Chad Ballay

Cybr 450-342

Week 6

1/11/2021

**CYBR 450 – Application Security Assignment**

NAME: \_\_CHAD BALLAY\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_\_\_01/10/2021\_

a) Provide an example of a Cross Site Scripting (XSS) attack. Include an explanation of the differences between stored and reflective XSS attacks.

One of the best things to have experienced while here at Bellevue was a previous class using WebGoat from OWASP. (<https://owasp.org/www-project-webgoat/>) This hands on simulator allows you to see a XSS attack in detail to understand conceptually how it works and hopefully how to prevent it. Here is one of the many publicly shared walkthroughs on that specific part of the simulator. (<https://www.youtube.com/watch?v=u6jZBK5oS6c>)

The broad answer is that an attacker somehow includes malicious scripts into the benign content that a user is being served. (<https://en.wikipedia.org/wiki/Cross-site_scripting>)

b) Provide an example of a SQL Injection attack. Why does the back end database make a difference in how it works? What type of information or access could be compromised with SQL injection attacks? Include at least one additional reference.

Yet another scenario handled by WebGoat if you wish to get hands on experience to help learn what SQLi’s are. Roughly it’s giving additional SQL commands on top of the expected content you are supplying as criteria for a query. Much of this becomes a problem when dynamic sql queries are programmatically allowed. Depending on the situation anything up to root access can be garnered through these types of attacks. Generally, though you are mostly able to run arbitrary SQL as the user that was used to run the initial SQL query you are exploiting. But when combined with other attacks this access will often get escalated into having higher privileges. I love the writeups people do for retired HackTheBox scenarios since you can often follow along with them. (<https://0xrick.github.io/hack-the-box/jarvis/>)

c) Watch this video on “Ethical Hacking - How Buffer Overflow Attacks Work”: <https://www.youtube.com/watch?v=iZTilLGAcFQ>. In your own words, explain how Buffer Overflow attacks work. Include at least one image or drawing. Include at least one additional reference.

Ever try to put 2lbs of something in a 1lb container. After a 1lb, physics stops you from pushing too far past that limit before either compressing the product, breaking the container, or leaking the product all over the place. That analogy can work for when we start talking about memory for applications. But this time we can’t compress the data most of the time. (Weak spot in the analogy.) Depending on the language or the OS we can’t destroy the container and instead a runtime error is thrown. What usually happens is that the excess spills over and starts putting what should be data into the code space which it could then become executed instructions for the application. I can’t find a specific single image that best conveys the concept but I’ve found a set of them in an article that does an amazing task at visualizing the concept. (<https://netsec.ws/?p=180>). This website does a phenomenal job at showing the overall concept.

Graphical user interface, application

Description automatically generated

d) Both the SANS (<http://www.sans.org/top25-software-errors/>) and OWASP (<https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project>) sites provide techniques for preventing application attacks. Discuss three of these techniques. Be sure to cite any sources used.

Improper use of Hardcoded Credentials (<https://cwe.mitre.org/data/definitions/798.html>)

Mitigation 1 – Create a keystore for all credentials and harden it. Then require applications to securely authenticate to it for authorization. This way the perimeter is decreased on where to focus security efforts.

Mitigation 2 – Automated Static Analysis of Source Code. Connection strings, credentials, keyword scanning can be identified through use of source code vulnerability tools. Several solutions exist and each attack the problems through different mechanisms. (<https://samate.nist.gov/index.php/Source_Code_Security_Analyzers.html>)

Mitigation 3 – Black Box testing of default credentials. Due to many out of the box installations coming enabled with standard credentials, a comprehensive scan and methodical exhaustive testing using these is recommended. Effectiveness is moderate but it will help identify the most egregious of holes. (<https://github.com/ztgrace/changeme>)

e) What part of the e-mail process does SSL/TLS usually secure? Is this end-to-end security? Explain. What standards and technologies provide end-to-end security for email?

SSL/TLS encrypts the connection from one computer to another computer so this would be the sending of your email as it routes from hop to hop. It is not an end-to-end security solution. Upon sending it the computer would have had access to the content and the receiving computer for this hop will as well. For true end-to-end encryption there are 2 main answers. S/MIME and PGP. (<https://techdifferences.com/difference-between-pgp-and-s-mime.html>)

Table

Description automatically generated