**Как работает Spring Security:**

Spring Security предоставляет свой filter chain, подобной той, что используется в сервлетах, то есть, у нас есть некий контейнер, внутри которого есть ряд фильтров и эти фильтры срабатывают один за другим, но в случае Spring Security, в этом контейнере есть также главный фильтр, DelegationFilterProxy, который хранит в себе бин FilterChainProxy и уже этот бин вызывает SequrityFilterChain, в котором реализована конечная цепочка вызова фильтров для обеспечения безопасности. Если условия фильтров будут соблюдены, тогда запрос от клиента попадет в DispetcherServlet.

SecurityFilterAutoConfiguration – это класс для авто конфигурации.

SecurityProperties — это класс, который содержит в себе настройки для properties.

FilterChainProxy — Это бин.

DelegationFilterProxyRegistrationBean - главный класс, который занимается тем, что создает DelegationFilterProxy.

DelegationFilterProxy — это фильтр, который подчиняется сервлет контейнеру и содержит в себе бин FilterChainProxy

Он может быть объявлен в web.xml или с помощью аннотации @ServletComponentScan для автоматического обнаружения.

SequrityFilterChain — представляет собой конкретную цепочку фильтров, предназначенную для обработки запросов, связанных с безопасностью.

Она содержит различные фильтры для выполнения задач аутентификации, авторизации и других аспектов безопасности.

SpringBootWebSecurityConfiguration - класс, который создает SequrityFilterChain. Дает возможность нам реализовать свою логику аутентификации

Жизненный цикл филтра в сервлетах

1. init () - инициализация фильтра

2. doFilter () - метод, в котором описана логика для авторизации, аутентификации

3. deStroy () - завершение