Risk Assessment of E-KTP Web Application Vulnerability Based on NIST 800-30 Framework

Darmanto Lie, Charles Lim
Swiss German University
EduTown BSD City Tangerang 15339 Indonesia, Phone: +622130450045 - Fax: +622130450001
e-mail: darmanto.li@gmail.com, Charles.lims@gmail.com

Abstract
During the development of “E-KTP search” web application, the programmer did not take security into consideration. There are no vulnerability assessments, mitigation strategies and penetration testing or vulnerability scanning to ensure the security. Therefore this research is conducted to create a vulnerability assessment for “E-KTP search” web application for countermeasures against the latest security threats. The methodology of this paper is based on NIST 800-30 and OWASP Top 10 vulnerabilities. The steps that were conducted are: vulnerability identification, current control analysis, likelihood determination, impact analysis, risk determination, prioritize action and implement control recommendation. The result of this research shows the existing “E-KTP search” web application have vulnerabilities that considered as low to medium risk level. The vulnerabilities can be mitigated or eliminated by implementing recommended controls which correspond to the vulnerabilities.

Keywords: vulnerability, risk, assessment, web application, E-KTP

1. Introduction
Since 2006 Indonesia has been preparing its own electronic identity card named e-KTP (Electronic Kartu Tanda Penduduk). BPPT (Ministry of Research and Technology) role as the agency for the assessment and application technology did the first batch prototype of e-KTP on September 2009 in 6 sub-districts in Indonesia. The pilot data is now stored in Depnakertrans, as the Indonesian government that record all Indonesian biodata. Now, the pilot data only record about one hundred thousand of Indonesian biodata. Once the whole plan is succeed in 2011, the data estimated will be around two hundred millions record. All these data will be used by the government for identifying people, analyze growth population in certain area, etc. A secure web application is needed to process all the data, so it can be viewed and analyzed properly.

Since the data is essential and crucial, security should be the main focus that needs to be considered in the “E-KTP search” web application. The web application should be assured that the data is shared among authorized user only and the data that being transferred is not being altered or sniffed by hacker. The problem is during the development of “E-KTP search” web application, the programmer did not take security into consideration. There are no vulnerability assessment, the mitigation strategy and attack simulation to ensure the security.

To solve this problem, vulnerability assessment should be done for “E-KTP search” web application. The assessment includes vulnerability identification, current control, planned control, and remediation process to assure security and reliability of the application.

The assessment is based on NIST 800-30 which is a document that contains guide for assessing risk and mitigating risk (Stoneburner et al. 2002). This document contains two methodology flow charts. The first one is known as risk assessment methodology flow chart which consists of nine steps. Some of the risk assessment steps essential for this research methodology are found in here. NIST 800-30 has been developed by NIST in furtherance of its statutory responsibilities under the Computer Security Act of 1987 and the Information Technology Management Reform Act of 1996. This guideline is used for many federal organizations to securing sensitive information thus it is suitable for “E-KTP search” web application that contains information of Indonesia citizens.

In this research, OWASP (Open Web Application Security Project) Top 10 Web Application Security Risk is used to spot on the top 10 most critical risk and identifying vulnerabilities that lies in the web application. The top 10 most critical risk in web application includes SQL injection, Cross-site scripting (XSS), broken authentication and session management control, insecure direct object reference, cross-site request forgery, security
misconfiguration, insecure cryptography, failure in protected restricted URL, failed in transport layer protection and failure in validating URL before redirect and forward user. On the other hand, this research is also use reference of security risk that is release by Shirey (2007) for dictionary attack, md5decrypter (2006) for rainbow table attack, Hjelmvik (2011) for Man-in-the-middle attack, and Scott (2010) for oracle padding attack.

2. Research Method

The research methodology is based on the NIST 800-30 (Stoneburner et al. 2002). In NIST 800-30, risk assessment methodology flow chart and risk mitigation flow chart are two methodologies that will be simplified. In assessing vulnerabilities, the methodology steps will be taken from risk assessment methodology flow chart. The steps that will be used for assessing vulnerabilities include vulnerability identification, control analysis, likelihood determination, impact analysis, risk determination and control recommendation. Since the research scope includes remediation process, two more steps from the risk mitigation methodology will be used, namely prioritize action and implementing the control recommendation.

For vulnerability identification, according to NIST 800-30, the inputs that needed for conducting vulnerability identification process are security checklist or guidelines and security test result. The security checklist or guideless is helpful in listing potential vulnerabilities along with the control in efficient and systematic manner. The security checklist or guidelines that will be used are from OWASP Top 10 Most Critical Web Application Security Risks (OWASP 2010) and security checklist in developing secure web application in ASP.NET.

On the other hand, the security test result will be gathered from penetration test and automated web vulnerability scanner such as acunetix web vulnerability web scanner and Lens for detecting vulnerabilities in ASP.NET web application. The approach for penetration test is grey box model. It is a hybrid of white and black box model. By using grey box model, the web application source code is given to be analyzed and penetrated using black box method (Simpson 2006). The output in this phase is list of potential vulnerabilities.

After a list of potential vulnerabilities is created, a control analysis will be conducted by looking and analyzing at “E-KTP search” source code. There are two goals of this step. The first one is to identify the controls that were implemented in the current system. The second one is eliminate the vulnerabilities that still exist despite of the implementation of the current control. The second goal can be achieve by executing recommended controls.

Before listing the recommended control for “E-KTP Search” web application, a likelihood determination, impact analysis, and risk level determination will be done first. These three steps are useful to prioritize which recommended control that needs to be executed first by looking at the risk level.

Once the recommended control is implemented, another security test and vulnerability scan will be performed to ensure that all the potential vulnerabilities have been eliminated or mitigated by implementing the recommended control.

3. Results and Analysis

For vulnerability identification, the vulnerability will be categorized into 6 different categories.

- Vulnerabilities in input or data validation
- Vulnerabilities in authentication process
- Vulnerabilities in authorization
- Vulnerabilities in error handling
- Vulnerabilities in working with sensitive data
- Vulnerabilities in cryptography

As an example, this research will emphasize on vulnerabilities that lies in cryptography area. The vulnerabilities are:

- Using own/custom cryptography
- Choosing speed (integrity) for hashing

The details of the vulnerabilities, current control, likelihood level, impact level is shown in Table 1.
Table 1. Example of Vulnerability Assessment in Cryptography

<table>
<thead>
<tr>
<th>Vulnerability</th>
<th>Current Control</th>
<th>Likelihood Level</th>
<th>Impact Level</th>
<th>Risk Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using own/custom cryptography</td>
<td>MD5 hashing</td>
<td>Medium</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>Choosing speed (integrity) for hashing</td>
<td>MD5 hashing</td>
<td>Medium</td>
<td>High</td>
<td>Medium</td>
</tr>
</tbody>
</table>

The current “E-KTP search” web application still stored password in MD5 hashing without adding salt. Moreover since MD5 is fast, the creation of pre-computed hash value will take a short time. MD5 can be easily cracked using rainbow table or online MD5 decrypter such as md5decrypter.co.uk. For showing the weakness of MD5 hashing method, the hashed password that stored in Oracle database will be decrypted using online MD5 decrypter. MD5decrypter.co.uk has a database of 8.7 billion pre-computed hash values (Blandy, 2007). When an attacker inputs a hash value, the MD5decrypter will search its database if it has any matches with the pre-computed hash values, and return the corresponding password based on the hash value that matched. The hashed password that will be used for testing is 5969d9d6a4c2c28906c5207b86e60084 that was taken directly from “E-KTP search” oracle database. The MD5 decryption result for 5969d9d6a4c2c28906c5207b86e60084 is “adminduk2011”. The result can be seen in Figure 1.

The recommended control for hashing method in critical web application is by using SHA256 hashing method plus salt (NIST, 2002). The reason for choosing SHA256 is because it takes a longer time to hash; therefore it takes more resources to create a pre-compute hash value. Adding salt is also useful for increasing the initialization value before hashing (NIST, 2002).

4. Conclusion

The problem in the current “E-KTP search” web application is that the developer did not fully consider the security of the web application. There were not any vulnerability assessment and mitigation during the development phase. Even though the current “E-KTP search” web application already had the capability to mitigate some common vulnerability, such as XSS and SQL injection attack, but there are others vulnerability that considered as critical and has not been mitigated.

The purpose of this research is to explore vulnerabilities that the web application has that were not discovered by the developers. By identifying potential vulnerabilities, current control, likelihood determination, impact analysis, risk determination and control recommendation; it makes the mitigation process more effective and efficient.

The 1st International Conference Integrated Government Academic and Business (ICG-GAB) 2011
Based on the complete research, “E-KTP search” web application has low to medium risk level. The vulnerabilities that considered as medium risk level are oracle padding attack, unencrypted web configuration file, hashing password using MD5, debugging mode still on, and auto-complete password still enabled during authentication.

Finally, the vulnerabilities in the existing application can be mitigated if the recommended control is being implemented. In the complete research, the recommended control that implemented locally such as generic custom error, disabling auto-complete, encrypting web configuration file could mitigated the vulnerabilities related to the controls.

Recommendation
This research was done with the current vulnerabilities found, however in the future it is recommended to keep update on security measures since there is a high possibilities that new vulnerabilities can be found. For now, to enable https in “E-KTP search” web application, a self-signed certificate is still being used. It is also highly recommended to use certificate that was already signed by CA and has been trusted and saved in CA browser list.

Future Works
Since this research is more focused on the technical architecture in “E-KTP search” web application; therefore the research does not cover all the vulnerabilities in OWAPS top ten project, especially for the vulnerabilities like security misconfiguration in network architecture such as failures in firewall rules and intrusion detection system. These problems can be look upon for the future research. The research also did not focus on countermeasure for social engineering attack. This is one factor that can be focused in the future. Because no matter how secure the system is, if it has internal issues such as disagreements like loyalty problems; than the system can be easily breached.

Acknowledgement
We would like to acknowledge BPPT E-KTP team, lead by Husni Fahmi, to provide all the necessary information needed for our research.

References
Anderson, T. How ASP.NET began in Java, 2007
Curphey, M., Scambray, J. & Olson, E. Improving Web Application Security, 2003

The 1st International Conference Integrated Government Academic and Business (ICI-GAB) 2011


Microsoft. ASP.NET Web Page Syntax Overview, 2002

NIST. FIPS 180-2, Secure Hash Standard, 2002


Otegem, M. v. (2007). Interview with Scott Guthrie, creator of ASP.NET.

OWASP. The Ten Most Critical Web Application Security Risks. 2010


Spicciati, J. Deploying a Secure Web Application. SANS Institute Reading Room site, 1-14, 2003
