INTERNET-DRAFT
Intended Status: Experimental
Expires: March 29, 2019

P.Chowdaiah
(ariksh software)

September 25, 2018

Method to pre-fetch Domain Names at HTTP Proxy Servers
draft-pchowdaiah-prefetch-dns-query-over-http-00

Abstract

This specification offers an approach for augmenting domain name resolution times by intercepting HTTP responses at HTTP Proxy servers and triggering DNS queries to the servers in lieu of user requests and returning responses without an explicit requests from clients. This functionality can be achieved by multiple approaches at HTTP and Domain Name System [DNS] Protocols, this document specifies approach that shall be employed at HTTP protocol [RFC 2616]. Methods/approaches pertaining to DNS shall be documented in [draft-pchowdaiah-dns-without-explicit-query.txt].

Table of Contents

1. Introduction

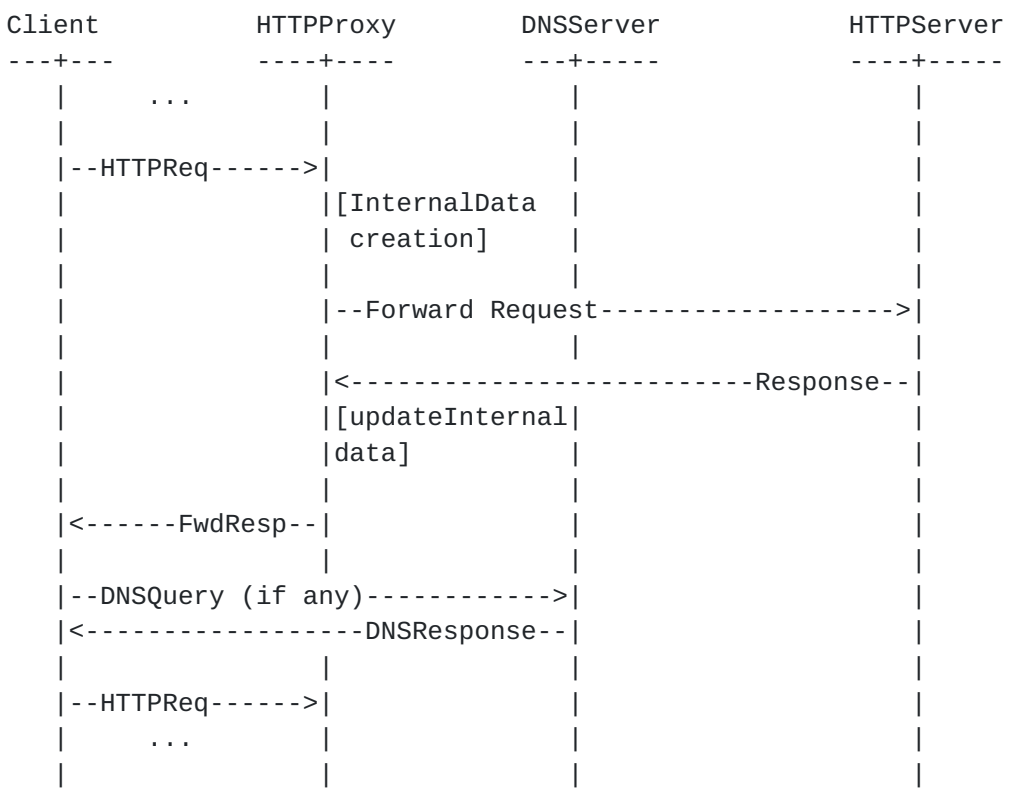
HTTP proxy servers have information that can be utilized to augment the resolution times of Domain names which could increase the speed of loading web pages on account of domain name resolution being pre resolved by the proxy servers and responses being sent before explicit requests from domain name clients. This document specifies a method to send DNS responses by pre-negotiating a time bound transport layer [TCP and UDP] connection and sending responses on this connection when HTTP Proxy server parses HTTP responses received for an earlier HTTP request.

1.1. Terminology

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 RFC 2119 [RFC2119].

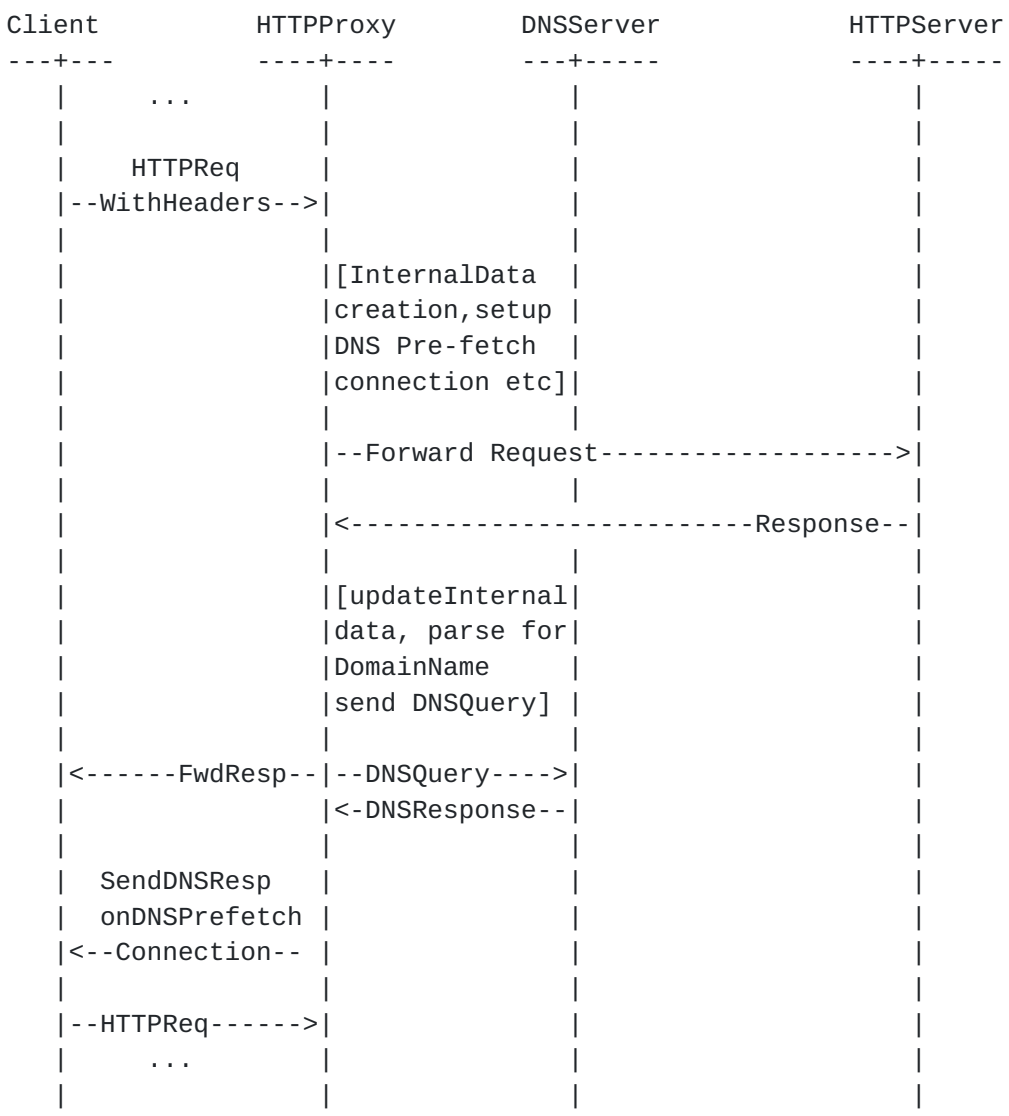
2. Functionality Overview

[Figure 1] shows high-level generic behavior in HTTP proxies, which is summarized here: A web Browser Client sends HTTP Protocol requests, (transport layer connections are not covered here), the request is received at Proxy, proxy interprets and parses requests into its internal data structures, forwards request to the destination server, on receiving responses from server, proxy updates internal data structures, might cache responses and forwards response back to Browser client.



[Figure 1]

[Figure 2] shows overview of proposed approach: A web browser client sends HTTP Protocol requests, request is received at proxy, proxy parses the requests and does bookkeeping into its internal data structures, forwards request to destination server, on receiving responses from server, proxy parses responses and updates internal data structures and looks for domain name [RFC 1034] queries inside the responses and forwards it to proxy DNS resolver co-located with HTTP proxy implementations, then forwards the received HTTP responses to browsing client. On receiving the DNS query responses, DNS module co-located in the HTTP proxy module constructs agreed implementation specific responses towards the client on the transport layer connection known as 'DNS Pre-fetch Connection' henceforth with pre-negotiated port and protocol specified in the HTTP requests received from the browser client which could be configured or dynamically generated. The browser client then constructs the HTTP requests as per received DNS resolution and proceeds with its activities, thereby avoiding explicit DNS requests. Detailed procedure is documented in section "Detailed Specification"



[Figure 2]

3. Detailed Specification

Today most of the HTTP proxy designs either transparent or non-transparent probably are proprietary implementations that are based on the knowledge gained through experience and deployments. The approach presented here is also an outcome of this knowledge. Some of the HTTP proxy implementations have full knowledge of the HTTP request and responses as they snoop into the messages that are exchanged between the client and server. This document's approach is to use this information available at the proxies to augment the domain name resolution times by pre-requesting the DNS queries for domain names in the HTTP responses being exchanged, the proxy might cache responses if domain names are found to spill over-to next

response packets. If the responses are encrypted then pre-requesting domain names shall be ignored and the responses are just forwarded. Here are the detailed steps that are involved:

a. This functionality shall be controlled at the client side with configuration options which needs to be set to have this functionality communicated to the proxies through HTTP message header field. This new HTTP message header field are detailed in section [3.2]. New message headers contain magic security cookie, transport and network layer information on protocol, port, Internet protocol address, timeout for connection keep-alive that client would listen on. The client would setup transport connections ('DNS Pre-fetch Connection') that it shall be listening on with parameters that it shall advertise. Network Address Translation (NAT)-Application Layer gateways (ALG) implementations shall need to be updated for the new functionality.

b. Received HTTP request at the proxy is parsed, internal bookkeeping carried out, an optional implementation ACK can be sent back on the connection that client has advertised in item [a]. Received request is forwarded to destination server sans the header specific to this documented functionality. If destination server receives a request having the new HTTP message headers, it needs to do a no-operation on it as per [RFC 2616].

c. On receiving HTTP response from the server, session identified by comparing the book kept data structures. Domain names are retrieved from the parsed responses, limited responses are cached if domain names span across multiple packets, domain names resolution is triggered to co-located DNS server. The HTTP responses is forwarded to the client.

d. Domain query responses received from co-located DNS server is forwarded on the connection to the client. Agreement on format of the message that is exchanged would be implementation specific. Implementations shall define formats of message types and information that it needs to be exchanged over DNS-prefetch-connection. Approaches shall include either limited exchanging of specific information from query response or approaches described in [draft-pchowdaiah-dns-without-explicit-query.txt]

e. Client with response received on the connection setup for this functionality shall use the domain name resolution information and send out new HTTP requests.

3.1. HTTP Header Fields

3.2. HTTP Request Header Fields

3.2.1. Dns-prefetch-over-http

The Dns-prefetch-over-http request header field communicates connection parameters.

```
Dns-prefetch-over-http  = "Dns-prefetch-over-http" ":"
                          1#( prefetch-params )

prefetch-params = "magic-cookie" ";" "hostIpAddress" ";" "protocol"
                  ";" "port" ";"  "connection timeout"
                  ";"  "response timeout"
```

where:

```
  magic-cookie       : dynamically generated session security
                       authenticator
  hostIpAddress      : Internet protocol address of client
  protocol           : transport layer protocol TCP and UDP
  port               : the port socket listening on
  connection timeout : keep-alive timeout in seconds
  response timeout   : time client waits for pre-fetch DNS responses
                       before initiating standard DNS query
```

Example of its usage:

```
 Dns-prefetch-over-http : MAGICCOOKIE;192.168.1.1;UDP;25001:60;500\r\n
```

4. Security Considerations

Any cookie related authentication mechanisms employed shall create security pitfalls. This proposed method shall not have any unknown security issues other than issues similarly identified for the HTTP standard cookie related HTTP header field though both are unrelated.

5. IANA Considerations

This specification being experimental shall not request any updates to any standard or registry. However upon review if this specification acquires standard status, this specification proposes and requests the following HTTP fields to be included in HTTP message header field registry [see RFC3864]:

5.1. Dns-prefetch-over-http

Header field name: Dns-prefetch-over-http

Applicable protocol: http

Status: provisional

Author/Change controller: this specification

Specification document: this specification (Section 3.2.1)

6. Implementation Source Code

Proof-of-concept source code shall be available on github at this web link <TODO: add link after resolving open source, copyright license issues>

7. References

7.1. Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

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

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, June 2014.

[RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", RFC 7235, June 2014.

7.2. Informative References

[RFC1919] Chatel, M., "Classical versus Transparent IP Proxies", RFC 1919, March 1996.

[RFC1945] Berners-Lee, T., Fielding, R., and H. Nielsen, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996.

[RFC2068] Fielding, R., Gettys, J., Mogul, J., Nielsen, H., and

T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, January 1997.

[RFC2145] Mogul, J., Fielding, R., Gettys, J., and H. Nielsen, "Use and Interpretation of HTTP Version Numbers", RFC 2145, May 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.

7.3. Notice

Please forward any comments and feedback to parikpub@gmail.com

Authors' Addresses

Parikshith Chowdaiah
ariksh software technologies
1578/A, BSK II Stage
Bangalore 560070
India

Email: parikpub@gmail.com