Towards Security Attack Event Monitoring for Cyber Physical-Systems

Elias Seid, Oliver Popov and Fredrik Blix
Department of Computer and Systems Sciences, Stockholm University, Sweden

Keywords: Cyber Physical-Systems, Industrial Internet of Things, Security Requirements, Goal Model, Attack Pattern, Domain Assumption.

Abstract: In today's software systems, security is one of the a major issues that need to be considered when designing Cyber Physical-Systems(CPS). CPS are engineered systems built from, and depend upon, the seamless integration of computational algorithms and physical components. Security breaches are on the rise, and CPS are challenged by a catastrophic damage which resulted in billions of losses. Security solutions to the Cyber Physical-Systems that we have are likely to become obsolete. Even though security agents issue new sets of vulnerability indicators and patches to address the security breach, these vulnerability indicators change over time, which is a perpetual process. We argue that any security solution for the Cyber Physical-Systems should be adaptive, based on the type of attacks and their frequency. The security solution should monitor its environment continuously to defend itself from a cyber-attack by modifying its defensive mechanism. We propose a framework for modelling, analyzing and monitoring security attacks (events) in the social, cyber and physical infrastructure realms of CPS. The framework is evaluated by using security attack scenarios taken from a recognized security knowledge repository.

1 INTRODUCTION

Industrial automation and control systems have been studied independently than together within the ICT framework. initially a firewall was used to mitigate any potential threat coming towards the core component of the system (Boyes et al., 2018). However, the disruption of internet of things has made a huge impact in changing the infrastructure of the Industrial automation and control system, particularly changes related to Industrial systems such the electrical management systems, manufacturing etc. The concept of IACS has been studied in the area of cyber physical systems. Since they were integrated into industries such as in manufacturing, energy management system and the like, they were named cyber-physical systems while doing only the operational activities of the industries. After the disruption by internet of things in 1999, it becomes familiar to be used within the connected devices in various business and industrial settings (Boyes et al., 2018).

A number of definitions of cyber physical-systems (CPS) have been studied in the literature (Griffor et al., 2017), (Henzinger et al., 2008), (Baheti and Gill, 2011), (Poovendran, 2010) and (Shafi, 2012). However, (Boyes et al., 2018) has summarized all and integrated all concepts addressed in literature that talk about CPS, and has set a common ground. And CPS is defined as system comprising a set of interacting physical and digital components that can be centralised or decentralised". Such definition has three major components; sensing, computation and networking that is essential to address the real world by employing physical process. Industrial Internet of Things(IIoT) extend the concepts and definitions used in CPS, and are defined as "smart connected assets or things as part of a larger system or system of systems that make up the smart manufacturing enterprise"(Conway, 2016).

Moreover, IIoT seek to connect industrial assets such as engines, power grids, and sensors to the cloud by using the network. (Helmiö et al., 2017), (Conway, 2016), (Helmiö et al., 2017) made an effort that industrial assets are considered core components of IIoT that are connected in the industrial setting. Furthermore, the industrial assets are connected to a cloud over a network, and thus industrial assets are able to generate, transfer, and analyse information at real time and monitor their surrounding environment based on the information exchanged (Jeschke et al., 2017).

The authors claim that this all exchange of information is done with out human intervene however, we argue that the industrial setting needs human in

Seid, E., Popov, O. and Blix, F.
Towards Security Attack Event Monitoring for Cyber Physical-Systems.
DOI: 10.5220/0011803400003405
In *Proceedings of the 9th International Conference on Information Systems Security and Privacy (ICISSP 2023)*, pages 722-732
ISBN: 978-989-758-624-8; ISSN: 2184-4356
Copyright © 2023 by SCITEPRESS – Science and Technology Publications, Lda. Under CC license (CC BY-NC-ND 4.0)

the loop to manage and monitor industrial assets as their physical world of cyber physical-systems run in an insecured environment. The assertion that CPS devices can generate, analyse, exchange and monitor information with out human in the loop does not make the definition complete. However, CPS are socio-technical ecosystems involving people, processes, technology, and infrastructure and are usually designed in a "piecemeal" rather than a holistic fashion, leaving parts of the system vulnerable (Müller et al., 2016).

There have been challenges in addressing multi-stage attacks on CPS as their components have become vulnerable for a *cyber-attack*. Tackling such attacks remains a taunting unsolved problem, since components come with its own complexities and vulnerabilities, and they need coordinated security solution (Wang et al., 2010).

Moreover, Cyber Physical-Systems(CPS) have the potential to transform the way we live, work, and interact with engineered systems as the Internet has transformed the way people interact with information. Smart cities, smart power grids, intelligent homes with network of appliances, robot assisted living, environmental monitoring and transportation systems are examples of such complex systems and applications. These complexities will drive innovation and competition in sectors such as agriculture, energy, transportation, building design and automation, health-care, and manufacturing.

In CPS, the physical world is integrated with sensing, communication, and computing components, and these components have complex interactions (Banerjee et al., 2012).

Motivation. Internet of things (IoT) are a particular type of CPS. As IoT scales to billions of connected devices with the capacity to sense, control, and otherwise interact with the human and physical world the requirements for dependability, security, safety, and privacy grow immensely.

Devices in the Internet of Things (IoT) generate, process, and exchange large amounts of security and safety-critical data and privacy-sensitive information, and that led to be potential targets of different security attacks (Poulsen, 2003), (Turk, 2005), (Buchs et al., 2013). For instance, a sensor that measures the temperature of a physical environment and transmits data accordingly is a smart item.

The domain of smart items encompasses heterogeneous, dynamic and flexible networks commonly referred to as sensor networks. Such networks consist of a large number of spatially distributed smart items that can be easily attached to physical entities and are able to monitor a number of parameters (e. g, temperature, sound, vibration, pressure and motion), and transmit data to remote software systems.

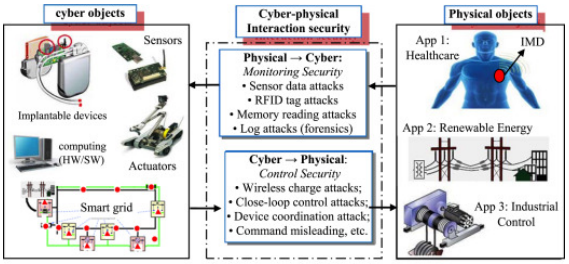

Figure 1: Cyber Physical-Systems (CPS)(Hu et al., 2016).

Ensuring a security solution for cyber physical-systems is much harder than security solution for software systems because designing security solution for CPS not only considers properties of component (sensing, communication, and computing components) but also their interaction with the physical environment (Banerjee et al., 2012). Attackers can exploit *CPS* components that are subject to increased risk and probability of attack, since these components (e. g., the sensors) communicate and interact in an open and thus insecure environment where security threats, such as illegal disclosure of information, transmission of falsified data, authentication and/or authorization violations, must be taken into consideration. For example, CPS items may be exposed to the public, for instance, by being attached to vehicles or containers, and their data may be easily manipulated, removed or destroyed. Particularly, sensors may provide, at runtime, critical security information to the application itself raising new security requirements that must be then considered and satisfied (Müller et al., 2016).

Objective of the Paper. We are interested in developing a framework named *Asfalia*[1], for modelling, analysing and monitoring of security attack-events for Cyber Physical-Systems. Specifically, we build security attack models, generate run time behaviors and events from behavioral models. We employed Common Attack Pattern Enumeration and Classification (CAPEC), a well-known attack pattern repository as an input for our model, taken from security knowledge sources[2].

To meet our objective, models have to be integrated representing 1) security attack patterns (Li et al., 2014), for modeling security attacks, 2) contextual goal models, to build attack models in terms of contextual goals (Ali et al., 2010) and 3) runtime goal models (Dalpiaz et al., 2013), to generate behavioral models. We also adopt some concepts from

[1] *Asfalia* (in Greek) means *security*
[2] https://capec.mitre.org/

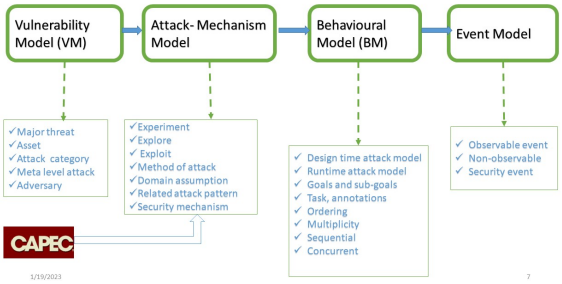

Figure 2: Components of the Attack Model.

the *Event Model* in (Cailliau and van Lamsweerde, 2017), to generate events using systematic approach from leaf-level tasks and their corresponding analysis.

The model consists of the following four major components: 1) **Vulnerability Model (VM)**: This model represents type of attack patterns, potential threats and type of assets. An *asset* is anything that has value to an organization (data or service) which is central as a target of the attack (Li et al., 2018). *Vulnerability* determines relevance of security goals, and a *security goal* is a desire to protect an asset (resource) or organizational policy, rules and regulations. An obstacle to an asset compromises security concerns such as integrity, confidentiality, availability and accountability (Türpe, 2017). 2) **Attack-Mechanism Model (AM)**: This model captures goal models, domain assumptions, attack mechanisms and task operationalization. In this model, we capture, understand and represent adversary behaviors of attack both at *design-time and runtime* that facilitates to derive security events. 3) **Behavioral Model (BM)**: This model captures fundamental system behaviors such as sequential, interleaving, multiplicity, alternative instances of attack goal models, sub-goals, domain assumptions, tasks (mechanisms) and events of attack patterns. System behavior annotations will be discussed in detail in section four. 4) **Event Model (EM)**: This model captures events that are derived from the attack models. These events could be grouped into observable and non- observable events. However, in this work we only focus on observable events that includes information about type, value and context.

The rest of the paper is structured as follows. Section 2 presents research baseline for our work, while section 3 presents *Asfalia* framework. We provide in sections 4 further details and examples concerning the models of the framework. In section 5 we present experiment and results, while section 6 deals with the discussions. Section 7 present related work, and section 8 concludes and discusses future work.

2 RESEARCH BASELINE

2.1 Goal-Based Contextual Requirements Modeling

A goal-oriented RE modelling and reasoning framework has been proposed for systems that run in numerous contexts, and establishes goal models to relate goals and contexts (Ali et al., 2010). The framework extended Tropos (Bresciani et al., 2004) goal model to capture the relationship between each variant to goal satisfaction. In particular, contexts are considered as labels that can be mapped to specific goal model artifact. Moreover, they define semantics for contexts that are modelled within goal models (Ali et al., 2010). We used and integrated some of their elements into *Asfalia* framework.

2.2 Runtime Goal Model

In the last decade, research in adaptive software systems have shown a rapid growth. Adaptive means that software systems should monitor its environment continuously to defend itself from external obstacles by changing its alternative mechanism (Souza and Mylopoulos, 2012). Recent approaches have shown that self-adaptive system softwares depends on variants of goal models to coordinate monitoring and adaptation mechanism. Dalpiaz (Dalpiaz et al., 2013) studied conceptual distinction between *design-time goal models* and *runtime goal models*. The authors have made the following contributions: 1) The relationship between design-time goal models, the derived *runtime goal models* and *runtime instance* in terms of Goal-Oriented Requirement Models. 2) Reasoning approach using runtime goal instance. In *Asfalia*, we extended (Dalpiaz et al., 2013) to generate behavioral models for the security attack strategies thus derived runtime attack pattern models.

2.3 Three-Layer Security Requirements Analysis Framework

The framework supports the analysis of security requirements for socio-technical systems(STSs) from a holistic perspective, meaning that the framework spans the three layers of STSs. Moreover, the framework supports ”cross layer analysis that spins off security requirements across realms”: Security solutions offered for security requirements in the social layer spin off security requirements in the cyber-layer, and solutions adopted for these requirements further spin off security requirements in the physical

layer(Li and Horkoff, 2014). *The three-layer framework has the following procedures*:

Our approach shares similarities with this framework in terms of layer specific-concepts as our models (*VM, AM, BM and EM*) analyze vulnerabilities, design attack mechanism, and build behavioral models in a realm-specific approach meaning they span the three realms of CPS (*cyber, physical infrastructure including the social one*).

2.4 Security Patterns

A pattern is defined in terms of context, problem and solution. Patterns might be specified to solve a problem in a certain context (Fernández et al., 2008). Moreover, Security pattern use well-established security knowledge to support analysts to solve security problems. These patterns combine security requirement and security mechanism, and have been used to solve a particular security problem. Many security patterns have been proposed in the literature. (Li and Horkoff, 2014) have used security patterns to operationalise security goals based on security mechanisms, and they leveraged organizational security pattern, physical security pattern physical security pattern and software security patterns for security requirement analysis process.

Modeling Security Patterns as Contextual Goal Model. An attack is the use of an exploit by an adversary to take advantage of a weakness with the intent of achieving a negative social and technical impact. Therefore, attacker's behavior is expressed in terms of the task and how to perform the task with respect to a target such as software application (asset). (Li et al., 2014) has integrated security patterns to support security requirements analysis technique by modeling security patterns as contextual goal model.

Attack Pattern Modeling. In the literature, (Li et al., 2015) have leveraged anti-goal approach (Van Lamsweerde, 2004) to analyse system adversaries to provide a security solution. This approach analyses anti-goal refinement from attacker's perspective by using different real attack scenarios at design-time. We extended (Li et al., 2014) to build behavioral models for security attack patterns and integrated in **(VM, AM, BM, EM)** models of our framework.

3 *Asfalia* FRAMEWORK

We describe here an overview of the framework named (*Asfalia*), and we provide in detail each model of the framework with illustrative example in section 4. The framework consists of the following elements: **Vulnerability Model (VM)**, **Attack-Mechanism Model (AM)**, **Behavioral Model (BM)** and **Event Model (EM)**. Figure 3 shows an extracted example models of the framework. Each components of the framework are described as follows:

Vulnerability Model (VM). Captures type of attack patterns, In[2], patterns are grouped into two categories: The first category of attack is *Domain-based attack* and the other is *Mechanizm-based attack*. For example, *Social Engineering* is a type of attack categorised under *domain-based attack* category while *Collect and Analyze Information* is type of attack under *mechanizm-based attack*[2].These category of attacks will be discussed in section five, in the experiment section.

Potential threats and type of asset are also captured in this model. An asset is anything that has value to an organization, and could be tangible (physical) and intangible (non-physical) assets in which the target of the attack is concerned. Security concerns such as confidentiality, integrity, availability and accountability to an asset can be associated to the business, application or infrastructure context. For example, (Haley et al., 2008) has analyzed asset and the corresponding security goal from business context of a system. In the next section, we show models (*VM, AM, BM and EM*) where vulnerabilities imposed to an asset are refined with respect to the *business*, *cyber* and *physical Infrastructure* context of the system, using different type of attack scenarios taken from well-known security knowledge sources[2].

Attack-Mechanism Model (AM). Captures goal models, domain assumptions, attack mechanisms and task operationalizations that are used to operate cyber-attack by taking advantage of a weaknesses of CPS with the intent of achieving a negative social and technical impact. Absent of adversaries (lack of actor) when designing a secured system have been realized as a major challenge for security requirement engineering researchers. Adversaries are anti-stakeholders who have a goal to compromise a system. For example, *threats* are indirectly discovered because adversaries whose goal, potential, and behaviors define threats of a system.

Behavioral Model (BM). Represents and captures runtime behaviors of an attack strategies. It annotates runtime attack behaviors of security attack. In this model, *Attack-Mechanism Model (AM)* is employed to generate attack behaviors. Moreover, *(VM and AM)* models use and extend concepts from the three-layer requirement analysis framework (Li and Horkoff, 2014) while *(BM and EM)* models use and extend (Dalpiaz et al., 2013). To fully understand threats and vulnerabilities in *VM and EM*, linear, top-

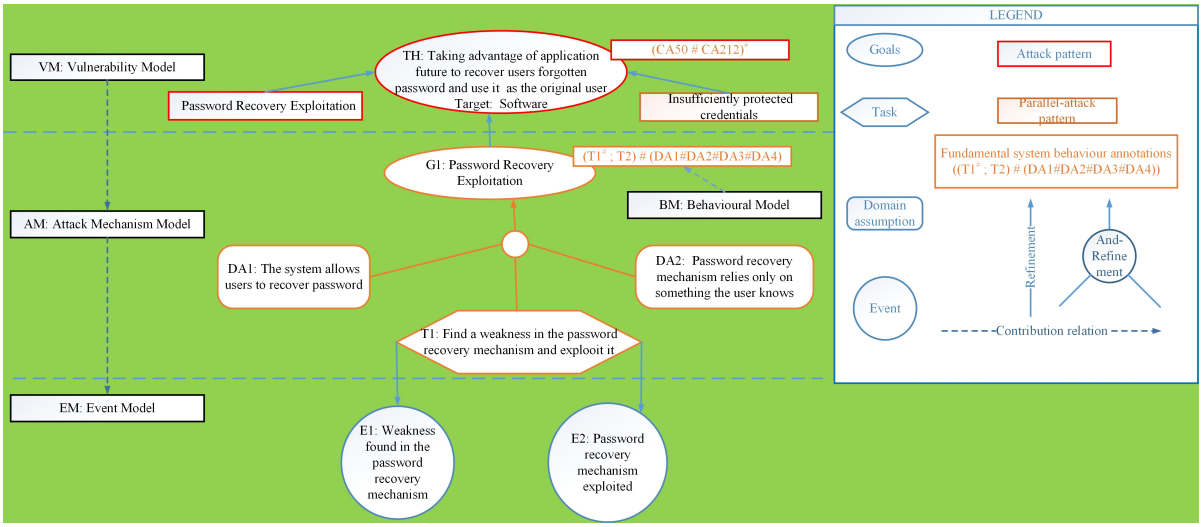

Figure 3: Extracted *Asfalia* Model.

down analysis approach may not be sufficient (Türpe, 2017). We integrated an *iterative* approach to analyze realm-specific vulnerabilities. We reuse and adopt syntax and denotation of goal expressions from (Dalpiaz et al., 2013), as shown in Figures 5 and 6

Event Model (EM). This model captures events that are derived from attack mechanism and behavioral models *(BM and AM)*. We describe *EM* model in the next section using various attack scenarios.

4 THE MODELS

We enriched the models (*VM*, *AM*, *EM* and *BM*) by using and revisiting the existing contextual guideline from attack pattern repositories[3]. These enriched guidelines are described as follows: 1) **Explore**: Searching for vulnerability (a weakness) that can be directly used by an adversary. 2) **Experiment**: Find a specific weakness with the intention of achieving a negative consequence. 3) **Exploit**: Is an input or action designed to take advantage of a weakness (or multiple weaknesses) and achieve a negative social and technical impact. 4) **Attack prerequisites**: Assumptions about properties of people, processes, technology and infrastructure that are assumed to hold from the attackers point of view for an adversary to succeed.

Specifically, we enriched the guidelines by following the *contextual goal model* approach (Ali et al., 2010). We applied them in our models, in order to analyze potential vulnerabilities, and captures knowledge about how specific parts of an attack are designed and executed. For instance, as shown in the model[4] *an attacker is able to cause a victim to load content* into their web-browser that *bypasses security* zone controls and gain access to increased privileges to *execute scripting code* (malicious content).

We illustrate this attack scenario using the abovementioned enriched contextual guidelines (procedures) as follows:

Explore. *Find systems susceptible to the attack*, find systems that contain functionality that is accessed from both the internet zone and the local zone.

Experiment. *Find the insertion point for the payload*, the code to be executed. The attacker first needs to find some system functionality or possibly another weakness in the system.

Exploit. *Craft and inject the payload*, the code to be executed, and *develop the payload* to be executed in the higher privileged zone in the user's browser.

Attack Prerequisites. *Insufficient Input Validation* by the system. In this section we provide the models of the framework using illustrative attack pattern scenarios.

4.1 Vulnerability Model(VM)

This model captures type of attack patterns, potential threats, and type of asset in which an asset is a potential target for cyber attacks. *VM* consists of the following sub-elements:

Threat. Is the potential for abuse of an asset that will cause harm in the context of the problem.

Vulnerability. Is a weakness in the system that an attack exploits.

[3]https://capec.mitre.org/news/index.html

[4]https://www.dropbox.com/s/xls46k033ug2o7h/Full20Model.pdf?dl=0

Expression E	Meaning	Denotation $\mathcal{L}(E)$ as set of traces
$skip$	Do nothing	$\{\langle\,\rangle\}$
$E_1\ ;\ E_2$	Sequential occurrence	$\{u^\frown w \mid u \in \mathcal{L}(E_1),\ w \in \mathcal{L}(E_2)\}$
$E_1 \mid E_2$	Alternation (exclusive choice)	$\mathcal{L}(E_1) \cup \mathcal{L}(E_2)$
$opt(E)$	E is optional	$\mathcal{L}(skip \mid E)$
E^+	One or more sequential occurrences of E (iterated concatenation)	$\mathcal{L}(E \mid E;E \mid (E;E);E \mid \ldots)$
G^{succ}	An instance of G starts and terminates in state succeeded	$\bigcup_{g \in G} \mathcal{L}(\langle g.start \rangle; opt(\langle g.susp, g.resm \rangle^+); \langle g.succ \rangle)$
G^{fail}	An instance of G starts and terminates in state failed	$\bigcup_{g \in G} \mathcal{L}(\langle g.start \rangle; opt(\langle g.susp, g.resm \rangle^+); \langle g.fail \rangle)$
$try(G)?E_1 : E_2$	If an instance of G succeeds, an E_1; otherwise, an E_2	$\mathcal{L}(G^{succ};E_1) \cup \mathcal{L}(G^{fail};E_2)$
$E_1 \ \#\ E_2$	Interleaved occurrence of E_1 and E_2	$\{u \odot w \mid u \in \mathcal{L}(E_1),\ w \in \mathcal{L}(E_2)\}$
$E^{\#}$	One or more instances of E occurring concurrently (iterated shuffle)	$\mathcal{L}(E \mid E\#E \mid (E\#E)\#E \mid \ldots)$

Figure 4: Syntax and denotation of goal expressions (Dalpiaz et al., 2013).

Asset. Is anything that has value to an organization, and could be tangible (physical) and intangible (non-physical) in which the target of the attack is concerned.

Attack Pattern. Is defined as generic description of a deliberate, malicious adversary that frequently occurs in a specific context (Moore et al., 2001). It describes the common elements and techniques used in attacks against vulnerable CPS components. These patterns generalize reusable attack knowledge from frequent adversaries in facilitating security requirements analysis for the system-to-be. An attack pattern consists of: *General goal of the attack*, *Antecedent*, *Steps to carry out the attack* and *Consequent*.

Antecedent. Are assumptions about properties of people, processes, technology and infrastructure that are assumed to hold from the attacker's point of view or prerequisite for an adversary to succeed. Antecedents can include the skills, resources, access, or knowledge that the attacker must have, and the level of risk that need to be tolerated (Haley and Laney,).

Consequent. Are consequences such as knowledge exploited by the the attacker and changes to the targeted system that occur as a result of attack step execution, and when the antecedent is fulfilled. Several studies such as (Ali et al., 2010; Li et al., 2015; Li et al., 2014) have been increasingly recognizing the leading role of Goal-oriented Requirements Engineering in modeling and analysis of security, safety and privacy concerns.

For instance, graphical representation of extracted security attack[5]

shows that an adversary captured by *VM Model*. Specifically, *Taking advantage of application future* to recover users forgotten password and use it as the original user (*Threat*) and software application (*Asset*) is target of the attack. This adversary occurred as a result of an attack, shown on the top-left side of the model: 1) *Password Recovery Exploitation* and a vulnerability, shown on the top-right side of the model: 2) *Insufficiently Protected Credentials*. Moreover, the asset (software application) refined in the *cyber realm* could encounter an attack *sequentially* that the first attack occurred (Password Recovery Exploitation) and followed by the second vulnerability *Insufficiently Protected Credentials* or can happen in parallel *(*concurrent) attack.

4.2 Attack-Mechanism Model (AM)

This model captures design strategies, different attack pattern mechanisms. More importantly, it builds attack mechanisms by employing goal models, domain assumptions, attack mechanisms and task operationalization artifacts. Each attack pattern captures knowledge about how specific parts of an attack are designed and executed, providing the adversary's perspective on the problem and the solution, and gives guidance on ways to mitigate the attack's effectiveness. *AM* model depends on *VM* model since it prepares its attacking mechanism based on the threat explored in *VM* model. However, threats can be refined not only in linear but also iteratively.

For example, consider a security solution to use *TLS protocol*. TLS protocol offers *confidentiality, integrity and authentication*. However, TLS further extends its support from other component (CA) to manage certificates and keys (Türpe, 2017). The threat might not directly compromise TLS protocol but can impose adversaries to the certificates authorities (CA) such as exploiting private keys since TLS depends on certificate authorities. Due to this interdependencies, threats that are recognized in one realm can affect further another realm. Attack-Mechanism Model, consists of sub-elements to perform attack strategies, and the elements adopt Goal-oriented principles (Van Lamsweerde, 2001).

[5]https://www.dropbox.com/s/i3qdtsc6vergnro/Runtime\%20Attack\%20Pattern.pdf?dl=0

Goals. A goal is a prescriptive statement of intent to be satisfied by cooperation of the agents forming the system *goal*. The word *system* refers to both the software and its environment, including people, legacy software, devices like sensors and actuators (Van Lamsweerde, 2009). Throughout the model, we use goals instead of anti-goals as anti-goals are considered to be goals from attacker's point of view.
Goal Model. Is an AND/OR-graph showing how goals contribute positively or negatively to each other, and *AND-refinement* captures a combination of sub-goals entailing the parent goal; and *OR-refinement* captures an alternative way of satisfying the parent goal. A goal may be refined into sub-goals by asking *How* questions whereas it may be abstracted into parent goals by asking *Why* questions. **Tasks**: Describe behaviors of the system-to-domain assumption.
Domain Assumptions. Are properties of the domain that are assumed to hold. Normally, assumptions considered as an assertions to check the correctness of goal refinement or operationalization. In general, like goals, assumptions cannot be enforced to be refined (Feather et al., 1998). However, in our model, domain assumptions are refined together with tasks in order to satisfy the root goal as shown in Figure 3, because they are used as antecedents that supports to perform cyber attack, and facilitates to choose type of attack mechanism.

Design-Time Attack Model. As shown in[4],a Goal *Password Recovery Exploitation (attack method)* refined into two tasks that satisfies the attack mechanism: *Understand password recovery mechanism (T1) Find a weakness in the password recovery mechanism and exploit it* (T2). Four domain assumptions refined together with tasks: *(The system allows* users to recover password (DA1) *Password mechanism* has been designed or implemented insecurely (DA2), *Password recovery* mechanism relies only on something the user knows (DA3) and *No third-party intervention* is required to use the password recovery mechanism) to attack the root level goal (DA4).

4.3 Behavioral Model (BM)

Building a complete behavioral model for a very complicated systems like CPS, with many complex and heterogeneous states is often quite challenging problem (Cailliau and van Lamsweerde, 2017). To understand how attacker's fulfill their target by compromising security concerns such as *Confidentiality*, *Integrity*, *Availability* and *Accountability*, one has to analyze how the threat environment is behaving with in the system, how the adversaries can compromise,

```
PROPAGUP(Event ev, GoalInst g)
 1  chnTrace(g) ← chnTrace(g)⌢ev
 2  if state(g) ∈ {succeeded, failed} then return
 3  State newState ← NIL
 4  Annotation ann-g ← annot(type(g))
 5  if ISFINALSTRINGOF(chnTrace(g), ann-g)
 6     then newState ← succeeded  // Rule R1
 7     else if NOTINITSTRINGOF(chnTrace(g), ann-g)
 8             then newState ← failed  // Rule R2
 9  if newState = NIL
10     then switch state(g)
11             case running :
12                  if CANSUSP(state(g), chnTrace(g), ann-g)
13                     then newState ← waiting  // Rule R4
14             case waiting :
15                  if CANRESM(state(g), chnTrace(g), ann-g)
16                     then newState ← running  // Rule R5
17  if newState = NIL then return
18  Event ev' ← GETEVENTBETWEEN(state(g), newState)
19  PROCESSEVENT(ev')
```

Figure 5: Bottom-up Propagation of an Event.

and how the system behaves under multi-stage attacks, and recognize system behaviors to facilitate such analysis. This model captures fundamental system behaviors such as *Sequential*, *Interleaving*, *Multiplicity*, *Alternative* instances of attack goal models, sub-goals, domain assumptions and tasks (mechanisms). *BM* model extends and integrates (Dalpiaz et al., 2013), and annotate runtime behaviors of *VM* and *AM*, thus transforms *design-time attack models to runt time attack models*.
Runtime Attack Model. *BM* model provides annotations of fundamental system behavior, derives *runtime attack models* from *design-time attack models* in order to support attack event monitoring. We illustrate these system behaviors using attack scenarios extracted from the full model[5].

Alternative System Behavior expresses that the system need to satisfy a goal, and depending on the type of obstacle facing, a change in behavior is expected to happen and *Multiple Instance* annotates running system behaviors that could have more than once instance, and the instances can be ordered in sequence *(Sequential)* or they can occur concurrently *(Interleaved)*. The behavioral annotations can be shown in the model[5].
The following example illustrates *BM* using attack scenario example, and the model can be found in[5].
Sequential and Interleaving. As shown in[4]two fundamental system behaviors are annotated. For example, Goals, G1 (*Phishing*) is refined to G2 (*Obtain domain name and certificate to spoof legitimate site*), G3 (*Explore legitimate website*), G4 (*Convince user to inter sensitive information*), and G5 (*Use stolen credentials to log into legitimate site*). Possible behavioral annotation of G1 is an Instance of G2 should be achieved first followed by *Interleaved* fulfillment of G3 and G4 and followed by G5. Formally, *Anno-*

Table 1: Security attack category used in our experiment.

Attack pattern category	Meta-level attack patterns
Privilege Escalation	Cross Zone Scripting
Overflow Buffers	Buffer Overflow via Parameter Expansion
Cross-Site Scripting	XSS through HTTP query String
Buffer Manipulation	Overflow Buffers
Input Data Manipulation	Relative Path Traversal
Communication	Interception
Software	Authentication abuse
Physical security	Bypass electronic card/ access control
Inject unexpected items	Argument Injection
Engage in Deceptive interactions	Phishing
Employee Probabilistic techniques	Brute Force
Subvert access control	Exploitation of Trusted Credentials
Functionality Misuse	Password Recovery Exploitation

tation of G1: (G2; (G3 # G4); G5)

Alternative System Behavior. G3 is refined to three tasks T3 (*Use spidering software to get copy of web pages*), T4 (*manually save copies of required webpages from legitimate site*), T5 (*Create webpages which look legitimate site but different in content*). Possible annotations of G3 can be specified that the system should either T3 (*Use spidering software to get copy of webpages*) or T4 (*Manually save copies of required webpages from legitimate site*). If T3 didn't work out due some obstacles then T5 (*Create webpages which look legitimate site but different in content*) is executed. Formally, *Annotation of G3: (Try (T3) ? Skip : T5) / T4).*

Multiplicity Behavior. For example as it can be seen in[4],T1 (*Obtain domain name that visually looks similar to the legitimate site*), T2 (*Obtain a legitimate SSL certificate for the new domain*) can be repeated multiple times without waiting for each other. Formally, *Annoations of G2: (T1; T2) +).*

4.4 Event Model (EM)

This model captures events derived from behavioral models (BM). These events could be grouped into observable and non- observable events from event-log perspective. Particularly, we focus on observable events. [BMEM] shows security related events, E1 (*Buffer region identified*), E2 (Injection vector identified), E3 (Excessive data sent to the targeted buffer), E4 (*Content injected into the targeted software*). Events *(E1, E2, E3, E4)* are derived from *(T1, T2, T3 and T4)*, thus when the root goal is satisfied.

5 EXPERIMENT AND RESULT

The overall model of the experiment contains 831 elements which consists of *20 attack patterns*, *20 parallel-attack patterns*, *88 Goals and Sub-goals*, *91 Tasks*, *41 Domain assumptions*, *99 Events* and *74 Behavioral annotations* as shown in Table 2.

The experiment has been performed on two types of attack pattern categories (Domain-Based Attack and Mechanizm-Based Attack) as shown in[6].

Table 2: Statistics of elements of the full model.

Elements of the Model	No
Attack patterns	20
Parallel attack patterns	20
Goals and sub-goals	88
Tasks	91
Domain assumptions	42
Behavioral annotations	74
Events	99
Relationships	398
Total	831

Domain-Based Attack. Organizes attack patterns hierarchically based on the attack domain. Attacks such as *Social Engineering*, *Supply Chain*, *Communication*, *Physical Security* and others are grouped under this category.

Mechanizm-Based Attack. Organizes attack patterns hierarchically based on mechanisms that are frequently employed when exploiting a vulnerability. The attack patterns that are members of this category represent the different techniques used to attack a system. They do not, however, represent the consequences or goals of the attacks. For example, *(Collect*

[6]https://www.dropbox.com/s/a22gjrttkrk29mo/Statistics20Runtime20Attack20Pattern.pdf?dl=0

and Analyse Information), *(Inject Unexpected Items)*, *(Engage in Deceptive Interactions)* and others are grouped in this category. Moreover, at the highest level, categories exist to group patterns that share a common characteristics.

Realm-Specific Adversaries. Our models focus on realm-specific adversaries meaning that it spans the three realms of CPS (*Cyber, Physical infrastructure, and including the Social one*). We also analysed the *inter-dependence relationship* among realm-specific attack models. *AM* model depends on *VM* model for achieving realm-specific vulnerabilities, and vulnerabilities captured in *(*VM model) spin off and provide an input to the the next realm *(*AM model) thus select a suitable attack mechanizm by taking an advantage of weaknesses explored in *VM* model.

6 DISCUSSION

We developed a framework for building behavioral models of security attacks and derive security-events for CPS. Our approach enriched and transformed *design-time security attack models* into *runtime attack model*. In our model, events are derived from runtime attack models. In *AM model*, we found that similar attack patterns use the same way of attacking mechanism to achieve a root goal. However, they differ when they use specific type of *task* and *domain assumptions* to achieve the same root level-goal. For example, let's consider two types of attack pattern from the full model[4],one *Password Recovery Exploitation* and the other *Functionality Misuse*. Both types of attacks have the same method of attacking mechanism, using *Brute force* (trial-and-error method), and with the same final goal, *t*o take advantage of the application feature in order to gain access into the system. Particularly, the former one uses a mechanizm: (*Find a weakness in the password recovery*) and exploit it, and assumes (*Password recovery mechanism* relies only on something the user knows while the later one uses a mechanism: *cryptanalysis*, and assumes that attack requires basic *scripting ability* to automate the exploration of the search space. We also recognized that *Behavioral Models (BM)* can be used to facilitate and support *security goal refinement* and *security goal simplification* proposed in (Li and Horkoff, 2014).

Novelty of the Framework. Building runtime behavioural models, and derive security events to support security attack event monitoring.

Threats to Validity. In the experiment section, the objective was to evaluate our approach by building behavioral models for security attacks using attack scenarios taken from[2].The results of our approach are encouraging but we consider them as preliminary, so they need to be further confirmed by performing experiments including a larger set of participants.

Internal Validity. Factors affecting subject performance was very important in the experiment section. We believe that the skill of the subject involved in building *design time* and *runtime* attack models, and the corresponding analysis was appropriate for the objective of our preliminary experiment.

External Validity. Although our security attacks scenarios are complicated, it was only performed by one subject. However, we need to do further experiments using *Asfalia* framework with more attack scenarios, and by including more participants.

7 RELATED WORK

Goal-Based Adaptive Systems. The complexity of software systems is increasing, software engineering researchers and practitioners have made several efforts, and have shown a huge interest in self-adaptive systems. (Silva Souza et al., 2011) has proposed a new class of requirements, called Awareness Requirements, which talk about the runtime status of other requirements. (DeLoach and Miller, 2010) have analyzed the difference between goal classes and instances for adaptive systems. (Dalpiaz et al., 2013) proposed runtime goal model in which they have provided conceptual distinction between design-time goal models and runtime goal model, and a structured pattern for runtime goal model have been derived. (Cailliau and van Lamsweerde, 2017) has proposed an approach of obstacle-driven runtime adaptation techniques to achieve increased satisfaction of probabilistic system goals under the current conditions. *We share similarities in these approaches in terms runtime goal model but our approach applies to attack scenarios and aimed on building behavioral models for attack strategies*.

Security Attack Pattern. Research in security pattern has made tremendous progress in collecting and organizing security patterns however, their applicability is very less. (Moore et al., 2001) have introduced attack patterns to encapsulate utilizable attack knowledge from multiple attack occurrences in order to support security requirement analysis. Other approach that model security patterns as goal models such as (Mouratidis et al., 2006) extended Secure Tropos using security patterns.

Attack Scenarios. Several proposals have made efforts to analyse and model attack scenario, whereby steps have been described to explore how attackers perform an attack. In the literature there are differ-

ent proposal that have focused in providing ways of representing attack scenarios, for example attack trees (Morais et al., 2013) and graphs (Phillips and Swiler, 1998) have been used as an approach to capture attack scenarios by different researchers. *In contrast with our approach, these efforts are somehow limited in terms of specifying how attack strategies are conducted*.

Further works have also been studied to analyse and model threats such as STRIDE (Shostack, 2014), however it fails to model attacker's intentions and are somehow limited in addressing multi-stage attacks. In contrast to this work, several proposals have been explicitly addressed to capture the rationale behind attacker actions using anti-goals such as (Van Lamsweerde, 2004). Similar approaches that goes beyond (Van Lamsweerde, 2004), (Li et al., 2015) have proposed to analyse security analysis from an attacker's perspective and detailed adversary's malicious intentions. *Our approach shares similarities with these approaches in terms of attack models, anti-goal refinement and the way how attack scenarios are constructed. However, our method goes beyond this by studying behavioural models of security attack scenarios and applied for constructing runtime attack models*.

8 CONCLUSION

Security solution for Cyber Physical-Systems should be adaptive, and needs a systematic approach to support monitoring of security attack events. We proposed *Asfalia*, a framework for modeling, analysing and monitoring of security attacks (events) where attack strategies are analysed in a realm-specific. The model captures security attack events and their corresponding behaviors. The framework has been evaluated using extensive security attack scenarios. Experiments have been conducted on two categories of attack pattern, namely *domain-based* and *mechanizm-based* attacks. In summary, *VM* model captures the adversaries, vulnerabilities, and analyse the target of the attack in terms of realm-specific approach. *AM* model constructs attack model based on the adversaries specified in *VM* model. *BM* model annotates system behaviors for *VM and AM* models and finally in *EM* models, events are derived from behavioral models.

We are in the process of developing a prototype tool that reflects *Asfalia* framework along with extensions and improvements necessary for thorough evaluation. The results of the preliminary experiments indicate that one of the limitations of the current framework is the requirement for almost an expert level understanding of modeling and security by the users. Further experiments will also include different combinations of users, those that know modeling, but very little security and vice versa.

REFERENCES

Ali, R., Dalpiaz, F., and Giorgini, P. (2010). A goal-based framework for contextual requirements modeling and analysis. *Requir. Eng.*, 15(4):439–458.

Baheti, R. and Gill, H. (2011). Cyber-physical systems. *The impact of control technology*, 12(1):161–166.

Banerjee, A., Venkatasubramanian, K. K., Mukherjee, T., and Gupta, S. K. S. (2012). Ensuring safety, security, and sustainability of mission-critical cyber–physical systems. *Proceedings of the IEEE*, 100(1):283–299.

Boyes, H., Hallaq, B., Cunningham, J., and Watson, T. (2018). The industrial internet of things (iiot): An analysis framework.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., and Mylopoulos, J. (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236.

Buchs, A. et al. (2013). Compte rendu de lecture: Rifkin jeremy, 2011. the third industrial revolution: How lateral power is transforming energy, the economy, and the world. palgrave macmillan, 2011, 304 p. Technical report.

Cailliau, A. and van Lamsweerde, A. (2017). Runtime monitoring and resolution of probabilistic obstacles to system goals. In *Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2017 IEEE/ACM 12th International Symposium on*, pages 1–11. IEEE.

Conway, J. (2016). The industrial internet of things: an evolution to a smart manufacturing enterprise. *Schneider Electric*.

Dalpiaz, F., Borgida, A., Horkoff, J., and Mylopoulos, J. (2013). Runtime goal models: Keynote. In *Research Challenges in Information Science (RCIS), 2013 IEEE Seventh International Conference on*, pages 1–11. IEEE.

DeLoach, S. A. and Miller, M. (2010). A goal model for adaptive complex systems. *International Journal of Computational Intelligence: Theory and Practice*, 5(2):83–92.

Feather, M. S., Fickas, S., Van Lamsweerde, A., and Ponsard, C. (1998). Reconciling system requirements and runtime behavior. In *Proceedings of the 9th international workshop on Software specification and design*, page 50. IEEE Computer Society.

Fernández, E. B., Fonoage, M., VanHilst, M., and Marta, M. (2008). The secure three-tier architecture pattern. *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 555–560.

Griffor, E. R., Greer, C., Wollman, D. A., Burns, M. J., et al. (2017). Framework for cyber-physical systems: Volume 1, overview.

Haley, C., Laney, R., Moffett, J., and Nuseibeh, B. (2008). Security requirements engineering: A framework for representation and analysis. *IEEE Transactions on Software Engineering*, 34(1):133–153.

Haley, C. B. and Laney, R. C. Using trust assumptions in security requirements engineering. Citeseer.

Helmiö, P. et al. (2017). Open source in industrial internet of things: A systematic literature review.

Henzinger, T., Lee, E. A., Sangiovanni-Vincentelli, A., Sastry, S., Aiken, A., Auslander, D., Bajcsy, R., Hedrick, K., Keutzer, K., Necula, G., et al. (2008). Center for hybrid and embedded software systems.

Hu, F., Lu, Y., Vasilakos, A. V., Hao, Q., Ma, R., Patil, Y., Zhang, T., Lu, J., Li, X., and Xiong, N. N. (2016). Robust cyber–physical systems: Concept, models, and implementation. *Future generation computer systems*, 56:449–475.

Jeschke, S., Brecher, C., Meisen, T., Özdemir, D., and Eschert, T. (2017). Industrial internet of things and cyber manufacturing systems. In *Industrial internet of things*, pages 3–19. Springer.

Li, T. and Horkoff, J. (2014). Dealing with security requirements for socio-technical systems: Holistic approach. In *Advanced Information Systems Engineering - 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, pages 285–300.

Li, T., Horkoff, J., and Mylopoulos, J. (2014). Integrating security patterns with security requirements analysis using contextual goal models. In *The Practice of Enterprise Modeling - 7th IFIP WG 8.1 Working Conference, PoEM 2014, Manchester, UK, November 12-13, 2014. Proceedings*, pages 208–223.

Li, T., Horkoff, J., and Mylopoulos, J. (2018). *Software & Systems Modeling*, 17(4):1253–1285.

Li, T., Horkoff, J., Paja, E., Beckers, K., and Mylopoulos, J. (2015). Analyzing attack strategies through anti-goal refinement. In *IFIP Working Conference on The Practice of Enterprise Modeling*, pages 75–90. Springer.

Moore, A. P., Ellison, R. J., and Linger, R. C. (2001). Attack modeling for information security and survivability. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Morais, A., Hwang, I., Cavalli, A., and Martins, E. (2013). Generating attack scenarios for the system security validation. *Networking science*, 2(3-4):69–80.

Mouratidis, H., Weiss, M., and Giorgini, P. (2006). Modeling secure systems using an agent-oriented approach and security patterns. *International Journal of Software Engineering and Knowledge Engineering*, 16(03):471–498.

Müller, H., Litoiu, M., and Mylopoulos, J. (2016). Engineering cybersecurity in cyber physical systems. In *Proceedings of the 26th Annual International Conference on Computer Science and Software Engineering*, pages 316–320. IBM Corp.

Phillips, C. and Swiler, L. P. (1998). A graph-based system for network-vulnerability analysis. In *Proceedings of the 1998 workshop on New security paradigms*, pages 71–79. ACM.

Poovendran, R. (2010). Cyber–physical systems: Close encounters between two parallel worlds [point of view]. *Proceedings of the IEEE*, 98(8):1363–1366.

Poulsen, K. (2003). Slammer worm crashed ohio nuke plant network. *http://www. securityfocus. com/news/6767*.

Shafi, Q. (2012). Cyber physical systems security: A brief survey. In *2012 12th International Conference on Computational Science and Its Applications*, pages 146–150. IEEE.

Shostack, A. (2014). *Threat modeling: Designing for security*. John Wiley & Sons.

Silva Souza, V. E., Lapouchnian, A., Robinson, W. N., and Mylopoulos, J. (2011). Awareness requirements for adaptive systems. In *Proceedings of the 6th international symposium on Software engineering for adaptive and self-managing systems*, pages 60–69. ACM.

Souza, S. and Mylopoulos, J. (2012). Requirements-based software system adaptation.

Turk, R. J. (2005). Cyber incidents involving control systems. Technical report, Idaho National Lab.(INL), Idaho Falls, ID (United States).

Türpe, S. (2017). The trouble with security requirements. In *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, pages 122–133. IEEE.

Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, pages 249–262. IEEE.

Van Lamsweerde, A. (2004). Elaborating security requirements by construction of intentional anti-models. In *Proceedings of the 26th International Conference on Software Engineering*, pages 148–157. IEEE Computer Society.

Van Lamsweerde, A. (2009). *Requirements engineering: From system goals to UML models to software*, volume 10.

Wang, E. K., Ye, Y., Xu, X., Yiu, S.-M., Hui, L. C. K., and Chow, K.-P. (2010). Security issues and challenges for cyber physical system. In *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, pages 733–738. IEEE Computer Society.