Stealthy Attacker Complexity in Cyber-Physical Systems

https://c-er.github.io/15400/

Uday Shankar <us@cmu.edu>

1 Description

Who: I will be working under Eunsuk Kang, assistant professor in the Institute for Software Research at Carnegie Mellon University.

What: I will be studying the relative complexity between supervisors and stealthy attackers in a cyber-physical system. This will involve defining an appropriate notion of complexity that makes sense for both attackers and supervisors and then considering questions such as

- How complex does an attacker have to be, as a function of the supervisor's complexity, to guarantee a stealthy attack?
- How complex does an attacker have to be, as a function of the supervisor's complexity, to make a stealthy attack possible in some cases?
- How complex does a supervisor have to be, as a function of the attacker's complexity, to prevent a stealthy attack?
- How complex does a supervisor have to be, as a function of the attacker's complexity, to make a stealthy attack preventable in some cases?

So What: Nowadays, cyber-physical systems are found in all sorts of safety-critical applications, such as in automotive vehicles. Consequently, attacks on cyber-physical systems are an important concern for system designers. Upon detection of an attack, it is often a viable option for a safety-critical system to simply shut itself down, thus preventing any harm. However, such detection is by definition impossible for stealthy attacks, which makes them particularly frightening and especially important to understand. My work will enable cyber-physical system designers to obtain security guarantees on their systems. By measuring the complexity of their systems and applying the bounds I will derive, cyber-physical system designers will be able to obtain lower bounds on the complexity of any stealthy attacker, which may be useful in ruling out threats from certain classes of actors.

2 Goals

75%: Come up with a set of small examples that motivates the need to study relative complexity. This might involve coming up with a system and supervisor for which all attackers that can guarantee a successful stealthy attack are "complex" in some intuitive sense. Then, extrapolate a formal definition of complexity that makes sense for both the attacker and the supervisor and that agrees with the examples.

100%: Investigate the consequences of this formal definition. Try to reason about lower bounds for complexity. This is where I will attempt to provide a proper answer to the questions posed in the "what" section above.

125%: Write a small tool that implements the ideas developed in the previous steps. More specifically, the tool should accept as input a supervisor or an attacker, compute its complexity and bounds on the complexity of its adversary, and maybe attempt to synthesize an optimal adversary.

150%: Package all the findings into a paper and submit it to a conference.

3 Milestones

End of 15-300: Learn the necessary background material to get started on the research. This will involve a careful reading of the book and the papers mentioned in the "literature search" section listed below. Then, look at some of the examples of systems given in those papers and see if any of them are suitable to meet the 75% goal.

January 27th: Complete analysis of the small examples given in the "literature search" section. Consider the attacker synthesized by the approach given in *Stealthy Deception Attacks for Cyber-Physical Systems* for each of the supervisors in the examples and see if it is "complex" in an intuitive sense.

February 10th: Come up with additional examples if the ones considered from the book and the papers are not sufficient. Generate a list of potential measures of complexity that make sense for both the attacker and the supervisor.

February 24th: Make a table that evaluates the complexity of the attacker and supervisor under each of the potential complexity measures. Select one that is compatible with the examples and seems to admit proofs of lower bounds. Write code to compute complexity according to this measure.

March 16th: Come up with proofs of some easy (probably not tight) lower bounds. Based on the examples, guess some possible tighter lower bounds and formally state them.

March 30th: Have several attempts at a proof of the tighter lower bounds.

April 13th: Formally write up any proofs of lower bounds that we have.

April 27th: Clean up the code written during the process so it is usable by someone else. It should at least compute the complexity of a given supervisor/attacker and compute the lower bounds that we proved. A stretch goal would be to attempt synthesis of an optimal adversary.

4 Literature Search

As of now, I plan to refer to the following papers and books during my research.

- *Stealthy Deception Attacks for Cyber-Physical Systems* by Góes, Kang, Kwong, and Lafortune. This paper is the background for my research as it provides a lot of the necessary formalism and an algorithm to synthesize stealthy attackers.
- *A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems* by Yin and Lafortune. This paper develops the *bipartite transition structure*, which is a formalism that captures the game played between a supervisor and an attacker. This formalism is extended in the above paper, but I found it difficult to understand, so reading this may help.
- *Introduction to Discrete Event Systems* by Cassandras and Lafortune. Since I haven't taken a class in cyber-physical systems, I will use this book to fill in any background knowledge that I cannot pick up just by reading the papers.

I currently have access to both papers and the book.

5 Resources Needed

Since I will be writing code that manipulates cyber-physical systems and associated structures, I will find frameworks built for this purpose useful. A couple of frameworks we have identified so far include

- DESUMA, a "tool used to build, analyze and control models of Discrete Event Systems (DES) as finite-state automata (FSA)."
- Supremica, a "tool for formal verification and synthesis of discrete event control functions based on discrete event models of the uncontrolled plant and specifications of the desired closed-loop behavior."

Both tools are available online, free of charge. For the small examples that I will be working with for most of the project, my laptop will be sufficient. But if I reach some of the stretch goals, I might need access to more powerful hardware to synthesize larger attackers/supervisors. We plan to cross that bridge if and when we come to it.