Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 20, 2012

A. Petersson
M. Nilsson
Opera Software

November 17, 2011

Forwarded HTTP Extension
draft-petersson-forwarded-for-02

Abstract

This document standardizes an HTTP extension header field that allows proxy components to disclose information lost in the proxying process, e.g. the originating IP number of a request or IP number of the proxy on the incoming side. Given a trusted path of proxying components, each subsequent component will have access to all IP numbers used in the chain of proxied HTTP requests.

This document also standardizes ways for a proxy administrator to anonymize the origin of a request.

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 May 20, 2012.

Copyright Notice

Copyright (c) 2011 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 3
2. Notational Conventions 3
3. Syntax Notations 3
4. Forwarded 4
5. Parameters 5
5.1. Forwarded by 5
5.2. Forwarded for 6
5.3. Forwarded host 6
5.4. Forwarded proto 6
5.5. Private extensions 6
5.6. Future extensions 6
6. Node identifiers 6
6.1. IPv4 and IPv6 identifiers 7
6.2. The "unknown" identifier 7
6.3. The "hidden" identifier 8
6.4. Obfuscated identifier 8
7. Implentation considerations 8
8. Security Considerations 8
8.1. Header validity 8
8.2. Information Leak 9
9. IANA Considerations 9
10. References 9
10.1. Normative references 9
10.2. Informative references 10
Appendix A. Forwarded BNF definition 10
Appendix B. Change Log (to be removed by RFC Editor before publication) 11
B.1. Since draft-petersson-forwarded-for-00 11
B.2. Since draft-petersson-forwarded-for-01 11
Authors' Addresses 11

1. Introduction

In today's HTTP landscape, there are a multitude of different applications acting as a proxy for the user agent and effectively anonymizing the requests to look as if they originated from the proxy IP number or in other ways changing the information in the original request. Examples of such applications include caching, content filtering, content compression, crypto offload, and load balancing. As most of the time this destructive behavior is not the primary purpose, or even a desired effect, a way of disclosing the original information on HTTP level instead of depending on the TCP/IP connection remote IP number is needed.

Except for the above mentioned problems, there may also be issues due to the use of NAT. This is further discussed in [RFC6269].

A common way to disclose this information is by using the de facto standard header fields such as X-Forwarded-For, X-Forwarded-By, and X-Forwarded-Proto. This document intends to standardize syntax and semantics for disclosing such information. The header field also combines all information within one single header field, making it possible to correlate the header fields to each other. With the header field format described in this document, it is possible to know what information belongs together, given that the proxies are trusted. Such conclusions are not possible to make with the X-Forwarded class of header fields. This new header field also extends the de facto standard of, e.g., X-Forwarded-For with features for which real life deployments have shown a need.

2. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Syntax Notations

This specification uses the augmented BNF notation defined in Section 2.1 of [RFC2616], including its rules for implied linear whitespace (LWS).

The BNF rules for "IPv6address" and "IPv4address" are defined in [RFC3986] The IPv6address SHOULD comply with textual representation recommendations [RFC5952] (e.g., lowercase, zero compression).

4. Forwarded

The Forwarded HTTP header field is an optional header field that, when used, contains a list of parameter-identifier pairs that disclose information that is altered or lost in a proxy. This applies to forwarding proxies, as well as reverse proxies. Information passed in this header can be, e.g., the source address of the request, the address of the incoming interface on the proxy, or whether HTTP or HTTPS is used. If the request is passing through several proxies, each proxy MAY add any parameters, it MAY also remove earlier added Forwarded-header fields.

The top-level list is represented as a list of HTTP header field-values [RFC2616]. The leftmost element in this list holds information added by the first proxy, followed by information added by any subsequent proxy. Each field-value is a semicolon-separated list, this sub-list consists of parameter-identifier pairs. Parameter-identifier pairs are grouped together by an equals sign. The header field can be defined in augmented BNF syntax as

```
    Forwarded: = "Forwarded" ":" LWS Forwarded-v
    Forwarded-v = 1#forwarded-element

    forwarded-element =
            OWS forwarded-value *( OWS ";" OWS forwarded-value ) OWS

    forwarded-value   = for-kv | by-kv | proto-kv | host-kv | ext-kv

    for-kv     = "for=" node
    by-kv      = "by="  node
    proto-kv   = "proto=" proto-name
    host-kv    = "host=" host
    proto-name = ALPHA *( ALPHA | DIGIT | "+" | "-" | "." )
```

Example:

```
    Forwarded: for=192.0.2.43,for=[2001:db8:cafe::17]
    Forwarded: proto=https;by=198.51.100.60
```

Given that a proxy wishes to add a Forwarded header field to the outgoing request, if the incoming request has no such header field, the proxy simply adds the header with the list of parameters desired. If, on the other hand, the incoming request has such a header field, the proxy simply adds a comma and the list of parameters a proxy MAY remove all Forwarded header fields from a request. It MUST, however, ensure that the correct header field is updated in case of multiple Forwarded header fields.

Example: A request from a client with IP number 192.0.2.43 passes through a proxy with IP number 198.51.100.17, then through another proxy with IP number 203.0.113.60 before reaching a origin server. This could, for example, be an office client behind a corporate malware filter talking to a origin server through a reverse proxy.

- The HTTP request between the client and the first proxy has no Forwarded header field.
- The HTTP request between the first and second proxy has a "Forwarded: for=192.0.2.43" header field.
- The HTTP request between the second proxy and the origin server has a "Forwarded: for=192.0.2.43, for=198.51.100.17;by=203.0.113.60;proto=http;host=example.com" header field.

Note that, at some points in a connection chain, the information might not be correctly updated in the Forwarded header field, either because of lack of support of this HTTP extension or because of a policy decision not to disclose information about this network component.

5. Parameters

Valid parameters are as follows:

- "by" identifies the incoming interface of the proxy
- "for" identifies the node making the request to the proxy
- "host" is the host request header-field as received by the proxy
- "proto" tells if the request was made in plain text or encrypted

5.1. Forwarded by

The "by" parameter is used to disclose the interface where the request came in to the proxy server. Typically, the value of this parameter is an IP-address, but it can, however, be some other kind of identifier. The parameter value MUST, however, be a node identifier as described in Section 6. This is primarily added by reverse proxies that wish to forward this information to the backend server.

5.2. Forwarded for

The "for" parameter is used to disclose information about the user agent that initiated the request. Typically the value of this parameter is an IP-address, but it can also be some other kind of identifier. The parameter value MUST, however, be a node identifier, as described in Section 6. In a chain of proxy servers where this is fully utilized, the first for-parameter will disclose the user agent where the request first was made, followed by any subsequent proxy identifiers. The last proxy in the chain is not part of the list of for-parameters. The last proxy's IP address is, however, readily available as the remote IP address of the TCP/IP connection.

5.3. Forwarded host

The "host" parameter is used to forward the original value of the "Host" header field. This can be used by the origin server if a reverse proxy is rewriting the "Host" header field to some internal host name.

5.4. Forwarded proto

The "proto" parameter has the value of the used protocol type. If present, it MUST contain the URI schema name as defined in section 3.1 in [RFC3986] and registered to IANA according to [RFC4395]. Typical values are "http" or "https". For example, in an environment where a reverse proxy is also used as a crypto offloader, this allows the origin server to rewrite URLs in a document to match the type of connection as the user agent requested, even though all connections to the origin server are unencrypted HTTP.

5.5. Private extensions

Private extensions allow for adding own parameters and values. This may be particularly useful in a reverse proxy environment.

5.6. Future extensions

One may extend this RFC by writing new RFCs that define new parameters. IANA should be notified if an RFC is updating this RFC with new valid parameters.

6. Node identifiers

The node identifiers are the IP address of the network node, a predefined token hiding the real identity, but signaling that such a component exists in the network path, or a generated token allowing

for tracing and debugging without revealing network internals.

```
nodename = IPv4address | IP-literal |
           "unknown" | "hidden" | obfnode
```

All of the identifiers may optionally have the port identifier, for example, allowing the identification of the end point in a NATted environment.

```
node           = nodename [ ":" node-port ]
```

The node-port can be identified either by its TCP port number or by a generated token obfuscating the real port number. If a node-port is appended to an IPv6address, the IPv6address MUST be enclosed by square brackets.

```
node-port      = port | obfport
port           = 1*5DIGIT
obfport        = 1*(ALPHA | DIGIT)
```

Note that this also allows port numbers to be appended to the "hidden" and the "unknown" identifiers. Interpretation of such notation is, however, left to the possessor of a proxy adding such a value to the header field. To distinguish an obfport from a port, we RECOMMEND that an obfport always should contain at least one ALPHA.

6.1. IPv4 and IPv6 identifiers

The IPv4address and IPv6address BNF tokens are defined as follows:

```
IPv6address = "[" addr6 "]"
 addr6 = hexpart [ ":" IPv4address ]
IPv4address = 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT

hexpart = hexseq | hexseq "::" [ hexseq ] | "::" [ hexseq ]
hexseq  = hex4 *( ":" hex4)
hex4    = 1*4HEXDIG
```

Note that the IP number may be one from the internal nets, as defined in [RFC1918] and [RFC4193].

6.2. The "unknown" identifier

The "unknown" identifier is used when the identity of the preceding entity is not known. One example would be a proxy server process generating an outgoing request without direct access to the incoming request TCP socket.

6.3. The "hidden" identifier

The "hidden" identifier is used when the administrator of a proxy server entity would like to keep the identity of that entity secret but still disclose that it exists.

6.4. Obfuscated identifier

A generated identifier may be used where there is a wish to keep the internal IP numbers secret, while still allowing the Forwarded header field to be used for tracing and debugging. The identifiers can be randomly generated for each request and do not need to be statically assigned to resources. To distinguish the obfuscated identifier from other identifiers, it MUST have a leading underscore "_". Further, it MUST also consist of only US-ASCII letters and US-ASCII digits.

```
obfnode         = "_" 1*( ALPHA | DIGIT )
```

7. Implentation considerations

Note that an HTTP list allows white spaces to occur between the identifiers, and the list may be split over multiple header fields. As an example, the header field

```
Forwarded: for=192.0.2.43,for=[2001:db8:cafe::17],for=unknown
```

is equivalent to the header field

```
Forwarded: for=192.0.2.43, for=[2001:db8:cafe::17], for=unknown
```

which is equivalent to the header fields

```
Forwarded: for=192.0.2.43
Forwarded: for=[2001:db8:cafe::17], for=unknown
```

Also, note that the draft [I-D.ietf-httpbis-p1-messaging] renders the use of folding within a list obsolete. The use of CRLF within the field-value list is therefore NOT RECOMMENDED.

8. Security Considerations

8.1. Header validity

The Forwarded HTTP header field cannot be relied upon to be correct, as it may be modified, whether mistakenly or for malicious reasons, by every node on the way to the server, including the client making

the request.

One approach is to verify the correctness of proxies and whitelist them as trusted. This approach has at least two weaknesses. First, the chain of IP numbers listed before the request came to the proxy cannot be trusted. Second, unless the communication between proxies and the end point is secured, the data can be modified by an attacker with access to the network.

8.2. Information Leak

The Forwarded HTTP header field can reveal internal structures of the network setup behind the NAT or proxy setup, which may be undesired. This can be addressed either by preventing the internal nodes from updating the HTTP header field or by having an egress proxy removing entries that reveals internal network information.

9. IANA Considerations

This document specifies the HTTP header listed below, which should be added to the permanent HTTP header registry defined in [RFC3864].

Header field: Forwarded
Applicable protocol: http/https
Status: standard
Author/Change controller:
    IETF (iesg@ietf.org)
    Internet Engineering Task Force
Specification document(s): this specification (Section 4)
Related information: none

10. References

10.1. Normative references

[I-D.ietf-httpbis-p1-messaging]
Fielding, R., Gettys, J., Mogul, J., Nielsen, H., Masinter, L., Leach, P., Berners-Lee, T., Lafon, Y., and J. Reschke, "HTTP/1.1, part 1: URIs, Connections, and Message Parsing", draft-ietf-httpbis-p1-messaging-17 (work in progress), October 2011.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC4395] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, August 2010.

10.2. Informative references

[RFC6269] Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, June 2011.

Appendix A. Forwarded BNF definition

This appendix defines the Forwarded header field.

```
Forwarded: = "Forwarded" ":" LWS Forwarded-v
Forwarded-v = 1#forwarded-element

forwarded-element =
        OWS forwarded-value *( OWS ";" OWS forwarded-value ) OWS

forwarded-value   = for-kv | by-kv | proto-kv | host-kv | ext-kv

for-kv     = "for=" node
by-kv      = "by="  node
proto-kv   = "proto=" proto-name
host-kv    = "host=" host
ext-kv     =  extension "=" ext-value

node = nodename [ ":" node-port ]
nodename = IPv4address | IP-literal |
  "unknown" | "hidden" | obfnode
obfnode = "_" 1*( ALPHA | DIGIT )
node-port = port | obfport
port = 1*5DIGIT
obfport = 1*( ALPHA | DIGIT )
proto-name = ALPHA *( ALPHA | DIGIT | "+" | "-" | "." )
extension = 1*( ALPHA | DIGIT | "-" )
ext-value = <any OCTET except CTLs, but including SP>
```

Appendix B. Change Log (to be removed by RFC Editor before publication)

B.1. Since draft-petersson-forwarded-for-00

Added IANA considerations.

Expanded scope and add parameterized list.

B.2. Since draft-petersson-forwarded-for-01

Removed "x-" from private extensions.

Allow for any protocol name.

Rename kv-v to forwarded-element and kv to forwarded-value.

Add informative reference RFC6269.

Authors' Addresses

Andreas Petersson
Opera Software
S:t Larsgatan 12
Linkoping SE-582 24

Email: pettson@opera.com

Martin Nilsson
Opera Software
S:t Larsgatan 12
Linkoping SE-582 24

Email: nilsson@opera.com