Towards Customized Cyber Exercises using a Process-based Lifecycle Model

Tobias Pfaller
AIT Austrian Institute of Technology
Vienna, Austria
tobias.pfaller@ait.ac.at

Florian Skopik
AIT Austrian Institute of Technology
Vienna, Austria
florian.skopik@ait.ac.at

Paul Smith
Lancaster University
Lancaster, United Kingdom
paul.smith@lancaster.ac.uk

Maria Leitner
University of Regensburg
Regensburg, Germany
maria.leitner@ur.de

ABSTRACT

Cyber exercises enable the effective training of cyber security skills in a simulated, yet realistic, environment for a wide variety of professional roles. However, planning, conducting, and evaluating customized (i.e., non-standard) cyber exercise scenarios involves numerous time- and resource-intensive activities, which are still mostly carried out manually today. Unfortunately, the high costs related to these activities limit the practical applicability of cyber exercises to serve widely as a regular tool for skill development. Today, the flow of cyber exercise scenarios usually consists of predefined and meticulously planned injects (e.g. events) that are sequentially rolled out and thus drive the exercise. The composition of such injects resembles a linear process in its simplest form. Therefore, we argue that the utilization of existing, standardized, and well-researched methods from the business process domain provides opportunities to improve the quality of cyber exercises and at the same time reduce the workload necessary for planning and conducting them. This paper reviews the challenges related to conducting customized cyber exercises and introduces a process-based cyber exercise lifecycle model that leverages the power of process modeling languages, process engines, and process mining tools to transform cyber exercises into transparent, dynamic, and highly automated endeavors. We further describe the application of this lifecycle model in course of a proof-of-concept implementation and discuss lessons learned from its utilization at a large-scale national cyber exercise together with CERTs and authorities. While the state of the art mostly focuses on optimizing individual tasks or phases within the cyber exercise lifecycle, our contribution aims to offer a comprehensive integrated framework that spans across the phases, providing interfaces between them, and enhancing the overall effectiveness and maintainability of cyber exercises.

EICC 2024, June 05–06, 2024, Xanthi, Greece
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1651-5/24/06
https://doi.org/10.1145/3655693.3655713

CCS CONCEPTS

• **Security and privacy** → *Human and societal aspects of security and privacy*; • **Applied computing** → *Interactive learning environments.*

KEYWORDS

Cyber Exercise, Cyber Exercise Scenario, Cyber Exercise Lifecycle, Cyber Range, Process Engine

ACM Reference Format:
Tobias Pfaller, Florian Skopik, Paul Smith, and Maria Leitner. 2024. Towards Customized Cyber Exercises using a Process-based Lifecycle Model. In *European Interdisciplinary Cybersecurity Conference (EICC 2024), June 05–06, 2024, Xanthi, Greece.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3655693.3655713

1 INTRODUCTION

Many new technological trends, such as cloud computing, virtualization, mobile computing, or the Internet of Things, have emerged in recent years and have substantially changed the way how cyber security is organized today [27]. While in previous times, standard on-premise enterprise networks looked quite similar to each other, this has changed tremendously recently. There are simply too many technological and organizational options to choose from and the increasing complexity of interconnected infrastructures has led to countless individual solutions. Additionally, many traditional industry domains are still being digitalized, resulting in an increasing need to deal with new risks and cyber threats. As a consequence, an almost unmanageable number of new roles were created with flexible, if not entirely diminishing, boundaries between technological areas, organization, and management. While cyber security was long seen as an administrator's job, today, a professional environment demands for appropriate governance and management roles.

The NICE framework [26] is the result of a systematic collection of cyber security roles and their required skills and currently consists of 52 entries. We argue that the emergence of all these new cyber security roles combined with their peculiarities in the different industry domains demand for individual cyber security training to improve incident response capabilities, forensic analysis skills, or simply staff awareness.

Today, a well-known and effective way to acquire and improve vital skills are specialised training activities in course of exercises on cyber ranges [5, 18]. The flow of these cyber exercises usually consists of predefined and meticulously planned injects (e.g. events) that are linearly rolled out to challenge the participants and thus drive the exercise [17]. The composition of such injects resembles a linear process in its simplest form. Therefore, we argue that the utilization of existing, standardized, and well-researched methods from the business process domain provides opportunities to increase the quality of cyber exercises.

We pick the ENISA exercise lifecycle [24], meant as a template for designing, implementing and delivering cyber exercises, as a starting point. This lifecycle provides a common structure and defines phases for identifying relevant aspects of exercises, as well as planning, conducting and evaluating them. Today, cyber exercises are still not widely adopted as they require numerous manual tasks to be carried out, which is costly and resource-intensive. We aim at expanding on ENISA's initial concept and investigating challenges and potential solutions to making cyber exercises more attractive for a larger number of users. In order to achieve this, we investigate means to automate several aspects of cyber exercises over their lifecycle, making them more adaptable, repeatable, customizable as well as effectively deliverable.

To sum up, the contributions of this paper are three-fold:

- **Challenges and Approaches.** This paper reviews the challenges related to conducting customized cyber exercises and introduces a process-based cyber exercise lifecycle model that leverages the power of process modeling languages, process engines, and process mining tools to transform cyber exercises into transparent, dynamic, and highly automated endeavors.
- **Proof-of-Concept Implementation.** We further describe insights into technical implementations of an integrated approach for delivering cyber exercises and discuss potential advantages compared to the state of the art.
- **Case Study.** We describe the application of this proof-of-concept in the course of a large-scale national cyber exercise together with CERTs (Computer Emergency Response Teams) and authorities and derive valuable lessons learned.

The remainder of the paper is organized as follows. Section 2 outlines important background and related work. Then, Sect. 3 summarizes the identified challenges as well as potential solutions from the literature. Section 4 describes a proof-of-concept (PoC) implementation of some of the outlined solutions that together realize an integrated version of ENISA's lifecycle model for cyber exercises. We applied this PoC in the course of a case study, which is described in Sect. 5, and discuss major findings. Finally, Sect. 6 concludes the paper.

2 BACKGROUND AND RELATED WORK

Cyber exercises can take place in a simulation environment that allows for training and testing cyber security capabilities. In that context, cyber security capabilities comprise anything that contribute to an improved cyber security posture, as for example, technical skills and cyber security awareness from an individual's perspective or incident response processes and communication structures from an organisational perspective. Depending on which capabilities should be trained, cyber exercises vary regarding their type and may range from abstract table-top exercises [2] to deep technical exercises [39][30][15], or even have a hybrid form [35][6][4].

The core of a cyber exercise consists of a (typically) fictional, yet realistic, scenario. The scenario drives the cyber exercise and defines the environment and roles into which the participants immerse themselves. According to Wen et al. [37] a cyber exercise scenario consists of a (1) *Scenario Information Model*, that contains high-level information about the scenario topic, a (2) *Scenario Operation Model* that represents the exercise infrastructure, and the flow of injects, and a (3) *Security Knowledge Model* that contains the security knowledge that should be conveyed to participants. Cyber exercises enable to test and train the responses to various cyber incidents within a practical, yet secure and realistic, environment. Therefore, they represent an excellent learning environment for cyber security professionals [12] and can be used to significantly increase their skill-level [13]. However, planning a cyber exercise is a cumbersome process that can last several months [35], in which a great deal of experience, technical know-how, and creativity is required. Furthermore, according to Yamin and Katt [41], existing inefficiencies in the cyber exercise lifecycle hinder the smooth planning, execution and evaluation of cyber exercises, and limit their ability to widely serve as a tool for cyber security skill development.

Therefore, some authors have already introduced lifecycles for conducting cyber exercises [10, 29, 35]; however, in this paper we pick the widely adopted ENISA lifecycle [24] as a basis (see Fig. 1).

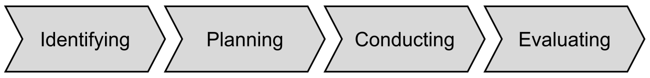


Figure 1: ENISA lifecycle (redrawn) [24]

In the *Identifying* phase of the ENISA lifecycle, the objectives for an exercise are defined, along with initial high-level design decisions (such as exercise type and rough scenario ideas). The *Planning* phase involves detailed planning (such as infrastructure and scenario development, and organizational decisions). The *Conducting* phase encompasses the actual execution of the exercise, and the *Evaluating* phase involves evaluating the participants as well as the scenario itself and creating final reports.

Researchers already address issues related to certain phases of the lifecycle and contribute associated solutions. When it comes to planning and conducting, Doupé et al. [8], for example, presented a novel cyber exercise design containing missions that were executed utilizing Petri nets. Skopik and Leitner [31] argue that the design of linear cyber exercise scenarios is a simplification that could be problematic in complex cyber exercises. Therefore, they propose an extension with the patterns playback, forking, pause/adapt/repeat, and fast-forward. Researchers developing the AIT Cyber Range [18] implemented a scenario engine called GameMaker, that is used to automatically handle a sequential scenario flow and execute non-technical injects, such as participant instructions or email communication. When it comes to evaluation, White et al. [36] argue that simple technical metrics like just measuring if a task was solved in cyber exercises does not give a clear indication of whether the task was fully understood nor whether the learning objectives were achieved. Andreolini et al. [1] developed a framework to model the behavior of participants in a graph. They compare the graphs to those of other participants as well as to a baseline in order to measure performance. White et al. [36] also created graphs from the bash history of participants to get more detailed insights into how they approached different problems they were challenged with.

In particular, the argument of Skopik and Leitner [31], that linear cyber exercise scenarios are a problematic simplification, is highly interesting for our research. The patterns they propose for introducing greater complexity into cyber exercises can also be represented in the form of workflow patterns [34], supported by existing process modeling languages such as BPMN [25]. Consequently, by using a process-like structure to depict complex cyber exercise scenarios, advantages could arise by leveraging Business Process Management (BPM) [38], which offers a comprehensive toolkit of concepts, methods, and techniques for supporting the design, administration, configuration, execution and analysis of processes. This allows for following a BPM lifecycle [9] and utilizing its existing and well-researched tools, including, for example, process engines [20] for automated scenario execution, or process mining [32] for evaluation.

3 CHALLENGES AND SOLUTIONS

To individually tailor cyber exercises to the skill levels of different participants or teams, a high degree of flexibility is required, leading to challenges in the planning, conducting, and evaluating phases of the cyber exercise lifecycle (according to ENISA - see Sect. 2), which will be explained below.

3.1 Challenges

Cyber exercises are currently executed in a linear fashion, leading to several simplifications that may decrease their realism and thus their use for the participants. As argued by Skopik and Leitner [31], this approach is a limiting simplification, and cyber exercises should be designed in a more complex manner to enhance the learning outcomes. Based on current literature, expanded by our experiences, we identified the challenges that currently exist in cyber exercises, as outlined in the rest of this section.

(1) **Adaption for different skill levels.** Cyber exercises are extensive training events in which participants of varying skill levels take part [4]. Nevertheless, they adhere to a predefined scenario, which, today, predominantly involves a linear sequence of injects [31]. In a linearly planned exercise, accounting for the individual skill levels of participants proves challenging, leading to persons with high skill levels potentially being underchallenged, while those with low skill levels may be easily overwhelmed. For exercise organizers, it is, therefore, a challenge to make the exercise flexible and adapt it to the progress of each team or participant, thus achieving an appropriate difficulty level for every skill level.

(2) **Clear representation of complex scenarios.** The design of cyber exercises is a process involving numerous steps with different tasks and people [22]. When executing the exercise, additional supporters and observers are frequently involved, each of whom must possess a solid view on the scenario. This complexity is further increased when accommodating varying skill levels. Consequently, ensuring a coherent representation of the cyber exercise scenario that enables a comprehensive visualization of the intended flow remains a considerable challenge.

(3) **Reusability of parts of a cyber exercise.** The planning and execution of a cyber exercise is a highly time- and resource-intensive task [35]. Due to the rapidly evolving nature of cyber security, typical attack vectors, and vulnerabilities, cyber exercise organizers are challenged to continually develop and improve their scenarios. To avoid having to plan a scenario with different learning objectives from scratch each time, it is a challenge to efficiently reuse parts of existing exercises without the need for extensive modifications.

(4) **Reduction of complexity during execution.** The execution of a cyber exercise is complex, often erratic, and involves many ad-hoc decisions. Organizers must orchestrate the scenario, which, with an increasing number of teams and the expanding scope of the scenario, significantly escalates the complexity of the execution. This complexity demands extensive observation and scenario management to stay current and deliver the right inject at the right time. It is a significant challenge to reduce this complexity.

(5) **Flexibility during execution.** In a linearly planned cyber exercise, if participants take unexpected actions, it is challenging to react appropriately. It is a significant challenge to make an exercise highly flexible, maintaining some room for maneuver during the execution to be able to respond in an ad-hoc manner to unexpected responses from the participants.

(6) **Traceability of exercise processes**. The actions of participants in cyber exercises are often not transparent and are difficult to determine in detail [36]. Often, exercise organizers try to gain a comprehensible insight into the progress of participants through manual observers, questionnaires [13], or reports. All these means cause costs. Increasing traceability by technical means is an important challenge in order to offer participants appropriate, yet cost-effective, feedback.

(7) **Individual evaluation.** When the participants' progresses and experiences diverge due to their individual pace, a uniform evaluation is not possible. Therefore, individual evaluations must be conducted, which represents a complex and resource-intensive challenge.

3.2 Applicable Solution

In order to overcome the challenges stated in Sect. 3.1, we propose to harness methods and techniques from business process management to plan, conduct and evaluate cyber exercises. Fig. 2 provides an overview of our approach based on the phases of the ENISA lifecycle, extended by process-based concepts that enable seamless transitions between the phases. A process model, which is the output of the planning phase, serves as input for the conducting phase, and exercise logs, that occur during conducting, serve as input for evaluation.

3.2.1 Identifying. According to the ENISA lifecycle [24], the identifying phase consists of identifying the requirements for a cyber

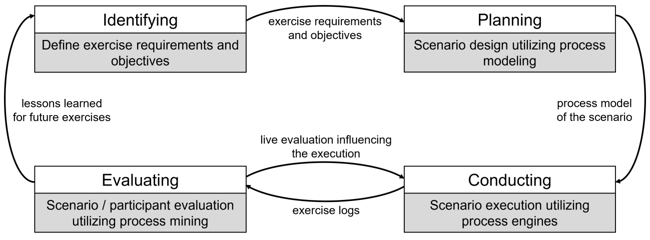

Identifying
Define exercise requirements and objectives
exercise requirements and objectives
Planning
Scenario design utilizing process modeling
lessons learned for future exercises
process model of the scenario
live evaluation influencing the execution
Evaluating
Scenario / participant evaluation utilizing process mining
exercise logs
Conducting
Scenario execution utilizing process engines

Figure 2: Process-Based Cyber Exercise Lifecycle - Overview

exercise and choosing an appropriate exercise type, size and high-level scenario. For this purpose, general methods from requirements engineering [7, 16] are applicable.

3.2.2 Planning. In the planning phase of a cyber exercise, our approach suggests utilizing process modeling languages to model the sequence of injects in a cyber exercise. The tasks of the process correspond, when used in the context of cyber exercises, to the delivered injects. This allows for overcoming the following challenges:

(1) **Adaption for different skill levels.** By utilizing workflow patterns provided in process modeling languages, such as BPMN [25] or Event-driven Process Chains (EPCs) [14], complex scenarios with alternative paths can be developed. Consequently, challenging paths can be created for participants with higher skill levels, while additional assistance or even simpler paths can be developed for participants with lower skill levels. Depending on how participants progress in the scenario, a more difficult or easier path is chosen. To adapt a cyber exercise through alternative paths based on participants' progress, this progress must also be measured and evaluated. In our approach, we complement each decision with an indicator that, depending on whether it exceeds certain thresholds or not, determines which specific path is chosen. The values of indicators can be determined, for example, through manual observations or extracted automatically from the observed infrastructure (e.g. log files that capture the participants' actions).

(2) **Clear representation of complex scenarios.** Specifically, the design of complex scenarios poses challenges in presenting them clearly [42]. The use of process modeling languages allows for the representation of complex cyber exercises in a comprehensible manner and also leverages existing standards and guidelines [23].

(3) **Reusability of parts of a cyber exercise.** Current process modeling standards also allow the use of subprocesses [25]. By using this concept, one can outsource a part of the exercise (e.g., an attack vector) into subprocesses and thus integrate them into multiple exercise scenarios. This allows for the reuse of parts of a cyber exercise in different settings.

3.2.3 Conducting. After designing cyber exercises with process modelling languages in the planning phase, the application of process engines for delivery in the conducting phase is obvious.

(4) **Reduction of complexity during execution.** Specifically, due to the fact that our approach adapts cyber exercises to participants' skill levels and utilizes alternative paths, the complexity of the execution increases significantly. In addition to the already complex task of orchestrating a scenario, decision indicators (a measurable value that serves as input for decisions within the model) must be observed and evaluated, and alternative paths must be considered. Since we already use process modeling languages in the planning phase, the use of a process engine [20] is ideal for overcoming the mentioned challenges. By utilizing a process engine, the process model developed in the planning phase can be instantiated and executed separately for each team and/or participant. Additionally, the values of the decision indicators, can be stored for each team in its own runtime variables. This allows for the automation of the scenario flow, and even complex scenarios that adapt individually to the team's progress, can be executed with a high degree of automation.

(5) **Flexibility during execution.** By planning different paths, a cyber exercise can become highly flexible, and different paths can be chosen based on the participants' progress. That allows to prepare paths for exceptional situations, enabling automatic exercise flow adaptations in such cases. However, in case an unforeseen situation arises, where no special path is prepared process engines allow for pausing the scenario execution, adjusting the subsequent elements (injects, decisions, paths, etc.), and afterwards restarting the process from a desired point [21]. Thus, there are no limits to flexibility, and cyber exercise organisers can respond appropriately to unexpected situations.

3.2.4 Evaluation. The evaluation phase may make heavy use of event logs and infrastructure artifacts, as well as the logs produced by the process engine that delivered the exercise injects, to (semi-)automatically evaluate the exercise success and the actions of participants.

(6) **Traceability of exercise processes.** Event logs [11] are chronological recordings of events captured within a process flow. They allow for meticulously document when an inject was sent, as well as additional information, such as the values of the runtime variables evaluated at (or before) decision points. Based on these logs, we can track the progress of a team during the exercise, and even compare the exercises' outcomes' across teams.

(7) **Individual evaluation.** The individual information from the event logs can be utilized to provide teams with detailed feedback on their progress in the exercise. Additionally, event logs can serve as input for process mining analyses to identify typical behavioral patterns, positive or negative outliers, or the degree of deviation from a predefined behavioral baseline [32].

4 PROOF-OF-CONCEPT IMPLEMENTATION

To demonstrate the practical applicability of the approach we presented, a proof-of-concept implementation was developed and implemented in the context of a national exercise (see Sect. 5). For this purpose, we use a small part (specifically, a web defacement attack vector) of the exercise scenario to illustrate the implementation of our concepts within the planning, conducting, and evaluating phases of the ENISA lifecycle. We place particular emphasis on demonstrating how the mechanisms applied in each phase of the lifecycle address the challenges mentioned in Sect. 3.1 and interconnect across phases.

4.1 Planning

Within the planning phase, the scenario of the exercise is represented in a process model. Therefore, we use the acknowledged and standardized process modeling language BPMN [25]. The upper part of Fig. 3 shows the entire implemented model. The concepts used within that model are described in the following sections.

4.1.1 Subprocess. Within the model, the first task “Web defacement preparation" represents a subprocess that encapsulates an attack chain [40], which encompasses a sequence of actions such as scanning, reconnaissance, payload delivery and exploitation. By using a subprocess at this point, the underlying attack vector can be flexibly exchanged on the one hand and can additionally be integrated into other exercise scenarios on the other.

4.1.2 Injects. The remaining tasks in the model comprise injects that serve the purpose of either conveying information (or technical artifacts) to participants or requesting specific information from the cyber exercise infrastructure, thus facilitating an adaptive environment throughout the scenario.

4.1.3 Timer events. The timer events are used to insert waiting times during the scenario.

4.1.4 Decisions and decision indicators. The decisions are intended to enable adaptive routing based on the participants' progress. Before each decision, a decision indicator is queried to utilize its value to decide for certain routes. At the task “Check whether participant(s) detected the web defacement", for example, we aim to determine, based on the web server log files, whether the participant has made a GET call to the website's index page since the time of the defacement. If this has occurred, we assume that the web defacement has been detected, otherwise we assume that it hasn't been detected. This information is used for the following decision, as stated in the process model. Before the second decision, the task “Check existence of defacement and backdoor", we inspect the web server's file system (either through automated methods, such as autitd rules or through manual observations) to verify if the defaced index page is still present or has already been replaced, and whether the backdoor that was placed on the system (in the subprocess “Web defacement preparation") is still present or has been removed. These decision indicators are again used to handle the following decisions, where three possible routes are available:

(1) *Neither the defacement has been removed nor the backdoor deleted:* We will revert and wait for an additional 5 minutes to allow participants more time.
(2) *The defacement has been removed, and the backdoor has been deleted:* The exercise part is finished.
(3) *The defacement has been removed, but the backdoor has not been deleted:* The process will return to the “Replace index page" task since the attacker still has the opportunity to perform another defacement through the remaining backdoor.

By applying a process modeling language in this example, (1) the scenario is adaptively tailored to the skill level of participants, (2) despite its complexity with various paths and decisions, an understandable and clear model of the scenario is created, and (3) the use of subprocesses ensured the reusability of exercise parts (an attack vector, in our case) in other exercises. Moreover, the subsequent conducting phase uses the process model as input for execution, therefore providing a seamless connection between these two phases.

4.2 Conducting

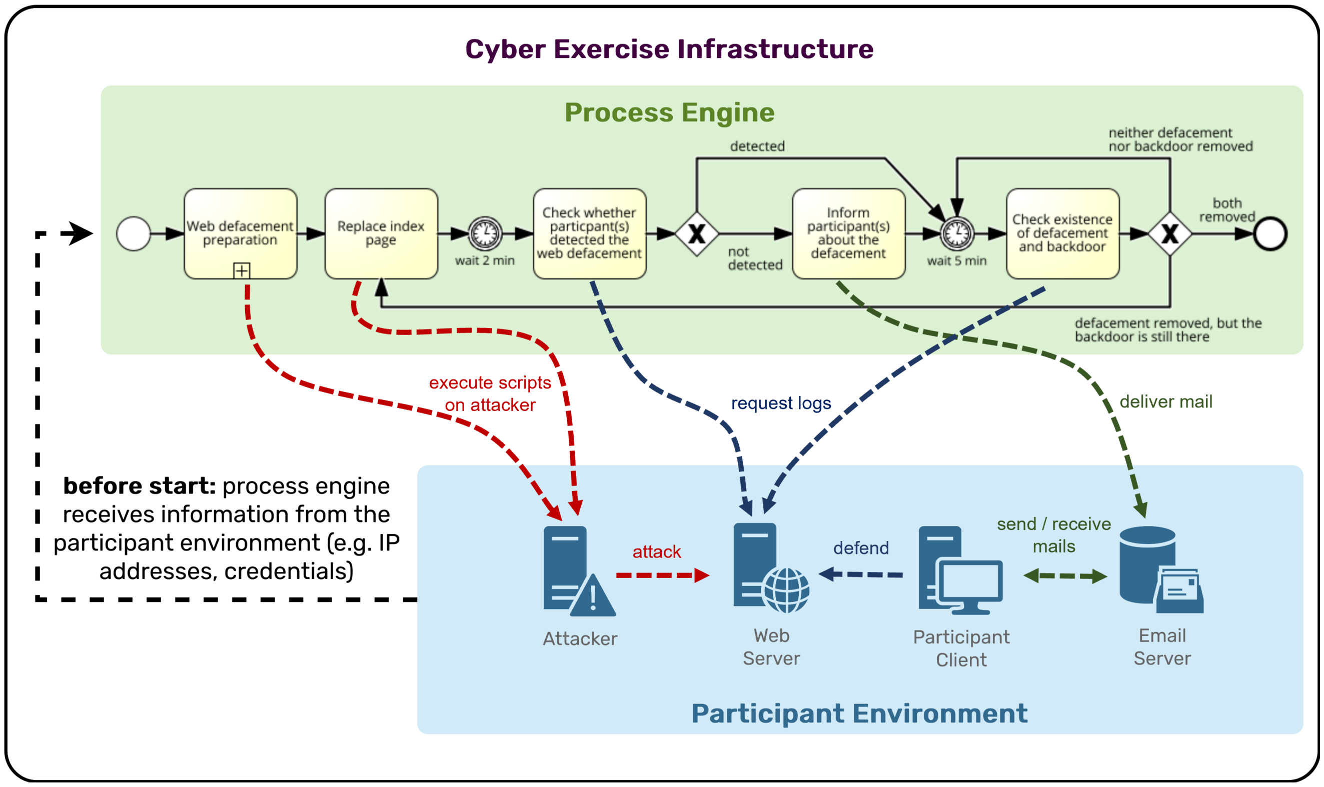
To execute the scenario model developed during the planning phase, we have developed a custom-built process engine called “flowgin", which handles the flow of given process models and supports the concepts mentioned in the planning phase (i.e., subprocesses, injects, timer events, decisions and decision indicators / runtime variables). Additionally, it has the capability to instantiate multiple processes simultaneously and manage variables during runtime.

Flowgin's primary purpose is to manage the flow of injects and does not engage in any other tasks. The execution of an inject is achieved through a REST call. Figure 3 serves as an illustration, showcasing the execution process by highlighting the interaction between the scenario flow executed within the process engine and the participants' exercise environment. For the sake of simplicity, the actual exercise infrastructure has been represented in a simplified manner, featuring only those machines/servers relevant to the website defacement attack vector.

To conduct this scenario individually for each team, flowgin instantiates separate process instances for them. However, since each team possesses its own infrastructure, including web servers, mail servers, attackers (and more) with different IP addresses, users and credentials (e.g. for the mail server login), it is necessary to prefill the model with these variables (e.g. attacker = 172.16.0.32; web-server-ip = 10.0.0.5).

Once all the necessary variables have been provided to the instantiated process, it starts executing and begins with enacting the injects in the specified order. The first task executed is the “Web defacement preparation" subprocess. When starting a subprocess, flowgin instantiates a new process, passes the required parameters (such as the attacker's IP and the target's IP), and starts it. Once the subprocess is completed, the main process continues. The next step is to use the backdoor placed on the web server (realized by the subprocess before) to replace the index page with a defaced page. This is achieved by flowgin making a REST call to a small web service running on the attacker, that receives the call and executes a script that connects to the web server (via the backdoor) and replaces the index page.

After waiting for a duration of two minutes, the next inject, "Check whether participant(s) detected the web defacement," is executed. Therefore, the web server logs are queried to determine if a GET call has been made to the defaced index page. If a corresponding GET call is detected, the runtime variable "defacement" (which was initially set to "not detected") is set to "detected". Then, the runtime variable "defacement" serves as input for the following decision. If the defacement was detected, the process continues with a 5-minute wait period. If it was not detected, the inject "Inform



Cyber Exercise Infrastructure
Process Engine
Web defacement preparation
Replace index page
wait 2 min
Check whether particpant(s) detected the web defacement
detected
not detected
Inform participant(s) about the defacement
wait 5 min
Check existence of defacement and backdoor
neither defacement nor backdoor removed
both removed
defacement removed, but the backdoor is still there
execute scripts on attacker
request logs
deliver mail
before start: process engine receives information from the participant environment (e.g. IP addresses, credentials)
attack
defend
send / receive mails
Attacker
Web Server
Participant Client
Email Server
Participant Environment

Figure 3: Interplay between process engine and cyber exercise infrastructure

participant(s) about the defacement" is executed beforehand. This inject technically sends a REST call to a web service on the email server, including the sender (with associated credentials for the email server), recipient, subject and content of an email. The email server's web service receives this information, logs onto the email service, and delivers the desired email to the recipient's mailbox. Afterwards, the 5-minute wait period is applied. Then, a check on the file system is performed to determine if the defacement or the backdoor still exist and the results are accordingly stored in runtime variables. These runtime variables are used afterwards to make the following routing decision.

The entire flow, including the path decisions, is automated. The only manual aspect is gathering information from the infrastructure, such as determining defacement detection and checking for removal of the defacement or the backdoor. We opt for manual requests to showcase practicality while keeping the technical infrastructure as it was before. The results of these manual requests are passed to the process flow by adjusting runtime variables.

The utilization of a process engine allows for the automation of scenario execution, which (1) significantly reduces the complexity of conducting an exercise despite adaptive adjustments through various paths and (2) still maintains the necessary flexibility to respond to unexpected reactions of participants. During execution, event logs are collected, which represent a fundamental basis for the subsequent evaluating phase.

4.3 Evaluating

During the execution of the cyber exercise using flowgin, event logs are generated. For this purpose, a log event is written to a log file for each inject execution, containing the event name, timestamp, and the current values of runtime variables. Since each team has its own instantiation of the scenario in flowgin, these logs can be examined individually for each team. Such event logs serve as input for process mining, allowing for both visualization and analysis of the processes followed by the teams. Beside of process mining, observable runtime variables and associated event logs serve as basis for calculating key performance indicators such as task detection / completion times, mean times per task, number of successful attacks, and more. These indicators are used as input, to extract evidence-based data from the learning environment [19] and are utilized to generate proper feedback. Therefore, runtime variables are carefully selected during the planning phase and filled during the conducting phase, where they also serve as decision indicators for choosing different paths. In the evaluation phase, they are used for analysis, applying process mining algorithms, or calculating key performance indicators. Therefore, the proper selection of runtime variables is crucial to facilitate a smooth transition between the phases of the cyber exercise lifecycle.

Listing 1 shows an example event log in the form of a YAML [3] file. Lines 1-3 contain general information about the process that is executed. Starting from line 5, all events are displayed in the order in which they where executed during process enactment.

```
instance-id: 100
instance-name: "Web Defacement Exercise Part"
...
events:
  - ...
  - event-name: "Check whether participant(s)
      detected the web defacement"
    event-timestamp: "2024-01-28T14:30:00"
    runtime-variables:
        - web-defacement: "detected"
        - backdoor: "not resolved"
  - ...
```

Listing 1: Event Log

In context of our proof-of-concept, event logs were created, but process mining techniques such as process discovery [33] or conformance checking [28] were not applied due to the small number of teams involved. We will place a particular emphasis on creating use cases involving a larger number of teams in our future endeavors, to strengthen our argumentation with applicable examples for process mining.

By individually generating event logs, the behavior of participants in cyber exercises is (1) traceable and comparable with other exercise runs, and allows for (2) individual evaluation of participants or teams, resulting in more detailed feedback.

5 CASE STUDY

The proof-of-concept shown in Sect. 4 was applied in a representative case study during a large-scale national cyber exercise. The subsequent sections provide information about the cyber exercise itself, and the use case that was conducted.

5.1 National Cyber Exercise

The exercise involved four teams, each comprising 8-10 participants. The teams consisted of a well-balanced composition, with approximately half of the participants being technical experts and the remaining half holding managerial positions. The team members were essentially unfamiliar with each other, with the exception of Team D, where some participants had prior experience working together and demonstrated excellent coordination. In a fictitious scenario, each team represented a critical entity within a supply chain, spanning the manufacturing, transportation, retail, and IT service provider sectors. Consequently, each team managed its own respective infrastructure. Over a five-hour period, the exercise subjected the teams' infrastructures to various attack vectors, including website defacements, SQL injections, cross-site scripting, and misconfigurations.

The teams' objective was to effectively respond to these attacks through appropriate incident response management and communication, thereby defending against the assaults and preserving the integrity of the supply chain. Additionally, personnel from national Computer Emergency Response Teams (CERTs) and governmental institutions provided support and were present throughout the exercise.

5.2 Use Case

The web defacement attack vector was used as a small part of the extensive national cyber exercise to demonstrate our approach. The entire sequence of the case study can be traced based on the entries in Table 1. Since the exercise flow was slightly different for each team, the execution times of the web defacement attack varied, but they were performed in the same manner and on the same day. The different teams are labeled as A, B, C, and D for demonstration purposes.

Since each team had its own exercise infrastructure and different attacker machines existed, the following information was provided to the process (in the form of variables) before its instantiation:

- Attacker's IP address (240.172.53.0; The same attacker machine was used for all teams. If a team would have already blocked this attacker, the process could be restarted with a different attacker IP).
- Web server domain (www.A-D.com)
- Email server IP (10.0.1-4.100)

These IP addresses and domains are fictitious elements that exist only within the exercise infrastructure. Additional credentials for the email server did not need to be provided because a master access was implemented, which is universally valid. Therefore, this information does not need to be dynamically provided at the process start but can be statically integrated into the body of the corresponding inject (i.e., "Inform participants about web defacement") beforehand.

After all the necessary variables were passed to the process, it was started. Table 1 shows the injects with associated timestamps and the development of the runtime variable *defacement* (which can take three different values: *not detected*, *detected* and *removed*). In line one of Table 1, the execution time of the first inject (i.e., subprocess "Web defacement preparation") is depicted. After 1-3 minutes (depending on how long the subprocess took), the preparation for the web defacement was successfully completed, and the next inject "Replace index page" was executed. After a two-minute wait (as foreseen in the process model), it was checked whether the participants had already detected the defacement. Therefore, a person from the organising team checked in the web server logs if after the defacement there has already been a GET call to the web server. In our case study, three out of the four teams had already detected the defacement. Only Team D had not yet discovered it, which is why they were the only team that received an information email about the defacement (see line number 4). By sending out this information email, the runtime variable was set to *detected.*

After a 5-minute wait (as foreseen in the process model), a check was made to see if the teams had already removed the defacement or the backdoor. Since no team discovered and removed the backdoor throughout the entire exercise, we will not place further emphasis on the backdoor in this use case. As shown in line 5, no team was able to remove the defacement within the first 5 minutes. Teams A, B, and D were able to remove it after 10 minutes. Team C removed it after a total of 20 minutes.

For the occurred case in which no team removed the backdoor, the original plan of the process entailed was a repetition of the defacement. Nevertheless, given the considerable challenges faced by the teams owing to prior attacks, a collective decision was made

Table 1: Event log analysis for teams

Number	Task	Team A	Team B	Team C	Team D
1	Web defacement preparation	09:42:38	09:43:25	11:27:23	13:39:49
2	Replace index page	09:44:22	09:45:59	11:28:48	13:40:38
3	Check whether participant(s) detected the web defacement	**09:46:22** defacement = detected	**09:47:59** defacement = detected	**11:30:48** defacement = detected	**13:42:38** defacement = not detected
4	Inform participants about the web defacement	-	-	-	**13:42:53** defacement = detected
5	Check existence of defacement and backdoor	**09:51:22** defacement = detected **09:56:22** defacement = removed	**09:52:59** defacement = detected **09:57:59** defacement = removed	**11:35:48** defacement = detected **11:50:48** defacement = removed	**13:47:53** defacement = detected **13:52:53** defacement = removed
6	Replace index page	-	-	-	**14:10:41** defacement = not detected
7	Check whether participant(s) detected the web defacement	-	-	-	**14:12:41** defacement = detected
8	Check existence of defacement and backdoor	-	-	-	**14:17:41** defacement = removed

to refrain from executing another defacement for Teams A, B and C. However, it was evident that Team D remained relatively unchallenged, so we decided to resume their process at 2:10:41 PM. Consequently, the index page was subjected to defacement once again, and the runtime variable *defacement* was set back to *not detected*. This time, the team promptly detected the defacement and was able to remove it within the initial five minutes. Nonetheless, the backdoor was still present.

In accordance with our approach, we employed the gathered event logs to conduct objective assessments and offer appropriate feedback to the participants. Accordingly, we have determined, that in essence, all teams exhibited very similar performance levels during the exercise, with comparable response times to detect and resolve the attacks. Teams A, B, and D notably succeeded in removing the website defacement within 10 minutes of detection, while Team C required 20 minutes to accomplish the same task. However, none of the teams were able to identify and eliminate the backdoor vulnerability. Therefore, given the overarching context of the exercise, which also encompassed other attack vectors beyond the scope of this specific use case, it became evident that Team D, in particular, delivered an exceptional performance. This achievement may be attributed to pre-existing familiarity and professional collaboration among some team members, while the other teams were randomly compiled with people from different organizations. Consequently, Team D was the only team for which we reintroduced the website defacement to demand its participants with additional challenges and prevent them from being under-challenged.

Planning the use case in a process model incorporating various paths and decision variables in combination with executing it in a process engine significantly improved the efficiency of the cyber exercise. This dynamic approach ensured that teams facing minimal challenges were demanded with additional tasks, while those grappling with overwhelming obstacles were either provided with helpful guidance or spared from additional burdens. The participants of Team D, in particular, serve as a compelling example that demonstrates how they might have been under-challenged in a linearly structured cyber exercise but got an additional task assigned thanks to our flexible approach, ensuring they were appropriately challenged. Therefore, we created a customizable exercise environment while ensuring a high degree of automation to accommodate rapid adaptations.

6 CONCLUSION AND FUTURE WORK

We have developed a process-based lifecycle model for planning, conducting, and evaluating customized cyber exercises. We have used the acknowledged ENISA cyber exercise lifecycle and extended it with methods from Business Process Management in order to improve the effectiveness of cyber exercises as well as increase their level of automation. For this purpose, we utilized a process modeling language to plan an adaptive cyber exercise scenario, whose model is then executed in a process engine. During conducting the exercise, comprehensive event logs, including runtime variables, are documented. This provides a solid foundation for the application of process mining algorithms in the evaluating phase. Additionally, our approach is holistic and enables seamless transitions between phases of the lifecycle. The process model from the planning phase is directly executed by a process engine during the conducting phase, and the logs from the conducting phase are then used as input for process mining techniques in the evaluating phase.

To demonstrate the practical applicability of our approach, we implemented a small part of a large-scale national exercise with the mechanisms outlined in this paper. Two significant improvements were identified:

(1) **Increase the quality of cyber exercises:** Our approach enables the development of more complex exercises that include different paths to adapt flexibly to participants' skill

levels. Furthermore, the individual assessment of teams and their chosen paths through a scenario allows for more targeted feedback.

(2) **Increase the efficiency and effectiveness of conducting cyber exercises.** The clear representation of cyber exercises in a process model allows for a more straightforward depiction of complex scenarios. Moreover, the use of subprocesses enhances the reusability of exercise components, and the utilization of process engines significantly increases automation and thus efficiency.

For future work, we plan to expand our approach in all phases of the ENISA lifecycle to conduct increasingly extensive and more complex cyber exercises. Our aim is to continuously enhance the exercise experience for participants while simultaneously increasing the level of automation, therefore, maximizing the potential for a wide range of users to apply cyber exercises as a feasible means for skill development.

ACKNOWLEDGMENTS

This work was partly funded by the FFG project STATURE (grant no. FO999888556).

REFERENCES

[1] Mauro Andreolini, Vincenzo Giuseppe Colacino, Michele Colajanni, and Mirco Marchetti. 2020. A framework for the evaluation of trainee performance in cyber range exercises. *Mobile Networks and Applications* 25, 1 (2020), 236–247.

[2] Giddeon N Angafor, Iryna Yevseyeva, and Ying He. 2020. Game-based learning: A review of tabletop exercises for cybersecurity incident response training. *Security and privacy* 3, 6 (2020), e126.

[3] Oren Ben-Kiki, Clark Evans, and Brian Ingerson. 2009. Yaml ain't markup language (yaml™) version 1.1. *Working Draft 2008* 5, 11 (2009).

[4] Agnė Brilingaitė, Linas Bukauskas, and Aušrius Juozapavičius. 2020. A framework for competence development and assessment in hybrid cybersecurity exercises. *Computers & Security* 88 (2020), 101607.

[5] Pavel Čeleda, Jakub Čegan, Jan Vykopal, Daniel Tovarňák, et al. 2015. Kypo–a platform for cyber defence exercises. *M&S Support to Operational Tasks Including War Gaming, Logistics, Cyber Defence. NATO Science and Technology Organization* (2015).

[6] Nikolaos Christoforatos, Ifigenia Lella, Evangelos Rekleitis, Christian Van Heurck, and Alexandros Zacharis. 2022. *Cyber Europe 2022: After Action Report.*

[7] Rosa Delima, Retantyo Wardoyo, and Khabib Mustofa. 2021. Goal-Oriented Requirements Engineering: State of the Art and Research Trend. *JUITA: Jurnal Informatika* 9, 1 (2021), 105–114.

[8] Adam Doupé, Manuel Egele, Benjamin Caillat, Gianluca Stringhini, Gorkem Yakin, Ali Zand, Ludovico Cavedon, and Giovanni Vigna. 2011. Hit 'em where it hurts: a live security exercise on cyber situational awareness. In *Proceedings of the 27th Annual Computer Security Applications Conference* (New York, NY, USA) *(ACSAC '11)*. ACM, 51–61. https://doi.org/10.1145/2076732.2076740

[9] Marlon Dumas, Marcello La Rosa, Jan Mendling, Hajo A Reijers, et al. 2018. *Fundamentals of business process management.* Vol. 2. Springer.

[10] Adrian Furtună, Victor-Valeriu Patriciu, and Ion Bica. [n. d.]. A structured approach for implementing cyber security exercises. In *2010 8th International Conference on Communications* (2010-06). 415–418. https://doi.org/10.1109/ICCOMM.2010.5509123

[11] Mieke Julie Jans, Michael Alles, and Miklos A Vasarhelyi. 2010. Process mining of event logs in auditing: Opportunities and challenges. *Available at SSRN 1578912* (2010).

[12] Mika Karjalainen, Tero Kokkonen, and Samir Puuska. 2019. Pedagogical aspects of cyber security exercises. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 103–108.

[13] Mika Karjalainen, Samir Puuska, and Tero Kokkonen. 2020. Measuring learning in a cyber security exercise. In *Proceedings of the 12th International Conference on Education Technology and Computers.* 205–209.

[14] Gerhard Keller, August-Wilhelm Scheer, and Markus Nüttgens. 1992. *Semantische Prozeßmodellierung auf der Grundlage" Ereignisgesteuerter Prozeßketten (EPK)".* Inst. für Wirtschaftsinformatik.

[15] Stela Kucek and Maria Leitner. 2020. An empirical survey of functions and configurations of open-source capture the flag (ctf) environments. *Journal of Network and Computer Applications* 151 (2020), 102470.

[16] Phillip A Laplante and Mohamad Kassab. 2022. *Requirements engineering for software and systems.* Auerbach Publications.

[17] Maria Leitner. 2023. A Scenario-Driven Cyber Security Awareness Exercise Utilizing Dynamic Polling: Methodology and Lessons Learned. In *Proceedings of the 9th International Conference on Information Systems Security and Privacy - ICISSP.* INSTICC, SciTePress, 634–642. https://doi.org/10.5220/0011780400003405

[18] Maria Leitner, Maximilian Frank, Wolfgang Hotwagner, Gregor Langner, Oliver Maurhart, Timea Pahi, Lenhard Reuter, Florian Skopik, Paul Smith, and Manuel Warum. 2020. AIT cyber range: flexible cyber security environment for exercises, training and research. In *Proceedings of the European Interdisciplinary Cybersecurity Conference.* 1–6.

[19] Kaie Maennel. 2020. Learning analytics perspective: Evidencing learning from digital datasets in cybersecurity exercises. In *2020 IEEE European symposium on security and privacy workshops (EuroS&PW).* IEEE, 27–36.

[20] Juergen Manger and Stefanie Rinderle-Ma. 2022. Cloud Process Execution Engine: Architecture and Interfaces. *arXiv preprint arXiv:2208.12214* (2022).

[21] Juergen Mangler, Gerhard Stuermer, and Erich Schikuta. 2010. Cloud process execution engine-evaluation of the core concepts. *arXiv preprint arXiv:1003.3330* (2010).

[22] Sten Mäses, Kaie Maennel, Mascia Toussaint, and Veronica Rosa. 2021. Success factors for designing a cybersecurity exercise on the example of incident response. In *2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW).* IEEE, 259–268.

[23] Jan Mendling, Hajo A Reijers, and Wil MP van der Aalst. 2010. Seven process modeling guidelines (7PMG). *Information and software technology* 52, 2 (2010), 127–136.

[24] ENISA European Network and Information Security Agency. 2009. *Good Practice Guide on National Exercises.* https://www.enisa.europa.eu/publications/national-exercise-good-practice-guide

[25] OMG. 2011. *Business Process Model and Notation (BPMN), Version 2.0.* Object Management Group. http://www.omg.org/spec/BPMN/2.0

[26] Rodney Petersen, Danielle Santos, Matthew Smith, and Gregory Witte. 2020. *Workforce framework for cybersecurity (NICE framework).* Technical Report. National Institute of Standards and Technology.

[27] KM Rajasekharaiah, Chhaya S Dule, and E Sudarshan. 2020. Cyber security challenges and its emerging trends on latest technologies. In *IOP Conference Series: Materials Science and Engineering*, Vol. 981. IOP Publishing, 022062.

[28] Anne Rozinat and Wil MP Van der Aalst. 2008. Conformance checking of processes based on monitoring real behavior. *Information Systems* 33, 1 (2008), 64–95.

[29] Ensar Seker and Hasan Huseyin Ozbenli. 2018. The concept of cyber defence exercises (cdx): Planning, execution, evaluation. In *2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security).* IEEE, 1–9.

[30] Sanggyu Shin and Yoichi Seto. 2020. Development of iot security exercise contents for cyber security exercise system. In *2020 13th International Conference on Human System Interaction (HSI).* IEEE, 1–6.

[31] Florian Skopik and Maria Leitner. 2021. Preparing for National Cyber Crises Using Non-linear Cyber Exercises. In *2021 18th International Conference on Privacy, Security and Trust (PST).* IEEE, 1–5.

[32] Wil Van Der Aalst. 2012. Process mining. *Commun. ACM* 55, 8 (2012), 76–83.

[33] Wil MP van der Aalst. 2010. Process discovery: Capturing the invisible. *IEEE Computational Intelligence Magazine* 5, 1 (2010), 28–41.

[34] Wil MP van Der Aalst, Arthur HM Ter Hofstede, Bartek Kiepuszewski, and Alistair P Barros. 2003. Workflow patterns. *Distributed and parallel databases* 14 (2003), 5–51.

[35] Jan Vykopal, Martin Vizvary, Radek Oslejsek, Pavel Celeda, and Daniel Tovarnak. 2017. Lessons learned from complex hands-on defence exercises in a cyber range. In *2017 IEEE Frontiers in Education Conference (FIE)* (2017-10). 1–8. https://doi.org/10.1109/FIE.2017.8190713

[36] Richard Weiss, Michael E Locasto, and Jens Mache. 2016. A reflective approach to assessing student performance in cybersecurity exercises. In *Proceedings of the 47th ACM technical symposium on computing science education.* 597–602.

[37] Shao-Fang Wen, Muhammad Mudassar Yamin, and Basel Katt. 2021. Ontology-Based Scenario Modeling for Cyber Security Exercise. In *2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW).* IEEE, 249–258.

[38] Mathias Weske et al. 2007. Concepts, languages, architectures. *Business Process Management* (2007).

[39] Charles V Wright, Jens Mache, and Richard Weiss. 2016. Hands-on exercises about DNS attacks: details, setup and lessons learned. *Journal of Computing Sciences in Colleges* 32, 1 (2016), 117–125.

[40] Tarun Yadav and Arvind Mallari Rao. 2015. Technical aspects of cyber kill chain. In *Security in Computing and Communications: Third International Symposium, SSCC 2015, Kochi, India, August 10-13, 2015. Proceedings 3.* Springer, 438–452.

[41] Muhammad Mudassar Yamin and Basel Katt. 2018. Inefficiencies in Cyber-Security Exercises Life-Cycle: A Position Paper. (2018), 3.

[42] Muhammad Mudassar Yamin and Basel Katt. 2022. Modeling and executing cyber security exercise scenarios in cyber ranges. *Computers & Security* 116 (2022), 102635.