***Securing with Spring Expressions***

* Most of the methods in table 9.4 are one-dimensional. That is, you can use hasRole()

to require a certain role, but you can’t also use hasIpAddress() to require a specific

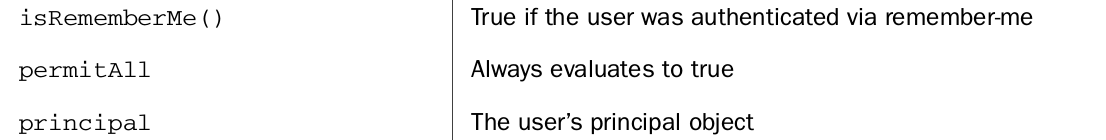
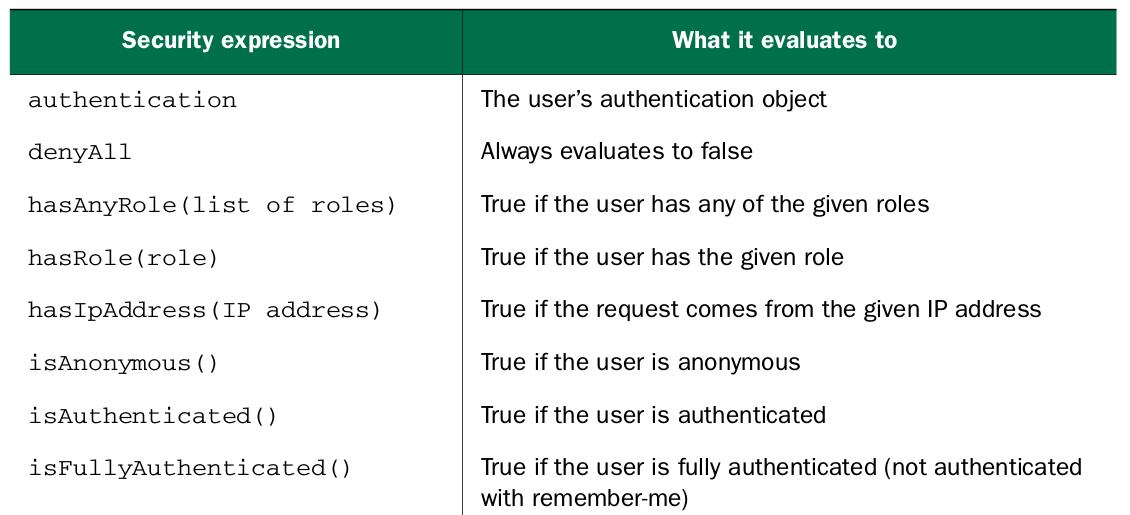
IP address on the same path.

* Moreover, there’s no way to work in any conditions that aren’t defined by the methods in table 9.4. What if you wanted to restrict access to certain roles only on Tuesday?
* Using the access() method, you can also use SpEL as a means for declaring access requirements. For example, here’s how you could use a SpEL expression to require ROLE\_SPITTER access for the /spitter/me URL pattern:



This security constraint placed on /spitter/me is equivalent to the one we started with, except that now it uses SpEL to express the security rules. The hasRole() expression evaluates to true if the current user has been granted the given authority. What makes SpEL a more powerful option here is that hasRole() is only one ofthe security-specific expressions supported.

* All of the SpEL expressions available in Spring Security.



For example, if you wanted to lock down the /spitter/me URL s to not only require ROLE\_SPITTER , but to also only be allowed from a given IP address, you might call the access() method like this:

