Explainable AI for Prioritizing and Deploying Defenses for Cyber-Physical System Resiliency

Indrajit Ray, Sarath Sreedharan, Rakesh Podder, Shadaab Kawnain Bashir and Indrakshi Ray
Computer Science Department, Colorado State University, Fort Collins, Colorado, USA
{Indrajit.Ray, Sarath.Sreedharan, Rakesh.Podder, Shadaab.Bashir, Indrakshi.Ray}@colostate.edu

***Abstract*—The adoption of digital technology in industrial control systems (ICS) enables improved control over operation, ease of system diagnostics and reduction in cost of maintenance of cyber physical systems (CPS). However, digital systems expose CPS to cyber-attacks. The problem is grave since these cyber-attacks can lead to cascading failures affecting safety in CPS. Unfortunately, the relationship between safety events and cyber-attacks in ICS is ill-understood and how cyber-attacks can lead to cascading failures affecting safety. Consequently, CPS operators are ill-prepared to handle cyber-attacks on their systems. In this work, we envision adopting *Explainable AI* to assist CPS operators in analyzing how a cyber-attack can trigger safety events in CPS and then interactively determining potential approaches to mitigate those threats. We outline the design of a formal framework, which is based on the notion of *transition systems*, and the associated toolsets for this purpose. The transition system is represented as an *AI Planning* problem and adopts the *causal formalism* of human reasoning to asssit CPS operators in their analyses. We discuss some of the research challenges that need to be addressed to bring this vision to fruition.**

***Index Terms*—cyber physical systems, resiliency, AI planning, natural language processing**

I. Introduction

The adoption of digital technology in industrial control systems (ICS) enables improved control over operation, ease of system diagnostics, and reduction in cost of maintenance of cyber physical systems (CPS). However, digital systems also expose the OT (Operational Technology) networks of CPS to cyber-attacks. The SQL Slammer Worm attack on the David-Besse nuclear power plant in 2003 [1], the StuxNet worm targeting Siemens Step 7 software controlling a programmable logic controller [2] or the Black Energy malware exploiting control system software vulnerability in different ICS [3], are all examples of cyber-attacks on CPS triggered by the use of digital systems. Such attacks can have very serious implications on a CPS's operation especially when the cyber-attacks trigger safety events either directly or via triggering cascading failures in the ICS. The emergence of Advanced Persistence Threats (APT) that can lie under the radar undetected for a very long time and Live Off the Land Binaries (LOLBins), where the attacker uses tools that are already present in the environment compounds the problem since they help facilitate lateral movements by attackers that can give rise to cascading failures. There is thus a critical need for operators of CPS OT networks to be cyber-prepared for resiliency. The first step towards this goal is to assess the security posture of the OT networks, what the weak spots are that can be leveraged by APTs, LOLBins, or other malware to launch cyber attacks affecting safety, and how to best deploy defenses for resiliency.

One potential approach to this is via Penetration Testing (or pen-testing). Pen-testing is a widely adopted paradigm for cyber preparedness evaluation in an IT (Information Technology) network. Pen-testing, when done well, results in a comprehensive list of vulnerabilities in the organizational network that can potentially be exploited by an attacker including, possibly, physical vulnerabilities that can enable the launching of cyber-attacks. However, it is a very involved process with a very broad scope and partly depends on the rules of engagement with the organization. While several aspects of pen-testing can be automated, the effectiveness of pen-testing depends very much on the skill set of the pen-tester and on the tools used. While pen-testing can produce a prioritized list of vulnerabilities ordered by their criticality, differing perspectives of pen-testers on what are the highest priorities, can result in scope creep. Moreover, pen-testing does not provide a list of actionable items which would allow what-if analysis to be conducted. The system administrator has very little to work with to determine the effectiveness of various defensive strategies, following a pen-test on their network. Most importantly, however, pen-testing cannot be exercised on the live OT network since it can potentially trigger safety events. Unlike IT networks, OT networks are deterministic and not designed to accommodate the performance impacts of IT network appliances. This concern becomes more exacerbated when considering low-level ICS controller buses where any effect on determinism within this bus triggered by pen-testing may cause undesirable effects on the operation of the CPS.

A complementary approach to understanding and mitigating network vulnerabilities is via attack tree/attack graph analysis (see, for example, [4]–[11]). Its origins are in fault tree analysis [12]–[15]. The ICS community has extensively studied potential problems caused by safety events occurring because of component failures and accidental human errors. ICS operations manual frequently provides documentation of potential safety situations and trains operators to perform what-if analysis to handle safety events. Fault Tree Analysis (FTA), a technique developed by H. Watson and Allison Mearns of Bell Labs for use on the Minute Man Guidance System in 1962 [12], is one of the most widely used systematic approaches to determine all credible ways by which an undesirable failure state may occur in a safety critical ICS. FTA and related techniques such as Event Tree Analysis can help to identify critical components, procedures, tasks in ICS, and combinations of system failures resulting in safety events. FTA has also been extended for evaluating the frequency/probability of undesired events occurring that can be utilized for quantitative safety and reliability analysis. In the same vein, an attack graph, also known as a threat graph, exploitability graph, or vulnerability graph, uses a description of IT assets of an organization, their configuration, the vulnerabilities present in those assets and dependencies among those vulnerabilities to present a picture of how the IT system can be attacked. Graph analysis techniques can be used on the attack graph to identify possible attack paths that can be utilized to launch cyber attacks, including lateral movement attacks. It provides security professionals with valuable insights into potential attack vectors and can help prioritize efforts to secure critical assets effectively.

However, there are several challenges related to the use of attack graphs for cyber preparedness analysis in OT networks. Unlike fault trees, there is no uniform formalism for attack graphs. Different researchers model the abstract notion of an attack graph in different manners resulting in differing properties. To the best of our knowledge, there does not exist a formal attack graph framework that can also model undesirable fault and failure states in OT networks. Moreover, even today, attack graph tool support for automated creation, management and what-if analysis is quite limited. In particular, what-if analysis tools that provide explanatory feedback about the goodness of defensive strategies are needed. Explanations are critical so that the domain experts are convinced that the strategies suggested by the tool are aligned well with their intuitions about potential solutions. To achieve this, the human operators need to be able to relate easily to the machine-generated explanations, which implies that explanation generators need to adopt the *causal formalism used in human reasoning*.

Some other limitations of existing tools include scalability to large problems and support for model reusability. As a system changes, for example, when defense placement is strategized during a what-if analysis, the corresponding attack graph also changes. Support for incremental updates and reuse of attack graphs facilitate scalability.

In this paper, we present an *Explainable AI* approach that combines the power of Natural Language Processing (NLP) and AI Planning to enable CPS operators to evaluate and analyze how a cyber-attack can trigger safety events in the CPS (that is, the *resiliency posture of the CPS*) and then interact with the analysis engine to determine potential approaches to mitigate the threats. The formal framework, on which this system is based, is based on the notion of transition systems [16]. We call this framework Explaianble Resiliency Graph (ERG). *The core design philosophy that we have adopted for this work, and which has resulted in the choices that we have made, is that the framework needs to provide human understandable explanations on how or why it arrived at a specific analysis result for CPS resiliency.* This is a work in progress and we discuss some of the research challeges that we are working on this vision.

Figure 1 provides an example of an ERG for a steam flow control system in a nuclear power plant. An ERG is built through a composition of attack graphs representing cyber-attacks in the CPS and fault trees representing failures. NLP techniques are used to generate both the attack graph portion of the ERG as well as the fault tree. Referring to Figure 1 the red dotted lines indicate the points where composition operations need to be applied to build the ERG. AI planning techniques are used to perform this composition.

We develop a novel description language that we call Resiliency Graph Description Language (RGDL) to represent the ERG-transition system for formal analysis. RGDL is an extension of the classical Planning Domain Definition Language (PDDL) [17] and leverages all its power for solving AI planning problems. The reasoning engine is based on AI Planning (which adopts the causal formalism of human reasoning for deductions). It identifies plans that shows the various attack paths in the attack graph part of the CPS that can eventually lead to cascading failures and safety issues (following the red dotted lines in Figure 1.

We present sketches of an initial proof-of-concept that allows the operator to interact with the AI planner to query the underlying transition system and perform what-if analysis. This analysis provides actionable suggestions from the tool including insights into potential attack vectors and help prioritize efforts to secure critical assets effectively. These suggestions comprise of a diverse set of solutions each of which can potentially take the CPS to a safe and secure state. To help the operator decide which of the actionable suggestions to implement, the toolset provides explanations in natural language. Since the CPS can evolve over time (for example, when new vulnerabilities are identified or new components added) we provide support for incremental updates to the ERG. Figure 2 gives an overview of the ERG workflow.

II. Overview of Approach

Attack-resilience for complex CPS involves direct action to be taken based upon a suspected event, which requires a mitigation response that includes at a minimum, *a semi-autonomous capability*. Such semi-autonomous action needs to provide context and explanation to human operators that a cyber-attack is occurring that necessitates the needed action outside the capability of the ICS as well as assurances that operational impacts are minimized. Moreover, the human operators need to be able to relate easily to the machine-generated explanations. This implies that explanation generator need to adopt the causal formalism used in human reasoning. Our work is driven by these requirements of semi-autonomous capabilities.

ERG-based CPS resiliency analysis can be broadly divided into four stages discussed below and elaborated upon in the next section, which also discusses the open research challenges involved in each step that need to be addressed to bring this vision to fruition.

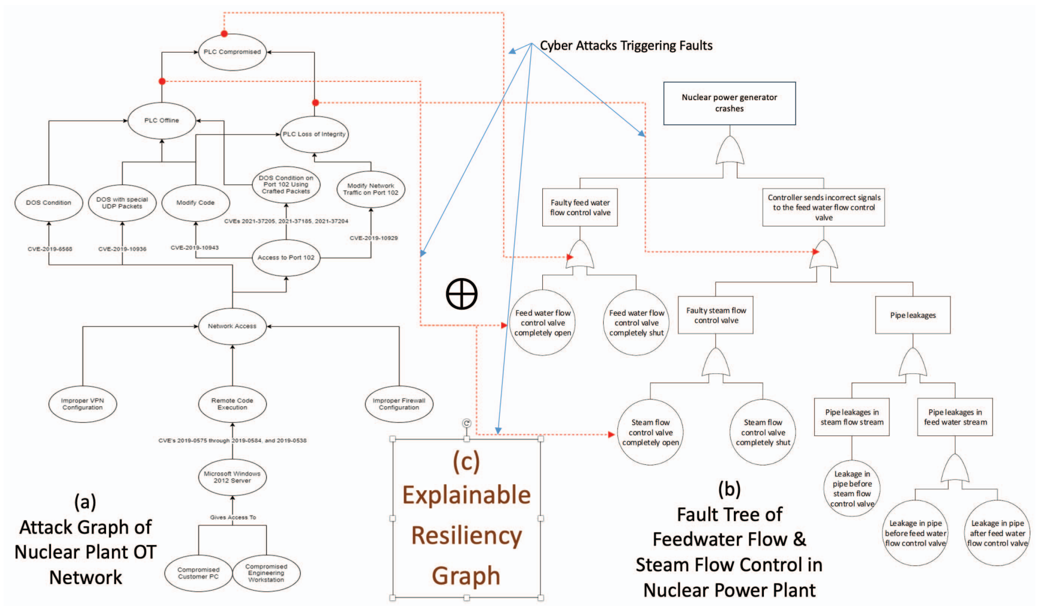

PLC Compromised
PLC Offline
PLC Loss of Integrity
DOS Condition
DOS with special UDP Packets
Modify Code
DOS Condition on Port 102 Using Crafted Packets
Modify Network Traffic on Port 102
CVEs 2021-37205, 2021-37185, 2021-37204
CVE-2019-6568
CVE-2019-10936
CVE-2019-10943
CVE-2019-10929
Access to Port 102
Network Access
Improper VPN Configuration
Remote Code Execution
Improper Firewall Configuration
CVE's 2019-0575 through 2019-0584, and 2019-0538
Microsoft Windows 2012 Server
Gives Access To
Compromised Customer PC
Compromised Engineering Workstation
(a) Attack Graph of Nuclear Plant OT Network
(c) Explainable Resiliency Graph
Cyber Attacks Triggering Faults
Nuclear power generator crashes
Faulty feed water flow control valve
Controller sends incorrect signals to the feed water flow control valve
Feed water flow control valve completely open
Feed water flow control valve completely shut
Faulty steam flow control valve
Pipe leakages
Steam flow control valve completely open
Steam flow control valve completely shut
Pipe leakages in steam flow stream
Pipe leakages in feed water stream
Leakage in pipe before steam flow control valve
Leakage in pipe before feed water flow control valve
Leakage in pipe after feed water flow control valve
(b) Fault Tree of Feedwater Flow & Steam Flow Control in Nuclear Power Plant

Fig. 1: A visualization of modeling cyber resiliency via ERG. Explainable Resiliency Graphs (c) allow us to compose ($\oplus$ represents composition) Attack Graphs (a) and Fault Trees (b) into a single unified representation. The dotted lines represent the areas where security and safety are composed.

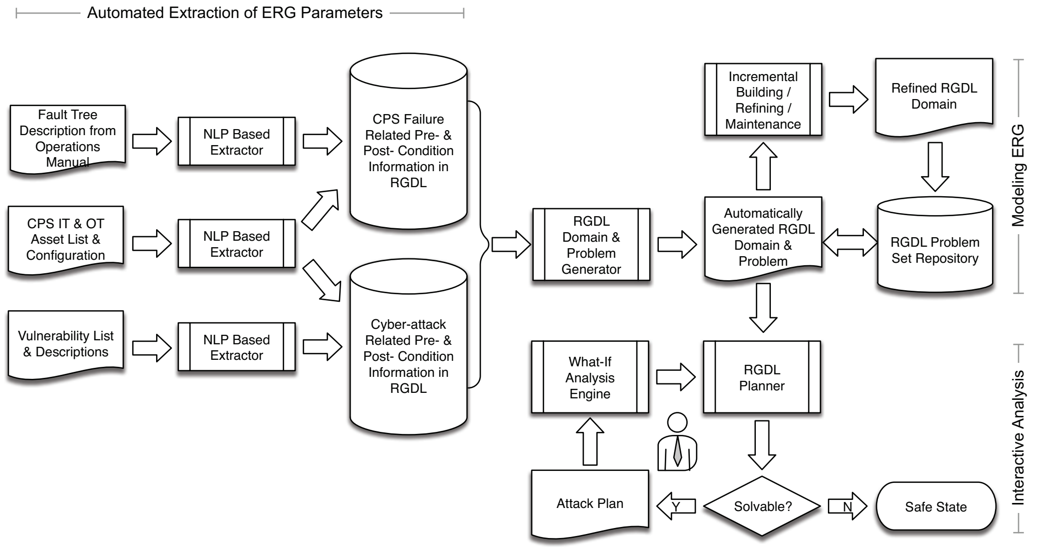

Automated Extraction of ERG Parameters
Fault Tree Description from Operations Manual
CPS IT & OT Asset List & Configuration
Vulnerability List & Descriptions
NLP Based Extractor
NLP Based Extractor
NLP Based Extractor
CPS Failure Related Pre- & Post- Condition Information in RGDL
Cyber-attack Related Pre- & Post- Condition Information in RGDL
RGDL Domain & Problem Generator
Automatically Generated RGDL Domain & Problem
Incremental Building / Refining / Maintenance
Refined RGDL Domain
RGDL Problem Set Repository
Modeling ERG
What-If Analysis Engine
RGDL Planner
Attack Plan
Solvable?
Y
N
Safe State
Interactive Analysis

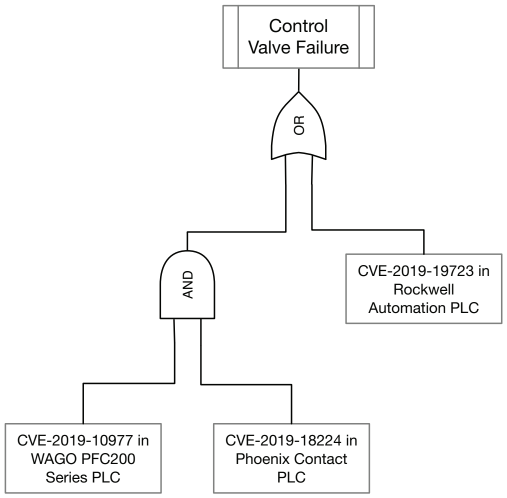
Fig. 2: Overall workflow showing the different steps involved in semi-automated ERG-based CPS resiliency. Step 1 in the Overview of Approach discussion has been omitted in this figure

Step 1: *Modeling CPS Resiliency* – We introduce an entirely new modeling paradigm for cyber-physical systems, named *Explainable Resiliency Graphs (ERG)*. Under this paradigm, we combine the information related to the security posture of a given network, popularly captured using attack graphs, with information related to safety events, traditionally represented through fault trees. This presents an entirely new set of modeling challenges, including capturing how the components in each part influence the other, consistently capturing qualitatively different information in a single unified framework, and supporting analysis over extremely large resiliency graphs.

Step 2: *Automated Extraction of ERG Parameters* – Even with a fully specified modeling paradigm, creating these graphs for large-scale application domains will remain a substantial challenge. Our initial works [18], [19] have shown how to create attack graphs automatically from CVE vulnerability descriptions using natural language processing (NLP) and AI Planning. We refine and adopt that work in this framework. However, the automatic generation of fault trees remains a mostly unstudied problem. We have done a pilot study to leverage state-of-the-art semantic parsing tools, including pre-trained large language models to extract formal descriptions of ERGs. However, we have identified several missing pieces in this puzzle where we plan to use a novel human-in-the-loop model acquisition method that will efficiently query domain experts to identify parts of the graph that may be incorrect or incomplete and get acquire the information required to fix them.

Step 3: *Developing Analysis, Detection and Interdiction over Resiliency Graphs* – We are in the process of developing a set of analysis and detection algorithms that allows the stakeholders to use the ERGs in two unique modes. As part of a set of tools used to analyze the security and safety posture, this involves performing worst-case analysis, identifying points of failure, and more importantly identifying potential fixes to these problems. Secondly, the graphs can be used as a way to monitor for and identify potential attacks or possible failure cascades. As with the previous use case, we also develop a method to come up with suggestions to stop the detected attacks/failures.

Step 4 *Developing Explanation Generation Methods for Experts to Support What-If Analysis –*. Developing powerful algorithms to analyze complex ERGs to detect possible sources of failure or vulnerability only forms a part of a system to empower users to leverage the true potential of the proposed model. To address this gap, we need to develop explanatory techniques that will help the users better understand both the analysis results and the suggestions being made to them. The explanatory algorithms need to be designed to address the two important challenges raised by this topic, namely the complexity of the model itself and the fact that results will be used for decision-making by users with different backgrounds and expertise. The former is the result of both the complexity and scale of modern-day CPS and our need to consider so many different factors in a single model. The second challenge is unavoidable due to the basic reality that the users who are best positioned to understand and analyze the security challenges need not be the same as the ones who are best suited to analyze safety issues and vice versa. This means the output generated by this system would need to be analyzed by a team of people with different backgrounds. As such, we would need to generate explanations that help build common ground between these different users.


Control Valve Failure
OR
AND
CVE-2019-19723 in Rockwell Automation PLC
CVE-2019-10977 in WAGO PFC200 Series PLC
CVE-2019-18224 in Phoenix Contact PLC
Fig. 3: Resiliency graph showing how a ICS control valve failure occurs via a cyber attack

III. Research Pathway and Challenges

In the following, we discuss in more detail the technical approach, identifying in the process some of the research challenges the approach opens up. Henceforth, we use the terms Explainable Resiliency Graphs and Resiliency Graphs to mean the same structure. For this discussion, we use a very small example. The resiliency graph, which is very tiny part of a bigger resiliency graph, is shown in Figure 3

A. Step 1: Modeling of Explainable Resiliency Graphs

Existing works in modeling network attacks have mostly separated the representation of the security posture of the network, from other potential sources of failure and safety events. However, past incidents have shown how attackers could potentially leverage security holes to initiate cascading failures, that might in turn affect the integrity of the entire CPS and even result in casualties. Analyzing such potential scenarios involves a richer modeling framework than what is used in practice. The Explainable Resiliency Graph paradigm is meant to bridge that gap. We start by trying to unify information that is generally captured through attack graphs and fault trees.

Attack graphs [20], are traditionally used to capture potential security vulnerabilities present in a given system, and for the most part capture how an attacker could potentially exploit them to compromise various components of the overall systems. A fault tree [21] on the other hand has been traditionally used to model potential sources of failures and dependencies between different events. While both models could be represented as transition systems with various states and potential transitions between them, there are qualitative differences between them that make a combined model technically challenging.

For one thing, the transitions within the attack graphs represent intentional actions carried out by an agent trying to achieve a specific objective, while transitions within fault trees represent events that are triggered by various physical or environmental conditions. Any effective modeling techniques should still retain such qualitative differences as they could have a profound impact on the potential consequences.

We adapt the classical Planning Domain Definition Language (PDDL) [17] as the base formalism to represent ERGs. PDDL was introduced as a domain-independent method to represent and specify goal-directed deterministic planning problems. However, vanilla versions of PDDL is not well-suited to retain all the qualitative differences between the different transitions that are possible under this modeling paradigm. At the very least, we are interested in separating three classes of transitions, namely, state transitions due to the attacker's actions, those caused due to the user's actions (either unintended or coaxed by the attacker), and finally events that are triggered by the occurrence of other physical or environmental events. We have observed that qualitative difference in these actions could make a considerable difference in the role performed in various analyses and corrective actions.

We refer to this novel description language as Resiliency Graph Description Language or RGDL. In particular, we define a resiliency graph under RGDL using a tuple of the form $\mathcal{R} = \langle F, A, U, E, I, T \rangle$, where F represents the set of state variables used to describe the transition system, A the actions available to the attacker, U the actions to be performed by the user, E events that could occur, I the initial state and T targets/nodes of importance and interest to the attacker. Under this definition, F defines the set of possible states in the underlying transition system, and $A \cup U \cup E$, the set of possible transitions.

B. Step 2: Automated Extraction of Resiliency Graph Parameters

PDDL-like languages have roots in folk psychological models of action [22] and have been known to boost explainability [23]. Thus, our choice to use languages that build on domain description languages like PDDL makes it well-suited for people to manually specify them. Unfortunately, the sheer complexity of modern-day CPS and the interdisciplinary nature of ERGs make them a bad candidate for pure manual declaration. On the other hand, a fully automated approach to extracting these graphs is also not currently possible. To address this, we use an expert-in-the-loop graph extraction pipeline that tries to extract as much of the graph automatically and queries the expert only when it identifies regions in the graph that may be incorrect or incomplete. Moreover, even when it queries an expert, the queries are tailored to best match their expertise and are posed so as to minimize the cognitive load imposed on them. Figure 4, presents an overview of the overall extraction pipeline.

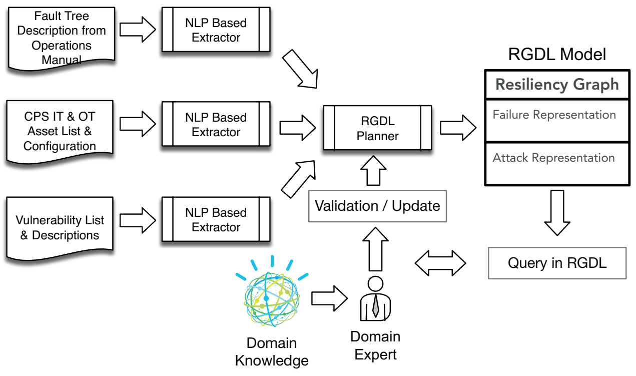


Fig. 4: A flowchart highlighting the process for extracting the model for the graph. The process involves multiple steps, with initial models being extracted from various sources. Information about various historic attacks and failures is used to validate and test learned models. Finally, domain experts are queried to identify missing information or to correct potential mistakes.

In the first phase of the pipeline, we look at the possibility of extracting parts of the graph from existing knowledge sources. Among the components considered, relatively more work has been done in extracting information related to security aspects. Vulnerability descriptions have proven to be a particularly rich source of information about attack graphs. Our own previous work, such as the AGBuilder system [24] has looked at the possibility of extracting PDDL description from vulnerability description and other information sources. However, even with the previous works in this area, extracting a complete attack graph remains a challenging problem in the context of complex infrastructure. When we move over to fault trees, the problem of extracting relevant information is much less explored. Unlike the security domain, there have been very few efforts to consolidate information related to potential safety events, and there is much more variability with regard to the specifics of the infrastructure. This remains an open area of further research. Nonetheless, there exist documents related to standard operating procedures and safety protocols that could be used to learn relevant information. We are investigating how to leverage state-of-the-art, large language model-based semantic parsers to extract formal representation from such unstructured textual sources. We denote the incomplete RGDL description learned in this phase as $\hat{\mathcal{R}}$.

The next phase of the pipeline takes the extracted model and tries to identify potential parts that are incomplete or incorrect. In particular, we perform this analysis by primarily leveraging two sources of information. First, we try to see if our model can support and potentially account for historical safety events and attacks (which are known to still exist in the network). We use semantic parsers to convert potential textual information into formal traces that can be validated against $\hat{\mathcal{R}}$. Whenever we find a trace (i.e., the path through the transition system) π, which cannot be supported by the current model, i.e $\pi \not\models \hat{\mathcal{R}}$, we generate a set of the hypothesis set $\mathbb{R} = \{\hat{\mathcal{R}}_1, ..., \hat{\mathcal{R}}_k\}$, where each hypothesis is generated by performing local edits on the original model and can support the trace in question.

We have observed that at this stage in the workflow we require expert-in-the loop to correct and refine the extracted ERG. This would involve presenting parts of the graph (or its descriptions) and asking the experts to identify potential mistakes or presenting them with the set of potential hypotheses and asking them to select the model that they believe may actually be correct. The critical challenges here are to ensure that queries about a specific part of the model are only posed to the user with the appropriate expertise and that the cognitive load posed by the query is minimized. Our earlier work modeling the expertise of specific users [25], which could be leveraged to address the former challenge, while the latter presents a unique set of challenges. While our group has worked on the use of abstraction as a means to facilitate presenting relevant parts of the domain model for correction [26], the heterogeneous nature of the modeling paradigm presents an additional complexity. Simply presenting a minimal abstraction of the relevant model component to the user may not be the easiest way to empower a user to effectively identify ways to correct the model. Instead, we will investigate and develop a method that creates abstractions that are user-specific and takes into account their particular technical background. For example, when parts of models are shown to a cybersecurity expert, the system should project out all but the most critical information about security events. Effectively, the system will be presenting an abstracted version of the corresponding attack graph with minimal information about safety events.

Figure 5 shows the output of the extraction process after it has been refined by domain experts.

```
(define (domain extended-plc-vulnerability)
  (:requirements :strips :typing :disjunctive-preconditions)

  (:types
    attacker
    plc
    service
  )

  (:predicates
    (has-access-plcnext ?who - attacker ?target - plc)
    (vulnerable-plcnext ?target - plc)
    (has-access-wago ?who - attacker ?target - plc)
    (vulnerable-wago ?target - plc)
    (has-access-phoenix ?who - attacker ?target - plc)
    (vulnerable-phoenix ?target - plc)
    (code-executed ?who - attacker ?target - plc)
    (commands-executed-wago ?who - attacker ?target - plc)
    (commands-executed-phoenix ?who - attacker ?target - plc)
    (plc-compromised ?Service - service)
    (running-service ?Service - service)
  )

  (:action exploit-buffer-overflow-DoS-Attack-CVE-2019-19723
    :parameters (?attacker - attacker ?target - plc)
    :precondition (and
                    (has-access-plcnext ?attacker ?target)
                    (vulnerable-plcnext ?target)
                  )
    :effect (code-executed ?attacker ?target)
  )

  (:action exploit-command-execution-wago-CVE-2019-10977
    :parameters (?attacker - attacker ?target - plc)
    :precondition (and
                    (has-access-wago ?attacker ?target)
                    (vulnerable-wago ?target)
                  )
    :effect (commands-executed-wago ?attacker ?target)
  )

  (:action exploit-command-execution-phoenix-CVE-2019-18224
    :parameters (?attacker - attacker ?target - plc)
    :precondition (and
                    (has-access-phoenix ?attacker ?target)
                    (vulnerable-phoenix ?target)
                  )
    :effect (commands-executed-phoenix ?attacker ?target)
  )

(:action compromise-overall-plc
  :parameters (?attacker - attacker ?target1 - plc
        ?target2 - plc ?target3 - plc ?Service - service)
  :precondition (and
                  (code-executed ?attacker ?target1)
                  (or
                    (commands-executed-wago ?attacker ?target2)
                    (commands-executed-phoenix ?attacker ?target3)
                  )
                  (running-service ?Service)
                )
  :effect (plc-compromised ?Service)
)

)


(define (problem extended-plc-problem)
  (:domain extended-plc-vulnerability)

  (:objects
    attacker-1 - attacker
    plcnext-controller - plc
    wago-pfc200 - plc
    phoenix-axc-f2152 - plc
    ICS-PLC - service
  )

  (:init
    ;(has-access-phoenix attacker-1 phoenix-axc-f2152)
    ;(vulnerable-phoenix phoenix-axc-f2152)
    (has-access-plcnext attacker-1 plcnext-controller)
    (vulnerable-plcnext plcnext-controller)
    (has-access-wago attacker-1 wago-pfc200)
    (vulnerable-wago wago-pfc200)
    (running-service ICS-PLC)
  )

  (:goal (and
        (plc-compromised ICS-PLC)
       )
  )
)
```

Fig. 5: RGDL model of ERG from Figure 3 automatically generated

C. Step 3: Developing Analysis, Detection and Interdiction over Resiliency Graphs

Before we discuss the exact analysis and inference problems we are studying, let us take a quick look at the three main components used to capture transition within RGDL and talk about their qualitative nature:

- Attacker Actions A: These correspond to the various actions that can be performed by the attacker. These actions will have consistent semantics across all the problems discussed in this section; namely, an intentional action performed by an attacker to achieve the attacker's objective, which is to compromise some of the targets of importance to the attacker.

- User Actions U: These are the actions performed by the users of the system. These actions are important to model; in many cases, attackers require or leverage user actions towards achieving their own objectives, for example to deliver an attack payload. While for some analysis user actions may be modeled as intentional actions, they may also be modeled as stochastic transitions that are triggered in response to attacker actions or because of reaching certain states. Here the specific probabilities associated with the transitions may be automatically learned by using user activity logs.
- Safety Events E: There are various safety or fault events that may be triggered intentionally by attackers or in response to other events or even randomly. As with the user actions, based on the form of the problem being considered, the events may either be treated as an intentional action or a stochastic transition. The exact probabilities here are either learned from the different procedural documents, from experts, or even from system logs of previous events.

The Explainable Resiliency Graph provides us with a basis for developing an extremely diverse set of highly valuable analytical tools. Below we discuss some of the more immediate and important ones that we have started investigating from the point of view of both actionability and usefulness in building a good understanding of the security/safety posture.

a) Worst-Case Analysis: One of the fundamental questions that we can ask for resiliency analysis is given an ERG, what is the worst damage an attacker could bring about. Here we are not only anticipating the potential attack paths that an attacker could utilize but also trying to account for potential events and user actions that could inadvertently occur that may compound and magnify the effectiveness of the attacker. We perform this analysis by treating each of the three sets of transitions, A, U, and E, as intentional actions an attacker could wield to compromise the targets. This in a sense captures the extreme case where everything works out in their favor. This problem will be modeled as a deterministic planning problem.

b) Average-Case Analysis: As alluded to previously, the worst-case analysis looks at the most extreme failure scenarios. In reality, this scenario could be extremely unlikely and would require a number of factors outside the control of the attacker to occur. While useful, this may not really reflect the most common failure and attack modes. To capture this, we can perform an analysis where only the attacker actions are treated as intentional actions, but the events and user actions are treated as stochastic effects that may or may not be triggered based on probability. This formulation allows us to compute various information, including most likely attack path, most likely failure and/or compromised assets, the expected value associated with each attacker action, and so on. This problem will be modeled as a probabilistic planning problem.

c) Attack-triggerd Cascading Failure Analysis: The previous analysis corresponds to ones that are performed preemptively to understand the security posture and the potential of cascading failures in a given system. In addition to such analysis, we need a real-time method to detect when the CPS may be under attack. We need to detect whether observed events may be part of some ongoing attack or simply unrelated events. We will be building on existing plan recognition work [27] to support such detection. Like the average case analysis, we model user actions and safety events as stochastic transitions and look for the most plausible set of attacker actions that can explain the given set of observations. An important aspect where more research is needed, is quantifying the likelihood of this attack. We can do this quantification either by using known models of attackers or by ascribing rational behavior to the attacker, i.e., an attack is less likely if there exist more effective ways of achieving the possible attacker goal. We can also perform similar detection of the start of a catastrophic cascade of safety events and failures.

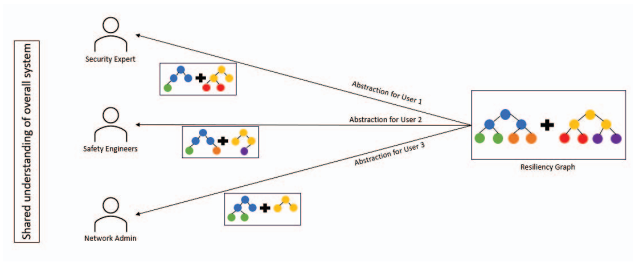


Fig. 6: Overview of the explanation process where the user's background is taken into account to generate explanations tailored to a specific user and their expertise.

d) Attack/Failure Interception and Disruption Analysis: Once an attack or failure is detected, the next course of action is to come up with a series of steps to stop the attack or failure. Such plans of recourse will consist of actions that are outside the set of actions considered within the ERG itself. For example, it may involve potentially disconnecting a compromised system from the network or changing user behavior. Regardless, each of these actions would correspond to modifying the resiliency graph such that the current attacker path (or the potential series of failures) is disrupted. As such, we can carry out the analysis of identifying such plans using the resiliency graph itself. We have done extensive previous work on model-space search problems [23], which looks at the meta-search problem of modifying a given planning model to get ones with different properties. We are be extending these works to support updating resiliency graphs.

D. Step 4: Developing Explanation Generation Methods for Experts to Support What-If Analysis

We are in the process of developing explanatory algorithms to support solutions generated for each of the problems described in the previous step. We are leveraging our previous works in developing explanation generation for planning-based systems and for decision-support systems [28], [29]. We adapt existing planning formalisms and extend those to develop explanation generation methods.

a) Generating Explanations Tailored to the Expert's Background: One of the critical challenges that we are faced with in our work is that any observation made about the system would probably be made by a team of experts with distinct backgrounds. This means that we need to generate multiple explanations each tailored to the unique background of the user. At the same time, we also need to make sure that over time the system is able to build a common ground between the users so they can agree on the final decision. This is a challenging problem and we welcome the community's suggestions in this matter.

For the time being, we are leaning on a strategy similar to the one discussed for Step 2, as part of the system to query the experts about the model. Specifically, we use abstractions of the graph that takes into account the particular technical background of each user. The abstraction will only contain information about parts outside their field of expertise to the degree it is needed to make sense of the specific analysis. The explanations will be generated using this abstraction. The user will also have the ability to concretize the model incrementally thus adding more information about other components of the system. This would allow each user to understand more about the overall system.

b) Supporting What-If Analysis to Strengthen Resiliency Graph: Our objective here is to not just generate explanations that allow the users to better understand the properties of the current system, but also to provide them the ability to update the system, so as to fix any potential issues that may have been detected as part of the analysis. The planned approach involves two closely related processes. First, the system is going to identify a set of ways through which the overall system could be made more robust. Next, the options would be presented to the user along with explanations as to how these changes would improve the overall system. The experts then can decide which of the changes they want to implement. For analysis, we can use methods similar to the one developed for the previous steps. We are interested in more than just disrupting a specific path. So, we consider broader goals like making the system more robust as measured under the worst-case analysis or the average-case one. Here, we will also have the additional objective of identifying multiple diverse ways of updating the graph to achieve the desired objective. By giving the users different options, they would be able to reason about the trade-offs of going with one set of updates over another.

IV. Conclusions

As more and more industrial control systems (ICS) become digitized they become targets of malicious cyber attacks. The potential for disruptions in these cyber physical systems (CPS) can be attributed to the dependence and the vulnerability of the networks interconnecting the physical plants and control centers. Such attacks can have very serious implications on a CPS's operation especially when the cyber-attacks trigger safety events either directly or via triggering cascading failures in the ICS. Therefore, there is a significant need for deeper insights into cyber risks to cyber physical systems – insights that help operators to determine what the weak spots are that can be leveraged by APTs, LOLBins, or other malware to launch cyber attacks affecting safety, and how to best deploy defenses for resiliency of these CPSs.

In this paper, we present an *Explainable AI* approach that combines the power of Natural Language Processing (NLP) and AI Planning to enable CPS operators to evaluate and analyze how a cyber-attack can trigger safety events in the CPS (that is, the *resiliency posture of the CPS*). The approach consists of a model, called Explainable Resiliency Graph, that expresses the dependencies between cyber attack and ICS failure. The model is supported by NLP and AI planner based tools for automated generation of the model for a specific CPS as well as tools that allow the operator to interact with the AI planner to query the underlying ERG transition system and perform what-if analysis. This analysis provides actionable suggestions from the tool including insights into potential attack vectors and help prioritize efforts to secure critical assets effectively. These suggestions comprise of a diverse set of solutions each of which can potentially take the CPS to a safe and secure state. To help the operator decide which of the actionable suggestions to implement, the toolset provides explanations in natural language.

Acknowledgment

This work was partially supported by the U.S. NSF under Grant No. 1822118 and 2226232 by support from the industry partners of the NSF IUCRC Center for Cybersecurity Analytics and Automation – ARL, Statnett, AMI, NewPush, and Cyber Risk Research – by the US DOE under contract DE-NE0008986, and by the U.S. NIST under Award Number 60NANB23D152. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF, DOE, NIST, other federal agencies or the industry.

References

[1] "SQL Slammer," Mar. 2023, page Version ID: 1146402163. [Online]. Available: https://en.wikipedia.org/w/index.php?title=SQL_Slammer&oldid=1146402163

[2] N. Falliere, Liam O Murchu, and Eric Chen, "W32.Stuxnet Dossier: Version 1.4," Symantec Corporation, Symantec Security Response, Feb. 2011. [Online]. Available: https://www.wired.com/images_blogs/threatlevel/2010/11/w32_stuxnet_dossier.pdf

[3] A. K. Lab, "BlackEnergy APT Attacks in Ukraine," Available from Kaspersky Resource Center - https://usa.kaspersky.com/resource-center/threats/blackenergy, Apr. 2023.

[4] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated Generation and Analysis of Attack Graphs," in *Proceedings 2002 IEEE Symposium on Security and Privacy*. IEEE, 2002, pp. 273–284.

[5] X. Ou, S. Govindavajhala, A. W. Appel *et al.*, "Mulval: A Logic-Based Network Security Analyzer." in *USENIX security symposium*, vol. 8. Baltimore, MD, 2005, pp. 113–128.

[6] X. Ou, W. F. Boyer, and M. A. McQueen, "A Scalable Approach to Attack Graph Generation," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 336–345.

[7] K. Kaynar and F. Sivrikaya, "Distributed Attack Graph Generation," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 5, pp. 519–532, 2015.

[8] N. Poolsappasit, R. Dewri, and I. Ray, "Dynamic Security Risk Management Using Bayesian Attack Graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 61–74, 2011.

[9] R. Dewri, N. Poolsappasit, I. Ray, and D. Whitley, "Optimal Security Hardening Using Multi-Objective Optimization on Attack Tree Models of Networks," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07. New York, NY, USA: Association for Computing Machinery, Oct. 2007, pp. 204–213. [Online]. Available: https://doi.org/10.1145/1315245.1315272

[10] R. Dewri, I. Ray, N. Poolsappasit, and D. Whitley, "Optimal Security Hardening on Attack Tree Models of Networks: A Cost-Benefit Analysis," *International Journal of Information Security*, vol. 11, no. 3, pp. 167–188, Jun. 2012. [Online]. Available: https://doi.org/10.1007/s10207-012-0160-y

[11] M. U. Aksu, K. Bicakci, M. H. Dilek, A. M. Ozbayoglu, and E. ı. Tatli, "Automated Generation of Attack Graphs Using NVD," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, 2018, pp. 135–142.

[12] H. A. Watson *et al.*, "Launch Control Safety Study," Bell Labs Report, 1961.

[13] K. J. Sullivan, J. B. Dugan, and D. Coppit, "The Galileo Fault Tree Analysis Tool," in *Digest of Papers. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing (Cat. No. 99CB36352)*. IEEE, 1999, pp. 232–235.

[14] L. Xing and S. V. Amari, "Fault Tree Analysis," *Handbook of performability engineering*, pp. 595–620, 2008.

[15] K. Aslansefat and G.-R. Latif-Shabgahi, "A Hierarchical Approach for Dynamic Fault Trees Solution Through Semi-Markov Process," *IEEE Transactions on Reliability*, vol. 69, no. 3, pp. 986–1003, 2020.

[16] C. Bäckström and P. Jonsson, "Abstracting Abstraction in Search with Applications to Planning," in *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press, 2012.

[17] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL - The Planning Domain Definition Language," Yale Center for Computational Vision and Control, Technical Report CVC TR-98-003/DCS TR-1165, October 1998.

[18] S. Weerawardhana, S. Mukherjee, I. Ray, and A. Howe, "Automated Extraction of Vulnerability Information for Home Computer Security," in *Foundations and Practicea of Security: 7th International Symposium, FPS 2014, Montreal, QC, Canada, November 3-5, 2014. Revised Selected Papers 7*. Springer, 2015, pp. 356–366.

[19] F. Hashemi Chaleshtori and I. Ray, "Automation of Vulnerability Information Extraction Using Transformer-Based Language Models," in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 645–665.

[20] C. A. Phillips and L. P. Swiler, "A Graph-Based System for Network-Vulnerability Analysis," in *Proceedings of the 1998 Workshop on New Security Paradigms, Charlottsville, VA, USA, September 22-25, 1998*. ACM, 1998, pp. 71–79.

[21] E. Ruijters and M. Stoelinga, "Fault Tree Analysis: A Survey of the State-of-the-Art in Modeling, Analysis and Tools," *Computer Science Review*, vol. 15-16, pp. 29–62, Feb. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013715000027

[22] B. F. Malle, *How the Mind Explains Behavior: Folk Explanations, Meaning, and Social Interaction*. MIT press, 2006.

[23] S. Sreedharan, "Foundations of Human-Aware Explanations for Sequential Decision-Making Problems," Ph.D. dissertation, Arizona State University, 2022.

[24] K. Tiwary, S. Weerawardhana, I. Ray, and A. Howe, "PDDLassistant: A Tool for Assisting Construction and Maintenance of Attack Graphs Using PDDLassistant: A Tool for Assisting Construction and Maintenance of Attack Graphs Using PDDL," in *Proceedings of the ACM Conference on Computer and Communications Security 2017 (CCS 2017)*, 2017.

[25] S. Sreedharan, S. Srivastava, and S. Kambhampati, "Hierarchical Expertise Level Modeling for User Specific Contrastive Explanations," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, J. Lang, Ed. ijcai.org, 2018, pp. 4829–4836.

[26] S. Sreedharan, T. Chakraborti, C. Muise, Y. Khazaeni, and S. Kambhampati, "- D3WA+ - A Case Study of XAIP in a Model Acquisition Task for Dialogue Planning," in *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020*. AAAI Press, 2020, pp. 488–498.

[27] R. Mirsky, S. Keren, and C. W. Geib, *Introduction to Symbolic Plan and Goal Recognition*, ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2021.

[28] K. Valmeekam, S. Sreedharan, S. Sengupta, and S. Kambhampati, "RADAR-X: An Interactive Mixed Initiative Planning Interface Pairing Contrastive Explanations and Revised Plan Suggestions," in *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. AAAI Press, 2022, pp. 508–517.

[29] S. Sengupta, T. Chakraborti, S. Sreedharan, S. G. Vadlamudi, and S. Kambhampati, "RADAR - A Proactive Decision Support System for Human-in-the-Loop Planning," in *2017 AAAI Fall Symposia, Arlington, Virginia, USA, November 9-11, 2017*. AAAI Press, 2017, pp. 269–276.