ASSESSMENT 3

[cosc2539]: security in computing and information technology

Date 17/01/2020

Members:

1. Tao Quang Minh – s3758834
2. Nguyen Kim Ngan Hong – s3757107
3. Tran Duy Long – s3752375

I declare that in submitting all work for this assessment I have read, understood and agree to the content and expectations of the Assessment declaration.

**TABLE OF CONTENTS**

* + - 1. Cross-Site scripting (XSS) 2
      2. Using components with known vulnerabilities 4
      3. Broken access control 6
      4. Sensitive data exposure 8
      5. References 10

#### *Cross-Site scripting (XSS)*

Cross-Site Scripting (XSS) has always been one of the most common and dangerous web application risks. According to OWASP Top 10, XSS is among the top 3 in 2013 and is still among the top 10 security risks in 2017. Nowadays it continues to be practised commonly to perform cyber-attack mostly in newly developing countries.

As stated in OWASP Top 10 - 2017: “XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites”. In simple words, an attacker can commence an attack by sending malicious lines of code (usually <script> code) from client-side into the server and database to manipulate and hijack data by other users’ browsers. Since the attack can be performed simply by data inputs, it is one of the most prevalent cyberattack methods even today. So far, there have been 3 types of cross-site scripting identified: Stored XSS, Reflected XSS and DOM Based XSS.

Reflected XSS (aka Non-Persistent or Type II) occurs when the malicious code is embedded in a URL made by an attacker. The attacker then has to send and convince the victim to access the link so that the code can be sent to the server. The server responds and send back the requested HTML content with the malicious code and execute it on the victim’s browser. This method is mostly used to hijack cookies and impersonate targeted victims.

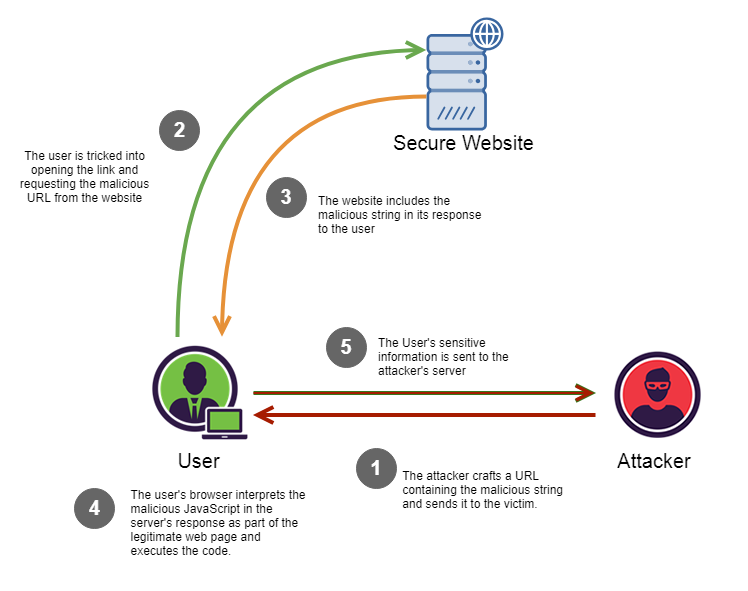


Diagram of Reflected XSS

Stored XSS (aka Persistent or Type I) is an attack form which can be performed by an attacker sending a malicious input to the target server of those such as databases of forums, vlogs, comments, etc. When requested, the targeted server send the HTML content including the malicious code to the clients, infecting their browsers. When sent to a victim’s browser, the <script> codes are then executed, concluding the attack with attacker’s success. The main difference between Stored and Reflected XSS is that the attacker has to somehow convince the target user to access the link for Reflected XSS to work whilst for Stored XSS, the attacker can store the code into the server and wait for the unfortunate victims. Another difference relating active sessions as the goal of attacker can be more easily achieved when the victim’s web session is operational. For Reflected XSS, the attacker needs to trick the user to log in to activate their session and access the link for code execution. For Stored XSS, since the code is stored in the server-side, there is a higher chance that the infected victims are still in session as they are using the web application. Because of that, Stored XSS is significantly more effective and dangerous than Reflected XSS.

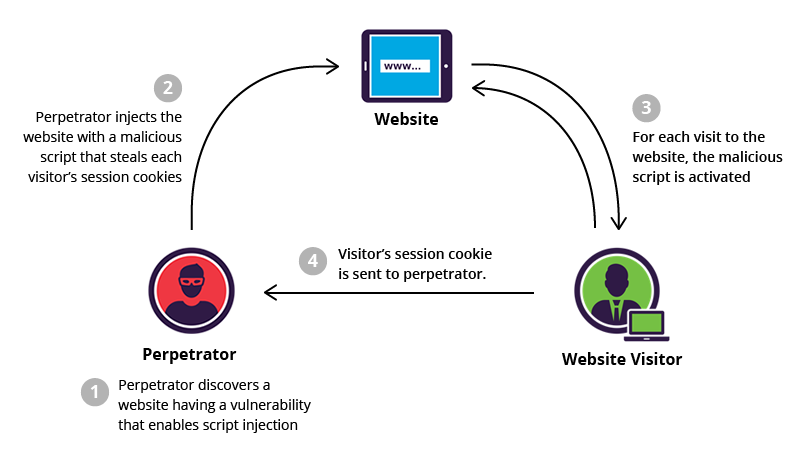


Diagram of Stored XSS

Another form of attack, which was defined by Amit Klein in 2005, is DOM Based XSS (Type-0 XSS). By definition, “DOM Based XSS is an XSS attack wherein the attack payload is executed as a result of modifying the DOM “environment” in the victim’s browser used by the original client side script, so that the client side code runs in an “unexpected” manner” (Owasp.org, 2015). This form of attack can be done by modifying a form and embed into it the malicious code which will execute when the user tries to input into the form.

The impact of the Cross-Site Scripting varies from moderate to severe depends on what type of attack is commenced. Out of all 3 types, Stored XSS attack poses the highest threat and causes the most severity as the attack can uncontrollably infect many users over time. For Reflected and DOM Based XSS, the severity is mostly moderate since the users are targeted, limiting the number of victims. According to Dionach, most attack vectors are to take over the  victim’s session by obtaining session cookies. Other vectors may aim to steal credentials and sensitive data or by drive-by downloads to take over a client computer of an organization. If succeed, an organization may have to cope with catastrophic consequences due to falsified information and documents, industrial espionage, stolen confidential which can put the organization out of business.

Although Cross-Site Scripting has moved down in ranking, the risk of being attacked is always high and growing. According to Akamai Security Report, XSS accounted for 10% of all attacks on web applications in the first quarter of 2017 and this number is rising. Furthermore, it was also reported by Akamai that the number of XSS attack vectors was the 3rd highest in 2019. More specifically, there are about 500 thousand to more than 1.5 million of XSS attacks was reported daily in 2019. With such a rising number, developing nations like Vietnam is not spared from the attacks as the country is premature in technology and cybersecurity. It is claimed in 2019 by VnExpress, 60% of Vietnamese websites are vulnerable to cyberattacks and there were more than 3 thousand attacks reported. Most of the attacks only targeted specific users. However, there is a high risk that an XSS attack can take down an important system like administrative websites and banking systems. Henceforth, countermeasures are needed to defense applications from being attacked.

Because of widespread prevalence and damaging impacts of XSS, solutions have been devised to cope with the risks. Mentioned by OWASP, the most common strategy is to separate untrusted data (data inputs from clients) from active browser content. That can be done by implementing frameworks that are able to escape XSS automatically by design, such as Ruby and ReactJS. The strategy can also be performed by escaping untrusted HTTP request depends on their outputs and enforcing context-sensitivity when modifying browser documents on client-side, which protects from DOM Based XSS. Last but not least, CSP should be implemented for mitigating control against XSS.

#### *Using components with known vulnerabilities*

Security risk posed by components with known vulnerabilities ranked among the top 10 of security risks in 2017 according to OWASP Top 10 - 2017. The chance of this risk is consistently high and ranked 9th in 2013 and maintained its place ever since. Furthermore, the exploitation of this fault can cause catastrophic consequences to web applications.

This security risks are caused by the components that have already-written vulnerabilities which can be exploited and attacked. The vulnerabilities can be from any components such as web server, database, libraries, plugins, any components that are between the core and the operating system. Because of that, vulnerabilities are difficult to detect, making the risk one of the most prevalent. This risk is also made more common since most developers concern more about their lines of codes and neglect the ones that are imported from other sources, making it vulnerable to attack. Besides that, this issue will likely arise if the components’ versions are not acknowledged or left outdated. Hence, scanning should be performed in order to identify vulnerabilities before they are exploited. On some occasions, security misconfiguration (rank 6 - Top 10 OWASP - 2017) also causes a higher risk of this issue as the components are not properly configured. As stated on Detectify, when a vulnerability is exploited, it will likely be published on the internet and a payload of ready-to-use hacking tools follows. Moreover, there are documentations about vulnerabilities of most components that are easy to access, leading to hackers following the instructions to devise a custom exploitation on the web application. However, the limitation of that is if no information about vulnerabilities of a specific component is published, it will be difficult for the attackers to improvise.

The risk of being attacked by exploiting components’ vulnerabilities is always high and has been maintaining its rank 9 in OWASP top 10 since 2013. More concerning, the chance of websites getting attacked by this method is only rising because 96% of applications make at least some use open-source components (Immuniweb.com, 2018). In developing countries such as Vietnam, this number has a significant effect on web applications since most local programmers tend to use already-written components rather than coding from scratch to ensure the speed of delivery, resulting in a great number of vulnerabilities are embedded into the applications. Another information stated by Gartner, 99% of the security vulnerabilities will have been identified in 2020, posing a threat to almost any existing components and it will remain so until the components are patched. Since most of the security flaws have been written down, developing country like Vietnam is extremely vulnerable to cyber-attack due to lack of experience. A proof to that is a report from VNA claiming that more than 10 thousand cyber-security vulnerabilities were found in the government agencies itself alone and the number is predicted to increase greatly. This is a grave danger to national security as hackers can exploit the flaw of components and sabotage the systems. If those systems are left vulnerable, the country can be at stake of terrorism. For other private organizations, a seem-to-be unimportant security flaw can be exploited for hackers to hijack the web application to impersonate personnel and steal confidential documents, which are acts of industrial espionage. With that few tasks, an organization can be put out of business. For that reason, Vietnamese developers must broaden their knowledge and be more cautious when designing a new web application.

Since security risk posed by components with known vulnerabilities has always been in the top 10 concerns since 2013, many countermeasures have been devised to cope with the issue. According to OWASP, the leading solution to this is patch management process. The process is to check and discard any unnecessary dependencies, components to lower the risk of vulnerabilities. Another task of the process is to keep the components up to date to ensure as many flaws as possible is patched and only obtain components from official and trusted source. Lastly, sources such CVE and NVD should be followed to receive vulnerability reports.

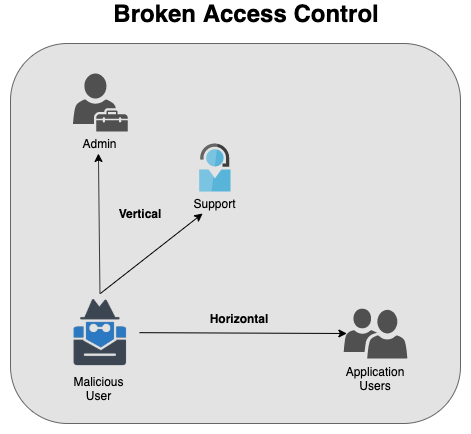
#### *Broken access control*

Broken access control is one of the top 10 OWASP  Web Application Security Risks which are 2 out of 10 significant vulnerabilities from OWASP Top 10 2013 list. In fact, Broken access control is originally a combination of Insecure Direct Object References and Missing Function Level Access Control. In 2017,  it is ranked top 5 among the top 10 common security risks by OWASP.

Access control is the way that a web application enforces policies that manage the users not to access to contents and functions which are not under their intended permission. The failure of application access policies is mainly a consequence of lack of effective functional testing by application developers, resulting in flaws that deny to legitimate users and allow the hackers to compromise the user's roles outside of application intended permissions (Contrastsecurity.com, n.d.). A broken access control can lead to unauthorized information disclosure, data modification and deconstruction, and performing business functions outside of the users’ limits. In other words, a broken access control is when no restriction is properly applied to authenticated users’ access rights. Broken access control can be exploited by any sophisticated attacks even with the simplest ones. Attackers can exploit this vulnerability to gain access to unauthorized functions such as viewing sensitive data, accessing and modifying other users’ accounts, or changing access privileges.

There are 2 common types of access control vulnerabilities which are horizontal privilege escalation and vertical privilege escalation:

* **Horizontal privilege escalation:** a horizontal privilege escalation access control failure is when a user can perform an action or gain access to other users’ account with the same level of permission (Packetlabs, 2019).
* **Vertical privilege escalation:**  a vertical privilege escalation access control failure is when a user can perform an action or gain access to other user’s account whose has higher access role (Packetlabs, 2019).



Broken access control diagram

In particular, as understanding of how a certain piece of data is used within the web app is the fundamental requirement for access control detection, access control risks are not sufficiently addressed by any dynamic vulnerability scanning or static source-code review technologies. Meanwhile, it is more effective to detect missing or broken access control by manual testing.

Maintaining within one of the most serious security risks ranked by OWASP Top 10, exploited broken access control delivers significant impacts to business and organization. In terms of business, the consequence of devastating. Specifically, hackers can act as application users or administrators to create, modify access rights, update and delete data records or even add external access. In the worst case, attackers can fully gain control of the organizations’ system. In fact, almost every worldwide web application such as Twitter, Facebook or Vimeo are vulnerable and Vietnamese website is not an exception. It is reported by VnExpress that 60% of Vietnamese website is easily attacked by hackers and the most common case is broken access control issues. VSEC has reported that on 15 October 2019, a security risk was found on CyberoamOS platform, allowing intruders to access to Cyberoam firewall to take over the administrative control of a business system. Hence, the hackers can discover and review all transaction records and steal the organization private information (An, 2019).

Due to the flaw’s significant consequence, there are some solutions which are recommended to prevent broken access control in a system. According to OWASP, it is said that access control is only effective in trusted server-side code or server-less API, where the access control check or metadata cannot be modified by the attackers (Owasp.org, n.d.). Therefore, access control mechanism should be implemented once and can be reused throughout the application, including minimal the CORS usage. Secondly, model access controls should enforce record ownership instead of allowing users to create, read, update or delete data. Thirdly, web server directory listing should be disabled and ensure that backup files are not present within web roots. In order to detect the access control is not secured in time, there should be alert to the administrator when log access failed for multiple times. Lastly, JSON Web Token (JWT) should be invalidated after the user logged out from the system to prevent external access.

#### *Sensitive data exposure*

Sensitive data exposure ranked top 3 among 10 common security topics by OWASP Top 10 - 2017. In fact, data exposure has been responsible for most of the impactful data breaches in the past few years. For instance, the catastrophic Marriott breach which result in over 300 million customer private records were stolen, and another 150 million when Equifax attacks (Insights: Secure Code Warrior, 2019). Sensitive data exposure normally happens whenever information that only meant for  authorized viewing is exposed to unauthorized users. In other words, sensitive data exposure occurs when an application or program that does not adequately protect secured data such as user account/password, session token, banking information or personal information, etc (Hdivsecurity.com, n.d.).

Sensitive data exposure can be caused by application lack of adequately protect sensitive information from being disclosed to hackers. In most cases of exposure of data, hackers take advantage of  inadequate security and unencrypted data stored, transmitted or processed. As it is stated by OWASP: Instead of breaking crypto directly, attackers steal keys, do man-in-the-middle attacks, or steal clear text data off the server, while in transit, or from the user’s browser, in fact, sensitive data exposure can be caused by both internal and external factors. Firstly, a person who has access to an organization’s information and the permission to control the internal system poses more danger compared to any outsider. Secondly, weak encryption is also a main factor leading to data exposure. For example, cloud storing is an ideal option for convenient storage of the company database. However, if the service is not properly implemented high secure password hashing algorithm, it provides an open platform for attacks. Lastly, accidentally clicking on suspicious links or visiting untruthful website or downloading of malicious malware can also cause malignant attacks (Telecom, 2016).

It is the fact that any security attack causes a vital impact to the victim company, also for sensitive data exposure case. In terms of business, sensitive data exposure leaves a considerable amount of financial cost to the company such as comprising the cost of security remediation, cost of victims notification and support, cost of regulatory fines, and the cost of legal actions against the company. Moreover, it significantly affects the company's reputation and truthfulness. In Vietnam, it is also a common phenomenon of a business being attacked by hackers. It is reported by a Vietnamese newspaper that private information of an organization called Concung is leak and published on the internet by an anonymous hacker. This rises an afraidness that while the employee’s information is leaked, the hacker can also publish all of the Concung’s customer personal information (Chanh, 2018).

Remaining the top 3 in OWASP Top 10 List, sensitive data exposure brings significant impact to a business that not only affects the business’s finances and also its reputation. Thus, OWASP has provided some strategies to prevent this kind of security risk. The most basic strategy is to consult the reference truthfulness before clicking or using it. Secondly, it is necessary to clarify processed data, stored or transmitted data by an application. Then, data should be categorized to which data is sensitive based on privacy laws, regulatory requirements, or business needs. After the classification, access controls should be applied. In order to prevent the data from being stolen, data should not be stored unnecessarily and ensure that sensitive data is encrypted. All the sensitive data is encrypted in transit by using secure protocols such as TLS with perfect toward secrecy (PTS). Also, modern and strong standard algorithms, protocols, and keys are in place and only use proper key management. Thirdly, all caching for response that contains sensitive data should be disabed. Lastly, password should be stored in a hashed format by using strong adaptive and complex algorithm ( salted hashing function with a work factor) such as Argon2, scrypt, bcrypt or PBKDF2. Also, the effectiveness of configurations and settings should be verified independently (Owasp.org, n.d.).

#### *References*

Owasp.org. (2017). *Types of Cross-Site Scripting - OWASP*. [online] Available at: [https://www.owasp.org/index.php/Types\_of\_Cross-Site\_Scripting#Stored\_XSS\_.28AKA\_Persistent\_or\_Type\_I.29](https://www.owasp.org/index.php/Types_of_Cross-Site_Scripting%23Stored_XSS_.28AKA_Persistent_or_Type_I.29%20) [Accessed 12 Jan. 2020].

Marius (2016). *The Real Impact of Cross-Site Scripting*. [online] Dionach. Available at: <https://www.dionach.com/blog/the-real-impact-of-cross-site-scripting> [Accessed 12 Jan. 2020].

Immuniweb.com. (2018). *Using Components with Known Vulnerabilities Security Vulnerability | OWASP Top 10 | Exploits and Solutions*. [online] Available at: [https://www.immuniweb.com/blog/OWASP-using-components-with-known-vulnerabilities.html](https://www.immuniweb.com/blog/OWASP-using-components-with-known-vulnerabilities.html%20) [Accessed 12 Jan. 2020].

Detectify Blog. (2016). *OWASP TOP 10: Using Components with Known Vulnerabilities | Detectify Blog*. [online] Available at: [https://blog.detectify.com/2016/08/04/owasp-top-10-using-components-with-known-vulnerabilities-9/](https://blog.detectify.com/2016/08/04/owasp-top-10-using-components-with-known-vulnerabilities-9/%20) [Accessed 12 Jan. 2020].

VietnamPlus. (2019). *Over 10,000 cyber-security vulnerabilities found in government agencies | Sci-Tech | Vietnam+ (VietnamPlus)*. [online] Available at: [https://en.vietnamplus.vn/over-10000-cybersecurity-vulnerabilities-found-in-government-agencies/161142.vnp](https://en.vietnamplus.vn/over-10000-cybersecurity-vulnerabilities-found-in-government-agencies/161142.vnp%20) [Accessed 13 Jan. 2020].

Contrastsecurity.com. (n.d.). Broken Access Control. [online] Available at: [https://www.contrastsecurity.com/knowledge-hub/glossary/broken-access-control](https://www.contrastsecurity.com/knowledge-hub/glossary/broken-access-control%20) [Accessed 13 Jan. 2020].

Packetlabs. (2019). Broken Access Control: Hidden Exposure for Sensitive Data | Packetlabs. [online] Available at: [https://www.packetlabs.net/broken-access-control/](https://www.packetlabs.net/broken-access-control/%20) [Accessed 13 Jan. 2020].

An, C. (2019). 60% Vietnamese website is easily attacked by hackers. [online] VnExpress. Available at: [https://vnexpress.net/so-hoa/60-website-tai-viet-nam-de-bi-hacker-tan-cong-3997951.html](https://vnexpress.net/so-hoa/60-website-tai-viet-nam-de-bi-hacker-tan-cong-3997951.html%20) [Accessed 13 Jan. 2020].

Owasp.org. (n.d.). A5-Broken Access Control. [online] Available at: <https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A5-Broken_Access_Control> [Accessed 13 Jan. 2020].

Hdivsecurity.com. (n.d.). Sensitive Data Exposure | Hdiv Documentation. [online] Available at: [https://hdivsecurity.com/docs/sensitive-data-exposure/](https://hdivsecurity.com/docs/sensitive-data-exposure/%20) [Accessed 13 Jan. 2020].

Insights: Secure Code Warrior. (2019). Coders Conquer Security: Share & Learn Series – Sensitive Data Exposure. [online] Available at: [https://insights.securecodewarrior.com/coders-conquer-security-share-learn-series-sensitive-data-exposure/](https://insights.securecodewarrior.com/coders-conquer-security-share-learn-series-sensitive-data-exposure/%20) [Accessed 13 Jan. 2020].

Telecom, B. (2016). Sensitive Data Exposure Vulnerability: Causes and Prevention. [online] Medium. Available at: [https://medium.com/@BreezeTelecom/sensitive-data-exposure-vulnerability-causes-and-prevention-4c86a19df70d](https://medium.com/@BreezeTelecom/sensitive-data-exposure-vulnerability-causes-and-prevention-4c86a19df70d%20) [Accessed 13 Jan. 2020].

Chanh, T. (2018). Concung employee's information is leaked on the internet. [online] https://nld.com.vn. Available at: [https://nld.com.vn/cong-nghe/lo-du-lieu-ca-nhan-hang-ngan-nhan-vien-con-cung-tren-mang-201811101554112.htm](https://nld.com.vn/cong-nghe/lo-du-lieu-ca-nhan-hang-ngan-nhan-vien-con-cung-tren-mang-201811101554112.htm%20) [Accessed 13 Jan. 2020].

Owasp.org. (n.d.). A3-Sensitive Data Exposure. [online] Available at: [https://owasp.org/www-project-top-ten/OWASP\_Top\_Ten\_2017/Top\_10-2017\_A3-Sensitive\_Data\_Exposure](https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A3-Sensitive_Data_Exposure%20) [Accessed 13 Jan. 2020].