Designing Secure and Resilient Cyber-Physical Systems Using Formal Models

1st Robert S. Lois II
Department of Mechanical Engineering
University of Pittsburgh
Pittsburgh, United States
robert.lois@pitt.edu

2nd Daniel G. Cole
Department of Mechanical Engineering
University of Pittsburgh
Pittsburgh, United States
dgcole@pitt.edu

***Abstract*—This work-in-progress paper proposes a design methodology that addresses the complexity and heterogeneity of cyber-physical systems (CPS) while simultaneously proving resilient control logic and security properties. The design methodology involves a formal methods-based approach by translating the complex control logic and security properties of a water flow CPS into timed automata. Timed automata are a formal model that describes system behaviors and properties using mathematics-based logic languages with precision. Due to the semantics that are used in developing the formal models, verification techniques, such as theorem proving and model checking, are used to mathematically prove the specifications and security properties of the CPS. This work-in-progress paper aims to highlight the need for formalizing plant models by creating a timed automata of the physical portions of the water flow CPS. Extending the time automata with control logic, network security, and privacy control processes is investigated. The final model will be formally verified to prove the design specifications of the water flow CPS to ensure efficacy and security.**

***Index Terms*—cyber-physical systems, formal methods, safety-critical systems**

I. Introduction

Recent technological advancements are providing a catalyst for the Fourth Industrial Revolution, Industry 4.0, by integrating the internet of things (IoT), artificial intelligence, and smart automation to physical systems. This next evolution is engineered from the integration of computational embedded systems and physical components defining them as a cyber-physical system (CPS) [1]. Examples of CPSs include today's automobiles, medical devices, power generation and distribution systems, building control systems, and robots. Many CPSs operate in safety-critical or mission-critical settings, and therefore, it is important to gain assurance CPSs operate correctly and are protected against the unauthorised exploitation of its system from cyber attacks.

However, designing CPSs with high levels of assurance for proper operation and cybersecurity properties is difficult. This is due to their use in complex environments — like smart power grids, fly-by-wire aircraft, and modern vehicles — and the need for intelligent decision-making — like in manufacturing processes to ensure zero downtime or autonomous vehicles. The complex environments includes competing control logic and connected networks, which increases the attack surface causing CPS to become more vulnerable to cyber attacks. These vulnerabilities can allow an adversary to cause disruptions with catastrophic consequences, especially when those CPS are used in safety-critical or mission-critical settings. The severity of these attacks are evident in the recent Colonial Pipeline, where operation of one of the nation's largest carriers of gasoline and jet fuel was forced to shutdown [2]. Additionally, these vulnerabilities were highlighted when the joint Cybersecurity Advisory (CSA) warned of persistent threat actors (APT) developed custom-made tools to gain full system access to CPS devices used in industrial control environments [3]. There is a critical need to verify safety-critical control properties while simultaneously securing vulnerabilities that may lead to cyber attacks. This would prevent adversaries from disrupting normal operations that negatively impacting the CPS environment and maintain the efficacy of safety-critical operations.

The main challenge when designing CPSs is how to verify safety-critical control properties that are resilient while simultaneously securing vulnerabilities that could lead to cyber-attacks. One method for doing this is to take a formal method-based approach to establish mathematically provable system properties. The formal method-based approach translates system properties to a mathematical formal model using sets and sequences that can then be used to verify and analyze system processes. The formal model is precise, unambiguous, analyzable, and can be processed by computer programs to prove properties of the model. When the formal model is verified, then it follows that the system properties guarantee proper operation with high levels of security and interoperability. As such, the mathematically proven formal models instill confidence that the overall system will not violate the properties and future versions of the system will retain the properties.

II. Cyber-Physical Systems

A cyber-physical system (CPS) is a system that integrates computational processes with physical components using sensors and network technologies. The intersection of the physical part (mechanical, chemical, electrical, biological) with computer hardware and software, advanced actuators, and smart technologies allow the physical part to perceive its changes and respond to them. In this sense, the CPS contains embedded

computers that enable users to monitor and control the physical processes through feedback mechanisms or automated control over network applications.

There are several key characteristics that define CPS systems [4]. The first distinctive characteristic is embedded and mobile sensing. This includes sensors and data exchange flows between embedded systems, which ultimately enables the interaction of cyber- and physical components. Another key characteristic is the ability for the CPS to train and adapt to its environment under its predetermined specifications. If performed correctly, then automatic control can ensure reliable and proper operation. Moreover, the CPS ensures system robustness through intelligent automated control. Finally, the CPS also contains a common cyberspace to exchange information between other systems and its environment, typically this interoperability is performed through the Internet. The cyberspace includes information security such as firewalls, anti-virus, or cryptosystems.

A. Cyber-Physical System Architecture

As mentioned previously, CPS are the intersection of the physical and the cyber/computations. The CPS architecture depicting this intersection can be divided into several fundamental layers but for the purpose of this research, the architecture is simplified to three layers: (1) physical layer, (2) network layer, and (3) application layer [5].

The physical layer includes the main physical infrastructure with its associated sensors and actuators. In this layer, the sensors and actuators are connected via wireless sensor or wired networks where network layer protocols are used for communication between the physical and network layer. The devices on this layer have little memory or processing power.

At the network layer, communication networks and protocols are used to deliver and receive data from the physical layer. The network and protocols depend on the CPS design, which usually requires a heterogeneity of networks. For example, the CPS design may include wireless sensor networks (WSN), internet protocol suites (TCP/IP), or Modbus protocols for transmitting information between the physical layer and the network layer. The heterogeneous networks are interconnected through gateways. Here, the packets of data are coordinated through dispatchers or relayed through routers.

The application layer aggregates the CPS environment with their functionalities and applications. It has the ability to store information in databases, monitor and make control decisions that can be visualized through human machine interfaces (HMI), all through multi-agent systems.

B. Current Vulnerabilities and Attacks on Cyber-Physical Systems

As a whole, the integration of different devices, networks, data, protocols in a heterogeneous CPS environment increases the vulnerabilities due to the increase attack space. The growing demand for connecting CPS in a complex environment has led to the use of open standards and network protocols. Since isolation is not an option for protection and CPS will have external communication, the use of standard protocols in design can be exploited by malicious attackers that can compromise CPS. Fig. 1 shows a diagram of threats and attacks that can lead a CPS to fail depending on where in the CPS architecture an attack was successful. For example, a trojan or virus leading to a software malfunction in the application layer that using computing tools can ultimately lead to a compromised CPS since the communication to the physical layer is no longer available. This was the case when a successful attack on the application layer of the Colonial Pipeline occurred, leading to the shutdown of the physical components.

Fig. 1. A tree diagram for showing how a cyber-physical system failure can occur due to a successful software malfunction attack in the communication environment [6].

C. Current Limitations in Modeling and Design Practices

As CPS environments become more complex, so do the requirements and specifications that are used to interpret the desired control behavior and security properties. These requirements and specifications may be expressed in high-level languages, distributed across multiple artifacts, which limits the modeling and design process. One of the limits of using high-level languages for modeling and design practices is that they cannot be used for mathematical analysis and verification. This is due to a lack of an authoritative source of truth for the requirements and specifications, which would enable true design specifications with requirements that can be readily analyzed.

Additionally, the requirements and specifications that are finalized are typically implemented with a human-in-the-loop effort. This effort is performed in a discretized fashion where

the widely accepted V-model approach is used [7]. The V-shape process splits the modeling and design into segmented human-in-the-loop phases. Since these phases for design were developed in isolation, they have to be integrated to test functionality and behavior, which further propagates errors and time spent debugging unverified algorithms, source code, and models. Employing this method when designing safety-critical CPS can lead to fatal design flaws as was the case when Uber's autonomous vehicle application killed a pedestrian due to sensor anomalies [8].

III. Formal Methods and Models

Traditional design processes in complex CPS environments can lead to implementations that fail to satisfy the required system properties yielding inefficient design cycles, cost overruns, and delays [9]. Additionally, the complexity of CPS leads to an increase in malicious attacks since the attack space and vulnerabilities grow with heterogeneous environments. To overcome these limitations, system design properties and security protocols are translated to languages that are suitable for formal mathematical analysis and verification through formal methods-based approaches.

Formal methods-based approaches offer a family of models and languages to provide a concise way of describing a system's behavior and desired properties as mathematical entities. The formalized properties are verified by either model checking or theorem proving techniques. Both model checking and theorem proving require mathematical entities that are formal definitions of rigorously defined syntax and semantics that can ultimately reduce ambiguity and imprecise descriptions about the modelled system. Model checking uses the state-space dynamics of the system and formally verifies the specifications that are specified in the formal language of temporal logic. Theorem proving involves developing a mathematical model of a given system and using computer-based methods to formally prove or mathematically reason that the properties of interest are complete and accurate. Ultimately, by proving the mathematically rigorous model through model checking or theorem proving, it is possible to verify the system's properties and desired behavior in a more thorough fashion than empirical testing.

Formal methods-based approaches are typically achieved following a three step process:

1) *Formal Specification:* Rigorous definitions of the system under investigation is undertaken using a formal modeling language. The language contains fixed grammars that represent the complex structures of the model, which helps define the overall problem, goal, and solution. The resulting model from the formal specification step is the desired system expressed in formal mathematical logic. Examples of mathematical models include finite-state machines, transition systems, and contract-based systems.
2) *Verification:* In this step, a formal proof on the formal specified model is performed. The formal proof either proves or disproves the system's intended behavior, which is typically achieved through software tools. An example of verification of a finite-state model is the method of model checking. In this method, an exhaustive exploration of the model is performed by checking all states and transitions given the formal specification. If the exploration of the model finds that the system does not satisfy a desired property, then a counter-example is provided.
3) *Implementation:* Once the model has been specified and verified, the formal model and its specifications are converted into code. The resulting code, as formally proved, contains the same behavior as the specification thus completing the implementation step.

Using the approach outlined above, the safety-critical functions and security protocols of CPS environments can be formally modeled. This includes the functional assessment from abstract models in each layer of the CPS architecture. Each layer, along with their control logic and security protocols, can adopt the methodology to be translated into a formal model. With the formal model, the desired interactions and behaviors of the CPS are mathematically verified to ensure proper functionality and security through model checking and theorem proving.

A. Applications of Formal Methods and Models

Formal methods and models are used in a variety of different fields and applications including security protocols, hardware and software development, financial computing, and industrial processes [10].

Possibly the most successful application of formal methods-based approaches is the development of the seL4 microkernel; it is open to the public for use on GitHub [11]. The seL4 microkernel is an open source, high-assurance, high-performance operating system microkernel that contains the world's most comprehensive proofs of correctness and security. This was achieved by formal, machine-checked verification, providing mathematical proof that the kernel implementation is consistent with its specification and free from programmer-induced implementation defects.

Those who engineered the seL4 microkernel set out to provide a verified operating system that would be useful in real world applications. In order to meet their goals, the main challenge was to integrate the skill sets of operating system knowledge with the rigorous techniques that are used in formal verification. The seL4 design was constructed in a manner to minimize the proof complexity without compromising performance. An iterative design approach was used by the engineers to develop the kernel where a prototype was coded in Haskell and the formal design was formulated from the mathematical orientation of the Haskell prototype. Simulations were made from the formal design and if the kernel was not functioning properly, then the Haskell prototype was adjusted. Once the design was complete, the Haskell prototype was translated into C code, which is the typical implementation method for operating systems. From the final formal design, proof

methods can be applied to produce the formal specifications [12].

The formally verified seL4 microkernel was supported by the Defense Advanced Research Projects Agency (DARPA) and used for the control of a Boeing Unmanned Little Bird autonomous helicopter. DARPA invited a red team of hackers to perform cyber attacks on the system. In the end, the red team failed to successfully penetrate the system demonstrating that the CPS (helicopter) was unhackable [13]. The application of formal methods and models to develop an operating system can perform its functions while having strong cybersecurity is evidence that the formal methods approach can be adapted to CPSs.

Applications of formal methods and models include the design of robotics CPS. This application formally validated the safety requirements and control functions for a robot system using the UPPAAL tool [14]. UPPAAL is a model checking tool that is employed to ensure consistent behavioral output of the designed control algorithm [15]. This application sought out to formally verify the specifications regarding the control of a manipulator and to formally prove that the user of the robot can take control if and only if the end-effector is adjusted to a specified angle. Consistency of required specifications for the robot system model was done in an iterative method between a model level and a verification level.

At the model level, the robot system is modeled in the UPPAAL tool using timed automata, which are finite state machines that describe the transition between system states in a directed graph structure. The formal structure of a timed automata contains several components defined using set theory that describe the states of the timed automata, the actions that the timed automata can take along with how states transition, and clock variables that the timed automata is constrained to. The robot system is formalized as a network of timed automata that describe the interactions between the components using labeled transitions of systems $\mathcal{S}(\mathcal{A}) = \langle S, s_0, \rightarrow \rangle$, where S is a set of states, s_0 is the initial state, and $\rightarrow$ is the set of relations that can be formalized into clauses for verification in UPPAAL.

At the verification level, the timed automata resembling the robot system model is formally verified using the UPPAAL model checking tool. Simulation of the time automata model occurs at this level to observe the system behavior and to verify that the behavior is correct according to the specifications. The correctness of the time automata model is ensured through computational tree logic (CTL), which generates a formal structure of the system that can branch to different states as time progresses. If a counter-example is found when attempting to prove correctness of the specifications then the time automata is reassessed in the model level.

Other formal methods-based approaches include designing and developing secure cryptographic protocols [16]. This approach highlighted the critical need for a developing a crytographic protocol that an adversary could not exploit by creating a language that would aid in protocol development. The language was designed so that the developer did not have to consider how an adversary may attack their protocol. Instead, the approach sought to formally verify the language using the Coq theorem prover [17] within a framework built upon the Strong Preservation Theorem. The resulting framework was called the Secure Protocols Implemented CorrectlY (SPICY), which guarantees that any protocol in the SPICY language is safe in an environment without an adversary while also being safe in an environment with an active adversary. To this end, the SPICY language enabled developers the ability to create secure cryptographic protocols, guaranteeing that reasoning about how an adversary may attack was no longer required since the language itself was formally verified.

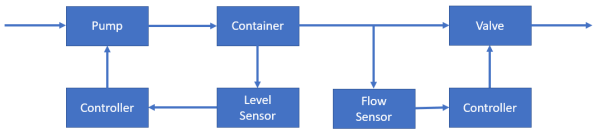

Fig. 2. A high-level diagram for the how the water flow control system will be designed.

IV. Proposed Methodology and Future Work

This work-in-progress paper proposes future work of formalizing the system properties for a water flow cyber-physical system, where the properties include both functional control logic and security protocols. The physical system is a water container, shown in Fig. 2, with an inlet pipe at the top and an outlet pipe at the bottom along with a water level sensor. At the outlet pipe, there will be a valve equipped with a flow sensor followed by pump where piping extends itself back to the inlet valve of the container.

The control and network fabric for the water flow system will follow the three layers of a CPS. The physical layer that is to be controlled is the pump and valve at the outlet of the container. At this layer, there will be a flow sensor on the valve and a water level sensor for the container. In the network layer, the control process will be dictated by an arduino that communicates with the physical layer. The application layer that communicates to the network layer will be performed by the University of Pittsburgh's Cyber Range [18]. The Pitt Cyber Range is a sandbox environment with virtual machines that will simulate the application layer where cyber-attacks will be performed.

All control logic processes and security protocols between layers will be formally modelled and analyzed as shown in cyber-physical system shown in Fig. 3. The system properties will be verified using the appropriate software that is available for the formal methods-based approach that is chosen. Thereafter, the formal model will be translated to the appropriate language to be deployed on a CPS.

A. Linear Temporal Logic and Automata Formal Models for Water Flow Cyber-Physical System

This work-in-progress paper will investigate linear temporal logic (LTL) formulas, which enable formal formulas that can

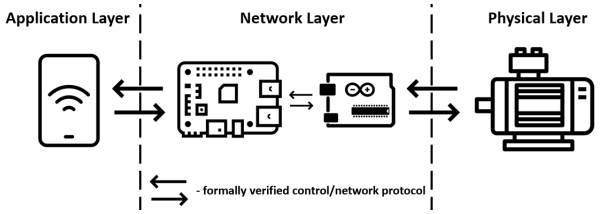

Fig. 3. Cyber-physical system architecture of a water flow system with formally verified control and security properties.

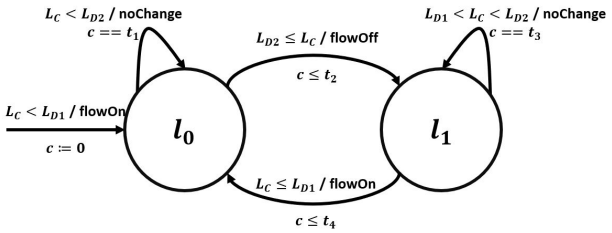

Fig. 4. Timed automaton for the pump in the water flow control system.

express the desired control logic and security properties for the water flow control system. Such formulas can capture safety properties, liveliness, and more complex combinations of Boolean and temporal statements. LTL uses standard Boolean operators with temporal operators $\circ$ to denote 'next' and U to denote 'until'. The syntax of LTL formulas ϕ over a given set of observations O is defined as:

$$\phi = \top | o | \phi_1 \wedge \phi_2 | \neg\phi | \circ\phi | \phi_1 U \phi_2 \tag{1}$$

where $\top$ is constant Boolean true, $o \in O$ is an observation and ϕ, ϕ_1, and ϕ_2 are LTL formulas. LTL contains temporal operators to construct more complication expressions:

$$\begin{aligned} \diamond\phi &:= \top U \phi \\ \Box\phi &:= \neg \diamond \neg\phi \end{aligned} \tag{2}$$

where $\diamond$ denotes the temporal operator 'eventually' and $\Box$ denotes the temporal operator 'always'. LTL semantics and satisfaction of formula over a set of observations can be found in Belta et al. [19].

Applying LTL to the water flow control system, assume that the flow into the container is defined by B and that the container level is defined by G. As the container fills, then the control system must avoid overfilling, which is defined by D. This simple task can be represented as the following LTL formula:

$$\phi = \Box \diamond G \wedge \Box \diamond B \wedge \Box \neg D. \tag{3}$$

This work-in-progress paper will investigate LTL applications for CPS and will attempt to integrate LTL formulas in formal models such as timed automata.

Timed automata will be used as the formal modelling methodology for both control logic and security control purposes. Timed automata are mathematical models of computation whose abstraction can be described using LTL formulas. As an example, the control logic for the pump that fills the container is modeled with the finite state automaton shown in Fig. 4. The container level, L_C, is assumed to be below the desired setpoint level, L_D, which enables the action 'flowOn' of the initial state, s_0, to start the pump and reset the clock variable $c := 0$. The desired setpoint level has a deadzone $L_{D1} < L_d < L_{D2}$. The pump will remain in the l_0 state while $L_C < L_{D2}$ at some clock time t_1. Once the level of the container is greater than or equal to the desired state, $L_{D2} \leq L_C$, the action 'flowOff' turns the pump off transitioning the pump to the l_1 state at some clock time t_2.

TABLE I
SECURITY AND PRIVACY CONTROL POLICIES WITH THEIR ASSOCIATED IDENTIFIER.

Control Identifier	Control Name
SC-24	Fail in Known State
SI-10	Information Input Validation
SI-15	Information Output Filtering

The pump will remain in this state as long as the level of the container is greater than the desired level. If the level of the container goes below the desired setpoint then the action 'flowOn' is enabled and the system transitions back to the l_0 state at some clock time t_4.

The timed automaton model will be amended to include security and privacy controls found in NIST Special Publication 800-53, Revision 5 [20]. This publication provides guidance on how to strengthen the underlying information systems, component products, and services used in critical infrastructure through a family of controls. The controls have unique protection capabilities and objectives according to the family that the control belongs to such as access control, system and information integrity, and systems and communications protection. Each family contains individual security and privacy control actions that can be carried out by information systems. Table I lists several control names that this proposed work plans to formally model along with their identifier that points to their control family.

The security and privacy controls are selected from the System and Communication Protection (SC) family and the System and Information Integrity (SI) family. The SC-24 Fail in Known State guideline forces a component to fail in a specified state where this state can be crucial for safety-critical applications. This control protocol will be simulated to demonstrate security concerns in the event that an adversary has caused a failure in the water flow system. The SI-10 Information Input Validation checks the validity of information inputs for a defined system. This work will implement this security control in order to enforce acceptable inputs for fields in the application layer of the water flow system. This will prevent injection attacks from adversaries while maintaining correct and accurate inputs from authorized personnel. The SI-15 Information Output Filtering validates output information from defined applications. For this work, the output data displayed in the application layer will be filtered to detect

anomalous behavior that may result from SQL injections.

Finally, this work-in-progress paper will define quantitative resilient properties for cybersecurity and physical process control. For cybersecurity, future work will investigate the minimum effort an adversary will need to find an exploitation in the system and to reach an undesirable effect. This involves a measurement of the effort in the formalized model and in an informal model, which will serve as a baseline. For the control of the physical process, future work will adopt control theory techniques to measure the time for the physical process to return to normal operation when forced into an undesirable state.

B. Model Checking Formalized Water Flow Cyber-Physical Systems

Model checkers take a formal model along with its specification as an input and determines if the model behavior satisfies the given specification. The formalized water flow CPS and its specifications will be verified using the UPPAAL and the PRISM model checker [21]. The UPPAAL model checker is a sequential model checker tool whereas the PRISM model checker uses probabilistic methods to verify finite state machines and their specified properties. Both model checkers will be used to verify the water flow CPS and compared in terms of their ability to verify the proposed system using different semantics to describe the system.

V. Summary

This work-in-progress paper proposes a methodology for developing and securing cyber-physical systems using timed automata formal models and linear temporal logic. Future work seeks to investigate using linear temporal logic with timed automata as a formalizing methodology. Designing the water flow control system using linear temporal logic (LTL) provides a mathematical, proof-based approach to overcome complex system properties and security protocols. The LTL language is extended through state-transition diagrams to fully model the water flow control CPS using timed automata. By translating the control and communication behaviors of a water flow control CPS to a formal timed automata model, then the control logic and security properties can be mathematically verified to ensure proper system behavior and a cyber secure environment. The formal model is precise and unambiguous, formally analyzable, and can be readily processed by computer programs to mathematically prove the model as was demonstrated by the seL4 microkernel and robot control examples. To this end, formal models can be used in the design of complex CPS environments in order to verify the desired control logic specifications and security properties.

Acknowledgment

This article was authored and/or co-authored by employees of Battelle Energy Alliance, LLC (BEA). BEA is the Management and Operating Contractor of the Idaho National Laboratory with the United States Department of Energy (DOE), under Contract Number DE-AC07-05ID14517 (BEA's DOE Prime Contract) with principal offices located at 2525 North Fremont Avenue, P.O. Box 1625, Idaho Falls, ID 83415-3899. The United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish and reproduce the published form of this article, or allow others to do so, for United States Government purposes.

References

[1] J. Jamaludin and J. M. Rohani, "Cyber-Physical System (CPS): State of the Art," 2018 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube), 2018, pp. 1-5.

[2] Sanger, David, "Cyberattack Forces a Shutdown of a Top U.S. Pipeline," New York Times, https://www.nytimes.com/2021/05/08/us/politics/cyberattack-colonial-pipeline.html, accessed 4 April 2022.

[3] National Cyber Awareness System, "APT Cyber Tools Targeting ICS/SCADA Devices," Cybersecurity and Infrastructure Security Agency (CISA), https://www.cisa.gov/uscert/ncas/alerts/aa22-103a, accessed 15 May 2022.

[4] Hanh, Adam, "A multi-layered and kill-chain based security analysis framework for cyber-physical systems," International Journal of Critical Infrastructure Protection, Volume 11, December 2015.

[5] Januário, Fábio, Alberto Cardoso, and Paulo Gil. "A distributed multi-agent framework for resilience enhancement in cyber-physical systems." IEEE Access 7: 31342-31357, 2019.

[6] Alguliyev, Rasim, Yadigar Imamverdiyev, and Lyudmila Sukhostat. "Cyber-physical systems and their security issues." Computers in Industry 100: 212-223, 2018.

[7] Sangiovanni-Vincentelli, Alberto, Werner Damm, and Roberto Passerone. "Taming Dr. Frankenstein: Contract-based design for cyber-physical systems." European journal of control 18.3: 217-238, 2012.

[8] Shepardson, David, "In review of fatal Arizona crash, U.S. agency says Uber software had flaws," Reuters, https://www.reuters.com/article/us-uber-crash-idUSKBN1XF2HA, accessed 15 May 2022.

[9] Nuzzo, Pierluigi, et al. "Contract-based design of control protocols for safety-critical cyber-physical systems." 2014 Design, Automation & Test in Europe Conference & Exhibition. IEEE, 2014.

[10] Kulik, Tomas, et al. "A Survey of Practical Formal Methods for Security." Formal Aspects of Computing, 2021.

[11] seL4 Project, "The seL4 microkernel operating system," github, https://github.com/seL4/seL4, accessed 22 April 2022.

[12] Cofer, Darren, "Secure mathematically-assured composition of control models," Airforce Research Laboratory, Rockwell Collins Cedar Rapids United States, 2017.

[13] Waterman, Shaun, "DARPA Drone Cybersecurity Software Foils Hackers in Demo," Airforce Magazine, https://www.airforcemag.com/def-con-hackers-foiled-by-darpa-formal-methods/, accessed 4 April 2022.

[14] Wang, Rui, et al. "A formal model-based design method for robotic systems." IEEE Systems Journal 13.1: 1096-1107, 2018.

[15] UPPAAL Model Checker, "UPPAAL Model Checker," UPPAAL, https://uppaal.org/, accessed 25 April 2022.

[16] Braje, Timothy M., et al. "Adversary safety by construction in a language of cryptographic protocols," unpublished.

[17] Coq Theorem Prover, "The Coq Proof Assistant," Coq, https://coq.inria.fr/, accessed 25 April 2022.

[18] University of Pittsburgh Cyber Range, "Pitt Cyber Range," University of Pittsburgh, https://www.planforpitt.pitt.edu/projects/pitt-cyber-range, accessed 29 May 2022.

[19] Kloetzer, Marius, and Calin Belta. "A fully automated framework for control of linear systems from temporal logic specifications." IEEE Transactions on Automatic Control 53.1: 287-297, 2008.

[20] Joint Task Force Transformation Initiative (2013) Security and Privacy Controls for Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Rev. 5, Includes updates as of December 10, 2020. https://doi.org/10.6028/NIST.SP.800- 53r4

[21] PRISM Model Checker, "PRISM - Probabilistic Symbolic Model Checker," PRISM, https://www.prismmodelchecker.org/, accessed 29 May 2022.