International Journal of
Learning Management Systems
ISSN: 2785-9568
VOLUME 1, ISSUES 2, 2013
www.egyptfuture.org/ojs/

Modeling Security Vulnerabilities in Learning Management Systems

HyunChul Joh

Gwangju Institute of Science and Technology, South Korea

Email: joh@gist.ac.kr

Recieved: 3 Feb. 2013, Revised: 17 Feb. 2013 , Accepted: 19 Feb. 2013

Published online: 1 Jul. 2013

Abstract: In many educational institutes, learning management systems are essential parts of delivering class materials not only for on-line courses but also on-campus classes. The primary purpose of learning management system is to provide proper virtual educational environments and convenient communicational channels between instructors and students letting them to overcome the barrier of time and space for the general schooling. In this paper, we are investigating the security vulnerabilities in such learning management systems. Especially, we are examining the vulnerability discovery process in the two popular learning management systems, Moodle and Blackboard, using datasets spanning a decade. We applied two vulnerability discovery models to examine the vulnerability discovery process quantitatively. The discovery models allow us to estimate the approximate number of vulnerabilities likely to be discovered in software systems in the future. It also helps policy makers when they need to decide which learning management systems will be adopted for their organizations when security is an extremely important factor. Result shows that the vulnerability discovery process in learning management systems could be predictable which could be utilized for IT risk assessment in organizations having the systems.

Keywords: Moodle, Blackboard, learning management system, vulnerability discovery model, quantitative analysis.

Introduction

For the past decade, learning management systems (LMS) have been adopted by numerous organizations, especially by educational institutes, including many prestigious schools such as California Institute of Technology (Moodle, https://courses.caltech.edu), Colorado State University (Blackboard, https://ramct.colostate.edu), or University of Oxford (Sakai, https://weblearn.ox.ac.uk). LMS provides us proper virtual educational environments and convenient communicational channels between instructors and students.

When we search the keyword learning management system in the Internet, we are going to have many versions of definitions for the terminology. However, generally, when we say learning management systems, we are indicating software systems which should be able to handle and provide certain requirements including delivering course

materials appropriately, tracking students' activities, giving and collecting assignments, having QnA bulletin boards, organizing event calendars, etc.
Currently, there are approximately 300 active LMSes in the market (http://www.capterra.com/learning-management-system-software), and among them, there are handful of key players dominating LMS market, such as Moodle (https://moodle.org), Blackboard (http://www.blackboard.com), or Sakai (http://www.sakaiproject.org). Some are proprietary, others are open source systems. According to Mallon and Clarey (2012), 1.8 to 1.9 billion dollars will be spent globally on LMS market in 2013. Hence, the LMS market has prosperous future.

However, like every coin has two sides, LMS also gives headaches to system administrators when LMS-connected school databases and private information are not properly managed nor protected well. When LMS systems are compromised, such as leakage of sensitive personally identifiable information, or tampering of integrity of school databases, it will causes tremendous damages on the reputations for academic institutes.

When organizations make decisions of which LMS they will adopt, there are several key factors that they need to mull over, such as how much does it cost, what kinds of functionalities does LMS provide, how secure the system is, etc. First of all, budget always matters. One of the reasons why Moodle occupies sheer number of market share is that because it is freely available. Also, some researchers think that free availability contributes that Moodle scores high in satisfaction surveys in general (http://elearningtech.blogspot.kr/2007/09/lms-satisfaction-features-and-barriers.html).

However, open source system does not necessarily mean it is free to run, or even free to install in a way of what they want. First, there must be staff members to operate the system, and then, faculty members need to be trained. Moreover, an open source system, such as Moodle, many of its key functionalities are depends on Plug-in or Add-on modules having no systemic design documentation. As a result, many organizations which like to adopt a LMS, frequently, customize the open source LMSes that costs not a small amount of money.

Among many choices, Moodle is the most popular LMS as an open source software system. It is an open source, so that people can customize it for their schools relatively easily. The challenge of its continuous popularity in the future depends on how the system is secure compared to its competitors. In this paper, we are examining security vulnerabilities in vulnerability discovery process, mainly in Moodle LMS. We found that Moodle has higher number of vulnerabilities compared to its commercial counterpart, Blackboard. For a comparison purpose, Blackboard LMS also examined.

In the meantime, a software vulnerability is a subset of vulnerability in general. As a result, a software vulnerability should inherit the characteristics what the general vulnerability has. In plain English, the word vulnerability represents susceptibility to malevolent manipulations. A software vulnerability has been defined in many ways, but in this paper, we follow the definition from CVE (http://cve.mitre.org/about/terminology.html): "An information

security vulnerability is a mistake in software that can be directly used by a hacker to gain access to a system or network". Just like what Frei stated in his Ph.D dissertation (Frei, 2009), we also consider vulnerabilities listed in the CVE directories (http://cve.mitre.org) only. Hence, it does make sense for this paper to use the definition from CVE since all the LMS vulnerability datasets used in this paper have CVE identification numbers. Common Vulnerabilities and Exposures (CVE) is a publicly available and free to use list or dictionary of standardized identifiers for common computer vulnerabilities and exposures.

A better understanding of the quantitative characteristics of LMS vulnerabilities allow for system administrators with better insight into their works to prevent security related accidents. Not only for system administrators but also staff members, faculties and students would potentially benefit from the quantitative vulnerability analysis because all the participants can take actions before the risk reaches unacceptable levels based on the results from the analysis.

In comparison with qualitative methods, quantitative analysis allows evaluations to be more precise when considering how much effort is needed to protect a LMS system or how high the system exploitation risk is when a security policy in place (Vache, 2009). An actual data driven empirical analysis, followed by some statistical tests to make a final decision, provides objective results, which seems to be a more proper technique for answering questions regarding to LMS security concerns.

This paper is organized as follows. The next section goes over the related works in LMS and some of the software vulnerability discovery process. In Section 3, software vulnerability discovery processes are introduced. In Section 4, we consider the vulnerabilities in Moodle and Blackboard, and examine how vulnerability discovery models fit the available datasets. Finally, some major observations are discussed, and the conclusion is presented.

Related Works

Hilmi et al. (2011) mainly give explanations of security elements of a learning management system according to the ten security domains defined by International Information Systems Security Certification Consortium (Tipton, 2009). The ten domains are i) Information security and risk management, ii) Access control, iii) Cryptography, iv) Physical security, v) Security architecture and design, vi) Business continuity and disaster recovery planning, vii) Telecommunication and network security, viii) Application security, ix) Operation security, and x) Legal, regulations, compliance and investigations. The authors say that the paper could provide an introductory guideline for a learning management system administrator to understand the issues and possibilities related to the safety of learning management system.

Ajlan and Zedan (2008) praise Moodle while doing their comparative studies among the LMSes available in the market both proprietary and open source software systems. They conducted the study in two phases. The first phase

is based on the features and capabilities of LMSes in view of Learner Tools (communication, productivity, and student involvement), Support Tools (Administration, course delivery, and content development), and Technical Specifications Tools (hardware/software tools and pricing/licensing tools). Based on the three categories, they concluded that Moodle and Sakai are the best LMS products. In the second phase, Moodle is compared with other LMSes in technical aspects. The results they provided show that Moodle beats all other comparatives.

Hernandez and Chavez (2008) analyzed Moodle security services with UML and the corresponding security vulnerabilities. The authors further suggest possible solutions how to avoid the security loopholes they presented. A similar analysis, yet focusing on security settings in Moodle, was studied by Stapic et al. (2008). They state that Moodle significantly depends on the security measures on a host server, and present a summary of most common security flaws and suggest optimal settings of Moodle with a host server. One thing that we miss from the paper is that none of them provide a specific example with CVE identifiers which might help to investigate further studies.

Meanwhile, researchers have proposed software vulnerability discovery models (VDMs) (Alhazmi & Malaiya, 2008) to project the current trend and to estimate the discovery process since vulnerability data have been publicly available from early 2000s. For the past decade, a few VDMs have been proposed by researchers which include Anderson (2002), Rescorla (2003), Alhazmi and Malaiya (2008), and Ozment and Schechter (2006). Vulnerability discovery models are separated into time-based and effort-based models. Time-based models use calendar time as the independent variable factor and an effort-based model uses the number of installed system or the number of system users as the main factor. These models incorporate the effect of the rising and declining market share on the software.

Vulnerability discovery process

In Figure 1, the solid S-shaped line shows the shape of the Alhazmi-Malaiya Logistic (AML) vulnerability discovery model (Alhazmi & Malaiya, 2008). The model is originally based on the observation that the attention given to an operating system increases as it gains market share, it peaks at some time and then drops when a newer competing version is introduced. It is assumed that the cumulative number of vulnerabilities is governed by two factors in Equation 1. The first factor increases with the time because of the rising share of the installed base. The second factors declines as the number of remaining undetected vulnerabilities declines. The saturation effect is modeled by the second factor. Assuming that the vulnerability discovery rate is given by Equation 1, Equation 2 can be obtained by solving the differential equation which gives the cumulative number of vulnerabilities Ω. Parameters A and B are empirical constants determined from the recorded data. C is a constant introduced while solving Equation 1. The model is defined for time values t from the negative infinity to the positive infinity.

$$\omega(t) = A\Omega(B - \Omega) \tag{1}$$

$$\Omega(\mathrm{t}) = \frac{B}{BC^{-ABt} + 1} \tag{2}$$

During software release period, vulnerability discovery rate gradually increases due to the gaining market share. This phase is called learning phase, as shown in Figure 1. In the linear phase, the discovery rate reaches the maximum due to the popularity, and finally, in the saturation phase, vulnerability discovery rate slows down. The two transition points and mid-point are mathematically defined in (Alhazmi & Malaiya, 2006). Even though the AML model was originally designed based on the behavior of vulnerability discovery in operating systems, with other types of software systems, the model performs well too (Woo et al., 2011).

Linear vulnerability discovery model is another example of time-based model that we examine in this paper for a comparison purpose. The linear model assumes that the vulnerability discovery rate is constant in any circumstance. The simple linear model can be expressed as Equation 3.

$$\Omega(t) = k + S \times t \tag{3}$$

where S is a slope that indicates vulnerability discovery rate and k is a constant. In general, applying the linear model to the entire lifecycle of a software system and obtaining a good result is difficult.

There are important factors to impact a vulnerability discovery rate. Most significant factors are code size, software age, popularity and software evolution. Several studies (Akiyama, 1971; Compton & Withrow, 1990; Hatton, 1997; Rosenberg, 1997) have examined the relationship between the code size and the number of defects. The studies show that the numbers of defects or errors are increasing as code size is getting bigger.

Market share is one of the most significant factors impacting the effort expended in exploring potential vulnerabilities. A higher market share provides more incentive to explore and exploit vulnerabilities for hackers since they would find it more profitable or satisfying to spend their time on a software system with a higher market share. The effect of rise-and-fall in the market share is implicit in the AML model. Meanwhile, time-based models reflect software age more explicitly than an effort-based model.

Discovery process in LMS

In this section, vulnerability discovery trends for the total number of known vulnerabilities of Moodle and Blackboard are fitted to VDMs, so that we could estimate how well the models reflect the actual vulnerability discovery process. The vulnerability datasets were downloaded at National Vulnerability Database (NVD; http://nvd.nist.org) managed by NIST. Notice that Moodle represents an open source software system while

Blackboard is a proprietary closed source system. We examine all the vulnerabilities reported at the NVD web site regardless how the defects had arisen.

Figure 2 shows the number of known vulnerabilities with model fittings on Total. In the figure, vulnerabilities are categorized by severity levels (http://nvd.nist.gov/cvss.cfm) of High, Medium and Low. Total is summation of all three of them. We analyze the total number of vulnerabilities only. There have been 159 and 17 vulnerabilities are reported for Moodle and Blackboard respectively on October 2012. Notice that after year 2009, there has not been any vulnerability report for Blackboard at NVD database which implies that the system might be in the saturation phase in Figure 1.

Market share is the most significant factor influencing on the effort putting on detecting vulnerabilities. The more number of users means the higher chances finding defects. Table 2 shows the amount of web traffics among the web sites including Compete traffic, Twitter followers, Facebook likes, LinkedIn followers and Klout score, posted on the Web on October 2012. The number of users and customers for Moodle overwhelm the numbers from Blackboard's. This might be because Moodle is open source, so that many users could try the system without any obligations. There could be a college student whose term project is related to LMS, and the student can install and register Moodle system without any cost. A flaw in the end user data is that each LMS vendor might have different methods to count their number of users.

Nevertheless, with all of those uncertainties, still Moodle has bigger market share than Blackboard when we consider all the educational institutes having not enough budges, especially from the third world countries, although many rich universities also use Moodle. And according to the philosophy of the AML model, software having bigger market should have higher number of vulnerabilities detected.

Here, we fit the two VDMs of AML and linear. First, with a visual observation, AML model fits better than the linear model for both LMSes. In Figure 2 (a), the growth trend of vulnerability discovery shows a strongly linear pattern until about the middle of 2010. After that, there are almost no vulnerabilities reported. Then, on July 2012, Moodle announced 83 vulnerabilities. Because of the sudden raise, linear model does not fit well. Meanwhile, AML tries to catch up the sudden raise on 2012. Interestingly, the upsurge is falling into right before the time frame when Moodle version 2 was released on November 2010, and 28 versions were released during the flat period (Oct 2010 ~ Jun 2012), which should produce more bugs in the system due to the active code evolutions. Moreover, Moodle enlarges its territory by supporting more language selections.

In Figure 2 (b), the growth pattern of total number of vulnerabilities shows a nice S-shaped growth trend. As a result, AML seems fit well. A figure from *LMS Market Share for All Institutions* (http://www.deltainitiative.com/bloggers/author-phil-hill/new-mentality-enters-lms-market), which shows some of the popular LMS market share, presents that, in year 2004 and 2005, Blackboard occupied big portion of the LMS

market which might affect the uprising in around at the end of year 2005 in Figure 2 (b). Another influence could be that, in February 2006, WebCT was officially acquired by rival company Blackboard.

To see whether model fittings are statistically significant, Chi-square (χ^2) goodness of fit test has been conducted. For the fit to be acceptable, the test statistic value should be less than the corresponding critical value for the given alpha level and the degrees of freedom. In this paper, P-values need to be greater than 0.05 to be statistically significant since we are using the alpha level of 0.05. P-value closed to 1 indicates a better fitting. Goonatilake et al. (2007) provide a material how to apply χ^2 test in detail.

Table 2 shows the results of the goodness of fit tests for the total number of vulnerabilities from Figure 2 and Figure 3 (a). The table says that AML performs better than the linear fitting because the P-values for AML are greater than the linear model. We also can say that Moodle is more actively evolved than Blackboard based on the values of parameter S; Moodle has higher slope value.

Figure 3 (a) shows the number of vulnerabilities with AML fitting for Moodle, but the number of vulnerabilities, between July 2010 and June 2012, is modified, so that the sudden uprising can be eliminated. During the modified period, we assume that the discovery process is linear since AML model indicates that the first transition point T1 from Figure 1 occurs at November 2012 in the fitting result from Figure 3 (a), and the modified period is relatively closed to the time point. The time point can be achieved by $T_1 = \ln\left[BC/\left(2+\sqrt{3}\right)\right]/AB$ (Alhazmi & Malaiya, 2006). Based on the AML model fitting in Figure 3 (a), we could expect continuous vulnerability reports for a while in Moodle since the second transition point will be falling into August 2019. However, since AML model assumes no evolution in code, a long-run projection might be not accurate. As shown in Table 2, in Figure 3 (a), the fit result for AML model is statistically significant now. We have not put the linear fitting on the modified data since the result is almost identical from Figure 2.

Discussion

The result found in this paper shows that, despite of significant code evolutions in LMSes, a vulnerability discovery model is found to be applicable, and this could give good indications to LMS administrators for optimal resource allocations. Current Moodle release policy (http://docs.moodle.org/dev/Updates) is that every six months (Jun and Dec), there are major releases while every two months (Jan, Mar, Jul, sep, and Nov) there are minor releases.

Further, they release updated version every week with new fixes produced by stable development process. This very active Moodle release policies might lead more number of vulnerabilities than its commercial counterpart.

Condon et al. (2008) observed that occurrences of software security incidents increase during the academic calendar, and the most relevant form of institutional type of seasonality is summer vacation from school (Koc & Altinay,

2007). As a result, it should be an interesting investigation to see whether there are any seasonal correlations between vulnerability detections in LMSes and the calendar time.

Table 3 shows the number of vulnerabilities reported by month for the two LMSes. In both LMSes, July and December have the highest frequencies which agrees to the statement of above (Condon et al., 2008). Figure 3 (b) shows boxplot for the number of vulnerabilities counted by month. We have consistent vulnerability reports on March, July, September, and December. Although Table 3 indicates there are eight incidents in Moodle on April, based on the boxplot, they are outliers. The reason why March and September have consistent vulnerability reports than other months might be that in some countries those months are the beginning of the spring and fall semesters when LMSes are actively accessed by new users.

Static analysis is frequently used in software reliability engineering where systems' characteristics are examined empirically. Alhazmi and Malaiya (2005) introduced *Vulnerability density* (V_{KD}) which shows how many vulnerabilities are in a system per thousand lines of source code. V_{KD} can be evaluated as Equation 4.

$$V_{KD} = \frac{Known\ Vulnerabilities}{Ksloc} \tag{4}$$

The *Kilo source lines of code size* (*Ksloc*) for Moodle was determined by using CLOC (http://cloc.sourceforge.net) tool. Total number of source code lines for Moodle 2.4, released on December 2012, is 1,253,692 excluding comments. Hence, V_{KD} value for Moodle 2.4 is about 0.126. Moodle 2.4 has less V_{KD} value than Apache 2.2.11 HTTP server (0.353), but has greater value than Windows NT (0.0109) (Woo et al., 2011). This reflects that operating systems usually have large parts which do not play active roles in accessibility while HTTP servers frequently deal with accessibility role and have smaller size of source code. It could be said that properties of LMS are positioned somewhere between operating systems and HTTP servers.

Conclusion

Security related concerns must considered at the very beginning of LMS adoption since features related to security and privacy policies are hard to be added to LMS which is already existing at runtime (Hommel, 2010). Applicability of quantitative models for the vulnerability discovery rates for the two popular learning management systems, Moodle and Blackboard, are explored, focused on Moodle. Results demonstrate that the vulnerability discovery process in LMSes follows certain patterns, which could be modeled well enough to be estimated. In case of Moodle, although the AML model fitting on the raw dataset is not applicable, we found a way how to fit the model with a reasonably modified dataset, so that we could estimate the future vulnerability discovery trend in the system.

Here, it has been proved that the AML model, originally proposed for operating systems, is also applicable to LMS systems. The model can be used to estimate vulnerability discovery rates, which might be integrated with risk assessment models in the future (Joh & Malaiya, 2011). Furthermore, we have seen that there is a possibility that vulnerability discovery models can let us to optimize the development and maintenance process to achieve more secure LMSes by telling us when vulnerabilities will be reported in high frequency. Most of all, the results found in this paper demonstrate the applicability of vulnerability discovery models for learning management systems that have undergone evolution significantly.

We might need a further research which considers evaluating the impact of evolution of LMS products that go through many versions by explicitly considering the shared codes and functionalities, vulnerabilities inserted and removed in the process and the impact on resource allocation for testing and patch development. Also since a good model fitting does not necessarily mean a good prediction capability, further study is needed to examine the ability of the models to estimate future vulnerability discovery trends in LMSes.

References

Akiyama, F. (1971). An example of software system debugging. In *World computer congress* (pp. 353–359).

Al-Ajlan, A., & Zedan, H. (2008). Why moodle. In *Future trends of distributed computing systems, 2008. ftdcs '08. 12th ieee international workshop on* (p. 58 -64). doi: 10.1109/ FTDCS.2008.22

Alhazmi, O. H., & Malaiya, Y. K. (2005). Quantitative vulnerability assessment of systems software. In *Rams '05: Proceedings of the rams '05. annual reliability and maintainability symposium*, 2005. (pp. 615–620). IEEE Computer Society.

Alhazmi, O. H., & Malaiya, Y. K. (2006). Prediction capabilities of vulnerability discovery models. In *Rams '06: Proceedings of the rams '06. annual reliability and maintainability symposium*, 2006. (pp. 86–91). Washington, DC, USA: IEEE Computer Society. doi: http://dx.doi.org/10.1109/RAMS.2006.1677355

Alhazmi, O. H., & Malaiya, Y. K. (2008). Application of vulnerability discovery models to major operating systems. *Reliability, IEEE Transactions on, 57*(1), 14–22. doi: 10.1109/ TR.2008.916872

Anderson, R. (2002). Security in open versus closed systems – the dance of boltzmann, coase and moore. In *Conf. on open source software: Economics, law and policy* (pp. 1–15).

Compton, B. T., & Withrow, C. (1990). Prediction and control of ADA software defects. *Journal of Systems and Software, 12*(3), 199 – 207. doi: DOI:10.1016/0164-1212(90)90040-S

Condon, E., He, A., & Cukier, M. (2008). Analysis of computer security incident data using time series models. In *Issre '08: Proceedings of the 2008 19th international symposium on software reliability engineering* (pp. 77–86). Washington, DC, USA: IEEE Computer Society. doi: http://dx.doi.org/10.1109/ISSRE.2008.39

Frei, S. (2009). *Security econometrics - the dynamics of (in)security*. Eth zurich, dissertation 18197, ETH Zurich. (ISBN 1-4392-5409-5, ISBN-13: 9781439254097)

Goonatilake, R., Herath, A., Herath, S., Herath, S., & Herath, J. (2007). Intrusion detection using the chi-square goodness-of-fit test for information assurance, network, forensics and software security. *J. Comput. Small Coll.*, *23*, 255–263.

Hatton, L. (1997). Reexamining the fault density-component size connection. *IEEE Softw.*, *14*, 89–97.

Hernandez, J., & Chavez, M. (2008). Moodle security vulnerabilities. In *Electrical engineering, computing science and automatic control, 2008. cce 2008. 5th international conference on* (p. 352 -357). doi: 10.1109/ICEEE.2008.4723399

Hilmi, M., Pawanchik, S., & Mustapha, Y. (2011). Exploring security perception of learning management system (lms) portal. In *Engineering education (iceed), 2011 3rd international congress on* (p. 132 -136). doi: 10.1109/ICEED.2011.6235375

Hommel, W. (2010). *Learning management system technologies and software solutions for online teaching: Tools and applications* (Y. Kats, Ed.). IGI Global. doi: 10.4018/978-1-61520-853-1.ch003

Joh, H., & Malaiya, Y. K. (2011). Defining and assessing quantitative security risk measures using vulnerability lifecycle and cvss metrics. In *The 2011 international conference on security and management (sam)* (p. 10-16).Gronlund, N.E. (2006). *Assessment of student achievement (8th ed.)*. Boston: Pearson Education, Inc.

Kim, J., Malaiya, Y. K., & Ray, I. (2007). Vulnerability discovery in multi-version software systems. In Hase '07: *Proceedings of the 10th ieee high assurance systems engineering symposium* (pp. 141–148). Washington, DC, USA: IEEE Computer Society. doi: http:// dx.doi.org/10.1109/HASE.2007.78

Koc, E., & Altinay, G. (2007). An analysis of seasonality in monthly per person tourist spending in turkish inbound tourism from a market segmentation perspective. *Tourism Management, 28*(1), 227 - 237. doi: DOI:10.1016/j.tourman.2006.01.003

Mallon, D., & Clarey, J. (2012). *Learning management systems 2013*: *The definitive buyer's guide to the global market for learning management solutions* (Tech. Rep.).

Ozment, A., & Schechter, S. E. (2006). Milk or wine: does software security improve with age? In *Usenix-ss'06: Proceedings of the 15th conference on usenix security symposium*. Berkeley, CA, USA: USENIX Association.

Rescorla, E. (2003). Security holes... who cares? In *Ssym'03: Proceedings of the 12th conference on usenix security symposium* (pp. 75–90). Berkeley, CA, USA: USENIX Association.

Rosenberg, J. (1997). Some misconceptions about lines of code. In *Proceedings of the 4th international symposium on software metrics* (pp. 137–142). Washington, DC, USA: IEEE Computer Society.

Stapic, Z., Orehovacki, T., & Danic, M. (2008). Determination of optimal security settings for lms moodle. In *31st mipro international convention on information and communication technology, electronics and microelectronics* (p. 84 -89). Opatija.

Tipton, H. F. (2009). Official (*isc*)2 guide to the *cissp cbk*. Boca raton: Auerbach Publications.

Vache, G. (2009). Vulnerability analysis for a quantitative security evaluation. In *Esem'09: Proceedings of the 2009 3rd international symposium on empirical software engineering and measurement* (pp. 526–534). Washington, DC, USA: IEEE Computer Society. doi: http://dx.doi.org/10.1109/ESEM.2009.5315969.

Woo, S.-W., Joh, H., Alhazmi, O. H., & Malaiya, Y. K. (2011). Modeling vulnerability discovery process in apache and iis http servers. *Computers & Security, 30*(1), 50 - 62. doi: DOI:10.1016/j.cose.2010.10.007

Table 1: χ^2 *Goodness of fit test results from Figure 2 and 3 (a).*

	AML						Linear			
	A	B	C	χ_s^2	χ_c^2	P-value	S	χ_s^2	χ_c^2	P-value
Moodle	2.58E-06	9243.1627	0.1086	300.3907	132.1444	1.00E-20	0.8244	343.3068	132.1444	4.38E-27
Blackboard	0.0020	23.1042	2.5762	59.9337	129.9179	0.9999	0.1284	86.2490	129.9179	0.9300
Figure 3 (a)	4.01E-05	807.5685	0.1514	121.3837	132.1444	0.1309	NA			

Table 2: Amount of web traffics among the popular web sites.

Vendor	Num. of Users	Num. of Customers	Compete Traffic[1]	Twitter followers[2]	Linkedin follwers[3]	Facebook likes[4]	Klout score[5]
Moodle	60,039,749	67,513	116,463	9,667	1,010	11,479	55
Blackboard	20,000,000	4,400	2,980,000	13,039	6,817	3,262	66

Source: http://www.capterra.com/blog/learning-management-systems/top-lms-software-solutions-infographic/

[1]www.compete.com, [2]www.twitter.com, [3]www.linkedin.com, [4]www.facebook.com, [5]www.klout.com

Table 3: Number of vulnerabilities by Month.

Vendor	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	Total
Moodle	3	6	4	8	0	4	89	1	19	1	3	21	159
Blackboard	0	1	0	2	0	0	4	1	0	2	0	7	17

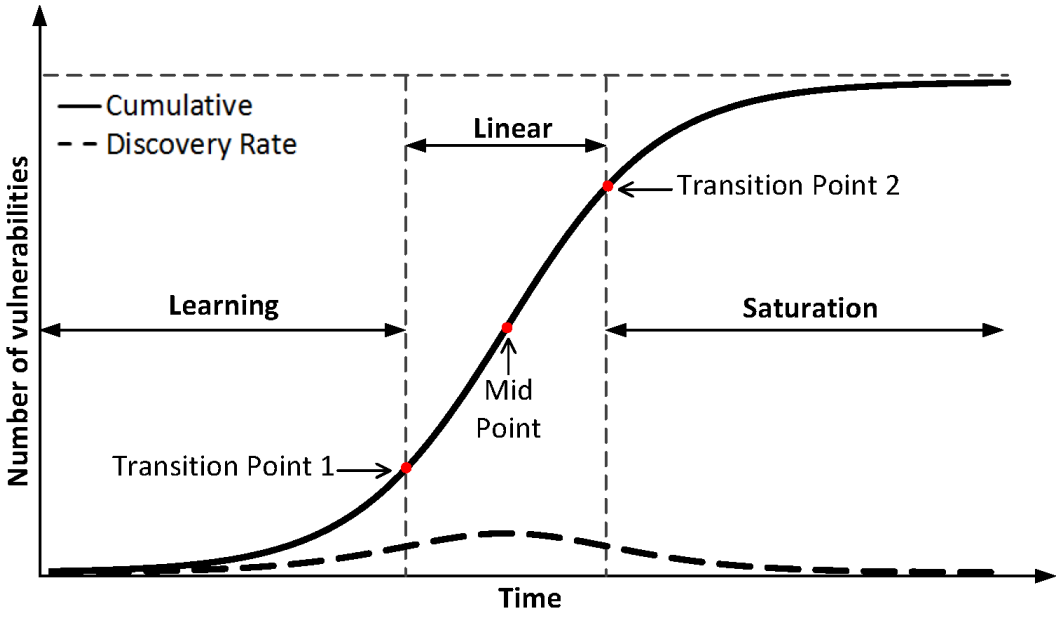

Figure 1. Three pphases in S-shaped AML model.

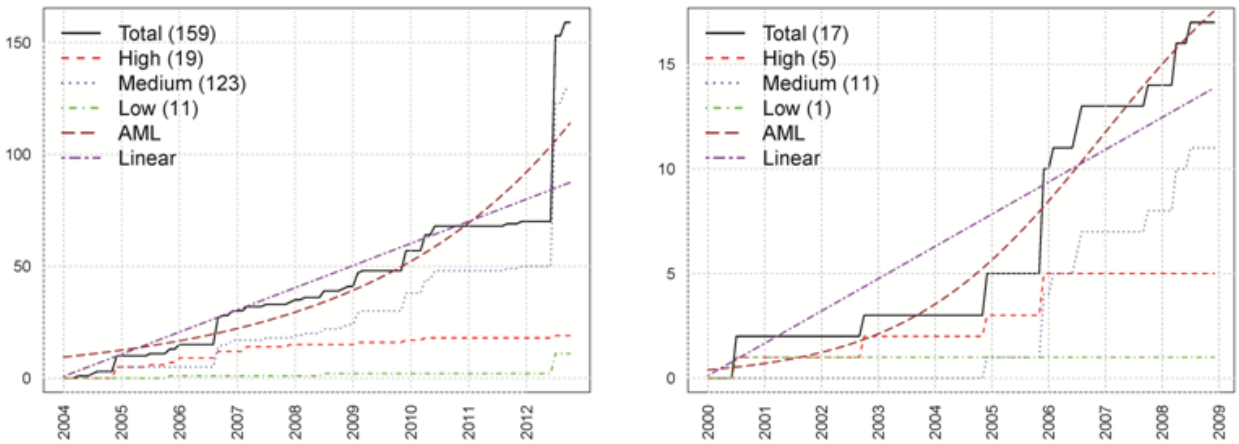

Figure 2. Number of vulnerabilities and model fittings; numbers in the parentheses are number of vulnerabilities reported on Oct 2012.

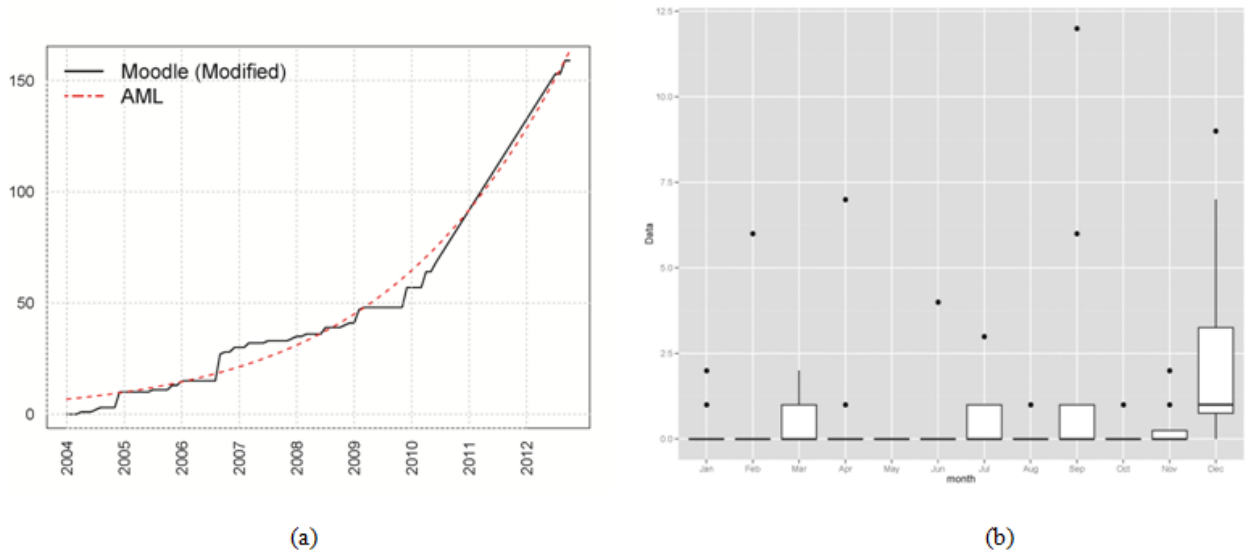

Figure 3. (a) Number of vulnerabilities with Modified datasets (2010-07 ~ 2012-06) (b) Boxplots for number of vulnerabilities by month in Moodle.