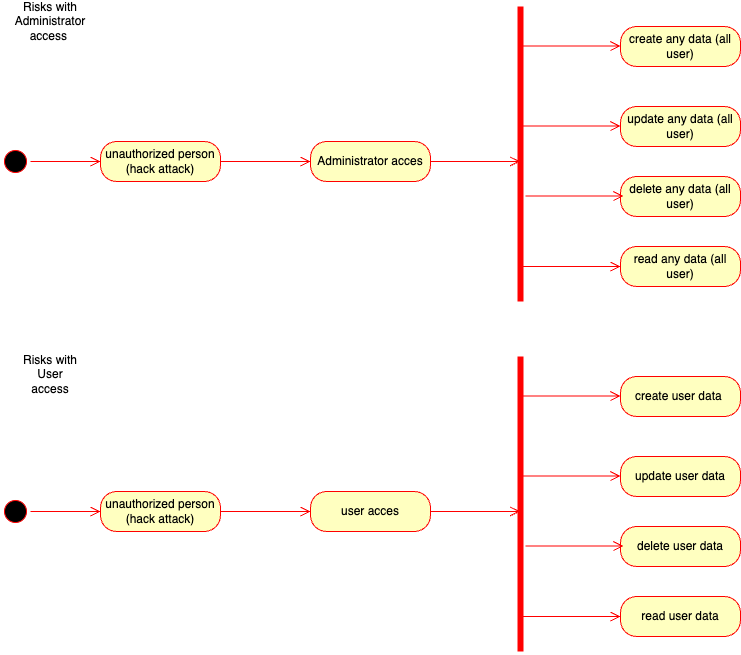
**Broken Access Control (Initial Post) by Gianluca Cannone**

Broken Access Control is now in first place and rightly so in my opinion (Fig. 1).

  
  
  
Figure 1: What's changed in the Top 10 for 2021 (OWASP, 2021a)  
  
Patchy access control is often a problem with applications that have grown in size over their version history. Instead of consciously designing access control schemes from the beginning, access control has been added and extended in an unstructured way over time. In cases where access control is not centralised but distributed in various places in the code, this often leads to poor manageability and complexity that is difficult to understand.  
  
A particularly dangerous form of Broken Access Control are interfaces through which administrators can manage a site via the internet. Because of their power, these interfaces are often targets for attacks. All known servers and web application environments are vulnerable to this type of problem. Even if a site is completely static, it is vulnerable to attack unless properly configured (Köhntopp, 2021) (OWASP, 2021b)  
  
The following activity diagram demonstrate a simple example what for risks can be occured if an unauthorized person have either the administrator access or user access.

  
  
  
Figure 2: Activity diagram User vs Administrator Power

The worst case is if the unauthorised person steals and deletes the users' data. For example, with the help of administrator access, the hacker can manipulate all user data. On the contrary, the hacker can only manipulate one user's data with user access. Therefore developer must secure the administrator access at the very highest level.

Avoidance

Almost all sites have access control requirements. Therefore, an access control policy should be documented and policies enforced for implementation. Access control code should be well structured, modular and centralised. Penetration testing can help determine if broken access control issues exist. Specialists should check any remote administrative access interface particularly carefully to ensure that only authorised persons have access according to their different roles (Onlinesolutionsgroup, 2022).

References:

Köhntopp, K. (2021) A01:2021 - Broken Access Control. Available from: https://blog.koehntopp.info/2021/11/16/a01-2021-broken-access-control.html [Accessed 27 September 2022].

Onlinesolutionsgroup (2022) Broken Access Control. Available from: https://www.onlinesolutionsgroup.de/blog/glossar/b/broken-access-control/ [Accessed 27 September 2022].

OWASP (2021a) OWASP Top 10 - 2021.Available from: https://owasp.org/Top10/ [Accessed 27 September 2022].

OWASP (2021b) OWASP Top 10 - 2021. Available from: https://owasp.org/Top10/A01\_2021-Broken\_Access\_Control/ [Accessed 27 September 2022].

**Replay by Mustafa Sibai:**

Thank you so much for the post. That was interesting to read. I think by default all users should have their default access to any page on a web application to be denied unless stated otherwise. On top of that adding a monitoring and alerting system and logging and flagging who accessed what and when will defiantly help figure out what is going on in the system. Enforcing 2FA and IP whitelisting will defiantly help protect sensitive data being accessed from outside the network.

A lot of web frameworks help you set all of these up. According to Django documentation, you are easily able to set permissions for users across the entire application. Furthermore, you are able to add group permissions which is excellent!

Automated testing can help figure out whether access levels and permissions have become broken for particular pages or not. Django allows developers to create their own unit testing. Running these unit tests every single build and tracking whether they succeeded or not is an essential part of software development and should not be skipped, no matter what. Creating secure software that is late for delivery is better than creating a none secure software on time.

In the end, no software is ever fully secure, and the battle between white and black hackers will always exist. However, as developers, we must try our best to write secure software and stay up to date with industry trends and news.

References

https://docs.djangoproject.com/en/4.1/topics/auth/default/

https://docs.djangoproject.com/en/4.1/topics/testing/overview/