Programming language concepts

Read Larson (2018) and Weidman (n.d.) then answer the questions below, adding them as evidence to your **e-portfolio**. You may want to complete this activity in conjunction with or after completing Seminar 2 preparation.

1. What is ReDOS and what part do ‘Evil Regex’ play?
2. What are the common problems associated with the use of regex? How can these be mitigated?
3. How and why could regex be used as part of a security solution?

1. The Regular expression Denial of Service (ReDoS) is a Denial-of-Service attack (DoS). A DoS attack cripples a resource (website, application, server) and makes it inaccessible for use. Attackers can, for example, cause service inaccessibility by manipulating network packets, programming, logical vulnerabilities or resource handling. Another example is when a service receives a large number of requests, this can lead to a shutdown of the use of the service for the users.

A ReDoS attack leads to a reduction in speed due to the implementation of a regular expression. The input size of the regular expression plays a significant role in the speed. The more possible paths, the exponentially slower the service becomes.

Evil Regex is a regex pattern which can get stuck on manipulated input. The attacker injects the evil Regex in order to make the system vulnerable. Evil Regex contains a group with repetitions. Examples:

- (a+)+
- ([a-zA-Z]+)*

- (a|aa)+
- (a|a?)+
- (.*a){x} for x \> 10

By entering the input aaaaaaaaaaaaaaaaaaaaaaaa! The system is vulnerable and can get stuck with the Regex above (Weidman, 2022).

2. A common problem with Regex is that it can lead to errors. For example, unbalanced brackets ( ) can lead to problems, while unbalanced [ ] is considered stable. Furthermore, some symbols have different meanings, which can confuse developers. For example, the ^ symbol can be the beginning of a string, a negated character set or a dependency on an element related to the regular expression. The most important aspect is that the regular expression is used correctly, as Regex can accept strings which should be refused. Moreover, the program can cause a crash or limit if an incorrect expression is accepted. The worst-case scenario is a server crash caused by a regex, e.g. a web form where a regex incorrectly used data validation.

Mitigation can be a checker which examines the Regex before running the entire program. Larson, E. (2018) elaborated on 11 automated checkers of Regex to reduce errors by using Regex.

1. Bad Range Checker
2. Separator in Character Set Checker
3. Duplicate Character Checker
4. Lone Brace in Character Set Checker

5. Optional Brace Checker
6. Duplicate Punctuation Only Character Set Checker
7. Wildcard Next to Punctuation Checker
8. Repeat Punctuation Checker
9. Digit Too Optional Checker
10. Anchor in the Middle Checker
11. Consistent Anchor Usage Checker (Larson, 2018)

3. With the ability to describe the pattern to be matched or define filters, Regex is used in cybersecurity to search logs and other large data files. Cybersecurity professionals must use technology like Regex to identify potential evidence in a file. For example, regex experts use keyword-searching techniques to speed up the process of a forensics investigation. Doing this allows them to locate the files that include the potential evidence. Moreover, Data Scientists can edit input data files through Regex to be used for machine learning. Furthermore, developers can validate input fields with the help of Regex (Sauver, 2022; Sharma & Nagpal, 2020: 339).

References:

Larson, E. (2018) Automatic Checking of Regular Expressions. *2018 IEEE 18th International Working Conference on Source Code Analysis and Manipulation (SCAM)* 225-234. DOI:10.1109/SCAM.2018.00034

Sauver, J. S. (2022) What's a Regular Expression?. Available from: https://www.farsightsecurity.com/blog/txt-record/regexp-

20200804/#:~:text=Regexes%20are%20routinely%20used%20in,input%20fields%20and%20so%20on [Accessed 12 October 2022].

Sharma, P. & Nagpal, B. (2020) Regex: an experimental approach for searching in cyber forensic. *Int. j. inf. tecnol.,* 12(2): 339-343. DOI: https://doi.org/10.1007/s41870-019-00401-y

Weidman, A. (2022) *Regular expression Denial of Service - ReDoS Author: Adar Weidman.* Available from: https://owasp.org/www-community/attacks/Regular_expression_Denial_of_Service_-_ReDoS [Accessed 11 October 2022].