Quantitative security and safety analysis with attack-fault trees

Rajesh Kumar and Mariëlle Stoelinga

University of Twente, the Netherlands

Email: r.kumar@utwente.nl, m.i.a.stoelinga@utwente.nl

Abstract—Cyber physical systems, like power plants, medical devices and data centers have to meet high standards, both in terms of safety (i.e. absence of unintentional failures) and security (i.e. no disruptions due to malicious attacks).

This paper presents *attack fault trees* (AFTs), a formalism that marries fault trees (safety) and attack trees (security). We equip AFTs with stochastic model checking techniques, enabling a rich plethora of qualitative and quantitative analyses. Qualitative metrics pinpoint to root causes of the system failure, while quantitative metrics concern the likelihood, cost, and impact of a disruption. Examples are: (1) the most likely attack path; (2) the most costly system failure; (3) the expected impact of an attack. Each of these metrics can be constrained, i.e., we can provide the most likely disruption within time t and/or budget B . Finally, we can use sensitivity analysis to find the attack step that has the most influence on a given metric. We demonstrate our approach through three realistic cases studies.

I. INTRODUCTION

Safety and security are heavily intertwined: measures that increase safety often decrease security and vice versa: Implementing access policies may deter adversaries, but act as a hindrance in case of emergency operations; a non encrypted industrial communication is a simpler design choice, however it can be exploited by attackers to spoof the network. Thus, given their mutual dependence, safety and security should be considered in a combination, so that tradeoffs can be made.

In recent years, several methods for risk identification, evaluation and analysis eliciting both safety hazards and security threats have been proposed such as SAHARA [1], CHASSIS [2], FMVEA [3]. A majority of them, are application specific models (automotive, power) and geared towards the risk identification phase. A few attempts have been undertaken to integrate the attack trees and fault trees, leading to extended fault trees [4], component fault trees [5] and FACT graphs [6]. However, as highlighted in [7], these models are static i.e. none of them models the propagation of *disruption* (i.e. accidental and/or the malicious failures) over *time*. Also, existing approaches do not support realistic cost structures or shared subtrees. Finally, little attention is given to quantify the impact of disruption inflicted by the different adversaries.

A. Our approach.

In this paper, we propose *attack-fault trees* (AFTs) as formalism for the combined safety-security analysis. As their name suggests, AFTs overarch both attack trees and fault trees. Fault trees (FTs) are a classical formalism for reliability engineering that is heavily used in industry [8], [9]. Attack

trees (ATs) were coined by Schneier in the 90s as a security analogue for fault trees [10], and have gained recent popularity, for instance as part of the UMLsec [11] and SysMLsec modelling languages [12].

While attack trees and fault trees are similar in nature, they feature several important differences. Both attack trees and fault trees model how respectively basic attack steps and component failures, modelled in the leaves of the tree, propagate through the system; and propagation happens via gates. ATs and FTs differ in the type of goal/gate nodes, and therefore in their analysis algorithms [9], [13]. Furthermore, security risks heavily depend on risk appetite, resources and the capabilities possessed by an attacker. For example, a malicious insider owning a better access and the knowledge of the vulnerabilities in an enterprise is a significant threat as compared to an opportunistic burglar. To get insight on which preventive measures can potentially dissuade each of these personas, various recent approaches include adversary attributes in attack tree analysis [14], [15].

The attack-fault trees we propose support all the syntactic constructs and the leaf behaviour from attack trees [13], [17] and dynamic fault trees [8], [9]. As such, they can very well capture the multistage and the dynamic temporal and causal safety and security interdependencies. An important contribution of this paper is the quantitative analysis of AFTs. Quantitative safety and security analysis, is pivotal for effective safety-security management, both to determine the most riskful scenarios and to select the most effective countermeasures. While quantitative fault tree analysis is a common practice [9], quantitative security is advocated in industry only recently [18], [19]. We consider multiple quantitative annotations on the AFTs like cost, time, failure probabilities and damage which can functionally be dependent on each other. Based on these annotations, we quantify several safety-security scenarios:

- *As-is scenarios*: Given an AFT, and a quantity of interest, we can compute its value: What is the probability, costs, or the skill level needed by an attacker persona to eventually disrupt or likely to disrupt the system within a certain mission time? How much damage does it inflict on the organization?
- *What-if scenarios*: How are the disruption values affected if the detection measures are implemented? What is the difference in the estimated disruption values if a more

skilled threat agent acts maliciously against the system?

- *Design alternatives*: Which design choices and preventive measures can lead to a more safe and secure system?

Technically, our approach is based on (SMC) [20], [21], a state-of-the-art method for stochastic analysis. SMC has been deployed in a wide number of areas, communication theory [22], systems biology [23], and many more [24]. Key feature of SMC technique is its compositionality, allowing one to build large models by composing smaller ones.

In this paper, we exploit the compositional SMC approach and translate each element (i.e., leaf or gate) of an AFT into a stochastic timed automaton (STA, [25]). STAs are a powerful and flexible formalism that supports many features: they are state-transition diagrams supporting different discrete and continuous probability distributions, hard and soft time constraints, cost variables etc. Powerful tool support is provided by the model checker Uppaal SMC [26]. Thus, our models are modular yielding a significant benefit in terms of comprehensibility. Additionally, SMC allows simulation of complex systems where a simple closed-form solution does not exist or a rigorous state space search is infeasible, as in our case studies.

Further, we provide an adversary-based pruning of AFT by introducing *attacker profiles*. It allows us to perform the analysis for different combinations of attacker profiles and system configurations, resulting in a comprehensive risk analysis. We choose several distinct safety security scenarios as case studies in the section VI illustrating our modelling approach and each conveying the lessons learnt through its quantitative analysis.

B. Related work.

Traditionally, risk standards have exclusively focussed on either safety – Failure Mode, Effect and Criticality Analysis (FMECA, [27]), and the SAE standardized language AADL with its error annex [28] or on security aspects – FAIR [29] and UML-based CORAS [30]. However, a need for an integrated safety and security analysis is also widely acknowledged, by the standards like ISO 26262 [31] and IEC 64443 [32] and in the European research projects [33], [34]. An excellent overview of the recent progress on the integration of safety and security can be found in [35].

Initial attempts combining fault and attack trees have appeared in [4], [5], [36]. However, many of these works focus on qualitative aspects, or handle fewer risk metrics than we do, or support only independent quantitative annotations of attributes on the leaves. Related to AFTs are Boolean-driven Markov processes (BDMPs, [37], [38]). However, there are several important differences: whereas our approach is compositional, BDMPs are monolithic, i.e. one single Markov model is constructed for each BDMP. This makes the BDMP formalism harder to extend with important features, like complex maintenance strategies, and attacker profiles. Further, cost structures are not considered. Also, triggers and phase clocks make the BDMP formalism cluttered. Another model based approach to study safety-security interactions are through

Petri-nets [39] and stochastic activity networks [40]. Both these modelling frameworks are powerful, and exhibit concurrency and synchronization characteristics. Earlier approaches deploying stochastic model checking to fault trees or attack trees appeared in [41]–[43].

II. ATTACK FAULT TREES

Attack-fault trees (AFTs) model how a top-level (safety or security) goal can be refined into smaller sub-goals, until no further refinement is possible. In that case, we arrive at the leaves of the tree that model either the basic component failures (BCF), the basic attack steps (BAS) or on demand instant failures (IFAIL). Since subtrees can be shared, AFTs are directed acyclic graphs, rather than trees.

Fig. 1. Standard and dynamic fault tree gates: AND, OR, VOT(k)/ n , PAND, FDEP and SPARE gate

AFT leaves. We take a standard approach from fault tree analysis [8], [9] and assume that all BCFs are governed by governed by exponential probability distributions, i.e. the probability of a disruption occurring before time t is given as: $P(t) = 1 - e^{-\lambda t}$ where λ is the rate of exponential distribution. Further, BCFs are enriched with cost structures such as damage.

Fig. 2. Attack tree gates: AND, OR, and SAND gate

BAS on the other hand, represent the active steps taken by an attacker to compromise the system. They are equipped with an exponential distribution representing the attack duration, discrete probabilities quantifying the attack success irrespective of the execution time and a rich cost structure that includes the cost incurred by an attacker and the damage inflicted on the organization. Though we use exponential distributions, our methodology can handle other complex distributions such as phase-type distributions. Here, we use exponential distribution as they are relatively easy to handle since their shape is completely defined by one parameter and is tractable. We characterize each BAS by a control strength (CS). The concept of CS is taken from [29] and indicates the difficulty level of the attack step. Here, we assign the CS of each BAS to an ordinal value in {low, medium, high}. e.g. we can label the CS of an atomic attack step of break-in through a tamper resistant window as *high* as compared to break-in through a glass window whose CS can be stated as *low*. An IFAIL leaf is used to model an on demand probabilistic failure of a component, typically electromechanical components start when needed (on demand) [44].

AFT gates. Complex multi-step disruption scenarios are modelled by a composition of multiple BAS, BCF and IFAIL through smart exploitation of gates — AND, OR, SAND, VOT(k)/ n , PAND, FDEP and SPARE taken from both dynamic fault trees (see figure 1 for the standard and dynamic fault tree gates) and attack trees (see Figure 2 for the supported attack tree gates).

In order to disrupt an AND gate, all its children needs to be disrupted. In this way, we can model a power outage when both the primary power supply and the backup supply are disrupted. Similarly, an OR gate is disrupted if either of its child is disrupted, and the VOT(k)/ n propagates a disruption if k of its n children are disrupted. The sequential AND (SAND) gate propagates a disruption when its children are disrupted from left to right with a restriction that the success of the preceding step determines whether successive step should be started. For example, installing a malware requires three sequential steps: creation of the malware, sending it across the network and then waiting for an user to run it. In contrast to a SAND gate, in a PAND gate all children are operational at the system start, however the disruption occurs only, if all events are executed from left to right. For example, an attacker can gain opportunistic access only if there is a transient fault in the signalling component (e.g. security camera) before.

Fig. 3. AFT for Example 1

Fig. 4. AFT for Example 2

The SPARE gate consists of a primary input and one or more spare input. At system start, the primary is active, and the spares are in standby mode. When the primary input fails, one of the spare inputs is activated and replaces the primary. If no more spares are available, the SPARE gate is disrupted. The FDEP (functional dependency) gate consists of a trigger event and several dependent events. When the trigger event occurs, all the dependent events fail, e.g. the disruption of the back up power immediately disrupts the security controls such as the security cameras and the alarms.

Examples. We show our approach on two distinct safety-security scenarios from [38] (Example 1) and [37] (Example 2).

Example 1: The AFT depicted in figure 3 models an interplay of malicious and accidental failures resulting in the top event, i.e. Pollution. The system consists of a pipeline carrying toxic substance which is monitored through a control device namely SiS (Safety instrumented system). The system is *disrupted* if there is a spillage due to first a failure of SiS and then a pipe breakdown (modelled through PAND) or a hybrid failure. A hybrid failure can occur only if there is first a pipe breakdown and then a failure of stand-by SiS (modelled through SAND). Both Pipe and SiS can be *disrupted* maliciously as well as accidentally. The parameters for the malicious events with their control strength is given in Table I. The parameters for the BCF and IFAIL are provided in figure 3. Note that the parameters for detection and the execution rate are directly taken from [38]. Other parameters like costs, damage and CS are arbitrarily chosen to illustrate the analysis method.

Here, $\lambda_{s/ND}$ is the rate of undetected execution, $\gamma_{D(I)}$ is the initial detection probability, $\lambda_{D/(O)}$ represents an ongoing detection, λ is the accidental failure rate and γ is an on demand failure probability.

BAS	CS	Cost to perform malicious acts (in US\$)	Parameters for detection and the execution rate
SiS disabled maliciously	high	Fixed cost= 1500 Variable cost = 5 damage= 0	$\lambda_{s/ND} = 4.166 \times 10^{-2}$ $\gamma_{D(I)} = 0.5$, $\lambda_{D/(O)} = 5.952 \times 10^{-3}$, Once detected: $\lambda_{s/d} = 1.377 \times 10^{-3}$
Pipe broken maliciously	low	Fixed cost= 500 Variable cost = 10 damage = 1000	$\lambda_{s/ND} = 5.952 \times 10^{-3}$ $\gamma_{D(I)} = 0.1$ $\lambda_{D/(O)} = 5.952 \times 10^{-3}$ Once detected: 0 (stop)

TABLE I
ANNOTATING MALICIOUS LEAVES IN FIGURE 3

Example 2: The AFT depicted in Figure 4, models a classical example of safety and security antagonism in the design of an emergency exit door. Here, the *design choice* is to be made between keeping the emergency exit door either always locked or always unlocked. While from the safety perspective it is important that the door remains unlocked, from the security perspective, an open door is a security vulnerability which could be exploited by an attacker. From the perspective of safety, the top event: integrity of person may be affected if there is a fire and the door remains locked. From the security point of view, first the attack needs to be initiated and then access must be obtained via breaking the door (either walk-in if the door is unlocked or force open the door). Note we model the status of the door with two IFAIL leaves — *Door locked* and *Door unlocked*, with a restriction that if the door is in a locked position, the probability of the door being unlocked is 0.

Attacker profiles. In order to assess how different threat agents influence the disruption values, we introduce *attacker profiles*. An attacker profile ($\mathcal{AP}$) quantifies the resource prerequisites and the ability of an attacker persona to launch an attack. Formally, we express these prerequisites as TC (*threat capability*) [29] which is a logical combination of propositional and real valued attributes (budget, skills, resources). It

is then mapped to a single ordinal value in $\{\text{low, medium, high}\}$. We take a constraint based approach where we assume that an attacker can choose non deterministically the attack steps to attempt, satisfying $TC \geq CS$. Such lookup tables for attacker persona are popular in TARA specifications [45] and can be devised based on stakeholders interest and expert opinion.

Attacker persona	Role	Attributes	TC
Ervin	Malicious insider	$(\text{Budget} \leq \text{US\$ } 5000) \wedge (\text{Skill} = \text{high}) \wedge (\text{Right Equipments}) \wedge (\text{Initial access}) \wedge (\text{Risk appetite} = \text{low})$	high
Ethan	Burglar	$(\text{Budget} \leq 5000) \wedge (\text{Skill} = \text{low}) \wedge (\text{No Right Equipments}) \wedge (\text{No initial access}) \wedge (\text{Risk appetite} = \text{high})$	low

TABLE II
ATTACKER PROFILE WITH THREAT CAPABILITY

For our analysis, we consider two attacker personas: Ervin and Ethan. Based on their TC in Table II and CS of the malicious leaves in Table I, we note that Ervin can attempt both malicious leaves in the AFT (in Figure 3) while Ethan can attempt only one malicious leaf ‘Pipe broken maliciously’.

III. STOCHASTIC TIMED AUTOMATA

Stochastic timed automata (STA) is an extension of timed automata with stochastic semantics. Here, the constraints on the edges and the invariants on the locations are used to enable or force certain transitions at certain times. These constraints and invariants are specified as clocks, which increase linearly over time, but may be reset when a transition is taken.

Consider the two tree nodes 1) BAS and 2) top_event. The stochastic timed automata (STA) representing a BAS is shown in the Figure 5 and consists of the location $\{\text{Initial, Wait, potentially_undetected, potentially_detected, ongoing, activated, execution, wait_success, success, fail, stop}\}$ and two types of transitions: *Time delays* governed by the probability distributions (here put as invariant (lamda and lambda1) over the locations $\{\text{activated, ongoing}\}$ respectively). and *probabilistic* where weights like w1 and w2 are specified over the dotted edges to specify the probability distributions of the discrete transitions.

The STA representing the top_event node (shown in Figure 6), consists of the locations – $\{\text{Initial, waiting_disrupt, Top}\}$. It initializes the system by emitting a broadcast signal (activate[id]!) and then waits for a broadcast signal disrupt[id]? from its child node. After receiving that signal, it makes a transition to the ‘Top’ location, which indicates the disruption of the AFT. Here, we use a clock x_{top} that keeps track of the global time.

By translating each element of the AFT into an equivalent STA and composing them together iteratively using proper broadcast signals, we obtain a network of STAs (NSTA), see Figure 7. The resulting NSTA is then used to perform model checking, i.e. we verify

Fig. 6. STA for top_event.

Fig. 5. STA for the template of a BAS. Here id is a unique identifier for the BAS and x is a clock to track the duration of BAS. $costs$ is a global variable to keep track of all the accumulated costs, $costs'$ represents the variable costs per time unit spent in the location and $damage$ is a global variable to keep track of the accumulated damages. f , v and d are suitable constant values.

Fig. 7. Graphical overview of compositional aggregation for AFT models.

Fig. 8. STA for BCF and IFAIL leaf.

the satisfiability of the safety-security metrics formalized as queries (as in section V) over the resulting NSTA.

Here, we use UPPAAL SMC [25] as the tool to perform the statistical model checking (SMC). Uppaal has an inbuilt graphical editor to construct and check the structural integrity of the models. Its inbuilt simulator and query engine can be used to obtain and visualize the probability distributions, evolution of the number of runs with timed bounds and compute the expected values.

IV. TRANSLATION OF ATTACK-FAULT TREES TO STOCHASTIC TIMED AUTOMATA

In this section, we provide an unambiguous semantics by constructing a STA template for each element of the AFT. Each construct is then identified with a unique id and is passed different input parameters to its template.

Fig. 9. STA representation of an AND gate. Here we name the children of the gate as A and B.

Basic attack step (BAS). A BAS as shown in Figure 5, models an atomic malicious step taken by an $\mathcal{AP}$. It is activated on receiving a $\text{activate}[id]?$ signal from its parent node. Thereafter, an attack can go either undetected (with a probability of $\frac{w1}{w2+w1}$) or be detected (with a probability of $\frac{w2}{w1+w2}$). The execution of the attack step requires an investment by an attacker in terms of fixed costs (f) and time (execution time is assumed as exponentially distributed with a mean of λ). During the execution of BAS, an attacker also incurs variable costs (v) that vary linearly with the time spent executing the attack in the location. Further, we consider

Fig. 10. STA representation of a SAND gate. Here we name the children of the gate as A and B.

the possibility that an attack which is undetected may get detected later in the future (modelled with ongoing detection rate given by λ_1). If the attack is detected at any stage, an attacker has to abort the attack, otherwise it is either successful with a probability of $\frac{p}{p+q}$ or fails with a probability of $\frac{q}{p+q}$. If the BAS is successful, it inflicts a damage d and informs its parent nodes by sending a $\text{disrupt}[\text{id}]!$ signal.

Basic Component Failure (BCF). The STA corresponding to the BCF as shown in Figure 8(a), models an accidental/random failure of a component. Initially, it waits in the left most state to be activated from the parent node. After activation, the component can fail with a failure rate given by λ_4 . The failing of the BCF results in a damage of d .

IFAIL. The IFAIL leaf shown in figure 8(b) is used to model an instantaneous disruption likely to arise with a probability of $\frac{p_1}{p_1+p_2}$.

Fig. 11. STA representation of a PAND gate. Here we name the children of the gate as A and B.

Parallel and sequential AND gate model. Both the parallel AND and the sequential AND (SAND) models a conjunctive composition of its leaves/ sub-trees. An AND gate upon activation (see figure 9) waits for a $\text{disrupt}[\text{id}]?$ signal. After receiving it from all of its children, it emits a $\text{disrupt}[\text{id}]!$ signal to its parent node. In contrast to an AND gate, a SAND gate (see figure 10) initially activates only its leftmost child. The disruption of this child leads to the activation of the second leftmost child. The disruption of the rightmost child indicates a disruption of the gate. The behaviour of the SPARE gate is similar to a SAND gate.

PAND gate and FDEP gate model. Unlike a SAND gate, a PAND after being activated gate from its parent node immediately activates both of its children, see Figure 11. It sends a $\text{disrupt}[\text{id}]!$ signal to its parent node when its children are compromised in an order from the left to the right.

A FDEP gate (see Figure 12) consists of a trigger event T and one (or several) dependent event. Once it gets

Fig. 12. STA representation of a FDEP gate. Here we name the trigger as T and the child of the gate as A.

activated on receiving $\text{activate}[\text{id}]?$, it listens for the disruption of the trigger. When it receives a $\text{disrupt}[\text{id}]?$, it sends a $\text{disrupt A}[\text{id}]!$ signal to its parent node that indicates the disruption of its dependent event.

OR gate model. An OR gate (see Figure 13) represents a disjunctive composition of its sub-trees. On activation, the gate enables both its child nodes A and B. As soon as one of its children is disrupted, the gate sends a $\text{disrupt}[\text{id}]!$ signal to its parent node.

Fig. 13. STA representation of a OR gate. Here we name the children of the gate as A and B.

- *as-is analysis:*

- *Probability of disruption:* Suppose we have a random variable $X(t)$. We say that $X_{AFT}^{AP}(t) = 1$ if the top event of the AFT is reached within time t and 0 if it is not. Then, the probability that the AFT is disrupted is given as i.e $P(X_{AFT}^{AP}(t) = 1)$.
- *Expected cost of a malicious disruption:* As our AFTs include cost structures, we can compute the expected cost for an AP to reach the top node of the AFT successfully. Formally, this is given as $\frac{E(Cost(t) \times X_{AFT}^{AP}(t))}{P(X_{AFT}^{AP}(t)=1)}$ where $Cost(t)$ is the accumulated costs incurred upto time 't' in reaching the top event of AFT. Similarly, we can compute the expected damage to reach the goal successfully.
- *Mean time to malicious disruption:* The mean time to attack is given as $\frac{E(W \times X_{AFT}^{AP}(t))}{P(X_{AFT}^{AP}(t)=1)}$ where W denotes the accumulated time in reaching the goal.

- *what if analysis:*

- *Constrained value:* We obtain the constrained values by varying the bound of one attribute over the other attribute (e.g. disruption values under certain time/ budget constraints).
- *Role of Adversaries:* We use different attacker personas, each time with a different combination of attributes to obtain the disruption values.
- *design alternatives:* By disabling different subtrees/ leaves and observing the percentage difference in the probability of disruption of the top event i.e $S = \frac{\partial P_{goal}}{\partial P_{leaf}}$, we can compare the different safety/ security design choices.

Uppaal SMC queries. We use the property specification language WMTL [46] to translate the safety-security metrics into Uppaal SMC queries. If we indicate the goal state in the STA of top_event as Top, the probability of a successful disruption within time t can be written as $\mathbb{P}[x_{top} \leq t](\langle \rangle top_event.Top)$ where $\langle \rangle$ is the existential operator and x_{top} is a clock in the STA to track the global time. In order to quantify the disruption only attributed to the malicious scenarios or obtain the disruption values for different adversaries, we disable different subtrees and re-run the simulations. The expected costs for a successful disruption can be obtained as $\frac{E[B;N](\max_{costs \times top_event.Top})}{\mathbb{P}(t=B)}$ where $top_event.Top = 1$ if the goal is reached and 0 otherwise. Here, $costs$ is the accumulated costs within the time bound B and N is the number of simulations. Similarly, we can obtain the expected time and the expected damage within a certain bound. In order to obtain the constrained values such as probability of the disruption within time t and within costs C , we run the bounded query $\mathbb{P}[time \leq t](\langle \rangle top_event.Top \wedge costs \leq C)$. We can perform similar, constrained queries to obtain the probability

V. SAFETY/ SECURITY METRICS ON AFT

In this section, we formalize the safety/ security metrics and translate it into the Uppaal SMC queries.

of a successful disruption with a damage D .

VI. CASE STUDIES

We demonstrate our approach through three well-known case studies taken from [38] and [37]. Our models produce the same result over the few existing metrics in these paper, thus validating our models. Moreover, we compute several additional risk metrics.

A. Case-study A: Pipeline spill

Fig. 14. Comparison of as-is and what-if scenarios.

Fig. 15. Time dynamic behaviour of Ervin

The objective of this case (refer example 1) is two fold: 1) To highlight the role of adversaries in a safety-security model; 2) To compare the *as-is scenarios* (no detection measures implemented) and the *what-if scenarios* (when detection measures are implemented). The data values are provided in Table I and in Figure 3.

Figure 14 succinctly captures the distinction between the *as-is scenarios* and the *what if scenarios* taking different adversaries, Ethan and Ervin into account. It shows that for *as-is scenarios* (Figure 14(a)), Ervin has a very high probability (0.98) of causing a disruption in a mission time of 350 days as compared to Ethan whose probability of successful disruption is 0.25 in the same time period. This is expected, as Ervin has a high TC and can execute more BAS as compared to Ethan. Further, we see that Ervin takes around 7 days as mean time to disrupt while incurring an expected cost of 3157 US\$ and inflicting an expected damage of 1521 US\$. Ethan on the other hand has to incur an average cost of 4687 US\$ and takes a mean time of 146 days to successfully cause a disruption inflicting damage of 1042 US\$. With the detection measures implemented (see Figure 14(b)), the probability of disruption decreases while the expected days, the expected costs and the expected damage increases for a successful disruption, in line with our intuition.

Figure 15(a), shows a Pareto frontier for Ervin, showcasing a tradeoff for him in expending costs and time. Here, for

example, we see that in order to disrupt the system with a probability of 0.5, he can spend an amount of 10500 US\$ while completing the attack in 100 days or just spend 3750 US\$ while completing the attack in 330 days. We observe that more resources are required (both time and costs) if he wants to disrupt the system with a probability of 0.7, which is consistent with our intuition. In Figure 15(b), we perform a cost-benefit analysis to identify those malicious events that fetch Ervin the more dividends (i.e. lower incurred costs with a higher probability of disruption). Interestingly, we see that Ervin has a higher chances of disruption with a lower incurred cost if he executes only ‘SiS disabled maliciously’ than performing both the malicious events. This is not surprising, as we have assumed that Ervin can disrupt the system by executing the leaf ‘SiS disabled maliciously’ even after getting detected, whereas he has to abort his attack if he gets detected during the execution of ‘Pipe broken maliciously’ (see Table I).

The different scenarios examined in the case study are useful to reason about the potential precursors of disruption (accidental/ malicious) and quantify its impact (expected damage). By stepping into the shoes of an adversary and knowing what is economically and temporally feasible (expected costs and time) for them, a risk manager can take informed decisions on security-hardening measures and prioritize investments over the most vulnerable components.

B. Case-study B: Emergency exit door

	Integrity affected		Accidental Scenario		Attack scenario	
	30 (days)	100 (days)	30 (days)	100 (days)	30 (days)	100 (days)
Door locked	0.015	0.044	0.0058	0.0016	0.015	0.042
Door unlocked	0.15	0.42	0	0	0.15	0.42

TABLE III
PROBABILITY OF DISRUPTION WITH TIME

The objective of this case study (refer example 2) is to analyse the different design choices when safety and security requirements are in contradiction with each other. Here, a design choice is to be made in keeping the emergency exit door either always locked or always unlocked.

In Table III, we see there is a significant reduction in the probability of attack if the door is locked (around 90% in a mission time of 30 days) in comparison to a scenario when the door is unlocked. If the door is unlocked the probability of accidental disruption is 0 as the person can escape the accidental event of fire easily. We also see that the probability of the top event is significantly higher (0.42 when the door is unlocked to 0.044 when door is locked, both in a mission time of 100 days).

Thus, we conclude that the security breaches are more likely than the accidental events. From this elementary case-study, we learn how an early analysis of the design alternatives can be helpful in ensuring a system is both safe and secure even before its deployment.

C. Case-study C: Oil Pipeline

The objective of this case study (taken from [37]) is to represent and analyse a complex industrial safety critical

scenario using AFTs. Here, we compare different disruption scenarios, namely: (1) only accidental component failures, (2) hybrid disruptions involving both malicious attacks and accidental failures. In addition, we identify which component failures/subtrees have a significant contribution to the disruption values.

System architecture. The system (see figure 16) consists of pumps and valves to regulate the flow of the polluting substance in a pipeline and sensors all along this pipeline, that communicate the pressure and the flow information to the *Remote terminal unit* (RTU). RTUs measure the pressure differences between the adjacent RTUs and send signals to the *Control centre* (CC) which controls the opening and the closing of the valves. All the field instrumentation (CC, RTU and sensors) are remotely connected to a Supervisory Control And Data Acquisition (SCADA) system.

The top event (see figure 18) *Pollution* can occur via spillage of toxic substance only if either there is an attack and then a pipeline failure (modelled with a PAND gate) or a pipeline failure followed by a sequentially an accidental protection failure (modelled with a SAND gate). The attack is assumed to happen once in a 5 year and involves malicious access to the SCADA. Here, the pipeline can break either accidentally or maliciously through the water-hammer attack. We also consider here a reflex action, which is a redundant safety action built at each RTU to shutdown the pump locally without waiting for the CC instructions.

Fig. 16. Schematic architecture of an oil pipeline taken from [37]

Fig. 17. Analysis results of oil pipeline case.

The results represented in Figure 17(a) show that in the case of hybrid scenario, the probability of pollution is 0.03 for a mission time of 1 year and approximately 0.13 for a mission time of 10 years. To find which leaf is the most critical, we perform a sensitivity analysis. This is performed by running the analysis multiple times while disabling the different subtrees and observing the percentage difference in the probability of disruption of the top event. As we see in Figure 17(b), "Access RTU" is the most critical subtree playing a significant role in the disruption, hence we put a detection mechanism on it. We re-run the simulations by putting two

Fig. 18. AFT for the oil pipe line

different detection mechanisms on RTU – a good one and a bad one. A good detection mechanism is the one which detects an attack with a discrete probability given as $\gamma = 0.5$ while a bad detection mechanism is one which detects an attack with a probability given as $\gamma = 0.1$. Figure 17(a) shows a good detection mechanism greatly reduces the probability of disruption (0.07 in 10 years). Thus, we can devise and compare the influence of mitigation strategies in both the accidental and the malicious disruption. The identification of critical components can prioritize preventive component maintenance and security hardening controls.

VII. CONCLUSION

In this paper, we provide a novel framework to capture the temporal and causal security and safety interactions through the AFTs. We have shown how AFT can be translated into a stochastic timed automata and analysed using statistical model checking. Our analysis allows us to quantify several combinations of accidental and malicious disruption scenarios yielding probabilistic estimates over time and compute several metrics, including the expected costs, the expected time and

the expected damage for the different adversaries. This is demonstrated through several case studies, enabling a risk manager to answer the *what-if scenarios* and thus prioritize the necessary steps to protect their critical infrastructures. In the future, we plan to extend our work by considering more cost structures in AFTs such as maintenance & repairs. Additionally, we also plan to refine our attacker model by making it fully adaptive.

ACKNOWLEDGEMENT

This work has been supported by the EU FP7 project TREsPASS (318003).

REFERENCES

- [1] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, "SAHARA: a security-aware hazard and risk analysis method," in *Proc. of the 2015 Design, Automation & Test in Europe Conf. & Exh., DATE*, 2015, pp. 621–624.
- [2] C. Raspotnig, V. Katta, P. Kárpáti, and A. L. Opdahl, "Enhancing CHASSIS: A method for combining safety and security," in *Int. Conf. on Availability, Reliability and Security*, 2013, pp. 766–773.
- [3] C. Schmittner, M. Zhendong, and P. Smith, "FMVEA for safety and security analysis of intelligent and cooperative vehicles," in *Proc. of Comp. Safety, Rel., and Sec. - SAFECOMP*, 2014, pp. 282–288.
- [4] I. N. Fovino, M. Masera, and A. D. Cian, "Integrating Cyber Attacks within Fault Trees," *Rel. Eng. & Sys. Safety*, pp. 1394–1402, 2009.
- [5] M. Steiner and P. Liggesmeyer, "Combination of Safety and Security Analysis - Finding Security Problems That Threaten The Safety of a System," in *SAFECOMP*, 2013.
- [6] G. Sabaliauskaite and A. P. Mathur, "Aligning Cyber-Physical System Safety and Security," in *1st Asia - Pacific Conf. on Complex Systems Design & Management, CSD&M*, 2014, pp. 41–53.
- [7] S. Chockalingam, D. Hadziosmanovic, W. Pieters, A. Teixeira, and P. Gelder, "Integrated safety and security risk assessment methods: A key of key characteristics and applications," to appear.
- [8] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, *Fault Tree Handbook*. Washington, DC: Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission, 1981.
- [9] E. Ruijters and M. Stoelinga, "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Computer Science Review*, 2015.
- [10] B. Schneier, "Attack trees," *Dr. Dobbs's Journal*, 1999.
- [11] "UMLsec: Extending UML for Secure Systems Development," in *5th Int. Conf. on the Unified Modeling Language, UML*, 2002, pp. 412–425.
- [12] Y. Roudier and L. Apvrille, "Sysml-sec - A model driven approach for designing safe and secure systems," in *Proc. of the 3rd Int. Conf. on Model-Driven Engg. and Software Development*, 2015, pp. 655–664.
- [13] B. Kordy, L. Piètre-Cambacédès, and P. Schweitzer, "Dag-based attack and defense modeling: Don't miss the forest for the attack trees," *Comp. Sc. Review*, pp. 1 – 38, 2014.
- [14] H. Hermanns, J. Krämer, J. Krcal, and M. Stoelinga, "The Value of Attack-Defence Diagrams," in *5th Int. Conf. on Principles of Security and Trust, POST*, 2016, pp. 163–165.
- [15] R. Kumar, E. Ruijters, and M. Stoelinga, "Quantitative attack tree analysis via priced timed automata," in *13th Int. Conf. on Formal Modeling and Analysis of Timed Systems, FORMATS*, 2015, pp. 156–171.
- [16] 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*, pp. 167–188, 2012.
- [17] F. Arnold, D. Guck, R. Kumar, and M. Stoelinga, "Sequential and parallel attack tree modelling," in *Computer Safety, Reliability, and Security - Workshops, ASSURE, DECSoS, ISSE, ReSA4CI, and SASSUR, Proc.*, 2015, pp. 291–299.
- [18] "Cyber value at risk in the netherlands," Deloitte, Tech. Rep., 2016.
- [19] "The insurance implications of a cyber attack on the us power grid," Cambridge centre of risk studies, Tech. Rep., 2015.
- [20] A. Remke and M. Stoelinga, Eds., *Stochastic Model Checking*, ser. LNCS, vol. 8453. Springer, 2014.
- [21] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT Press, 2008.
- [22] M. Duflot, L. Fribourg, T. Herault, R. Lassaigne, F. Magniette, S. Mes-sika, S. Peyronnet, and C. Picaronny, "Probabilistic model checking of the csma/cd protocol using prism and apmc," *Elect. Notes in Th. Computer Sc.*, pp. 195–214, 2005.
- [23] M. Kwiatkowska, G. Norman, and D. Parker, "Using probabilistic model checking in systems biology," *ACM SIGMETRICS Performance Eval. Review*, pp. 14–21, 2008.
- [24] —, "Probabilistic model checking in practice: Case studies with PRISM," *ACM SIGMETRICS Performance Evaluation Review*, pp. 16–21, 2005.
- [25] A. David, K. G. Larsen, A. Legay, M. Mikučionis, D. B. Poulsen, J. Vliet, and Z. Wang, *Statistical Model Checking for Networks of Priced Timed Automata*. Springer, 2011, pp. 80–96.
- [26] A. David, K. G. Larsen, A. Legay, M. Mikučionis, and D. B. Poulsen, "Uppaal smc tutorial," *Int. J. on Software Tools for Tech. Transfer*, pp. 397–415, 2015.
- [27] C. Carlson, *Effective FMEAs: Achieving Safe, Reliable, and Economical Products and Processes using Failure Mode and Effects Analysis*. Wiley, 2012.
- [28] M. Bozzano, A. Cimatti, J. P. Katoen, V. Y. Nguyen, T. Noll, and M. Roveri, "Safety, Dependability and Performance Analysis of Extended AADL Models," *Comput. J.*, pp. 754–775, 2011.
- [29] J. Jones, "An Introduction to Factor Analysis of Information Risk (FAIR)," *Norwich J. of Info. Ass.*, pp. 6–7, 2006.
- [30] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis - The CORAS Approach*. Springer, 2011.
- [31] I. S. Organization, "ISO/DIS 26262: Road vehicles, Functional safety," Geneva, Switzerland, Tech. Rep., 2009.
- [32] ISO/IEC, *ISO/IEC 64443. Software engineering - Product quality*. ISO/IEC, 2001.
- [33] "The sesamo project: Security and safety modelling." [Online]. Available: <http://sesamo-project.eu/>
- [34] "The safure project: Safety and security." [Online]. Available: <http://safure.eu>
- [35] S. Kriaa, L. P. Cambacédès, M. Bouissou, and Y. Halgand, "A survey of approaches combining safety and security for industrial control systems," *Rel. Eng. & Sys. Safety*, vol. 139, pp. 156–178, 2015.
- [36] P. J. Brooke and R. F. Paige, "Fault Trees for Security System Design and Analysis," *Computers & Security*, pp. 256–264, 2003.
- [37] S. Kriaa, M. Bouissou, F. Colin, Y. Halgand, and L. Piètre-Cambacédès, "Safety and security interactions modeling using the BDMP formalism: Case study of a pipeline," in *Computer Safety, Reliability, and Security - 33rd Int. Conf. Proc.*, 2014, pp. 326–341.
- [38] L. P. Cambacédès and M. Bouissou, "Modeling safety and security interdependencies with BDMP (Boolean logic driven markov processes)," in *Proc. of the IEEE Int. Conf. on Systems, and Cybernetics*, 2010, pp. 2852–2861.
- [39] F. Flammini, U. Gentile, S. Marrone, R. Nardone, and V. Vittorini, "A petri net pattern-oriented approach for the design of physical protection systems," in *Computer Safety, Reliability, and Security - 33rd Int. Conf. Proc.*, 2014, pp. 230–245.
- [40] P. T. Popov, "Stochastic modeling of safety and security of the e-motor, an ASIL-D device," in *Computer Safety, Reliability, and Security - 34th Int. Conf. Proc.*, 2015, pp. 385–399.
- [41] F. Arnold, H. Hermanns, R. Pulungan, and M. Stoelinga, "Time-Dependent Analysis of Attacks," in *3rd International Conference on Principles of Security and Trust, POST*, ser. LNCS, vol. 8414. Springer, 2014, pp. 285–305.
- [42] F. Arnold, A. Belinfante, F. V. der Berg, D. Guck, and M. Stoelinga, "DFTCalc: A Tool for Efficient Fault Tree Analysis," in *32nd International Conference on Computer Safety, Reliability, and Security. SAFECOMP*, vol. 8153. Springer, 2013, pp. 293–301.
- [43] H. Boudali, P. Crouzen, and M. Stoelinga, "A Rigorous, Compositional, and Extensible Framework for Dynamic Fault Tree Analysis," *IEEE Trans. Dependable Sec. Comput.*, vol. 7, no. 2, pp. 128–143, 2010.
- [44] L. Meshkat, J. B. Dugan, and J. D. Andrews, "Dependability analysis of systems with on-demand and active failure modes, using dynamic fault trees," *IEEE Tran. on Reliability*, pp. 240–251, 2002.
- [45] J. Wynn, J. Whitmore, G. Upton, L. Spriggs, D. McKinnon, R. McInnes, R. Graubart, and L. Clausen, "Threat assessment & remediation analysis (tara)," Mitre Corp., Tech. Rep., 2011.
- [46] P. Bulychyev, A. David, K. G. Larsen, A. Legay, D. Li, G. and Bøgsted Poulsen, and A. Stainer, *Monitor-Based Statistical Model Checking for Weighted Metric Temporal Logic*, 2012, pp. 168–182.