Demo Abstract: Analysing CPS Security with Falsification on the Microsoft Flight Simulator

Tanmay Khandait
tkhandai@asu.edu
School of Computing and Augmented Intelligence, Arizona State University
Tempe, Arizona, USA

Aniruddh Chandratre
achand75@asu.edu
School of Computing and Augmented Intelligence, Arizona State University
Tempe, Arizona, USA

Walstan Baptista
wbaptist@asu.edu
School of Computing and Augmented Intelligence, Arizona State University
Tempe, Arizona, USA

Giulia Pedrielli
gpedriel@asu.edu
School of Computing and Augmented Intelligence, Arizona State University
Tempe, Arizona, USA

Georgios Fainekos
Toyota NA R&D
Ann Arbor, Michigan, USA
fainekos@acm.org

ABSTRACT

In the paper titled " Stealthy attacks formalized as STL formulas for Falsification of CPS Security", we investigate a broad class of attacks on the sensor and actuation blocks in the form of additive perturbation that impacts the measurement and control, respectively. In this demo, we demonstrate the usage of our framework and the underlying technologies along with a case study on aviation systems using Microsoft Flight Simulator (MSFS).

CCS CONCEPTS

• **Security and privacy**; • **Theory of computation**; • **Computing methodologies**;

KEYWORDS

CPS Security, Falsification, Signal Temporal Logic, Test Generation

ACM Reference Format:
Tanmay Khandait, Aniruddh Chandratre, Walstan Baptista, Giulia Pedrielli, and Georgios Fainekos. 2023. Demo Abstract: Analysing CPS Security with Falsification on the Microsoft Flight Simulator. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control (HSCC '23), May 09–12, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3575870.3589550

1 INTRODUCTION

Safety-critical Cyber-Physical Systems (CPS) are susceptible to malicious security related attacks. In [1], we proposed a CPS security framework to generate attacks on the sensors and/or actuators in the form of additive perturbations that can be physical (e.g., achieved by an agent changing the sensor measurements), or cyber (e.g., achieved by taking over the bus and changing packets).

A successful attack is one that deviates the system from its intended behaviour while being undetected by the anomaly detection systems. In this demo, we present the user interface of our framework, and demonstrate its use on the Microsoft Flight Simulator.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
HSCC '23, May 09–12, 2023, San Antonio, TX, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0033-0/23/05.
https://doi.org/10.1145/3575870.3589550

2 ATTACK GENERATION IN PSY-TALIRO

Within PSY-TaLiRo [2], attacks are iteratively generated by a search algorithm (**Optimizer** in Fig.1). Each attack is executed (simulated) and the resulting dynamical traces are collected (data flow associated with the **Black-box model** in Fig. 1). The quality of the attack is then quantified based on a desired security property formulated using a specification tool, and evaluated using the trajectories and the specification monitor (information flow associated with the **Security Specifications** box in Fig.1). The search algorithm uses the information on the quality of the attack to continue the generation process until stopping condition is met (defined as part of the **PSY-TaLiRo options**, Fig. 1). We detail the interface to the key PSY-TaLiRo components [2] in Fig. 1.

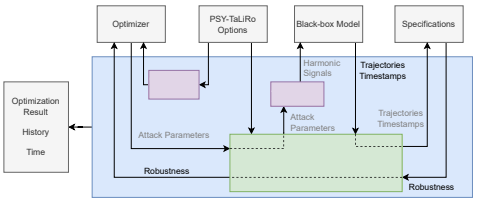

Figure 1: Architecture Diagram of PSY-TaLiRo in the context of the security analysis framework.

(1) **Black-box Model**: The System Under Test (SUT) is given as black-box model, a function that receives the interpolated input signal times and values corresponding to the attack parameters selected by the Optimizer, returning the trajectories with timestamps of the simulated model. In this demo, the Black-box returns two signals, where `sig_1` refers to the probability of detection of an anomaly, while `sig_2` refers to the probability of occurrence of a state violation [1].

```
def uav_model(static, times, signals) -> ModelData[]:
    trajectories = np.concatenate((sig_1, sig_2))
    return ModelData(trajectories.T, timestamps)
```

Listing 1: Template for defining a black-box function

(2) **Security Specifications**: The security specifications are formulated using Signal Temporal Logic (STL). Listing 2 (line 1) encodes the Instantaneous Attack specification. In Line 2 the `STLMonitor` receives as input the specification [1], along with a dictionary to map the variables in the specification to the indices in the trajectories from the black-box.

```
1 phi = "not((G[0,t]sig_1<=eps)and(F[0,t]sig_2>=gam))")
2 spec = STLMonitor(phi, {"sig_1": 0, "sig_2": 1})
```

Listing 2: Defining the Instantaneous Attack in STL

(3) **Attack Vector Generation**: Our attacks are harmonic signal parameterized by bias (β), amplitude (A), frequency (f), and phase shift (ϕ). In Listing 3 (Lines 1-2), the bounds for β, A, f, ϕ are provided as list of pairs specifying each the upper and lower bound for the corresponding parameter. Line 3 shows the use of control points to encode the signals,which are passed as `SignalOptions` in line 3 of Listing 4.

```
1 control_pt=[(-0.03, 0.03),(-0.06, 0.06),
2                 (0.0, 150.0),(0.0, pi)]
3 signals = [SignalOptions(control_points=control_pt,
      factory=delayed(signal_factory = harmonic)]
```

Listing 3: Defining Harmonic Attack Vector

(4) **Optimizer**: We generate a successful attack falsifying the specification by running a Bayesian optimization (BO) algorithm as detailed in [1] over the hyper-rectangle defined by the bounds of the parameters associated to the harmonic attack signal.

(5) **PSY-TaLiRo Options**: The PSY-TaLiRo *options* allow to specify the experiment configuration parameters. Finally, all the components defined above are passed to *staliro* function (Listing 4 - line 3) to run the search for the SUT vulnerabilities.

```
1 options = Options(runs=100, iterations=300,
2     interval=(0, t), signals=signals)
3 result = staliro(uav_model, spec, optimizer, options)
```

Listing 4: Tying all the components together and running the experiments

3 DEMO WITH MICROSOFT FLIGHT SIMULATOR (MSFS)

Aviation systems are a prime example of safety-critical CPS, which may be vulnerable to security attacks by malicious adversaries. Models for avionics that represent sensors and control systems with satisfactory realism are hard to obtain for research use. The MSFS offers a high fidelity simulator, thus making it effective to evaluate various system-level attacks once the adversary has gained access to the system. It provides programmatic access to write values to control the aircraft (navigation systems, GPS location, Autopilot (AP), Fly-by-Wire (FBW) systems, actuator systems, etc.) as well as read the aircraft data (sensor values, etc.) using the SimConnect-SDK.

3.1 Experimental Setting

The A320-Neo aircraft flies from Phoenix to Newark, and is set to reach the cruising altitude level of 30,000 feet before the AP takes over. Attacks to the actuator system (specifically the aileron and the elevator) are defined using a black-box function, which returns the AP status along with the flight position and orientation. When the AP is turned on, an attack is generated. If the aircraft enters Alpha Protection (when the angle of attack gets too high), or similar scenarios, the internal systems detects and disengages the AP, returning control back to the pilot. Every attack lasts for 4 seconds or until the AP is disengaged. Thus, a successful stealthy attack deviates the aircraft from its flight plan without alarming the AP. After each simulation, the flight path is restored, and a new attack is initiated until a successful stealthy attack is found. Fig. 3 is a screenshot from one of the successful stealthy attacks.

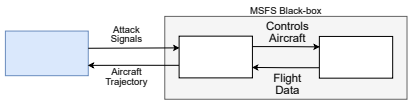

Figure 2: SimConnect-SDK Communicates with the MSFS to control the aircraft and read values back from it.

On the top, the left pane shows the deviation from the flight plan on a Visual Flight Rules (VFR) map, while the right pane shows the external view of the flight. On the bottom, the left pane shows the position and orientation on the Navigation Display and Primary Flight Displays, while the right pane shows the attack in progress. Notice that the flight has deviated from the flight-plan, while the AP is still active.

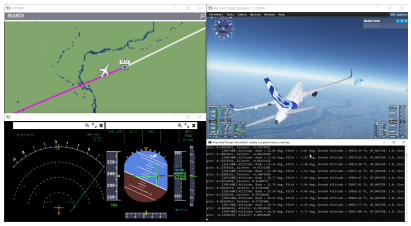

Figure 3: A screenshot from a successful stealthy attack.

4 CONCLUSION

In this preliminary work, we apply the security framework to the aviation systems using simple attacks on the elevator and aileron. As the next step, we plan to develop complex attack scenarios involving GPS and sensor spoofing, and the Instrument Landing System.

ACKNOWLEDGMENTS

This work is supported by the DARPA ARCOS program under contract FA8750-20-C-0507.

REFERENCES

[1] Aniruddh Chandratre, Tomas Acosta, Tanmay Khandait, Giulia Pedrielli, and Georgios Fainekos. 2023. Stealthy attacks formalized as STL formulas for Falsification of CPS Security. In *26th ACM International Conference on Hybrid Systems: Computation and Control* (San Antonio, TX) *(HSCC '23)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3575870.3587122

[2] Quinn Thibeault, Jacob Anderson, Aniruddh Chandratre, Giulia Pedrielli, and Georgios Fainekos. 2021. PSY-TaLiRo: A Python Toolbox for Search-Based Test Generation for Cyber-Physical Systems. In *Formal Methods for Industrial Critical Systems*, Alberto Lluch Lafuente and Anastasia Mavridou (Eds.). Springer International Publishing, Cham, 223–231.