Implement Login

Cory House
PRINCIPAL CONSULTANT

@housecor   reactjsconsulting.com

Agenda

Implement login in React

- auth0-js
- Auth0 Dashboard

Recommended if your React app has a dedicated server to set an HttpOnly cookie

Storing Tokens

	LocalStorage	SessionStorage	Cookie	In Memory
Persists on browser close	Y		Optional	Via silent auth
Persists on tab close	Y		Y	Via silent auth
Persists across tabs	Y		Y	Via silent auth
Access from subdomains			Y	
Sent on all requests			Y	
Max size	2 - 10MB	5MB - unlimited	4K / domain	No limit
XSRF vulnerable			Y	
XSS vulnerable	Y	Y	Y w/out HttpOnly	Y
Supports secure flag			Y	
Supports HTTP only flag			Y	

Cross-site Scripting

Attacker injects client-side script

Risk: Mishandling user content

- Example: Not HTML encoding

Cross-site Scripting

```
yourapp.com?q=<script%20type='text/javascript'>alert('xss');</script>
```

Imagine code here that reads the user's JWT from localStorage or cookies and sends it to the attacker.

Cross-site Scripting

React helps protect from XSS

- Automatically escapes variables
- Risky behaviors:
  - Using dangerouslySetInnerHTML
  - Passing user supplied vals to props

ARTICLES

- Getting Started
- Architecture Scenarios
- Applications
- Authentication
- APIs
- Connections
- Hosted Pages
- Rules
- Hooks (Beta)
- Email
- Password Policies
- User Management
- Multifactor Authentication
- Security
  - Bulletins
- Anomaly Detection
- Analytics
- Integrations

Where to Store Tokens

In this article

Not sure where to store tokens? This guide outlines how to securely store tokens used in token-based authentication.

Don't store tokens in local storage

Browser local storage (or session storage) is not secure. Any data stored there may be vulnerable to cross-site scripting. If an attacker steals a token, they can gain access to and make requests to your API. Treat tokens like credit card numbers or passwords: don't store them in local storage.

If a backend is present

If your application has a backend server at all, then tokens should be handled server-side using the Authorization Code flow, Authorization Code flow with Proof Key for Code Exchange, or hybrid flow.

Single page applications

If you have a single page application (SPA) with no corresponding backend server, your SPA should request new tokens on page load and store them in memory without any persistence. To make API calls, your SPA would then use the in-memory copy of the token.

This module

Storing Tokens

	LocalStorage	SessionStorage	Cookie	In Memory
Persists on browser close	Y		Optional	Via silent auth
Persists on tab close	Y		Y	Via silent auth
Persists across tabs	Y		Y	Via silent auth
Access from subdomains			Y	
Sent on all requests			Y	
Max size	2 - 10MB	5MB - unlimited	4K / domain	No limit
XSRF vulnerable			Y	
XSS vulnerable	Y	Y	Y w/out HttpOnly	Y
Supports secure flag			Y	
Supports HTTP only flag			Y	

We'll switch to this in the final module

Demo

Add login

Summary

Login complete!

Next Up: Logout, signup, and user profile