

Red Team Redemption: A Structured Comparison of Open-Source Tools for Adversary Emulation

Max Landauer, Klaus Mayer, Florian Skopik, Markus Wurzenberger, Manuel Kern
Center for Digital Safety & Security
Austrian Institute of Technology
Vienna, Austria
firstname.lastname@ait.ac.at

Abstract—Red teams simulate adversaries and conduct sophisticated attacks against defenders without informing them about used tactics in advance. These interactive cyber exercises are highly beneficial to assess and improve the security posture of organizations, detect vulnerabilities, and train employees. Unfortunately, they are also time-consuming and expensive, which often limits their scale or prevents them entirely. To address this situation, adversary emulation tools partially automate attacker behavior and enable fast, continuous, and repeatable security testing even when involved personnel lacks red teaming experience. Currently, a wide range of tools designed for specific use-cases and requirements exist. To obtain an overview of these solutions, we conduct a review and structured comparison of nine open-source adversary emulation tools. To this end, we assemble a questionnaire with 80 questions addressing relevant aspects, including setup, support, documentation, usability, and technical features. In addition, we conduct a user study with domain experts to investigate the importance of these aspects for distinct user roles. Based on the evaluation and user feedback, we rank the tools and find MITRE Caldera, Metasploit, and Atomic Red Team on top.

Index Terms—adversary emulation, open-source tools, user study, red teaming

I. INTRODUCTION

The ever increasing number and sophistication of cyber attacks poses many challenges to organizations [1], [2]. As a consequence, security specialists employ a wide range of countermeasures, including technical approaches for automatic monitoring and intrusion detection [3], educational measures such as security awareness trainings [4], policies or strategic approaches such as incident response planning [5], vulnerability management [6], risk assessments [7], and situational awareness [8], among many more. However, even with many diverse measures in place and functional, it is often difficult to assess their effectiveness and completeness since blind spots are easily missed [9].

It is thus common practice to conduct regular tests that challenge security defenses of organizations. The term penetration testing generally refers to any activities related to the execution of cyber attacks for the purpose of identifying risk areas and weaknesses in applications, systems, and networks [10]. The main purpose of penetration testing is to proactively recognize security issues and improve the organization’s security posture before an actual attacker with malicious intent is able to intrude the network and cause any damage [11].

Red teaming has emerged as one of the most realistic and advanced strategies to conduct comprehensive security tests. Thereby, red teams, which comprise people with diverse skills in cyber security domains and beyond, simulate adversaries and conduct planned attack exercises without informing defenders in advance [12]. In contrast to vulnerability assessment and penetration testing, which only achieve to identify weaknesses and assess the risk associated with them, red teaming relies on interaction between attackers and defenders and thereby facilitates training of security teams and improvement of their detection and response capabilities [13].

Unfortunately, the implication of such an interactive exercise is that red teaming is a mostly manual process that incurs significant costs as it involves time-consuming tasks conducted by experienced personnel [14]. To alleviate this issue, adversary emulation tools are frequently employed to automate some of the tasks in red teaming exercises, which helps practitioners to establish a baseline of security measures. Thereby, capabilities of these tools range from simple technique execution to full emulation of an adversary [14], [15].

While there is obviously some benefit in having a large pool of adversary emulation tools to choose from, it is often difficult to do so when one tries to select a suitable emulator for a specific use-case and there is only limited time to explore and experiment with several of the tools [16]. For example, in situations where security professionals aim to conduct attacks against very specific components and applications, suitable tools should enable the creation of custom attack procedures and not just support a predefined set of attack cases. Other use-cases could revolve around more basic attack scenarios, but require that they are executed by personnel without red teaming experience, in which case usability of the tool is more important than advanced technical capabilities. Anyway, it is necessary to consider the needs and experience of the users of adversary emulation tools to make an informed decision.

There are several challenges that need to be addressed. First, it is non-trivial to identify all properties of adversary emulation tools that should be taken into account. Second, it is difficult to differentiate which properties are more relevant than others, in particular, since users in different roles may have distinct requirements and needs. Third, it is very time-consuming to review multiple tools in a hands-on manner to evaluate their appropriateness for certain use-cases. To

address these challenges, we conduct a structured review and comparison of adversary emulation tools comprising (i) a literature study to collect relevant properties that we assemble into a questionnaire, (ii) technical assessment with hands-on experiments using a set of nine tools, and (iii) a user study to ascertain the importance of properties for certain user roles. In our work we thus answer the following research questions: *RQ1: What properties of adversary emulation tools are the most relevant for stakeholders? RQ2: Which adversary emulation tools are best suited to fulfill the needs of certain user groups?*

To the best of our knowledge, this work provides the most comprehensive review and comparison of adversary emulation tools. Other studies do not assess how important users rate certain properties of tools [16], focus on detectability of tools [17], [18], are limited to specific operating systems [19], [20], or consider fewer questions [16] and tools [17] for evaluation. Our study, on the other hand, aims at a broad comparison of open-source tools. We summarize our contributions as follows:

- A questionnaire capturing properties and features of adversary emulation tools,
- an online survey for assessment of relevance scores for various aspects of these tools, and
- an evaluation and ranking of publicly available tools.

The remainder of the paper is structured as follows. Section II summarizes the background of red teaming and related works in the research field of adversary emulation tools. Section III describes the methodology of our study, including tool selection, questionnaire design, survey design, tool evaluation, and scoring. Section IV provides an overview and brief description of all selected tools. We present the results the evaluation study and online survey in Sect. V and answer our research questions in Sect. VI. Section VII concludes the paper. We provide our questionnaire in Appendix A.

II. BACKGROUND & RELATED WORK

The term *red teaming* is often incorrectly used interchangeably with other types of security tests, in particular, penetration tests. Kovačević et al. [12] thus conduct a literature study on security tests. They conclude that while penetration tests are short-term exercises that validate an organization's security posture, red teaming provides an ongoing training of security personnel. The authors also state that beside simulating real attackers, red teams also support organizations by acting as devil's advocates or consultants.

Given their definition, it is easy to understand that red teaming is a mostly manual and interactive task that is difficult to automate; nonetheless, several authors of scientific works have proposed adversary emulation tools in an attempt to take steps in that direction. For example, Plot et al. [21] propose a Cyber Automated Red Team Tool (CARTT) that provides an easy-to-use interface to carry out vulnerability scans and obtain recommendations on how to mitigate threats based on its findings. Another example is LACCOLITH, an agent presented by Orbinato et al. [18], that was specifically designed to evade detection by antivirus. Chen et al. [19], on

the other hand, design an adversary emulation tool for both red and blue teaming exercises. However, their tool is only designed for the macOS operating system.

Miller et al. [14] state that adversary emulation tools can be a cost-effective alternative to red teaming events. They mention several benefits of adversary emulation tools, such as providing defenders with a view on their network from the point of an attacker, identifying weaknesses or misconfigurations, testing of deployed security measures, and providing empirical evidence for a defensive blue team. The authors state that adversary emulation tools should be (i) intelligent, i.e., select and chain actions similar to an actual adversary, (ii) usable, i.e., require low overhead for utilization, (iii) realistic, i.e., execute attack techniques in such a way as they occur in the real world, and (iv) modular, i.e., allow users to customize techniques and create new attack procedures.

These and similar requirements are at the core of scientific reviews and surveys on the topic of adversary emulation. Zilberman et al. [16] provide one of the most comprehensive surveys, comprising eleven tools and 45 criteria. One of the focus points of their study is to assess how many attack techniques from MITRE ATT&CK are covered by each tool. In addition, they evaluate compatibility with operating systems, prerequisites for installation and running the tools, ease-of-use, documentation, and many technical features such as logging, cleanup, and the creation of custom attack scenarios. The main difference to our work is that we conduct a study with domain experts to better understand and weight the importance of each of these requirements. Contrary to their work, we also put less focus on the coverage of attack techniques and instead include more fine-granular questions on other aspects, which we outline in Sect. III.

Other studies on adversary emulation tools have a more narrow focus. Orbinato et al. [18] compare multiple adversary emulation tools with respect to their ability to evade detection by antivirus. Their results suggest that their own approach, which involves an agent that resides in the kernel, is more reliable in evading detection than most other tools, which require to manually configure exceptions in detection tools. The issue of detection is also studied by Elgh et al. [17], who analyze the number of Sysmon log events generated on a Windows target host while four adversary emulation tools are actively used to attack the machine. Since log events are a main source for intrusion detection, it is essential that automated attack simulations do not produce significantly more or more severe events than manual execution of attacks. However, their comparative study suggest that most tools are too noisy to represent realistic attacks from actual advanced persistence threats. Stockenreiter et al. [20] compare four tools with respect to their coverage of MITRE ATT&CK techniques when it comes to predefined attack procedures targeting Windows and Active Directory. The study shows that most tools only have limited capabilities in Windows environments. Other than these works, our survey is not targeted at specific properties of adversary emulation tools but instead relies on user studies to identify important aspects.

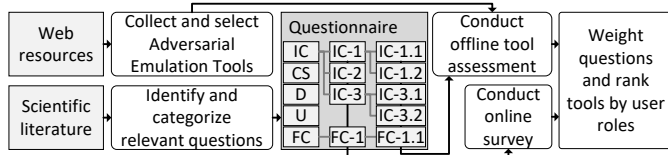


Fig. 1. Overview of the methodology of our research.

III. METHODOLOGY

This section outlines our research methodology, including strategies to select adversary emulation tools and create questionnaires for tool evaluations and user requirements analysis.

A. Overview

Figure 1 depicts our methodology as a flowchart. Initially, we select a set of adversary emulation tools for our study by conducting a web search for well-known open-source tools (cf. Sect. III-B). Based on literature research and insights gained from related studies, we then assemble a questionnaire that enables assessment of these tools. Thereby, we consider the following five categories of questions:

- **Installation & Configuration (IC).** Preparing systems and configurations in such a way to enable effective utilization of certain tools can often be time-consuming and pose a burden to analysts. This category covers several sub-categories, including compatibility of the tool with different operating systems, requirements on third-party tools, and tool configurability.
- **Community & Support (CS).** An active community behind an open-source tool is highly valuable when questions or issues emerge during its setup or utilization as they provide rapid support and may even fix or extend the tool in future releases following user feedback. This category of questions aims to assess aspects regarding the popularity of the repository as well as the activity of involved developers.
- **Documentation (D).** The availability and extensiveness of documentation is an important source of information for analysts who deploy and use the tool. This category covers metrics such as the comprehensiveness, currency, comprehensibility, and standardization of tool documentation.
- **Usability (U).** Well-designed user interfaces can vastly improve user experience when interacting with a tool. Questions in this category thus focus on assessing the intuitiveness, appearance, and customizability of available user interfaces, such as graphical user interfaces or command line interfaces.
- **Features & Capabilities (FC).** While some core functionalities required for adversary emulation are shared among all tools, some of them come with additional features and capabilities that are essential for certain use-cases. The sub-categories in this group of question address relevant technical aspects such as the overall workflow, restoring target systems after attacks, reporting and logging, chaining of attacks, attack automation, coverage of diverse

attack techniques, customization and scripting of attacks, and flexibility of attack execution.

The central part of Fig. 1 visualizes the structure of the questionnaire; specifically, each category of questions (e.g., *IC*) are divided into one or more sub-categories (e.g., *IC-1*), which in turn comprise one or more questions (e.g., *IC-1.1*). We process the questionnaire in a two-step procedure. First, we conduct an offline evaluation and assess how each question is fulfilled by every tool through experimentation (cf. Sect. III-C). Second, we carry out an online survey where we ask stakeholders about their preferences and requirements on adversary emulation tools (cf. Sect. III-D). As a final part of our study, we combine the results from the offline evaluation and online survey to assign a single score to each tool for the purpose of ranking (cf. Sect. III-E). In the following sections, we describe the steps of our methodology in more detail.

B. Selection of adversary emulation tools

For our comparative study, we aim to select a manageable set of well-known open-source tools for adversary emulation. To ensure that the identified tools are established technologies of practical relevance, we scan through articles comparing various penetration testing tools¹ and validate with curated lists of threat hunting software² and tools analyzed in scientific state of the art [16]–[18]. Thereby, we only include tools that are suitable for attack simulation and exclude tools for other types of security analytics, such as NSA Unfetter³ that enables vulnerability identification but does not support attack execution. Moreover, we focus on stand-alone tools and exclude frameworks that only build on top of such tools, such as MATE⁴, Purple Team ATT&CK⁵, Splunk Attack Range⁶, or ATT&CK Simulator⁷.

Eventually, we end up with the following nine open-source tools for adversary emulation (sorted alphabetically): ATTPwn⁸, Atomic Red Team⁹, APTSimulator¹⁰, MITRE Caldera¹¹, DumpsterFire¹², Infection Monkey¹³, Invoke Adversary¹⁴, Metasploit¹⁵, and Purplesharp¹⁶. We provide descriptions for each of these tools in Sect. IV. We point out that several commercial products for adversary emulation exist but are excluded from this study for licensing issues.

¹<https://tinyurl.com/wayback-2021/pentestit.com/adversary-emulation-tools-list/>

²<https://github.com/0x4D31/awesome-threat-detection>

³<https://nsacyber.github.io/unfetter/>

⁴<https://github.com/fugawi/mate>

⁵<https://github.com/praetorian-inc/purple-team-attack-automation>

⁶https://github.com/splunk/attack_range

⁷<https://github.com/timfrazier1/AdversarySimulation>

⁸<https://github.com/Telefonica/ATTPwn>

⁹<https://github.com/redcanaryco/atomic-red-team>

¹⁰<https://github.com/NextronSystems/APTSimulator>

¹¹<https://github.com/mitre/caldera>

¹²<https://github.com/TryCatchHCF/DumpsterFire>

¹³<https://github.com/guardicore/monkey>

¹⁴<https://github.com/CyberMonitor/Invoke-Adversary>

¹⁵<https://github.com/rapid7/metasploit-framework>

¹⁶<https://github.com/mvelazco/PurpleSharp>

C. Offline Tool Assessment

This section describes how we assembled and subsequently answered the questions from the questionnaire. We used the study by Zilberman et al. [16] as a guide to identify and validate relevant questions for most categories. Specifically, we use six questions on prerequisites and compatibility from that study in category *IC*, six questions about comprehensiveness of documentation for category *D*, six questions about availability of core functionalities over various interfaces (e.g., attack execution and configuration over graphical or command line interface) for category *U*, and 29 questions on technical aspects for category *FC*. Based on the work by Joy et al. [22], who analyze open-source projects and validate that their performance can be assessed through public metrics such as number of forks, we formulate six questions for category *IC*. Aversano et al. [23] propose quality indicators for documentation of open-source software, which we use to formulate eight questions in category *D*. Richter et al. [24], state several criteria for usability in human-machine-systems, which we use to formulate 13 additional questions for category *U*. Based on our own experience, we pose five questions that deal with attack chaining, blue teaming, and tool workflow in category *FC* and one question that checks conformity to the *IEEE Standard for User Documentation* [25] in category *D*.

Eventually, we end up with 80 questions (see Appendix A for a list of all questions) that we use to evaluate each of the selected adversary emulation tools. We answer questions from categories *CS* and *D* by reviewing the documentation and resources provided in the respective repository of the tool. Regarding questions from categories *IC*, *U*, and *FC*, we proceed by installing each adversary emulation tool in a virtual environment and testing the features according to the questions, e.g., by executing attack cases. The setup comprises a Kali-Linux¹⁷ machine (Version 2024 with 4 GB RAM and 60 GB disk space) that acts as a command-and-control server, two Metasploitable3¹⁸ machines (Ubuntu 14.04 with 4 GB RAM and 20 GB disk space as well as Windows Server 2008 with 4 GB RAM and 40 GB disk space) that act as target systems, and two Windows machines (Versions 10 and 11, each with 8 GB RAM and 80 GB disk space) that act as both command-and-control servers and target systems. We also cross-check the documentation to ensure that no functions supported by the tools are missed or misapplied.

Similar to the survey conducted by Zilberman et al. [16], our assessment scheme consists of a four-point-scale, where we assign 0 points if the question is not fulfilled, 1 point for partial fulfillment, 2 points when the question is mostly fulfilled, and 3 points if it is entirely fulfilled. For questions that are answered with yes or no, we assign 3 and 0 points respectively. Since category *CS* comprises quantitative questions, we assign 3 points if the retrieved value is in the top 25% of all gathered values for that question, 1 point if it is in the bottom 25%, and 2 points if it is in between.

¹⁷<https://www.kali.org/>

¹⁸<https://github.com/rapid7/metasploitable3>

D. Online Survey of User Requirements

The second part of our study aims to assess which of the aforementioned functions and properties of adversary emulation tools are most important for stakeholders to enable weighting and ranking of tools. To this end we invite domain experts to an online survey and ask them to assign relevance scores to each of the 30 sub-categories of questions. We opted for sub-categories rather than the entire questionnaire (c.f. Sect. III-C) to reduce the number of questions from 80 to 30 and avoid asking for many technical specifics. One of them is a general statement *G-1: Tool is available for free* that we add even though it is not related to any category, but only used to assess the preference of freely available tools in contrast to commercial products. We design the survey following the work by Achimugu et al. [26], who outline the Fuzzy Multi-Criteria Decision-Making (FMCDM) method for prioritization of software requirements. In particular, participants can rate the importance of each sub-category on a five point scale ranging from *Not Important* to *Very Important*, also including *No answer* as an additional option.

Beside these questions, we ask participants to state their professional experience, where we differentiate between categories *Low* (less than 3 years), *Medium* (3 to 6 years), and *High* (more than 6 years), as well as their expertise with adversary emulation tools, where we differentiate between categories *Low* (not familiar), *Medium* (somewhat familiar), and *High* (very familiar). In addition, we ask participants about their current job position to identify differences in preferences across groups of users. Specifically, we group all participants to one of the following roles: *Leaders* (i.e., executives and managers), *Security Architects* (e.g., security engineers and system administrators), *Security Consultants*, *Security Analysts*, and *Security Researchers*.

The survey was hosted on a web platform from January 15, 2024, to March 15, 2024, and shared through a link that we posted on cyber security mailing lists and within project consortia to attract participants. At the beginning of the survey, all participants were informed about the purpose of the survey and consented that their responses will be published anonymously as part of this study.

E. Feature Weights and Tool Ranking

We combine the results obtained from our offline tool evaluation with the responses from our online survey to weight properties of adversary emulation tools and create rankings for certain user groups. To this end we map the five point scale of user importance ratings to numeric weights as follows: *Not Important* has a weight of 0.5, *Rather Unimportant* has a weight of 0.75, *Neutral* has a weight of 1, *Important* has a weight of 1.25, and *Very Important* has a weight of 1.5. For each user role, we then compute the weight of each sub-category of questions as the average of all responses from participants that belong to that role. In the following, we denote the average weights as $\bar{w}_{subcat(q),role}$, where $subcat(q)$ is the sub-category of question q .

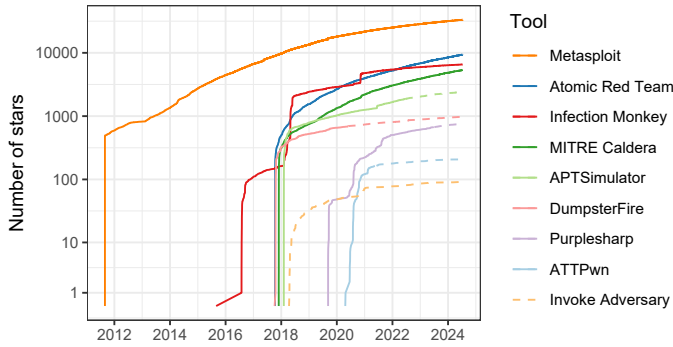


Fig. 2. Progression of received stars for the respective GitHub repositories of each tool. Transition from solid to dashed lines indicate the last commit made to the repository.

TABLE I
OPERATING SYSTEM SUPPORT AND REQUIREMENT TO INSTALL AGENTS
ON TARGET SYSTEMS FOR ATTACK EXECUTIONS

Tool	Supported Operating Systems			Agents
	Windows	Linux	MacOS	
ATTPwn	✓			✓
Atomic Red Team	✓	✓	✓	
APTSimulator	✓			
MITRE Caldera		✓	✓	✓
DumpsterFire	✓	✓	✓	
Infection Monkey	✓	✓		✓
Invoke Adversary	✓			
Metasploit	✓	✓	✓	~
Purplesharp	✓			✓

To compute an overall score for one of the reviewed adversary emulation tools, we iterate over the list of questions Q_{tool} that we evaluated in the offline tool assessment and multiply the score $score(q) \in \{0, 1, 2, 3\}$ of question q with the weight corresponding to $subcat(q)$. As depicted in Eq. 1, we compute the final score $s_{tool,role} \in [0, 1]$ by normalizing with $3 \cdot |Q_{tool}|$, since 3 is the maximum score for each question.

$$s_{tool,role} = \frac{1}{3 \cdot |Q_{tool}|} \sum_{q \in Q_{tool}} score(q) \cdot \bar{w}_{subcat(q),role} \quad (1)$$

Note that the total highest achievable number of points is different for each tool, because there are two questions in sub-category FC that are only applicable to tools that rely on agents. Accordingly, these questions are not evaluated for tools without agents and thus do not contribute to their scores.

IV. ADVERSARY EMULATION TOOLS

This section briefly describes each of the selected adversary emulation tools and highlights some of their properties that differentiate them from others. Since all tools are available on GitHub, we provide a visual summary of the age and popularity of the respective repositories in Fig. 2. The change from solid to dashed lines indicates the point in time where the most recent commit was made to the repository. As visible in the plot, the most popular repositories are still maintained while some of the other repositories have not been updated for

years and can be considered as discontinued by developers. As visible in the plot, most repositories receive a significant amount of stars in the first few weeks after release and then continue to slowly grow in popularity over time.

We also summarize the compatibility of each tool with operating systems as well as the need to install an agent on the target system in Table I. The overview shows that Windows is the most widely supported operating system, but popular repositories (according to Fig. 2) also support alternative operating systems. In the following, we go through each tool in alphabetical order.

ATTPwn emulates threats following the schema defined by MITRE ATT&CK and specifically covers many techniques from the *Defense Evasion* and *Discovery* stages. The tool focuses on Windows since installation requires Powershell 3.0 or above and most available attack techniques target Windows machines, even though procedures for other operating systems are available. Running the attacks requires that an agent is downloaded from the command-and-control server to the target system, which requires to add exceptions to firewalls and antivirus. The graphical user interface is simple and well-arranged, but does not provide any support or feedback for users. Nonetheless, it allows to review and modify attack scripts, even for attack chains. The most striking difference to other tools is the lack of documentation that is limited to a short readme file and some video tutorials.

Atomic Red Team comes with a large library of predefined attack procedures, which is one of its key features in comparison to other tools. Moreover, it includes scripts in markdown and YAML format that support efficient setup of reproducible and portable test environments. Both Atomic Red Team and its attack-executor Invoke-Atomic are simple to install on all common operating systems. The framework provides a graphical user interface that allows efficient generation of attack procedures as well as a command line interface to execute attack procedures from the library. Knowledge of the Python programming language is required to modify, combine, or create entirely new attack steps. Due to the straightforward design of the tools, the comparatively short documentation is sufficient to understand and use all available functions.

APTSimulator is a Windows batch script that emulates a compromised Windows system without the need for an agent. Due to the straightforward nature of the tool, the short documentation in form of a readme file is sufficient to retrieve and run the script. However, we noticed during our experiments that it is necessary to apply exceptions to antivirus systems in Windows in order to run the script. Attack procedures are organized in a schema similar to MITRE ATT&CK and can be executed in batches; in particular, all available attacks can be run in a sequence. Creation of new attack cases is not supported through the command line interface, but requires to modify existing batch scripts or create new ones.

MITRE Caldera is designed to support cyber security teams with autonomous and reproducible attack simulations that are useful for testing of detection, analysis, and response capabilities. Interestingly, it is the only tool that is not com-

patible with Windows, since the command-and-control server needs to be installed either on Linux or MacOS. The tool excels in terms of quality of documentation, which contains comprehensive explanations of all relevant components, tutorials for execution of attacks, and even a chapter for developers. In addition, MITRE Caldera offers interactive training to learn the basic features of the tool.

The tool also stands out from the others due to the combination of high usability and extensive number of features. The graphical user interface is accessible through a browser and provides configuration options, an enumeration of available attack procedures, and an overview of statuses for agents as well as currently ongoing attacks. Attack executions may be stopped or interrupted to add new procedures. To run the attack simulations, agents must be installed on the target systems, which can also be carried out through the graphical user interface. Similar to most other tools, agent installation requires exceptions in firewall and antivirus on target systems.

Beside preconfigured attack procedures, MITRE Caldera integrates procedures from other frameworks such as Metasploit. Users have the possibility to create new attack procedures as well as to modify existing ones, arrange them in chains for parallel or sequential execution, and specify cleanup functions that are executed after the fact. The tool also provides comprehensive logging and reporting functionalities, which support output in JSON, CSV, text, or PDF files that even contain visualizations of the compromised infrastructure.

DumpsterFire is a platform-independent tool based on the Python programming language that is designed to generate repeatable, delayed, and distributed security events. Users may change existing and add new attack procedures as Python scripts. Other than for most of the reviewed tools, the predefined attack procedures are not categorized following any model such as MITRE ATT&CK and include activities without direct security implications, such as opening a browser and playing videos. Moreover, the tool lacks some features present in other tools, such as cleanup functions to restore target systems or the option to stop currently ongoing attack procedures. DumpsterFire only provides a command line interface with some basic help pages as well as a readme file. While the console output itself can be regarded as a basic report, the tool does not produce any logs or output files. Given that there is only a single contributor to the repository and the most recent commit dates to the year 2020, we assume that the project has been discontinued.

Infection Monkey is designed to test security solutions. The tool relies on a worm-like agent on the target system that scans the network, executes attack procedures, and simulates lateral movement. Both agent and command-and-control server are compatible with Windows and Linux. The command-and-control server is a web server providing a graphical user interface that allows users to start and configure attack procedures or store them as reusable templates. Moreover, it provides a so-called Infectionmap showing compromised infrastructures as well as information about target hosts. As for most tools, firewall and antivirus block both the agent and server and must

be disabled. Infection Monkey generates detailed log files for executed procedures and allows to generate a comprehensive report. The documentation covers all functions of the tool, but does not explain how users can create their own attack procedures, which requires programming skills and cannot be accomplished in the graphical user interface.

Invoke Adversary is a Powershell script useful to assess security tools. Similar to APTSimulator, it is straightforward to get started since there is no installation or deployment of agents required; running the script is sufficient. The tool comprises a menu in the command line interface that allows to select and execute attack procedures. As there are no predefined techniques for lateral movement available, the tool is primarily useful to assess endpoint detection systems. To generate new attack procedures, the script itself must be adapted accordingly. Logging of executed procedures is limited to the command line as no log files or reports are stored. Note that exceptions for antivirus are necessary to run the tool. The community and support behind the tool appears to be comparatively limited, and there have not been any updates for the tool since its initial release.

Metasploit is the oldest and by far the most popular penetration testing framework among all reviewed tools. It is backed up by a large and active community of users and developers and mainly used to detect and assess vulnerabilities of systems and networks. In contrast to other tools from our study that either require an agent on the target system or not, Metasploit finds a middle ground as its Meterpreter allows to establish an interactive shell, execute commands, and reload payloads on the target similar to an agent. Attack procedures may be represented as reusable templates and chained together; however, to generate new procedures, users need to define them in scripts in the Ruby programming language.

Logging in Metasploit is configurable for different log levels. While Metasploit is a command line tool, we point out that there is also a commercial version of Metasploit that provides a graphical user interface but is excluded from our study as we only focus on open-source tools. Metasploit comes with 875 pages of documentation describing all modules in detail and providing guidance for developers; this is the most comprehensive documentation of all reviewed tools.

Purplesharp is written in C-Sharp and designed to carry out attacks on Windows Active Directory. On the target host, three agents are used for reconnaissance, attack execution, and orchestration of attacks. Antivirus tools need to be disabled to run the agents. The tool is started through the Windows command line, but also provides a graphical user interface in a browser that can be used to configure available attack procedures. To generate new attack procedures, however, programming skills are necessary. In general, the simple nature of the tool enables automation and chaining of attack procedures as well as inclusion of cleanup functions through playbooks in JSON format. The documentation is sufficient to install the tools and carry out predefined attack procedures. Purplesharp also produces log files containing relevant information from executed attack procedures.

V. EVALUATION

This section contains our evaluation results. We first present a qualitative comparison of adversary emulation tools based on our offline evaluation. Subsequently, we provide an overview of the prioritized user requirements that we obtain from our online survey. Finally, using the weighted features, we rank adversary emulation tools for different user roles.

A. Technical Comparison

We conducted the offline evaluation of adversary emulation tools according to our outline from Sect. III-C. Table II summarizes the results of the evaluation. For each category of questions, we state the total number of points achieved by an adversary emulation tool (*Pts. abs.*), the maximum number of achievable points (*Pts. max*), and the relative number of achieved points (*Pts. rel. (%)*). Note that the maximum number of points is different in each category but also within category *FC* since some questions only apply to agent-based tools and are not evaluated for tools that do not make use of agents.

In category *IC*, most tools achieve relatively high points, with the minimum score of 72.2% being achieved by four tools. This indicates that technical obstructions that make it difficult to get started with a tool are quite successfully kept to a minimum. In general, most points in this category are deducted for incompatibility with common operating systems (*IC-1.1*) and required changes to security settings (*IC-1.2*). In particular, running the tools often requires to add exceptions in the firewall and antivirus of systems where the tool itself or its agents are deployed. Atomic Red Team is the only tool to achieve the highest possible score of 18 points as it is compatible with all three considered operating systems and does not rely on agents. MITRE Caldera, despite relying on agents, notably achieves the second best score with 17 points.

Given that Metasploit is by far the most widely used tool for adversary emulation, it is not surprising that it achieves the highest possible score of 18 points in category *CS*. For comparison, even though Atomic Red Team achieves 17 points in that category, Metasploit has around three times as many contributors (*CS-1.2*) and five times as many forks (*CS-1.1*) as Atomic Red Team. Regarding category *D*, the table shows that MITRE Caldera yields the highest score and is closely followed by Atomic Red Team and Metasploit, while many other tools fall behind. Our analysis reveals that most documentations suffer from low readability scores (*D-4.1*) and leave out descriptions on how to interpret the output of executed procedures (*D-1.6*), generate custom procedures (*D-1.4*), or arrange procedures as chains (*D-1.5*). Captions of figures are also missing in almost all documentations (*D-5.1*).

Category *U* turned out to be a critical one, where many tools only yield comparatively few points. MITRE Caldera yields the highest score with 36 out of 57 points and is followed by Infection Monkey with 31 points; other tools all achieve only around 20 points. This is primarily caused by limitations of user interfaces (*U-5.3-U-5.6*) and low flexibility (*U-7.2*), in particular, many essential functions are often only available through one interface (e.g., command line) but not

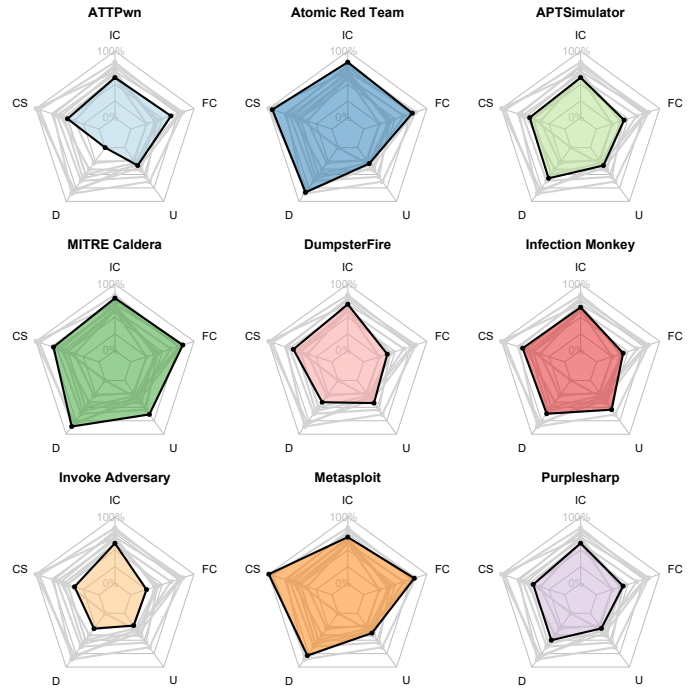


Fig. 3. Radar plots of scores for categories Installation & Configuration (IC), Community & Support (CS), Documentation (D), Usability (U), and Features & Capabilities (FC).

through another (e.g., graphical user interface). Other common issues include lack of customizability of interfaces (*U-6.1*) and limited guidance within the tool itself (*U-3.1*, *U-3.2*). One a positive note, most tools yield reproducible results (*U-1.1*), are hardly affected by errors (*U-1.3*), and their core functions are easy to access and efficient to execute (*U-7.1*).

High diversity of results is also prevalent in category *FC*, with MITRE Caldera (84 out of 102 points), Metasploit (77 out of 96 points), and Atomic Red Team (74 out of 96 points) again forming the top. The main reasons for point reductions are issues with firewall and antivirus during operation (*FC-2.2*, *FC-2.3*, *FC-2.6*, *FC-2.7*), limited control of attack execution (*FC-5.3*, *FC-6.1*, *FC-9.1*), unsuitable presentation of results of attack executions (*FC-4.3*), and lack of blue teaming functionalities (*FC-7.4*).

The radar plots displayed in Fig. 3 provide a visual overview of the scores achieved by each tool. The plots show that Atomic Red Team and Metasploit have comparatively high and similar scores across all categories. MITRE Caldera exceeds them in terms of usability (category *U*) but falls behind when it comes to support from the community (category *CS*). The radar plots of DumpsterFire, APTSimulator, Infection Monkey, and Purplesharp have somewhat similar shapes that indicate medium fulfillment across all categories. ATTPwn would fall into a similar range, except that it yields the lowest possible score in category *D*. Finally, Invoke Adversary appears to be at the lower end of the spectrum in most categories.

Figure 4 visualizes the scores of pairs of categories for all tools. The top plot shows that Atomic Red Team is ahead of all other tools for categories *CS* and *IC*. The center plot

TABLE II
RESULTS OF THE OFFLINE EVALUATION

	Installation & Configuration (IC)			Community & Support (CS)			Documentation (D)			Usability (U)			Features & Capabilities (FC)		
	Pts. abs.	Pts. max.	Pts. rel. (%)	Pts. abs.	Pts. max.	Pts. rel. (%)	Pts. abs.	Pts. max.	Pts. rel. (%)	Pts. abs.	Pts. max.	Pts. rel. (%)	Pts. abs.	Pts. max.	Pts. rel. (%)
ATTPwn	13	18	72.2	9	18	50.0	0	42	0.0	19	57	33.3	65	102	63.7
Atomic Red Team	18	18	100.0	17	18	94.4	35	42	83.3	17	57	29.8	74	96	77.1
APTSimulator	13	18	72.2	10	18	55.6	24	42	57.1	19	57	33.3	42	96	43.8
MITRE Caldera	17	18	94.4	13	18	72.2	36	42	85.7	36	57	63.2	84	102	82.4
DumpsterFire	15	18	83.3	11	18	61.1	17	42	40.5	24	57	42.1	36	96	37.5
Infection Monkey	14	18	77.8	12	18	66.7	26	42	61.9	31	57	54.4	43	102	42.2
Invoke Adversary	13	18	72.2	7	18	38.9	12	42	28.6	13	57	22.8	24	96	25.0
Metasploit	15	18	83.3	18	18	100.0	33	42	78.6	21	57	36.8	77	96	80.2
Purplesharp	13	18	72.2	9	18	50.0	21	42	50.0	16	57	28.1	43	102	42.2

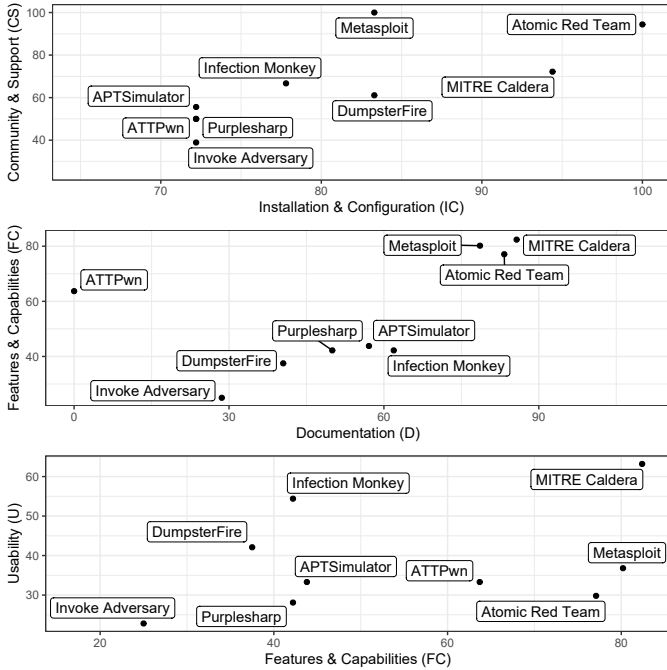


Fig. 4. Pairwise comparison of category scores.

compares categories *FC* and *D*, which reveals that MITRE Caldera, Atomic Red Team, and Metasploit form a group that outperforms all other tools. While AttPwn comes close in terms of *FC*, it yields the lowest score in terms of *D*. The bottom plot shows that MITRE Caldera is the only tool that combines high scores in categories *U* and *FC*. Metasploit and Atomic Red Team have comparable scores in category *FC* but fall behind in terms of *U*, and the exact opposite is the case for Infection Monkey. We provide a unified ranking that considers all categories in Sect. V-C, but first present the results of our user requirement study in the following section that we subsequently use for weighting aforementioned criteria.

B. Online Survey Results

This section contains the results of our online survey. We first provide some details on the participants of our survey and then present an overview of their responses.

TABLE III
ONLINE SURVEY PARTICIPANTS

Role	Job Title	Experience	Expertise
Leaders	Head of Cyber Security Research*	High	High
	Head of Research*	High	High
	Chief Information Security Officer	High	Medium
	Head of department	High	Medium
	Head of department	High	High
Security Architects	Senior Security Engineer	High	High
	Enterprise Security Architect	High	Low
	Security Engineer	High	High
	DevSecOps Administrator	High	Low
Security Consultants	Senior Security Consultant	High	High
	Security Consultant	Low	Low
	Security Consultant	High	Medium
Security Analysts	SOC Analyst	Low	Medium
	Senior Penetration Tester	Medium	High
	Security Analyst	High	High
	Security Specialist	Medium	Low
Security Researchers	Head of Cyber Security Research*	High	High
	Head of Research*	High	High
	Scientist	Medium	Medium
Unknown	Research Engineer	High	Medium
	N/A	N/A	High
	N/A	High	Low

1) *Participants*: Following our methodology from Sect. III-D we published the survey over a period of two months, after which 20 domain experts fully completed the survey. Table III provides an overview of the meta information we collected from all participants, specifically, their job title, job experience, and expertise with adversary emulation tools. After manually reviewing the job titles we grouped each participant into one of five roles, except for two participants (*Head of Cyber Security Research* and *Head of Research*) that fit into roles *Leaders* and *Security Researchers* and whose responses are thus counted in both groups. These participants are marked with an asterisk in Table III. We opted for this setup to ensure that we base our estimations on sufficiently many users per role. Moreover, two participants did not state their job position and are thus assigned to the *Unknown* role. Across all roles, most participants state high job experience, which we consider as an indicator for high-quality responses. On the other hand, participants indicate mixed expertise with adversary emulation tools, providing us with a diverse set of opinions on requirements.

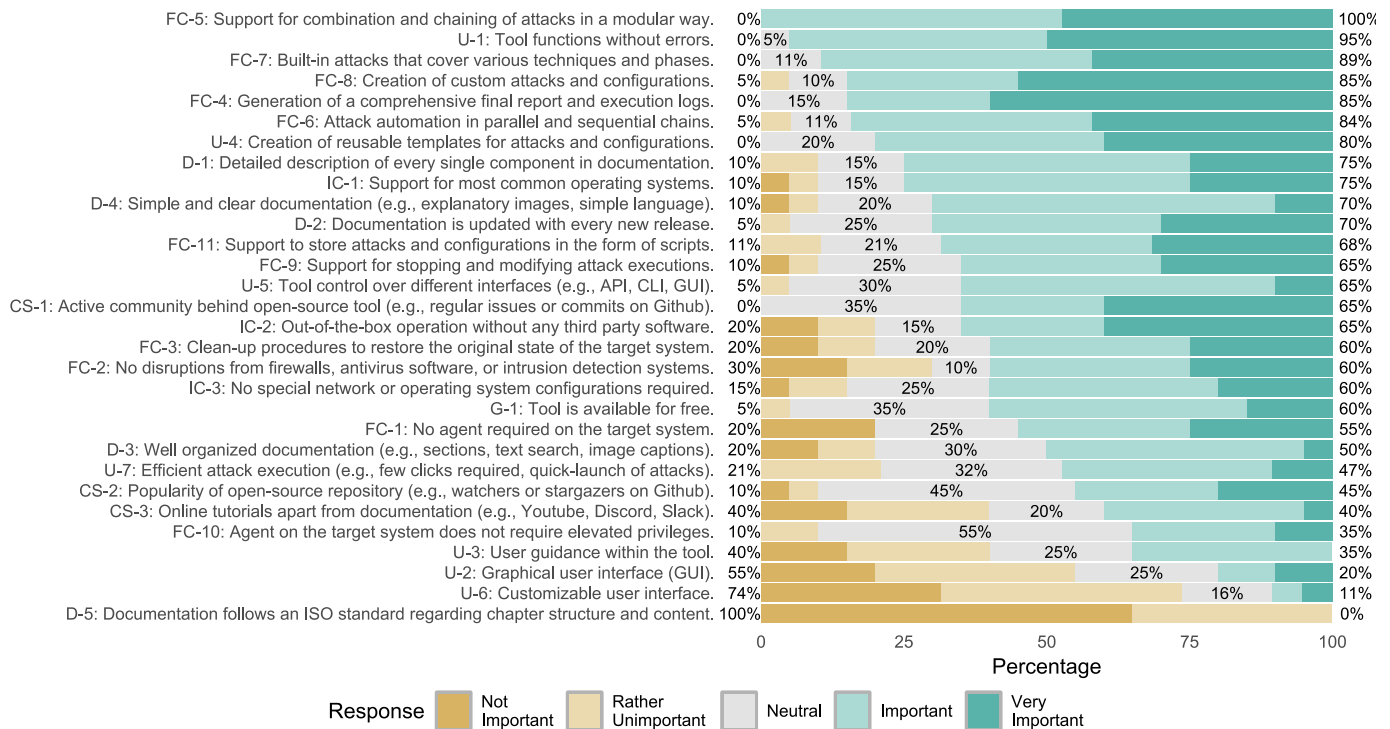


Fig. 5. Likert scale of domain experts rating the importance of aspects of adversary emulation tools.

2) *Responses*: We count the responses from all participants and visualize the resulting distributions in Fig. 5, which enumerates all 30 sub-categories of questions. The sub-categories in the plot are sorted by the relative number of positive responses, i.e., categories mostly rated as *Important* or *Very Important* appear in the top of the plot. Our survey reveals that technical features from category *FC* that primarily concern attack procedures form the most crucial requirements for adversary emulation tools, in particular, tools should come with a broad set of predefined attack procedures (*FC-7*), but also enable to generate new attack procedures (*FC-8*) and automate their execution in chains (*FC-5*, *FC-6*). Other relevant aspects are reporting (*FC-4*) as well as stability (*U-1*). The bottom part of the figure indicates that many aspects from categories *D*, *CS*, and *U* are considered less relevant.

We also investigate the responses with respect to roles, in particular, we counted the number of times participants of certain roles selected *Important* and *Very Important* for each category of questions. Our analysis suggests that *Leaders* prioritize *FC* and *CS* over other categories, *Security Architects* prioritize *CS* and *IC*, *Security Consultants* prioritize *CS* and *FC*, and *Security Analysts* prioritize *IC* and *FC*. Out of all roles, *Security Researchers* assign the highest weight to *FC* and the lowest weight to *U*. In the following, we make use of these role-specific weights to compare and rank tools.

C. Weighted Results and Tool Ranking

Using Eq. (1) stated in Sect. III-E, we are able to compute a single score for each tool and user role. Figure 6 visualizes these scores and ranks the adversary emulation tools in

ascending order according to their average score across all roles. As visible in the plot, MITRE Caldera is ahead of all tools but closely followed by Metasploit and Atomic Red Team that are roughly on par. These tools already emerged as top performers during our analysis of Sect. V-A, which suggests that weighting based on user requirements only has minor influence on the overall ranking. We also observe that Infection Monkey is slightly ahead, APTSimulator, ATTPwn, DumpsterFire, and Purplesharp all roughly achieve similar scores, and Invoke Adversary falls behind.

The influence of weights from different user roles does not change the overall ranking of the top three tools. While the ranks of other tools change, this is mostly due to the fact that their average scores are close together and minor influences on the scores are sufficient to affect the ranking. Note that we mapped the responses of survey participants to quantitative weights in a static way (cf. Sect. III-E), but this mapping can be changed to arbitrary magnitudes and thus have a stronger influence on the weighted scores. To obtain a better view on the influence of user roles on the tool ranking, we thus measure the deviation of each score from the mean score achieved across all roles and visualize the results in Fig. 7. The plots indicate that almost all tools are better suited to fulfill the needs of *Leaders* and *Analysts* than *Architects* and *Consultants*; there is no clear tendency for researchers. Moreover, functions and properties of some tools address the requirements from specific user groups, for example, Infection Monkey and Purplesharp yield significantly higher scores for *Security Analysts* than for any other role, where scores are either below or around the average across all roles. We emphasize that this interpretation

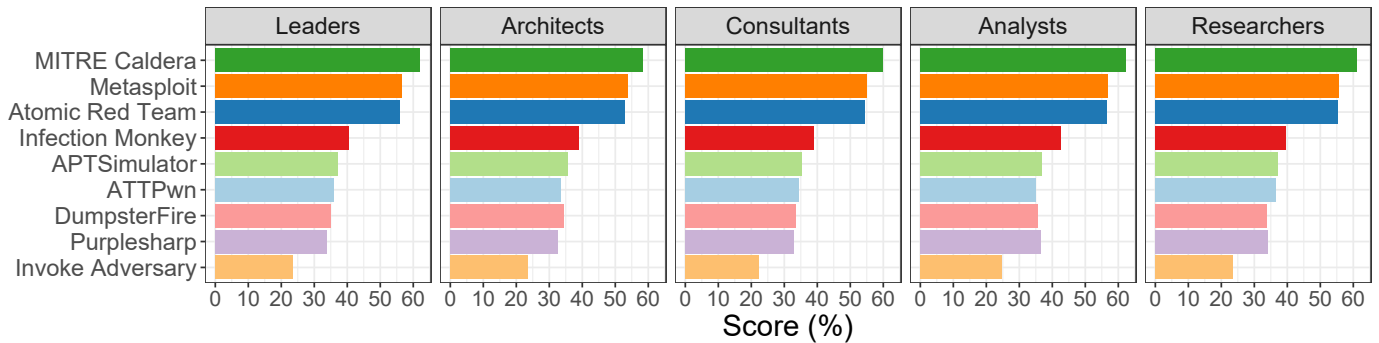


Fig. 6. Scores of adversary emulation tools show that the ranks of tools are mostly the same across user roles.

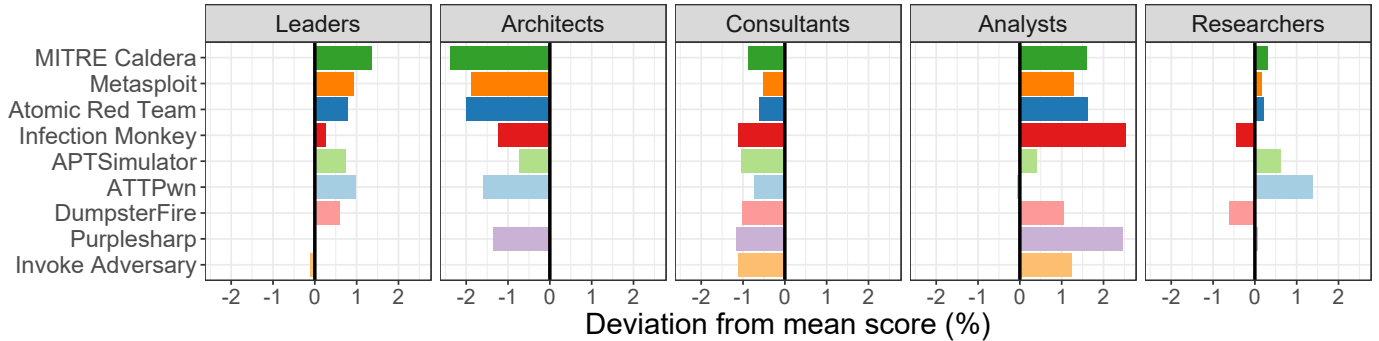


Fig. 7. Deviations from the mean score shows which tools involve functions and properties that align with the requirements of specific user roles.

of the results only indicate that the tools are well suited for the requirements of specific user groups in comparison to an average user and not that they are necessarily the most suitable tools to be used by the respective user group.

VI. DISCUSSION

In this paper we outline a methodology to assess and rank adversary emulation tools based on technical evaluations and user requirements. Our questionnaire, survey results, and interpretations can be an assistance for organizations and security professionals when it comes to the selection of adequate tools for certain use-cases. Thereby, we recommend to consider updating the weights of relevant aspects so that they better reflect the requirements of the respective situations, for example, high usability (category U) and availability of many predefined attack procedures (sub-category $FC-7$) may be beneficial when the tool should be used by novices.

We recognize some limitations of our work. The selection of tools (cf. Sect. III-B) is based on public sources, which can be subject to bias. While we ensured to consult multiple sources, they may fail to include suitable tools that have only gained little attention or tools that have only been released very recently. Moreover, we notice that our selected tools vary strongly in terms of popularity (e.g., community size and number of forks on GitHub) and that tools with stronger public support usually outperform those that have fewer contributors or have even been abandoned by developers. In particular, comparing Fig. 2 and Fig. 6 shows that the ranking of tools we obtain from our evaluation mostly followed the ranking based

on stars on GitHub. A notable exception to this observation is MITRE Caldera, which ranks first based on our evaluation but only fourth based on stars. Another limitation of our work is the small number of participants in our online survey. While we ensured that each group of users comprises at least 3 participants, we expect that a higher number of participants could yield more fine-granular insights into the differences between user roles. With these limitations in mind, we formulate the answers to our research questions as follows.

RQ1: What properties of adversary emulation tools are the most relevant for stakeholders? To answer this question, we worked out a questionnaire comprising categories and sub-categories of questions that address various aspects of adversary emulation tools. We then conducted a survey with domain experts to assess the relevance of each of the identified sub-categories. We present a detailed enumeration of user ratings in Sect. V-B2, which shows that specific technical features, including automation, configuration, and execution of attack procedures, are among the most relevant features for stakeholders. We also found that the ability to create reports for results and absence of errors are important features of tools.

RQ2: Which adversary emulation tools are best suited to fulfill the needs of certain user groups? Our analysis reveals that users have different priorities depending on their roles, for example, researchers prioritize technical capabilities over usability. We state the prioritized categories of questions for different user roles in Sect. V-B2 and analyze the influence of roles on the scores and ranking of adversary emulation tools in Sect. V-C. Considering the weighted scores, we identify

as MITRE Caldera, Metasploit, and Atomic Red Team as the most suitable tools across all user groups. These findings align with similar conclusions drawn from earlier studies [16].

VII. CONCLUSION

This paper presents a structured review and comparison of adversary emulation tools. Based on studies in related research areas, we design a questionnaire comprising five categories, namely *Installation & Configuration*, *Community & Support*, *Documentation*, *Usability*, and *Features & Capabilities*, with a total of 30 sub-categories and 80 questions. We select nine publicly available adversary emulation tools and evaluate to what degree they fulfill each of the questions. In addition, we conduct an online survey of domain experts to assign relevance scores to each sub-category of questions. Finally, we weight the evaluation results with user feedback to compute a single score for each tool. Our results suggest that independent from the user role, MITRE Caldera appears as the most suitable tool, followed by Metasploit and Atomic Red Team. For future work, we suggest to expand the range of adversary emulation tools to obtain more detailed insights into their peculiarities. In particular, we suggest to categorize tools and design questionnaires for each group to better capture and analyze properties that are specific to certain use-cases. Moreover, expert interviews that complement online surveys could be valuable to validate or come up with additional questions or categories of questions. Finally, while we only consider open-source tools in our survey, a similar study on commercial products could yield insights on specialized tools.

ACKNOWLEDGMENT

Parts of this work were carried out in course of a Master's Thesis at the University of Applied Sciences Technikum Vienna [27]. The work in this paper has received funding from the European Union - European Defence Fund under GA no. 101103385 (AInception) and GA no. 101121403 (NEWSROOM). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. The European Union cannot be held responsible for them.

REFERENCES

- [1] Mandiant, "M-trends 2023," <https://www.mandiant.com/m-trends>, 2023, online; accessed 2024-08-09.
- [2] CrowdStrike, "Global threat report 2024," <https://www.crowdstrike.com/global-threat-report>, 2024, online; accessed 2024-08-09.
- [3] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, 2019.
- [4] M. Zwilling, G. Klien, D. Lesjak, L. Wiecheteck, F. Cetin, and H. N. Basim, "Cyber security awareness, knowledge and behavior: A comparative study," *Journal of Computer Information Systems*, vol. 62, no. 1, pp. 82–97, 2022.
- [5] A. Ahmad, K. C. Desouza, S. B. Maynard, H. Naseer, and R. L. Baskerville, "How integration of cyber security management and incident response enables organizational learning," *Journal of the Association for Information Science and Technology*, vol. 71, no. 8, pp. 939–953, 2020.
- [6] R. Syed, "Cybersecurity vulnerability management: A conceptual ontology and cyber intelligence alert system," *Information & Management*, vol. 57, no. 6, p. 103334, 2020.
- [7] F. Cremer, B. Sheehan, M. Fortmann, A. N. Kia, M. Mullins, F. Murphy, and S. Materne, "Cyber risk and cybersecurity: a systematic review of data availability," *The Geneva papers on risk and insurance – Issues and practice*, vol. 47, no. 3, p. 698, 2022.
- [8] A. Ahmad, S. B. Maynard, K. C. Desouza, J. Kotsias, M. T. Whitty, and R. L. Baskerville, "How can organizations develop situation awareness for incident response: A case study of management practice," *Computers & Security*, vol. 101, p. 102122, 2021.
- [9] F. Skopik, M. Landauer, and M. Wurzenberger, "Blind spots of security monitoring in enterprise infrastructures: a survey," *IEEE Security & Privacy*, vol. 20, no. 6, pp. 18–26, 2022.
- [10] M. Alhamed and M. H. Rahman, "A systematic literature review on penetration testing in networks: Future research directions," *Applied Sciences*, vol. 13, no. 12, p. 6986, 2023.
- [11] A. A. Mughal, "Building and securing the modern security operations center (SOC)," *International Journal of Business Intelligence and Big Data Analytics*, vol. 5, no. 1, pp. 1–15, 2022.
- [12] I. Kovačević and S. Groš, "Red teams-pentesters, APTs, or neither," in *International Convention on Information, Communication and Electronic Technology*. IEEE, 2020, pp. 1242–1249.
- [13] M. Vos, "Capability maturity measurement of a security operations center through analysis detection," Master's thesis, University of Twente, 2022.
- [14] D. Miller, R. Alford, A. Applebaum, H. Foster, C. Little, and B. Strom, "Automated adversary emulation: A case for planning and acting with unknowns," MITRE, 2018.
- [15] A. Applebaum, D. Miller, B. Strom, H. Foster, and C. Thomas, "Analysis of automated adversary emulation techniques," in *Summer simulation multi-conference*, 2017, pp. 1–12.
- [16] P. Zilberman, R. Puzis, S. Bruskin, S. Shwarz, and Y. Elovici, "Sok: A survey of open-source threat emulators," *arXiv preprint arXiv:2003.01518*, 2020.
- [17] J. Elgh, "Comparison of adversary emulation tools for reproducing behavior in cyber attacks," Master's thesis, Linköping University, 2022.
- [18] V. Orbinato, M. C. Feliciano, D. Cotroneo, and R. Natella, "Laccolith: Hypervisor-based adversary emulation with anti-detection," *IEEE Transactions on Dependable and Secure Computing*, 2024.
- [19] Y.-H. Chen, Y.-D. Lin, C.-K. Chen, C.-L. Lei, and C.-Y. Huang, "Construct macOS cyber range for red/blue teams," in *Asia Conference on Computer and Communications Security*, 2020, pp. 934–936.
- [20] C. Stockenreiter, "Fähigkeitsanalyse von open-source Breach-and-Attack-Simulation-Tools in Windows- und Active-Directory-Umgebungen," Master's thesis, University of Applied Sciences Technikum Vienna, 2022.
- [21] J. Plot, A. Shaffer, and G. Singh, "Cartt: Cyber automated red team tool." HICSS, 2020.
- [22] A. Joy, S. Thangavelu, and A. Jyotishi, "Performance of github open-source software project: an empirical analysis," in *International Conference on Advances in Electronics, Computers and Communications*. IEEE, 2018, pp. 1–6.
- [23] L. Aversano, D. Guardabascio, and M. Tortorella, "Evaluating the quality of the documentation of open source software," in *International Conference on Evaluation of Novel Approaches to Software Engineering*, vol. 2. SciTePress, 2017, pp. 308–313.
- [24] M. Richter, "Kriterien der Benutzerfreundlichkeit," *Philosophische Fakultät der Universität Zürich*, 1997.
- [25] "IEEE Standard for Adoption of ISO/IEC 26514:2008 Systems and Software Engineering—Requirements for Designers and Developers of User Documentation," *IEEE Std 26514-2010*, pp. 1–72, 2011.
- [26] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. Mahrin, "An adaptive fuzzy decision matrix model for software requirements prioritization," *Advanced approaches to intelligent information and database systems*, pp. 129–138, 2014.
- [27] K. Mayer, "Evaluierung und Vergleich von Adversary Emulation Tools," Master's thesis, University of Applied Sciences Technikum Vienna, 2024.

A. Questionnaire

This section contains the list of questions used to evaluate the selected adversary emulation tools in an offline setting. In the following, categories are written in bold, sub-categories are indicated by the number before the period, and question identifiers are indicated by the number after the period. We refer to Fig. 5 for an enumeration of all sub-categories.

Installation & Configuration (IC) > IC-1.1: Which operating systems are supported? > IC-1.2: Are changes to security settings or permissions required? > IC-2.1: Is third party software required? > IC-2.2: Are there other requirements specific to operating systems? > IC-3.1: Is it necessary to pre-configure the tool? > IC-3.2: Are special network configurations required?

Community & Support (CS) > CS-1.1: How many forks does the project have? > CS-1.2: How many contributors does the project have? > CS-1.3: What is the age of the project (in days)? > CS-1.4: What is the size of the project (in kilobytes)? > CS-2.1: How many programming languages does the project involve? > CS-2.2: How many watchers does the project have?

Documentation (D) > D-1.1: Is there any kind of documentation for the tool? > D-1.2: Is it sufficient for setting up all of the tool's components? > D-1.3: Is it sufficient for launching built-in attacks? > D-1.4: Is it sufficient for creating new custom attack procedures? > D-1.5: Is it sufficient for creating new attack chains? > D-1.6: Does the documentation describe how to interpret results of executed attack procedures? > D-2.1: Is the available documentation updated with each release or other regular intervals? > D-3.1: What is the average length of chapters? > D-3.2: What is the average tree depth of chapters? > D-4.1: What is the level of readability of the documentation? > D-4.2: What is the medium length of sentences? > D-4.3: How frequent are images or tables? > D-5.1: If figures or tables are present, do they have an index number and caption? > D-5.2: Is the documentation organized in accordance to the IEEE Standard for User Documentation?

Usability (U) > U-1.1: Are executions of processes consistent when tasks are performed multiple times? > U-1.2: Is it possible to change tasks before the last step without starting the entire task from the beginning? > U-1.3: Are there recurring errors that affect tasks and processes? > U-2.1: Are interfaces and functions self-explanatory? > U-2.2: Do users receive direct feedback for important intermediate steps and actions when using the tool? > U-2.3: Is the appearance of the tool appealing? > U-3.1: Are there any interactive help functions other than static explanations of interfaces and functions? > U-3.2: Are subsequent steps highlighted after completing a task? > U-3.3: Are there help and information displays for buttons, functions, and interfaces? > U-4.1: Does the tool support the creation of attack templates? > U-5.1: Do all available user interfaces allow attack execution? > U-5.2: Do all available user interfaces allow configuration of procedures? > U-5.3: Do all available user interfaces allow to

stop attack executions while they are still active? > U-5.4: Do all available user interfaces allow to access log data of attack executions? > U-5.5: Do all available user interfaces allow to create or add new custom procedures? > U-5.6: Do all available user interfaces allow to create or add new attack chains? > U-6.1: Is it possible to customize the user interface? > U-7.1: Is the average number of clicks or interactions appropriate for common tasks? > U-7.2: Can tasks be solved by several approaches?

Features & Capabilities (FC) > FC-1.1: Does the tool use agents? > FC-2.1: Are firewalls interrupting the workflow of the tool? > FC-2.2: Are firewalls blocking the connection between the command-and-control server and the agent? > FC-2.3: Is real-time antivirus interrupting remote access on the target system? > FC-2.4: Is real-time antivirus interrupting the workflow of the tool? > FC-2.5: Is real-time antivirus interrupting the functionality of the agent? > FC-2.6: Is real-time antivirus deleting the script or interrupting its functionality? > FC-2.7: Is real-time antivirus interrupting functionality of third party tools? > FC-3.1: Does the tool support cleanup of the target system after attack? > FC-3.2: Does cleanup occur immediately after the relevant attack procedure? > FC-3.3: Is cleanup of the relevant procedures only possible at the end of an attack chain? > FC-4.1: Does the tool have logging capabilities? > FC-4.2: Is every executed procedure logged during an attack? > FC-4.3: How is the result of attacks presented? > FC-5.1: Does the tool support repeated execution of the same attack procedure but with different parameters? > FC-5.2: Are custom attack procedures created and handled in the same way as predefined procedures? > FC-5.3: Is it possible to resume incomplete attack procedures or actions after restarting the app? > FC-6.1: Is parallel execution of attack procedures supported? > FC-6.2: Is it possible to automatically execute several attack procedures one after another? > FC-7.1: Are the instructions provided within the tool sufficient so that even a novice is able to execute built-in attack procedures? > FC-7.2: Are predefined attack chains that consist of several attack procedures available in the tool? > FC-7.3: What range of tactics and techniques from MITRE ATT&CK are covered by predefined attack procedures? > FC-7.4: Is the tool designed to support blue teaming? > FC-8.1: Is it possible to reconfigure predefined attack procedures? > FC-8.2: Is it possible to create new custom attack chains? > FC-8.3: Is it possible to create new custom attack procedures through any of the interfaces? > FC-8.4: Is it possible to add new custom attack procedures as scripts? > FC-9.1: Is it possible to stop and reconfigure an ongoing attack at any point in time? > FC-10.1: Are special privileges required for the agents deployed on the target systems? > FC-10.2: Are special privileges required for any of the scripts? > FC-10.3: Are special privileges required for any of the involved third party tools? > FC-11.1: Does the tool execute scripts on the endpoints? > FC-11.2: Does the tool support attack scripting? > FC-11.3: Are attack procedures implemented using scripts?