Journal of Recent Trends in Computer Science and Engineering (JRTCSE)
Vol. 12, No.5, November - December 2024, PP. 45-49.
ISSN: 2322-0872
DOI: https://doi.org/10.70589/JRTCSE.2024.5.6
https://jrtcse.com

User Identification Through First-Party Cookies

Sunil Kishor Pathak,
https://www.linkedin.com/in/sunilkpathak/

Abstract

First-party cookies play a pivotal role in web applications by enabling seamless user identification, authentication, and personalization. This article explores the technical implementation of first-party cookies in HTTP, detailing their creation and usage on both the server and client sides. It further discusses their role in enhancing user experiences, including techniques for secure data handling and personalization.

Keywords: First-party cookies, User authentication, HTTP cookie attributes, Java cookie implementation, Personalization through cookies, Secure cookie practices, Cookie management, Data encryption in cookies

Citation: Sunil Kishor Pathak. (2024). User Identification Through First-Party Cookies. *Journal of Recent Trends in Computer Science and Engineering*, 12(5), 45-49.
DOI: https://doi.org/10.70589/JRTCSE.2024.5.6

1. Introduction

Cookies are small pieces of data stored on the client's browser, facilitating communication between the server and the client. First-party cookies, in particular, are set by the domain of the website being visited and are integral to user identification and session management. This article examines the technical implementation of first-party cookies using HTTP headers and Java, their retrieval and inspection on the client side, and their applications in user personalization.

2. What is a Cookie in HTTP?

A cookie is a key-value pair sent by the server to the client using the Set-Cookie HTTP header. The client stores this data and sends it back to the server with subsequent requests via the Cookie header. Attributes like Max-Age, Domain, HttpOnly, Secure, Path, and SameSite control the behavior and scope of cookies.

Explanation of Attributes:

- **Max-Age**: Specifies the lifetime of the cookie in seconds from the time it is set.
- **Domain**: Defines the host to which the cookie will be sent.
- **HttpOnly**: Ensures the cookie is not accessible to client-side JavaScript, enhancing security e.g. session token which is not expected to be read on client side.
- **Secure**: Ensures the cookie is only sent over HTTPS connections.
- **Path**: Defines the scope of the cookie within the website (e.g., / makes it available across all pages or /doc will available for /doc only).

- **SameSite**: Controls cross-site sharing of cookies. For first-party cookies, SameSite=Strict is typically used to prevent cross-site usage. Other values are lax or none.

3. Setting a First-Party Cookie in Java

The following Java code demonstrates how to set a first-party cookie using the Set-Cookie header:

```
public Response setUserCookie(int id) {
  String userName = userMap.get(id);
  if (userName == null) {
    return Response.status(Response.Status.NOT_FOUND)
        .entity("User not found for id: " + id)
        .build();
  }
  String cookieHeader = "user=" + userName + "; Path=/; Max-Age=3600; HttpOnly; Secure; SameSite=Strict; Domain=localhost";

  return Response.status(Response.Status.OK)
      .entity("User cookie set successfully for id: " + id)
      .header("Set-Cookie", cookieHeader)
      .build();
}
```

Explanation:

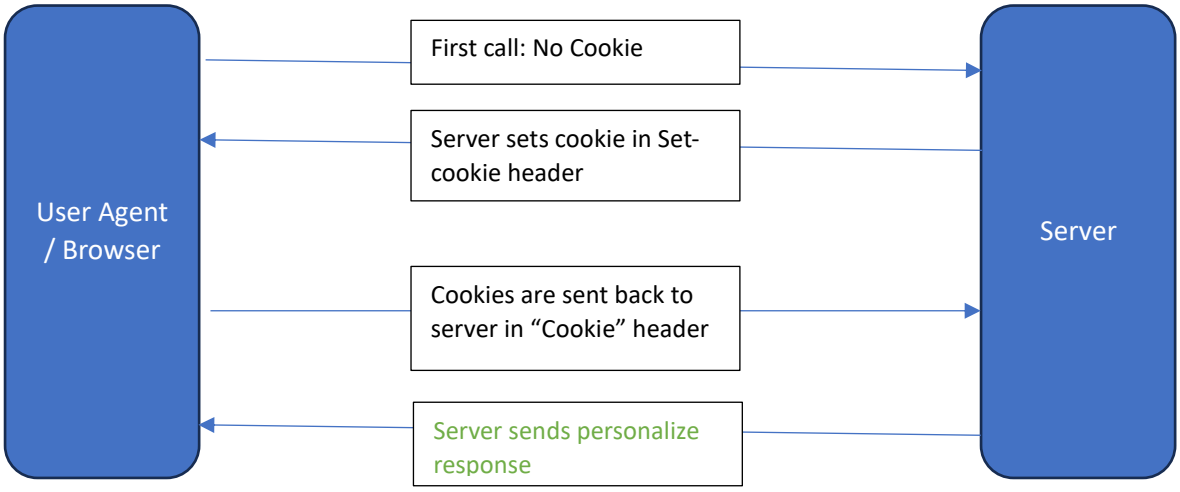
- **Max-Age**: Specifies the cookie's lifetime in seconds.
- **HttpOnly**: Prevents client-side scripts from accessing the cookie.
- **Secure**: Ensures the cookie is sent only over HTTPS.
- **SameSite**: Restricts cross-site sharing of cookies to Strict for first-party cookies.
- **Domain**: Specifies localhost as the cookie's domain.

4. Retrieving and Verifying Cookies

Cookies sent by the client are included in the Cookie header of the HTTP request. The following code extracts and verifies cookies:

```
public Response verifyCookie(HttpServletRequest request) {
  String cookieHeader = request.getHeader("Cookie");

  if (cookieHeader == null || !cookieHeader.contains("user=")) {
```

```
        return Response.status(Response.Status.BAD_REQUEST)
                .entity("No valid user cookie found")
                .build();
    }

    String[] cookies = cookieHeader.split(";");
    String userName = null;
    for (String cookie : cookies) {
        if (cookie.trim().startsWith("user=")) {
            userName = cookie.split("=")[1].trim();
            break;
        }
    }

    if (userName == null) {
        return Response.status(Response.Status.BAD_REQUEST)
                .entity("User not found in cookie")
                .build();
    }

    return Response.status(Response.Status.OK)
            .entity("Cookie verified successfully. User: " + userName)
            .build();
}
```



5. Visualizing First-Party Cookies in Chrome

After setting the cookie, you can inspect it in the browser:

1. Open Chrome Developer Tools (Ctrl+Shift+I / Cmd+Opt+I).
2. Navigate to the **Application** tab and select **Cookies** under **Storage**.
3. Choose the domain to view stored cookies, including their attributes.

6. Using First-Party Cookies for Personalization

First-party cookies can store user-specific data, such as email addresses, first name, last name, etc., for personalization. When next time the user visits, this encoded cookie will come to the server, the server will decode it to retrieve the first name and last name, and use it to greet the user before they log in.

Example Use Case:

- A user logs in with their email and name. The server encodes this data using a secure key and stores it in a cookie.
- On subsequent visits, the server decodes the cookie to personalize the user experience.

Code Snippet:

String encodedValue = Base64.getEncoder().encodeToString(("email=" + email + ";" + "name=" + "Sunil Pathak").getBytes());

String cookieHeader = "userData=" + encodedValue + "; Path=/; Max-Age=3600; HttpOnly; Secure; SameSite=Strict";

7. Security Considerations

- **Encoding and Encryption**: Always encode or encrypt sensitive data before storing it in cookies to prevent misuse.
- **Secure Attribute**: Ensure cookies are transmitted only over HTTPS.
- **HttpOnly Attribute**: Prevent client-side JavaScript access.
- **Expiration Management**: Use Max-Age judiciously to limit cookie lifespan.

8. Conclusion

First-party cookies are essential for implementing secure and seamless user identification. By combining them with best practices for security and personalization, developers can enhance user experiences while maintaining robust data protection.

References

1. Mozilla Developer Network (MDN). "HTTP Cookies." https://developer.mozilla.org/
2. RFC 6265. "HTTP State Management Mechanism." https://tools.ietf.org/html/rfc6265
3. Google Developers. "Chrome DevTools Overview." https://developers.google.com/web/tools/chrome-devtools

4. "SameSite Cookie Changes." https://www.w3.org/2020/Talks/chrome-samesite-20200416.pdf

5. Base64 Encoding in Java. "Java 8 Encoding and Decoding." https://docs.oracle.com/