Internet Engineering Task Force	M. Lentczner	
Internet-Draft	Linden Research, Inc.	
Intended status: Informational	D. Preston	
Expires: September 5, 2009	March 04, 2009	

TOC

Reverse HTTP
draft-lentczner-rhttp-00

Status of this Memo

Copyright Notice

Abstract

This memo explains a method for making HTTP requests to a host that cannot be contacted directly. Typically, such a host is behind a firewall and/or a network address translation system.

Table of Contents

---

1. Introduction

TOC

HTTP has become widely used as an application layer transport. Systems such as XML-RPC and SOAP, as well as numerous "web APIs" such as the Flickr API, all use HTTP as a method for having a client make requests of a server.

The design of HTTP is such that the entity that initiates the TCP connection over which HTTP flows is also the entity that issues the request. When building applications over HTTP where the hosts involved can both function as TCP servers, this is no problem: When either host wishes to invoke an function on the other, it can initiate a TCP connection to the other, and acting as the client in the HTTP protocol, issue the request.

In many HTTP based applications, one of the hosts cannot function as an HTTP server. Typically, the host may be behind a firewall, or not have a publicly routable IP address. In this case, application layer interfaces can no longer be symmetric: If a host needs to invoke a function on a host that cannot function as an HTTP server, than that function cannot be easily layered on top of HTTP. Typical designs to work around this constraint involve different encodings and semantics for requests from the server host to the non-server host, and the use of the long-poll technique.

Reverse HTTP is designed to enable the application layer encoding and semantics to be built upon HTTP uniformly for requests in either direction between a non-server host and server host. Since the non-server host cannot be contacted directly, the non-server host still needs to establish the TCP connection, but once the Reverse HTTP

protocol is negotiated, the full HTTP protocol is used in the reverse of the normal direction: from the server to the client

---

1.1. Requirements Language

TOC

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 (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.) [RFC2119].

---

2. Protocol

TOC

2.1. Overview

TOC

A Reverse HTTP connection is established between hosts A and B as follows:

1. Host A establishes a TCP connection to host B.

2. Host A makes an HTTP request to host B. This request indicates A's willingness to engage in Reverse HTTP, and supplies any identification, authentication and authorization needed by B. The request is signaled by the Upgrade header in the request. Typically, the request is also indicated by a particular URL, and the other information encoded in the URL, body, or cookies, but applications are free to use any methods they wish for these purposes.

3. Host B signals its agreement to enter into Reverse HTTP by use of a 101 status code and the Upgrade header in its response. At the conclusion of this response, the original HTTP connection is considered abandoned, and a new HTTP connection is established where host B is in the role of client and host A in the role of server. From then on, request messages can be sent from B, with responses from A.

TOC

2.2. Upgrade

The switch to Reverse HTTP is signaled via HTTP's Upgrade header and upgrade mechanism, as described in sections 14.42 and 10.1.2. of RFC2616 (Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," June 1999.) [RFC2616].
The protocol token used in the Upgrade header is:

```
PTTH/1.0
```

Section 10.1.2 of HTTP (Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," June 1999.) [RFC2616] defines the precise point of the protocol switch: Directly after the empty line of the response message containing the Upgrade header. At this point, the new connection proceeds with the full HTTP protocol, as if the TCP connection had just been established, but with the parties in reverse: the original server acting as the HTTP client, sending request messages, and the original client acting as the HTTP server, sending response messages.

2.3. Example

TOC

In this example, host A will contact host B, and indicate it's willingness to enter into Reverse HTTP, and then process request messages from B:

```
A --> B: POST /message-queue?id=iamsam HTTP/1.1
         Host: b.example.com
         Upgrade: PTTH/1.0
         Connection: Upgrade

A <-- B: HTTP/1.1 101 Switching Protocols
         Upgrade: PTTH/1.0
         Connection: Upgrade

A <-- B: GET /status HTTP/1.1
         Host:

A --> B: HTTP/1.1 200 OK
         Content-type: text/plain
         Content-Length: 4

         Good
```

3. Considerations

TOC

3.1. Host Header

TOC

Section 14.23 of HTTP (Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," June 1999.) [RFC2616] requires that all HTTP/1.1 requests have a Host header. However, if the URI requested doesn't include an internet host, then the Host field is empty. In the case of Reverse HTTP, the URI requested refers to a resource on Host A, which does not have an host name or IP accessible to Host B. As such, the Host field in request messages from B SHOULD contain an empty Host field.

3.2. Connection

TOC

The Upgrade mechanism is defined to be a hop-by-hop level in HTTP. As such, the Upgrade header is required to be listed in a Connection header in both the request message and the response message. Reverse HTTP, however, is establishing an end-to-end establishment of an HTTP connection in the reverse direction. Thus, Reverse HTTP upgrade can only be used with a proxy if the proxy is willing to establish Reverse HTTP on the next hop, or the CONNECT verb is used with the proxy to first establish a tunnel. Where Reverse HTTP is used over HTTPS connections, there is no problem, even in the face of proxies, because those proxies establish an end-to-end tunnel that constitutes a single HTTP hop.

3.3. Persistent Connections

TOC

Once established, the HTTP connection operating in reverse is just like any other HTTP connection. In particular, the persistent connection mechanisms of both HTTP 1.0 and 1.1 are available. In this way, a host can establish a bi-directional HTTP based communication to another host using two TCP connections: One running a persistent HTTP connection to the remote host, and another, after negotiating Reverse HTTP, running a persistent HTTP connection from the remote host back to itself.

3.4. Fall Back

TOC

Applications can be developed where not all hosts have to understand Reverse HTTP. If the non-server host doesn't understand Reverse HTTP, then the server will not see an Upgrade token of PTTH/1.0 and so will know to fall back to some other application method of routing requests to the non-server host. If the server doesn't understand Reverse HTTP, then it will respond to the HTTP request, ignoring the Upgrade token. In such cases, the non-server host will see a response that corresponds to the fall back method.
The actual method of fall back is beyond the scope of this memo. The authors have experimented with various forms including JSON and XML encoding of application level requests, and more generic encoding of HTTP messages into HTTP message bodies. In general, existing applications can migrate to Reverse HTTP and keep their existing techniques as fall back.

---

4. Security Considerations

TOC

Reverse HTTP introduces no new security concerns beyond those known with HTTP. However, as it applies HTTP in a novel way, the common security concerns need to be applied to the parties in reverse. In particular, client software that initiates Reverse HTTP must realize that it will act as a server once the upgrade is complete, and so prepare it self for almost all the same issues any HTTP server must be concerned with. The security concerns can be lessened somewhat in that the client need not be prepared for arbitrary requests from the Internet. HTTP requests can only come from the host the client chooses to connect and establish Reverse HTTP to. Standard precautions, such as negotiating TLS or using HTTPS, should be use so that the client can rule out man-in-the-middle and impersonation attacks.

---

5. IANA Considerations

TOC

This document asks IANA to register the token PTTH in the HTTP Upgrade Token registry.

---

6. Normative References

TOC

[RFC2119]

	Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[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 (TXT, PS, PDF, HTML, XML).

Authors' Addresses

TOC

	Mark Lentczner
	Linden Research, Inc.
	945 Battery St.
	San Francisco, CA 94111
	US
Phone:	+1 415 243 9000
Email:	zero@lindenlab.com
	Donovan Preston
Email:	dsposx@mac.com