Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 9, 2018

L. Barguno
Google

August 08, 2017

Geolocation Header for HTTP over a Secure Context
draft-luisbarguno-geolocation-header-00

Abstract

The Geolocation header introduces a mechanism to send a device location over an HTTP Secure Context from a user agent to a server.

The Geolocation-Request header is used by a server to inform the user agent when a Geolocation header is requested to be sent.

This mechanism, through persistent Geolocation-Requests, provides a single-roundtrip solution to obtain location-aware responses for location-aware services, as oposed to existing JS-based Geolocation API that require two round trips.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 9, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Geolocation headers provide a mechanism to request and share the device location in an HTTP Secure Context.

The Javascript W3C Geolocation API [1] is the only existing mechanism to share a location with a host. This leads to some limitations. First, in order to have the server know the client's location there are two full roundtrips required (one roundtrip to load the page with Javascript code, and a second roundtrip to actually send the location to the server and get back a location-aware response). While not as significant as the first limitation, the second limitation is that the client must execute Javascript in order to acquire location.

Having a performant and safe mechanism to share a fresh device location, with and without a client being able to run Javascript, is increasingly necessary, especially on mobile devices.

The proposal detailed in this document is purely based on HTTP headers, and guarantees a single roundtrip with hosts that already have permission to access the device location.

2. Example

Header sent from the server to the user agent when a host is compatible with the Geolocation Header and is requesting geolocation to be attached in subsequent requests.

```
Geolocation-Request: Path="/localService"; Type=IfAlreadyGranted;
Expires=Thu, 18 Dec 2017 12:00:00 UTC
```

Header sent from user agent to server to share a location over a Full Secure Context [2] when permissions are granted, and the request Path matches a valid previously requested Geolocation-Request.

```
Geolocation: Position=[47.368684, 8.535741, 345]; Accuracy=10;
Timestamp=1495804846156; AltitudeAccuracy=20; Speed=1.5;
Heading=27.53;
```

3. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

4. Geolocation Header

A user agent request MAY include a Geolocation header.

The header identifier used for this purpose is "Geolocation", and the content of the header MUST include the attributes Position, Accuracy and Timestamp (in this order). After the required attributes, it MAY also include optional attributes AltitudeAccuracy, Speed and Heading (in this order).

All attributes are separated by semicolon. Each attribute MUST be specified as Attribute=Value.

These attributes match the data exposed through the Javascript Geolocation API [3].

This header is used to send geolocation data from the user agent to the server.

4.1. Attributes

o Position

A position is an array of numbers and MUST always be provided. There MUST be two or three elements. The first two elements are longitude and latitude, in that order and using decimal numbers. Altitude or elevation in meters above sea level MAY be included as an optional third element. This position format is based on RFC 7946 [RFC7946]

o Accuracy

Accuracy MUST always be provided and represents the level of accuracy of the latitude and longitude components of the Position. It is specified in meters. The value of the accuracy attribute must be a non-negative decimal number.

o Timestamp

Time when Position was computed. It is a positive value, representing milliseconds since the UNIX epoch of 1 January 1970 at 00:00 UTC.

Optional attributes that MAY be included:

o AltitudeAccuracy

It represents the accuracy level of the altitude, and it is specified in meters. When altitude is not provided as part of the Position, this attribute MUST not be included. Otherwise, the value of this attribute MUST be a non-negative decimal number.

o Speed

Denotes the device's current speed, in meters per second. When provided, the value of the speed attribute MUST be a non-negative decimal number. If the device is not moving, the speed attribute MUST not be included.

o Heading

Represents the bearing, that is the direction of the device and is specified in degrees. The value MUST be a decimal number between 0 and 360, as the clockwise direction relative to the north. If the device is not moving, the heading attribute MUST not be included.

5. Geolocation-Request Header

A server response MAY include one or more Geolocation-Request headers in a single response.

The header identifier used for this purpose is "Geolocation-Request", and the content of the header MUST include the attributes Path and Type. It MAY also include the optional attribute Expires.

All attributes are separated by semicolon. Each attribute MUST be specified as Attribute=Value.

This header is used to send a geolocation request from the server to the user agent. A geolocation request will be scoped to a particular Path on the server, and this geolocation request state per-Path MAY be persisted and maintained by the user agent.

5.1. Attributes

o Path

The scope of a Geolocation request is limited to a particular path, controlled by the Path attribute, mimicking the Path attribute in Set-cookie RFC 6265 [RFC6265]. However, in the Geolocation-Request case, the Path attribute MUST always be provided.

The user agent will include the Geolocation Header in a request only if the Path portion of the request-uri matches (or is a subdirectory of) the Geolocation-Request Path attribute, where the %x2F ("/") character is interpreted as a directory separator.

Permission to access geolocation is granted or denied to an entire origin, rather than individual paths within an origin (see "Negotiation, Privacy and Security" section).

o Type

There are two possible values for this attribute: "IfAlreadyGranted" or "MayPrompt".

"IfAlreadyGranted" is used to specify that the user agent MUST not prompt the user for permission, and MUST only include a Geolocation header in requests matching the Path if the user already granted permission to share location with that origin in the past.

"MayPrompt" is used to specify that the user agent MAY actively prompt the user for permission to share the geolocation with the server's origin. When this Type of request is specified, the server

is telling the user agent that it is acceptable to prompt the user, but the user agent is not forced to do so. For example if the user agent already prompted the user for that origin in a recent request and the user denied sharing geolocation with the origin, the user agent can decide not to prompt this user for the same origin again. Also note that this option should never block the browser from working without Geolocation with the origin, since "MayPrompt" MUST not be interpreted as Geolocation is a requirement, but just as the browser MAY prompt the user if possible.

Optional attribute that MAY be included:

o Expires

This mimics the Expires attribute in Set-cookie RFC 6265 [RFC6265], and follows the same format and semantics. The Expires attribute indicates the maximum lifetime of the Geolocation Request, represented as the date and time at which the request expires.

This attribute is optional, and when this attribute is not provided, the lifetime of the request will be undefined and decided by the user agent.

5.2. Handling multiple headers

If a server response contains multiple Geolocation-Request headers, each MUST be handled independently by the user agent.

The scope of a Geolocation-Request is the Path attribute, so the latest Geolocation-Request for a given Path MUST always overwrite any older Geolocation-Request for that particular Path (regardless of Type).

5.3. Best effort

A Geolocation-Request should follow a best effort approach from the user agent. Computing location can be expensive, and it is recommended that the user agent does not block a request only because Geolocation is missing. It is recommended that Geolocation is only attached when it is already available to the user agent.

However, the user agent MAY internally precompute, cache, and keep geolocation information available and fresh, so when it is required to be attached in a server request, it is already available.

6. Negotiation, Privacy and Security

The data exposed in the Geolocation header is very sensitive information, and thereby potentially compromises the user's privacy. Any implementation of this technology must include the following mechanisms to guarantee the privacy of the user is protected.

6.1. User agent Considerations

- Ensure that no location information is made available through this header without the user's permission.

Permission must be acquired by the user agents through a user interface, which MUST expose at least the host name and include "Location" as the key piece of data being shared with that origin. The user agent MAY persist the user response and follow up requests MAY rely on older responses from the user for that particular origin, but before sending the Geolocation header, the user agent MUST guarantee that the latest response from the user for that origin was granting Location permission. When these decisions are persisted, the user agent MUST provide another user interface to change the decision later on.

Given that the data exposed through the Geolocation Header matches the data exposed through the W3C Javascript geolocation API, the user agent MAY use a consistent and unified permission-control mechanism for both location-sharing technologies, which is recommended for user clarity. In particular, it is recommended to rely on the state of the geolocation permission [4].

Since different paths within a single origin can access each other's' data, the permission to get location MUST be granted to the origin (domain+port+scheme, not a specific Path).

- Ensure that location information is only sent when the Path matches a valid Geolocation-Request.

A user agent MUST only include a Geolocation header when permission has been granted and the server registered a Geolocation-Request in the past that has a matching Path and is still valid (not expired).

- Only send Geolocation header on Secure Contexts.

The Geolocation header will only be included in Full Secure Contexts with the authorized origin, and MUST not be sent in a non-encrypted connection, so the user privacy is protected.

- Be transparent that the location information is sent

Due to the sensitivity of the data, the user agent MUST include a visual indicator informing the user that a location information was shared. The nature of the indicator is not mandated.

6.2. Server considerations

- Location information MUST be requested only when necessary.

The server MUST only send a Geolocation-Request header when it offers location-aware services, and the Path attribute must be used to limit the scope of the request, to only relevant services.

- Location data MUST be handled responsibly and transparently.

The server MUST only use location information for the service for which it was provided by the user agent. Location data MUST be protected from unauthorized access by third parties. It MUST also be disposed once the task is completed, or retained according to terms of service acknowledged by the user. When this data is stored, users should have access to it and the right to delete it at anytime.

The server MUST disclose the following: The fact that they are collecting location data, why they are collection location data, security policy for location data, retention policy of location data, whether this location data is shared and how, and explain to the user how to access and delete this data on the server.

7. Header field registration

The message header fields should be added to the permanent registry RFC 3864 [RFC3864]: Geolocation and Geolocation-Request.

The applicable protocol is HTTP. Format specification is detailed in this document.

Geolocation and Geolocation-Request are defined using US-ASCII.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, DOI 10.17487/RFC3864, September 2004, <http://www.rfc-editor.org/info/rfc3864>.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, DOI 10.17487/RFC6265, April 2011, <http://www.rfc-editor.org/info/rfc6265>.

[RFC7946] Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S., and T. Schaub, "The GeoJSON Format", RFC 7946, DOI 10.17487/RFC7946, August 2016, <http://www.rfc-editor.org/info/rfc7946>.

8.3. URIs

[1] https://www.w3.org/TR/geolocation-API/

[2] https://w3c.github.io/webappsec-secure-contexts/

[3] https://www.w3.org/TR/geolocation-API/

[4] https://w3c.github.io/permissions/#reading-current-states

Author's Address

Luis Barguno Jane
Google
Brandschenkestrasse 110
Zurich 8002
Switzerland

Email: lbargu@google.com