Part → 3

⑫ → यदि कोई security filter Block करता है तो response में 401/403 error आयेगा।

"

Authentication architecture in Spring security

"

→ The Proces of identifying and Verifying the user is called authentication.

Eg :→ Suppose you want to verify the identity of a person, so we have different mechanism available in our country India like - • Aadhar Card
• PAN card
• Voter card
... etc

"

Similarly in our application, we can authenticate or Verifies the identity of user by following mechanism":-

Authentication Mechanism (in Spring security) :→

① Username and Password
② OAuth 2.0 Login
③ SAML 2.0 Login
④ Central Authentication server (CAS)
⑤ JAAS Authentication
⑥ Pre-Authentication Scenarios
⑦ X509 Authentication
⑧ JWT Authentication System

→ "In Spring security, we are free to choose any of the authentication mechanism that is supported by spring security"

Authorization :→ Authorization means giving the user access capability so that user can perform only granted activities in our app.

e.g :→ e-commerce platform app :-

normal user / customer : are only allowed to search products, buy products and etc.

Seller :- are allowed to edit their products / inventory. He can update the Product details, its price and much more

admin :- are allowed to cancel Order, accelerate delivery. and see the sales report.

* Architecture of Authentication in spring security *

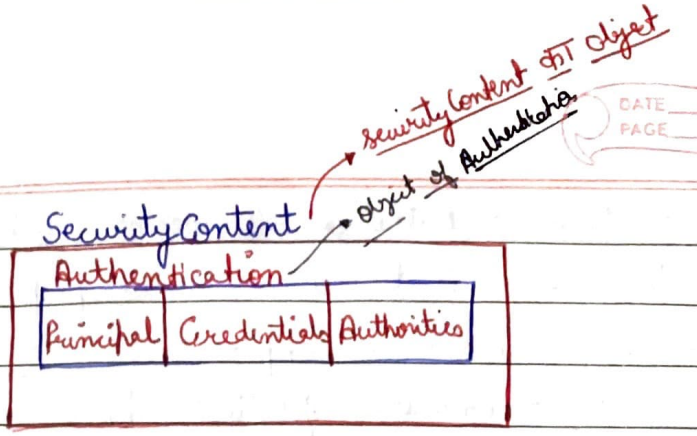
There are very important key components of Authentication in Spring security :→

- SecurityContext
- SecurityContextHolder
- Authentication
- GrantedAuthority
- AuthenticationManager
- ProviderManager
- AuthenticationProvider
- RequestCredentials with AuthenticationEntryPoint
- AbstractAuthenticationProcessingFilter

① SecurityContext :→ • It is an "Interface" present in org.springframework.security.core.context package.

→ SecurityContextImpl is the implementing class of spring security-Context.

→ SecurityContext is the core component of spring security because It object holds the authenticated user object / Information



Authentication :→ Authentication in a spring security is an Interface present in the org.springframework.security.core.

- There are so many Implementing class of Authentication Interface
  - UsernamePasswordAuthenticationToken
  - JwtAuthenticationToken
  - OAuth2AuthenticationToken
  - AbstractAuthenticationToken

  .... etc यहाँ बहुत सारी हैं।

- AuthenticationManager will give the object of Implementing class.

Note :→ "Based upon the ~~requirement~~ authentication mechanism that we are using to authenticate the Person or user in our application, AuthenticationManager will return you the one of the following Authentication Implementing class object.

AuthenticationManager	AuthenticationManager returning the object
→ Username And Password	→ UsernamePasswordAuthenticationToken
→ OAuth 2.0 login	→ OAuth2LoginAuthenticationToken
→ Saml2Authentication	→ Saml2AuthenticationToken
⋮	⋮
x	→ xAuthenticationToken

Security Content को कोई फ़िक्र नहीं पड़ता है की तुम Authentication का object कौन-से Mechanism से बना रहे हो। इसका मतलब की हम कोई सा भी Authentication mechanism use कर सकते हैं।

→ SecurityContext का हिस्सा Authentication के object से संबंधित है।

→ Authentication object जो की store है springSecurity content में. जो represent करेगा current Authenticated user को।

→ Suppose Rahul Login कर रहा है means अपने आप को Authenticate करना चाह रहा है। तो Basically Authentication का object Create होगा। Authentication Object के अंदर Basically 3 Part है।

① Principal → User से related Information → Userid etc. (UserDetails Object)

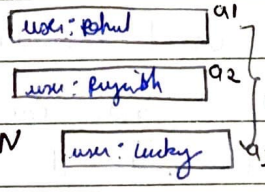
② Credentials → Password (return null)

③ Authority → क्या Role है (Means User / admin) Permission that are granted.
(GrantedAuthority object) → Roles and scopes

→ GrantedAuthority :→

- GrantedAuthority Object represents the authority that is given or granted to the authenticated user means.... Authentication Object.

- Generally, We assign roles to the authenticated user i.e authentication object.
  ROLE_USER
  ROLE_ADMIN
  ROLE_SUPER_ADMIN



∴ It is possible that one user is having multiple granted authorities. i.e one authentication object can have multiple granted authorities.

like a1 → ROLE_USER
ROLE_ADMIN

a2 → ROLE_USER
ROLE_ADMIN
ROLE_SUPER_ADMIN

(2) → Authentication Manager :- • It is a "interface" present in org.springframework.security.authentication.

- AuthenticationManager tries to authenticate the given user.
- It has one method :-

Authenticate authenticate (Authentication authen) throws AuthenticateException

Q. Suppose, Rahul is a user, and wants to use the spring security configured Application, so how he can be authenticated by his information?

→ AuthenticationManager takes the help of authenticate (Authentication auth) method to attempt authentication process.

→ authenticate (Authentication authentication) method

authenticate (....) method here, is asking one Authentication Object.

- So, we give the authentication object to the authenticate method. Now this, authentication object could contain the username, password of the user here.

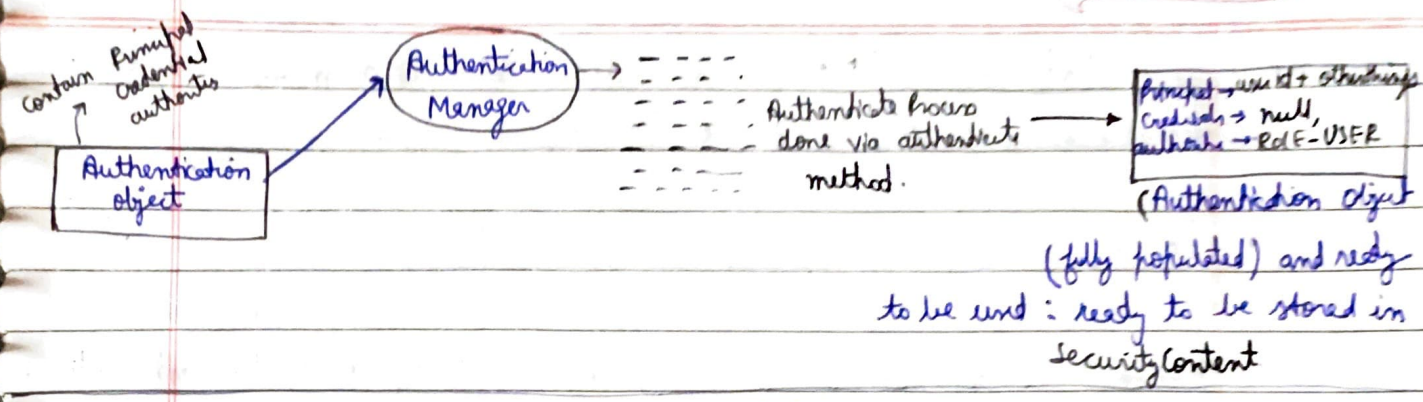
Example → Rahul will give the username & password then username, password will be set to the authentication object

Principal	Credentials	authorities

authentication object

Que → How to give the username and password details (principal and Credentials details).

Ans → Through the Object of UserDetails (Interface) you can give the principal and Credentials details to the Authentication Object that is going to be (in process) authenticated



Note :→ AuthenticationManager is authenticating the user by authenticate(Authentication auth)

- authenticate method is asking "Authentication Object".
- authenticate method is returning the fully populated "Authentication Object". → return type Authentication

"Yes sir, it is returning the Authentication Object and also, it is asking for Authentication Object".

- After successful authentication, the Authentication Object returned by the Authentication Manager is ready to be stored in the SecurityContext. Hence you can say that user is authenticated successfully if SecurityContext is populated.
means..... security Context is having the Authentication Object.

Note :- "If authentication fails for the given Authentication Object, then - AuthenticationException, the AuthenticationException

→ When we talk about AuthenticationManager then it means we are talking about its implementing class.

→ "ProviderManager" is a implementing class of Authentication-Manager Interface

→

③ ProviderManager :→ • It is the most commonly used implementing class of AuthenticationManager

- It is given by SpringSecurity itself.
- ProviderManager (AuthenticationManager) has list of AuthenticationProviders. (interface)
- AuthenticationProvider is the main component in Spring security which has the actual implementation of authentication i.e actual implementation of authenticate(....) method.

Means AuthenticationProvider will decide or will do the actual Authentication and return the result to the Authenticationmanager (ProviderManager).

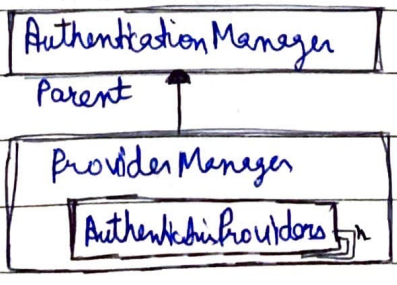
- AuthenticateProvider decides how to authenticate any User.

There are different-different AuthenticationProvider :→

→ Suppose you are giving username & Password then XAuthenticationProvider is there authenticate the user you.

If the user is trying to be authenticated by OAuth (google, github, apple, facebook) then there is YAuthenticatProvider.

If the user trying to be authenticated by JwtToken then there is ZAuthenticationProvider. is.



→ ProviderManager (AuthenticationManager) is attempting the authentication by delegating the authentication proces to these list of AuthenticationProviders one by one.

En :- if AuthenticationProvider0 is failed to authenticate then control goes to AuthenticationProvider1 .... if it also failed then the authentication then control goes to AuthenticationProvider2 .... and so on .... or until not-null response is returned until the authentication is succesfully .....

Suppose authentication succesfully done by the AuthenticationProviders then it will return the result to the AuthenticationManager (AuthenticationProvider).

→ But, if no any AuthenticationProvider from the list performs the succesfull authentication means - all the AuthenticationProviders in the list fails to authenticate the user then, exception will be raised - ProviderNotFoundException and this Exception leads to raisid AuthenticationException.

There are so many list of AuthenticationProviders

- AnonymousAuthenticationProvider
- CasAuthenticationProvider
- JwtAuthenticationProvider
- TestingAuthenticationProvider
- DaoAuthenticationProvider
- RememberMeAuthenticationProvider

..... etc

→

Note :-

public class XYZAuthenticationProvider implements AuthenticationProvider
{ if (auth instanceOf XYZAuthenticationProvider || supports (authentication))
  // some logic will be written there

  public Authentication authenticate (Authentication auth)
  (authentication object coming from AuthenticationManager this object is not fully populated).

  return auth;
  return fully populated Authentication Object when there successfull authentication happens.
}

→ If authenticate method return null for the given XYZ AuthenticationProvider then this auth provider will delegate to the next AbcAuthenticationProvider and this process will go on..... until not-null result and a fully populated authentication object is achieved.

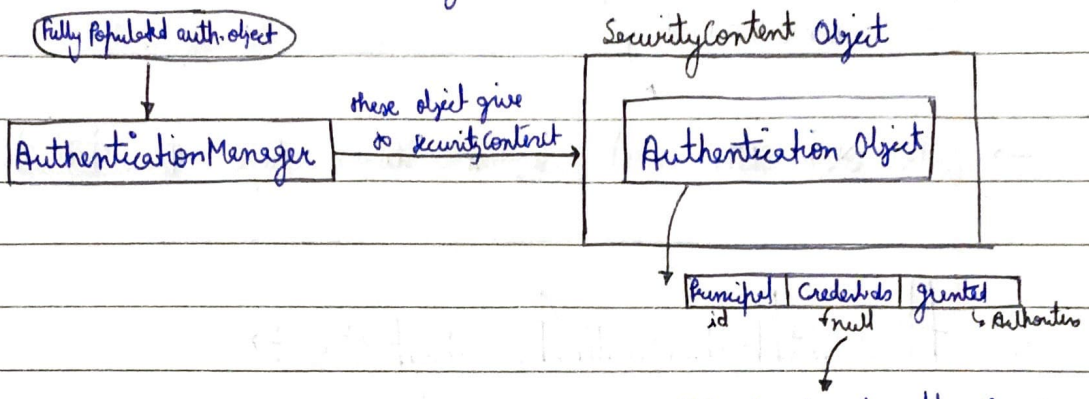
principal (user-details)	credentials : null (password)	granted authority → (ROLE_USER, ROLE_ADMIN)

- After the successfull authentication, AuthenticationProvider is returning the fully populated authentication Object where credentials are erased (becomes null here) for the security purposes so that credentials (often passwords) can never be leaked.

→ Now, this fully populated authentication object is ready to be stored in the SecurityContext.

AuthenticationManager → [AbcAuthenticationProvider] → authentication object

→ ये Method का return type अगर Authentication है तो ये object अब AuthenticationManager को भेजा जायेगा।



this is cleared after the succesfull authentication but still you can change the option not to be clear the credentials even after the succesfull authentication by using - setEraseCredentialsAfterAuthentication(false) method of AuthenticationManager

(4) AuthenticationEntryPoint :→ • It is an "interface" present in org.springframework.security.web.

- When we are talking the AuthenticationEntryPoint it means we are talking about the one of its implementing class.
- This Component is used to send HttpResponse where it says :- hey user .... please give credentials / correct Credentials with this request.
- if the user is already sending the credentials (username & password) then it will not say (give response) to give Credentials, because we are already giving it in credentials.

☆☆

Note - If the user is not giving the Credentials and hitting the request then it means, it is unauthenticated so in these Case, an implement class object of AuthenticationEntryPoint

→

will be used to send the required response that says - hey user ! send the credentials ... with response like 401 - unauthorized :)

→ और आपके अपने को again login Page पर फेंक देगा !