Veilig programmeren begint met goed beheren.

Je beheert de toegang tot objecten, die het waard zijn te beveiligen.

Voorbeeld:

Toegang tot een lokaal:

* De student heeft geen toegang
* De docent heeft toegang
* De gebouwenbeheerder geeft toegang aan gebruikers
  + Het beheer gaat via rollen
  + Dus een gebruiker heeft een rol en via deze rol krijgt hij of zij toegang
  + Toegangsrollen zijn meestal stapelend, d.w.z. dat de gebouwenbeheerder heeft ook toegang tot de lokalen. Militairen: laagste rang is soldaat, sergeant is meer dan soldaat dus hij kan hetzelfde doen als dec soldaat + de sergeants toegangen. Luitenant is meer dan sergeant dus kan wat de soldaat kan, wat de sergeant kan en wat de luitenant extra kan
  + Standaard heb je geen toegang. Toegang moet je verleend worden.
  + Er is een rol die andere rollen toegang verleend: hoogste recht(grant) : eigenaar.

File rechten:

* Degene die de file maakt is eigenaar.
* De admin van het operating systeem(root) is van alle files eigenaar.
* Standaard heb je geen recht tot een file.
* Bij file systemen heb je een speciale groep: iedereen.
* De rechten:
  + X : execute: opstarten van een file , mits de file executable is: exe,bat,com,php,html.
  + R: read: leest de inhoud van een file.
  + W:write: maak de file of maak de file leeg als hij bestaat en vul deze.
  + A: append: aan een bestaande file, aan het einde toevoegen: logfiles
  + U: update: inhoud van een bestaande file wijzigen
  + M: modify: een-na-hoogste recht: file rechten wijzigen zoals de aanmaak datum wijzigen

De tabellen van een database:

Tabelrechten:

* Standaard geen toegang
* CRUD:
  + C:create:tabel maken of een rij toevoegen aan de tabel(insert)
  + R:read: lezen van de inhoud van de tabel.
  + U: update: wijzigen van een rij in de tabel of structuur van de tabel (kolommen)
  + D: delete: verwijderen rij of tabel.
* Root : degene die toegang tot de database verdeeld.
* Mag nooit de DB verbinding maken als root.

Beheren gaat via RBAC: role based acces control.

Beheren van de toegang gaat via rollen:

* Een gebruiker is lid van een of meer rollen
* Een rol is meestal een beroep of speciale taak
* Docent/student zijn beroep
* Speciale taak: de medewerker die de kassa opmaakt
* Het systeem voegt een aantal speciale rollen toe:
  + Iedereen
  + Eigenaar
* Ook een speciale rol is: klas
* Dus kees uit AM3D heeft de volgende rollen:
  + Hij is student
  + Hij is AM3D
  + Iedereen

Webapplicatie:

Toegang tot schermen beveiligen:

* In de requirement staat: de verkoper moet een order kunnen aanmaken.
* Dus de toegang tot het orderscherm moet worden afgeschermd
* Dan krijg je de volgende regels:
  + Iedereen heeft geen toegang
  + Een verkoper heeft toegang
  + De eigenaar regelt de toegang: programmeur.

Er zijn drie verschillende systemen::

* Domeinen
* Internet
* Domeinen in verbinding met internet

Domeinen:

Zijn afgesloten omgevingen zoals deze school.

Mboutrecht.nl

Een domein wordt beheerd via RBAC: role based access control.

Dus iedere gebruiker krijgt een rol.

Dus als je inlogt krijg je gelijk meerdere rollen in ieder geval: iedereen.

Er wordt een ticket gemaakt. In het ticket staat je userID, je rollen en een eindtijd van het ticket.

Rollen die verkregen worden bij het inloggen heten globale rollen. De rol is bruikbaar binnen het hele domein.

Er zijn ook lokale rollen: deze beheren de toegang tot een object.

Stel: odijk is doecent en iedereen: globale rollen.

Beheren van onderwijsonline: lokale rollen

* Lezen van onderwijsonline
* Wijzigen van onderwijsonline: aanmaken van mappen(e-learning) en wijzigen daarvan
* Toegang geven tot de map
* Eigenaar van onderwijsonline

Aan deze lokale rollen moeten de globale rollen gekoppeld worden.

Dus:

* Iedereen heeft leesrecht op onderwijsonline
* Docent heeft create,update,delete rechten op onderwijsonline betreffende de mappen
* Eigenaar (programmeur??) beheert alles en regelt de rechten.

Probleem:

* Een persoon kan zich voordoen als een docent
* Het ticket is dus veranderd
* Hij komt met het ticket bij onderwijsonline en die denkt het is een docent dus: hupsakee

Ticket komt uit de bioscoop:

* Je betaalt voor een film en krijgt een ticket
* Bij de zaal, controleert iemand het ticket.

Dus bij een ticketsysteem moet iemand controleren dat het ticket goed is.

Dus binnen het domein is een server verantwoordelijk voor de tickets: deze garandeert het ticket

Deze server wordt meestal cerberus genoemd.

Dus de procedure:

* Odijk logt in
* Odijk ontvangt van cerberus een ticket met zijn globale rollen
* Odijk gaat naar onderwijsonline
* Onderwijsonline ontvangt het ticket
* Onderwijsonline vraagt aan cerberus of het ticket betrouwbaar is.
* Als cerberus zegt dat het betrouwbaar is krijgt odijk toegang

e-mail:

truus met [Truus@gmail.com](mailto:Truus@gmail.com) stuurt naar [klaas@mail.nl](mailto:klaas@mail.nl)

* google vraagt aan een ticketsysteem een ticket voor truus
* de e-mail wordt voorzien van een ticket
* het ticketsysteem weet dat er een ticket geleverd is aan [truus@gmail.com](mailto:truus@gmail.com)
* klaas ontvangt de e-mail en vraagt aan het ticketsysteem of dit ticket hoort bij truus.