Editors: D. Balzarotti, davide.balzarotti@eurecom.fr | W. Enck, whenk@ncsu.edu | T. Holz, thorsten.holz@rub.de | A. Stavrou, astavrou@gmu.edu

Can Data Provenance Put an End to the Data Breach?

Adam Bates and Wajih Ul Hassan | University of Illinois at Urbana–Champaign

In September 2017, the world awoke to the news that Equifax, a consumer reporting agency and one of the pillars of the American credit system, fell prey to a data breach that led to the exposure of 147 million individuals' personal information. For Equifax, the coming weeks would include high-profile executive resignations, a steep drop in its stock prices, and an infamously ill-conceived public outreach effort; however, eventually the public's attention turned elsewhere. After all, Equifax was just the latest in a seemingly endless parade of data breach victims that included commercial titans like Target and eBay, political campaigns like Hillary Clinton's, and government agencies like the Office of Personnel Management. Today, the threat of the next data breach looms invisibly over every aspect of society.

The story of the Equifax intrusion, while frightening in scope, was run-of-the-mill at a technical level.[9] In early March 2017, attackers established an initial presence in the Equifax network by exploiting a recently identified vulnerability in a server that had not yet been patched. The vulnerable machine hosted a customer dispute portal that, while not offering much in the way of valuable data, allowed attackers to perform reconnaissance from inside of the Equifax network. Moving laterally through the network, the attackers would eventually come to access 51 Equifax databases, many of which included personally identifying consumer information. The attackers were ready to exfiltrate the data by May 2017, but data transfers at this size would arouse suspicion. Instead, the attackers slowly transmitted the data over a period of months by using 9,000 small database queries. On day 76 of the data exfiltration, the behavior was finally noticed by system administrators, who then painstakingly reconstructed this sequence of events by poring over months of system audit logs. Equifax did not begin its internal investigation until August, 145 days since the initial break-in.

How could an intrusion of such a massive scale go unnoticed for so long? In fact, it is unlikely that the attacker's actions went entirely undetected—large enterprise networks are protected by a battery of security monitoring products that search for everything from malware in email attachments to suspicious network traffic. During their time in the Equifax network, the attackers almost assuredly made some small misstep that resulted in a security alert being triggered. The problem is that this alert was just one in a sea of thousands of alerts that system administrators receive each week. A recent report by FireEye finds that most organizations in the United States receive upwards of tens of thousands of alerts per month; of these, only 48% are true alerts and only 4% of alerts are properly investigated.[2] For companies like Equifax, this creates a problem that is known to the industry as *threat-alert fatigue*. Against such odds, attackers will inevitably triumph over the system defenders.

While break-ins of this nature are inevitable, we see sources for hope and opportunity in the story of the Equifax breach. First among these is the matter of the attacker dwell time, which is the period of time the intruders spent on the system. To remain undetected, the attackers required a period of 145 days to accomplish their objective. Had the defenders detected and responded to the threat more quickly, they would have mitigated or altogether avoided the consequences of the intrusion. Second, the audit logs collected at Equifax were sufficient to reconstruct the intrusion and accurately estimate the extent of the damage. That audit logs are capable of providing such after-the-fact insight suggests untapped opportunities for protecting systems during live attacks.

In this article, we describe the emergence of data provenance as a means of analyzing audit logs for system security, reporting on both

Digital Object Identifier 10.1109/MSEC.2019.2913693
Date of publication: 9 July 2019

 Copublished by the IEEE Computer and Reliability Societies

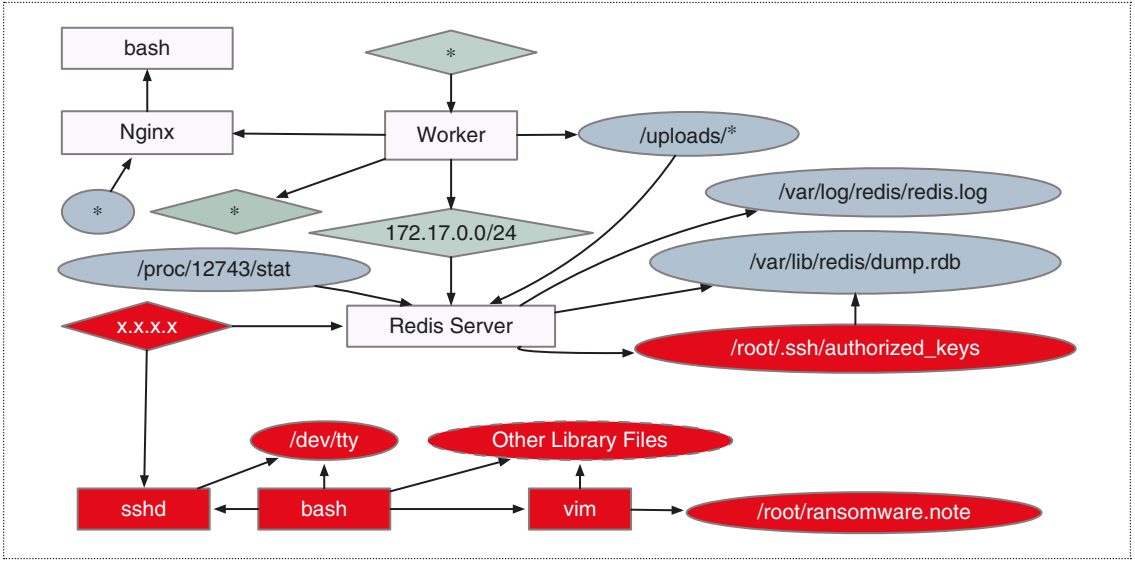

Figure 1. A provenance graph describing a ransomware attack on a Redis server. The attack provenance, shown in the bottom half of the graph, is highlighted in red.

community results as well as our own experiences. Provenance has already demonstrated its tremendous ability to provide transparency for security-sensitive events in systems. Through further research and integration, we argue that provenance-aware systems have the potential to effectively mitigate data breaches and other forms of advanced persistent threats.

What Is Data Provenance?

Data provenance describes the actions taken on a data object from its creation up to the present.[1] In a provenance-aware system, a stream of individual audit events (e.g., process A read from file 1) is incrementally parsed into a dependency graph that encodes the entire history of the system's execution. Data provenance is of growing interest to security researchers because it enables the causal analysis of suspicious events, providing transparency to large opaque systems. This event stream can be sourced by commodity auditing frameworks, such as Linux Audit or Microsoft's Event Tracing for Windows. Once constructed, this graph can then be queried to answer questions about the present state of the system. A back-trace query answers questions about the root cause of an event in the system by traversing the inbound edges from a given vertex. Once a root cause for a suspicious behavior has been identified, a forward-trace query can iteratively traverse the outbound edges from a vertex to trace the full propagation of the attack (sometimes called *provenance*). Critically, the results of these queries are not based on inferences of event correlation but, instead, encode the ground truth of the actual causal relationships between data objects.

Figure 1, an example provenance graph, depicts a phony ransomware attack on a Redis in-memory database server that affected more than 18,000 machines. The remote attackers (represented by x.x.x.x) execute a Redis configuration command over an open Transmission Control Protocol port to erase the contents of the database, write their own secure socket shell (SSH) key to the database, and then use another configuration command to copy the database to the root account's .ssh directory. In a separate connection, the attackers then open an SSH session and leave a ransomware note in the root account's home directory. When administrators discover the note, they can issue a back-trace query to attribute the note to a remote attacker. From there, they can use a forward-trace query to discover not only the attacker's method of entry but also that the ransomware attack is, in fact, a hoax and their data were erased.

Limitations of Current Enterprise Security Products

There exists a rich ecosystem of security products for monitoring suspicious activity on hosts. Such tools have been especially effective at scaling to support huge networks with relatively little administrative support, but current offerings are not without their limitations. The following are some issues with these products that could potentially be addressed by incorporating the historical context that data provenance provides.

Signature-Based Threat Detection

Antivirus scanning has been the first line of defense against

cyberattacks for many organizations over the last quarter of a century. It compares suspicious files' similarities against malware signature data sets to uncover threats in an organization. Antivirus software can only discover previously seen malware variants; it cannot detect the zero-day attacks or runtime compromises of legitimate software. Furthermore, attackers are intrinsically adaptive, regularly creating new variants.

Behavior-Based Threat Detection

In contrast to antivirus scanning, behavior-based threat-detection searches the system for activities that match anomalous behaviors. Enterprise threat-detection products not only employ anomaly detection classifiers to identify variations from typical behavior but also match against common patterns of malicious activity that can be either procedurally derived or manually defined by experts. Whether based on actual anomaly detection or handcrafted heuristics, behavior-based threat-detection products are prone to high rates of false alerts. They typically consider a limited window of each process' recent activity, not its entire historical context.

Security Indicator and Event Management

Security Indicator and Event Management (SIEM) products provide an orchestration layer for the various security products that are employed by an organization. Due to the high volume of alerts generated by connected products, an important role of SIEM software is to manage alerts by providing aggregation, deduplication, and correlation services. However, SIEM does not address the general problem of false alerts, making existing products only a partial solution to the threat-alert fatigue problem.

Traditional Auditing and Forensic Investigation

When determining the root cause of a security alert, investigators ultimately turn to audit logs as the definitive ground truth of system events. Traditionally performed semimanually, data provenance automates the process of tracking dependencies across log entries, improving the time-to-insight of investigations. However, traditional auditing frameworks also suffer from significant limitations. Among these is the dependency explosion problem.[6] In long-running processes, each output event appears to be dependent on all prior input events from the operating system's perspective. Early efforts to perform a graph-based analysis of audit events, such as King and Chen's BackTracker system,[5] attempt to rein in the dependency explosion problem by considering only the window of time that the attacker was active on the system. However, this still leads to false dependencies even during short timespans and is completely impractical for Equifax's 145-day attack window. Furthermore, audit logs grow rapidly in size and are dominated by records that describe the typical activity of the system. This imposes a tremendous storage and analysis burden when, in fact, only a small percentage of the logs will ever be relevant to an investigation.

What accounts for the capability gap between commercial offerings and the provenance-based techniques being proposed in the literature? One reason is that commercial products have not historically retained audit data at the fidelity required for data-provenance tracing. Consider two examples: 1) Microsoft Azure Sentinel offers graph-based explanations of threat alerts but only captures audit data sufficient to visualize the attack from the network perspective (i.e., host events are opaque),[10] and 2) Lacework's tracing offers a partial view into the host but monitors only process events and coarse-grained network flows, potentially overlooking vital interprocess dependencies like file activity.[11] In the remainder of this article, we will describe how current research is shifting the cost-benefit proposition of fine-grained auditing at scale.

Developing Practical Provenance-Based Security Tools

Automating Forensic Audits

When security alerts appear, forensic audits are the most reliable means of detecting the presence of a true attack; however, unfortunately, to conduct them they require one or more dedicated person hours from an expert analyst. In a recent study, we asked whether or not forensic audits could be partially automated to improve the responsiveness to security alerts. These efforts led to the development of the NoDoze system,[4] an automated mechanism for triaging security alerts that uses provenance analysis to differentiate false alerts from true attacks.

A demonstrative example of NoDoze in action is provided in Figure 2. This graph describes a web-browsing session in which two unusual activities occur, causing an existing threat-detection system to fire two security alerts. The first alert (on the left side of the graph) was caused by a user who downloaded a malicious 7-Zip attachment, leading to a ransomware attack. However, the second alert (on the right side of the graph) is not truly malicious; it was caused by a system administrator who downloaded and ran a set of diagnostic tools on the user's workstation.

When these two alerts are fired, NoDoze issues a back-trace and forward-trace query on the event that caused the alert. To each event in the resulting graph, it then assigns a raw anomaly score

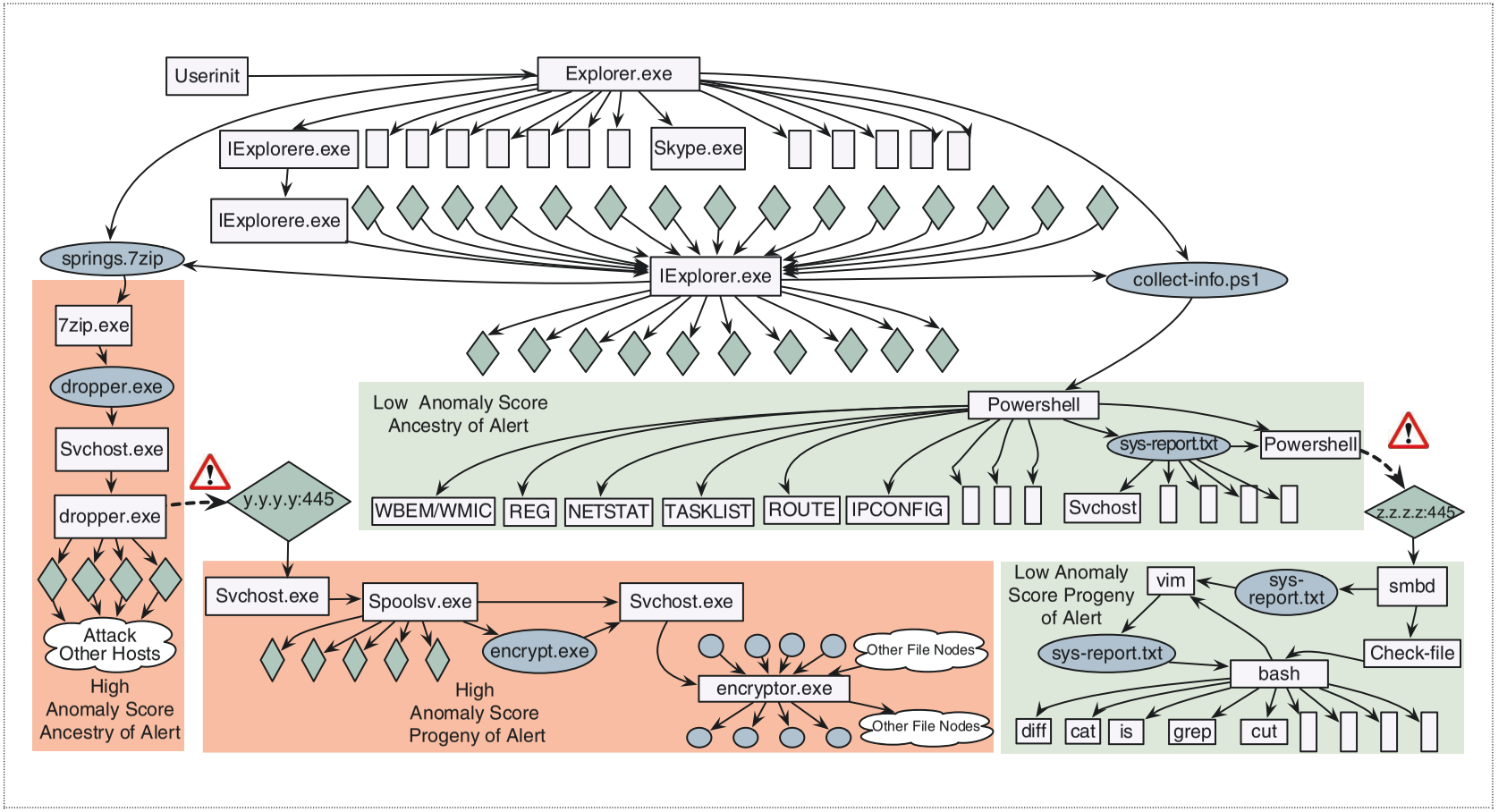

Figure 2. The provenance of another ransomware scenario, this time on a user workstation. The threat-detection software issued two alerts based on this workstation's activity; examining the historical context of each alert makes it easy to differentiate false alerts from true attacks. y.y.y.y/z.z.z.z.: blinded IP addresses.

based on the relative frequency of that event in the overall network. Finally, NoDoze calculates an aggregate anomaly score for the alert by conducting network diffusion on the scores in the graph and uses this aggregate score to sort the alerts. We deployed a NoDoze prototype on an actual 200-host enterprise network hosted by our collaborators at NEC Laboratories. When tested against a battery of more than 350 security alerts, we found that NoDoze effectively prioritizes true attacks over false alerts; in fact, the discriminative power of NoDoze was such that 84% of alerts could be immediately dismissed as false alarms. We believe that this first effort only scratches the surface of the potential for automated provenance-based forensics.

Leveraging Application Semantics

Provenance tracing at the operating-system level establishes connectivity between processes but is limited by the fact that many key pieces of forensic evidence exist only within application semantics. For example, the specific customer data stolen in the Equifax breach existed as database records that were opaque to kernel auditing frameworks. We are encouraged by the development of minimally invasive approaches to integrate application semantics into provenance-aware systems. An early effort in this space, Lee et al.'s BEEP,[6] provided a specific solution to the dependency explosion problem by decomposing the audit trail of a long-running application into individual units of work, allowing for more precise attack tracing. More recently, instrumentation-free methods for multilog analysis have emerged that provide tighter integration between system and application logs. An example of this approach is Pei et al.'s HERCULE system,[8] which identifies correlated events in different applications' log entries based on a social network analysis, enabling malicious communities of events to be identified. Bridging the semantic gap between software layers will be vital to the next generation of threat investigation tools.

Optimizing Audit Logs for Cyber Forensics

In the past several years, a rich body of literature has emerged that focuses on improving the cost–benefit ratio of system auditing. This work notes an interesting observation—auditing systems often collect far more information than is necessary or relevant to a threat investigation, such that many event records can be removed without the loss of forensic validity. A canonical example of such optimizations, presented in Lee et al.'s LogGC system,[7] concerns audit records that describe temporary files. Processes often create temporary files during execution that are deleted without ever interacting with another process. While the records of temporary file activity may be useful for performance profiling, they do not convey information flow between processes and can, therefore, be erased. In addition to systems like LogGC that eliminate specific sources of inefficiency, other tools, including our own Winnower system,[3] attempt to create a generic learning mechanism to remove semantically redundant log events.

Driven by the relentless nature of modern attackers, methods that leverage data provenance to defend systems are growing in popularity. As researchers are only just beginning to consider the meaningful automation of provenance analysis as well as the integration of multiple streams of audit data, we anticipate that this will continue to be an innovative space in security research. By simplifying the threat investigation and dramatically improving defender response times, data provenance is poised to put an end to data breaches in the years to come. ■

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under grants CNS-16-57534 and CNS-17-5002. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

References

1. L. Moreau and P. Missier, Eds., "PROV-DM: The PROV data model," World Wide Web Consortium, Apr. 30, 2013. [Online]. Available: https://www.w3.org/TR/prov-dm/
2. FireEye, Inc., "The numbers game: How many alerts is too many to handle?" Aug. 2018. [Online] Available: https://www.fireeye.com/offers/rpt-idc-the-numbers-game.html
3. W. Hassan, M. Lemay, N. Aguse, A. Bates, and T. Moyer, "Towards scalable cluster auditing through grammatical inference over provenance graphs," in *Proc. ISOC Network and Distributed System Security Symp.*, 2018. [Online]. Available: https://adambates.org/documents/Hassan_Ndss18.pdf
4. W. Hassan et al., "NoDoze: Combatting threat alert fatigue with automated provenance triage," in *Proc. ISOC Network and Distributed System Security Symp.*, 2019. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/nodoze-combatting-threat-alert-fatigue-with-automated-provenance-triage/
5. S. King and P. Chen, "Backtracking intrusions," in *Proc. ACM Symp. Operating System Principles (SOSP 02)*, 2003. [Online]. Available: https://dl.acm.org/citation.cfm?id=945467
6. K. H. Lee, X. Zhang, and D. Xu, "High accuracy attack provenance via binary-based execution partition," in *Proc. ISOC*

Network and Distributed System Security Symp., 2013. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2017/09/03_1_0.pdf
7. K. H. Lee, X. Zhang, and D. Xu, "LogGC: Garbage collecting audit log," in *Proc. CCS ACM SIGSAC Conf. Computer and Communications Security*, 2013, pp. 1005–1016. [Online]. Available: https://dl.acm.org/citation.cfm?id=2516731
8. K. Pei et al., "HERCULE: Attack story reconstruction via community discovery on a correlated log graph," in *Proc. 32nd Annu. Conf. Computer Security Applications*, 2016, pp. 583–595.
9. U.S. Government Accountability Office, "Data Protection: Actions taken by Equifax and federal agencies in response to the 2017 breach," Aug. 2018. [Online]. Available: https://www.gao.gov/assets/700/694158.pdf
10. E. Levi, "Introducing Microsoft Azure Sentinel, intelligent security analytics for your entire enterprise," Microsoft Corporation, Feb. 2019. [Online]. Available: https://azure.microsoft.com/en-us/blog/introducing-microsoft-azure-sentinel-intelligent-security-analytics-for-your-entire-enterprise/
11. V. Kapoor, "Introduction to polygraphs," Lacework, Inc., July 2017. [Online]. Available: https://www.lacework.com/introduction-to-polygraphs/

Adam Bates is an assistant professor at the University of Illinois at Urbana–Champaign. Bates received a Ph.D. in computer science from the University of Florida, Gainesville, in 2016. He has received the National Science Foundation Career award and was an Association for Computing Machinery (ACM) Special Interest Group on Security, Audit and Control Doctoral Dissertation Award runner-up. He has been a Program Committee member of the IEEE Symposium on Security and Privacy, USENIX Security Symposium, and Internet Society Network and Distributed Systems Security Symposium. He is a Member of the IEEE and ACM. Contact him at batesa@illinois.edu.

Wajih Ul Hassan is a Ph.D. student at the University of Illinois at Urbana–Champaign. Ul Hassan received a B.S. in computer science from the Lahore University of Management Sciences, Pakistan, in 2015. He is a 2019 Heidelberg Laureate Forum Young Researcher and received a 2019 Symantec Research Labs Graduate Fellowship. Contact him at whassan3@illinois.edu.

Digital Object Identifier 10.1109/MSEC.2019.2922810