W05 HomeWork – Talha Arıç, talharic@gmail.com

1 – What are Authentication and Authorization ?

Authentication is the act of validating that users are whom they claim to be. This is the first step in any security process.

Complete an authentication process with:

Passwords. Usernames and passwords are the most common authentication factors. If a user enters the correct data, the system assumes the identity is valid and grants access.

One-time pins. Grant access for only one session or transaction.

Authentication apps. Generate security codes via an outside party that grants access.

Biometrics. A user presents a fingerprint or eye scan to gain access to the system.

In some instances, systems require the successful verification of more than one factor before granting access. This multi-factor authentication (MFA) requirement is often deployed to increase security beyond what passwords alone can provide.

Authorization in system security is the process of giving the user permission to access a specific resource or function. This term is often used interchangeably with access control or client privilege.

Giving someone permission to download a particular file on a server or providing individual users with administrative access to an application are good examples of authorization.

In secure environments, authorization must always follow authentication. Users should first prove that their identities are genuine before an organization’s administrators grant them access to the requested resources.

2 - What is Hashing in Spring Security ?

Hashing is the process of generating a string, or hash, from a given message using a mathematical function known as a cryptographic hash function.

While there are several hash functions out there, those tailored to hashing passwords need to have four main properties to be secure:

It should be deterministic: the same message processed by the same hash function should always produce the same hash

It's not reversible: it's impractical to generate a message from its hash

It has high entropy: a small change to a message should produce a vastly different hash

And it resists collisions: two different messages should not produce the same hash

A hash function that has all four properties is a strong candidate for password hashing since together they dramatically increase the difficulty in reverse-engineering the password from the hash.

Also, though, password hashing functions should be slow. A fast algorithm would aid brute force attacks in which a hacker will attempt to guess a password by hashing and comparing billions (or trillions) of potential passwords per second.

3 - What is Salting and why do we use the process of Salting ?

Salting hashes sounds like one of the steps of a hash browns recipe, but in cryptography, the expression refers to adding random data to the input of a hash function to guarantee a unique output, the hash, even when the inputs are the same. Consequently, the unique hash produced by adding the salt can protect us against different attack vectors, such as hash table attacks, while slowing down dictionary and brute-force offline attacks.

However, there are limitations in the protections that a salt can provide. If the attacker is hitting an online service with a credential stuffing attack, a subset of the brute force attack category, salts won't help at all because the legitimate server is doing the salting+hashing for you.

4 - What is “intercept-url” pattern ?

Most web applications using Spring Security only have a couple of intercept-urls because they only have very basic security requirements. You need to have unauthenticated access to the login and login-error screens and usually some aspect of the public site, so that can be a few URL patterns. Then there's often an admin section, and then everything else is ROLE_USER.

5 - What do you mean by session management in Spring Security ?

HTTP session related functonality is handled by a combination of the SessionManagementFilter and the SessionAuthenticationStrategy interface, which the filter delegates to. Typical usage includes session-fixation protection attack prevention, detection of session timeouts and restrictions on how many sessions an authenticated user may have open concurrently.

6 – Why we need Exception Handling ?

The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors so that the normal flow of the application can be maintained.

Suppose have func1 calling func2 with some input.

Now, suppose func2 fails for some reason.

Your suggestion is to handle the failure within func2, and then return to func1.

How will func1 "know" what error (if any) has occurred in func2 and how to proceed from that point?

The first solution that comes to mind is an error-code that func2 will return, where typically, a zero value will represent "OK", and each of the other (non-zero) values will represent a specific error that has occurred.

The problem with this mechanism is that it limits your flexibility in adding / handling new error-codes.

With the exception mechanism, you have a generic Exception object, which can be extended to any specific type of exception. In a way, it is similar to an error-code, but it can contain more information (for example, an error-message string).

7 - Explain what is AuthenticationManager in Spring security ?

The AuthenticationManager is really just a container for authentication providers, giving a consistent interface to them all. In most cases, the default AuthenticationManager is more than sufficient.

.authenticate(new UsernamePasswordAuthenticationToken(username, password))

it is passing the UsernamePasswordAuthenticationToken to the default AuthenticationProvider, which will use the userDetailsService to get the user based on username and compare that user's password with the one in the authentication token.

In general, the AuthenticationManager passes some sort of AuthenticationToken to the each of it's AuthenticationProviders and they each inspect it and, if they can use it to authenticate, they return with an indication of "Authenticated", "Unauthenticated", or "Could not authenticate" (which indicates the provider did not know how to handle the token, so it passed on processing it)

8 - What is Spring Security Filter Chain ?

Spring Security's web infrastructure is based entirely on standard servlet filters. It doesn't use servlets or any other servlet-based frameworks (such as Spring MVC) internally, so it has no strong links to any particular web technology. It deals in HttpServletRequests and HttpServletResponses and doesn't care whether the requests come from a browser, a web service client, an HttpInvoker or an AJAX application.

Spring Security maintains a filter chain internally where each of the filters has a particular responsibility and filters are added or removed from the configuration depending on which services are required. The ordering of the filters is important as there are dependencies between them. If you have been using namespace configuration, then the filters are automatically configured for you and you don't have to define any Spring beans explicitly but here may be times when you want full control over the security filter chain, either because you are using features which aren't supported in the namespace, or you are using your own customized versions of classes.

9 – What are the differences between OAuth2 and JWT ?

-OAuth 2.0 defines a protocol, i.e. specifies how tokens are transferred, JWT defines a token format.

-OAuth can use either JWT as a token format or access token which is a bearer token.

-OpenID connect mostly use JWT as a token format.

10 - What is method security and why do we need it ?

In a properly designed application the backend and frontend are disconnected. The backend security system can't assume any specific frontend will correctly handle security, so it has to handle it itself.

11 – What Proxy means and how and where can be used ?

A proxy is a system or router that provides a gateway between users and the internet. Therefore, it helps prevent cyber attackers from entering a private network. It is a server, referred to as an “intermediary” because it goes between end-users and the web pages they visit online.

When a computer connects to the internet, it uses an IP address. This is similar to your home’s street address, telling incoming data where to go and marking outgoing data with a return address for other devices to authenticate. A proxy server is essentially a computer on the internet that has an IP address of its own.

12 – Waht is Wrapper Class and where can be used ?

A Wrapper class is a class whose object wraps or contains primitive data types. When we create an object to a wrapper class, it contains a field and in this field, we can store primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

They convert primitive data types into objects. Objects are needed if we wish to modify the arguments passed into a method (because primitive types are passed by value).

The classes in java.util package handles only objects and hence wrapper classes help in this case also.

Data structures in the Collection framework, such as ArrayList and Vector, store only objects (reference types) and not primitive types.

An object is needed to support synchronization in multithreading.

13 – What is SSL ? What is TLS ? What is the difference ? How can we use them ?

Secure Sockets Layer (SSL) is a security protocol that uses a modern encryption method to send and receive sensitive information over the internet. It works by creating a secure channel between a user's browser and the server of the user's desired website. Any information passing through this channel is encrypted at one end and decrypted when received at the other end. Thus, even if someone gets their hands on this information, it will be of no use because the information is encrypted.

TLS (Transport Layer, Security) or Transport Layer Security is the security layer that encrypts data between two communication applications and ensures that it is transmitted securely. TLS, the content of which was revealed for the first time in 1999, uses asymmetric encryption algorithm to authenticate. TLS is accepted as a more advanced and secure version of SSL (Secure Sockets Layer) protocol, which is a different security layer developed by Netscape.

TLS is used to provide secure data transfer between applications that are in communication with each other. Even if you are not aware of it, you may be using TLS in different ways every day on the internet. TLS is widely used in secure internet site connections, instant messaging software, file transfers, various software, internet applications, VPN connections and even VOIP connections, which give serious importance to security.

SSL and TSL are both essentially encryption and authentication protocols. With the help of these protocols, data transfer is carried out securely from the server to the server or from the server to the client. SSL is the more primitive version of TSL. As a result of years of development in the IT and internet world, TSL has emerged, which I can consider as the newer version of SSL.

14 - Why do you need the intercept-url ?

To use expressions to secure individual URLs, you would first need to set the use-expressions attribute in the <http> element to true. Spring Security will then expect the access attributes of the <intercept-url> elements to contain Spring EL expressions. The expressions should evaluate to a boolean, defining whether access should be allowed or not.