Intelligent Threat Detection System

Alon Malach - 207024878, Prasanna Wudali - 850091653

July 2023

1 Introduction

Security operations center (SOC) teams deploy a wide variety of security tools to ensure visibility of the activity taking place on the organization's network [1]. Examples of such tools include web application firewalls (WAF), network firewalls, host security agents, proxy servers, etc. All these tools operate independently and SOC analysts often find it very challenging to derive a holistic view of an organization's security posture, which is a necessity for detecting and responding to multi-stage attacks. Of these tools, the two most important are the security information event management (SIEM) tool as it helps the SOC team perform its daily operations. The main reason for this is that individual security tools often lack the context required to identify a multi-stage attack [2]. Furthermore, the lack of standardization in security-related data representation and exchange and the huge amount of data generated by security tools makes it very difficult for SOC analysts to draw meaningful conclusions quickly, which is a requirement for real-time incident detection and response. SIEM systems are designed to address the above mentioned challenges.

Specifically, a SIEM system is a security-related log management system designed to enable SOC analysts to collect, ingest, store, detect, investigate, and manage security-related events for real-time incident detection and response [3]. These systems use rule based analytics to detect, correlate, and aggregate incident-related events in the data and provide security analysts with a comprehensive view of the organization's security posture. Modern SIEM tools also implement ML-based anomaly detection methods to detect security incidents Security-related logs and network flows from various security products, networking devices, and endpoints across the entire network act as sources that forward logs and data to SIEM. These sources should be chosen carefully so that the SOC team has visibility of the entire network and no blind spots exist, since the SOC team cannot protect devices which they cannot see and analyze.

Even with SIEM systems in place, the performance of SOC teams is limited due to the following:

- a high rate of false positives and alert fatigue [4];
- problems related to correlation and contextualization of the alert [5];

- problems related to consistency in the processes being followed [6]
- the need for SOC analysts to manually perform downstream tasks after an alert has been detected, resulting in high mean-time-to-respond (MTTR) and dwell time [7]; and
- heavy reliance on SOC analysts [6].

The disadvantages of having a significant number of false positives are:

- the needle-in-a-haystack problem,
- the exhaustion of SOC analyst, which results in alert fatigue, and
- the need to invest a significant amount of time and resources.

To tackle the limitations of heuristic based and anomaly based threat detection methods, we designed an intelligent threat detection system that utilizes the combines both these methods and provides improved threat detection.

2 Literature Review

Recent advances in natural language processing (NLP) have enabled the use of language models in the automatic analysis of unstructured data. Various methods utilizing NLP techniques (specifically designed to model security-related logs) to identify anomalies have been proposed. Methods such as DeepLog [8], LogRobust [9] and Atlas [10] use simple LSTM based neural network (NN) models to detect anomalies in the data. Tiresias [11] implemented a similar LSTM model to predict future events on a machine, based on previous events observed.

Methods like the recurrent neural network attention mechanism[12], which uses an attention mechanism to further improve the LSTM based anomaly detection model, have also been proposed. DEEPCASE [4] uses an attention mechanism similar to that proposed by [12] to learn causality based correlation between the events in the data, whereas, ATTACK2VEC [13] uses temporal word embeddings to capture the correlations between the events in a sequence. In LogBert [14], a transformer-encoder architecture is used in order to reconstruct a sequence of events; like autoencoders, the reconstruction error of the masked tokens can be used to calculate an anomaly score.

Meng et al [15] proposed LogAnomaly, a method that relies on sequences of logs to learn behavior over time and text contextual analysis. The authors created a Template2Vec model, which is based on the idea of Word2Vec [16]; Template2Vec was designed to provide more contextual meaning to both synonyms and antonyms. This is useful in the case which there are two logs that have a similar text template but differ by just one word; however that one word may be the most important one. For example, in the template "The service is now %s," the variable word is the most important one. It could be either "up" or "down" and the choice of one or the other would change the meaning of the sentence entirely.

3 Methodology

Our method implements a threat detection system aimed at producing minimal to no false positives. The proposed method consists of two stages - threat detection and alert event aggregation. In the threat detection stage both rule based and ML based anomaly threat detection approaches are employed in order to detect suspicious events in the input data. In the alert event aggregation stage an LSTM based deep learning model is utilized to find meaningful relations between the suspicious events (both true positives and false positives) which serve as input to the model. We refer to the ML model used in this stage as an aggregator. The aggregator identifies the true positive alert events among the suspicious events and provides context for these alert events and helps in building the attack story (see Fig. 1).

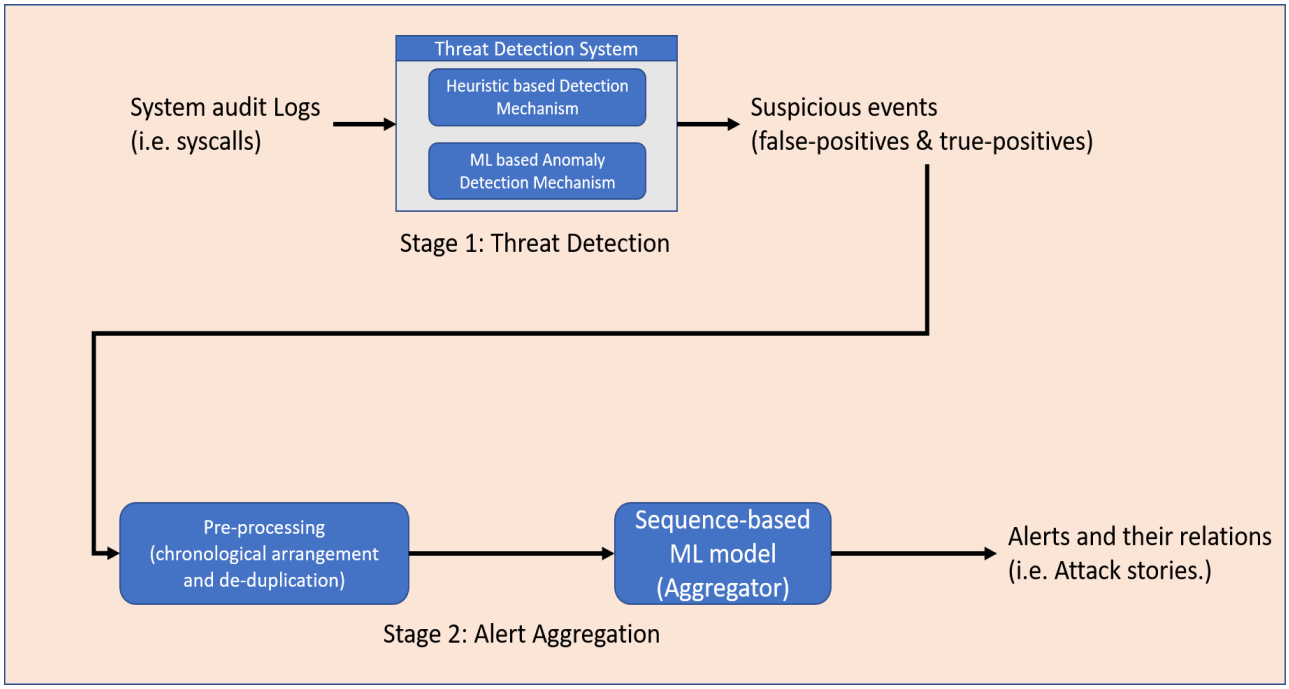

Figure 1: Threat detection model

We adapt KRYSTAL's [17] rule based system to detect alerts in system audit logs (i.e., system calls captured by Windows ETW or Linux auditd). KRYSTAL generates PGs from these audit logs and then detects alerts in them. PGs typically consist of nodes and edges, where nodes represent subjects (i.e., processes) and objects (i.e., subprocesses, files, and sockets), and edges represent the relationships (i.e., events) between them. For alert detection, KRYSTAL implements Sigma rules [18] on nodes and custom rules (also known as provenance rules) on edges in the PGs.

For the ML based anomaly detection approach, we implement an LSTM based NLP model that performs the task of next token prediction[8]. The input to the LSTM model is a sequence of events, and the model tries to predict the next event in the sequence.

To further improve this model we shall be implementing graph based (or path based) anomaly detection approach. In this approach, the input to the model will be graph embedding instead of sequence of events. The model in this approach would predict malicious components of the graph and provides them as alerts to SOC analyasts.

The aggregator model is an LSTM based NLP model that receives chronologically ordered sequences of suspicious events (i.e. event based context) as input and predicts the next event in the sequence (see Fig. 2). The insight behind implementing this model is that a single suspicious event may be insufficient for determining whether a specific activity is malicious [4]. A security analyst frequently needs to analyze multiple events that are causally correlated to identify if they pose a real threat.

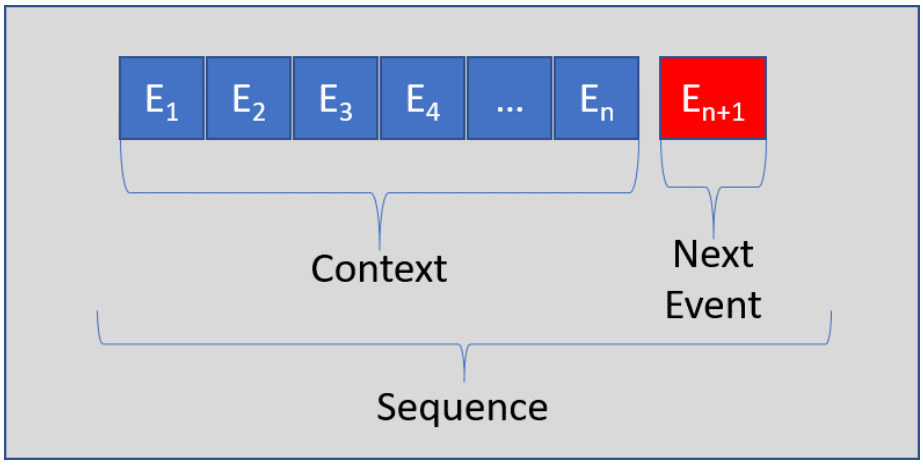

Figure 2: Sequence of events

We train the aggregator as an anomaly detector. The underlying hypothesis guiding this approach is that if the aggregator model is able to predict the next event in the sequence, such sequences might have been experienced before. If the aggregator model generates an anomaly, it means that the sequence is a novel sequence that should be investigated by the SOC analyst. When a new sequence is encountered by the aggregator model, the model is retrained. This enables the aggregator to learn the detected anomalies and improves its ability to identify new anomalies in the future (see Fig. 3).

The aggregator model uses a feature representation that incorporates the nodes and event types (namely subject and object) as its input. This provides event based context for alerts (i.e., correlation between events). However, to improve the efficiency of the threat detection system, event based context alone

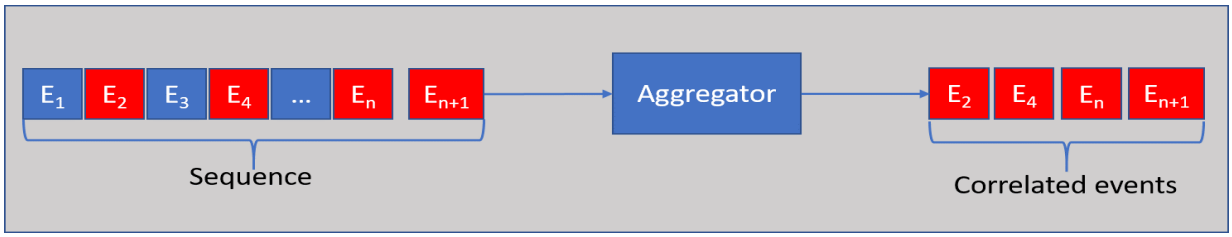

Figure 3: Aggregator model

is insufficient. Additional forms of context that provide more meaning regarding the alerts are:

- Threat intelligence feeds
- user and asset related context
- application related context
- organizational context (i.e., context related to business processes, compliance requirements, policies, and procedures)

4 Implementation

In this section, we describe our proposed experimental setup. We started by implementing the KRYSTAL framework [17] on the DARPA Engagement 3 [19] dataset. KRYSTAL is a modular framework that can perform multiple functionalities, including collect, parse, reduce, and store the events from the dataset in a graph database. KRYSTAL reduces and parses the input data into Provenance Graphs (PG) based on OWL standards [20] and stores the data in the TTL (Turtle) file format. To build the PGs, we loaded the TTL files into Ontotext GraphDB [21] on a Windows host with a 500GB hard disk and 32 GB RAM.

This PG serves as the input to the rule-based threat detection system. KRYSTAL also employs two different types of rule-based threat detection modules: Sigma rule-based threat detection [18] and provenance-based threat detection [22]. The Sigma module defines rules for the detection of malicious nodes, and provenance-based rules are defined for the detection of malicious events that occur between the nodes.

We implemented two different architectures for our aggregator model which is based on LSTM neural network.

Our first model consists of Two LSTM layers with an input size of 41 features and an output size of 128 features. The final layer is a fully connected layer that was implemented with an input size of 128 features and an output size of six features (one feature for each type of event). The aggregator model is trained on previously seen sequences and identifies unseen sequences as anomalies.

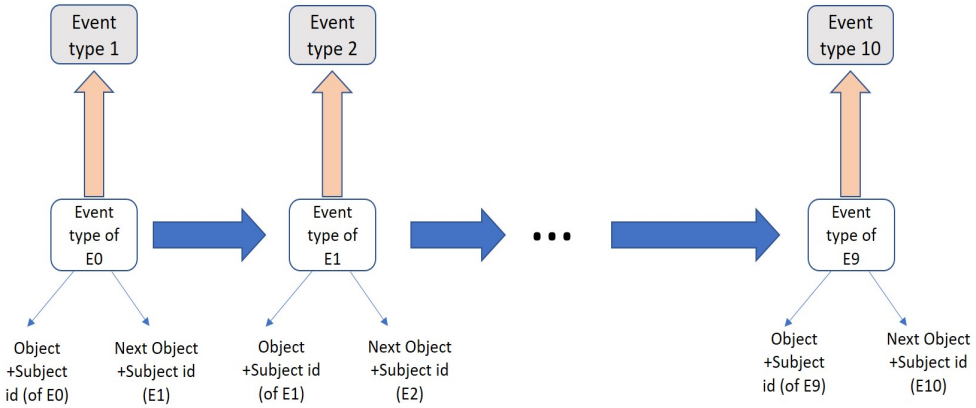

Figure 4: Model 1 - Simple LSTM architecture

Our second model is an LSTM neural network-based encoder-decoder model with an input size of 46 features and an output size of six features. The encoder LSTM encodes the sequence into a vector, and the decoder LSTM reconstructs this vector to the sequence event types. This model is also trained as anomaly detector.

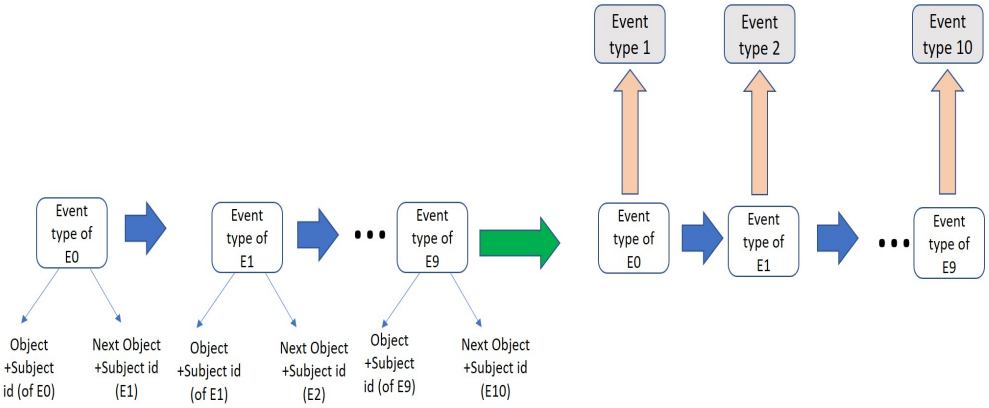

Figure 5: Model 2 - LSTM encoder-decoder architecture

5 Results & Conclusions

We performed our initial evaluation on the DARPA Engagement 3 Cadets and Theia dataset [19].

To evaluate our aggregator model, we created sequences from the 6,958 and 3,388 events flagged by KRYSTAL as suspicious in the Cadets and Theia datasets respectively. We considered a context window of 10 events; the 11th

Table 1: Summary of attacks in the DARPA Engagement 3 Cadets & Theia datasets

Dataset	Attack Name	Attack Start Time	Attack End Time	Duration	Attack Success/Failure
Cadets	Nginx Backdoor w/ Drakon In-Memory	06-04-2018 11:21 hrs	06-04-2018 12:08 hrs	0:47 hrs	Failure
Cadets	Nginx Backdoor w/ Drakon In-Memory	12-04-2018 14:00 hrs	12-04-2018 14:38 hrs	0:38 hrs	Success
Cadets	Nginx Backdoor w/ Drakon In-Memory	13-04-2018 09:04 hrs	13-04-2018 09:20 hrs	0:16 hrs	Success
Theia	Phishing E-mail w/ Link	10-04-2018 12:25 hrs	10-04-2018 13:45 hrs	1:20 hrs	Success
Theia	Firefox Backdoor w/ Drakon In-Memory	10-04-2018 14:00 hrs	10-04-2018 14:56 hrs	0:56 hrs	Success
Theia	Browser Extension w/ Drakon Dropper	12-04-2018 12:25 hrs	12-04-2018 13:30 hrs	1:05 hrs	Success

event served as the label for the sequence and we used a sliding window of one event. From the resulting 6,958 suspicious events in the Cadets dataset, we considered a training set size of 6,500 events, a test set size of 350 events, and a cross validation set size of 106 events. In the test set of 250 events, 240 sequences were created, and seven sequences were labeled as true positives. Similarly, from 3,388 suspicious events in the Theia dataset, we considered a training set size of 2,890 events, a test set size of 400 events, and a cross validation set size of 98 events. In the test set of 400 events, 390 sequences were created, and two sequences were labeled as true positives.

A sequence of 11 events is labeled as malicious if at least three events [23] of the actual attack were present in the context window and the 11th event is part of the attack. Our evaluation showed that the aggregator model was able to detect attack sequences with a recall of 100% on both datasets. Our evaluation results are summarized in Table 4. Based on true positives(TP), false positives(FP), treu negatives(TN) and false negatives(FN), precision and recall values have been calculated.

Table 2: Summary of aggregator model's performance on the DARPA Engagement 3 Cadets & Theia datasets

Dataset	Evaluation Criteria	TP	FP	TN	FN	Precision	Recall	FP rate
Simple LSTM architecture								
Cadets	Baseline	17	63	270	0	0.212	1	0.189
Theia	Baseline	26	41	333	0	0.388	1	0.110
LSTM encoder-decoder architecture								
Cadets	Baseline	17	137	196	0	0.110	1	0.411
Theia	Baseline	26	30	344	0	0.464	1	0.080

To conclude, the combination of rule-based approach with ML approach (LSTM) suits for anomaly detection for such data that contains causal relationships between events and lead us to raise an alert for all of the malicious activities with relative low ratio of false positive alerts. More over, we compared the usage of one LSTM to just predict the next event with 2 LSTMs as encoder decoder architecture (as auto encoder). Both architectures performed

well on both Theia and Cadets data sets. This reduction in false-positives helps SOC analysts to analyze less number of alerts and thus reduce the issue of alert fatigue in SOC teams.

References

[1] P. A. Networks. SOC architecture. [Online]. Available: https://www.paloaltonetworks.com/resources/guides/prevention-detection-and-response-for-security-operations-architecture-guide

[2] J. Navarro, A. Deruyver, and P. Parrend, "A systematic survey on multi-step attack detection," Comput. Secur., vol. 76, pp. 214–249, 2018.

[3] L. A. Johnson Kinyua, "AI/ML in security orchestration, automation and response: Future research directions," Intelligent Automation & Soft Computing, vol. 28, no. 2, pp. 527–545, 2021. [Online]. Available: http://www.techscience.com/iasc/v28n2/42057

[4] T. van Ede, H. Aghakhani, N. Spahn, R. Bortolameotti, M. Cova, A. Continella, M. van Steen, A. Peter, C. Kruegel, and G. Vigna, "Deepcase: Semi-supervised contextual analysis of security events," IEEE Security and Privacy, 2022.

[5] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," in Proceedings 2002 IEEE Symposium on Security and Privacy, 2002, pp. 202–215.

[6] "Testing soar tools in use," Computers & Security, vol. 129, p. 103201, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404823001116

[7] G. Sadowski. SOC maturity model. [Online]. Available: https://gorkasadowski.medium.com/the-old-soc-maturity-model-based-on-speeds-and-feeds-dcb17851a22e

[8] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 1285–1298.

[9] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li, J. Chen, X. He, R. Yao, J.-G. Lou, M. Chintalapati, F. Shen, and D. Zhang, "Robust log-based anomaly detection on unstable log data," in Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ser. ESEC/FSE 2019. New York, NY, USA: Association for Computing Machinery, 2019, p. 807–817. [Online]. Available: https://doi.org/10.1145/3338906.3338931

[10] A. Alsaheel, Y. Nan, S. Ma, L. Yu, G. Walkup, Z. B. Celik, X. Zhang, and D. Xu, "{ATLAS}: A sequence-based learning approach for attack investigation," in 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 3005–3022.

[11] Y. Shen, E. Mariconti, P. A. Vervier, and G. Stringhini, "Tiresias: Predicting security events through deep learning," ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 592–605. [Online]. Available: https://doi.org/10.1145/3243734.3243811

[12] A. Brown, A. Tuor, B. Hutchinson, and N. Nichols, "Recurrent neural network attention mechanisms for interpretable system log anomaly detection," in Proceedings of the First Workshop on Machine Learning for Computing Systems, 2018, pp. 1–8.

[13] Y. Shen and G. Stringhini, "{ATTACK2VEC}: Leveraging temporal word embeddings to understand the evolution of cyberattacks," in 28th USENIX Security Symposium (USENIX Security 19), 2019, pp. 905–921.

[14] H. Guo, S. Yuan, and X. Wu, "Logbert: Log anomaly detection via bert," in 2021 International Joint Conference on Neural Networks (IJCNN). IEEE, 2021, pp. 1–8.

[15] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun, and R. Zhou, "Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs," in Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 4739–4745. [Online]. Available: https://doi.org/10.24963/ijcai.2019/658

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[17] K. Kurniawan, A. Ekelhart, E. Kiesling, G. Quirchmayr, and A. M. Tjoa, "Krystal: Knowledge graph-based framework for tactical attack discovery in audit data," Computers & Security, vol. 121, p. 102828, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740482200222X

[18] SigmaHQ. Sigma rules. [Online]. Available: https://github.com/SigmaHQ/sigma

[19] DARPA. (2018) Transparent computing engagement 3. [Online]. Available: https://github.com/darpa-i2o/Transparent-Computing/blob/master/README-E3.md

[20] W3C. Web ontology language. [Online]. Available: https://www.w3.org/OWL/

[21] Ontotext. Graphdb. [Online]. Available: https://www.ontotext.com/products/graphdb/

[22] M. N. Hossain, S. Sheikhi, and R. Sekar, "Combating dependence explosion in forensic analysis using alternative tag propagation semantics," in 2020 IEEE Symposium on Security and Privacy (SP), 2020, pp. 1139–1155.

[23] Z. Li, Q. A. Chen, R. Yang, and Y. Chen, "Threat detection and investigation with system-level provenance graphs: A survey," CoRR, vol. abs/2006.01722, 2020. [Online]. Available: https://arxiv.org/abs/2006.01722