Are we there yet? An Industrial Viewpoint on Provenance-based Endpoint Detection and Response Tools

Feng Dong*
Huazhong University of Science and Technology
dongfeng@hust.edu.cn

Shaofei Li†
Peking University
lishaofei@pku.edu.cn

Peng Jiang†
Peking University
pengjiang_pku2020@stu.pku.edu.cn

Ding Li†‡
Peking University
ding_li@pku.edu.cn

Haoyu Wang*‡
Huazhong University of Science and Technology
haoyuwang@hust.edu.cn

Liangyi Huang
Arizona State University
lhuan139@asu.edu

Xusheng Xiao‡
Arizona State University
Xusheng.xiao@asu.edu

Jiedong Chen
Sangfor Technologies Inc.
chenjiedong@sangfor.com.cn

Xiapu Luo
The Hong Kong Polytechnic University
csxluo@comp.polyu.edu.hk

Yao Guo†
Peking University
yaoguo@pku.edu.cn

Xiangqun Chen†
Peking University
cherry@sei.pku.edu.cn

ABSTRACT

Provenance-Based Endpoint Detection and Response (P-EDR) systems are deemed crucial for future Advanced Persistent Threats (APT) defenses. Despite the fact that numerous new techniques to improve P-EDR systems have been proposed in academia, it is still unclear whether the industry will adopt P-EDR systems and what improvements the industry desires for P-EDR systems. To this end, we conduct the first set of systematic studies on the effectiveness and the limitations of P-EDR systems. Our study consists of four components: a one-to-one interview, an online questionnaire study, a survey of the relevant literature, and a systematic measurement study. Our research indicates that all industry experts consider P-EDR systems to be more effective than conventional Endpoint Detection and Response (EDR) systems. However, industry experts are concerned about the operating cost of P-EDR systems. In addition, our research reveals three significant gaps between academia and industry: (1) overlooking client-side overhead; (2) imbalanced alarm triage cost and interpretation cost; and (3) excessive server-side memory consumption. This paper's findings provide objective data on the effectiveness of P-EDR systems and how much improvements are needed to adopt P-EDR systems in industry.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**.

KEYWORDS

Provenance-Based EDR, APT, systematic study, gaps

ACM Reference Format:
Feng Dong, Shaofei Li, Peng Jiang, Ding Li, Haoyu Wang, Liangyi Huang, Xusheng Xiao, Jiedong Chen, Xiapu Luo, Yao Guo, and Xiangqun Chen. 2023. Are we there yet? An Industrial Viewpoint on Provenance-based Endpoint Detection and Response Tools. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26–30, 2023, Copenhagen, Denmark.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3576915.3616580

1 INTRODUCTION

P-EDR is a rising next-generation system for APT attack defending [19, 27, 32, 33, 51, 54, 70]. Compared with conventional EDR systems, P-EDR systems introduce provenance graph, a data structure that models dependencies between system activities, so that they can correlate multiple alarms, leading to higher detection accuracy and better interpretability [30]. As such, we have witnessed a rapid growth of P-EDR research recently from security/system top conferences and industry adoption of P-EDR in commercial products. According to a recent study [35], there are over 50 P-EDR related papers published in the most prestigious security and systems conferences in recent five years. Substantial research efforts have been

*Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology.
†Key Laboratory of High-Confidence Software Technologies (MOE), School of Computer Science, Peking University.
‡Co-corresponding authors.

CCS '23, November 26–30, 2023, Copenhagen, Denmark

ACM ISBN 979-8-4007-0050-7/23/11…$15.00
https://doi.org/10.1145/3576915.3616580

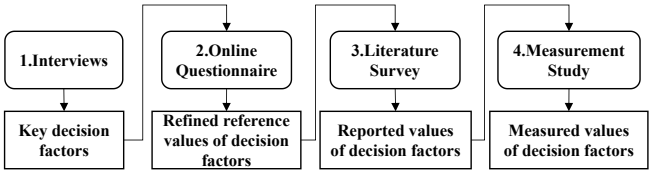

Figure 1: The overview of workflow of our study.

put forth to improve P-EDR systems in terms of system optimizations [32, 57, 67, 76], detection algorithms [27, 30, 53, 54, 70, 78], and broader security applications [69].

While these works have shown promising early results based on evaluations in the academic setting, it is however still unclear whether the industry values the potential of P-EDR systems and would like to adopt some of these works [35]. Moreover, if the industry has not adopted P-EDR systems yet due to various limitations, how these systems can be improved remains unknown. Knowing the answers to these questions is particularly important, as it can guide future research efforts to focus on the most critical directions based on the industry feedback. Specifically, there are three key research questions that need to be addressed:

- **RQ1**: How does the industry compare the effectiveness of P-EDR and EDR? This RQ can help us understand whether the research values of P-EDR systems have been recognized by the industry.
- **RQ2**: What are the bottlenecks for the industry to adopt EDR systems? It is natural that fundamental research takes years before it can be deployed for practical use. This RQ can help us focus the efforts in addressing the major bottlenecks and reduce the turnaround time for P-EDR systems to be put into practice.
- **RQ3**: How well can existing P-EDR systems proposed in academia meet the expectations of the industry? This RQ can help us understand the gaps between the techniques developed in academia and the expectations of the industry.

To this end, we conduct the first set of systematic studies to understand what are the industry's expectations about P-EDR systems and how to close the gaps in adopting P-EDR systems. More specifically, as shown in Figure 1, our study consists of four parts:

- *Interviews*: we first conducted one-to-one interviews to seek feedback on the effectiveness of P-EDR systems and identify their key decision factors in adopting P-EDR. We successfully recruited ten experienced technical managers of security teams from top IT companies to join our interviews (Section 3). These companies include both vendors and consumers of EDR/P-EDR products.
- *Online Questionnaire*: based on the key decision factors found in the interviews, we further designed a structured online questionnaire to get feedback from a broader scope of security engineers for refining the reference values of the key decision factors. Our questionnaire received responses from 48 security engineers in a variety of companies (Section 4).
- *Literature Survey*: based on the identified key decision factors, we surveyed the P-EDR systems described in recent publications and evaluated whether they can satisfy these decision factors (Section 5). Our study revealed that none of the existing systems provide evaluation results for all the key decision factors.
- *Measurement Study*: as many existing systems lack evaluation results for the key decision factors, we further conducted a measurement study on representative P-EDR systems using real industry datasets to measure whether these systems can satisfy these factors and identify the gap (Section 6).

We perform an in-depth analysis of the study results and summarize the findings to answer the three research questions:

- **RQ1**: All the interviewed managers acknowledged that P-EDR systems are superior than conventional EDR systems due to better interpretability. Experienced security analysts can easily understand the provenance data even if it contains only low-level system audit events. Surprisingly, while it is natural that fundamental research takes years for it to be deployed in practice, there are already some security teams (2 out of the 10 interviewed teams) that have adopted P-EDR systems. Furthermore, they have even started to provide training sessions for P-EDR systems. These results show that EDR systems have the potential to replace the EDR systems and become the dominating security defense systems for advanced cyber attacks.
- **RQ2**: Most managers considered the operating cost of P-EDR systems, including the computing cost on both the client-side and server-side and the labor cost on alarm triage and attack investigation, as the primary bottleneck in adopting P-EDR systems, even though intuitively we may generally consider detection accuracy as the most important factor. In fact, most security teams have experiences working with EDR systems that produce a high number of false positives, and P-EDR systems generally have higher detection accuracy, and thus they found no problems in using P-EDR. However, most security teams cannot afford the operating cost of existing P-EDR. For example, provenance data collectors such as Auditd [62] can add at most 821% more runtime overhead to applications running on the client side, and some P-EDR systems require more than 200MB memory to process the data for a protected host, which is 10 times more than the industry expectation (20MB/host). These results show that future research efforts should focus on optimizing the operating cost of P-EDR systems on both the client-side and the server-side.
- **RQ3**: By performing a deeper analysis of our study results, we identify three important gaps between the P-EDR techniques proposed by the academia and the expectations of the industry:
  (1) **Overlooking Client-Side Overhead**: 19 of the 20 surveyed P-EDR systems rely on third-party provenance data collectors such as Sysdig [16] and neglect the client-side overhead.
  (2) **Imbalance between Alarm Triage Cost and Interpretation Cost**: some research focuses on optimizing the precision in reducing alarm triage cost, but it introduces significant interpretation cost by producing a large amount of provenance data to inspect. Similarly, some research focuses on optimizing the interpretation cost but overlooking the precision, producing lots of false positives. Few research has considered both of these factors together, which makes most P-EDR systems impractical in industry settings.
  (3) **Excessive Server-Side Memory Consumption**: most P-EDR systems cache system auditing events in the memory, resulting in very high memory consumption. More research efforts are in dire need to optimize memory consumption.

These identified gaps shed light on what important factors are neglected by the academia and how much improvement of P-EDR systems is needed to meet the industry expectations.

In summary, the contributions of this paper are as follows:

- We are *the first to investigate the industry's expectations about P-EDR systems* and *provide guidelines on how to close the gaps in adopting P-EDR systems.*
- We conduct a one-to-one interview with technical managers from top IT companies and follow up with an online questionnaire to obtain industry expectations on P-EDR systems.
- We conduct a measurement study on three representatives P-EDR systems to measure whether existing P-EDR systems meet the industry expectations and how much improvement is needed. We make the dataset and the systems publicly available [3] to enable the reproducible study and facilitate further research on APT detection and investigation.
- We perform in-depth data analysis of the study results to identify the gaps between academic techniques for P-EDR systems and the industry expectations and provide guidelines for future research.

2 BACKGROUND

In recent years, research on provenance analysis is emerging in academia, and it has gradually become an effective tool for APT detection. Muhammad [35] describes provenance analysis as the totality of system execution and facilitates causal analysis of system activities by reconstructing the chain of events that lead to an attack as well as the ramifications of the attack. BackTracker [40] identifies files and processes that may affect the detection point and displays the chain of events in a provenance graph, which is the first attempt on provenance-based intrusion detection. Due to provenance auditing can record system activities in detail and is hard to evade, provenance-based APT detection models [27, 29, 32, 33, 51, 54, 70] have emerged in the past few years. However, according to our survey, provenance-based techniques have not been widely used in industrial commercial EDR. There are still unacceptable gaps between academic research and industrial deployment.

2.1 Overview of the P-EDR System

The overall process of a P-EDR system is shown in Figure 3. In general, a P-EDR system is the core part of a commercial Security Operation Center (SOC) that monitors the endpoint hosts (e.g., servers, desktops, laptops, e.t.c.) and detects attacks on the hosts. A typical P-EDR system consists of two key components: the client-side component and the server-side component. The client-side component is an agent installed on the monitored hosts that collects provenance data from the hosts. The server-side component is a dedicated server that processes the collected provenance data and detects APT attacks. A typical P-EDR system [27, 29, 30, 54] contains four key steps.

The first step is data collection, which runs on the monitored hosts to collect provenance data and do some preliminary refinement and cleaning. Normally, the collected provenance data contains process, file, register, and network operation logs. Then, the P-EDR system sends the collected data to the server. In commercial systems, the agent may also compress the provenance data before sending it to the server.

Table 1: System events of the provenance analysis

Entity↔Entity	Operation Types
Process→File	read, write, create, chmod, rename
Process↔Process	fork, clone, execve, pipe
Process→IP	sendto, recvfrom, recvmsg, sendmsg

The next three steps are on the server side. The second step is detection, in which the P-EDR system detects APT attacks from the collected provenance data, using manually crafted rules [29, 54] or machine learning algorithms [27, 70]. The third step is the investigation, in which the P-EDR automatically helps security admins correlate related alarms and investigate the root causes of alarms. In the last step, security experts validate the generated alarms and respond to possible attacks.

2.2 Provenance Analysis and Provenance Graph

Compared with conventional EDR systems, the unique advantage of a P-EDR system is that it automatically reconstructs the dependencies between log entries and alarms in the step of investigation [30, 54]. The alarms of conventional EDR systems are isolated. Thus, it is particularly hard for security admins to combine related alarms or recover their root causes. On the flip side, P-EDR systems use provenance graphs to model the data and control dependencies between events in provenance data, automatically linking related alarms and their root causes together, leading to more interpretable detection results. In P-EDR systems [14, 24, 32, 33, 54, 70, 77], a provenance graph is a directed graph constructed from *system auditing events*, where each event represents a system activity. Formally, system auditing events are represented as three-tuples ⟨subject, operation, object⟩. The subject and the object represent *system entities*, and the operation represents an action performed by the subject on the object. The typical values for the three-tuple are shown in Table 1, in which ↔ means the entities on both sides can be subjects or objects. In a provenance graph, the nodes are system entities, and the edges are the actions. The directions of edges represent the dependencies of data or control flow.

2.3 Example Provenance Analysis

In Figure 2, we show an example of the provenance graph for a real APT attack. In this attack, the adversary first hijacks the Windows IIS Web Server "w3wp.exe" through a web shell. Then she uses "csc.exe" to execute a trojan. The adversary also runs the remote tools "GotoHTTP_x64.exe" to modify the registry privilege escalation. Lastly, she leaves a backdoor "Wlw.exe" for intranet blasting with "fscan.exe" and uses "wevtutil" to clear footprints. The orange nodes are alarms generated by the detection system.

In this example, we notice that the provenance graph links multiple alarms based on their dependencies. It also backtracks the entry of the attacks so that security admins can recover the root causes of alarms. Therefore, security analysts consider P-EDR systems more accurate and intuitive for APT attack detection and investigation, leading to the popularity in academia [14, 24, 32, 33, 54, 70, 77].

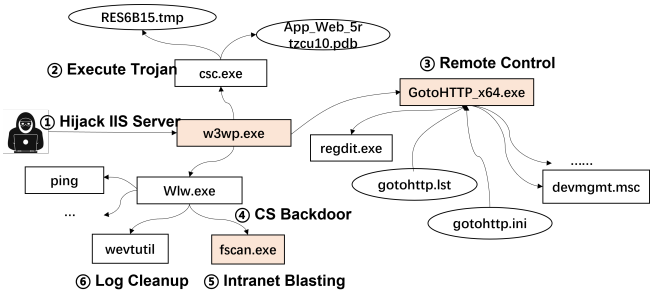

Figure 2: An example of an APT attack that hijacks Windows IIS Web server and leaves a Cobalt Strike (CS) backdoor for lateral movement. Then it uploads the remote control tool for privilege escalation.

2.4 Ethical Consideration of This Work

This work was approved by our institution, and we strictly follow our institution's research data management policy, including data storage, sharing, and disposal. The data collected from the participants in the interviews and questionnaires were carefully processed. In both the interview and the online questionnaire, we acquired the consent of the participants and confirmed that our interviews accurately reflected their own opinions.

3 ONE-TO-ONE INTERVIEWS

To seek feedback on the effectiveness of P-EDR systems and identify the decision factors for the adoption of P-EDR systems in the industry, we conducted one-on-one interviews with experienced technical managers from top IT companies.

3.1 Participant Recruitment

We recruited participants from EDR developers and consumers, who have the first-hand experiences of EDR in the industry. We chose 6 EDR vendors from top-tier endpoint security companies, and 6 consumers from diverse kinds of organizations, including IT, education, transportation, and manufacturing. The consumers of EDR and P-EDR systems include ByteDance (the world-leading social media provider), MeiTuan (one of the leading AI companies in China), Peking University (one of the most famous universities in China), S.F Express (the biggest express company in China), and FiberHome (the famous manufacturer for IoT devices). The vendors of EDR and P-EDR systems are among the top security vendors in China [2] and the world [1], including Tencent Security [9], Trend Micro [11], Sangfor [8], Rising [7], and NSFOCUS [6]. We first found the points of contact (POC) of EDR through the company website, social media, and product technical support list, and these POCs recommended 12 technical managers. Ten managers (average of 10+ years of experience) agreed to participate in our interview.

Participant Background: Our participants are experienced leaders in security. They have, on average, 10.5 years of experience, ranging from 5-21 years. Each of them leads a technical team with 25-30 engineers on average. Our participants are all very familiar with provenance analysis techniques and P-EDR systems. Specifically, $E1$ and $E2$ are already using P-EDR systems in their companies, and $E6$ and $E7$ are the developers of the P-EDR systems. $E3$, $E4$, and $E5$ who are not using P-EDR are familiar with provenance analysis techniques and are considering using these techniques in the future. Lastly, the remaining three ($E8$, $E9$, and $E10$) who are not developing provenance analysis techniques in their current products are also very knowledgeable about the recent progress in academia and may adopt P-EDR when it is necessary. Table 2 shows the detailed background information of the participants.

3.2 Interview Methodology

We interviewed each manager via a 30-min online video conference. All managers chose to participate in our study voluntarily as they expect our research results can better help them develop and use EDR/P-EDR systems. To ensure the objectiveness of our interview, we followed the principles in *Qualitative Interview Design* [50, 68]. Specifically, we explained the purpose of our interview before the interviews and told them how to get in touch with us later if they want to. We designed all the interview questions to be open-ended, and the participants are able to choose their own terms when answering questions. We also designed our questions to avoid words that might influence answers.

Interview Questions: Our interview questions consist of two parts. The first part is the background, where we ask the participants to introduce their technical backgrounds, including organization name, job title, years of experience, team size, and experiences with P-EDR systems. The second part of our interview questions is the opinion session, which includes questions about the participants' opinions on the key decision factors and the limitations of EDR and P-EDR systems. Below are our interview questions:

- Do you think P-EDR systems are more effective than conventional EDR systems?
- What are the limitations of existing EDR/P-EDR products?
- What are the key decision factors when you decided to adopt your current EDR/P-EDR solution? Are these factors must-meet or optional? Can you rank these factors?
- What metrics do you use to measure these key decision factors?

Besides the background session and the opinion session, we also asked the participants several casual questions to help them relax. These questions are not related to our research objective but facilitate the participants to share their true opinions [50].

Data Processing: With the authorization of the interviewees, their identities were anonymized, and their interviews were saved in audio form. We transcribed the audio into text using a popular Speech-to-Text conversion tool. Two authors independently inspected the converted texts and cross-checked the results. We sent the verified texts back to the interviewees for confirmation. The data retention period is one year, and we will get further authorizations from the interviewees when the retention period expires.

3.3 Results

Effectiveness of P-EDR: Table 3 summarizes the answers of our participants regarding the effectiveness of P-EDR. Overall, all of them agree that P-EDR systems are more effective than conventional EDR systems, and four managers have already adopted P-EDR systems, which will be further detailed in Section 7.

Key Decision Factors: We have summarized seven key decision factors mentioned by the participants in Table 4, including Network,

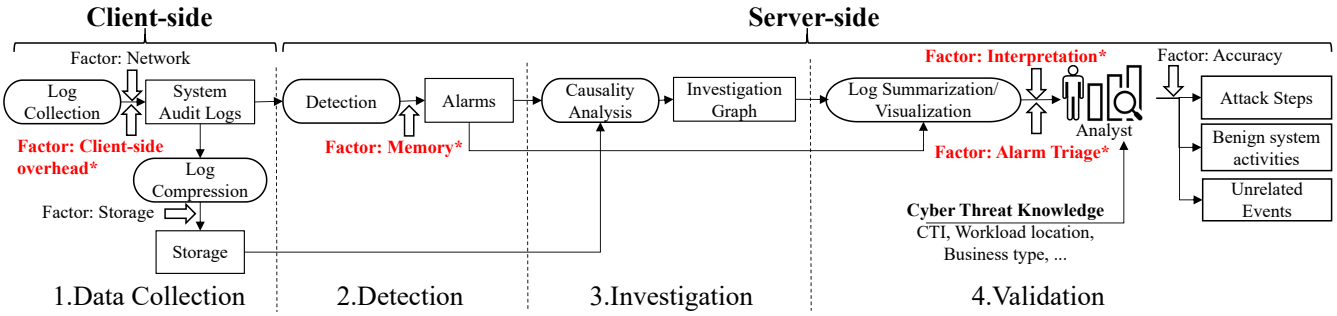

Figure 3: The workflow of using P-EDR to detect attacks

Table 2: Background information of the interview participants

ID	Role	Company Name	Industry Area	Job Title	Years of Exp.	Team Size	Adopt P-EDR
E1	Consumer	ByteDance	Technology	Head of Server Security	6	20~25	Yes
E2		MeiTuan	Technology	Cloud Workload Security Leader	5	20~25	Yes
E3		Peking University	Education	Director of Network Security Office	19	10~15	No
E4		S.F. Express	Transportation	Endpoint Security Manager	10	20~25	No
E5		FiberHome	Manufacturing	Endpoint Security Manager	8	5~10	No
E6	Vendor	Tencent Security	Security	Director of EDR	10	10~15	Yes
E7		Trend Micro	Security	Detection Engine Architect of EDR	9	20~25	Yes
E8		Sangfor	Security	Director of Workload Protection Product	8	65~70	No
E9		Rising	Security	EDR Architect	21	50~55	No
E10		NSFOCUS	Security	EDR Product Manager	9	30~35	No

Table 3: Interview results for P-EDR effectiveness

Answers	Participants
Limitations of EDR/P-EDR	
High Client-Side Overhead	E1, E2, E3, E4, E5, E6, E7, E8, E9, E10
Too Many False Alarms	E1, E2, E4, E5, E6, E7, E8
Incomplete Rule Set	E1, E2, E4, E5, E7, E9, E10
Data Privacy	E3
Effectiveness of P-EDR	
P-EDR Already Deployed	E1, E2, E6, E7
P-EDR Better Than EDR	E1, E2, E3, E4, E5, E6, E7, E8, E9, E10

Storage, Memory, Client-Side Overhead, Interpretation, Alarm Triage, and Accuracy. Particularly, Accuracy represents the *detection effectiveness* of a P-EDR system, while the other factors represent the *operating cost* of a P-EDR system. Thus, we further classified them into three major categories: "Computing Cost" (Network, Storage, Memory, Client-Side Overhead), "Labor Cost" (Interpretation and Alarm Triage), and "Performance" (Accuracy). Among these factors, we have identified four must-meet factors (i.e., highlighted by all the participants who mentioned such factors), including Memory, Client-Side Overhead, Interpretation, and Alarm Triage. For clarity's sake, we present these must-meet factors in the workflow of P-EDR (see Fig. 3).

Further, we summarize the expected values for these key decision factors provided by each participant in Table 5. The last row of Table 5 shows the reference ranges for each decision factor. The lower bound and the higher bound of each reference range are the minimum and maximum estimated values provided by our participants, respectively. We next depict each of them.

Computing Cost: For the computing cost, the participants have expressed concerns about the average memory consumption on the server side (`ServerMem`). On average, they expect the server to consume less than 27.6MB of memory per monitored host.

The developers of EDR systems (*E*6, *E*7, *E*8, *E*9 and *E*10) consider network cost and storage cost as optional decision factors. In other words, the developers agreed that the network and storage costs were important, but they were acceptable if a P-EDR system did not meet the requirements. With respect to the metrics, the developers agreed to use the average bandwidth utilization of a P-EDR and the average disk utilization to measure the network cost and storage cost, respectively. The consumers of EDR systems (*E*1, *E*2, *E*4, and *E*5) did not mention the requirements for network and storage costs, except for *E*3. *E*3 requires the storage cost should not exceed 10% of the total disk size of the monitored systems.

For the client-side overhead, the participants believe that two metrics are useful. The first metric is the average runtime overhead on the monitored machine (`RT OH`), and the second one is the average memory consumption on the monitored machine (`ClientMem`). On average, the participants expect a P-EDR system introduces a performance overhead of less than 5.2% and consumes less than 160MB of memory on each monitored host.

Labor Cost: Labor cost lies in the interpretation and alarm triage. For the cost of interpretation, the four participants (*E*1, *E*2, *E*6, and *E*7) who are either using provenance analysis techniques or developing provenance-analysis-based solutions also marked the interpretation cost as a must-meet factor. They agreed to use the average number of nodes of provenance graphs of alarms as the metric for the interpretation cost. The reason is that P-EDR systems generate a provenance graph for each alarm to reveal its context. Thus, the size of the provenance graphs determines the workload

Table 4: Definitions of the Seven Key Factors

Factor	Description
Computing Cost	
CC1: Client-Side Overhead	how much an EDR system slows down the protected hosts
CC2: Network	bandwidth occupied by transmitting system audit logs to the server
CC3: Storage	hard-disk used to store the system logs
CC4: Memory	server memory size required to analyze the collected logs
Labor Cost	
LC1: Alarm Triage	man-hour required to detect false alarms
LC2: Interpretation	man-hour required to interpret attack results
Performance	
Accuracy	attack detection accuracy

for the security team to interpret the alarms. Particularly, the participants expect the number of nodes in provenance graphs to be between 10 to 100.

For alarm triage, the participants agreed to use the average number of alarms per monitored host per day to measure the triage cost. Even though precision is directly related to the triage cost, the average number of alarms per monitored host per day is more intuitive for cost estimation since it is positively correlated with the number of alarms. 8 out of the 10 participants mentioned that their teams or customers have a fixed number of analysts to investigate the alarms, and thus they can only process a limited number of alarms per day. The expected average number of alarms per host per day ranges from 0.001 to 0.1.

Performance: Only two participants ($E2$ and $E3$) considered accuracy as one of the decision factors for choosing EDR/P-EDR systems. Others argue that while accuracy is important, accuracy-related issues can be resolved by upgrading security rules or models within a reasonable time. In terms of the metrics, both $E2$ and $E3$ agreed to use precision to measure the accuracy. The reason is that, in practice, the recall and other metrics are difficult to evaluate due to the lack of ground truth. The expected value for precision ranges from 0.85 to 0.9. Note that the participants acknowledged the importance of precision, but they preferred to use the average number of alarms per monitored host per day to evaluate the performance of a P-EDR system.

4 ONLINE QUESTIONNAIRE

The interview participants helped us identify the key decision factors in adopting P-EDR systems, and provided the reference ranges for the metrics used in these decision factors. However, these reference ranges were too coarse-grained and some participants did not provide their reference ranges for certain metrics. To eliminate research bias, we further designed an online questionnaire and recruited a broader scope of security engineers from a variety of companies to participate in our online questionnaire, and obtain more accurate reference values of these metrics.

4.1 Participant Recruitment

To recruit more professionals who have experiences with EDR systems, we disseminated our recruitment in two ways. First, we asked the interviewees in Table 2 to help disseminate our recruitment information to their colleagues and security engineers who have experiences with P-EDR and EDR systems. Second, we searched on business and employment-focused social media platforms such as LinkedIn [4] and MaiMai [5] (the Chinese LinkedIn) to actively contact the people who are working on endpoint security. Specifically, we selected the keywords, "Network Security", "Endpoint Detection and Response", and "Endpoint Security", to narrow down the search scope to get the contacts of people that are knowledgeable with EDR. We phoned them first to introduce our purpose and know more about their backgrounds. For the qualified candidates, we invited them to participate in our online questionnaire if they were willing to get involved.

Participant Backgrounds: We invited 100+ participants in total, and 48 completed the questionnaires. The background information of the participants has been made available on our website [3]. The participants of our questionnaire come from companies in different industrial sectors, including government, IT technology, the security industry, financial services, manufacturing, etc. They all have experiences in enterprise security and have 4.4 years of the APT combating experience on average.

4.2 Questionnaire Survey Methodology

We sent the online questionnaire link to the participants. Same as the participants of our interviews, we offered the participants of our online questionnaire to use our survey results for improving their uses of EDR/P-EDR systems. We controlled the length of the questionnaire within the range that the respondents can complete the questionnaire within 10 minutes to ensure a high response rate [10]. After we received the 48 responses, we further conducted attention-check to remove low-quality responses. Specifically, we computed the average answering time and the average percentage of unknown answers in a response. We then rejected a response if it was an outlier in terms of the answering time or the percentage of unknown answers.

Questionnaire Questions: We design the questionnaire based on the results from the interview. In the interview, we have identified four must-meet factors: Memory, Client-Side Overhead, Interpretation, and Triage. Thus, in the online questionnaire, we have four questions to determine the fine-grained reference values for these four must-meet factors, respectively. For each question, we divide the reference range obtained in the interviews into five equal-sized sub-ranges and use five options to represent these sub-ranges. In this way, we can obtain more fine-grained results for each decision factor. Note that we have an option "I don't know" to allow the participants to omit the questions that they are not confident about.

We then followed the principles in *How to Design and Frame a Questionnaire* [25] to curate the questionnaire. Specifically, we sent our questionnaire to the interview participants and asked them to confirm that our question design follows their interview answers and that descriptions can be easily understood. The detailed questionnaire can be accessed at our website [3].

Table 5: Interview results for key decision factors. Each cell for a factor shows a rank of importance provided by the corresponding participant followed by a list of metric values to evaluate the factor. Symbol * indicates the must-meet factor.

	Computing Cost				Labor Cost		Performance
ID	Network	Storage	Memory*	Client-Side Overhead*	Interpretation Cost*	Alarm Triage Cost*	Accuracy
E1	None	None	3, ServerMem*: 30MB/host	2, ClientMem*: 100MB/host, RT OH*:1%	4, Number of nodes*: 100	1, Alarms*: 0.001/day/host	None
E2	None	None	3, ServerMem*: 50MB/host	1, ClientMem*: 150MB/host, RT OH*:5%	4, Number of nodes*: 10	2, Alarms*: 0.001/day/host	5, Precision, > 0.85
E3	None	3, Disk: 60MB/day/host	2, ServerMem*: 30MB/host,	1, ClientMem*: 100MB/host, RT OH*:5%	None	None	5, Precision, > 0.9
E4	None	None	3, ServerMem*: 50MB/host,	1, ClientMem*: 200MB/host, RT OH*:10%	None	2, Alarms*: 0.004/day/host	None
E5	None	None	3, ServerMem*: 30MB/host,	1, ClientMem*: 100MB/host, RT OH*:5%	None	2, Alarms*: 0.02/day/host	None
E6	5, Net: 100MB/day/host	6, Disk: 15MB/day/host	3, ServerMem*: 30MB/host,	1, ClientMem*: 200MB/host, RT OH*:1%	4, Number of nodes*: 100	2, Alarms*: 0.1/day/host	None
E7	5, Net: 10MB/day/host	6, Disk: 70MB/day/host	3, ServerMem*: 20MB/host,	1, ClientMem*: 50MB/host, RT OH*:5%	4, Number of nodes*: 100	2, Alarms*: 0.1/day/host	None
E8	5, Net: 42MB/day/host	4, Disk: 100MB/day/host	3, ServerMem*: 26MB/host,	2, ClientMem*: 250MB/host, RT OH*:5%	None	1, Alarms*: 0.05/day/host	None
E9	4, Net: 1MB/day/host	3, Disk: 15MB/day/host	2, ServerMem*: 10MB/host,	1, ClientMem*: 150MB/host, RT OH*:10%	None	None	None
E10	4, Net: 100MB/day/host	5, Disk: 35MB/day/host	3, ServerMem*: 30MB/host,	1, ClientMem*: 100MB/host, RT OH*:5%	None	2, Alarms*: 0.1/day/host	None
Reference Range	1~100MB /day/host	15~100MB day/host	10~50MB/host	50~250MB/host, 1~10%	10~100	0.001~0.1 /day/host	> 0.85

Table 6: Summarized results of online questionnaire

Must-meet Factors	Summarized Result
Memory	< 20 MB/host
Client-side Overhead (RT OH)	< 3 %
Client-side Overhead (ClientMem)	< 100 MB/host
Interpretation	< 50 nodes
Alarm Triage	< 0.1 alarms/day/host

Data Processing: We use a SaaS-based questionnaire platform for managing the questionnaire data. Two authors independently computed questionnaire data statistics and cross-checked the results. The questionnaire data retention is one year, same as that of the interview data.

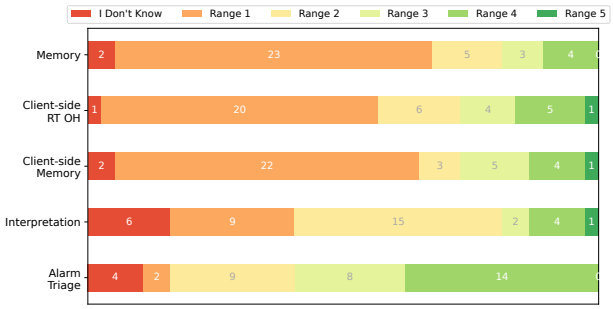

Figure 4: Metric Results of Our Questionnaire Study.

4.3 Results

In total, we received 48 questionnaire responses. The average answering time was 7 minutes) and the average percentage of unknown answers was 14%. We rejected 11 responses as they were outliers in terms of the answering time (<100 seconds) or the percentage of unknown answers (>50% of the answers). Thus, we had 37 valid responses, and the distribution of the answers for each option is shown in Figure 4. For each key factor, we chose the option selected by the largest number of participants as the reference value. We summarize the results of the reference values in Table 6. These results are later used to guide our literature survey and measurement study.

5 LITERATURE SURVEY

The research objective of our literature survey is to investigate the P-EDR systems described in recent publications and check whether they can satisfy the four must-meet factors (Client-Side Overhead, Memory, Interpretation, and Alert Triage).

5.1 Methodology

We systematically inspected all the provenance analysis papers published in top conferences and journals during 2017-2022, including IEEE S&P, USENIX Security, CCS, NDSS, ACSAC, TDSC, and TIFS. We carefully read these papers to classify whether they are in our scope. Finally, we selected 20 papers on P-EDR systems and classified their approaches into rule-based approaches (5), anomaly-based approaches (7), and investigation approaches (8).

For these 20 approaches, we investigate whether they have been evaluated against the seven decision factors. Specifically, two authors independently inspected the evaluation results of these 20 papers on the four must-meet factors and cross-checked the results. Table 7 shows the reported values for each paper. For the systems that were evaluated on different datasets, we calculate the average values on different datasets listed in their papers by default. For Accuracy, we cannot simply use their average reported values, and

thus we summarized the range of the reported values. We use "–" to indicate an approach was not evaluated against a decision factor.

5.2 Surveyed Papers

We can roughly divide existing approaches into two categories: *Rule-based Detection* and *Anomaly-based Detection* [35].

- Rule-based detection approaches leverage prior expert knowledge and experience of attacks to design policies for event matching and behavior extraction. Tag propagation and rule matching are the two most commonly used methods. SLEUTH [32] is the first provenance-based tag policy framework that assigns trustworthiness and confidentiality tags to system entities and propagates on the provenance graph. MORSE [33] is designed based on SLEUTH with refined policies to reduce the amount of false positive alarms. HOLMES [54] and RapSheet [29] leverage the MITRE ATT&CK knowledge-base to configure their rules, mapping low-level events to high-level Tactics, Techniques, and Procedures (TTPs), High-level Scenario Graph (HSG) and Tactical Provenance Graphs (TPG) for attack detection and investigation. Pagoda [72] combines the abnormality of a single path and the entire provenance graph.
- Anomaly-based detection systems are diverse in strategies. Overall, they always learn normal behaviors from historical data and treat deviations from normal behavior as malicious. Both StreamSpot [51] and UNICORN [27] extract the provenance graph into sketches, a vector, as features for clustering and label the outliers as anomalies. UNICORN chooses StreamSpot as the baseline and uses the public dataset collected by StreamSpot, achieving better performance. ProvDetector [70] leverages probability density-based Local Outlier Factors to detect stealth malware paths, embedded into fixed length vectors using graph embedding methods. ZePro [66] uses Bayesian Networks for zero-day attack path identification, and P-Gaussian [73] uses Gaussian Distribution for sequence similarity detection. Poirot needs to manually design Query Graph (QG), generated from Cyber Threat Intelligence (CTI) report with pre-known expert knowledge. SHADEWATCHER [78] extracts the interaction from the provenance graph and constructs a recommendation model for learning to classify system entity interactions into normal and adversarial.

5.3 Results

We next report our analysis of these 20 approaches for each of the must-meet factors.

Client-Side Overhead. We found that only one paper (RTAG) provided evaluations against the client-side overhead. RTAG is an improvement on RAIN, that implemented the system logging logic with comprehensive semantics to record whole-system activities to enable cross-host attack investigation. It mainly measured the runtime overhead and compared it with existing full-system provenance systems. The other 19 systems focus on building detection and investigation algorithms and rely on third-party collectors to monitor provenance data. Thus, *these papers omit the evaluations of the client-side overhead introduced by the collectors.*

Due to the lack of evaluations on provenance collectors in these papers, we further surveyed the available provenance collectors used in industry and academia and listed them in Table 8. Unfortunately, we found that there were no systematic evaluations of the client-overhead introduced by existing provenance collectors. Although there were six collectors that had evaluated the runtime overhead, the three most commonly used collectors, Sysdig, Auditd, and ETW, did not have evaluations of their introduced overheads on other client-side applications. Even worse, none of the existing provenance collectors can satisfy the reference value of runtime overhead (< 3%). Moreover, we found no evaluations of the memory consumption for these collectors. Therefore, we further carried out a measurement study on these three collectors in Section 6.1.

Memory. 8 out of 20 approaches have evaluated the memory consumption on the server side, but none of them directly report the average memory consumption for each monitored machine, and we calculate this value by dividing the overall memory consumption by the number of hosts in their datasets. The results show that most of the reported values are much higher than the reference values we obtained (< 20MB/host), except for RAPID [46]. Particularly, SHADEWATCHER exceeds the expected value by 209 times, which indicates that it can hardly be deployed in the industrial environment. Although RAPID can satisfy the requirement, it needs to utilize third part detection systems for investigation. These results indicate that there is a gap in the methodology of memory consumption evaluation between academia and industry.

Interpretation. The investigation cost is measured by the size of generated provenance graphs. This metric is well-evaluated by existing systems. However, the results vary from 12 nodes to 1.76×10^5 nodes as this factor is highly correlated with the system design. In general, rule-based systems, such as SLEUTH, MORSE, and HOLMES, can generate smaller provenance graphs in alarms than anomaly-based systems, such as StreamSpot and UNICORN. Nevertheless, most of the rule-based and anomaly-based systems have to optimize their interpretation cost by 1 - 4 orders of magnitude in order to meet the industrial requirements (<50 nodes).

Alarm Triage. None of the papers provide evaluations for the cost of alarm triage. In fact, all these papers ignore the factor of triage cost when they evaluate the accuracy of a P-EDR system. Note that in practice, the ratio of attack-related data is very low (less than 0.1%) [30, 48]. Thus, even though they can achieve high accuracy (0.95 averaged from their best-reported values), the triage costs are usually not acceptable in practice.

5.4 Summary

Our literature survey shows that almost all the existing systems do not provide evaluations against the four must-meet factors. Interpretation cost is the only factor that is evaluated by most of the papers, while a large proportion of the surveyed systems cannot meet the requirement from the industry. Similarly, a small set of papers provide evaluations for part of the four factors, and their results show that these systems fail to satisfy the reference values obtained from our studies.

6 MEASUREMENT STUDY

To better understand whether existing P-EDR systems can satisfy the four must-meet factors and how much improvement on these

Table 7: Summarization of literature review. The values are directly reported or averaged the values on different datasets listed in their paper. The empty cells mean the corresponding paper does not evaluate the corresponding decision factor.

Type	Tool Name	Client-side Overhead			Storage (/MB/host/day)	Memory (MB/host)	Alarm Triage (#Alarm/host/day)	Interpretation (#Node, #Edge)	Precision	Recall	Accuracy
		Agent	RT OH(%)	ClientMem (MB)							
Detection	SLEUTH [32]	Auditd	-	-	362.87	81.93	-	(52, -)	-	-	-
	MORSE [33]	Auditd, DTrace	-	-	1266.67	230.4	-	(283, -)	≈ 0	1.00	-
	HOLEMS [54]	Auditd, Dtrace, ETW	-	-	179.23	104.76	-	(-, 400)	1.00	1.00	1.00
	RapSheet [29]	Symantec EDR	-	-	358.00	-	-	(12, 39)	0.26	1.00	0.75 - 0.95
	Pagoda [72]	Karma [18], PASS [55]	-	-	1126.40	-	-	(13315, 10964)	0.92-1.00	1.00	0.75 - 0.95
	StreamSpot [51]	SystemTap [36]	-	-	-	-	-	(8315,173857)	0.50-1.00	-	0.50 - 0.80
	UNICORN [27]	CamFlow [58]	-	-	24917.33	-	-	$(1.76 \times 10^5, 2.82 \times 10^6)$	0.80 - 0.99	0.88 - 1.00	0.84 - 0.99
	ProvDetector [70]	-	-	-	-	-	-	(-, -)	0.96	0.99	-
	ZePro [66]	-	-	-	266.67	57.14	-	(1853, 2249)	-	-	-
	P-Gaussian [73]	-	-	-	864	152.5	-	(1949, 3045)	-	0.66 - 0.94	0.65 - 0.95
	Poirot [53]	Auditd, Dtrace, ETW	-	-	6500.55	122.39	-	(-, -)	1.00	1.00	1.00
	SHADEWATCHER [78]	Auditd	-	-	59112.73	4194.30	-	(-, -)	0.86 - 1.00	0.95 - 1.00	0.98 - 1.00
Investigation	RTAG [38]	RAIN	4.84	-	1536 - 4096	-	-	(164.67, 3200)	-	-	1.00
	MCI [41]	Auditd, Dtrace, ETW	-	-	-	-	-	(34.56, 62.87)	0.92- 1.00	0.95 - 1.00	-
	PrioTracker [47]	Auditd, ETW	-	-	998.64	-	-	(-, 1219)	-	-	-
	NoDoze [30]	Auditd, ETW	-	-	428.90	-	-	(14, 14)	0.50	1.00	0.86
	ATLAS [14]	-	-	-	2286.93	-	-	(-, -)	0.91	0.97	0.99
	DEPCOMM [74]	Sysdig	-	-	-	-	-	(289, -)	-	-	-
	DEPIMPACT [24]	Sysdig	-	-	-	-	-	(-, 234.27)	0.79 - 0.85	1.00	-
	RAPID [46]	Auditd, Dtrace, ETW	-	-	4743.40	30.04	-	(-, -)	-	-	-

Table 8: List of existing provenance data collectors. "RT OH" is the average runtime overhead of applications in their evaluations. "Mem" is the average memory consumption.

	Platform	Owner	Affect	RT OH (%)	Mem (MB)
Sysdig [16]	Linux	Sysdig.Inc	[24, 74]	NA	NA
Auditd [62]	Linux	Linux Foundation	[30, 32, 33, 41, 46, 47, 53, 54, 78]	NA	NA
DTrace [17, 71]	Linux	Sun Microsystems	[33, 41, 46, 53, 54]	3.2	NA
Camflow [58]	Linux	University of Cambridge	[27]	9.7	NA
LTTng [21]	Linux	EfficiOS	NA	NA	NA
ETW [22]	Windows	Microsoft	[30, 41, 46, 47, 53, 54]	NA	NA
KennyLoggings [57]	Linux	UIUC	NA	4.6	NA
Hardlog [12]	Linux	Microsoft	NA	6.3	NA
Quicklog [31]	Linux	Florida State University	NA	5.3	NA
SystemTap [23, 36]	Linux	Linux Foundation	[51]	NA	NA
RAIN [37]	Linux	Georgia Institute of Technology	[37, 38]	NA	NA
Karma [18, 65]	Linux	Indiana University	[72]	NA	NA
PASS [55]	Linux	Harvard University	[72]	10.5	NA

factors is needed, we conducted a measurement study on the representative systems described in the surveyed papers and obtained their metric values to compare with the collected reference values. We next describe our measurement study on both the client-side and the server-side.

6.1 Client-Side Measurement Study

We empirically study the Client-Side Overhead factor using three representative collectors. We deployed these collectors to hosts with different hardware configurations and also measured their introduced overheads on seven representative applications.

Representative Collectors. In our measurement study, we chose three most widely used industrial open-source collectors, Sysdig, LTTng, and Auditd, from the collectors listed in Table 8. These three collectors are adopted by the majority of the existing P-EDR systems. They have industrial quality and are actively maintained. We excluded DTrace because it has a similar performance to Sysdig, and it requires significant technical knowledge to utilize and optimize, which may cause potential bias [20]. We did not measure the client-side overhead of ETW due to two reasons. First, we found no way to turn off the kernel module of ETW completely. Second, ETW does not have an official user-space collector, and our study of it could be significantly biased. We also excluded other collectors because they are outdated and lack downstream users.

Representative Applications. We chose seven representative applications used in the surveyed papers, which can be classified into two categories:

- *I/O-intensive applications*: We first chose commonly used applications of C++, including Nginx [63], Redis [49], Postmark [39]. We also chose two applications of other languages, namely Django [42] for Python and http [26] for Golang.
- *CPU-intensive applications*: We chose OpenSSL [61] and 7-ZIP [56].

Experiment Setup. In our measurement study, we followed the minimal workload principle to avoid possible biases introduced by extra provenance data processing. We simply directed the three collectors to dump their collected data into a file in an in-memory file system. Note that this protocol measures the lower bound of the client-side overhead of the provenance collectors as they usually contain more complex processing logic or need to dump data into much slower devices, such as networks and hard disks. Therefore, we expect the real client-side overhead of P-EDR systems should be higher than the values we reported in this paper. We ran experiments on these three tools under four hardware configurations with different numbers of cores and different sizes of memory on both virtual and physical machines, as shown in Table 9.

We ran these applications using their official benchmarks while measuring their performance. Specifically, we used *wrk* [60] with 1,000 concurrent connections to benchmark Nginx. For Redis, we used the *redis-benchmark* configured to send 1,000,000 requests and measure the speed of operation *get*. We used the built-in benchmark with the configuration of manipulating 500 files concurrently and launching 100,000 transactions to evaluate the performance of PostMark. We used the Phoronix Test Suite, one of the most comprehensive benchmark suites of web applications [43, 64], to benchmark Django and http. For OpenSSL, we relied on the default *speed* benchmark configured to utilize all CPU cores and measure the time to compute one rsa4096 signature. For 7-ZIP, we used the

Table 9: Hardware configurations for our measurement study

Physical Machine	C1	C2	C3	C4
	1CPU + 2GB	4CPU + 8GB	16 CPU + 32GB	32 CPU + 64GB
Virtual Machine	**C5**	**C6**	**C7**	**C8**
	1CPU + 2GB	4CPU + 8GB	16 CPU + 32GB	32 CPU + 64GB

built-in benchmark configured to utilize all CPU cores and measure the compression speed in MIPS. We repeated each experiment ten times and reported the average metric values of the benchmarks.

Runtime Overhead. We show the experiment results in Table 10. We notice that for I/O-intensive applications, there are relatively high overheads compared to the case without turning on the collectors. We also notice that as the number of CPU cores increases, the overhead decreases. This is because all the collectors are single-threaded, which can only utilize one CPU core. When the number of CPU cores is small, the collectors will compete for the resources with the applications. For example, for the single-core machines (C1 and C5), the collectors add at most 821% more overhead to Nginx. Particularly, we find that Auditd introduces a significant overhead because it uses Netlink and has heavy processing logic. For CPU-intensive applications, the overheads compared to the case without turning on the collectors are much smaller, which is less than 3% on average across all configurations. To conclude, all the provenance collectors introduce inevitable overhead compared to the case without turning on the collectors since they record and consume the provenance events.

Memory. The memory consumption of the collectors is listed in Table 11. Memory consumption of collectors consists of two parts: user-mode and kernel-mode memory. For Sysdig, LTTng, and Auditd, the user-mode memory consumption is 30M, 15.9M, and 1.9M, respectively, which is independent of hardware configurations and applications. For kernel-mode memory cost, Auditd allocated a fixed size buffer of 64MB by default; Sysdig and LTTng allocated a fixed size buffer for each core, 8MB and 2MB respectively.

6.2 Server-Side Measurement Study

In this section, we empirically study the Memory, Interpretation, and Triage factors using three representative EDR systems.

Representative P-EDRs. We chose ProvDetector [70], UNICORN [27], and HOLMES [54] due to the following reasons. First, these three systems have the highest precision according to Table 7. Since the average number of alarms approximates the precision, we expected these three systems to have the lowest number of alarms per host per day. Second, these three systems cover the two categories of P-EDR systems. HOLMES is one of the state-of-the-art rule-based systems, while UNICORN and ProvDetector are the two leading learning-based systems. We cannot implement MORSE [33], Poirot [53], and RapSheet [29] because they rely on unpublished rules or CTI reports. We fail to implement SHADEWATCHER [78] because it depends on an unpublished recommendation model. We omit SLEUTH [32] and StreamSpot [51] because they are inferior to HOLMES and UNICRON, respectively. We exclude pagoda, ZePro, and P-gaussian because they adopt similar techniques as ProvDetecor, and ProvDetecor is the most recognized approach among them. We also exclude the investigation systems like PriоTracker [47], NoDoze [30], ATLAS [14], and DEPCOMM [74] because they need to work with unpublished third-party attack detection tools. We implemented HOLMES using the detection rules provided in its paper and configured it to achieve the best performance based on our empirical knowledge. We implemented ProvDetector according to the description in its paper and adopted its default configurations. We directly used the published source code of UNICORN and adopted its default parameters but modified its data parser to accept our data format. We conducted our experiments on the following datasets.

Datasets. We used five datasets to evaluate the server-side cost of ProvDetector, UNICORN, and HOLMES. Among the three datasets, DARPA-Cadets, DARPA-Theia, and DARPA-Trace are open datasets from DARPA [59]. Production dataset is the real auditing data collected from a security company AnonymousSec. Simulation dataset is an in-lab dataset we created for attack simulation. We provide more information of these datasets in Table 13.

Particularly, DARPA-Cadets contains three attacks during a 3-day-long period. The attacker exploited the vulnerabilities of an Nginx server and achieved C&C by injecting the payload to an "sshd" process. The attacker repeated the attack 3 times. He failed the first 2 times but succeeded in the last one. DARPA-Theia contains one attack. The attacker first exploited the Firefox backdoor to install executable files to disk. After two days, he used the vulnerability of a browser extension and resumed the prior attack by injecting the file that had been previously dropped to disk into the "sshd" process. DARPA-Trace contains two attacks. The first one is a Firefox backdoor with the DRAKON malware in memory, and the second one is a Pine backdoor with a DRAKON dropper.

The Production dataset was collected by AnonymousSec's EDR deployed in the customers' network, which includes 300+ servers and working machines of employees of 10 real customers from AnonymousSec, including schools, research institutes, factories, and healthcare providers. We monitored the customers for five days. We used the first three days (training period) to train the detection model and used the last two days (test period) for testing.

The Simulation dataset was collected from five hosts: one Ubuntu 20.04 server (U1), two Windows Server 2012 R2 Datacenters (S1,S2), one Windows Server 2019 Datacenter (S3), and one Windows 10 desktop host (D1). We deployed Apache and PostgreSQL on Windows Servers and Nginx and PostgraSQL on Ubuntu 20.04 to simulate servers in the AnonymousSec. We used the Windows 10 desktop to simulate the PCs used by the employees in the AnonymousSec. The collected data had the same format as Sysdig for Linux and ETW for Windows.

Memory. Table 13 shows the memory consumption results. The memory consumed by HOLMES and ProvDetector was positively correlated with the data volume of the provenance graphs, which both exceeded the reference value (<20MB/host) by 1-2 orders of magnitude. For UNICORN, it had a relatively stable memory consumption because it used Parallel Sliding Windows (PSW) algorithm to analyze the whole provenance graph, which was independent of memory constraints. However, it exceeded the reference value by 11.9 times. Therefore, none of these systems meet the requirement for the Memory factor and more memory consumption optimizations are needed for these systems.

Interpretation. Table 14 shows the result for the Interpretation factor. The provenance graphs generated by ProvDetecor can satisfy

Table 10: Application benchmarking results. We measure the processing time per request/transaction seven representative applications. We report the median values across 10 runs. All values are shown as the relative runtime overhead (%).

Application	Collector	C1	C2	C3	C4	C5	C6	C7	C8	Avg
Nginx	Auditd	597.30	101.30	34.60	34.80	821.10	186.30	23.70	10.90	226.25
	Sysdig	70.20	26.10	14.60	15.60	68.10	21.20	9.50	7.20	29.06
	LTTng	24.80	10.70	10.00	11.70	26.30	25.80	7.00	1.40	14.71
Redis	Auditd	457.00	58.10	41.70	50.20	512.00	53.20	46.00	43.20	157.67
	Sysdig	17.90	20.00	17.20	16.20	21.00	16.40	15.60	5.70	16.25
	LTTng	8.30	8.40	10.00	5.10	13.60	6.90	1.40	2.70	7.05
Postmark	Auditd	406.00	81.80	84.30	78.40	658.00	149.40	157.20	116.20	216.41
	Sysdig	88.80	19.20	18.00	22.00	98.80	23.20	16.50	7.50	36.75
	LTTng	10.30	9.40	12.30	18.10	12.90	10.30	10.90	11.60	11.98
Django (Python)	Auditd	2.50	0.70	2.10	2.30	1.20	0.50	1.50	2.10	1.62
	Sysdig	1.00	1.00	0.40	1.10	1.10	1.40	0.10	0.30	0.80
	LTTng	1.70	2.10	1.70	1.00	1.20	0.30	0.80	1.10	1.24
http (Golang)	Auditd	341.00	97.30	31.20	11.30	516.00	91.60	35.30	15.50	142.40
	Sysdig	60.70	13.90	10.60	2.80	76.70	11.90	4.10	2.20	22.86
	LTTng	13.80	6.50	4.20	4.10	13.40	6.20	5.80	4.20	7.28
OpenSSL	Auditd	2.90	1.80	1.20	1.00	6.90	0.10	1.70	0.20	1.98
	Sysdig	0.50	0.80	0.40	0.10	0.50	1.40	0.30	0.10	0.51
	LTTng	2.50	0.50	0.10	0.10	0.20	0.20	1.70	0.60	0.74
7-ZIP	Auditd	17.40	10.90	5.40	3.70	16.90	5.60	2.40	2.00	8.04
	Sysdig	1.50	1.30	1.10	1.10	1.20	1.00	0.80	0.70	1.08
	LTTng	2.40	1.80	0.90	0.80	4.70	2.30	0.10	0.10	1.64

Table 11: Memory consumption of provenance data collectors

Agent	C1/C5	C2/C6	C3/C7	C4/C8
Auditd	65.9M	65.9M	65.9M	65.9M
Sysdig	38M	62M	158M	286M
LTTng	17.9M	23.9M	47.9M	79.9M

Table 12: Overview of the evaluation datasets

Dataset	Host Num	Days	Data Size	Event Num	Event Rate	Event Size
DARPA-Cadets	1	11	14 GB	15 M	16.87 eps	1013 Byte
DARPA-Theia	1	11	7.5 GB	10 M	11.25 eps	810 Byte
DARPA-Trace	1	11	62 GB	72 M	75.76 eps	925 Byte
Simulation	5	12	23 GB	50 M	48.23 eps	483 Byte
Production	300+	5	16.85 GB	17 M	39.35 eps	1064 Byte

Table 13: Average number of graph nodes for the evaluation datasets and memory consumption of three P-EDR systems

Dataset	# of Graph Nodes	Memory (MB/host)		
		HOLMES	ProvDetector	UNICORN
DARPA-Cadets	280W+	5683	10240	274
DARPA-Theia	125W+	3870	6574	242
DARPA-Trace	325W+	9605	-	242
Simulation	3W+	73	195	213
Production	5W+	84	240	219

the reference value (< 50 nodes). HOLMES generates alarms within ten times larger than the reference value. Even worse, UNICORN reports the whole graph as an alarm and cannot pinpoint the concise location of attacks. Thus, it generates too coarse-grained provenance graphs 3 to 4 orders of magnitude larger than the reference value, which is not practical in industry.

Alarm Triage. As shown in Table 15, only UNICORN can roughly satisfy the reference value (<0.1 alarms/host/day). HOLMES and ProvDetector still need to reduce the number of alarms by more than 2 orders of magnitude to meet the reference value. Specifically, since improving precision can reduce the number of alarms per host per day (See Section 3.3), HOLMES and ProvDetector will need to improve their precision significantly.

Table 14: Interpretation cost (# of graph nodes)

Dataset	HOLMES	ProvDetector	UNICORN
DARPA-Cadets	173	15	154730
DARPA-Theia	73	8	522735
DARPA-Trace	450	-	1454033
Simulation	566	7	11587
Production	81	5	17853

Table 15: Average Alarm Number (alarms/host/day)

Dataset	HOLMES	ProvDetector	UNICORN
DARPA-Cadets	21	90	0.3
DARPA-Theia	36.7	90	0.1
DARPA-Trace	13.9	-	0.45
Simulation	2.3	23	0.09
Production	12.1	56.3	0.13

7 FINDINGS OF OUR STUDY

In this section, we summarize the key findings of our study and answer the three research questions. In particular, we address RQ1 and RQ2 based on the results of the interviews, and address RQ3 based on the results of all the four studies.

7.1 RQ 1: Effectiveness of P-EDR

In our interviews, we found that the managers in the industry all agreed that P-EDR was more effective than conventional EDR systems due to better interpretability. They all showed great interest in P-EDR systems and agreed that P-EDR systems had great potential to replace conventional EDR systems.

Replacing EDR Systems. As shown in Table 2, 4 out of the 10 managers have adopted P-EDR systems to replace the conventional EDR systems in their products or environments. For instance, $E1$ said: "We use provenance analysis techniques for attack investigation. The P-EDR takes the alarm event as the starting point and generates a limited provenance graph through causal analysis for manual confirmation. The contextual information contained in the provenance graph greatly improves the efficiency of attack investigation." $E7$, the developer of a P-EDR system, also said: "Our customers are interested in the improvement of attack detection and investigation brought by provenance analysis techniques, so we decide to focus on P-EDR systems."

Even the managers who were not using P-EDR systems showed great interest in P-EDR systems. They had not adopted P-EDR systems yet due to the higher cost. For example, $E8$ said: "We attempted to detect attacks using provenance graphs on a customer with 1,200 hosts. However, 800MB of memory is required to detect the attack for the provenance graph data of only one host, and the experimental server runs out of memory after running only 40 host data. We cannot afford the memory cost. Nevertheless, we still hope to find a feasible method."

Interpretability. The managers agreed that it was straightforward to interpret the results of P-EDR systems. Surprisingly, even the basic provenance graphs that consist of low-level system audit events are easy to interpret for security analysts as long as they are concise. For example, $E2$ says: "An analyst's ability to translate alarm semantics is related to his experience, and most matured analysts seldom encounter this problem. Even inexperienced newbies can understand the provenance graphs by taking a quick training.". On average, a novice analyst can understand most provenance graphs by taking a 7-14 days training session, as mentioned by $E1$ and $E6$. Lastly, the managers all agreed that existing techniques that abstract the basic provenance graphs to more intuitive levels, such as technique and tactic levels [15, 52], can potentially improve the interpretability of provenance graphs.

Answer to RQ 1

The industry acknowledges that P-EDR systems are superior to conventional EDR systems due to better interpretability. Experienced security analysts can easily understand basic provenance graphs that consist of low-level system audit events, and companies have designed training sessions in provenance analysis for training novice analysts.

7.2 RQ 2: Adoption Bottlenecks

According to the results, *the primary bottleneck for the industry to adopt P-EDR systems is the cost instead of the performance.* In fact, only two managers ($E2$ and $E3$) considered detection accuracy as a decision factor, and they still considered it as an optional factor and ranked it after other factors. The major reason is that these managers already have mature processes in working with existing EDR systems that generate lots of false positives, and P-EDR systems generally have better detection accuracy than EDR systems.

Through discussions with these managers, we realized that the decision process of an industrial manager to adopt a P-EDR system, or an EDR in general, was to minimize the potential loss of successful attacks and the cost of running an EDR system. Formally, the managers aim to minimize the $overall\_cost$, where $overall\_cost = loss_e + op\_cost$, $loss_e$ is the expected loss, and op_cost is the operating cost that consists of computing cost and labor cost (Section 3.3). Here, $loss_e$ is considered as a constant because it is not observable in practice. Thus, when a manager was evaluating an EDR system, he first tested whether the EDR system could detect attacks in a testing environment with sufficient accuracy. As long as the detection recall exceeds a certain threshold, the manager replaces $loss_e$ with a constant. Furthermore, almost all existing P-EDR systems can achieve higher recalls than their thresholds, as mentioned by the managers. Therefore, the managers only considered the operating cost as the primary bottleneck of a P-EDR system. This also explains why none of them chose recall as one of decision factors.

Answer to RQ2

The operating cost, which consists of the four-must factors: Memory, Client-Side Overhead, Interpretation, and Alarm Triage, is the primary bottleneck for the industry to adopt an EDR/P-EDR system.

7.3 RQ3: Gaps Between Industry and Academia

According to the results of all the four studies, we find that there are three important gaps between the P-EDR techniques proposed by academia and the expectations of the industry.

Gap 1: Overlooking Client-Side Overhead. Although the industry considers the Client-Side Overhead as one of the most important factors for adopting P-EDR systems, academia often overlooks it. Based on our interviews, 8 out of 10 managers identified the client-side overhead as the most important decision factor. However, all the 20 surveyed papers, except for RTAG, did not evaluate the client-side overhead of their approaches. Worse still, by investigating existing provenance collectors in academia and industry, we found that there were no comprehensive evaluations on the client-side overhead of these collectors, even though the most popular commercial provenance collectors (Auditd, Sysdig, LTTng, and ETW) shown in Table 8. Through our literature review (Section 5) and measurement study (Section 6.2), we found that existing provenance collectors could not satisfy the reference value of runtime overhead (<3%).

Gap 2: Imbalance between Alarm Triage and Interpretation. Alarm triage, and Interpretation are two must-meet factors for the P-EDR systems. Our study shows that none of the existing P-EDR systems meet both of these factors. With deeper investigation, we realize that these P-EDR systems implicitly sacrifice one factor to enhance the other. Consider the three representative systems in Table 15, UNICORN has a satisfying alarm triage cost. However, this comes with the interpretation cost of several orders of magnitudes higher than the other two systems. This is because

UNICORN projects provenance graphs into embedding vectors, which improves detection accuracy, but the projection also prevents UNICORN from pruning irrelevant events from the provenance graphs, leading to huge graphs (millions of nodes). On the contrary, HOLMES and ProveDetector detect anomaly paths in provenance graphs, generating much smaller graphs (low investigation costs) but resulting in much higher false positives (dozens on average).

Gap 3: Excessive Server-Side Memory Consumption. Memory is a must-meet factor for adopting P-EDR systems. But our literature survey shows that academia has not paid attention to server-side memory consumption, and our measurement study shows that existing P-EDR systems cannot satisfy the reference value (<20MB/host). The root cause for such intolerable memory consumption is that these systems cache all the provenance data in the memory, such as HOLMES and ProveDetector. Therefore, these systems cannot scale to monitor large clusters of hosts. For example, ProveDetector failed to conduct detection on the DARPA-Trace dataset due to memory explosion. Unlike these two systems, UNICORN adopts a stream-based processing approach that uses a sliding window to cache only the most recent provenance data. Even so, its memory consumption is still 200MB/host, which is about 10× of the reference value (20MB/host).

Answer to RQ 3 (derived from all four-part studies)

There exist three important gaps (overlooking client-side overhead, the imbalance between alarm triage cost and interpretation cost, and excessive server-side memory consumption) between the academic research and the industry expectations.

8 DISCUSSION

8.1 Study Limitations

The limited number of participants in our one-to-one interview may harm the generalizability of our study. To address this threat, we recruited participants from different top IT companies, including both customers and providers of P-EDR systems. Further, we followed up the interviews with an online questionnaire that expanded the scope of the participants. We also strictly followed the principles in *Qualitative Interview Design* [50, 68] and *How to Design and Frame a Questionnaire* [25] when conducting the interviews and the follow-up questionnaires. Further, inaccurate implementation and inappropriate parameter configurations of the chosen P-EDR system may also harm the validity of our study. To mitigate this threat, we used the original implementations if they were available or strictly followed the paper descriptions to implement and configure the systems (e.g., ProvDetector and HOLMEs). We also share the systems [3] and the evaluation datasets with the community for subsequent reproducible research.

8.2 Implications

Our study findings (Section 7) identify potential areas to improve P-EDR techniques. We summarize the study implications with the focus on filling the important gaps as follows.

Adopting Data Reduction for Gap 1: To date, client-side overhead has received less attention than others and more efforts are desired to optimize the runtime overhead of collectors. Recent studies on provenance data reduction [34, 67, 75] show that there are a large number of repeated and similar logs in the collected logs, which waste a lot of memory on the client side. Thus, a promising approach is to integrate causality-preserving reduction [75] and other data reduction techniques to provenance collectors to greatly reduce the volume of log data. However, existing data reduction techniques are mainly designed to run on the server side, and complex compression algorithms are too expensive to be directly applied to the client's collector. For example, NodeMerge [67] requires 928.61MB of memory, and efficient collectors pursue smaller overhead rather than data compression ratio. Therefore, we can develop a lightweight collection and filtering framework to reduce the collection of irrelevant log data through lightweight computation such as heuristic rules on identifying temp files [44] or deprioritizing chronicle maintenance processes.

Integrating Alarm Filtering for Gap 2: Due to the lack of industry insights, existing work mainly focuses on how to reduce the number of alarms and ignores the size of the alarm graph. In addition, many of the key papers [29, 30, 54] related to alarm filtering mostly adopt a single filtering method such as alarm correlation or alarm ranking, and the filtering effect on large-scale clusters is insufficient. For example, NoDoze [30] is an alarm ranking technique that assigns an anomaly score based on the frequency to combat threat alarm fatigue produced by the rule-based host IDPS. The filtering effectiveness of NoDoze is only around 84%. If it is applied to the production data set of the HOLMES in Section 6.2, there are still 1.94 alarms/host/day, which is far from the industry reference value (< 0.1 alarms/host/day). To reduce the amount of alarms, we can adopt a systematic alarm filtering method, which can integrate alarm aggregation, correlation, and ranking methods, reaching the desirable alarm level. At the same time, for those systems that generate a large-scale alarm graph, we can design an alarm graph clipping algorithm to identify and delete irrelevant nodes and edges in the alarm graph, so as to control the graph size to a reasonable range.

Distributing Server Workload and Archiving Events for Gap 3: The key papers discussed in our study all adopt a centralized architecture, which uploads the full amount of logs to the server, and then builds a complete provenance graph for complex graph clipping and matching calculations to detect attacks. However, building and maintaining provenance graphs require a lot of memory, and yet a large portion of nodes and edges in the provenance graph is irrelevant to actual attacks [29], wasting a lot of memory. Thus, a promising solution is to adopt a distributed architecture to utilize client computing if clients have spared computing capacity to reduce server memory burdens. For example, we can design a lightweight filtering algorithm on the client side to identify suspicious events, and only upload information related to suspected attack events to the server. Unlike the centralized architecture, which needs to reconstruct and maintain provenance graphs and perform complex computations on these graphs, a distributed architecture only processes localized data of suspicious events, and the required memory is greatly reduced. Furthermore, we can design an algorithm to periodically evict the events cached in the memory to the hard disk during attack detection and fetch the associated data from the disk when it is needed during attack investigation.

9 RELATED WORK

Researchers have shown great interest in understanding the challenges and opportunities of P-EDR systems. Han et.al. [28] summarized the opportunities and challenges associated with P-EDR and provide insights based on their research experience in this area. Li et.al. [45] conducted a literature review on existing P-EDRs in academia. The most recent measurement study conducted by Inam et.al. [35] summarizes P-EDR related techniques published in the top-tier system and security conferences and builds taxonomy based on the system auditing pipeline. Alahmadi et.al. [13] carried out a qualitative study of conventional SOC analysts' perspectives on security alarms through an online survey and semi-structured interviews. Yet, none of the existing papers have studied the effectiveness and bottlenecks of P-EDR systems from the perspective of the industry. Note that, the most well-known P-EDR systems are introduced in Section 5.

10 CONCLUSION

In this paper, we conduct the first set of systematic studies on the effectiveness and the bottlenecks of existing P-EDR systems from the industrial perspective. We also conduct a literature survey and a measurement study to identify the gaps between the techniques developed in academia and the expectations of the industry. Our study shows that the industry believes that P-EDR systems are superior to convention EDR systems. However, the industry is also concerned about the operating cost of P-EDR systems. We further identify three gaps between academia and the industry. Particularly, we find the academia (1) overlooks the client-side overhead of P-EDR systems, (2) fails to balance alarm triage and interpretation, and (3) needs to significantly reduce the server-side memory consumption for P-EDR systems. Taken together, we expect these findings to help improve researchers' understanding of the expectations of P-EDR systems from the industry.

ACKNOWLEDGMENTS

This work was partly supported by the National Key R&D Program of China (2021YFB2701000), the Key R&D Program of Hubei Province (2023BAB017, 2023BAB079), the National Natural Science Foundation of China (grants No.62302181, 62172009, and 62072046), Huawei Research Fund, Hong Kong RGC Project (No. PolyU15219319) and HUST-FiberHome Joint Research Center for Network Security. Xusheng Xiao's work is partially supported by the National Science Foundation under the grant CNS-2028748.

REFERENCES

[1] 2022. Top Cybersecurity Companies for 2022. https://www.esecurityplanet.com/products/top-cybersecurity-companies/.
[2] 2022. Top Cybersecurity Companies in China for 2022. https://www.maigoo.com/maigoo/9400wlaq_index.html/.
[3] 2023. An Empirical Study of Provenance-based Endpoint Detection and Response Tools. https://github.com/EmpiricalStudy2023/EDREmpiricalStudy.
[4] 2023. LinkedIn. https://www.linkedin.com/.
[5] 2023. MaiMai. https://maimai.cn/.
[6] 2023. NSFOCUS. https://www.nsfocus.com/.
[7] 2023. Rising. http://www.rising.com.cn/.
[8] 2023. Sangfor. https://www.sangfor.com/.
[9] 2023. Tencent Security. https://s.tencent.com/.
[10] 2023. Top 8 Challenges With Designing Accurate Surveys. https://surveytown.com/top-8-challenges-with-designing-accurate-surveys/.
[11] 2023. Trend Micro. https://www.trendmicro.com/en_hk/business.html.
[12] Adil Ahmad, Sangho Lee, and Marcus Peinado. 2022. HARDLOG: Practical Tamper-Proof System Auditing Using a Novel Audit Device. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 1554–1554.
[13] Bushra A Alahmadi, Louise Axon, and Ivan Martinovic. 2022. 99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms. In *Proceedings of the 31st USENIX Security Symposium, Boston, MA, USA*. 10–12.
[14] Abdulellah Alsaheel, Yuhong Nan, Shiqing Ma, Le Yu, Gregory Walkup, Z. Berkay Celik, Xiangyu Zhang, and Dongyan Xu. 2021. ATLAS: A Sequence-based Learning Approach for Attack Investigation. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 3005–3022.
[15] Pooneh Nikkhah Bahrami, Ali Dehghantanha, Tooska Dargahi, Reza M Parizi, Kim-Kwang Raymond Choo, and Hamid HS Javadi. 2019. Cyber kill chain-based taxonomy of advanced persistent threat actors: Analogy of tactics, techniques, and procedures. *Journal of information processing systems* 15, 4 (2019), 865–889.
[16] Gianluca Borello. 2015. System and application monitoring and troubleshooting with sysdig. (2015).
[17] Bryan M. Cantrill, Michael W. Shapiro, and Adam H. Leventhal. 2004. Dynamic Instrumentation of Production Systems. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference* (Boston, MA) *(ATEC '04)*. USENIX Association, USA, 2.
[18] Bin Cao, Beth Plale, Girish Subramanian, Ed Robertson, and Yogesh Simmhan. 2009. Provenance Information Model of Karma Version 3. *SERVICES 2009 - 5th 2009 World Congress on Services*, 348–351.
[19] Jun Dai, Xiaoyan Sun, and Peng Liu. 2013. Patrol: Revealing Zero-Day Attack Paths through Network-Wide System Object Dependencies. In *Computer Security – ESORICS 2013*, Jason Crampton, Sushil Jajodia, and Keith Mayes (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 536–555.
[20] LORIS DEGIOANNI. 2014. Sysdig vs DTrace vs Strace: A technical discussion. https://sysdig.com/blog/sysdig-vs-dtrace-vs-strace-a-technical-discussion/.
[21] Mathieu Desnoyers and Michel Dagenais. 2008. LTTng: Tracing across execution layers, from the hypervisor to user-space. In *Linux symposium*, Vol. 101.
[22] DOMARS. 2021. Event Tracing for Windows (ETW). https://learn.microsoft.com/en-us/windows-hardware/drivers/devtest/event-tracing-for-windows–etw-.
[23] Frank C Eigler, Vara Prasad, Will Cohen, Hien Nguyen, Martin Hunt, Jim Keniston, and Brad Chen. 2005. Architecture of systemtap: a Linux trace/probe tool. (2005).
[24] Pengcheng Fang, Peng Gao, Changlin Liu, Erman Ayday, Kangkook Jee, Ting Wang, Yanfang Ye, Zhuotao Liu, and Xusheng Xiao. 2022. Back-Propagating System Dependency Impact for Attack Investigation. In *Proceedings of the USENIX Security Symposium*.
[25] Rayees Farooq. 2018. How to design and frame a questionnaire. In *Innovations in measuring and evaluating scientific information*. IGI Global, 50–60.
[26] Athenas Jimenez Gabriela Cervantes. 2022. Go Benchmarks. https://openbenchmarking.org/test/pts/go-benchmark.
[27] Xueyuan Han, Thomas Pasquier, Adam Bates, James Mickens, and Margo Seltzer. 2020. Unicorn: Runtime provenance-based detector for advanced persistent threats. (2020).
[28] Xueyuan Han, Thomas Pasquier, and Margo Seltzer. 2018. Provenance-based intrusion detection: opportunities and challenges. In *10th USENIX Workshop on the Theory and Practice of Provenance (TaPP 2018)*.
[29] Wajih Ul Hassan, Adam Bates, and Daniel Marino. 2020. Tactical provenance analysis for endpoint detection and response systems. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1172–1189.
[30] Wajih Ul Hassan, Shengjian Guo, Ding Li, Zhengzhang Chen, Kangkook Jee, Zhichun Li, and Adam Bates. 2019. Nodoze: Combatting threat alert fatigue with automated provenance triage. In *Network and Distributed Systems Security Symposium*.
[31] Viet Tung Hoang, Cong Wu, and Xin Yuan. 2022. Faster Yet Safer: Logging System Via Fixed-Key Blockcipher. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 2389–2406. https://www.usenix.org/conference/usenixsecurity22/presentation/hoang
[32] Md Nahid Hossain, Sadegh M Milajerdi, Junao Wang, Birhanu Eshete, Rigel Gjomemo, R Sekar, Scott Stoller, and VN Venkatakrishnan. 2017. {SLEUTH}: Real-time attack scenario reconstruction from {COTS} audit data. In *26th USENIX Security Symposium (USENIX Security 17)*. 487–504.
[33] Md Nahid Hossain, Sanaz Sheikhi, and R Sekar. 2020. Combating dependence explosion in forensic analysis using alternative tag propagation semantics. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1139–1155.
[34] Md Nahid Hossain, Jun Ao Wang, R. Sekar, and Scott D. Stoller. 2018. Dependence-Preserving Data Compaction for Scalable Forensic Analysis. In *Proceedings of the USENIX Security Symposium*. 1723–1740.
[35] M. Inam, Y. Chen, A. Goyal, J. Liu, J. Mink, N. Michael, S. Gaur, A. Bates, and W. Ul Hassan. 2023. SoK: History is a Vast Early Warning System: Auditing the Provenance of System Intrusions. In *2023 2023 IEEE Symposium on Security and Privacy (SP) (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 307–325.
[36] Bart Jacob, Paul Larson, B Leitao, and SAMM Da Silva. 2008. SystemTap: instrumenting the Linux kernel for analyzing performance and functional problems. *IBM Redbook* 116 (2008).

[37] Yang Ji, Sangho Lee, Evan Downing, Weiren Wang, Mattia Fazzini, Taesoo Kim, Alessandro Orso, and Wenke Lee. 2017. RAIN: Refinable Attack Investigation with On-Demand Inter-Process Information Flow Tracking *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 377–390.
[38] Yang Ji, Sangho Lee, Mattia Fazzini, Joey Allen, Evan Downing, Taesoo Kim, Alessandro Orso, and Wenke Lee. 2018. Enabling Refinable Cross-Host Attack Investigation with Efficient Data Flow Tagging and Tracking. In *USENIX Security Symposium*.
[39] Jeffrey Katcher. 1997. Postmark: A new file system benchmark. *TR3022* (1997).
[40] Samuel T King and Peter M Chen. 2003. Backtracking intrusions. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*. 223–236.
[41] Yonghwi Kwon, Fei Wang, Weihang Wang, Kyu Hyung Lee, Wen-Chuan Lee, Shiqing Ma, X. Zhang, Dongyan Xu, Somesh Jha, Gabriela F. Cretu-Ciocarlie, Ashish Gehani, and Vinod Yegneswaran. 2018. MCI : Modeling-based Causality Inference in Audit Logging for Attack Investigation. In *Network and Distributed System Security Symposium*.
[42] Michael Larabel. 2022. PyPerformance Benchmark. https://openbenchmarking.org/test/pts/pyperformance.
[43] Michael Larabel and Matthew Tippett. 2011. Phoronix test suite. *Phoronix Media, [Online]. Available: http:// www.phoronix-test-suite.com/. [Accessed October 2022]* (2011).
[44] Kyu Hyung Lee, Xiangyu Zhang, and Dongyan Xu. 2013. LogGC: garbage collecting audit log. In *CCS*.
[45] Zhenyuan Li, Qi Alfred Chen, Runqing Yang, Yan Chen, and Wei Ruan. 2021. Threat detection and investigation with system-level provenance graphs: a survey. *Computers & Security* 106 (2021), 102282.
[46] Yushan Liu, Xiaokui Shu, Yixin Sun, Jiyong Jang, and Prateek Mittal. 2022. RAPID: Real-Time Alert Investigation with Context-Aware Prioritization for Efficient Threat Discovery. In *Proceedings of the 38th Annual Computer Security Applications Conference* (Austin, TX, USA) *(ACSAC '22)*. Association for Computing Machinery, New York, NY, USA, 827–840.
[47] Yushan Liu, Mu Zhang, Ding Li, Kangkook Jee, Zhichun Li, Zhenyu Wu, Junghwan Rhee, and Prateek Mittal. 2018. Towards a Timely Causality Analysis for Enterprise Security.. In *NDSS*.
[48] Yushan Liu, Mu Zhang, Ding Li, Kangkook Jee, Zhichun Li, Zhenyu Wu, Junghwan Rhee, and Prateek Mittal. 2018. Towards a Timely Causality Analysis for Enterprise Security.. In *NDSS*.
[49] Redis Ltd. 2022. Redis 6.0.9. https://redis.io/.
[50] Steve Mann. 2016. The research interview. *Reflective practice and reflexivity in research processes* (2016).
[51] Emaad A. Manzoor, Sadegh Momeni, Venkat N. Venkatakrishnan, and Leman Akoglu. 2016. Fast Memory-efficient Anomaly Detection in Streaming Heterogeneous Graphs. *CoRR* abs/1602.04844 (2016). arXiv:1602.04844 http://arxiv.org/abs/1602.04844
[52] Fernando Maymí, Robert Bixler, Randolph Jones, and Scott Lathrop. 2017. Towards a definition of cyberspace tactics, techniques and procedures. In *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, Boston, MA, USA, 4674–4679.
[53] Sadegh M Milajerdi, Birhanu Eshete, Rigel Gjomemo, and VN Venkatakrishnan. 2019. Poirot: Aligning attack behavior with kernel audit records for cyber threat hunting. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1795–1812.
[54] Sadegh M Milajerdi, Rigel Gjomemo, Birhanu Eshete, R Sekar, and VN Venkatakrishnan. 2019. HOLMES: real-time APT detection through correlation of suspicious information flows. In *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S&P)*. IEEE, 1137–1152.
[55] Kiran-Kumar Muniswamy-Reddy, David A. Holland, Uri Braun, and Margo Seltzer. 2006. Provenance-Aware Storage Systems. In *Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference* (Boston, MA) *(ATEC '06)*. USENIX Association, USA, 4.
[56] p7zip. 2022. p7zip Version 16.02. https://www.7-zip.org/.
[57] Riccardo Paccagnella, Kevin Liao, Dave Tian, and Adam Bates. 2020. Logging to the Danger Zone: Race Condition Attacks and Defenses on System Audit Frameworks. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, USA) *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 1551–1574.
[58] Thomas Pasquier, Xueyuan Han, Mark Goldstein, Thomas Moyer, David Eyers, Margo Seltzer, and Jean Bacon. 2017. Practical whole-system provenance capture. In *Proceedings of the 2017 Symposium on Cloud Computing*. 405–418.
[59] DARPA Transparent Computing Program. 2022. The DARPA Transparent Computing (TC) program Data Release. https://github.com/darpa-i2o/Transparent-Computing.
[60] Open Source Project. 2022. Wrk. https://github.com/wg/wrk.
[61] The OpenSSL Project. 2022. OpenSSL 1.1.1. https://www.openssl.org/.
[62] Redhat. 2017. The Linux audit framework. https://github.com/linux-audit/.
[63] Will Reese. 2008. Nginx: the high-performance web server and reverse proxy. *Linux Journal* 2008, 173 (2008), 2.
[64] Basu A Sharath S. 2013. Performance of Eucalyptus and OpenStack Clouds on FutureGrid. In *International Journal of Computer Applications*.
[65] Yogesh Simmhan, Beth Plale, Dennis Gannon, and Suresh Marru. 2006. Performance Evaluation of the Karma Provenance Framework for Scientific Workflows. In *International Provenance and Annotation Workshop (IPAW)* (international provenance and annotation workshop (ipaw) ed.) *(Lecture Notes in Computer Science (LNCS), Vol. 4145)*. Springer, 222–236. https://www.microsoft.com/en-us/research/publication/performance-evaluation-of-the-karma-provenance-framework-for-scientific-workflows/
[66] Xiaoyan Sun, Jun Dai, Peng Liu, Anoop Singhal, and John Yen. 2018. Using Bayesian Networks for Probabilistic Identification of Zero-Day Attack Paths. *IEEE Transactions on Information Forensics and Security* 13, 10 (2018), 2506–2521.
[67] Yutao Tang, Ding Li, Zhichun Li, Mu Zhang, Kangkook Jee, Xusheng Xiao, Zhenyu Wu, Junghwan Rhee, Fengyuan Xu, and Qun Li. 2018. NodeMerge: Template Based Efficient Data Reduction For Big-Data Causality Analysis. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*.
[68] Daniel W Turner III and Nicole Hagstrom-Schmidt. 2022. Qualitative interview design. *Howdy or Hello? Technical and Professional Communication* (2022).
[69] Benjamin E Ujcich, Samuel Jero, Richard Skowyra, Adam Bates, William H Sanders, and Hamed Okhravi. 2021. Causal Analysis for {Software-Defined} Networking Attacks. In *30th USENIX Security Symposium (USENIX Security 21)*. 3183–3200.
[70] Qi Wang, Wajih Ul Hassan, Ding Li, Kangkook Jee, Xiao Yu, Kexuan Zou, Junghwan Rhee, Zhengzhang Chen, Wei Cheng, Carl A Gunter, et al. 2020. You Are What You Do: Hunting Stealthy Malware via Data Provenance Analysis.. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*.
[71] Xiayang Wang, Fuqian Huang, and Haibo Chen. 2019. DTrace: Fine-grained and efficient data integrity checking with hardware instruction tracing. *Cybersecurity* 2, 1 (2019), 1–15.
[72] Yulai Xie, Dan Feng, Yuchong Hu, Yan Li, Staunton Sample, and Darrell Long. 2020. Pagoda: A Hybrid Approach to Enable Efficient Real-Time Provenance Based Intrusion Detection in Big Data Environments. *IEEE Transactions on Dependable and Secure Computing* 17, 6 (2020), 1283–1296.
[73] Yulai Xie, Yafeng Wu, Dan Feng, and Darrell Long. 2021. P-Gaussian: Provenance-Based Gaussian Distribution for Detecting Intrusion Behavior Variants Using High Efficient and Real Time Memory Databases. *IEEE Transactions on Dependable and Secure Computing* 18, 6 (2021), 2658–2674.
[74] Zhiqiang Xu, Pengcheng Fang, Changlin Liu, Xusheng Xiao, Yu Wen, and Dan Meng. 2022. Graph Summarization on System Audit Logs for Attack Investigation. In *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S & P)*.
[75] Zhang Xu, Zhenyu Wu, Zhichun Li, Kangkook Jee, Junghwan Rhee, Xusheng Xiao, Fengyuan Xu, Haining Wang, and Guofei Jiang. 2016. High Fidelity Data Reduction for Big Data Security Dependency Analyses. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*. 504–516.
[76] Carter Yagemann, Mohammad A. Noureddine, Wajih Ul Hassan, Simon Chung, Adam Bates, and Wenke Lee. 2021. Validating the Integrity of Audit Logs Against Execution Repartitioning Attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, Republic of Korea) *(CCS '21)*. Association for Computing Machinery, New York, NY, USA, 3337–3351.
[77] Jun Zeng, Zheng Leong Chua, Yinfang Chen, Kaihang Ji, Zhenkai Liang, and Jian Mao. 2021. Watson: Abstracting behaviors from audit logs via aggregation of contextual semantics. In *Proceedings of the 28th Annual Network and Distributed System Security Symposium, NDSS*.
[78] Jun Zeng, Xiang Wang, Jiahao Liu, Yinfang Chen, Zhenkai Liang, Tat-Seng Chua, and Zheng Leong Chua. 2022. Shadewatcher: Recommendation-guided cyber threat analysis using system audit records. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 489–506.