Creating Data from Applications for Detecting Stealth Attacks

C. Warren Axelrod, Ph.D., Delta Risk LLC

Abstract. A major reason for security professionals not seeming able to protect fully against the rapidly changing threat environment and sophisticated attacks is that they often don't have available the necessary application security data for detecting and responding to such increasingly stealthy attacks. Furthermore, easily generated application security data are generally not reported timely or accurately enough for appropriate preventative action. In this article, we describe the issues confronting those attempting to create, collect, report, and respond to data–and which are more useful but harder to come by. We also suggest how these impediments might be overcome.

Introduction

Cyber security staffs collect and analyze huge volumes of security-related data. The problem is that applications do not generate the most useful data in the first place [1]. So many major data breaches reportedly occur without the knowledge of their victims. It appears that ChoicePoint [2], Heartland Payments [3], NASDAQ [4], and Epsilon [5], for example, only find out the degree to which data in their custody has been compromised when they are notified by third parties, such as VISA International, or called in forensics specialists, such as the NSA. For example, VISA International might observe concentrations of fraudulent payment card activity and trace them back to the company that had suffered the data leak. Even then, victim companies are frequently not able to determine who did what, and when.

There is ample evidence for the need for such information. According to the Verizon Business 2010 Data Breach Investigations Report [6], a large percentage of total breaches originate from insiders, specifically:

- **48 % of data breaches were caused by insiders**
- **48 % of data breaches involved privilege misuse**

It should be noted that the sample upon which the results of the Verizon Business analysis is based refers to data collected from their own clients, and therefore contains considerable bias. However, the general observations still make sense, even though the data are not necessarily representative of all organizations. A lesson taken from these statistics is that the lack of identification by victim organizations that a breach, particularly an inside job, has occurred leads one to the view that either the data are available but not analyzed (the supposition of the report) or they are not available, to which view the author subscribes. This premise, namely that the data are not generated unless specific actions are taken to create them, is the basis for this article.

According to a Veriphyr white paper [7], the major causes of insider attacks are the granting of excess unneeded privileged access rights, which often results from such practices as not deleting obsolete access rights, and so on. While this clearly contributes to the problem, it is by no means the only cause. Many insider attacks result from valid access rights that allow certain fraudulent activities to proceed undetected. In other cases, the system granting user access does not have fine enough granularity to be able to carve out only those access rights that are needed to perform a particular function.

Scope

Software systems and the data that they access are usually protected at a number of levels. Traditionally we find controls for system access, use, and access termination. Access security services include authentication and authorization, such as are embodied in the DoD's Common Access Card (CAC). The CAC was developed in order to be in compliance with the policy for a common identification standard for federal employees and contractors as described in Homeland Security Presidential Directive 12/HSPD-12 [8]. There are readily available monitoring and reporting products and services that identify and analyze user access logs, as well as high-level system use and data exfiltration logs. The latter typically cover e-mail and file transfer systems. Current products and services use readily available logged data, with few requiring the generation of additional data. This feature of not having to create additional data is often given as a benefit of such systems, and to the extent that they are simpler to install and get running, this may be the case. However, there are limitations in such an approach, not the least of which is the risk that key data may not be available for analysis.

Some Widely Publicized Examples

There are many cases in the government and private sectors where a lack of application-generated data has resulted in major breaches or system failures. An April 2011 attack on Sony PlayStation customer data–where the perpetrators used Amazon Web Services as their point of takeoff–is just such an example. In an article on the case [9], Pete Malcolm, CEO of Abiquo Inc., is quoted as saying, "There is no way of telling who's a good guy and who's a bad guy." An earlier release of the article, which was subsequently replaced with the referenced article, claims that Malcolm said that, "Amazon does not have the means to detect illegal uses of its servers."

A widely publicized example relating to the DoD is that of Private First Class Bradley E. Manning who was charged with unauthorized use and disclosure of U.S. diplomatic cables to WikiLeaks. An overview on Wikipedia [10] states the following:

"Manning had been assigned in October 2009 to a support battalion with the 2nd Brigade Combat Team, 10th Mountain Division, based at Forward Operating Base Hammer, near Baghdad. There he had access to the Secret Internet Protocol Router Network, used by the United States government to transmit classified information."

This would appear to be a case in which the perpetrator was authorized to access certain classified information but his use of the information was not flagged as suspect nor brought to the attention of authorities.

In this article, we propose an approach wherein one first determines in advance what additional data should be generated above and beyond the standard data logs. Data collection capabilities are introduced by injecting data requirements throughout the System Development Lifecycle (SDLC). When it comes to COTS and open-source software, where the acquiring entity does not have any control over the software development process, one must request that the vendor or community incorporate the necessary data collection capabilities.

Data Creation

As alluded to above, the author believes that this failure to discover intrusions is largely due to applications not collecting and reporting key information needed to detect hacker activity at the time of an attack or to trace such activities forensically after the fact. This lack of information occurs even when organizations have installed a full complement of monitoring, logging, and Security Information and Event Management (SIEM) tools and have staff diligently reviewing the outputs from these tools.

It is important to ensure that applications now in development, or currently being planned, will incorporate the instrumentation necessary to collect data required to alert security staff that a compromise is taking place or that one has happened. Consequently, application security professionals must be involved in the requirements, specifications, design, testing, deployment and operational phases of the SDLC. Some individual or group, with specific in-depth knowledge of not only an application's functionality but also what security data should be collected, has to be involved in all the above phases of the SDLC. The security data requirements generated by such specialists, though likely to be onerous, must be accepted and incorporated into applications if there is to be any hope of tackling the issue of undetected and undetectable attacks.

Producing the Data

Creating and monitoring relevant data are the keys to success but, as one would expect, the task of defining the full range of required security data is a major challenge since the required multidiscipline expertise rarely exists in the desired quantity and quality. Nevertheless, defining requirements is the right place to start.

Ideally one should collect data for every conceivable user or machine action and misuse that might occur within applications and against databases. This is, of course, an impossible task and, even were it theoretically attainable, it would be prohibitively expensive. The point is not that we must anticipate every possibility of misuse, but that the right individuals need to be involved in the SDLC process. This will encourage those responsible for protecting information assets to think about what instrumentation needs to be included in order to capture significant misbehavior.

If additional monitoring is implemented, not only does it serve to improve capturing inappropriate and suspicious activities, it also allows test engineers to cover a greater range of possible security gaps since data, which were previously not captured, now become available for testing a fuller range of functionality [11].

Current Data Collection Approach

Many IT shops just collect and report when users gain access to and depart from systems and networks as such events are readily obtained and are easy to specify and understand. They might also collect data about successful and unsuccessful logons in an attempt to uncover those trying to guess the passwords of others. While gathering such information is useful and often mandated by auditors, it has limited value since it does not tell you what authorized and unauthorized users did during their sessions. Also, if generic IDs are permitted and users are allowed to share passwords (both of which practices should be prohibited), there is no way of knowing if the authorized user or a different person is logging on. The user might know this other person or not.

As stated above, the main benefit to be derived from logon/logoff data is that they are easily defined and collected. It also does have some forensics value since one can determine retrospectively who supposedly accessed systems and when. On the other hand, such data are a rich source of audit findings, since auditors can easily cross reference employee lists against user IDs and determine if the user ID was used after the person left the organization.

The Need for a Better Approach

Stepping beyond collecting simple logon/logoff data usually requires a quantum jump in knowledge and effort, which raises questions about return on investment. However, since it is relatively easy for criminals to obtain application access credentials through social engineering means, such as phishing, checking someone at the gate is becoming less effective. Therefore, one is forced to look to other means of ferreting out unauthorized intruders, particularly when they are already inside the application. The sequence of complexity of data collection is shown in Figure 1. Even in this simple example, we see the need to provide more detailed requirements to the development team as we progress in complexity from Level 1 through Level 3.

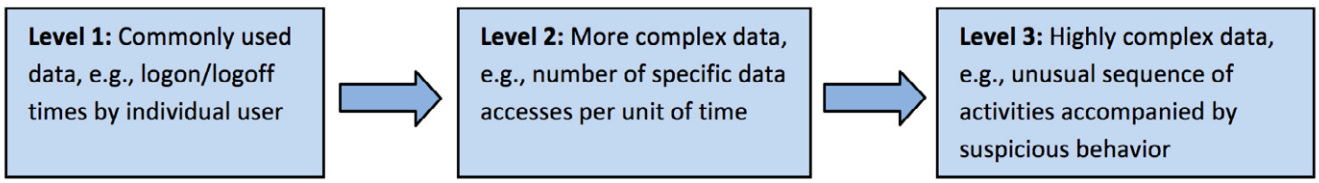

Figure 1: Increasing Complexity of Security Data

Table 1: Event Data Collected by Type of Data

Logon/Logoff Data	User Data	Transaction Data
Type of event (login/logoff) User name Terminal name Timestamps for logons and logoffs Timestamps for department changes Original and new departments Source IP of user Module of application in use Connection serial number Session ID	Type of event (user) Internal ID of user Internal company ID User's company Time when user inserted into database Time when data updated Time when user leaves system User's control number User's account number	Type of event (transaction) Transaction ID ID of user who inserted transaction ID of client requesting service IDs of corporate business group and division Time when transaction inserted into database Time when transaction closed User's company User's control number User's account number

We shall now examine the various levels of complexity according to the difficulty of security data identification, collection, analysis, and limitations on decision-makers' ability to respond to the metrics obtained.

Level 1: Generally Available Data

As indicated above, the easiest data to collect are generally those that are included with most information systems and security tools, such as Identity and Access Management systems. The need to know who logged on and when they did so is a basic requirement for auditing purposes. Even at this level, however, reports showing which applications a person may access are often inaccurate, out of date and difficult to interpret, according to the author's experience with many such reports. Also, the information is often available too late to be able to mitigate the impact of breaches.

Level 2: More Difficult to Identify, Create, Collect and Understand

In the moderately complex category, some data may be collected directly or calculated from other data. For example, the logon/logoff files can be analyzed to show the number of accesses by person by time of day, day of week, etc. Some systems have built-in capabilities to drill down further. For example, special-purpose e-mail monitoring systems provide the following:

- **Records of senders and recipients of e-mails**
- **Whether attachments were included**
- **Whether specific personal and other sensitive information was included in the body of the e-mail or in attachments, and**
- **What that specific information was.**

On the other hand, general business applications, particularly those that have been custom-built, are unlikely to have such capabilities.

In order to achieve this next level of data collection and analysis, it is often necessary to set up a specific initiative as part of the SDLC effort to add event monitoring, data collection, storage, analysis and reporting capabilities. This can typically take several man-months of effort, depending on the size and complexity of the systems. The data items in Table 1 were obtained through just such a project conducted by two companies and supported by a SIEM system vendor. As the table entries show, with a little effort, one can achieve much greater insight as to activities occurring within critical systems.

Level 3: Very Difficult to Identify, Create, Collect and Analyze

As we progress along the difficulty-complexity spectrum, we arrive at a level where a major difficulty is to determine what should be collected, and then the challenge is to come up with ways of creating, collecting, analyzing, reporting, and responding to the data in a timely fashion, often in real time.

In this category of data we look for more complex behavior patterns in order to identify anomalous behavior. For example, we might look at a series of user actions such as entering one application, retrieving specific high-value data, then moving directly to another application, such as e-mail or a social network, particularly if there is a policy against personal use of the organization's computer and network resources during business hours. This might suggest that data are being exfiltrated (or leaked) by the user. The major challenge here is identifying what can be considered normal behavior, as described in [12].

Reasons for the Lack of Data

Beyond the costs of collecting, logging and analyzing security data (which can be significant), the basic reason that we do not have those data that are needed for tracing specific activities through applications is that there are few advocates among stakeholders. These stakeholders might include business owners, and software and security engineers. While software and security engineers are often very supportive of including logging and monitoring of security-related data, they do not usually have the requisite understanding of the applications and knowledge as to which data would be helpful in tracking activities. The business owners, on the other hand, do not understand what is needed for managing system security.

There is some question of our very ability to gather meaningful data and come up with relevant abnormal behavior patterns, which might suggest malicious or otherwise damaging activities. This perhaps suggests that we are not looking in the right

Table 2: Detection and Mitigation of Unauthorized Access and Activities

Category	Characteristics	Detection	Mitigation
Unauthorized access	Multiple failed attempts at logging on. Series of failed attempts followed by a successful logon.	Login to account of employees or customers who have left, but their accounts were not inactivated.	Create/enforce procedure to delete user accounts immediately when someone changes role or leaves. Check for anomalies such as more than one person trying to log on using the same credentials.
Inappropriately authorized access	High level of "trolling", i.e., seemingly unrelated use of applications and access to data.	Atypical activities within applications and against databases. Numerous repeated attempts to access data for which the user is not authorized.	Restrict user access to applications' functionality and data on a need-to-know basis Modify access rights as soon as role changes.
Authorized access, outside "normal parameters"	Seemingly excessive number of reads, writes, data exports for someone with a specific role.	Monitor use of critical applications and access to data. If a person leaves the organization, check activities retrospectively for unusual behavior.	Proactively – Be aware of, and respond to, significant changes in business environment, user responsibilities, etc. Responsively – Observe events and/or question users after events.
Authorized access, within "normal parameters"	No obvious deviation from normal activities.	Other aspects of behavior, such as dissatisfaction with job, company, recent changes, etc.	Introduce a process to get feedback about worker work attitudes and other influencing attributes.
Unauthorized access, outside "normal parameters"	Excessive number of logon attempts and/or attempts to get at unauthorized functions and data. Excessive number of reads, writes, data exports, and the like.	This should be detected before access is granted, but if not, then other behavioral discrepancies should be considered. Number of logon attempts, number and type of data accesses suggest anomalous behavior.	Mitigation includes combinations of processes to detect unauthorized access and anomalous behavior within systems and networks.
Unauthorized access, within "normal parameters"	No obvious deviation from normal activities.	Other aspects outside system, such as unusual times of activity.	Implement monitors that look for suspicious factors, such as time of day, day of week, frequency of access attempts, etc.

places. The most relevant information could well be external to the systems being monitored. For example, if someone quits an organization with little or no notice, or begins to behave erratically, or badmouths his or her organization and management, then it might behoove the organization to more carefully monitor that person's activities. Such activities include suddenly sending out streams of e-mails containing sensitive information. Such "symptoms" are more likely to suggest that certain information might be copied that otherwise would not.

Approach to Collecting the Data

Nevertheless, there may well be behaviors that do indicate improper activities, but data about them are just not collected. There is a need to define security data requirements and build the necessary data collectors into the applications during the design and development stages.

There has to be collaboration among software engineers, systems engineers, developers, application security experts, business owners, etc. But first we need to understand the various categories of security data: their characteristics, how they might be detected, and what must be done to reduce or eliminate the impact of the activities being recorded. We indicate these aspects in Table 2 for various categories of access.

While the Department of Defense exerts strong management in terms of who is authorized to access systems and maintains control over that access via the CAC, as mentioned above, there are still possibilities with respect to inappropriate registration of users and inappropriate use of those access credentials, as exemplified by the Manning case described above.

The Economics of the Proposed Approach

Costs of generating the data consist of the costs of the time and effort to include security data requirements, building the collectors in, populating the logs, analyzing the data, and so on. These can often be estimated fairly accurately.

On the other hand, it is very difficult, if not impossible, to determine specific and accurate returns on investment for building instrumentation into applications and developing metrics based on the new data generated and collected. Generally the justification for such efforts to improve security takes a few high-profile breaches, such as those mentioned above, where having had specific data might have prevented or mitigated an attack, or at least facilitated forensics efforts.

The costs and losses depend upon the entity, which was the victim, and its affiliations. However, there is a limit to the true out-of-pocket costs that an entity and its stakeholders (management, shareholders, employees, and customers) can incur. For example, the upper limit of loss, to which shareholders might be subjected, is the value of the stock they hold. Company officers and executives might be sued, which eventuality is often covered by insurance. However, management and other employees might lose their jobs resulting in monetary losses, psychological distress, etc. If the entity is forced to downsize or goes out of business, customers and business partners will have to find other sources and affiliations respectively, which might be difficult and expensive to achieve in certain markets.

In order to detect stealth behavior within applications, it is necessary to generate and analyze meaningful data about user activities. A major issue is that much of the required data is not created because the necessary data generators have not been incorporated into the application software.

Actual losses may far exceed what an entity and/or its stakeholders have the ability to pay. Thus a company might file for bankruptcy and any excess losses above what is covered by insurance and assets are borne by creditors or the public. This is in essence a case of "moral hazard" whereby the losses of the victim organization are limited. If another entity, such as the government, does not make up the difference between what the victim company can pay and what is believed to be the full cost of an incident, then part of the responsibility and uncovered costs fall upon individual victims and/or society at large. These are the so-called social or indirect costs.

Some Illustrative Examples

The author was involved in a situation where an institutional customer of a financial broker/dealer requested that the latter provide details of the activities within a specific system of a particular individual over a specified period of time. Not only was the broker/dealer unable to provide a report of generally available login and logout data because the data were comingled with the same information of other customers, but the application did not collect and log specific activities within the application itself. The author recommended that the requisite data collection capabilities be incorporated in a subsequent release of the application.

In another case, an organization detected hacking software residing on an employee's desktop computer. The employee was suspended pending the results of a forensic analysis, which took several weeks to complete. The result of the analysis was that the software had not been used. Here is a case where appropriate instrumentation might have avoided the unpleasantness of the employee's suspension altogether.

Summary and Conclusions

In order to detect stealth behavior within applications, it is necessary to generate and analyze meaningful data about user activities. A major issue is that much of the required data is not created because the necessary data generators have not been incorporated into the application software. In this article we have shown areas for improvement in data generation so as to increase situational awareness as it relates to stealth attacks against applications. While the process involves significant enhancements to the software engineering process, it is claimed that the resulting actionable information is well worth the effort.

ABOUT THE AUTHOR

C. Warren Axelrod, Ph.D., is a senior consultant with Delta Risk, a consultancy specializing in cyber defense, resiliency and risk management. Previously, he was the chief privacy officer and business information security officer for U.S. Trust, the private wealth management division of Bank of America. He was a co-founder of the Financial Services Information Sharing and Analysis Center. Dr. Axelrod won the 2009 Michael Cangemi Best Book/Best Article Award for his article "Accounting for Value and Uncertainty in Security Metrics," published in the ISACA Journal, Volume 6, 2008. He was honored with the prestigious Information Security Executive Luminary Leadership Award in 2007. He received a Computerworld Premier 100 IT Leaders Award in 2003.

Dr. Axelrod has written three books, two on computer management, and numerous articles on information technology and information security topics. His third book is Outsourcing Information Security, published in 2004 by Artech House. His articles "Investing in Software Resiliency" and "The Need for Functional Security Testing" appeared in the September/October 2009 and March/April 2011 issues of CrossTalk magazine, respectively. He holds a Ph.D. in managerial economics from Cornell University, as well as an honors M.A. in economics and statistics and a first-class honors B.Sc. in electrical engineering, both from the University of Glasgow. He is certified as a Certified Information Systems Security Professional and Certified Information Security Manager.

REFERENCES

1. Axelrod, C. Warren, "Accounting for Value and Uncertainty in Security Metrics," ISACA Information Systems Control Journal (November 2008).
2. Scalet, Sarah D., "ChoicePoint Data Breach: The Plot Thickens – A timeline of key events surrounding the ChoicePoint data breach," CSO Online, May 2005. Available at <http://www.csoonline.com/article/220341/choicepoint-data-breach-the-plot-thickens>.
3. Vijayan, Jaikumar, "Update: Heartland breach shows why compliance is not enough – The huge data breach one year ago hammers home the need for multilayered security controls, Computerworld, January 2010. Available at <http://www.computerworld.com/s/article/9143158/Update_Heartland_breach_shows_why_compliance_is_not_enough>.
4. Jones, Penny, "NASDAQ brings in NSA to investigate breach –NSA involvement points to seriousness of 2010 hack, and importance of NASDAQ to US," DataCenterDynamics Focus, April 2011. Available at <http://www.datacenterdynamics.com/focus/archive/2011/04/nasdaq-brings-in-nsa-to-investigate-breach>.
5. Bradley, Tony, "Epsilon Data Breach: Expect a Surge in Spear Phishing Attacks," PCWorld, April 2011. Available at <http://www.pcworld.com/businesscenter/article/224192/epsilon_data_breach_expect_a_surge_in_spear_phishing_attacks.html>.
6. 2010 Data Breach Investigations Report, Verizon, Inc., 2010. Available at <http://www.verizonbusiness.com/resources/reports/rp_2010-data-breach-report_en_xg.pdf>.
7. Glithero, Bob, "From Insider Abuse to Insider Accountability: Identity Analytics Discover Insider Threats," Veriphyr Inc., 2011. Available at <http://www.veriphyr.com/download/Veriphyer_WP_From-Insider-Abuse-to-Insider-Accountability.pdf>.
8. Homeland Security Presidential Directive 12/HSPD-12, 2004. Available at <http://www.cac.mil/assets/pdfs/HSPD_12.pdf>.
9. Galante, Joseph et al., "Sony Network Breach Shows Amazon Cloud's Appeal for Hackers." Bloomberg News. Available at <http://www.bloomberg.com/news/print/2011-05-15/sony-attack-shows-amazon-s-cloud-service-lures-hackers-at-pennies-an-hour.html>.
10. Overview available at <http://en.wikipedia.org/wiki/Bradley_Manning>.
11. Axelrod, C. Warren, "The Need for Functional Security Testing," Crosstalk, March/April 2011. Available at <http://www.crosstalkonline.org/storage/issue-archives/2011/201103/201103-Axelrod.pdf>.
12. NITRD (Networking and Information Technology Research and Development) Workshop, "Abnormal Behavior Detection Finds Malicious Actors," June 2011. Available at <http://www.nitrd.gov/fileupload/files/MaliciousBehavior_2011_NITRD_workshop.pdf>.