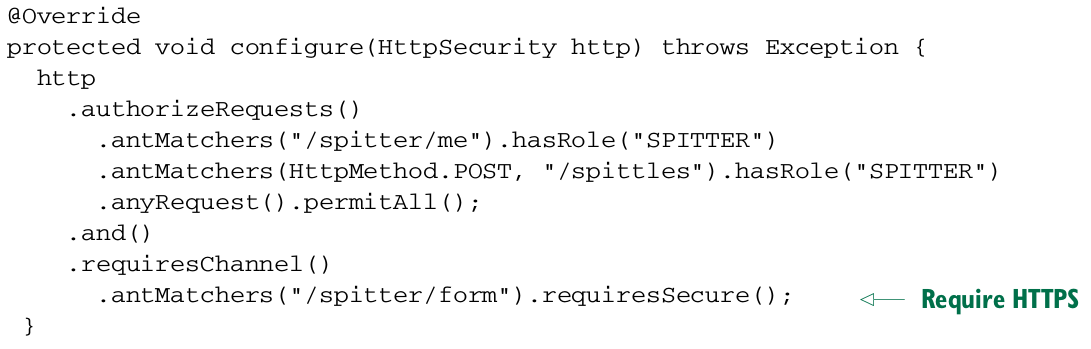
***Enforcing channel security***

Submitting data across HTTP can be a risky proposition. if you’re passing sensitive information such as passwords and credit card numbers across HTTP , then you’re asking for trouble. Data is sent over HTTP unencrypted, leaving an open opportunity for a hacker to intercept the request and see information you don’t want them to see. That’s why sensitive information should be sent encrypted over HTTPS .

In addition to the authorizeRequests() method, the HttpSecurity object passed into configure() has a requiresChannel() method that lets you declare channel requirements for various URL patterns.

To ensure that the registration form is sent over HTTPS , you can add requiresChannel() to the

configuration, as in the following listing.



Any time a request comes in for /spitter/form, Spring Security will see that it requires a secure channel (per the call to requiresSecure() ) and automatically redirect the request to go over HTTPS .

Conversely, some pages don’t need to be sent over HTTPS . The home page, for example, doesn’t carry any sensitive information and should be sent over HTTP . You can declare that the home page always be sent over HTTP by using requires Insecure() instead of requiresSecure :



If a request for / comes in over HTTPS , Spring Security will redirect the request to flow over the insecure HTTP .

Notice that the path selection options for channel enforcement are the same as for authorizeRequests() . In above listing you’re using antMatches() , but regexMatchers() is also available for selecting path patterns with regular expressions.