Open Web Application Security Project (OWASP) is an organisation that focuses on creating and maintaining a list of the most critical software security vulnerability risks (Mirkovic et al., 2019). This is achieved by analysing data from both individual and organisational contributors. Contributors provide Common Weakness Enumerations (CWEs) detailing the total applications and time frames as defined by MITRE. The CWEs mapped to broken access control had most occurrences in the 2021(Chen et al., 2021).

There are a number of factors which can lead to a broken access control threat:

- Breaking least privilege rule (Mell and Grance, 2011) – This means giving the user access to a greater set of resources / actions than they are required to have access privileges to, based on their permission level.
- Changing elements of the URL sent to the server (Liu and Li, 2015) – By manipulating the URL sent to the server, attackers can potentially bypass access controls if necessary controls are not in place to detect and/or prevent this.
- Neglecting the use of well-established and up-to-date software development kits (SDKs) and application programming interfaces (APIs) (Chen et al., 2021) – In these cases, the application will likely contain outdated code with known vulnerabilities which can be exploited by attackers to obtain access to privileged resources.
- Lack of multifactor authentication (MFA) in the software (Alzahrani and Jiang, 2018) – A lack of MFA makes it easier for attackers to gain unauthorized access if one or more passwords are compromised.

- Recycling of passwords across multiple applications (Kwon and Johnson, 2014) – Coupled with a lack of MFA, this can mean that a data leak in one application can provide attackers with unauthorized access to one or more other systems.
- Generalized grouping interfering with the least privilege (Kaur and Bala, 2017) – Grouping users based on their role or department rather than based on their specific job duties or tasks can result in users having access to resources they do not need, which increases the risk of unauthorized access.

Figure 1 is an activity diagram that illustrates key points above, demonstrating that without proper controls, an attacker can gain privileged access to a system.

There are various ways to counter the threat of broken access control as detailed below:

- Designing and developing robust systems (Kizza, 2014).
- Incorporate multifactor or biometric authentication (Alzahrani and Jiang, 2018).
- Implement input validation and ensuring the output is correctly encoded (Li et al., 2017).
- Regular audit and training (Krutz and Vines, 2010).
- Incorporate error handling mechanisms (Chen et al., 2021).

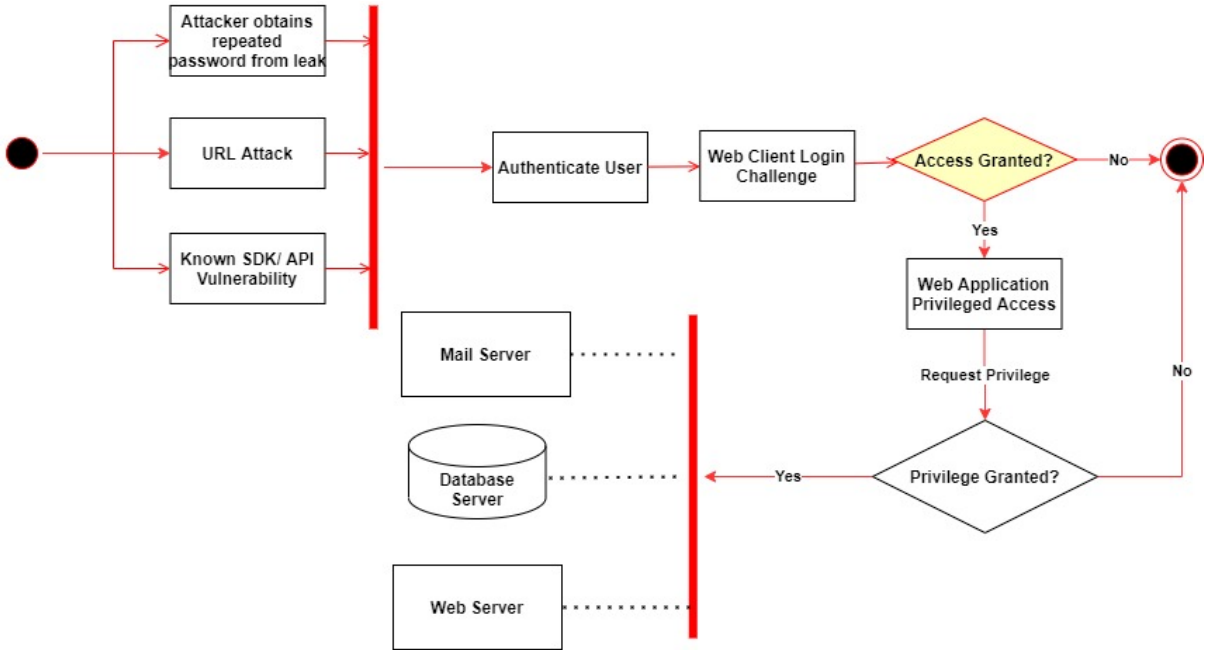
Attacker obtains repeated password from leak
URL Attack
Known SDK/ API Vulnerability
Authenticate User
Web Client Login Challenge
Access Granted?
No
Yes
Web Application Privileged Access
Request Privilege
Privilege Granted?
No
Yes
Mail Server
Database Server
Web Server

Figure 1: Broken access controls resulting in privileged access provided to an attacker (Activity Diagram)

Figure 2 is an update to Figure 1 showing how the incorporation of a 2-factor authentication scheme can be highly instrumental in mitigating a range of attacks mentioned previously.

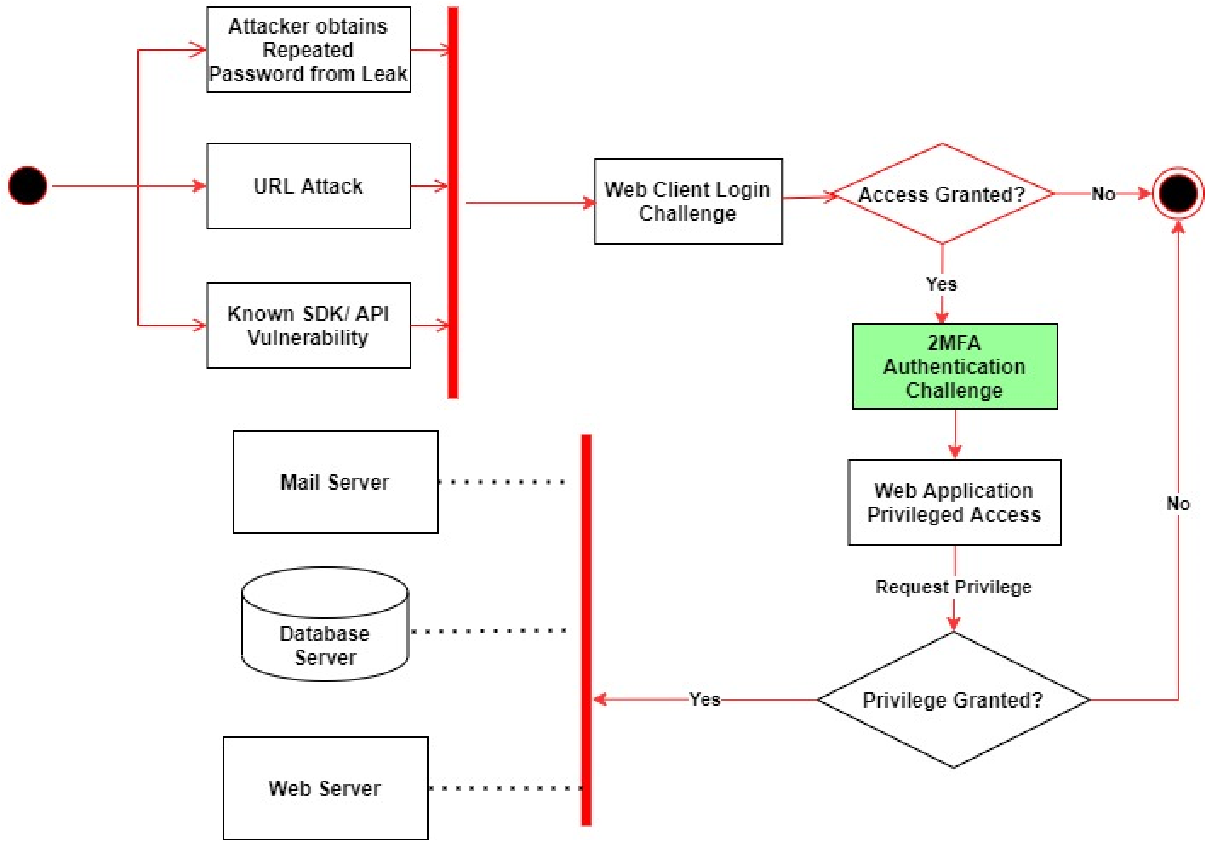
Attacker obtains Repeated Password from Leak
URL Attack
Known SDK/ API Vulnerability
Web Client Login Challenge
Access Granted?
No
Yes
2MFA Authentication Challenge
Web Application Privileged Access
Request Privilege
Privilege Granted?
No
Yes
Mail Server
Database Server
Web Server

Figure 2: Two-factor authentication scheme helps mitigate a range of attacks (Activity Diagram)

Below-par coding practices and improper validation often lead to application-level vulnerabilities that provide elevated access to attackers. Improved access control security is paramount to protecting cloud-based infrastructure and assets.

## References

Alzahrani, A., & Jiang, Z. (2018). An analysis of factors affecting multi-factor authentication adoption. Information & Computer Security, 26(2), 193-212.

Chen, H., Wang, L., Li, Y., Li, C., & Liu, J. (2021). A Hybrid Approach for Identifying Access Control Vulnerabilities in Web Applications. IEEE Access, 9, 102934-102947.

Kaur, S., & Bala, A. (2017). A comprehensive study of access control in cloud computing environments. Journal of Ambient Intelligence and Humanized Computing, 8(3), 341-359.

Kizza, J. M. (2014). Ethical and social issues in the information age. Springer.

Krutz, R. L., & Vines, R. D. (2010). Cloud security: a comprehensive guide to secure cloud computing. Wiley Publishing.

Kwon, O., & Johnson, M. E. (2014). Preventing password reuse across multiple online accounts. IEEE Transactions on Dependable and Secure Computing, 11(5), 435-447.

Li, C., Xie, S., Zhang, X., & Lu, K. (2017). Characterizing and detecting input validation vulnerabilities in web applications. IEEE Transactions on Software Engineering, 44(10), 970-984.

Liu, J., & Li, M. (2015). An analysis of security vulnerabilities in web applications. Journal of software, 10(3), 308-319.

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology, 53(6), 50.

Mirkovic, J., Zhang, K., Amann, J., Javed, M. F., Karim, T., & Bhattacharya, P. (2019). Automated Web Security Testing: A Recipe-based Approach. IEEE Transactions on Reliability, 68(2), 743-764.