Deterministic URI Encoding

http://tools.ietf.org/id/draft-montenegro-httpbis-uri-encoding/

HTTPbis WG, IETF 89, London
5 March, 2014

Osama Mazahir
Dave Thaler
Matthew Cox
Gabriel Montenegro
(Microsoft)
Background

• “http” and “https” URI schemes don’t have a fixed character encoding
• URI RFC [RFC3986] on generic syntax for URI components:
  1. Legacy URI components (before 2005) tend to use UTF-8 “or some other superset of the US-ASCII character encoding”
  2. New schemes (after 2005) use UTF-8 with percent encoding for reserved characters.
• Unfortunately, “http” and “https” schemes are not new schemes (#2), so their character encoding is not fixed (#1).
• Different URI components have different implications.
  • Host: uses A-labels per IDNA rules [RFC5890]
    • ok
  • Path and Query: no fixed character encoding, and both don’t necessarily use the same one
    • Not ok
### Current State


<table>
<thead>
<tr>
<th>Path Encoding</th>
<th>MSIE</th>
<th>FF2</th>
<th>FF3</th>
<th>Safari</th>
<th>Opera</th>
<th>Chrome</th>
<th>Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Request URL when following plain links</td>
<td>UTF-8</td>
<td>page encoding</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
</tr>
<tr>
<td>In Request URL for XMLHttpRequest calls</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
</tr>
<tr>
<td>In Request URL for manually entered URLs</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query String Encoding</th>
<th>MSIE</th>
<th>FF2</th>
<th>FF3</th>
<th>Safari</th>
<th>Opera</th>
<th>Chrome</th>
<th>Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Request URL when following plain links</td>
<td>page encoding, no escaping</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
<td>page encoding</td>
</tr>
<tr>
<td>In Request URL for XMLHttpRequest calls</td>
<td>page encoding, no escaping</td>
<td>page encoding</td>
<td>page encoding</td>
<td>mangled</td>
<td>page encoding</td>
<td>mangled</td>
<td>mangled</td>
</tr>
<tr>
<td>In Request URL for manually entered URLs</td>
<td>transcoded to 7-bit</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>UTF-8</td>
<td>stripped to ?</td>
<td>UTF-8</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>
Without Deterministic URI Character Encoding

- Browsers often use the charset of the containing HTML page
  - URI is pointed to from different pages using different encodings
  - The server linked to will see URIs with different encodings

- Problematic when parsing URIs at the server side or at intermediate proxies (e.g., when looking for a cache hit).

- URI parsing currently may involve trying different possible character encodings searching for a match.

- This represents a potential attack vector [RFC6943]
  - Possibility of unintended consequences.

- To mitigate this: Deterministic interpretation of data within a URI.
Proposal

• Enable character encoding indication (charset *before* percent encoding)

• Example:
  • If path was formed from percent-encoded UTF-8 then add header
    URI-Path-Encoding: UTF-8
  • If query was formed from percent-encoded UTF-8 then add header
    URI-Query-Encoding: UTF-8

• Absence of header: legacy behavior
• Unrecognized charset: legacy behavior