# Authentication 2

By default, KFS uses a modified version of CAS for authentication. The version of CAS packaged with KFS is fully functional. It is able to validate passwords and works with https. In addition to the clearly marked sections of the `web.xml` file delivered with KFS that configure CAS itself, filters called `cas` (backed by `KualiCasFilter`) and `UserLoginFilter` are used to integrate it with KFS and KEW respectively via the `WebAuthenticationService` configured in `SpringBeans.xml`. Note that the same `WebAuthenticationService` is being used for KFS and KEW by default, since the `webAuthenticationService` bean is given an alias of `enWebAuthenticationService`. KFS and the Rice Nervous System also depend on some additional processing that occurs in the `processPreprocess` method of the `KualiRequestProcessor` for authentication-related work. But, since that code uses the `WebAuthenticationService`, there should be no need for you to modify it to tweak or override the delivered authentication.

## Configuring the Delivered Authentication Service

There are several configuration properties that allow configuration of the KFS version of CAS and the default `WebAuthenticationService` implementation (`WebAuthenticationServiceCas`). In addition to those specifically designated as pertaining to authentication in the configuration properties overview, the `production.environment.code` property is what ensures that the backdoor login functionality, useful in development and testing of KFS, is disabled in production (see the `setBackdoorUser` method of `UserSession`). Also, the `UNIVERSAL_USER_EDIT_GROUP` parameter determines who can edit institutional user data and who can see the password on the user document.

## Overriding the Delivered Authentication Service

If you want to use another version of CAS or an entirely different authentication mechanism, you will want to...

- Remove the sections of the `web.xml` that configure KFS CAS
- Replace the section that configures the `KualiCasFilter` with an appropriate filter for your authenctication mechanism
- Override the `webAuthenticationService` bean defined in `SpringBeans.xml` with your own implementation of `org.kuali.core.service.WebAuthenticationService` (see the bean overrides documentation for more information).

If you also want to implement your own backdoor login mechanism because you are not using the KFS menu, the easiest way to do this is ensure that your functionality sets a parameter named to match `RiceConstants.BACKDOOR_PARAMETER`.

Unable to render {include} The included page could not be found.