

A METHODOLOGY FOR CONTINUOUS COMPUTER SECURITY AUDITING

Urko Zurutuza, Roberto Uribeetxeberria, Jesus Lizarraga, Inaki Velez de Mendizabal
Computer Science Department
Mondragon University, Spain
{uzurutuza,ruribeetxeberria,jlizarraga,ivelez}@eps.mondragon.edu

ABSTRACT

This paper presents an approach to a methodology for continuous computer security auditing. It consists on measuring and controlling the security level of any organisation as a continuous process. This process establishes a method that permits organisations to control and to monitor the security level in real time, to be able to take the appropriate countermeasures in case a deviation occurs. The ability to measure the current state of the security is essential to continue improving the safeguard of our information. This will allow a proactive position regarding to security issues as one can be aware of the level acquired as well as the level required. The paper gives a brief overview of security metrics, discusses how the metrics are obtained and provides an example of carrying out a continuous audit.

KEYWORDS

Computer security, continuous auditing, security auditing, security metrics.

1. INTRODUCTION

Measuring our organisations security level has always been a difficult challenge. Current audit techniques allow obtaining a snapshot of the state of the security in a given moment. When an audit is accomplished, corrective measures are taken to increase the security level and solve deviations. Nevertheless, the period between two audits is not evaluated and important events may happen during this time (see figure 1). These events can decrease the security level of the systems significantly making successful attacks possible as the system may not be secure anymore.

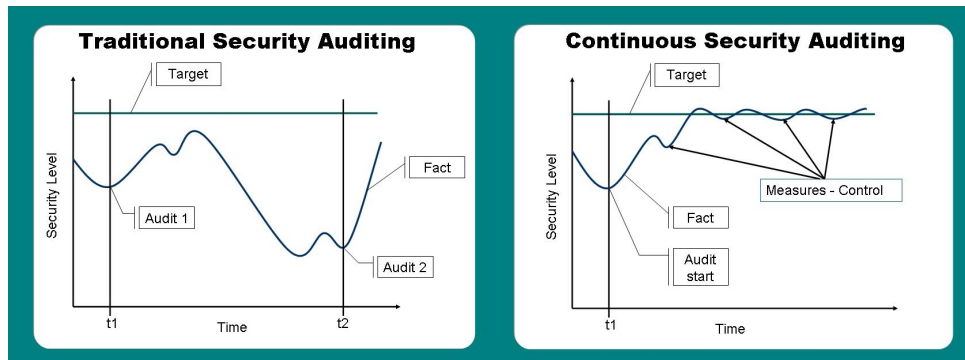


Figure 1. Traditional security auditing Vs. continuous security auditing

While the concept of continuous auditing (CA) [CICA99] is over a decade old, the rapid advances in technology have now made it feasible [Kog99] [Gro89]. Up to the moment, CA had only a financial meaning. In a computer security context, the process of CA consists on measuring and controlling the security level, performing continuous monitoring in time and taking proofreaders actions if necessary. This

process establishes a method that allows organisations to control and to monitor the security level in real time to be able to take the appropriate measures in case a deviation occurs.

At this moment, we are working in a project called secu-AUDIT, which seeks to analyse and define a methodology for the realisation of continuous audits of network security in organisations.

2. METHODOLOGY

The CA model starts on an initial meditation of what do we need to measure in order to get an overall security state. In this new method, it is necessary to select automatically measurable metrics and the tools that will provide those measurements. Once we collect the security metrics in a database, we suggest a formula based on a statistical analysis of the behavior of each metric. This way, the security administrator and even the enterprise management can easily analyse the organizations information system's security level in real time just by means of checking its representation and assess the risk that involves any change of the level.

2.1 Tool of tools

A security tool, "munix", has been developed in this University as a result of the secu-AUDIT project. This tool audits the system in a continuous and automatic way and updates periodically its data bases of vulnerabilities directly from Internet.

The idea of "munix" is to obtain the information related to security from the maximum quantity of tools or information sources in order to measure the security level as accurately as possible. "munix" is not limited to only discover which are the vulnerabilities of our organisation's IT system (as many auditors do), but also detects intrusions and monitors the most significant servers and services.

Based on a Linux system, the first approach integrates three known security tools available under GNU General Public License [GNU89]: Nessus vulnerability scanner, Snort Intrusion Detection System [Roe99] (rule-based system [Cas93]) and Nagios monitoring package.

2.2 Auditing the data: security metrics

We understand security metrics as a uniform monitoring method and an objective way to document our organisation's security attitude. Security metrics are required to understand the current state of security and its improvement in time. It is necessary to establish control points in order to evaluate the state of the effectiveness of the security. This will improve the protection level. The metrics will permit us to work on security as a dynamic process and not as a final product.

In order to calculate the security level, a set of metrics that allow to carry out the measures and the pursuit of the security level in situ and in a completely automatic way has been identified (which could be expandable). Measuring the security metrics, it is possible to determine their influence in the audit process and see whether a continuous improvement is obtained.

This way, starting from the data that are capable to catch the previously installed tools, we suggest 6 different measures:

- Number of high/medium/low risk vulnerabilities
- Number of intrusions
- Time of down servers
- Time of down services

These variables are in their simpler form, and they will be modified later for an easier analysis.

It seems logical to evaluate the interaction of the variables with the environment. The security level can differ from big organisations to smaller ones, even if the measure of a variable is the same. For example, it would not be unusual to have a lot of login attempts, intrusions or viruses in a big organisation, and this would give us a certain security level. On the other hand, if a small organisation had the same values, the security level value should be much lower.

To solve this problem, the measures have been transformed into metrics that are independent from environment.

Taking the number of vulnerabilities as an example, we realise that it was necessary to divide the measure into two metrics; in one hand the total number of vulnerabilities per host, and in the other, the number of different types of vulnerabilities. The reason of dividing the metric is that, for example, we can find a great number of vulnerabilities in an organisation, but most of them can be the same one (in different hosts). In other words, it is like one key that opens several doors with the same bolt. Instead, finding few vulnerabilities but all of them different could be even more dangerous, and an attacker could have much more possibilities of breaking into the system (several keys that open several doors).

Now, these metrics stop from being environment dependant, and it would be valid for companies of any size.

In order to define more exactly our metrics, their behavior have been reflected in a graphical way. We consider a continuous model as a base for their behavior: the bigger the independent variable is, the lower will be the dependent one (security level). After that, it is only necessary to translate the behaviors into a mathematical form, trying to reflect them in the simplest form (for example polynomial). Continuing with the previous example, the result would be the following:

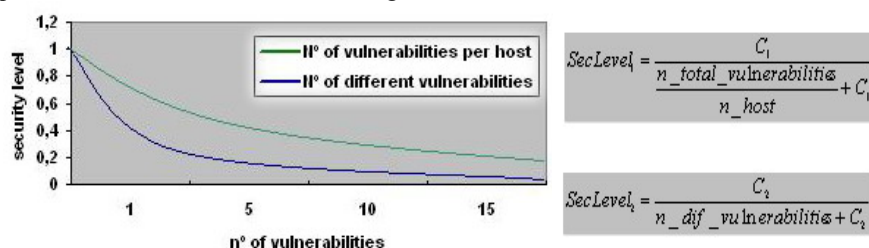


Figure 2. Behavior of the vulnerabilities, and formulas that represent the behaviors

being C_i the number of vulnerabilities that make security level decrease down to 50%. This way, each company administrator can establish what number of vulnerabilities can tolerate in his system (the tolerance to the risk may differ from one organisation to other).

Now, it is necessary to link these metrics with the calculation of the level of global security. The global security level depends on a function involving all metrics (and time): $level = f(m_1, m_2, \dots, m_i, t)$

Based on a study accomplished by the Computer Sciences Corporation [CSC02], the impact of each metric within the total security of the system has been evaluated. The threats or metrics that can be automatically obtained have been extracted from this study. Then, it has been calculated the weighting or ratio scale of the impact to include in the final formula. Table 1 shows how the study has been interpreted:

Table 1. Impact of the metrics within the total security of the system

Metric	Impact	%	Tool
1 Number of high risk vulnerabilities per host	***	10.34	Nessus
2 Number of medium risk vulnerabilities per host	**	6.9	Nessus
3 Number of low risk vulnerabilities per host	*	3.44	Nessus
4 Number of different kind of high risk vulnerabilities	***	10.34	Nessus
5 Number of different kind of medium risk vulnerabilities	**	6.9	Nessus
6 Number of different kind of low risk vulnerabilities	*	3.44	Nessus
7 Number of intrusions	****	13.8	Snort
8 Time of down servers	****	13.8	Nagios
9 Time of down services	****	13.8	Nagios
Total		100	

Finally, linking these metrics, their behavior within the security and their impact over the total security, the formula that will measure the security level in any given moment is obtained:

$$\begin{aligned}
Level = & \left(\frac{C_1}{\Gamma_Down_Servers + C_1} \right) * 13.8 + \left(\frac{C_2}{\Gamma_Parada_Services + C_2} \right) * 13.8 + \left(\frac{C_3}{N_Intrusions + C_3} \right) * 13.8 + \\
& + \left(\frac{C_4}{\frac{N_tot_Vul_h}{N_Hosts} + C_4} \right) * 10.34 + \left(\frac{C_5}{\frac{N_tot_Vul_m}{N_Hosts} + C_5} \right) * 6.9 + \left(\frac{C_6}{\frac{N_tot_Vul_l}{N_Hosts} + C_6} \right) * 3.44 + \\
& + \left(\frac{C_7}{\frac{N_dif_Vul_h}{N_Hosts} + C_7} \right) * 10.34 + \left(\frac{C_8}{\frac{N_dif_Vul_m}{N_Hosts} + C_8} \right) * 6.9 + \left(\frac{C_9}{\frac{N_dif_Vul_l}{N_Hosts} + C_9} \right) * 3.44
\end{aligned}$$

Figure 3. Global security formula

being C_i the constants that make security level decrease down to 50% for each variable (defined by the system administrator).

3. FURTHER WORK: OVERALL RISK ASSESSMENT

Traditionally, risk assessment methodologies are based upon a simplistic model of risk which identifies threats and the vulnerabilities they exploit to affect a security breach. Countermeasures which mitigate the threat/vulnerability pairs are identified. Loss due to a security breach is calculated based on the probability of the threat overcoming the countermeasure and creating the breach [Dra94].

The security CA model can be extremely helpful to asses those risks. It is useful for evaluating the effectiveness of each countermeasure against each threat/vulnerability pair, analysing the impact of the overall security level after any action is taken. The next step to be taken during this research is how to obtain the cost of improving the security level of each metric. This will depend on the technology and resources used for that meaning.

4. CONCLUSION

Continuous security auditing is possible with little effort by means of combining existing security tools. Although this type of combined system is not enough by itself, it is a good starting point to make our site a safer place. The system that has been implemented can be improved adding more security metrics so more aspects of the security of our company will be covered, and this will give higher reliability to the system.

Measuring security effectiveness is a challenging enterprise. None of the metrics can be used productively without understanding the relative importance of system security for the organisation's mission.

An improvement on the global security is achieved when a patch is installed on an organisation. Thanks to this unique system, it is possible to minimise the efforts by only choosing the most significative ones.

REFERENCES

- [Cas93] Caswell, S. et al, 2003. *Snort 2.0 Intrusion Detection*. Syngress, USA.
- [CICA99] Canadian Institute of Chartered Accountants. Research Report on Continuous Auditing. Toronto, 1999.
- [CSC02] Computer Science Corporation, CSC Global Information Security Services. Security value metrics. http://www.csc.com/aboutus/lef/mds69_off/uploads/Enterprise_Info_Risk_Management.pdf. Page 8.
- [Dra94] David L. Drake, Katherine L. Morse, 1994. The security-specific eight stage risk assessment methodology. *In Proceedings of 17th NIST-NCSC National Computer Security Conference*. Baltimore, USA, pp. 441-450.
- [GNU89] Stallman, Richard, 1989. GNU General Public License, <http://www.gnu.org/copyleft/gpl.txt>.
- [Gro89] Groomer, K., Murthy, U., "Continuous Auditing of Database Applications: An Embedded Audit Module Approach," *Journal of Information Systems*, Spring, 1989.
- [Kog99] Kogan, A. et al, 1999. Continuous Online Auditing: A Program of Research. *Journal of Information Systems*, Vol. 13, No. 2, pp. 87-103.
- [Roe99] M. Roesch, Martin, 1999. Snort - Lightweight Intrusion Detection for Networks. *In Proceedings of Thirteenth Systems Administration Conference (LISA '99)*. Berkeley, California, USA, pp. 229-238.