber-physical systems, *Computer*, vol. 50(4), pp. 17–26, 2017.

[6] A. Humayed, J. Lin, F. Li and B. Luo, Cyber-physical systems security - A survey, *IEEE Internet of Things Journal*, vol. 4(6), pp. 1802–1831, 2017.

[7] Y. Kim, J. Y. Jo and S. Lee, ADS-B vulnerabilities and a security solution with a timestamp, *IEEE Aerospace and Electronic Systems Magazine*, vol. 32(11), pp. 52–61, 2017.

[8] Y. Liu, P. Ning and M. K. Reiter, False data injection attacks against state estimation in electric power grids, *ACM Transactions on Information and System Security*, vol. 14(1), pp. 1–33, 2011.

[9] M.R. Manesh and N. Kaabouch, Analysis of vulnerabilities, attacks, countermeasures and overall risk of the Automatic Dependent Surveillance-Broadcast (ADS-B) system, *International Journal of Critical Infrastructure Protection*, vol. 19, pp. 16–31, 2017.

[10] D. McCallie, Exploring Potential ADS-B Vulnerabilities in the FAA's NextGen Air Transportation System, Graduate Research Project, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2011.

[11] D. McCallie, J. Butts and R. Mills, Security analysis of the ADS-B implementation in the next generation air transportation system, *International Journal of Critical Infrastructure Protection*, vol. 4(2), pp. 78–87, 2011.

[12] Y. Mo and B. Sinopoli, Secure control against replay attacks, *Proceedings of the Forty-Seventh Annual Allerton Conference on Communication, Control, and Computing*, pp. 911–918, 2009.

[13] A. C. Myers, L. Zheng, S. Zdancewic, S. Chong and N. Nystrom, Jif: Java information flow, Cornell University, Ithaca, NY (`www.cs.cornell.edu/jif/`), 2001.

[14] F. Pasqualetti, F. Dörfler and F. Bullo, Cyber-physical security via geometric control: Distributed monitoring and malicious attacks, *Proceedings of the Fifty-First IEEE Conference on Decision and Control*, pp. 3418–3425, 2012.

[15] F. Pottier and V. Simonet, Information flow inference for ML, *Proceedings of the Twenty-Ninth ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 319–330, 2002.

[16] A. Sabelfeld and A. C. Myers, Language-based information-flow security, *IEEE Journal on Selected Areas in Communications*, vol. 21(1), pp. 5–19, 2003.

[17] V. Simonet, Flow Caml in a nutshell, *Proceedings of the First Applied Semantics II Workshop*, pp. 152–165, 2003.

[18] M. Strohmeier, M. Schäfer, V. Lenders and I. Martinovic, Realities and challenges of nextgen air traffic management: the case of ADS-B, *IEEE Communications Magazine*, vol. 52(5), pp. 111–118, 2014.

[19] A. Teixeira, S. Amin, H. Sandberg, K. H. Johansson and S. S. Sastry, Cyber security analysis of state estimators in electric power systems, *Proceedings of the Forty-Ninth IEEE Conference on Decision and Control*, pp. 5991–5998, 2010.

[20] A. Thudimilla and B. McMillin, Multiple security domain nondeducibility air traffic surveillance systems, *Proceedings of the Eighteenth International Symposium on High Assurance Systems Engineering*, pp. 136–139, 2017.

[21] H. Yang, M. Yao, Z. Xu and B. Liu, LHCSAS: A lightweight and highly-compatible solution for ADS-B security, *Proceedings of the IEEE Global Communications Conference*, 2017.

[22] H. Yang, Q. Zhou, M. Yao, R. Lu, H. Li and X. Zhang, A practical and compatible cryptographic solution to ADS-B security, *IEEE Internet of Things Journal*, vol. 6(2), pp. 3322–3334, 2018.

[23] S. Zdancewic, Challenges for information-flow security, *Proceedings of the First International Workshop on the Programming Language Interference and Dependence*, 2004.