Remember-Me

> 记住我或持久登录身份验证是指网站能够记住会话之间主体的身份。这通常是通过向浏览器发送 cookie 来完成的，该 cookie 在未来的会话中被检测到并导致自动登录发生。 Spring Security 为这些操作的发生提供了必要的钩子，并且有两个具体的 remember-me 实现。一种使用散列来保护基于 cookie 的令牌的安全性，另一种使用数据库或其他持久存储机制来存储生成的令牌。
>
> 请注意，这两种实现都需要 UserDetailsService。如果您使用的身份验证提供程序不使用 UserDetailsService（例如，LDAP 提供程序），那么除非您的应用程序上下文中还有 UserDetailsService bean，否则它将无法工作

1. 基于 Hash 的 Token 方法 Simple Hash-Based Token Approach

这种方法使用散列来实现有用的记住我的策略。本质上，在交互式身份验证成功后，会将 cookie 发送到浏览器，cookie 的组成如下

```
base64(username + ":" + expirationTime + ":" +
md5Hex(username + ":" + expirationTime + ":" password + ":" + key))

username:          As identifiable to the UserDetailsService
password:          That matches the one in the retrieved UserDetails
expirationTime:    The date and time when the remember-me token expires, ex
key:               A private key to prevent modification of the remember-me
```

纯文本

因此，remember-me 令牌仅在指定的时间段内有效，前提是用户名、密码和密钥不更改。值得注意的是，这有一个潜在的安全问题，即在令牌过期之前，任何用户代理都可以使用捕获的记住我令牌。这与摘要式身份验证的问题相同。如果委托人知道令牌已被捕获，他们可以轻松更改密码并立即使所有已发布的记住我令牌无效。如果需要更重要的安全性，您应该使用下一节中描述的方法。或者，根本不应该使用记住我的服务

开启记住我方法

```xml
<http>
...
<remember-me key="myAppKey"/>
</http>
```

通常会自动选择 UserDetailsService。如果您的应用程序上下文中有多个，则需要指定应与 user-service-ref 属性一起使用的一个，其中值是您的 UserDetailsService bean 的名称

2. 令牌持久化方法 Persistent Token Approach

此方法基于文章 http://jaspan.com/improved_persistent_login_cookie_best_practice 并进行了一些小的修改[1]。要将这种方法与命名空间配置一起使用，您需要提供一个数据源引用

```xml
<http>
...
<remember-me data-source-ref="someDataSource"/>
</http>
```

数据库应该包含一个 persistent_logins 表，使用以下 SQL（或等效的）创建

```sql
create table persistent_logins (username varchar(64) not null,
                                series varchar(64) primary key,
                                token varchar(64) not null,
                                last_used timestamp not null)
```

3. 记住我接口及实现 Remember-Me Interfaces and Implementations

Remember-me 与 UsernamePasswordAuthenticationFilter 一起使用，并通过 AbstractAuthenticationProcessingFilter 超类中的钩子实现。它也被用于 BasicAuthenticationFilter 中。这些钩子将在适当的时候调用具体的 RememberMeServices。该接口看起来像这样

```java
Authentication autoLogin(HttpServletRequest request, HttpServletResponse re

void loginFail(HttpServletRequest request, HttpServletResponse response);
void loginSuccess(HttpServletRequest request, HttpServletResponse response,
    Authentication successfulAuthentication);
```

此阶段请注意 AbstractAuthenticationProcessingFilter 仅调用 loginFail() 和 loginSuccess() 方法。每当 SecurityContextHolder 不包含 Authentication 时，都会由 RememberMeAuthenticationFilter 调用 autoLogin() 方法。因此，该接口为底层的 remember-me 实现提供了足够的身份验证相关事件通知，并在候选 Web 请求可能包含 cookie 并希望被记住时委托给实现。这种设计允许任意数量的记住我的实现策略。我们在上面已经看到 Spring Security 提供了两种实现。

基于令牌的记忆服务 TokenBasedRememberMeServices

这个实现支持简单的基于哈希的令牌方法中描述的更简单的方法。TokenBasedRememberMeServices 生成一个 RememberMeAuthenticationToken，由 RememberMeAuthenticationProvider 处理。这个认证提供者和 TokenBasedRememberMeServices 之间共享一个密钥。此外，TokenBasedRememberMeServices 需要一个 UserDetailsService，它可以从中获取用户名和密码用于签名比较，并生成 RememberMeAuthenticationToken 以包含正确的 GrantedAuthoritys。TokenBasedRememberMeServices 也实现了 Spring Security 的 LogoutHandler 接口，因此可以与 LogoutFilter 一起使用，使 cookie 自动清除

```xml
<!-- 应用程序上下文中启用记住我服务所需的 bean  -->
<bean id="rememberMeFilter" class=
"org.springframework.security.web.authentication.rememberme.RememberMeAuthe
<property name="rememberMeServices" ref="rememberMeServices"/>
<property name="authenticationManager" ref="theAuthenticationManager" />
</bean>

<bean id="rememberMeServices" class=
```

```
"org.springframework.security.web.authentication.rememberme.TokenBasedRemem
<property name="userDetailsService" ref="myUserDetailsService"/>
<property name="key" value="springRocks"/>
</bean>

<bean id="rememberMeAuthenticationProvider" class=
"org.springframework.security.authentication.RememberMeAuthenticationProvid
<property name="key" value="springRocks"/>
</bean>
```

其他存储库来实现令牌 PersistentTokenBasedRememberMeServices

- `InMemoryTokenRepositoryImpl` 仅用于测试.
- `JdbcTokenRepositoryImpl` 它将令牌存储在数据库中.

脚注列表

1. 本质上，用户名不包含在 cookie 中，以防止不必要地暴露登录名。