Network Working Group   M. Blanchet
Internet-Draft   Viagenie
Expires: December 30, 2002   July 1, 2002

Tunnel Setup Protocol (TSP): A Control Protocol to Setup IPv6 or IPv4 Tunnels
draft-vg-ngtrans-tsp-01

Status of this Memo

Copyright Notice

Abstract

This document proposes a control protocol to setup tunnels between a client and a tunnel server or broker. It provides a framework for the negociation of tunnel parameters between the two entities. It is a generic TCP protocol based on simple XML messaging. This framework protocol enables the negociation of any kind of tunnel, and is extensible to support new parameters or extensions. The first target application is to setup IPv6 over IPv4 tunnels which is one of the transition mechanism identified by the ngtrans and ipv6 working groups. This IPv6 over IPv4 tunnel setup application of the generic TSP protocol is defined by a profile of the TSP protocol, in a companion document.

Table of Contents

1. Rationale for a tunnel setup protocol

Tunnelling techniques are often used to enable new networking functions while still preserving the underlying network as is. Configuring tunnels means handling many static parameters (IP address of the end-points, address or overlay network info), which is a tedious task for a network manager for a large number of tunnels. Some of those parameters may change over time, like the IPv4 address of a client node, which means changing the configuration on the other end.

A tunnel broker model (RFC3053) [1] has been defined in the context of IPv6 over IPv4 tunnels, where the tunnel broker enables the use of tunnels from a client using a web interface to tunnel servers. Attempts have been made to generalize the idea using a MIME-type [7], but still no protocol has been defined to enable the negociation of parameters over time for a given tunnel. This draft generalize the concept of the tunnel broker model by having a control protocol between the broker and the client. It enables negociation between the two parties: prefix assignment information, dns delegation, routing information. As another example, a client might request a feature that the server can not provide. In this context, the client may decide to continue anyway without using that feature or the server could send a list of other servers who might offer the service to the client. The control protocol can optionaly be used to verify the sustainability of the underlying network: similar to the PPP control protocols who verify the link and close the connection when the link is down. It also enables the concept of the degenerated case where the broker and the server are together.

This framework protocol enables any kind of current and future tunnel techniques to be defined by a profile of this protocol.

2. Terminology

Tunnel Broker (TB) In a Tunnel Broker model, the broker is taking charge of all communication between Tunnel Servers (TS) and Tunnel Clients (TC). Tunnel clients query brokers for a tunnel and the broker find a suitable tunnel server, ask the Tunnel server to setup the tunnel and send the tunnel information to the Tunnel Client.

Tunnel Server (TS) Tunnel Servers are providing the specific tunnel service to a Tunnel Client. It can reveive the tunnel request from a Tunnel Broker (as in the Tunnel Broker model) or directly from the Tunnel Client as in the Tunnel Setup Protocol option. The Tunnel Server is the tunnel end-point.

Tunnel Client (TC) The Tunnel Client is the entity that need a tunnel for a particular service or connectivity. A Tunnel Client can be either a host or a router. The tunnel client is the other tunnel end-point.

3. Protocol Description

3.1 Topology

The following diagrams describe typical TSP scenarios. The goal is to establish a tunnel between Tunnel client and Tunnel server.

Tunnel Setup Protocol used on Tunnel Broker model

```
                              _______________
                             | TUNNEL BROKER |--> Databases (dns, whois)
                             |               |
                             |  TSP    TSP   |
                             | SERVER CLIENT |
                             |_______________|
                                 |     |
            __________           |     |          ________
           |           |         |     |         |  TSP   |
           |   TSP     |--[TSP]--      +--[TSP]--| SERVER |
           |  CLIENT   |                         |        |--[NETWORK]--
   [HOST]--|           |<==[CONFIGURED TUNNEL]==>| TUNNEL |
           |___________|                         | SERVER |
                                                 |________|
```

Tunnel Setup Protocol used on Tunnel Server model

```
            ___________                           ________
           |           |                         |  TSP   |
           |   TSP     |-----------[TSP]---------| SERVER |
           |  CLIENT   |                         |        |--[NETWORK]--
   [HOST]--|           |<==[CONFIGURED TUNNEL]==>| TUNNEL |
           |___________|                         | SERVER |
                                                 |________|
```

3.2 Overview

The Tunnel Setup Protocol has three phases:

Authentication phase: The Authentication phase is when the Tunnel Brokers/Servers advertises their capability to Tunnel Clients and when Tunnel clients authenticate to the server.

Command phase: The command phase is where the client requests or updates a tunnel.

Response phase: The response phase is where the respond to the client

For each command sent by a Tunnel Client their is an expected response by the server.

3.3 Authentication phase

The authentication phase has 3 steps :

- o Client's protocol version identification
- o Server's capability advertisement
- o Client authentication

When a TCP session is established to a Tunnel Server, the Tunnel Client sends the current protocol version it is supporting. The version number syntax is:

```
VERSION=1.0 CR LF
```

Version 1.0 is the version number of this specification.

If the server doesn't support the protocol version it sends an error message and closes the session. The server can optionally send a server list that may support the protocol version of the client.

Example of a Version not supported (without a server list)

```
-- Successful TCP Connection --
C:VERSION=0.1 CR LF
S:302 Unsupported client version CR LF
-- Connection closed --
```

Example of a Version not supported (with a server list)

```
-- Successful TCP Connection --
C:VERSION=1.1 CR LF
S:1302 Unsupported client version CR LF
  <tunnel action="list" type="broker">
     <broker>
        <address type="ipv4">1.2.3.4</address>
  </broker>
  <broker>
     <address type="dn">ts1.isp1.com</address>
  </broker>
  </tunnel>
-- Connection closed --
```

If the server supports the version sent by the client, then the server sends a list of the capabilities supported for authentication and tunnels.

```
CAPABILITY TUNNEL=V6V4 AUTH=DIGEST-MD5 AUTH=ANONYMOUS CR LF
```

Tunnel types must be registered with IANA and their profiles are defined in other documents. Authentication is done using SASL (RFC2222) [3]. Each authentication mechanism must be a registered SASL mechanism. Description of such mechanism is not in the scope of this document.

The Tunnel Client can then choose to close the session if none of the capabilities fits its needs. If the Tunnel Client chooses to continue, it must authenticate itself to the server using one of the adevertised mechanism. If the authentication fails the server sends an error message and closes the session.

Example of failed authentication

```
-- Successful TCP Connection --
C:VERSION=0.1 CR LF
S:CAPABILITY TUNNEL=V6V4 AUTH=DIGEST-MD5 CR LF
C:AUTHENTICATE ANONYMOUS CR LF
S:300 Authentication failed CR LF
```

If the authentication succeed, the server sends a success return code and the protocol enter the Command phase.

Successful authentication

```
-- Successful TCP Connection --
C:VERSION=0.1 CR LF
S:CAPABILITY TUNNEL=V6V4 AUTH=DIGEST-MD5 AUTH=ANONYMOUS CR LF
C:AUTHENTICATE ANONYMOUS CR LF
S:200 Authentication successful CR LF
```

Upon successful authentication the protocol enters the command phase as described in the next section.

3.4 Command phase

The Command phase is where the Tunnel Client send a tunnel request or a tunnel update to the server. In this phase, commands are sent as XML messages. The first line is a "Content-length" directive that tells the size of the following XML message. This makes it easier for protocol implementation to tell when they got the whole XML message. When the server sends a response, the first line is the "Content-length" directive, the second is the return code and third one is the XML message if any. The size of the response for the "Content-length" directive is the first character of the return code line to the last character of the XML message.

Spaces can be inserted freely.

Example of a command/response sequence

```
-- Successful TCP Connection --
C:VERSION=0.1 CR LF
S:CAPABILITY TUNNEL=V6V4 AUTH=DIGEST-MD5 AUTH=ANONYMOUS CR LF
C:AUTHENTICATE ANONYMOUS CR LF
S:200 Authentication successful CR LF
C: Content-length: 123 CR LF
  <tunnel action="create" type="v6v4">
    <client>
      <address type="ipv4">1.1.1.1</address>
    </client>
  </tunnel> CR LF

S: Content-length: 234 CR LF
   200 OK CR LF
   <tunnel action="info" type="v6v4" lifetime="1440">
    <server>
     <address type="ipv4">206.123.31.114</address>
     <address type="ipv6">3ffe:b00:c18:ffff:0000:0000:0000:0000</
address>
    </server>
    <client>
     <address type="ipv4">1.1.1.1</address>
     <address type="ipv6">3ffe:b00:c18:ffff::0000:0000:0000:0001</
address>
     <address type="dn">userid.domain</address>
    </client>
   </tunnel> CR LF
-- TCP Connection closed --
```

4. Error codes

Error codes are sent as a numeric value followed by a text message describing the code. The Tunnel Setup Protocol defines error code numbers 1 through 499 and 1000 through 1499. Profile dependant error codes are defined within the 500 through 999 and 1500 through 1999 range.

The predifined values are :

200 Success

Successful operation

300 Authentication failed

Invalid userid, password or authentication mechanism.

301 No more tunnels available

The server as reach its capacity limit.

302 Unsupported client version

The client version is not supported by the server.

303 Unsupported tunnel type

The server does not provide the requested tunnel type.

if a list of tunnel servers is following the error code as a referal service, then 1000 is added to the error code.

5. IANA Considerations

Tunnel types should be assigned by IANA based on a published RFC document.

A port number must be assigned for that protocol.

6. Security considerations

This protocol does not have encryption. When authenticating clients, SASL provides the necessary mechanism for negociating the authentication mechanism. As stated in SASL, the PLAIN authentication must not be used. The suggested method is DIGEST-MD5 (RFC2831) [4].

Tunnels generate routing entries that may be abused [6], while this is not specific to this TSP protocol

7. Acknowledgements

Alain Durand is the author of the seminal idea of tunnel brokers. This work is a follow-up based on many years of operating the freenet6.net tunnel broker where we saw additional needs for a control protocol to establish the tunnels.

Jun-Ichiro Itojun Hagino was, as usual, a great helper in refining and commenting this work.

This work has been done on a team basis so all people here contributed to the original work: Andre Cormier, Regis Desmeules, Florent Parent, Jocelyn Picard, Guy Turcotte.

References

[1] Durand, A., Fasano, P., Guardini, I. and D. Lento, "IPv6 Tunnel Broker", RFC 3053, January 2001.

[2] Narten, T., Nordmark, E. and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[3] Myers, J., "Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

[4] Leach, P. and C. Newman, "Using Digest Authentication as a SASL Mechanism", RFC 2831, May 2000.

[5] Durand, A., "IPv6 over IPv4 tunnels for home to Internet access method", July 2000.

[6] Hagino, J., "Possible abuse against IPv6 transition technologies", July 2000.

[7] "MIME-type extension for IPv6 configured tunnels".

Author's Address

Marc Blanchet
Viagenie
2875 boul. Laurier, bureau 300
Sainte-Foy, QC G1V 2M2
Canada

Phone: +1 418 656 9254
EMail: Marc.Blanchet@viagenie.qc.ca
URI: http://www.viagenie.qc.ca/

Appendix A. DTD

A DTD should be placed here for the protocol.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.