以帳號密碼認證機制介紹 spring security 運作方式

Martin

自我介紹

喜歡探索並應用軟體科技解決問題。

為了解 Java 相關科技的發展而參與 TWJUG 的活動。

簡報目的與期望

- 回饋社群
- 教學相長

現學現賣, 若有錯誤、疏漏請不吝指正。

歡迎轉載。

大綱

Spring Security 架構概觀 ── FilterChainProxy、SecurityFilterChain

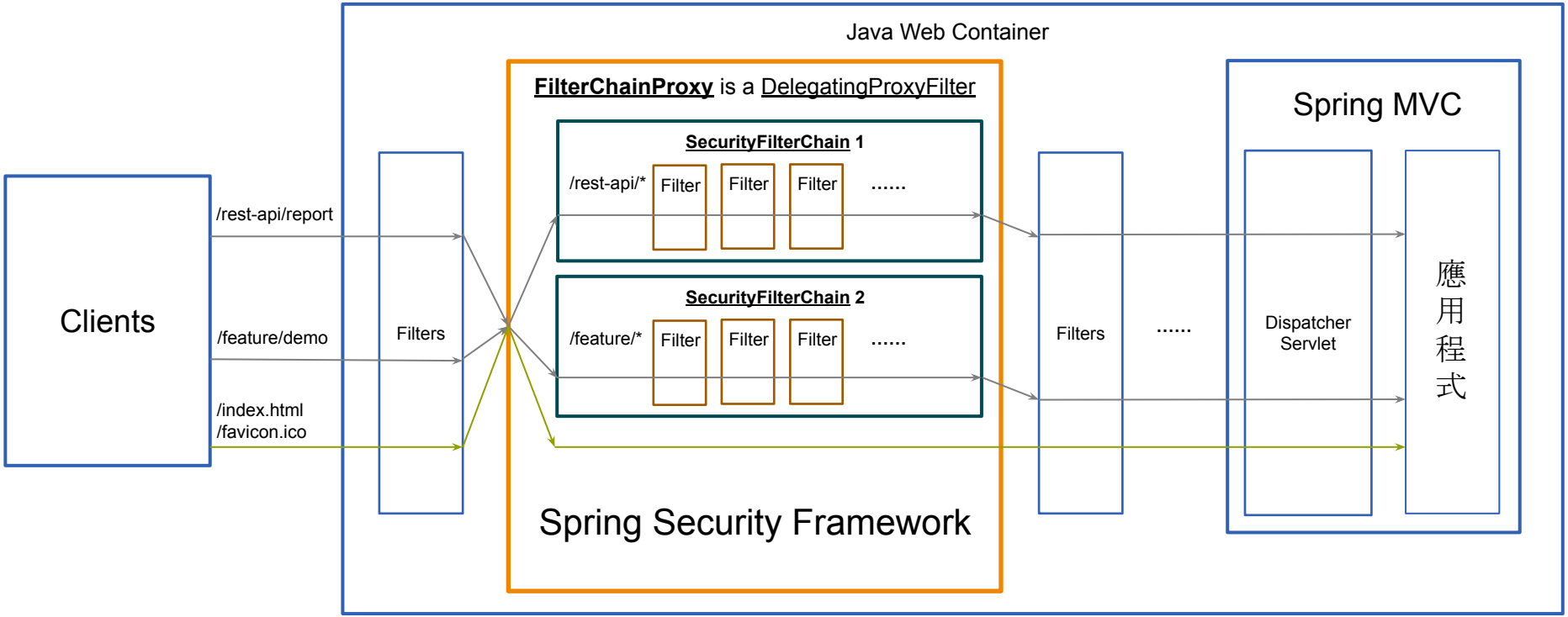
Security Filter Chain 的運作方式

以 UsernamePasswordAuthenticationFilter 等元件實現帳密認證的方式

滿足實際需求的對策

Spring Security 架構

擋在 Spring MVC 前面，可與 Spring MVC 緊密合作，可不依賴 Spring MVC 的功能。



Security Filter Chain

當系統收到存取非公開資源的請求時:

讀取認證紀錄
SecurityContextPersistenceFilter:建立 SecurityContext 給後續 Filter 存取認證紀錄。
儲存 Authentication 到 SecurityContext。

審查認證資格
ChannelProcessingFilter:確保發送請求的連線有加密。

認證使用者
UsernamePasswordAuthenticationFilter、CasAuthenticationFilter、BasicAuthenticationFilter 等認證 Filter:
依據請求中夾帶的信物(Credentials)認證請求發送者的身份。

AnonymousAuthenticationFilter:斷定未通過上述認證的請求之使用者為匿名者。

進行防護工作
SessionManagementFilter:
替換 session id 以防止 session-fixation 攻擊。

依據認證紀錄斷定權限
FilterSecurityInterceptor(這是個 Filter):
檢查權限, 在無權限時拋出 AccessDeniedException …… 一種RuntimeException

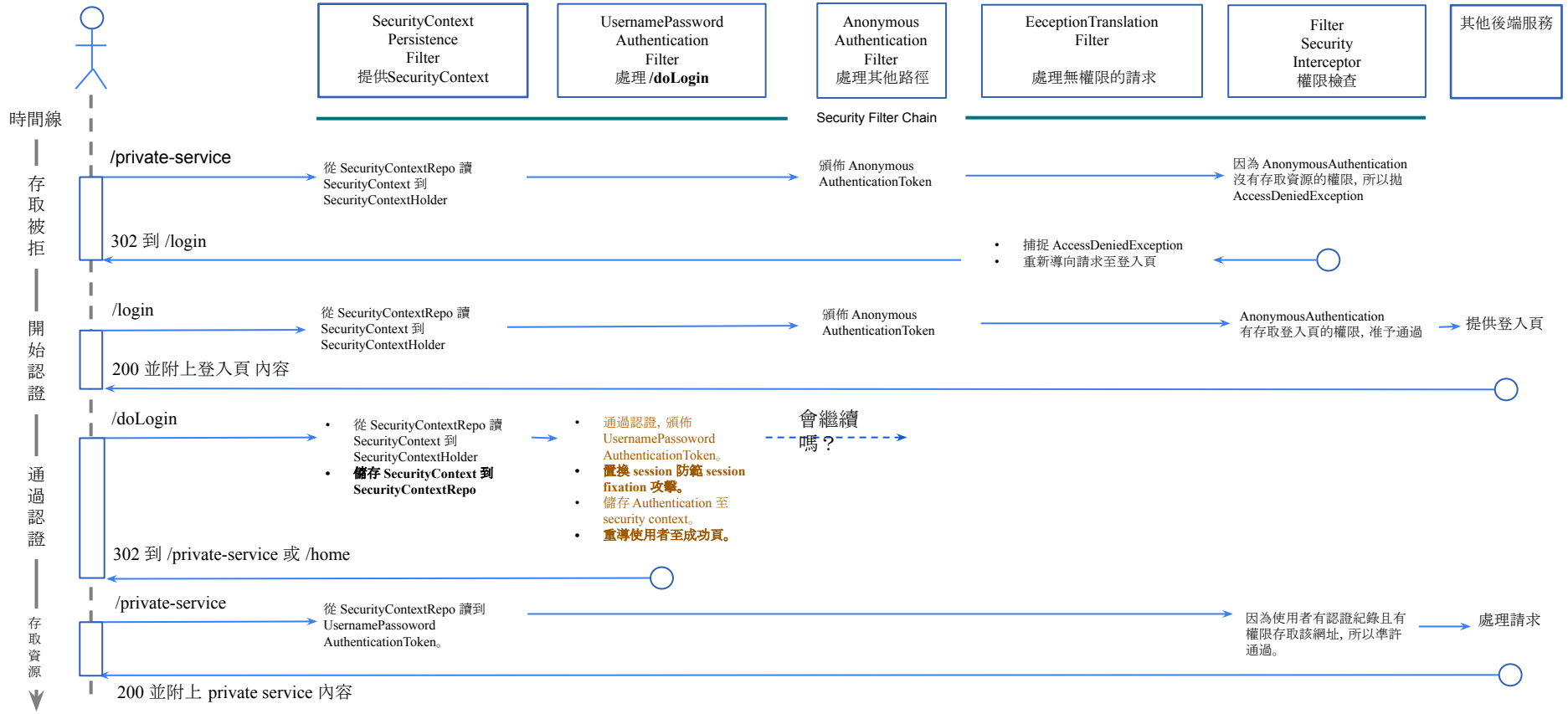
ExceptionTranslationFilter:
捕捉前者拋出的例外並視情況提供request 與 response 給實作 AuthenticationEntryPoint 的類別進行導引客戶認證身份的工作作業。
與前者合作實現權限控管機制。

常見的 Filter Chain 成員

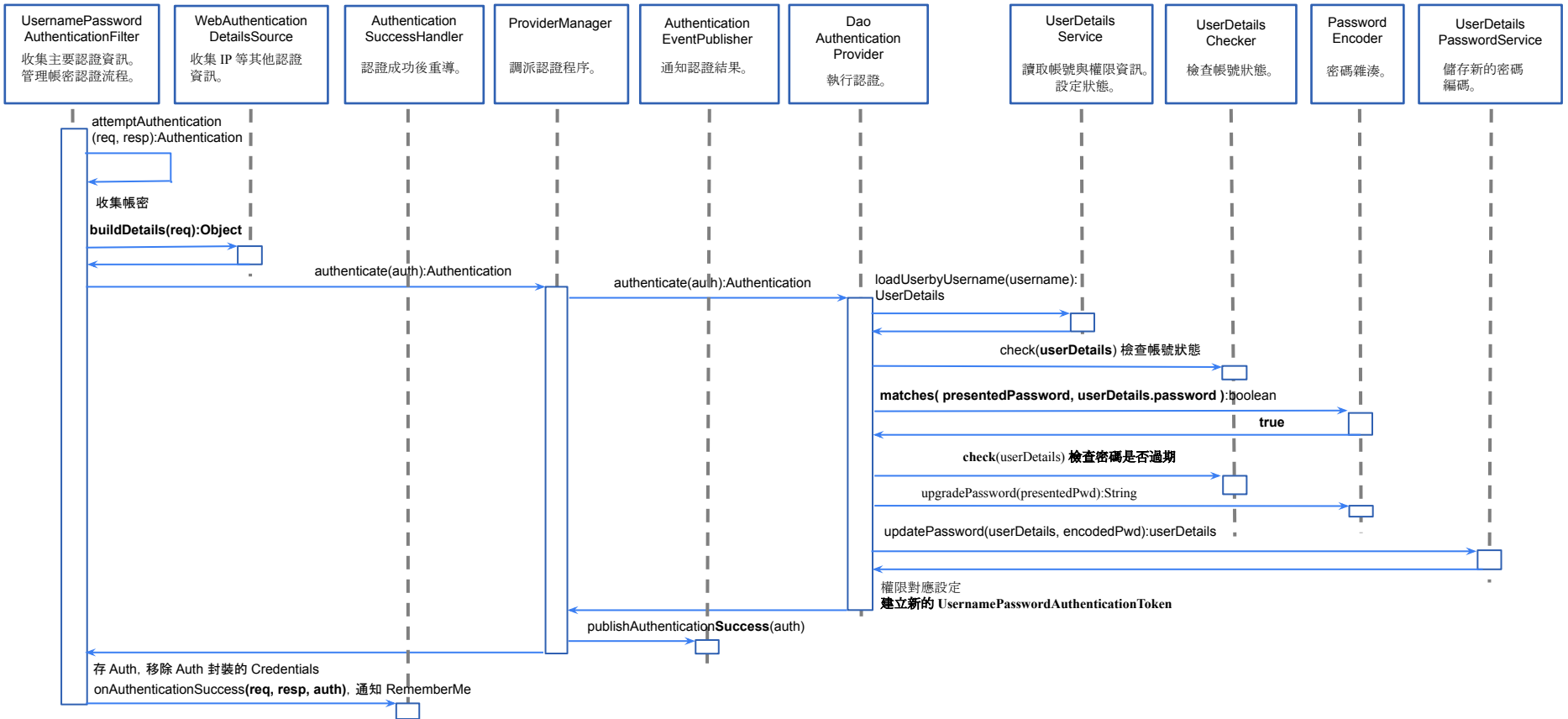
- ChannelProcessingFilter
- SecurityContextPersistenceFilter
- LogoutFilter
- UsernamePasswordAuthenticationFilter
- AnonymousAuthenticationFilter
- SessionManagementFilter
- ExceptionTranslationFilter
- FilterSecurityInterceptor

Filter Chain 的順序

Security Filter Chain 的運作方式



帳密認證 — 認證成功


UsernamePassword AuthenticationFilter
收集主要認證資訊。管理帳密認證流程。
WebAuthentication DetailsSource
收集 IP 等其他認證資訊。
Authentication SuccessHandler
認證成功後重導。
ProviderManager
調派認證程序。
Authentication EventPublisher
通知認證結果。
Dao Authentication Provider
執行認證。
UserDetails Service
讀取帳號與權限資訊。設定狀態。
UserDetails Checker
檢查帳號狀態。
Password Encoder
密碼雜湊。
UserDetails PasswordService
儲存新的密碼編碼。
attemptAuthentication (req, resp):Authentication
收集帳密
buildDetails(req):Object
authenticate(auth):Authentication
authenticate(auth):Authentication
loadUserbyUsername(username): UserDetails
check(userDetails) 檢查帳號狀態
matches(presentedPassword, userDetails.password):boolean
true
check(userDetails) 檢查密碼是否過期
upgradePassword(presentedPwd):String
updatePassword(userDetails, encodedPwd):userDetails
權限對應設定
建立新的 UsernamePasswordAuthenticationToken
publishAuthenticationSuccess(auth)
存 Auth，移除 Auth 封裝的 Credentials
onAuthenticationSuccess(req, resp, auth)，通知 RememberMe

帳密認證 — 密碼錯誤導致認證未通過

帳密認證 ── 注意事項

各元件的介面深深地影響了我們組織一切工作的方式。

元件	主要方法與其參數	注意事項
UsernamePassword AuthenticationFilter	attemptAuthentication(HttpServletRequest, FilterChain):Authentication	
AuthenticationManager AuthenticationProvider	authenticate(org.springframework.security.core.Authentication):Authentication	• Authentication: Principle ── 帳號名稱字串 Credentials ── 信物 Details ── 從請求中拿到的其他資訊 • 塞進去的 Authentication 跟拿回來的不太一樣。
UserDetailsService	loadUserByUsername(**String** username):UserDetails throws **UsernameNotFoundException**	拋其他 AuthenticationException 將導致 DaoAuthenticationProvider 視為程序異常，因而不套用一般認證失敗的處置對策。
DetailsChecker	check(org.springframework.security.core.userdetails.UserDetails):void **可拋** RuntimeException ── AuthenticationException	• 拿到的是 UserDetails 而非 Authentication。 • UserDetails 通常不會附帶認證錯誤訊息。
Authentication EventPublisher	publishAuthenticationSuccess(Authentication):void publishAuthenticationFailure(AuthenticationException, Authentication):void	拿到的是 Authentication 而非 UserDetails。
Authentication SuccessHandler	HttpServletRequest, HttpServletResponse, Authentication	Authentication 之 Principle 屬性變成 UserDetails 而非原本的帳號字串。
Authentication FailureHandler	HttpServletRequest, HttpServletResponse, **AuthenticationException**	拿不到 Authentication 和 UserDetails

帳密認證 — 附註

- 被 UsernamePasswordAuthenticationFilter 繼承的 AbstractAuthenticationProcessingFilter:
  定義高層次的帳密認證流程並委派AuthenticationSuccessHandler、FailureHandler 處理認證善後工作。
- ProviderManager 幫多個過濾器管理 AuthenticationProvider
  一個請求只委派給一個Provider 執行認證工作, 不會採納多個Provider 的意見。若認證不過就嘗試以父認證管理器完成認證。
- 被 DaoAuthenticationProvider 繼承的 AbstractUserDetailsAuthenticationProvider:
  定義認證步驟、提供使用者帳號資訊快取。
- DaoAuthenticationProvider 還負責 Timing attack 的防護和密碼編碼置換作業。
- DaoAuthenticationProvider 使用的元件也可以用來給 SwitchUserFilter 實現系統管理者代登入。

滿足常見需求的對策

- 記錄密碼輸入錯誤次數。

  在 Event Publisher 還是 Failure Handler 做？

  Event Publisher 拿得到封裝相關資訊的 Authentication 物件，不像 Failure Handler 要重新再收集，因此可能較合適。

- 授權完畢後，在 request 或 session 屬性中加入非認證用的資料。

  例：設定使用者有權存取的功能之路徑列表到request attribute 以利授權或建立畫面內容

  因為 Success Handler 拿到的 request 是未經 FilterChainProxy 調整過的原始 request，所以適合在此處加上相關參數。

滿足罕見需求的對策

- 主帳號底下還有子帳號，子帳號有各自的密碼。
  沒有對應的欄位該怎麼辦？UserDetailsService 只接受單一字串作為主帳號該怎麼辦？

  建立新類別繼承 UsernamePasswordAuthenticationFilter，覆寫其 obtainUsername 方法將主帳號和子帳號合併為單一字串，然後再於 UserDetailsService 讀取帳號時從帳號字串解析出主帳號和子帳號。

- 登入時要輸入自製的機器人辨別碼。
  在哪裡收集相關資訊？在哪裡判斷有無通過機器人辨識？

  建立新類別繼承 UsernamePasswordAuthenticationFilter，覆寫其 attemptAuthentication 方法。在該方法先比對辨別碼是否正確。若正確則呼叫父類別的相同方法，否則就拋自訂的 AuthenticationException ── 例如 CaptchaNotPassedException ── 然後在自行實作的 Failure Handler 安排捕捉到此類型例外的處理對策。

- 使用者在鎖定期間嘗試再次登入時，要顯示剩餘鎖定時間。
  可以不要在 Failure Handler 重做前面已執行過的查詢作業嗎？

  不在 UserDetailsService 判斷帳號是否鎖定，改在 DetailsChecker 查資料完成此事。若鎖定時間未結束則拋出自訂用以封裝剩餘鎖定時間的 AuthenticationException，然後再讓 Failure Handler 收到此例外時，塞入相關資訊到 request 或 session 裡供重導向的頁面呈現資訊。

參考資料、感謝名單與 Q&A

Spring Security 5.5.2 原始碼

Spring Security 5.5.2 JavaDoc

Spring Security Reference

感謝我主管堅持以較理想的方式組織程式碼, 同時又願意讓我投入時間鑽研。

感謝同事幫忙分擔因為處理認證機制而遲遲無法去處理的工作。

謝謝大家