Session management:

The current session management implementation was developed entirely by myself from the grounds up. The session is maintained between the android client and the wcf service by means of a cookie file which is injected as an HTTP header for every wcf method call requiring authentication. This helps me to avoid having to pass user’s username and password with each call to the wcf service.

Below is a detailed description of how the session management works in practice.

When the android client application is launched on the device, the application checks whether a cookie file exists in the FindNDrive application folder located in /sdcard/.

If the file exists, the application will call a wcf service responsible for logging a user in using the session cookie file. Prior to making the call, the application will read the contents of the cookie file and inject its contents as an http header.

In order to prevent users from copying the session file onto another device to hijack the session of that user, the application client adds a dynamically generated unique identifier to the wcf service call and inserts it as yet another http header. The unique identifier is difficult to spoof due to the fact that it is made up of a number of unique identifiers concatenated together.

After receiving the call to attempt automatic login using the session file, the wcf extracts the http headers injected by the application and performs the following actions:

* Extract the id of the user from the session token and find the user’s session data in the database.
* If the id cannot be is extracted, the session file has been tampered with and is malformed, return false and ask the user to manually log in.

Application starts.

Create new instance

Application Manager

Is session information present?

Perform start-up check

Yes

No

Show login activity

Attempt auto-login

Traffic encryption achieved with HTTPS

Inject HTTP headers containing current session information.

Create new LoginDTO object containing username & password.

WCF Service

Failure, ask to log in manually.

Log user in.

Session manager.

Verify user’s username and password.

Extract session information from HTTP headers.

Malformed session data, return failure and ask the user to manually log in.

Success?

No

Success?

Yes

Perform the following checks:

* Check device ID
* Check session ID
* Check randomly generated ID
* Check expiry date

No

Yes

Return failure & display an error message.

Return success. User logged in.

Success?

No

Yes

Return failure, ask user to log in manually.

Return success. User logged in.

Perform manual login verification

Attempt to log user in using session cookie file.

Verify user’s username and password.

Extract user id from token and user’s session data from the database.

Success?

Malformed session cookie file, return failure and user to manually log in.

WCF Service

Session manager.

No

Credentials valid?

Yes

Yes

No

Perform the following comparisons against data from the database:

* DeviceID == DeviceID?
* SessionID == SessionID?
* Cookie expired?

Return success and attach user object.

Return failure and null as user object.

Display ‘wrong username and password’

Success?

Yes

No

Failure, ask to log in manually.

Log user in.