telecom

MDPI

Article

A Novel System-Theoretic Matrix-Based Approach to Analysing Safety and Security of Cyber-Physical Systems

Lin-Shen Liew [1,*], Giedre Sabaliauskaite [2], Nandha Kumar Kandasamy [1] and Choong-Yew William Wong [3]

[1] iTrust, Singapore University of Technology and Design (SUTD), Singapore 487372, Singapore; nandha001@e.ntu.edu.sg
[2] Institute for Future Transport and Cities (IFTC), Coventry University, Coventry CV1 5FB, UK; ad5315@coventry.ac.uk
[3] Panasonic R&D Centre Singapore (PRDCSG), Singapore 469332, Singapore; william.wongcy@sg.panasonic.com
* Correspondence: linshen.liew@gmail.com

Abstract: Cyber-Physical Systems (CPSs) are getting increasingly complex and interconnected. Consequently, their inherent safety risks and security risks are so intertwined that the conventional analysis approaches which address them separately may be rendered inadequate. STPA (Systems-Theoretic Process Analysis) is a top-down hazard analysis technique that has been incorporated into several recently proposed integrated Safety and Security (S&S) analysis methods. This paper presents a novel methodology that leverages not only STPA, but also custom matrices to ensure a more comprehensive S&S analysis. The proposed methodology is demonstrated using a case study of particular commercial cloud-based monitoring and control system for residential energy storage systems.

Keywords: cyber-physical system; hazard analysis; safety analysis; security analysis; STPA; STAMP; FMEA; STRIDE; matrix

Citation: Liew, L.-S.; Sabaliauskaite, G.; Kandasamy, N.K.; Wong, C.-Y.W. A Novel System-Theoretic Matrix-Based Approach to Analysing Safety and Security of Cyber-Physical Systems. *Telecom* **2021**, *2*, 536–553. https://doi.org/10.3390/telecom2040030

Academic Editor: Sotirios K. Goudos

Received: 19 October 2021
Accepted: 6 December 2021
Published: 9 December 2021

1. Introduction

Thanks to the advances in ICT (Information and Communications Technology), critical infrastructure systems such as power grids, public transportation and water distribution networks are getting increasingly connected and sophisticated. Desire for better real-time monitoring and control, additional functionality, greater efficiency, etc., is also a driving factor. Such modern system or even systems of systems, whose physical processes are remotely controllable/accessible over any telecommunication means (e.g., computer networks and the Internet), are also known as Cyber-Physical Systems (CPSs).

CPSs, like their traditional counterparts, have inherent safety risks that must be identified and sufficiently addressed. However, assessing the safety of CPS can be comparatively challenging and complicated due to the intertwining nature of CPS' physical processes and cyber elements. Most of the popular hazard/safety analysis techniques such as FMEA (Failure Modes and Effects Analysis) [1] and FTA (Fault Tree Analysis) were initially created for analysing systems whose reliance on software and network connectivity is minimal. Though progressively improved over the years, these techniques seem insufficient for analysing today's complex software-intensive systems (including CPSs). In CPSs, the faults/failures are no longer primarily caused by hardware reliability issues but hazardous interactions between system components. Hazardous interactions may occur accidentally because of poor/inadequate system design, or perhaps more likely be triggered by security attacks. Security attacks can result in significant physical damage to CPSs (e.g., German steel mill [2] and Iranian nuclear facilities [3]), which in turn jeopardize the overall safety.

Realizing the pressing need for holistic approaches to addressing Safety and Security (S&S) of CPSs, various integrated S&S analysis methods have been recently proposed in the

literature [4,5]. Among them are systems theory-based approaches that adopt/extend STPA (System Theoretic Process Analysis). STPA is a hazard analysis technique that has been applied in various domains such as aerospace [6], automobile [7], and nuclear facilities [8]. Some studies have shown that STPA can identify not only the loss scenarios found by other traditional safety analysis techniques (e.g., FTA and FMEA), but also many that are irrelevant to component failures; meanwhile, it consumes less resources including time. Another notable advantage of adopting STPA is that it can be applied in early phases of the system development lifecycle. Analysis can commence as soon as the high-level goals of the system are defined, and can be further elaborated as the system design specification matures progressively.

Nevertheless, STPA has some limitations that may make it difficult to apply especially when analysing the security of a system. Therefore, some extended versions of STPA have been proposed recently. However, even from these existing STPA-based analysis methods, we note two major gaps as follows: (1) little or no explicit guidance on deriving Security requirements; and (2) lack of discussion on managing the relationships (especially conflicts) between the derived Safety and Security requirements. To address these gaps, this paper proposes a novel S&S analysis method that utilizes custom matrices.

Graphical and tabular representations of data/information are typically used to facilitate analysis; and matrices are no exception. The custom matrices presented in this paper are inspired by our earlier works—e.g., [9,10]—where unique combination of matrices were utilized in managing the S&S of automated vehicles (a kind of CPS). Nonetheless, these combinations of matrices and even other existing ones are generally different in terms of (1) quantity of matrices, (2) sequence/arrangement of matrices, and (3) semantics. Obviously, there's no one-size-fits-all combination of matrices. Even when analysing the same system different analysts may employ different combinations of matrices, depending on their interests, ways of thinking, expertise level, etc. A particular combination of matrices is considered effective (though not perfect) as long as it eases particular problem(s) faced by the analysts—which, in the context of this paper, shall be overseeing the correlations between Safety and Security of the analysed system.

The remainder of this paper is structured as follows. Section 2 reviews STPA and related analysis approaches. Section 3 gives a detailed description of the proposed S&S analysis methodology. Section 4 briefly illustrates how the proposed methodology is applied to analyse a realistic CPS in power domain. Section 5 concludes the paper and outlines the future work.

2. Literature Review

Various integrated safety and/or security analysis approaches have been proposed in the literature, focusing on different phases of the system life-cycle. Regardless, they can be classified into three categories based on the discussion in [4,5,11]:

- Security-informed Safety approaches that leverage on security-related information (e.g., intentional causes) for enhancing safety analysis;
- Safety-informed Security approaches that utilize safety-related information (e.g., component failures) for enhancing security analysis;
- Combined S&S approaches that analyse Safety and Security either in series or parallel and their respective artefacts are mutually integrated into subsequent analysis steps/phases/stages.

STPA is an emerging hazard analysis technique that has been integrated into many S&S analysis approaches proposed in recent years. The S&S analysis methodology proposed in this paper is also built on STPA. Therefore, this section reviews some existing approaches that adopt or extend STPA, and highlights their limitations which the proposed methodology intends to address.

2.1. System Theoretic Process Analysis (STPA)

STPA [12], proposed by Leveson in 2011, is a hazard analysis technique that has gained popularity among researchers and practitioners in engineering the safety of complex systems in various domains. It is developed based on STAMP (Systems-Theoretic Accidents Model and Process)—a model that considers the safety of a complex system as a control problem rather than a component failure or reliability issue. It aims to identify hazardous control scenarios that might lead to unwanted losses, and then accordingly define the constraints/requirements to avoid/mitigate such scenarios. Inadequate controls can arise because of human error, software error, inappropriate interaction between system components, etc. Detailed reasons why traditional safety engineering approaches are unfit for addressing the safety of complex systems can be found in [12]. However, STPA has not explicitly considered the security aspects of the system.

Basically, the flow of STPA can be summarized as follows: (1) Identify the high-level (systemic) accidents/losses that are to be prevented; (2) Identify the potential hazards that would lead to the identified accidents/losses; (3) Define the safety constraints with regards to identified hazards; (4) Draw the control structure diagram showing the components of the system of interest as well as their interfaces and boundaries; (5) Identify all the control actions involved in the control structure diagram; (6) Identify the hazards that may arise when these control actions are inadequate—i.e., not provided, incorrectly provided, untimely provided, and stopped/applied too soon/long; (7) Translate the hazardous control actions into safety constraints; (8) Identify how the hazardous control actions would occur based on the causal factors shown in Figure 1; and (9) Translate the identified causal scenarios into safety constraints.

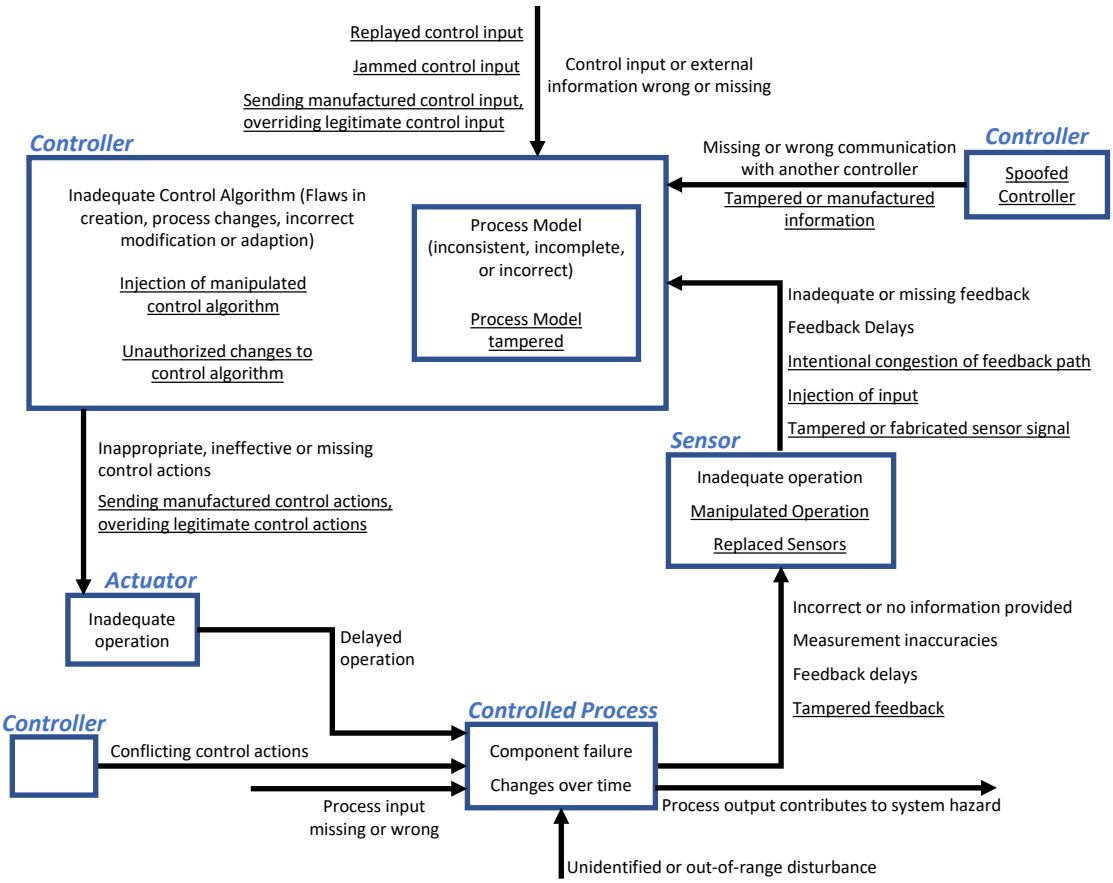

Figure 1. STPA's generic control loop diagram annotated with relevant causal factors resulting in hazardous/insecure control actions [12–14]; note that the underlined texts are security related guide-words proposed by Schmittner et al. [15] to help deducing intentional/malicious scenarios.

2.2. STPA-Sec (STPA for Security)

Most of the conventional security assessment approaches have adopted tactics models [13,14]. In such models, the emphasis is placed on identifying the chains of events in which the particular threats of interest precipitate in order to reach their (malicious) goals, and then devising the tactics to break those chains. Such a threat-based approach works best when the security analysts possess a good understanding of the system assets as well as attackers/adversaries' capabilities, motives, etc. It enables the security analysts to pin-point the entry points (vulnerabilities of logical/physical system components) that are likely to be exploited by the attackers/adversaries, and thereafter suggest the most cost-effective ways to prevent attacks, hence losses. Such approach however, as argued in [13], seems lacking in securing today's diverse, interconnected, software-intensive systems. Security analysts adopting it might conduct the assessment without having the right set of priorities that aligns well with the higher-level goals or intended functions of the system; for instance, while disabling the access to particular data might be deemed imperative by the security team, it bothers those (e.g., operation team) who need the access to perform certain higher-level system functions.

In view of the aforesaid limitations of threat-based (bottom-up) approaches, Young and Leveson introduced a top-down security analysis method, namely STPA-Sec [13,14]. The process of STPA-Sec is largely identical to that of STPA. The major difference between STPA-Sec and STPA is that the former aims to prevent the system from getting into vulnerable states while the latter—hazardous states. However, STPA-Sec utilizes the same control loop diagram (depicted in Figure 1) of STPA, which does not explicitly show which causal factors are relevant to security; this may not be an issue to the experienced analysts though. Nevertheless, a control loop diagram containing more explicit and security-related information (including terminologies) is more likely to facilitate identification of hazardous/vulnerable scenarios, as advocated in [15] where the said diagram enhanced with some relevant security guide-words was applied in analysing a real-world complex and connected automotive system.

2.3. STPA-SafeSec

There is a growing recognition that Safety and Security should be considered holistically when analysing/designing CPSs such as industrial control systems [16], automated vehicles [17–19], and intelligent transport systems [20–22].

STPA analyses the system mainly from Safety perspective while STPA-Sec—Security perspective. As an effort to treat safety and security equally, Friedberg et al. combines STPA and STPA-Sec into a concise approach called STPA-SafeSec [23]. To complement the STPA's generic control loop diagram (shown in Figure 1), STPA-SafeSec introduces a generic component layer diagram encompassing the physical components (e.g., network device and connection) that typically exist in any control loop. In addition, it suggests the likelihoods of certain security threats violating availability/integrity can be exploited on particular system components; these threats serve as additional causal factors, besides those included in Figure 1, that the analyst should consider in identifying the S&S constraints. The constraints are then mapped to the system components using a graph, as illustrated in Figure 2, to highlight which constraints may be violated at each component. The greater the number/criticality of constraints potentially violated at a component, the greater the urgency to perform in-depth security analysis on that component. By leveraging on the graph and the component layer diagram, the analysts might see which S&S constraints are in fact more crucial for addressing the hazards, thereby determining the most cost-effective set of constraints/solutions.

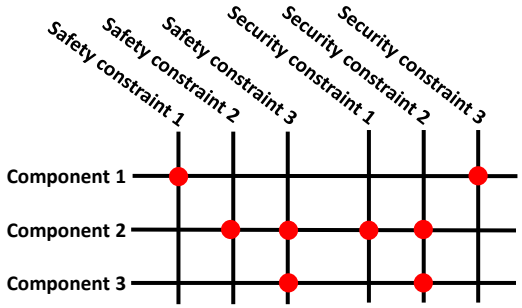

Figure 2. An example of graphical representation that shows which safety or security constraints could potentially be violated at every component of the system.

However, it is unclear how STPA-SafeSec defines those Availability/Integrity threats and even the likelihoods of exploiting them. Obviously, it does not address Confidentiality—one of the three key Security properties besides Availability and Integrity. Please note that, in contrast with STPA-SafeSec, the proposed S&S analysis methodology (see Section 3) employs STRIDE model [24] and thus covers all three key Security properties—i.e., Confidentiality, Integrity, and Availability.

Furthermore, the inter-dependency between S&S constraints (aka requirements), which STPA-SafeSec considers, seems not falling into any of these three categories (discussed in [25,26]), namely Conditional Dependency, Conflict, and Reinforcement. There is a consensus that the conflicts between requirements should be identified and resolved as early as possible especially during system development [27–32]. Identifying or resolving conflicts between requirements can be challenging [33]. There seems to be a paucity of literature on how to identify conflicts between Safety and Security; Pereira et al. [32] suggest that both safety analysts and security analysts should work together; Kriaa et al. [26] utilize a modeling formalism called BDMP (Boolean logic Driven Markov Processes). The latter the conflicts get spotted, the greater the cost/effort to resolve them [33].

2.4. SAFE (Systematic Analysis of Faults and Errors)

Inspired by STPA, Procter et al. initiates SAFE [34] with steps identical to those of STPA—i.e., defining high-level losses, followed by identifying potential hazards and modelling the system's control structure. From the control structure model, instead of examining any control actions like STPA does, SAFE immediately looks into the system component that is nearest to the controlled process. On this component there are basically five steps performed: (i) define the process model of the component (ii) define the erroneous inputs that could cause the component to produce hazardous output; (iii) document how erroneous inputs could lead to hazardous outputs as well as how to prevent/mitigate them; (iv) identify the component's internal faults based on a standardized set of fault classes; (v) document how internal faults could lead to hazardous outputs as well as how to detect/prevent them at design or run-time. These five steps are then recursively performed on every component that precedes the current component, until all system components have been covered.

As for addressing Safety and Security in SAFE, in step (iii) Procter et al. examines how the receiving component is negatively affected by seven categories of erroneous inputs derived from a particular security adversary model [35] established in 1983. These seven categories of erroneous inputs basically summarize all possible ways that the inputs can be erroneous regardless of whether they arise due to accidental faults or intentional attacks. Simply put, SAFE disregards the causes or sources of erroneous inputs; it claims that both Safety and Security concerns can be addressed merely by focusing on the effects of a minimal set of erroneous inputs on every component of the system and not on the underlying causes of erroneous inputs. In Procter's earlier work [36], the erroneous inputs considered in SAFE were derived from a taxonomy proposed by Avizienis et al. [37] in 2004.

It is undeniable that both accidental faults (e.g., broken input transmission cable) and intentional attacks (e.g., injection by Man-In-The-Middle (MITM)) would result in the same erroneous inputs (e.g., corrupted message) to the receiving components. However, identifying potential causes/sources of erroneous inputs is likely to result in more effectively planned measures; note that this is a common advantage of bottom-up hazard/threat analysis techniques. For instance, if input injection by MITM is deemed impossible in light of the component's realistic environment, then perhaps it would be more feasible or cost-effective to just ensure the input transmission cable's reliability than requiring the legacy component to filter out injected messages. Besides, while SAFE acknowledges that the system design has to be updated once the measures are introduced to fulfil the derived requirements, it seems to not mention whether/how the resultant changes should be analysed.

In contrast with SAFE, the proposed S&S analysis methodology—as discussed later in Section 4—explicitly considers both accidental/natural factors and intentional/malicious causes (of erroneous inputs) in order to determine a more practical and cost-effective set of S&S requirements (and thus S&S measures); besides, any resultant changes in system design shall be reflected by the proposed matrices to help the analyst check the overall equilibrium among various aspects of the analysed system.

2.5. Other Works That Employ STPA

In [15], STPA-Sec [13] is extended by introducing a standardized set of safety and security terminologies and adding some security elements into its generic control loop diagram; such extended STPA-Sec is demonstrated via analysing a vehicular battery management system [15]; besides, limitations of the original STPA-Sec with respect to relevant automotive standards (e.g., ISO26262 and SAEJ3061) are discussed. In [38], the extended STPA-Sec proposed in [15] is adopted to identify the hazardous scenarios which are then inputted to a component-centric approach called FMVEA (Failure Mode, Vulnerabilities, and Effect Analysis) [39] to evaluate the likelihood and severity of the Safety and Security risks of a realistic train braking system. In [32], STPA and NIST SP 800-30 [40] are employed to identify the Safety constraints and Security constraints, respectively; the security team performing NIST SP 800-30 also works with the safety team in certain steps of STPA to identify the potential causal factors and scenarios for identifying the Safety constraints; then, Safety team and Security team jointly process the two individual sets of constraints to identify the potential conflicts or pairwise reinforcements between them, and subsequently define the appropriate security measures and safety recommendations; such methodology is illustrated via analysing a revolving door system. In [41], STPA is employed to identify how and when hazardous scenarios may emerge due to generic causes and intentional causes like jamming and tampering; the identified hazardous scenarios are then translated into critical requirements on a formal model processed via a specific tool to formally verify whether the constraints sufficiently mitigate the identified hazardous behaviours; such methodology is demonstrated via analysing a hydroponics facility. In [10], safety requirements and security requirements are derived from STPA and attack trees, respectively; the derived requirements along with other relevant information are then integrated into a Six-Step Model [42] for achieving integration and alignment safety and security of automated vehicle.

3. Proposed S&S Analysis Methodology

The proposed analysis method basically consists of 7 steps, as summarized in Figure 3, which may be iterative until the applicable safety and security requirements are reasonably satisfied. Note that the proposed methodology's flow (from *Step 1* to *Step 6*) is similar to those of STPA [12] and STPA-Sec [13]; all seven steps of the proposed methodology are described in the following.

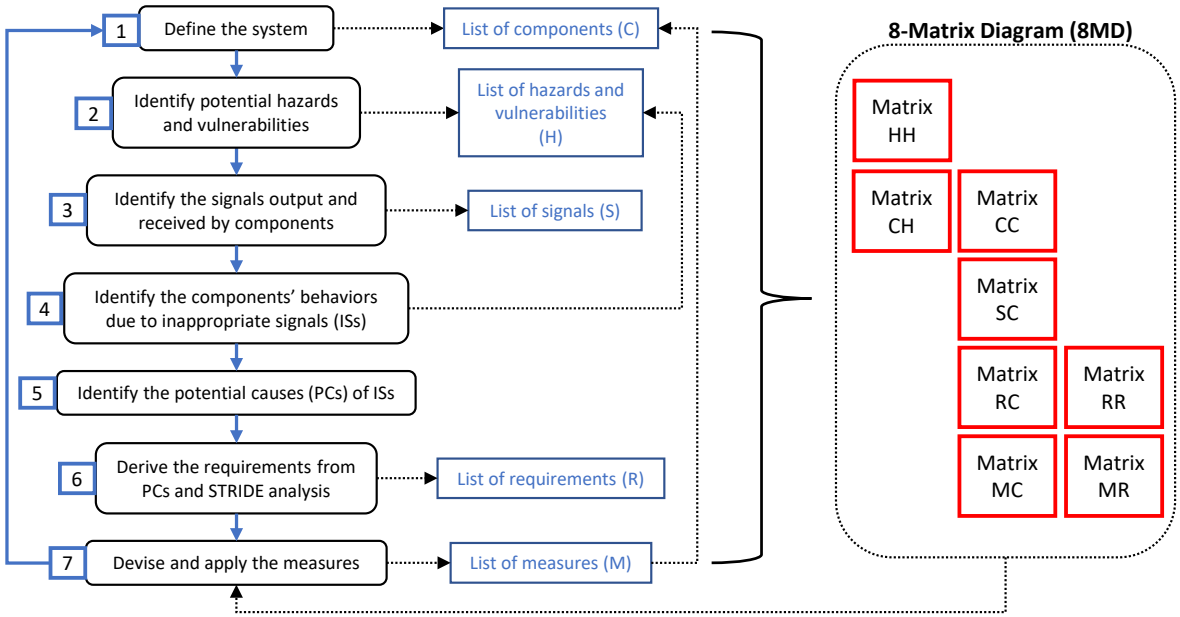

Figure 3. Overview of the proposed Safety and Security (S&S) analysis method. The methodology is basically comprised of 7 steps, which may be iterated until the derived set of S&S requirements is deemed reasonably satisfied. The 8-Matrix Diagram (8MD), constructed and constantly updated using analysis artefacts, is utilized to oversee correlations between Safety and Security of the analysed system.

The analysis (*Step 1*) starts with definition of the system. It should describe the interactions between the system and its environment (e.g., stakeholders and weather) as well as between hardware/software components of the system. This initial step may involve materials and diagrams that describe the system from various perspectives—for example, control structure/loop (as in STPA), hardware architecture, and data flow. The gathered information is then utilized in *Step 2* to deduce the relevant hazards/vulnerabilities that may result in undesired accidents or loss events. For example, an ill-trained maintenance worker overlooking particular ageing machine parts, which eventually leads to machine failure and hence service interruption.

Step 3 of the proposed method aims to identify all necessary and unnecessary signals that may exist within the system. Necessary signals (e.g., commands, data, feedback, and power) are those transmitted among the system components to ensure intended functionality of the system. Unnecessary signals are usually byproducts of their emitters, or those not beneficial to the components who receive them intentionally or unintentionally. For instance, loud noises or high heat output by machines can be unnecessary (and thus hazardous) signals to nearby users or flammable substances; besides, a router's Wi-Fi signal, which is needed for enabling authorized personnel to remotely access the networked devices, also exposes the router to hacking if it can be captured by attacker using dedicated tools. Therefore, any unnecessary signals identified ought to be removed immediately if possible.

For each necessary signal, *Step 4* identifies the adverse impact that may arise should the signal become inappropriate in the following ways: (1) the signal is not provided; (2) the signal provided is wrong; (3) the signal is untimely provided; and (4) the signal is stopped/applied too soon/long. These four types of inappropriate signal are adapted from four types of unsafe control action mentioned in STPA; note that not all four ways of inappropriate signal (or unsafe control action) are always applicable. The adverse impact is deduced considering how the components (receiving inappropriate signals) would react internally as well as its resultant actions towards all of its dependent components; as this probably looks into the behaviours/algorithms of the involved components, additional or more specific hazards/vulnerabilities (e.g., hardware limitations and software bugs) may be identified. Moreover, any pre-identified unnecessary yet non-removable

signals are also considered as inappropriate signals. This step ultimately produces a list of inappropriate signals.

Step 5 identifies the potential causes (PCs) of identified inappropriate signals, by utilizing the *template* illustrated in Figure 4. The *template* is manifested as a tree that summarizes the generic reasons behind a malfunctioning component. Component Malfunction (CM) denotes that the component is unable to produce the correct/desired output; it can potentially be due to any of three main factors, namely Algorithm Failure (AF), Input Failure (IF), and Damage/Disruption (DD). AF denotes that the component's algorithm is inadequate; IF—the input required by the component is unavailable or inadequate; DD—the component is damaged or disrupted. Certainly, not all nodes of the tree are applicable to every component of the system; each node may be decomposed into low-level nodes to further explain the causal factor; the analysts are required to apply only the relevant ones during elicitation of PCs. It is worth noting that the *template* is designed considering the information shown in Figure 1; any causal factors (shown in Figure 1) can be mapped to the proposed *template*; for example, "feedback delays", "inadequate or missing feedback", "inappropriate, ineffective or missing control actions" are basically input failures to their receivers; besides, "inadequate operation", "inadequate control algorithm", and "process model tampered" can be considered as algorithm failures; furthermore, any elements including actuators may potentially be tampered, which is clearly not included in STPA's control loop diagram (c.f. Figure 1). Therefore, the *template* is deemed useful as it provides a systematic means of stating the PCs using consistent and concise words. Stating the PCs succinctly is important to minimize confusions, as a large quantity of PCs are usually identified during the analysis and processing them in subsequent analysis stages can be time-consuming. Nevertheless, the *template* could be tailored for specific applications by comprising more precise and relevant causes/nodes.

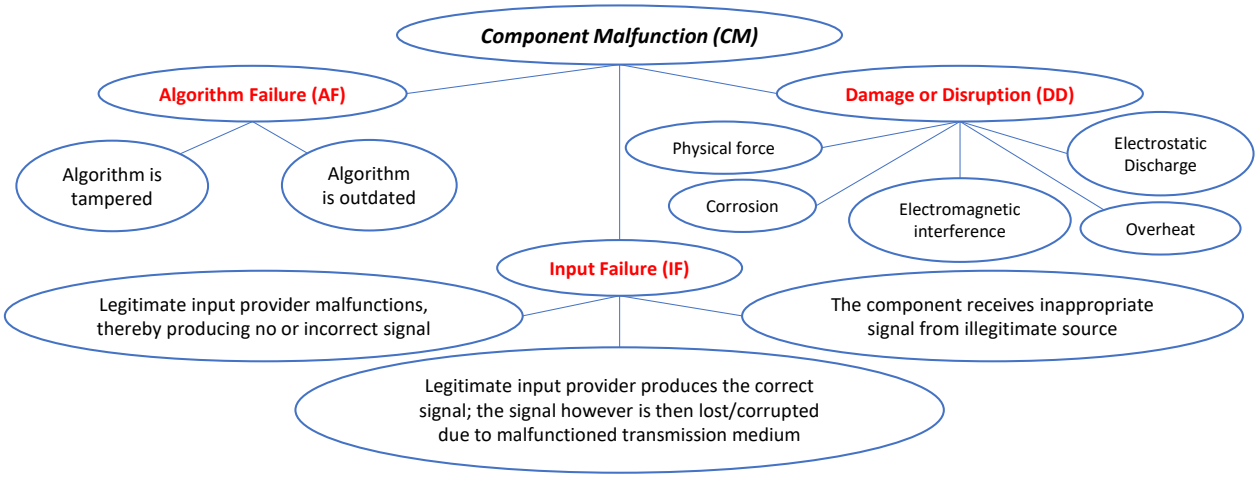

Figure 4. A template, manifested as a tree, that shows the three generic factors of Component Malfunction (CM), namely Algorithm Failure (AF), Input Failure (IF), and Damage/Disruption (DD).

Step 6 is intended to derive the S&S requirements from identified PCs and STRIDE analysis. STRIDE is among the most popular threat-modelling methods [24,43]; it evaluates a system against six main categories of threats, namely, Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. Specifically, we employ MTMT (Microsoft Threat Modelling Tool) to perform STRIDE analysis. Based on the Data Flow Diagrams (DFDs) which we build corresponding to the analysed system, MTMT would automatically output a list of potential security threats. By addressing the MTMT-identified security threats, it is reasonable to assume that CIA (Confidentiality, Integrity, and Availability) of analysed system is adequately ensured. The MTMT output can be regarded as security requirements which could also be used to refine the pre-identified PCs. Put crudely, Spoofing or Tampering may correspond to AF, IF, or DD;

Denial of service—IF or DD; and Elevation of privilege—AF or DD. All the identified PCs (called *causal scenarios* in STPA) are then translated into requirements (called *constraints* in STPA), simply by negating them. For instance, a PC stating "Adam provides wrong command to Bob because Adam receives incorrect data from Collin" can be translated into a requirement stating "Adam must receive correct data from Collin so that Adam provides correct command to Bob".

Step 7 aims to ensure that the derived set of S&S requirements has been adequately fulfilled. The process is typically lengthy because the analyst team is required to meticulously check a large set of requirements for the following reasons: (1) refining vague requirements with more precise details and keywords (e.g., components and signals); (2) removing repetitive, redundant, or obsolete requirements; (3) identifying potential conflicts between requirements; (4) planning and deciding the measures for satisfying the requirements based on criteria like cost incurred and measures' effectiveness. Making decisions (especially prioritizing which requirements or measures) can be challenging without having an accurate understanding of the impact which a decision may bring. This can be facilitated by leveraging on a graphical tool called 8-Matrix Diagram (8MD) that enables the analyst team to holistically view the essences of analysis artefacts. As illustrated in Figure 3, 8MD is formed by 8 matrices (described in Table 1) built of artefacts obtained throughout the analysis process. In each matrix, as described in Table 1, particular symbols may be placed at intersections of row headers and column headers to denote their relationships. An example of 8MD is given in Section 4 to demonstrate the application of the proposed methodology for a realistic CPS in power domain.

Table 1. Description of eight matrices that form 8-Matrix Diagram (8MD).

Matrices	Rows	Columns	Possible Symbols and Their Meanings
HH	Hazards	Hazards	X—The (row) hazard arises due to the (column) hazard.
CH	Components	Hazards	X—The component is the final trigger of the hazard;
CC	Components	Components	<—The (row) component receives signal from the (column) component; >—The (row) component provides signal to the (column) component.
SC	Signals	Components	P—The component is the legitimate provider of the signal; R—The component is the intended receiver of the signal; T—The component is the transmission medium of the signal.
RC	Requirements	Components	X—The requirement may be violated if the component malfunctions.
RR	Requirements	Requirements	X—The (row) requirement conflicts the (column) requirement.
MC	Measures	Components	X—The component is required for implementing the measure.
MR	Measures	Requirements	O—The measure contributes to satisfying the requirement; X—The measure may violate the requirement.

The analysis could just terminate at the end of *Step 7*, if the derived set of S&S requirements is considered reasonably satisfied. Or, with all the artefacts (including 8MD) established thus far, it can be iterated from *Step 1*. Such iteration would be necessary, if there are changes in the system architecture because of addition/removal/modification of particular system elements. The changes also need to be analysed to make sure they do not introduce unacceptable risks. At each iteration the analysis artefacts are revised accordingly. It is worth noting that the scales, metrics, criteria, or benchmarks used for judging if the derived set of S&S requirements is reasonably satisfied are determined on a case-by-case basis, which typically includes stakeholder expectations, applicable national/international standards, local rules and regulations.

4. Case Study

4.1. System Definition

The Cyber-Physical System (CPS) in this case study is a cloud-based monitoring and control system for large residential lithium-ion (Li-on) energy storage systems. An overview of the said CPS is shown in Figure 5 to illustrate the interactions among the system hardware components; note that it is not intended to be exhaustive but showing some necessary signals (including commands, measurements, actions, etc.) transmitted between components. Please also note that technology-specific details are abstracted owing to commercial sensitivity; though not explicitly shown in Figure 5, various telecommunication means (e.g., cellular networks, computer networks, and the Internet) are in fact involved to enable seamless signal exchange among a plethora of the system components and sub-components.

In the following, the term *system* is used to denote the CPS of this case study. In the *system*, Lithium-ion (Li-ion) batteries are used to store the energy. Such large-scale energy storage, as compared to small and medium-size applications like consumer electronics, is more likely to be exposed to extreme conditions. Li-on batteries can fail owing to cell's chemical imbalance, internal short-circuit, or external causes such as exposure to high heat, mechanical damages, and electrical abuse conditions. Failure of Li-on batteries is probably among the most undesired risks, as it can lead to fire or explosion, and hence significant losses. A good discussion on potential hazards related to Li-on battery systems as well as their mitigation may be found in [44]. Please note that the analysis is carried out on a commercial system and not a research grade equipment; therefore, the analysis is practically validated on a system that is commercially viable.

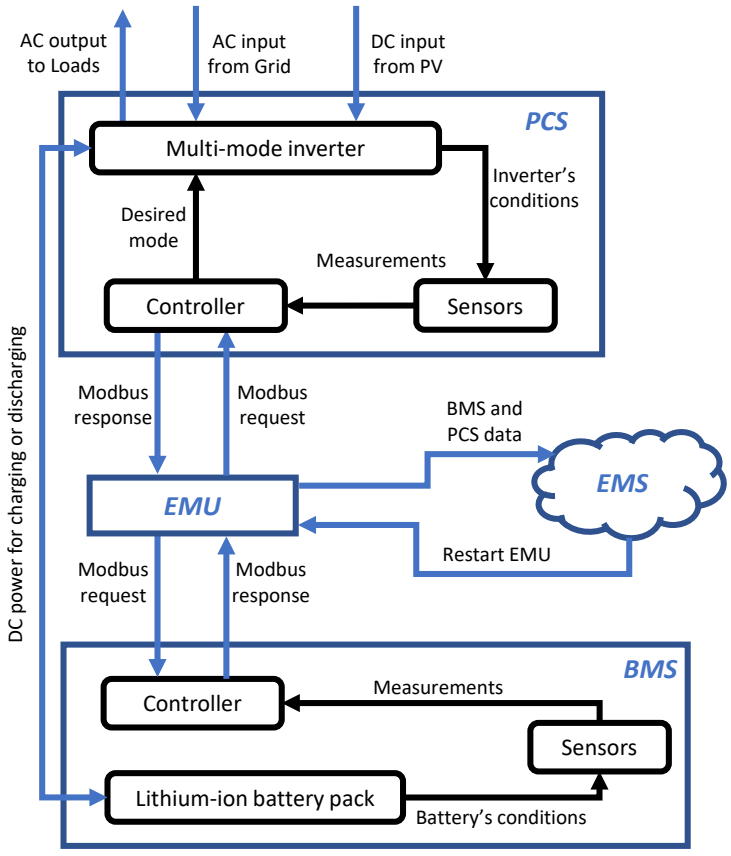

Figure 5. A simplified architecture of the cloud-based monitoring and control system for residential Li-on battery storage systems.

As shown in Figure 5, there are four major components in the *system*, namely EMU (Energy Monitoring Unit), PCS (Power Conversion System), BMS (Battery Management System), and EMS (Energy Management System). The EMU communicates with PCS/BMS

via Modbus RTU using serial RS-485 cables. The EMU is the master while PCS and BMS are the slaves. Periodically, EMU uploads to EMS the data it requests from both PCS and BMS. On certain occasions, EMU may receive operational commands (e.g., shutdown or reboot EMU) from EMS. The PCS serves as an interface between the battery and the grid/load, and determines the rate of charging/discharging. It converts AC power into DC power (or vice versa) in the event of battery charging (or discharging).Furthermore, it monitors local measurements from current and voltage transformers, and contains some features to protect the system from over-current, phase imbalance, etc. Periodically, PCS reports its status and applies the required configuration, as requested by EMU. The BMS measures the battery's conditions (e.g., cell voltage, cell temperature, and current). Based on the measured values, it estimates the battery's SOC (State of Charge) and SOH (State of Health) with a pre-defined process model of the battery behaviour. It regulates/controls charging and discharging of the battery, thereby ensuring the battery system is operating safely. Appropriate charging and discharging of battery is essential for maximizing the lifespan and performance of the *system*. Overcharging of battery could result in physical damage (including explosion) while over-discharging would increase the batteries' aging rate. Periodically BMS responds to EMU's request for PCS status data. The EMS is basically a website hosted on a AWS (Amazon Web Services) cloud server, which allows the authorized personnel (using valid login accounts) to perform the following activities: (1) see both historical and real-time data regarding PCS and BMS; (2) shutdown/restart EMU; (3) carry out energy management functions such as configuring the charging/discharging limits; and (4) override the set-points on PCS and BMS via EMU.

4.2. Analysis

Due to space limitation, the analysis artefacts presented in the following are not intended to be exhaustive, but illustrative, to demonstrate the application of the proposed methodology. Additionally, technology-specific details have also been abstracted for commercial sensitivity. An overview of the proposed methodology is provided in Figure 3. The information provided in Section 4.1 can be regarded as data gathered by *Step 1* of the proposed methodology. Architecture diagrams like Figure 5 can be used to derive a list of the system components. Surely, more details gathered (if possible) in *Step 1* would provide more useful inputs to the subsequent steps.

Based on the data gathered in *Step 1*, *Step 2* deduces the unacceptable accidents/losses and their causal hazards. Considering the properties of Li-ion batteries, the unacceptable losses should include fire/explosion caused by Li-ion batteries, and the ageing of Li-ion batteries. These losses are usually associated with some hazards shown in Table 2. Based on the specifications of the *system* components, potential vulnerabilities may be identified by utilizing relevant vulnerability databases, security testing tools, etc. For example, CVEs (Common Vulnerabilities and Exposures) of EMU can be identified via a particular network scanner called Nmap.

Table 2. Some examples of the *system* hazard.

ID	Description of hazard
H1	Battery ignition
H2	Battery temperature high (beyond 60 degrees)
H3	Battery over-charged
H4	Battery over-discharged
H5	Acidic/corrosive/flammable/toxic substances due to electrolyte leakage

Step 3 identifies a list of signals that the *system* components may receive and output, considering the data gathered by *Step 1*. All signals shown in Figure 5, such as "Modbus request" and "Measurements", are necessary signals. As for unnecessary signals, it was found that EMU (by default Bluetooth-enabled) emits Bluetooth signals that are not required by any operations of the *system*; Bluetooth device could be hacked; therefore,

from the point of view of Security, any wireless connectivity (including Bluetooth) that is not being used or needed should be disabled to minimize the attack surface.

For every identified signal, *Step 4* examines how would its receiver (*system* components) behave in response to inappropriate signal. As mentioned in Section 3, a signal can be inappropriate in four ways; not all of them are always applicable. An excerpt of result of *Step 4* is depicted in Figure 6, describing how PCS Controller behaves both internally and towards the *system* components which it interacts with when the input signals S1 and S3 are not provided or incorrectly provided. As shown in Figure 5, PCS controller is supposed to interact with PCS inverter and EMU. Basically all behaviors of PCS Controller, described in Figure 6, can be hazardous. They may also imply flaws in the algorithm of PCS Controller that should be fixed for improving the safety or security; for instance, in case of not receiving latest measurements, PCS Controller should somehow alert the maintenance personnel to check on PCS Sensors.

ID #	Signal	Signal provider	Signal receiver		How signal receiver behaves if the signal is not provided	How signal receiver behaves if the signal is incorrectly provided
S1	*Configuration needed*	*EMU*	*PCS Controller*	**internally**	PCS Controller remains operating with last configuration which may lead to inappropriate operation	PCS Controller would be wrongly configured, thereby leading to inappropriate operation
				towards EMU	-	-
				towards PCS Inverter	-	PCS Controller would command Inverter to operate with wrong mode
S3	*Measurements*	*PCS Sensors*	*PCS Controller*	**internally**	PCS Controller does not alert about the absence of latest measurements	-
				towards EMU	-	-
				towards PCS Inverter	-	-

Figure 6. Identifying how PCS Controller behaves due to inappropriate signals S1 and S3.

For every identified signal, *Step 5* identifies the potential causes (PCs) of inappropriate signals. Figure 7 gives an example of PCs for inappropriate signal S1 ("Configuration needed"), which are derived from the *template* (shown in Figure 4); from the perspective of PCS Controller, receiving incorrect or not receiving S1 means input failure; therefore, according to the branch "Input Failure (IF)" of the *template* (depicted in Figure 4), S1 failure is mainly caused by S1-I1, S1-I2 and S1-I3. These PCs can be further elaborated by applying the relevant nodes of the *template*; the elaboration is iterative until a reasonable middle ground is achieved. For instance, S1-I1 mentions component (EMU) malfunction; so, according to the *template*, all three branches for "Algorithm Failure", "Input Failure" and "Damage or Disruption" may be applied to elaborate S1-I1; this results in S1-I1-A1, S1-I1-A2, S1-I1-I1, S1-I1-I2, S1-I1-I3, and S1-I1-D1. Likewise, S1-I1-I2 mentions component (transmission medium) malfunction; thus, the entire *template* may be applied again to elaborate S1-I1-I2, thereby leading to S1-I1-I2-D1. Nonetheless, not all branches or nodes of the *template* are always applicable to any *system* components.

Step 6 outputs a set of S&S requirements based on STRIDE analysis and also by negating the identified PCs. MTMT result can not only be converted into Security requirements, but may also be used to refine some identified PCs; for instance, as MTMT result mentions the possibilities of replay attacks on particular data like Modbus messages, S1-I3 can therefore be refined as "PCS Controller receives replayed signal S1".

In the following, we discuss the 8MD illustrated in Figure 8; note that the matrices' symbols are described in Table 1.

ID#	Potential causes of inappropriate signal S1 ("Configuration needed")
S1-I1	The legitimate input provider, i.e. EMU, malfunctions;
S1-I1-A1	EMU's algorithm is tampered;
S1-I1-A2	EMU's algorithm is outdated;
S1-I1-I1	EMU's legitimate input provider (e.g. power adapter) malfunctions;
S1-I1-I2	The transmission medium (e.g. cable of power adapter) malfunctions;
S1-I1-I2-D1	Cable of power adapter is damaged/disrupted due to mechanical force;
S1-I1-I3	EMU receives inappropriate signal from illegitimate source;
S1-I1-D1	EMU is damaged/disrupted due to overheat;
S1-I2	The transmission medium, i.e. Modbus serial adapter cable (MSAC), malfunctions;
S1-I2-A1	MSAC's algorithm is tampered;
S1-I2-A2	MSAC's algorithm is outdated;
S1-I2-D1	MSAC is damaged/disrupted due to corrosion;
S1-I2-D2	MSAC is damaged/disrupted due to electromagnetic interference;
S1-I2-I1	MSAC's legitimate input provider (i.e. EMU) malfunctions;
S1-I2-I3	MSAC receives inappropriate signal from illegitimate source;
S1-I3	PCS Controller receives inappropriate signal from illegitimate source;

Figure 7. Example of Potential Causes (PCs) derived from the *template*. These PCs describe why S1 ("Configuration needed") can be inappropriate (i.e., incorrect or not provided).

ID#		H1	H2	H3	C1	C2	C3.1	C3.2	C3.3	C4.1	C4.2	C4.3	C5	R1	R2	R3	R4
H1	Battery ignition		X														
H2	Battery temperature too high			X													
H3	Battery overcharged		**HH**														
C1	EMS					<>			**CC**				<>				
C2	EMU		**CH**				<>			<>			<>				
C3.1	PCS Controller					<>		>	<								
C3.2	PCS Inverter			X			<		>		<>						
C3.3	PCS Sensors						>	<									
C4.1	BMS Controller					<>						<					
C4.2	BMS Battery Pack	X	X	X				<>				>					
C4.3	BMS Sensors									>	<						
C5	Router				<>	<>											
S1	Configuration needed					P	R										
S3	PCS measurements							R	P								
S7	BMS measurements									R		P					
S8	Restart EMU				P	R			**SC**				T				
S9	BMS and PCS data				R	P							T				
R1	Ensure integrity of **C2**					X										X	**RR**
R2	Ensure integrity of **C5**												X			X	
R3	Ensure availability of **C5**'s Wi-Fi								**RC**				X	X	X		
R4	**C2** generates correct **S9**					X	X			X							
M1	Disable **C5**'s Wi-Fi												X	O	O	X	
M2	Password complexity								**MC**				X		O		
M3	Limit number of SSH login attempts					X								O			
M4	Anomaly detection mechanism					X	X			X					**MR**		O

Figure 8. A highly simplified version of 8-Matrix Diagram (8MD) built for the case study.

4.2.1. Matrices Constructed Prior to *Step 7*

Matrix HH captures the cause-and-effect linkages among the identified hazards. For instance, PCS inverter failure may lead to battery overcharging which would subsequently cause the battery's temperature to rise beyond the safe range; then, if thermal fuses fails, then thermal runaway might happen and therefore leads to battery ignition. Nonetheless, this matrix can also be converted into fault trees if preferred.

Matrix CH captures which components are the direct/immediate causes of the hazards. For instance, C3.2 (PCS inverter) is directly responsible for H3 (Battery overcharged because it is the component that realizes (or actuates) charging/discharging even though in reality it is EMU who inappropriately commands PCS controller (which then commands PCS inverter) to continue charging the battery. Additionally, C4.2 (BMS Battery Pack) is immediately responsible for H1, H2, and H3 simply because it is the component where these hazards manifest themselves. In other words, this matrix indicates which component should be investigated first in case of hazard.

Matrix CC captures all necessary interactions between the system components. It can be useful to point out: firstly, which components are immediately affected in case of component malfunction/compromised/removal; for instance, in case of C3.2 malfunction/removal, directly affected components are C3.3 and C4.2 as their inputs are provided by C3.2; and secondly, which components could be directly causing a component to malfunction; for example, in case of C4.1 malfunction, immediate suspects would be C2 and 4.2 as they provide inputs to C4.1.

Matrix SC captures which components are the intended providers, receivers, or transmission medium of the necessary signals. In case of observing a signal being inappropriate, we can find out which components are its receivers, sources, or transmission mediums, and thereafter conduct checks on them if necessary. Depending on the system architecture, there could be multiple providers, receivers, or transmission mediums for the same signal.

Matrix RC captures which requirements might be violated in case of component malfunction/compromised. The purpose of this matrix is identical to that of [23]. The criticality of a component may be determined according to the number/criticality of requirements that may be violated due to its malfunction. The criticality of the components/requirements is a major factor in the mitigation strategy.

4.2.2. Matrices Constructed during *Step 7*

Matrix RR captures the potential conflicts between requirements. Ensuring confidentiality/availability/integrity is a common high-level security requirement, thereby giving rise to R1, R2, and R3. Another reason behind R3 is that the Wi-Fi of C5 is intentionally enabled to allow on-site maintenance engineers to remotely connect/access any device (including C2) within C5's network. Note that C2 is connected to C5 via Ethernet cable. Initially, no conflict was identified among them until C5's WPA2 (Wi-Fi Protected Access II) protocol is found hackable, and this thus implies potential violation of R2. Once successfully hacking C5, the adversary can perform various attacks on C5's network such as hacking SSH (Secure Shell) login of C2, and intercepting/forging/tampering with S8 and S9. Such attacks would lead to violating several requirements including R1. Simply put, satisfying R3 may violate R1 and R2, if no measure has been introduced to prevent/mitigate them; so, we can say that R3 conflicts with R1 and R2, as reflected by Matrix RR in Figure 8. Matrix MC captures which components are involved in implementing the measures. A measure could be realized by a single component alone or a combination of individual components. For example, M4 is a particular mechanism that leverages on a combination of C2, C3.1, and C4.1 to detect anomalous S3 or S7; this would help C2 to produce accurate S9, thereby contributing to satisfying R4. Matrix MR captures which measures would violate or help satisfying the requirements. For instance, as aforementioned that R3 conflicts with R1 and R2, applying M1 would of course help satisfying R1 and R2 while violating R3.

Suppose now we want to fulfil both R2 and R3. From Matrix MR, we can see that M1 would violate R3, and may thus consider it undesirable. As compared to M1, M2 seems

more attractive because it satisfies R2 while not violating R3. However, before jumping to conclusions that M1 should not be applied, we check: firstly, Matrix RR, to identify which requirements would potentially be violated; it shows that R1 and R2 are conflicted by R3; the number/criticality of requirements should be taken into consideration; secondly, Matrix RC, to identify which components could be affected due to the conflicted requirements; for instance, R1 violated would render C2's integrity questionable; to foresee the impact of compromised C2, we could refer to Matrix MC, Matrix SC or Matrix CC; thirdly, Matrix MC, to identify which measures might be compromised due to component malfunction/compromised; for instance, C2 malfunction/compromised would render M4 ineffective as M4 requires a functional combination of C2, C3.1 and C4.1; to foresee the consequence of ineffective measure (e.g., M4), we could refer to Matrix MR for its associated requirements; fourthly, Matrix SC, to identify which signals could be affected due to component malfunction/compromised; for instance, C2 malfunction/compromised would cause inappropriate S1 and S9 that subsequently affect their intended receivers, i.e., C1 and C3.1; fifthly, Matrix CC, to identify which components would be immediately affected due to component malfunction/compromised; for instance, C2 malfunction/compromised would directly affect C1, C3.1, C4.1, and C5.

Nevertheless, we could check on these matrices in different order, depending on which aspects of the *system* we take into consideration. After much consideration, our decision regarding M1 and R3 is described as follows. The decision criteria, typically including implementation cost and time, may vary on a case-by-case basis. In this case study, we consider that the advantages of applying M1 far outweigh its disadvantages. Securing C5 with M2 alone is not promising. M2, which imposes a longer complex password on C5, would only increase the time needed to crack C5's Wi-Fi. In other words, given enough time and additionally through social engineering, C5's Wi-Fi can still be hacked; as a result, the entire network of C5 may be compromised. In view of the current set of measures, we have determined to maintain M1 and nullify R3. Please note that the example shown in Figure 8 is a highly simplified version of the actual 8MD of the case study to demonstrate the application of the proposed methodology. Besides, some measures may be realized by utilizing the current set of components while some may introduce additional/new components to the system. Any later/resultant changes in the system architecture, including removal or addition of component or signal, would iterate the analysis from *Step 1*. Such iteration of analysis however could be accelerated by re-using the previously generated artefacts including 8MD. Based on the 8MD, analysts could narrow down the elements which need revision; subsequent revisions would update the 8MD.

5. Conclusions

STPA is a hazard analysis technique that has been widely applied in various domains including aerospace and automobile. In this paper, we review a selection of existing works which adopt or extend STPA for addressing both Safety and Security of CPS, and note two major gaps as follows: (a) little or no explicit guidance on deriving Security requirements; and (b) lack of discussion on overseeing the correlations between the derived Safety and Security requirements.

The proposed S&S analysis methodology aims to address the aforementioned gaps. Specifically, we derive both safety and security requirements from a template (depicted in Figure 4), and subsequently integrating them with the results of STRIDE analysis. It is worth mentioning that the presented template is not exhaustive but could be tailored for specific applications. For instance, a template tailored for an (offline) autonomous vehicle is probably inadequate for defining the S&S requirements for an CAV (Connected and Automated Vehicle); the latter, as it is online and tele-operated, it is likely to have greater attack surface and thus more cybersecurity concerns; to derive relevant templates with high level of granularity, domain/application-specific expertise is needed.

To facilitate overseeing the correlations between the derived S&S requirements (e.g., conflict resolution), we introduce an 8-Matrix Diagram (8MD); as illustrated in Figure 8, it consists of eight matrices that capture various correlations among five crucial aspects of the analysed system. We illustrate the proposed methodology using a case study of a realistic and commercial safety-critical CPS, and also discuss how 8MD is utilized for addressing a particular conflict between the derived requirements.

Nevertheless, this paper by no means intends to suggest that STPA outperforms other existing techniques (e.g., FTA, FMEA and attack trees/graphs) in terms of analysing hazards/threats. The authors believe that all these existing techniques are useful, as in general they help analysts perceive/deduce potential safety/security issues of the analysed system from different angles with different scopes. Additionally, some techniques may be user-friendly to one but complex to another. There is a learning curve in mastering any of these techniques.

As for future work, the authors shall explore enhancing the user-friendliness and comprehensiveness of the proposed S&S analysis methodology by (1) integrating aforementioned existing techniques besides STPA, (2) designing a systematic set of rules/instructions on applying the matrices, and (3) discussing how the S&S risks may be estimated.

Author Contributions: Conceptualization, L.-S.L. and G.S.; methodology, L.-S.L. and G.S.; validation, G.S., N.K.K., and C.-Y.W.W.; formal analysis, L.-S.L. and G.S.; investigation, L.-S.L.; resources, N.K.K. and C.-Y.W.W.; software, L.-S.L.; data curation, N.K.K. and C.-Y.W.W.; writing—original draft preparation, L.-S.L.; writing—review and editing, L.-S.L. and G.S.; visualization, L.-S.L.; supervision, G.S., N.K.K., and C.-Y.W.W.; project administration, N.K.K. and C.-Y.W.W.; funding acquisition, N.K.K. and C.-Y.W.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its National Cybersecurity R&D Programme (NRF2018-NCR003-0018) and administered by the Energy Market Authority (EMA) Singapore.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. *IEC 60812: Analysis Techniques for System Reliability—Procedure for Failure Mode and Effects Analysis (FMEA)*; Standards Australia & New Zealand: Sydney, Australia, 2006.
2. Lee, R.M.; Assante, M.J.; Conway, T. *ICS CP/PE (Cyber-to-Physical or Process Effects) Case Study Paper—German Steel Mill Cyber Attack*; SANS Institute: Bethesda, MD, USA, 2014.
3. Falco, M.D. *Stuxnet Facts Report (A Technical and Strategic Analysis)*; NATO Cooperative Cyber Defence Centre of Excellence: Tallinn, Estonia, 2012.
4. Lisova, E.; Šljivo, I.; Čaušević, A. Safety and Security Co-Analyses: A Systematic Literature Review. *IEEE Syst. J.* **2019**, *13*, 2189–2200. [CrossRef]
5. Kavallieratos, G.; Katsikas, S.; Gkioulos, V. Cybersecurity and Safety Co-Engineering of Cyberphysical Systems—A Comprehensive Survey. *Future Internet* **2020**, *12*, 65. [CrossRef]
6. Ishimatsu, T.; Leveson, N.; Thomas, J.; Fleming, C.; Katahira, M.; Miyamoto, Y.; Ujiie, R.; Nakao, H.; Hoshino, N. Hazard Analysis of Complex Spacecraft Using Systems-Theoretic Process Analysis. *J. Spacecr. Rocket.* **2014**, *51*, 509–522. [CrossRef]
7. Wróbel, K.; Montewka, J.; Kujala, P. Towards the development of a system-theoretic model for safety assessment of autonomous merchant vessels. *Reliab. Eng. Syst. Saf.* **2018**, *178*, 209–224. [CrossRef]
8. Lee, S.H.; Shin, S.M.; Hwang, J.S.; Park, J. Operational Vulnerability Identification Procedure for Nuclear Facilities Using STAMP/STPA. *IEEE Access* **2020**, *8*, 166034–166046. [CrossRef]
9. Sabaliauskaite, G.; Liew, L.S.; Zhou, F. *AVES—Automated Vehicle Safety and Security Analysis Framework*; ACM Computer Science in Cars Symposium; Association for Computing Machinery: New York, NY, USA, 2019. [CrossRef]
10. Sabaliauskaite, G.; Liew, L.; Cui, J. Integrating Autonomous Vehicle Safety and Security Analysis Using STPA Method and the Six-Step Model. *Int. J. Adv. Secur.* **2018**, *11*, 160–169.

11. Paul, S.; Rioux, L. *Over 20 Years of Research into Cybersecurity and Safety Engineering: A Short Bibliography*; WIT Press: Southampton, UK, 2015; pp. 335–349. [CrossRef]
12. Leveson, N. *Engineering a Safer World: Systems Thinking Applied to Safety*; MIT Press: Cambridge, MA, USA, 2011.
13. Young, W.; Leveson, N. Systems Thinking for Safety and Security. In *Proceedings of the 29th Annual Computer Security Applications Conference*; Association for Computing Machinery: New York, NY, USA, 2013; ACSAC '13, pp. 1–8. [CrossRef]
14. Young, W.; Leveson, N.G. An Integrated Approach to Safety and Security Based on Systems Theory. *Commun. ACM* **2014**, *57*, 31–35. [CrossRef]
15. Schmittner, C.; Ma, Z.; Puschner, P. Limitation and Improvement of STPA-Sec for Safety and Security Co-analysis. In *Computer Safety, Reliability, and Security*; Skavhaug, A., Guiochet, J., Schoitsch, E., Bitsch, F., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 195–209.
16. Kriaa, S.; Pietre-Cambacedes, L.; Bouissou, M.; Halgand, Y. A survey of approaches combining safety and security for industrial control systems. *Reliab. Eng. Syst. Saf.* **2015**, *139*, 156–178. [CrossRef]
17. Islam, M.M.; Lautenbach, A.; Sandberg, C.; Olovsson, T. *A Risk Assessment Framework for Automotive Embedded Systems*; Association for Computing Machinery: New York, NY, USA, 2016; CPSS '16, pp. 3–14. [CrossRef]
18. Schoitsch, E.; Schmittner, C.; Ma, Z.; Gruber, T. The Need for Safety and Cyber-Security Co-engineering and Standardization for Highly Automated Automotive Vehicles. In *Advanced Microsystems for Automotive Applications 2015*; Schulze, T., Müller, B., Meyer, G., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 251–261.
19. Cui, J.; Sabaliauskaite, G.; Liew, L.S.; Zhou, F.; Zhang, B. Collaborative Analysis Framework of Safety and Security for Autonomous Vehicles. *IEEE Access* **2019**, *7*, 148672–148683. [CrossRef]
20. Schmittner, C.; Ma, Z.; Schoitsch, E.; Gruber, T. A Case Study of FMVEA and CHASSIS as Safety and Security Co-Analysis Method for Automotive Cyber-Physical Systems. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*; Association for Computing Machinery: New York, NY, USA, 2015; CPSS '15, pp. 69–80. [CrossRef]
21. Sabaliauskaite, G.; Cui, J.; Liew, L.S.; Zhou, F. Modelling Safe and Secure Cooperative Intelligent Transport Systems. In *Complex Systems Design & Management Asia*; Cardin, M.A., Hastings, D., Jackson, P., Krob, D., Lui, P.C., Schmitt, G., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 62–72.
22. Sabaliauskaite, G.; Liew, L.S.; Zhou, F.; Cui, J. Designing Safe and Secure Mixed Traffic Systems. In Proceedings of the 2019 IEEE 19th International Symposium on High Assurance Systems Engineering (HASE), Hangzhou, China, 3–5 January 2019; pp. 222–227. [CrossRef]
23. Friedberg, I.; McLaughlin, K.; Smith, P.; Laverty, D.; Sezer, S. STPA-SafeSec: Safety and security analysis for cyber-physical systems. *J. Inf. Secur. Appl.* **2017**, *34*, 183–196. [CrossRef]
24. Shevchenko, N.; Chick, T.A.; O'Riordan, P.; Scanlon, T.P.; Woody, C. *Threat Modeling: A Summary of Available Methods*; Software Engineering Institute: Pittsburgh, PA, USA, 2018.
25. Piètre-Cambacédès, L.; Bouissou, M. Modeling safety and security interdependencies with BDMP (Boolean logic Driven Markov Processes). In Proceedings of the 2010 IEEE International Conference on Systems, Man and Cybernetics, Istanbul, Turkey, 10–13 October 2010; pp. 2852–2861. [CrossRef]
26. Kriaa, S.; Bouissou, M.; Colin, F.; Halgand, Y.; Pietre-Cambacedes, L. Safety and Security Interactions Modeling Using the BDMP Formalism: Case Study of a Pipeline. In *Computer Safety, Reliability, and Security*; Bondavalli, A., Di Giandomenico, F., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 326–341.
27. Egyed, A.; Grunbacher, P. Identifying requirements conflicts and cooperation: How quality attributes and automated traceability can help. *IEEE Softw.* **2004**, *21*, 50–58. [CrossRef]
28. Thomas, J. Extending and Automating a Systems-Theoretic Hazard Analysis for Requirements Generation and Analysis. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, UK, 2013.
29. Tabassum, M.R.; Siddik, M.S.; Shoyaib, M.; Khaled, S.M. Determining interdependency among non-functional requirements to reduce conflict. In Proceedings of the 2014 International Conference on Informatics, Electronics Vision (ICIEV), Dhaka, Bangladesh, 23–24 May 2014; pp. 1–6. [CrossRef]
30. Gu, T.; Lu, M.; Li, L. Extracting interdependent requirements and resolving conflicted requirements of safety and security for industrial control systems. In Proceedings of the 2015 First International Conference on Reliability Systems Engineering (ICRSE), Beijing, China, 21–23 October 2015; pp. 1–8. [CrossRef]
31. Hu, H.; Ma, Q.; Zhang, T.; Tan, Y.; Xiang, H.; Fu, C.; Feng, Y. Semantic modelling and automated reasoning of non-functional requirement conflicts in the context of softgoal interdependencies. *IET Softw.* **2015**, *9*, 145–156. [CrossRef]
32. Pereira, D.; Hirata, C.; Pagliares, R.; Nadjm-Tehrani, S. Towards Combined Safety and Security Constraints Analysis. In *Computer Safety, Reliability, and Security*; Tonetta, S., Schoitsch, E., Bitsch, F., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 70–80.
33. Salado, A.; Nilchiani, R. The Concept of Order of Conflict in Requirements Engineering. *IEEE Syst. J.* **2016**, *10*, 25–35. [CrossRef]
34. Procter, S.; Vasserman, E.Y.; Hatcliff, J. *SAFE and Secure: Deeply Integrating Security in a New Hazard Analysis*; Association for Computing Machinery: New York, NY, USA, 2017; ARES '17. [CrossRef]
35. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [CrossRef]
36. Procter, S. A Development and Assurance Process for Medical Application Platform Apps. Ph.D. Thesis, Kansas State University, Manhattan, KS, USA, 2016.

37. Avizienis, A.; Laprie, J.C.; Randell, B.; Landwehr, C. Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dependable Secur. Comput.* **2004**, *1*, 11–33. [CrossRef]
38. Temple, W.G.; Wu, Y.; Chen, B.; Kalbarczyk, Z. Systems-Theoretic Likelihood and Severity Analysis for Safety and Security Co-engineering. In *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*; Fantechi, A., Lecomte, T., Romanovsky, A., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 51–67.
39. Schmittner, C.; Gruber, T.; Puschner, P.; Schoitsch, E. Security Application of Failure Mode and Effect Analysis (FMEA). In *Computer Safety, Reliability, and Security*; Bondavalli, A., Di Giandomenico, F., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 310–325.
40. *NIST SP 800-30 Rev. 1 Guide for Conducting Risk Assessments*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2012.
41. Howard, G.; Butler, M.; Colley, J.; Sassone, V. Formal Analysis of Safety and Security Requirements of Critical Systems Supported by an Extended STPA Methodology. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS PW), Paris, France, 26–28 April 2017; pp. 174–180. [CrossRef]
42. Sabaliauskaite, G.; Adepu, S. Integrating Six-Step Model with Information Flow Diagrams for Comprehensive Analysis of Cyber-Physical System Safety and Security. In Proceedings of the 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), Singapore, 12–14 January 2017; pp. 41–48. [CrossRef]
43. Shevchenko, N.; Frye, B.R.; Woody, C. *Threat Modeling for Cyber-Physical System-of-Systems: Methods Evaluation*; Software Engineering Institute: Pittsburgh, PA, USA, 2018.
44. Soares, F.; Carvalho, L.; Costa, I.; Iria, J.; Bodet, J.M.; Jacinto, G.; Lecocq, A.; Roessner, J.; Caillard, B.; Salvi, O. The STABALID project: Risk analysis of stationary Li-ion batteries for power system applications. *Reliab. Eng. Syst. Saf.* **2015**, *140*, 142–175. [CrossRef]