

9. | | | | | | | |--|--|--|--|--|--| | | | | | | | |--|--|--|--|--|--|

--	--	--	--

 9.

--	--	--	--	--	--

 (9. Native Applications)

9. Native Applications

- `URI` Web
- `cookies`
-

9. Native Applications

Native applications are clients installed and executed on the device used by the resource owner (i.e., desktop application, native mobile application). Native applications require special consideration

related to security, platform capabilities, and overall end-user experience.

The authorization endpoint requires interaction between the client and the resource owner's user-agent. Native applications can invoke an external user-agent or embed a user-agent within the application.

For example:

- o External user-agent - the native application can capture the response from the authorization server using a redirection URI with a scheme registered with the operating system to invoke the

client as the handler, manual copy-and-paste of the credentials, running a local web server, installing a user-agent extension, or by providing a redirection URI identifying a server-hosted

resource under the client's control, which in turn makes the response available to the native application.

- o Embedded user-agent - the native application obtains the response by directly communicating with the embedded user-agent by monitoring state changes emitted during the resource load, or

accessing the user-agent's cookies storage.

When choosing between an external or embedded user-agent, developers should consider the following:

- o An external user-agent may improve completion rate, as the resource owner may already have an active session with the authorization server, removing the need to re-authenticate. It

provides a familiar end-user experience and functionality. The

Hardt

Standards Track

[Page 52]

RFC 6749

OAuth 2.0

October 2012

resource owner may also rely on user-agent features or extensions to assist with authentication (e.g., password manager, 2-factor device reader).

- o An embedded user-agent may offer improved usability, as it removes the need to switch context and open new windows.

- o An embedded user-agent poses a security challenge because resource owners are authenticating in an unidentified window without access to the visual protections found in most external user-agents. An embedded user-agent educates end-users to trust unidentified

requests for authentication (making phishing attacks easier to execute).

When choosing between the implicit grant type and the authorization code grant type, the following should be considered:

- o Native applications that use the authorization code grant type SHOULD do so without using client credentials, due to the native application's inability to keep client credentials confidential.

- o When using the implicit grant type flow, a refresh token is not returned, which requires repeating the authorization process once the access token expires.