Applying Process Discovery to Cybersecurity Training: An Experience Report

Martin Macak
Faculty of Informatics
Masaryk University
Brno, Czech Republic
macak@mail.muni.cz

Radek Oslejsek
Faculty of Informatics
Masaryk University
Brno, Czech Republic
oslejsek@mail.muni.cz

Barbora Buhnova
Faculty of Informatics
Masaryk University
Brno, Czech Republic
buhnova@mail.muni.cz

***Abstract*—Quality improvement of practical cybersecurity training is challenging due to the process-oriented nature of this learning domain. Event logs provide only a sparse preview of trainees' behavior in a form that is difficult to analyze. Process mining has great potential in converting events into behavioral graphs that could provide better cognitive features for understanding users' behavior than the raw data. However, practical usability for learning analytics is affected by many aspects. This paper aims to provide an experience report summarizing key features and obstacles in integrating process discovery into cyber ranges. We describe our lessons learned from applying process mining techniques to data captured in a cyber range, which we have been developing and operating for almost ten years. We discuss lessons learned from the whole workflow that covers data preprocessing, data mapping, and the utilization of process models for the post-training analysis of Capture the Flag games. Tactics addressing scalability are explicitly discussed because scalability has proven to be a challenging task. Interactive data mapping and Capture the Flag specific features are used to address this issue.**

***Index Terms*—cybersecurity, hands-on training, process mining, data analysis, learning analytics**

1. Introduction

As cybersecurity skills require higher-order thinking, the best way to develop and ameliorate these abilities is through hands-on training [14], [20]. However, practical cybersecurity is process-oriented. The learning tasks are focused on attack or defense skills like scanning the computer network for vulnerable servers or protecting the server by a firewall. Although modern cyber ranges, in which practical cybersecurity training is organized [9], [30], [40], gather behavioral data in the form of event logs, they provide only a limited view of trainees' behavior, and they are very difficult to explore.

Process mining. Although not yet fully explored in the context of cyber-security training analysis, process mining (PM) has great potential to generate comprehensible behavioral models. It has been successful in aiding with similar types of tasks in domains like education [6], cybersecurity [18], healthcare [39], software reliability [17], and several others [11].

PM [31] is a discipline that aims to understand and analyze processes, thus providing more perspectives to data interpretation. It provides a better understanding of the operations by extracting knowledge from event logs, which makes it a great candidate to address the challenges in identifying problematic situations and the reasons behind them in the hands-on training sessions. In particular, the process mining is booming in three analytical categories.

Process discovery aims to find a formal descriptive model for the underlying event logs. The model usually has the form of a dependency graph capturing activities and traces inferred from the raw log data, as demonstrated in Figure 1 on the hacking process. However, the exact appearance and then usability of the graph depend on many factors that are the subject of process discovery.

Conformance checking is used to monitor and inspect how much the observed behavior conforms to the expected behavior. Techniques of comparative analysis are used to compare models of expected behavior with actual executions of processes.

Process enhancement aims to improve and extend an existing process model based on event logs capturing actual executions of the process.

In this paper, we focus on the potential of *process discovery* for cybersecurity training session analysis. Our goal is not to study the impact of process models on students' skills. Instead, this experience report summarizes the necessary steps, limits, and obstacles of the reconstruction of behavioral graphs from sparse cybersecurity training data and suggests their possible usability for post-training analysis on several examples. We describe our experience using the data collected from cybersecurity Capture the Flag games [35] as an input for process discovery algorithms. We demonstrate the usefulness of process mining models to answer analytical questions that might help with the further improvement of training programs. We also discuss the possibilities of tackling graph complexity and achieving a scalable solution. Our approach is based on specific features of the training data and the combination of a traditional graph representation of process models with complementary visualizations.

2. Related Work

Nowadays, there is plenty of various cyber ranges and testbeds that differ in their features and primary purpose. This paper deals with modern cyber ranges that emulate computer networks and then support the organization of

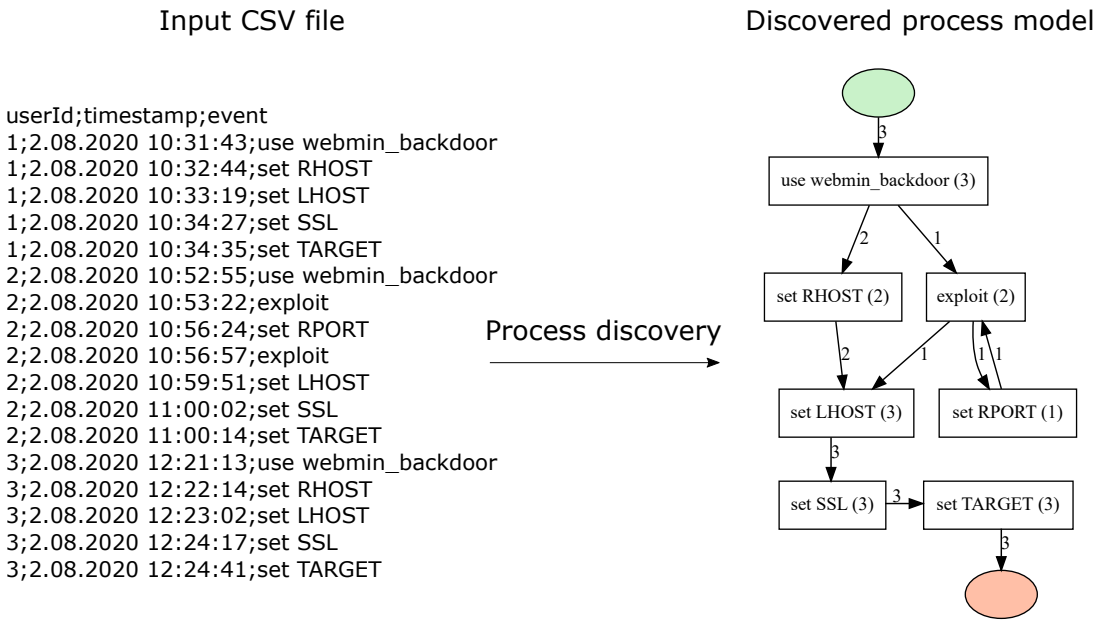

Figure 1. Process discovery example of the hacking process using Metasploit command line history

hands-on training programs with advanced data collection, e.g., *SimSpace Cyber Range* [26], *EDURange* [37], DETERlab [21], *CyRIS* [24], or *CyTrONE* [5].

The results of this experience report are based on data collected from the *KYPO Cyber Range*[1] [34] that we have been operating since 2013. The system serves as a platform for regular practical training of our university students. We believe that process discovery techniques can improve existing analytical tools [12], [33] of the cyber range.

Process mining techniques have already been applied in numerous educational situations [6]. The term *educational process mining* (EPM) is often used in this context. Mukala et al. [22] use EPM to obtain and analyze students' learning habits from the Coursera learning system, while Bannert et al. [3] applied these techniques to detect students' learning difficulties. Macak et al. [19] used process mining to understand student coding-behavior patterns from the Git log of their projects. Multiple other articles use EPM to gain a better understanding of the educational process from Moodle logs [7], [25], for curriculum mining [29], analysis of student registrations [2], or professional training [8].

Our research builds on the same EPM ideas and principles. Still, it addresses a specific application domain—hands-on cybersecurity training—aiming to research how the data properties affect the process mining.

Only a few papers address cybersecurity learning. Weiss et al. [36], [37] demonstrated that carefully constructed directed graphs constructed from command history can provide useful information about trainees' progress and behavior. The evaluation of two different graph models is also presented in [41].

Another solution can be found in Andreolini et al. [1]. Their method utilizes a reference graph of ideal behavior to automatically construct trainee graphs from monitored trainee activities and system logs, e.g., command history, web browsing history, GUI interactions, network events, or OS process activities.

These approaches to the graph-based evaluation of cybersecurity exercises share some important concepts. They use special algorithms for graph construction that were precisely designed for the specific data, especially command-line histories. Our goal is to use well-established process discovery algorithms to reconstruct trainees' walkthroughs. Moreover, the previous methods address *conformance checking* where reference graphs of expected behavior have to be defined in advance. However, creating reference graphs is laborious and tricky. We aim to explore the possibilities and limits of a more generic *process discovery* analysis where no reference graphs are required.

We are aware that process mining algorithms can produce huge graphs. The previous papers deal with the complexity by restricting data or models, e.g., by using command-line histories only or analyzing the behavior of a single trainee. This paper describes our experience with combining multiple data sources and multiple participants. It is assumed that the information complexity will be solved by combining domain-specific data filtering with exploratory visual analysis techniques that complement standard graph-based views.

3. Methodology and Data Description

This section provides a data and methodology background related to the conclusions presented in the remainder of the paper. The structure reflects three significant steps of the process mining workflow.

Data collection and preprocessing. We analyzed event logs of fifteen training sessions of six different training scenarios to identify possible errors, inconsistencies, and low-level data properties that could affect process discovery. The number of participants whose data were collected in each scenario varied between 8 and 58, with an average

1. KYPO is a Czech acronym for Cybersecurity Polygon.

of 25. The average number of participants in each training session was 9.8. All the training sessions were organized in KYPO Cyber Range.

Section 4 summarizes our lessons learned from this phase. We abstract away from specific monitoring infrastructures and focus on data only. Although the experience is based on the data from only a single cyber range, the presented results are generic. They can appear in any modern cyber range that generates event logs with timestamps.

Data mapping. Datasets used in our experimental evaluation consist of three different data types produced by two different data sources of the distributed architecture of KYPO Cyber Range.

The first two data types are *commands* executed on the UNIX command line (shell) and commands typed within the *Metasploit* tool. They both are captured in *sandboxes* that implement isolated virtualized computer networks where trainees perform cybersecurity tasks. Besides the commands and their arguments, additional pieces of information are captured in event records, e.g., trainee's ID, working directory under which the command was executed, hostname and IP address of the host, trainee's ID, or timestamp.

The third data type is *game events*. While shell and Metasploit commands represent generic data types that are produced by many cyber ranges, game events introduce higher-level semantics into datasets because they capture the state and assessment of the training session. They can vary between cyber ranges and training types.

Our training data follows so-called *Capture The Flag* (CTF) games [10], [28], [32], [38] consisting of well-described tasks divided into consecutive levels. Completing each task yields a text string called the *flag* that must be inserted into the system to proceed to the subsequent task. Trainees can take hints or show the complete solution (step-by-step instructions leading to the correct flag). Points are awarded or deducted for these actions so that the final scores of individual trainees are mutually comparable and can be used for their basic evaluation.

TABLE 1. GAME EVENTS AND THEIR MEANING.

Event	Description. The trainee ...
TrainingRunStarted	... started the training.
TrainingRunEnded	... finished the training.
LevelStarted	... started a level.
LevelCompleted	... submitted a correct flag.
WrongFlagSubmitted	... submitted a wrong flag.
CorrectFlagSubmitted	... submitted the right flag.
HintTaken	... took a hint.
AssessmentAnswers	... answered a questionnaire.
SolutionDisplayed	... viewed the level solution.

Types of game events produced by KYPO Cyber Range are summarized in Table 1. They are common for all games regardless of the specific cybersecurity content. Each event is extended with many other pieces of information, such as the trainee's identifier, timestamp, and the game level in which the event appeared. Moreover, individual event types can have specific mandatory or optional data. For example, submitting the wrong or correct flag always includes the flag value. A hint is always equipped with a hint number, brief description, and penalty.

Process discovery algorithms require special input data formatting. In particular, each event in the log needs to refer to a single process instance, named *case*. Also, each event needs to refer to a step in the process named *activity*. The last minimal requirement is that events within the case must be ordered, either sequentially or by including *timestamp*. Otherwise, it is not possible to apply process discovery to the data. Section 5 discusses how to map raw events of training logs into the input of process discovery algorithms.

Utilization of Process Models for Training Analysis. Having the data in the format suitable for process discovery algorithms does not guarantee that generated process models are reasonable. Detailed research into the practical usability of various models for learning analytics is out of the scope of this paper. Therefore, we demonstrate the usefulness on several examples and discuss our experience gained from experiments.

The experiments were performed in the *Disco* [13] process mining system and the *PM4py* process mining library. Nevertheless, process mining tools differ only in the user interface and usage purposes (e.g., a complete GUI-based tool versus an API-based library), while the principles of underlying process discovery algorithms are the same. Therefore, it does not matter which tool is used, and we did not notice any important differences in this sense.

Although we conducted experiments with multiple datasets, the process models presented in this paper were generated from a single dataset to provide a consistent preview of our experience and demonstrate the key principles. The dataset was captured in 2020 during the Online Hacking Day, where computer science undergraduate or graduate students played CTF games invented by their colleagues within a specialized cybersecurity course. During the event, multiple games were played by the participants. We use datasets of 3 trainees who played the same game, which is a good number to demonstrate the goals and obstacles in using process mining techniques for CTF game analysis without being overburdened with huge amounts of data.

Section 6 summarizes our observations from using process mining models for the analysis of hands-on training data. Concrete examples demonstrate possible usability. Although only simple process graphs are presented to demonstrate the principles, tactics for tackling the scalability of process graphs are explicitly discussed.

4. Lessons Learned from Data Preprocessing

Process discovery methods are sensitive to data errors that can produce misleading process graphs. Moreover, the incorrectness may not be visible at first sight. Therefore, it is necessary to pay attention to data preprocessing and validation. We classify our observations into two categories: generic data errors removable by data cleansing and time-related inconsistencies.

4.1. Data Cleansing

The analysis of 15 datasets revealed that almost all of them include rubbish pieces of information that can mislead analysts. They can be classified as follows.

Dummy data. Organizers of exercises very often conduct dry runs or testing loops before the real exercise to check the training scenario and cyber range infrastructure. Our experience shows that such testing is a standard part of the training life cycle, even for regular training courses. Unfortunately, this unwarily created data often remains in the datasets and must be removed before the learning analytics is used. Technically, the removal is easy due to the presence of timestamps. But the problem occurs when the dummy data is overlooked in the dataset. They can significantly affect obtained process graphs, while it is difficult to notice that something is wrong.

Missing data. Modern cyber ranges are often designed as complex distributed systems where many things can go wrong not only due to bugs in code but as a consequence of failures in the runtime environment. Failures in network connectivity or unreliable low-level virtualization services, for example, can cause some data to be missing and then provide confusing insight into the behavior of related trainees.

Unrelated data. The raw datasets include many technical records, e.g., IDs of virtual sandboxes where the training is performed, IDs of the training sessions, or score development values that are irrelevant to the behavior analysis of a single training session. Removing these unrelated pieces of information helped us to obscure details and infer conceptional views on the data.

Duplicate data. Another data reduction resided in the aggregation of log records. For example, every completion of one level is immediately followed by the next level (except the very last level of the training). In the raw datasets, the corresponding events *LevelCompleted* and *LevelStarted* are recorded just behind each other. The time difference is always negligible, and these events are never interrupted by another event. Therefore, we removed the *LevelCompleted* events from the datasets as they bring no new semantic information and make the models pointlessly complex. We also removed the *TrainingRunStarted* for a similar reason.

4.2. Time Unification

Events need to be sorted sequentially in the dataset, or timestamps must be included in data records. Then it is possible to reconstruct traces of users' activities. As the training data is captured from multiple resources, the timestamp-based ordering usually represents the only possible way. Therefore, time records require special attention.

Synchronization. Process models are sensitive to events' ordering. Only a tiny shift in times can re-order events in joint process graphs of multiple trainees and produce confusing models. It is necessary to precisely synchronize time, especially in distributed cyber ranges.

Relativity. Analytical goals usually include tasks that compare the behavior of a group of trainees, e.g., students of a single course. In this case, the absolute time has to be converted to the relative time of the training session, i.e., the time elapsed since the *TrainingRunStarted* event). Participants of exercises organized as supervised fixed-time group sessions usually start almost at the same time. However, time-shifting is specifically required if the training content is available online at any time.

Gaps. Participants of training courses with a tight schedule are forced to work intensively without significant time for the rest. On the other hand, loosely conceived training programs would enable participants to stop playing for a while and continue with tasks later, even the next day. These gaps affect process models significantly when timestamps are used for process discovery. Therefore, datasets from the loosely organized training events require much more attention and expertise to be paid by the analyst, who has to take care of the correct time unification, its employment in process discovery, and interpretation of obtained models.

Format. Another problem we faced was the diversity of time formats. Our datasets included three different time formats, depending on the event source in the distributed environment: the POSIX time (number of seconds that elapsed since "the epoch" – 00:00:00 UTC on 1 January 1970) and two variants of human-readable formats: "2020-05-14T08:22:11.563Z" and "2020-05-14T12:22:11.563715+02:00". They had to be unified.

5. Lessons Learned from Data Mapping

As data from hands-on cybersecurity exercises differ in their semantics and content, the selection of proper mapping into process mining *cases* and *activities* is an important step in the analytical workflow.

Case. In our datasets, every event is equipped with an anonymized unique identifier of the trainee who produced the event. We took this TRAINEE ID as the *case ID* of process mining so that the process discovery reconstructs the traces of individual trainees. These traces are the primary subjects of the analysis, enabling the analyst to compare trainees' behavior and analyze the expected vs. anomalous behavior with respect to the training definition. Traces appear as paths in process graphs.

Activity. The process mining *activities* appear as nodes in process graphs. Their mapping depends on the type of event record. For game events, the event type column from Table 1 can be used as activity. For shell or Metasploit commands, we can use the basic command, e.g., SSH. However, it is important to point out that incorporating other data from event records can significantly affect obtained process models. For example, consider the event "the trainee took the hint No. 4.1" encoded in the raw data. If the record is mapped into the process discovery so that EVENT = "*HintTaken*" and the hint's number is omitted, then the process model will interpret "taking a hint" as single activity regardless of what hint has been taken or when (in which level). Only the "HintTaken" node appears in the process graph. On the contrary, merging both values into the event mapping, i.e., EVENT = "*HintTaken* hint-number" will produce a much finer model with nodes like "HintTaken 4.1", "HintTaken 4.2", etc.

We chose reasonable mapping depending on the event and data available in our experiments. For example, the *HintTaken* events are always mapped to *activity* together with hint ID to distinguish different hints in the model. *WrongFlagSubmitted* events are always equipped with the flag value for a similar reason – to distinguish between different unsuccessful attempts.

6. Lessons Learned from Process Discovery

The usability of process models generated by process discovery depends on the algorithms used and analytical goals. We focus on the analysis of trainees' behavior and the detection of flows in training scenarios to demonstrate possible usage in these important analytical tasks. Moreover, the scalability of process graphs is explicitly discussed because it may pose an important challenge for practical usability.

6.1. Behavior of Trainees

Process mining can produce different types of process models based on different mining algorithms, e.g., alpha-miner, inductive miner, or heuristic-miner [31]. Moreover, these algorithms can use various internal metrics that affect the produced models.

Process models. It is out of the scope of this paper to analyze the expressiveness of different process models for learning analytics tasks. We conducted only basic experiments with various algorithms and then chose heuristic nets for further discussion. They have been shown to provide sufficient insight into trainees' behavior where an analyst can glimpse walkthroughs, identify anomalies, and ask questions related to learning analytics.

A basic heuristic net for level 40 depicted in Figure 2 demonstrates these possibilities. Nodes in the graph represent *activities*, numbers at arrows and in the brackets inside nodes denote for the number of passes through the path (one trainee can pass the node multiple times).

We can clearly see that two trainees used the first hint right after the beginning of the level. One trainee tried to solve the task by using several commands (`ip`, `ipconfig`, `ls`, and `connect`) but failed and then finally took the hint as well. This can indicate either the confusion in the task description, improper difficulty with respect to learners' knowledge, or the lack of learners' motivation, which may stem from a bad experience in previous levels.

After taking the first hint, two trainees successfully finished the level by using the `nmap` command. One participant had trouble due to misspelling the command (the capital "N") and then finally took the second hint that helped him finish the task and submit a correct flag. Also, the useless usage of the `sudo` command is visible in the model.

Complementary model metrics. The process model depicted in Figure 2 uses so-called absolute frequency. It depicts the total number of times a particular activity has been performed or a specific transition has been made. The switching of the focus of the process model to alternate process metrics can bring a complementary view that can

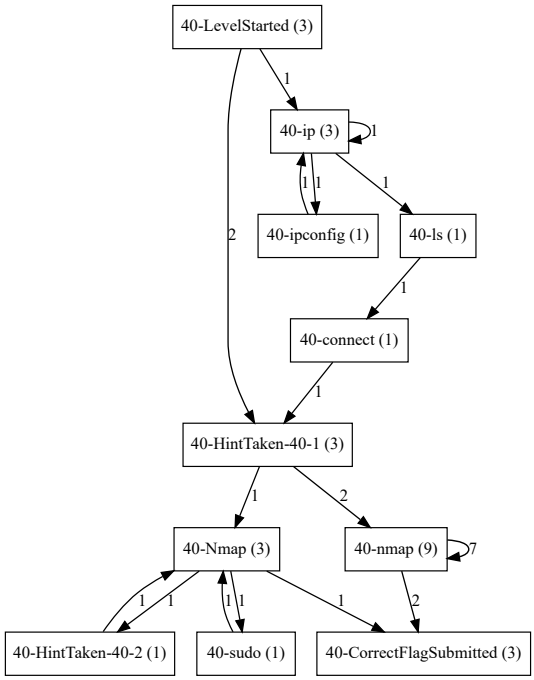

Figure 2. Detailed process model for game level 40. Numbers at arrows and in the brackets inside nodes denote for the number of passes through the path.

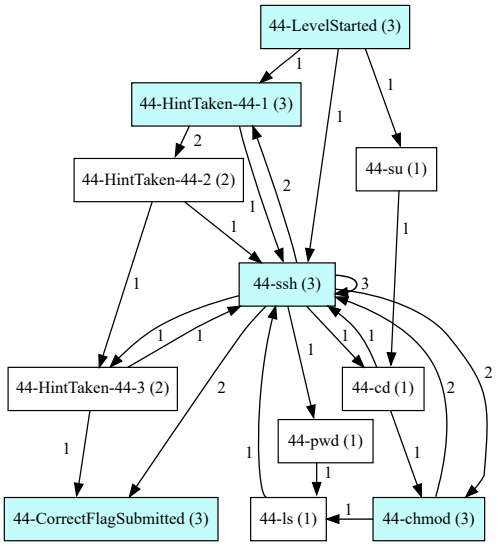

Figure 3. Process model using frequency metric for level 44. Nodes used by all three trainees are emphasized.

reveal or emphasize different behavioral aspects of the data.

Using case frequency, for instance, the numbers on transitions and inside nodes of the model represent the number of unique cases that pass through the transition or event. Based on this metric, the analyst can infer the importance of steps that trainees performed.

In Figure 3, it is evident that for the successful pass of level 44, all three trainees needed to take the first hint, use `ssh` command multiple times, and use `chmod` command. On the contrary, using commands like `pwd` or `cd` was not necessary for solving the task.

A closer look at the model in Figure 3 reveals yet

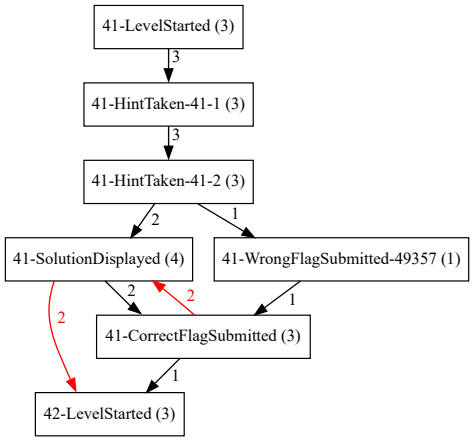

Figure 4. Level of detail based on event type. Game events of only level 41 are displayed. The red arrows between highlight a flow in the gameplay caused by a bug in KYPO Cyber Range implementation.

another behavior that is not visible at first sight. There is one direct transition between taking the third hint (44-3) and providing the correct flag. However, it is highly unlikely that a trainee would correctly complete the level without using the command line in the meantime. Therefore, we can hypothesize that one trainee took the hint after retrieving the correct flag just to be sure that he got it right. An in-depth data analysis performed out of this model confirmed this hypothesis.

Similarly to case frequency, it is also possible to check other process model metrics like the duration of transitions. With this metric, the analyst can see the process model from the time perspective, identifying the parts of CTF games that were too easy or where the trainees struggled.

For the sake of simplicity, we use models of absolute frequency in the remainder of this paper.

6.2. Detection of Flaws in the Exercise

The identification of possible flaws in the design of training content is demonstrated in Figure 4, which captures only game events of the game level 41. The level can be considered tough for the trainees' group because all of them took two hints, then two gave up (they looped at the solution with the correct flag). One trainee found and submitted the correct flag on the second try.

Moreover, we have discovered a flaw in the implementation of the KYPO Cyber Range platform. The red arrows in Figure 4 indicate that if trainees submit the correct flag after displaying a task solution (where the flag is shown), then they are redirected back to the solution page instead of being moved to the next level immediately. As this behavior was visible in all levels that contain the *SolutionDisplayed* activity, it was reported as a bug. In the same way, flaws in game design can be visually identified from real gameplay data.

6.3. Scalability

The process graphs presented so far captured only a few trainees and focused on only a part of the collected data to demonstrate their usability for analytical tasks.

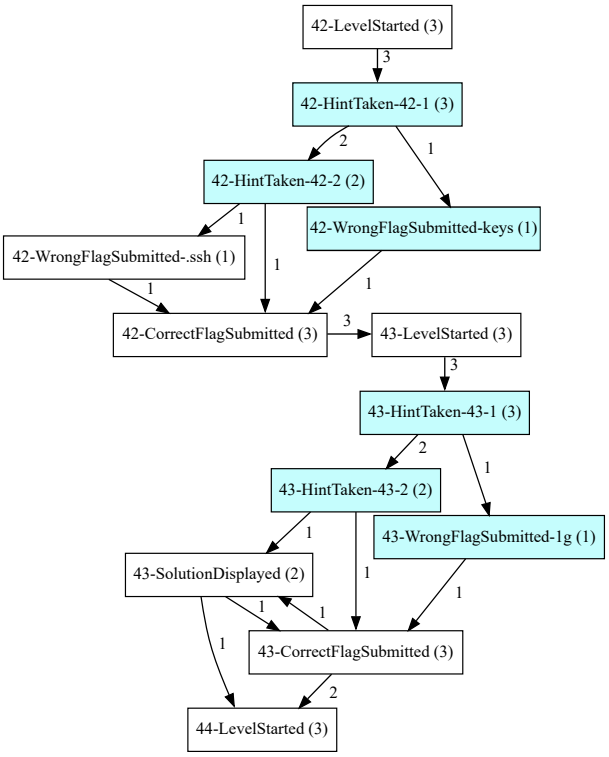

Figure 5. Cross-level process model for levels 42 and 43.

However, the experiments turned out that obtained process graphs can become too complex and incomprehensible for effective analysis. As the walkthroughs of trainees differ, increasing their number leads to the rapid growth of different events and their relationships, making the behavioral graphs too complex. Similarly, incorporating multiple levels into a single model increases the model as well.

As the complexity of process graphs can pose a critical aspect for practical usability, we discuss possible strategies for tackling this problem in this section.

Per-level fragmentation. Levels of CTF games follow each other. There is no way a trainee can skip some level or return to the previous one. Therefore, the heuristic graph can be logically split into loosely coupled coherent parts that correspond to individual levels. It is achieved by adding the level number to all events. Since that, the `42-ssh` and `47-ssh` events, for instance, are considered two different `ssh` commands used in two different levels, and, vice versa, every sub-graph corresponding to a level includes only events that appeared in that level. The fragmentation can be seen in Figure 5, where two coherent sub-graphs for levels 42 and 43 are clearly visible.

Based on this observation, we can fragment the complex process model and either navigate the analyst into individual levels or provide a simplified, high-level overview of the training session.

Events selection. Different events can be considered the different levels of details of the gameplay. For example, game events in our datasets capture key milestones in the exercise, while bash and Metasploit commands can be considered lower-level steps used to achieve these

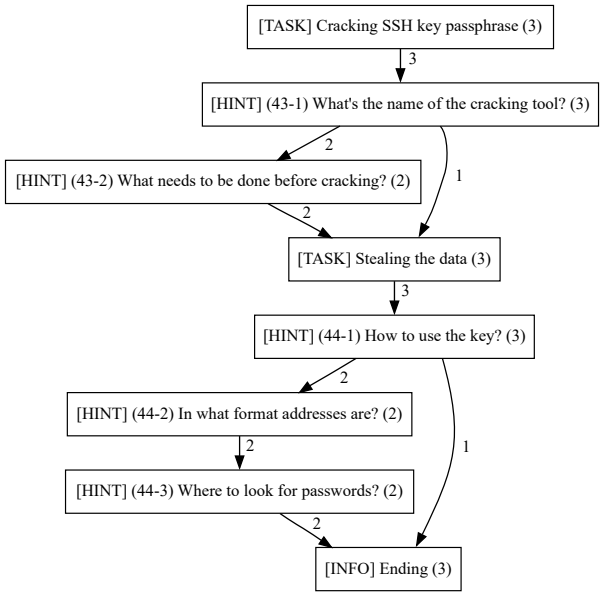

Figure 6. Training structure – only levels 43, 44, and 45 are depicted.

milestones. Selecting events of only the preferred ”level of details” enables analysts to narrow their attention from an overall preview to detailed parts.

The process model in Figure 6 demonstrates the principle. It provides a comprehensible high-level overview of the last three levels of the training session by including game events only and completely omitting commands. Events like *43-LevelStarted* are equipped with relevant text from raw records to make the model even more readable.

Compare this view with the model in Figure 2 that captures all events of only a single level. Although Figure 6 depicts three levels, it is easy to spot that all trainees successfully finished the game, but nobody went through without the help of hints. It is also evident that at both levels, there is one trainee who took only one hint. However, we do not know whether it is the same participant. Therefore, this example also demonstrates the limits of static process models. Verification of the hypothesis that this is the same trainee requires in-depth data exploration.

This filtering approach is generic. It does not depend on a level-based structure and is also usable for other types of cybersecurity exercises. On the other hand, some semantics classification of events reflecting their level of detail has to be done before this technique can be used.

Events mapping. Event mapping discussed in Section 5 can also affect the complexity of process graphs. For example, if only `ssh` is taken into account regardless of other parameters, then only one node is produced. The number of trainees (number of different logs of the ssh usage) does not affect the complexity of the model. On the contrary, considering `ssh 127.0.01` and `ssh 127.0.02` two distinct commands will produce two distinct nodes. In this case, the more trainees included in the model, the more probably they will introduce a new node with a new remote address.

In general, scalability depends on the freedom or uncertainty of values provided by users. For example, the upper bound of nodes representing the ”taking concrete hint” event is limited because the number of hints defined in the exercise is limited. On the contrary, the number of nodes representing the ”incorrect flag with the typed value” is unlimited since users can provide arbitrary values due to improperly solved tasks. The ”ssh connection with remote address” demonstrates the case between these extremes. On one side, users could provide an arbitrary address of the remote host. On the other side, they are limited by a few hosts presented in their network (sandbox). Other addresses are usually typos and then rare.

Therefore, properly used event mapping can manage the complexity of generated models and then become part of the exploratory workflow. However, precise rules and tactics remain open for future work.

Exploratory Techniques of Visual Analytics. Another approach to deal with model complexity is by transforming graph-based views of process models into alternate concepts by applying techniques of visual analytics that support human decisions in the analytical loops.

The main research goal of visual analytics is revealing relationships hidden in the raw data [15], proving analytical hypotheses [27], and tackling graph complexity by using principles of exploratory data visualizations [4].

With regards to this theory, process models can be considered so-called *multivariate graphs* that appear in many application domains. Dealing with them by complementary visualizations is a broadly researched field within the visual analytics discipline [16]. Although graph properties of cybersecurity training programs may differ, the visualization principles of multivariate graphs are generic and then applicable.

According to the classification formulated in Nobre et al. [23], process graphs of our CTF games are of a medium (less than 1.000 nodes) or large size (more than 1.000 nodes), k-partite; they have heterogeneous nodes with few attributes and homogeneous edges. Based on these observations, the best alternate visualization techniques should combine *attribute-driven faceting* with *quilts* and *integrated* or *juxtaposed* view operations. However, the design of concrete analytical visualizations remains a challenging task requiring further research.

7. Conclusion and Future Work

In this paper, we contributed to the state of the art of applying process mining in practical cybersecurity training sessions.

We described our experience of using data captured during hands-on exercises organized in KYPO Cyber Range as the input of external generic process discovery tools. We confirmed the usefulness of the collected data for this purpose. However, many preprocessing steps have to be done to unify and synchronize the data gathered from multiple sources in the distributed training environment. Therefore, tight integration of data preprocessing steps and active avoidance of inconsistencies inside the cyber range infrastructure has to be done for routine usage.

We demonstrated successful mapping of raw event logs into the required data format of process discovery methods. Our datasets reflect obvious data monitored by modern cyber ranges. However, we are aware exact mapping of available raw data into process mining events sig-

nificantly affects the process models and their complexity. Therefore, they have to be carefully chosen based on the analytical goals. A unified data abstraction and mapping rules would help to formalize and generalize the analytical workflow in the future.

In our follow-up research, we also would like to compare the usefulness of different process models for different analytical goals like run-time situational awareness, detection of patterns in trainees' behavior, or the identification of flows in training design and assessment.

The scalability appears to be the most challenging task of the process mining workflow. However, we are convinced that the combination of domain-specific data filtering with the exploratory visualization of multivariate graphs enables us to bring a suitable solution to this open problem. We plan to integrate these methods into the existing visual-analysis dashboards of KYPO Cyber Range.

Acknowledgment

The work was supported by ERDF "Cyber-Security, CyberCrime and Critical Information Infrastructures Center of Excellence" (No. CZ.02.1.01/0.0/0.0/16_019/0000822).

References

[1] M. Andreolini, V. G. Colacino, M. Colajanni, and M. Marchetti. A framework for the evaluation of trainee performance in cyber range exercises. *Mobile Networks and Applications*, 25(1):236–247, 2020.

[2] S. Anuwatvisit, A. Tungkasthan, and W. Premchaiswadi. Bottleneck mining and petri net simulation in education situations. In *2012 Tenth International Conference on ICT and Knowledge Engineering*, pp. 244–251, 2012.

[3] M. Bannert, P. Reimann, and C. Sonnenberg. Process mining techniques for analysing patterns and strategies in students' self-regulated learning. *Metacognition and learning*, 9(2):161–185, 2014.

[4] A. Batch and N. Elmqvist. The interactive visualization gap in initial exploratory data analysis. *IEEE transactions on visualization and computer graphics*, 24(1):278–287, 2017.

[5] R. Beuran, D. Tang, C. Pham, K.-i. Chinen, Y. Tan, and Y. Shinoda. Integrated framework for hands-on cybersecurity training: CyTrONE. *Computers & Security*, 78:43–59, 2018.

[6] A. Bogarín, R. Cerezo, and C. Romero. A survey on educational process mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(1):e1230, 2018.

[7] A. Bogarín, C. Romero, R. Cerezo, and M. Sánchez-Santillán. Clustering for improving educational process mining. In *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*, LAK '14, p. 11–15. ACM, New York, NY, USA, 2014.

[8] A. H. Cairns, B. Gueni, M. Fhima, A. Cairns, S. David, and N. Khelifa. Towards custom-designed professional training contents and curriculums through educational process mining. In *The Fourth International Conference on Advances in Information Mining and Management*, pp. 53–58, 2014.

[9] N. Chouliaras, G. Kittes, I. Kantzavelou, L. Maglaras, G. Pantziou, and M. A. Ferrag. Cyber ranges and testbeds for education, training, and research. *Applied Sciences*, 11(4), 2021.

[10] A. Davis, T. Leek, M. Zhivich, K. Gwinnup, and W. Leonard. The Fun and Future of CTF. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*. USENIX Association, San Diego, CA, 2014.

[11] C. dos Santos Garcia, A. Meincheim, E. R. F. Junior, M. R. Dallagassa, D. M. V. Sato, D. R. Carvalho, et al. Process mining techniques and applications - a systematic mapping study. *Expert Systems with Applications*, 133:260 – 295, 2019.

[12] K. Dočkalová Burská, V. Rusňák, and R. Ošlejšek. Data-driven insight into the puzzle-based cybersecurity training. *Computers & Graphics*, 102, 2022.

[13] C. W. Günther and A. Rozinat. Disco: Discover your processes. *BPM (Demos)*, 940(1):40–44, 2012.

[14] G. Hatzivasilis, S. Ioannidis, M. Smyrlis, G. Spanoudakis, F. Frati, L. Goeke, T. Hildebrandt, G. Tsakirakis, F. Oikonomou, G. Leftheriotis, and et al. Modern aspects of cyber-security training and continuous adaptation of programmes to trainees. *Applied Sciences*, 10:5702, Aug 2020.

[15] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, eds. *Mastering the Information Age: Solving Problems with Visual Analytics*. Goslar : Eurographics Association, 2010.

[16] S. Kriglstein, M. Pohl, S. Rinderle-Ma, and M. Stallinger. Visual analytics in process mining: Classification of process mining techniques. In *EuroVA@ EuroVis*, pp. 43–47, 2016.

[17] M. Macak, L. Daubner, M. F. Sani, and B. Buhnova. Process mining usage in cybersecurity and software reliability analysis: A systematic literature review. *Array*, p. 100120, 2021.

[18] M. Macak, L. Daubner, M. F. Sani, and B. Buhnova. Cybersecurity analysis via process mining: A systematic literature review. In *International Conference on Advanced Data Mining and Applications*, pp. 393–407. Springer, 2022.

[19] M. Macak, D. Kruzelova, S. Chren, and B. Buhnova. Using process mining for git log analysis of projects in a software development course. *Education and Information Technologies*, pp. 1–31, 2021.

[20] M. E. McMurtrey, J. P. Downey, S. M. Zeltmann, and W. H. Friedman. Critical Skill Sets of Entry-level IT Professionals: An Empirical Examination of Perceptions from Field Personnel. *Journal of Inf. Technology Education: Research*, 7:101–120, 2008.

[21] J. Mirkovic, T. V. Benzel, T. Faber, R. Braden, J. T. Wroclawski, and S. Schwab. The Deter Project. 2010.

[22] P. Mukala, J. Buijs, M. Leemans, and W. van der Aalst. Learning analytics on coursera event data: a process mining approach. In *5th International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA 2015*, pp. 18–32. CEUR-WS. org, 2015.

[23] C. Nobre, M. Meyer, M. Streit, and A. Lex. The state of the art in visualizing multivariate networks. In *Computer Graphics Forum*, vol. 38, pp. 807–832. Wiley Online Library, 2019.

[24] C. Pham, D. Tang, K.-i. Chinen, and R. Beuran. Cyris: A cyber range instantiation system for facilitating security training. In *Proceedings of the Seventh Symposium on Information and Communication Technology*, SoICT '16, pp. 251–258. ACM, New York, NY, USA, 2016.

[25] C. Romero, R. Cerezo, A. Bogarín, and M. Sánchez-Santillán. Educational process mining: A tutorial and case study using moodle data sets. *DATA MINING AND LEARNING ANALYTICS*, p. 1, 2016.

[26] L. Rossey. SimSpace cyber range. ACSAC 2015 Panel: Cyber Experimentation of the Future (CEF): Catalyzing a New Generation of Experimental Cybersecurity Research.

[27] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim. Knowledge Generation Model for Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1604–1613, Dec 2014.

[28] V. Švábenský, J. Vykopal, M. Cermak, and M. Laštovička. Enhancing Cybersecurity Skills by Creating Serious Games. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, pp. 194–199. ACM, ACM, New York, NY, USA, 2018.

[29] N. Trcka and M. Pechenizkiy. From local patterns to global models: Towards domain driven educational process mining. In *2009 Ninth International Conference on Intelligent Systems Design and Applications*, pp. 1114–1119, 2009.

[30] E. Ukwandu, M. A. B. Farah, H. Hindy, D. Brosset, D. Kavallieros, R. Atkinson, C. Tachtatzis, M. Bures, I. Andonovic, and X. Bellekens. A review of cyber-ranges and test-beds: Current and future trends. *Sensors*, 20(24), 2020.

[31] W. van der Aalst. *Process Mining: Data Science in Action*. Springer Publishing Company, Incorporated, 2nd ed., 2016.

[32] G. Vigna, K. Borgolte, J. Corbetta, A. Doupé, Y. Fratantonio, L. Invernizzi, D. Kirat, and Y. Shoshitaishvili. Ten Years of iCTF: The Good, The Bad, and The Ugly. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education*. USENIX Association, San Diego, CA, 2014.

[33] J. Vykopal, R. Ošlejšek, K. Burská, and K. Zákopčanová. Timely feedback in unstructured cybersecurity exercises. In *Proceedings of Special Interest Group on Computer Science Education, Baltimore, Maryland, USA, February 21–24, 2018(SIGCSE'18)*, pp. 173–178. ACM, New York, NY, USA, 2018.

[34] J. Vykopal, R. Ošlejšek, P. Čeleda, M. Vizváry, and D. Tovarňák. KYPO Cyber Range: Design and Use Cases. In *Proceedings of the 12th International Conference on Software Technologies - Volume 1: ICSOFT*, pp. 310–321. SciTePress, Madrid, Spain, 2017.

[35] J. Vykopal, V. Švábenský, and E. Chang. Benefits and pitfalls of using capture the flag games in university courses. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*, pp. 752–758. ACM, 2020.

[36] R. Weiss, M. E. Locasto, and J. Mache. A reflective approach to assessing student performance in cybersecurity exercises. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, pp. 597–602, 2016.

[37] R. Weiss, F. Turbak, J. Mache, and M. E. Locasto. Cybersecurity education and assessment in edurange. *IEEE Security & Privacy*, (3):90–95, 2017.

[38] J. Werther, M. Zhivich, T. Leek, and N. Zeldovich. Experiences in Cyber Security Education: The MIT Lincoln Laboratory Capture-the-flag Exercise. In *Proceedings of the 4th Conference on Cyber Security Experimentation and Test*, CSET'11. USENIX Association, Berkeley, CA, USA, 2011.

[39] R. Williams, E. Rojas, N. Peek, and O. A. Johnson. Process mining in primary care: A literature review. *Studies in health technology and informatics*, 247:376–380, 2018.

[40] M. M. Yamin, B. Katt, and V. Gkioulos. Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. *Computers & Security*, 88:101636, 2020.

[41] V. Švábenský, R. Weiss, J. Cook, J. Vykopal, P. Čeleda, J. Mache, R. Chudovský, and A. Chattopadhyay. Evaluating two approaches to assessing student progress in cybersecurity exercises. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE '22) (To appear)*. ACM, New York, NY, USA, 2022.