

PCP
Internet-Draft
Intended status: Standards Track
Expires: August 9, 2014

D. Wing
T. Reddy
P. Patil
R. Penno
Cisco
February 5, 2014

PCP Extension for Third Party Authorization
draft-wing-pcp-third-party-authz-02

Abstract

It is often desirable for an application server to permit a flow across a firewall, as happens today when a firewall includes an Application Layer Gateway (ALG) function. However, an ALG has several weaknesses.

This document describes a cryptographic technique for an application server to permit a flow across a firewall. This technique uses OAuth and a new PCP option.

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 August 9, 2014.

Copyright Notice

Copyright (c) 2014 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	2
2. Notational Conventions	3
3. Problem Statement	3
4. Solution Overview	5
5. Obtaining a Token Using OAuth	6
5.1. ACCESS_TOKEN Option	7
5.2. Generating the ACCESS_TOKEN option	9
5.3. PCP server processing ACCESS_TOKEN option	10
5.4. Processing the PCP response	11
6. PCP Server and Proxy behavior	11
7. Usage with PCP Authentication mechanism	12
8. Security Considerations	12
9. IANA Considerations	13
10. Acknowledgements	13
11. References	13
11.1. Normative References	13
11.2. Informative References	14
Authors' Addresses	15

1. Introduction

It is desirable for a third party to permit flows across a firewall. A typical use-case is a SIP proxy (which is aware of legitimate calls) which is not co-located with a firewall. Today, this functionality is provided by a firewall implementing a SIP-aware Application Layer Gateway function, which examines the SIP signaling to that SIP proxy and opens the appropriate pinholes for the RTP media. This has disadvantages, as described in detail in section Section 3.

This document addresses requirement "Third Party Authorization" explained in section 4 of [I-D.reddy-pcp-auth-req].

This document proposes that a PCP [RFC6887] client communicate with an OAuth Authorization Server to obtain a cryptographic token for its media flow. That token is included in the PCP request and validated by the PCP server.

Note: There is no relationship with the THIRD_PARTY option defined in [RFC6887], which serves a different purpose. THIRD_PARTY Option for

MAP and PEER Opcodes described in [RFC6887] is only applicable when all entities i.e the PCP client, PCP server and Application Server, are deployed within the same administrative domain. Since PCP server does not listen on a public interface, an Application Server outside the site will not be able to use THIRD_PARTY option to request services on behalf of the client.

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].

WebRTC Server: A web server that supports WebRTC [I-D.ietf-rtcweb-overview].

3. Problem Statement

To protect networks using real-time communications, firewalls or session border controllers [RFC5853] are typically deployed. Firewalls usually implement Application Layer Gateway functionality, which intercepts and analyzes session signaling traffic such as Session Initiation Protocol (SIP) [RFC3261] messages and creates a dynamic mapping to permit the corresponding media traffic. In particular, a firewall extracts media transport addresses, transport protocol and ports from session description and creates a dynamic mapping for media to flow through. This model will not work in the following cases:

1. Session signaling is end-to-end encrypted (say, using TLS).
2. Firewall does not understand the session signaling protocol, or extensions to the protocol, used by the endpoints.
3. Session signaling and media traverse different firewalls (e.g., signaling exits a network via one firewall whereas media exits a network via a different firewall)

When an enterprise deploys WebRTC, the above problems are relevant because:

1. Session signaling between WebRTC application running in a browser and a web server will use TLS.
2. WebRTC does not enforce a particular session signaling protocol; therefore, a firewall is unlikely to understand the signaling protocol.

3. Session signaling and peer-to-peer media may traverse different firewalls.

As a result firewalls block media traffic.

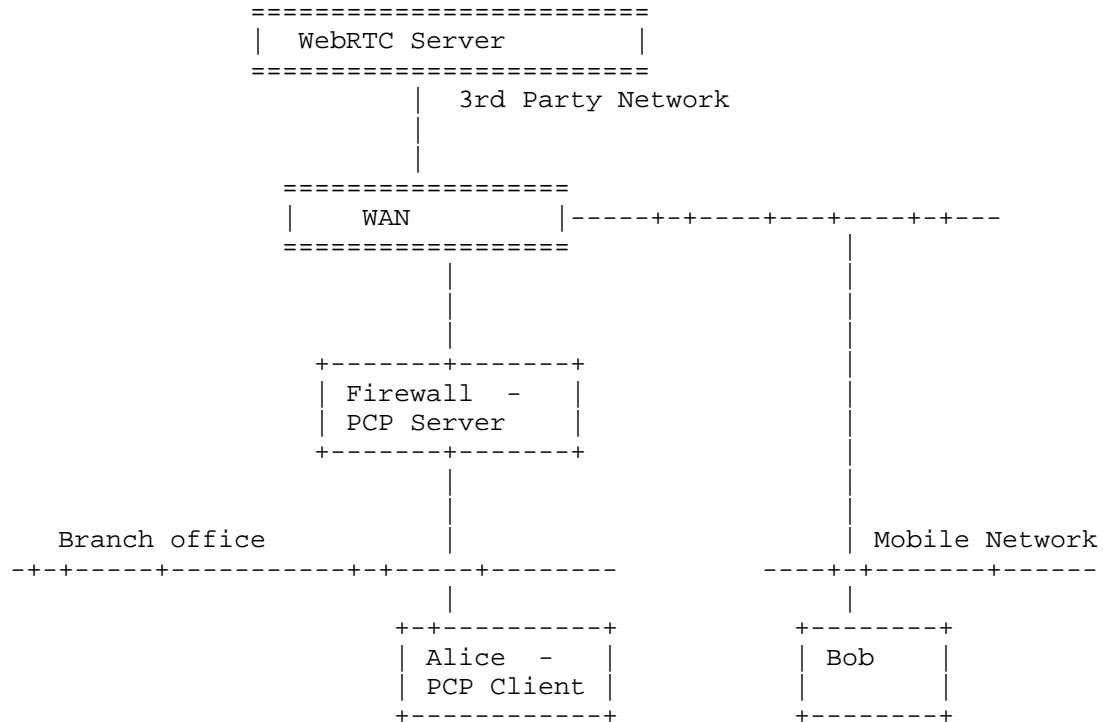
A mitigation to the problems above is for an enterprise to deploy a TURN server in the DMZ and have WebRTC clients use the TURN server. The use-case explained in Section 4.2.5.1 of [I-D.ietf-rtcweb-use-cases-and-requirements] refers to deploying a TURN [RFC5766] server to audit all media sessions from inside the company premises to any external peer.

However, using TURN for all such communication causes some problems for an enterprise network administrator :

- o Enterprise firewalls would typically have granular policies to permit calls initiated using selected WebRTC servers (Dr. Good) it trusts and block the rest (Dr. Evil).
- o A TURN server just provides a 5-tuple (source IP address, destination IP address, protocol number, source port number, and destination port number) for auditing and no other details of the WebRTC or SIP server being used to establish the call.
- o A TURN server could increase media latency as explained in section 4.1.2.2 of [RFC5245].
- o A TURN server could either be located in the DMZ of the enterprise network or located in the public Internet. If the TURN server is located in the public Internet it comes at a high cost to the provider of the TURN server, since the server typically needs a high-bandwidth connection to the Internet as explained in the Introduction of [RFC5766]. As a consequence, it is best to use a TURN server only when a direct communication path cannot be found. When the client and a peer use ICE to determine communication path, ICE will use hole punching techniques to search for a direct path first and only use a TURN server when a direct path cannot be found.
- o Other limitations of TURN are explained in section 2.6 of [RFC5766]. For example the value of Diffserv field may not be preserved, Explicit Congestion Notification (ECN) field may be reset etc.

4. Solution Overview

In the below topology, the main functional elements involved are :



Users : Alice, Bob

WebRTC Server : OAuth 2.0 Authorization server

Figure 1: WebRTC server in a different administrative domain

In the topology, a WebRTC Server is deployed in a third party network trusted by the Enterprise. For the two endpoints to successfully establish media sessions, a firewall needs to permit ICE [RFC5245] connectivity checks and subsequent media traffic.

In such a scenario this specification proposes that a PCP client follows the steps described below:

1. The PCP client makes a PCP request without any authorization. If the PCP server returns an `AUTHORIZATION_REQUIRED` error message,

the PCP client concludes that the PCP server is mandating the use of third party authorization.

2. The PCP client then obtains a cryptographic token from an OAuth 2.0 Authorization server.
3. The PCP client sends a PCP request including the cryptographic token in the TOKEN_ACCESS option, defined below. Alternatively, the PCP client could first obtain a cryptographic token from the OAuth 2.0 Authorization server and send the PCP request with the TOKEN_ACCESS option by default.
4. The PCP server uses the TOKEN_ACCESS option to perform third party authorization.

The technique proposed in the specification can be used by any other Application Function trusted by the network to permit time-bound, encrypted, peer-to-peer traffic.

5. Obtaining a Token Using OAuth

This section explains OAuth 2.0 authorization framework [RFC6749] to solve the "Third Party Authorization" requirement explained in section 4 of [I-D.reddy-pcp-auth-req].

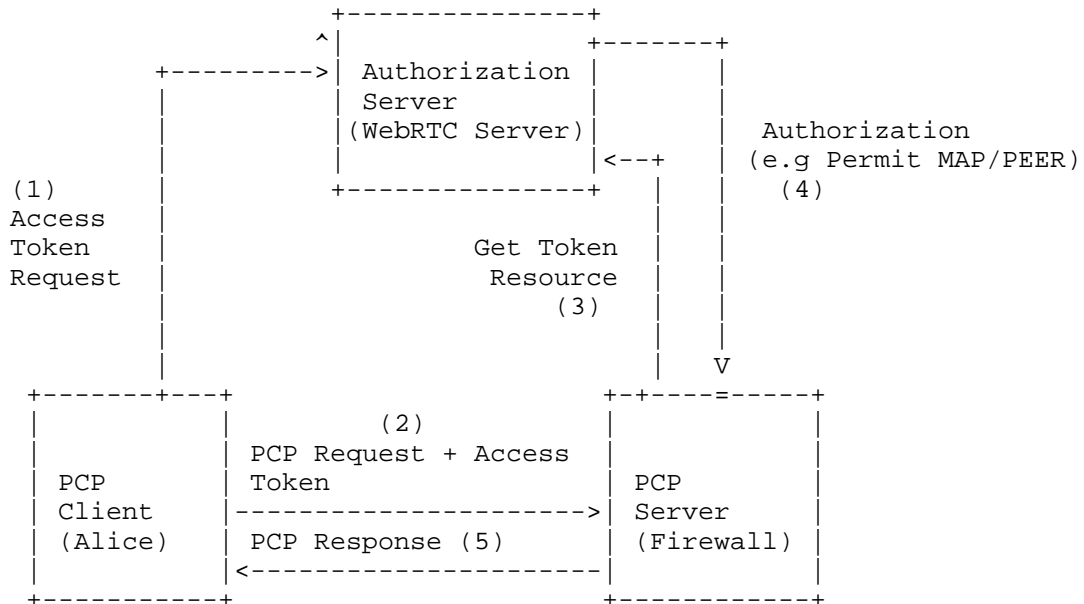
The following mapping of OAuth concepts to PCP is used :

OAuth	PCP
Client	PCP Client
Resource owner	Authorization Server. For example the WebRTC server
Authorization server	Authorization server.
Resource server	PCP Server

Figure 2: OAuth terminology mapped to PCP terminology

Using the OAuth 2.0 authorization framework, a PCP client (third-party application) obtains limited access to a PCP server (resource server) on behalf of the WebRTC server (resource owner or authorization server). The PCP client requests access to resources controlled by the resource owner (WebRTC server) and hosted by the resource server (PCP server). The PCP client obtains an access

token, lifetime, and other access attributes like the PCP options and opcodes that the PCP client is permitted to use from the authorization server. The PCP client conveys the token in the PCP ACCESS_TOKEN option to access the protected resources hosted by the resource server (PCP server). The PCP server validates the token and takes appropriate action e.g., allows the PCP request to create mappings on the PCP server.



User : Alice

Figure 3: Interactions

OAuth in [RFC6749] defines four grant types. This specification uses the OAuth grant type "Implicit" explained in section 1.3.2 of [RFC6749] where the PCP client is issued an access token directly. The scope of the access token explained in section 3.3 of [RFC6749] MUST be PCP.

5.1. ACCESS_TOKEN Option

This specification defines a new PCP ACCESS_TOKEN Option that is described in Figure 4.

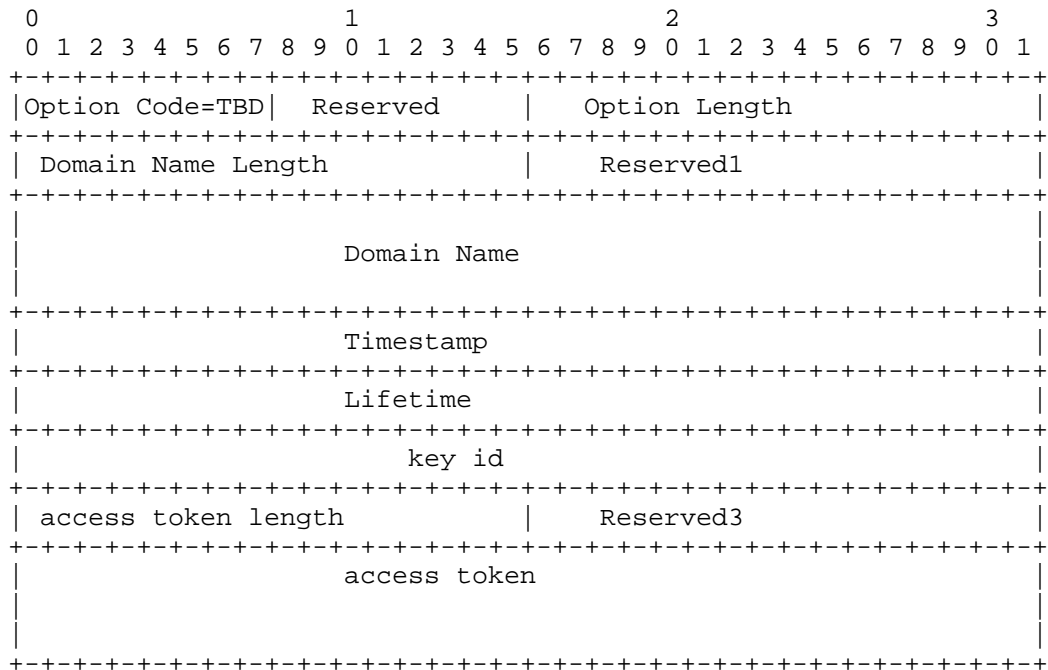


Figure 4: PCP ACCESS_TOKEN Option

The fields are described below:

Option Length: 16 bits. Indicates the length of the enclosed data, in octets. Variable, but MUST NOT be 0.

Domain Name Length: Length of the 'Domain Name' field in octets.

Reserved1: set to 0 by sender and ignored by the receiver.

Server Domain Name: The domain name of the Authorized Server that generated the access token.

Timestamp: 64-bit unsigned integer field containing a timestamp. The value indicates the time since January 1, 1970, 00:00 UTC, by using a fixed point format. In this format, the integer number of seconds is contained in the first 48 bits of the field, and the remaining 16 bits indicate the number of 1/64K fractions of a second (Native format - Unix).

Lifetime: The lifetime of the access token since the response was generated, in seconds. For example, the value 3600 indicates one

hour. The Lifetime value SHOULD be equal to the "expires_in" parameter defined in section 4.2.2 of [RFC6749].

key id: key id, which is an identifier generated by the authorization server. It generates this key id by computing a hash over the access token using SHA-1 and truncating the hash to 96 bits (retaining the left most bits).

access token length: Length of the access token field in octets. OAuth does not impose any limitation on the length of the access token but since PCP messages cannot exceed 1100 octets (Section 7 of [RFC6887]), access token length needs to be restricted to fit within the maximum PCP message size. The access token is defined in section 1.4 of [RFC6749]. TBD : what is the recommended/maximum token length for PCP. We need a discussion of this maximum length and analysis of what that means

Reserved3: set to 0 by sender and ignored by the receiver.

access token: The access token issued by the authorization server.

Option Name: ACCESS_TOKEN

Number: TBA in the mandatory-to-process range (IANA)

Purpose: This option conveys the token granted by the authorization server for third party authorization.

Valid for Opcodes: MAP, PEER

May appear in : request. May appear in response only if it appeared in the associated request.

Maximum occurrences : 1

5.2. Generating the ACCESS_TOKEN option

The mechanism used by an OAuth client to obtain a token from the OAuth authorization server is outside the scope of this document. The OAuth client could obtain the token via in-band signaling or an exclusive out-of-band protocol. This specification uses the token type Handle described in [RFC6819]. A handle token is a reference to some internal data structure within the OAuth authorization server; the internal data structure contains the attributes of the token such as allowed PCP Opcode or PCP Option, etc. The PCP client, after receiving the access token from the OAuth authorization server, generates the ACCESS_TOKEN option which is included in the PCP request to the PCP server.

5.3. PCP server processing ACCESS_TOKEN option

A PCP server performs processing in the order described below.

When a PCP server receives a PCP request with an ACCESS_TOKEN option, it will verify that the access token is valid. To address replay attacks, the PCP server MUST perform the following check :

When a PCP request with an ACCESS_TOKEN Option is received, the received timestamp (TSnew in the Timestamp field) is checked and the cryptographic token is accepted if the timestamp is recent enough to the reception time of the PCP request, RDnew :

$$\text{Lifetime} + \text{Delta} > \text{abs}(\text{RDnew} - \text{TSnew})$$

The RECOMMENDED value for the allowed Delta is 5 seconds. If the timestamp is NOT within the boundaries then discard the PCP request with AUTHORIZATION-FAILED error response defined in [I-D.ietf-pcp-authentication].

After the validation described above, the PCP server communicates with the authorization server in order to validate the token and obtain token-bound data. The mechanism for communication is outside the scope of this document. The PCP server makes a request to the authorization server to validate the token but produces no other data with the request. If the token is successfully validated, the authorization server just returns the token bound authorization data in the response. The PCP server then matches this authorization data with what is requested in the PCP request sent by the PCP client. If the authorization sets match, the PCP server honors the PCP request made by the PCP client.

If the token is invalid or the request exceeds what is authorized by the token then the PCP server generates an AUTHORIZATION-FAILED error response. An example might be that an OAuth authorization server permits creating 5 mappings, and the PCP request made by the client is trying to create a 6th mapping.

Handle token type was selected for the following reasons :

1. The Authorization Server can inform the PCP server to revoke the access token after the call is terminated. This mechanism ensures that even if the PCP client does not close the dynamic mapping created, the PCP server based on the revocation notification from the Authorization Server can close the dynamic mapping.

2. A PCP-controlled Firewall with restrictive policies may also want to validate with the Authorization Server if the selected candidate pairs in the final offer/answer match the 5-tuple {dest addr, source addr, protocol, dest port, source port} sessions traversing the Firewall. This validation ensures that the PCP client is using the token only to send and receive the media streams finalized in the call to the remote peer. Thus the PCP server can make sure that the token cannot be used for anything else.
3. If PCP authentication [I-D.ietf-pcp-authentication] is used then the PCP server may also validate with the authorization server if the access token is issued and used by the same user or not.

Another approach, not discussed in this document, is a self-contained token where all the information necessary to authenticate the validity of the token is contained within the token itself. This approach has the benefit of avoiding a protocol between the PCP server and the OAuth authentication server for token validation, thus reducing latency. However, this approach has the drawback of needing a large PCP packet to accommodate the token and requiring the authorization server to generate its message integrity over exactly the PCP fields, in the same order, that will be sent by the PCP client. Because PCP messages are limited to 1100 octets, using the handle approach is more flexible and the trade-off for additional latency is reasonable. The other disadvantages of self-contained tokens, such as difficulties with revocation etc., are discussed in[RFC6819].

5.4. Processing the PCP response

Upon receiving a PCP response, the PCP client performs the normal processing described in Section 8.3 of [RFC6887]. If the PCP response was SUCCESS (0), the PCP server has determined that the token is valid. If the PCP response was AUTHORIZATION-FAILED, it indicates that the token could be invalid, expired or the PCP request exceeded what is authorized by the token.

6. PCP Server and Proxy behavior

The ACCESS_TOKEN option is mandatory-to-process (its most significant bit is clear). Thus, per existing behavior described in [RFC6887], a PCP server receiving this option MUST return the error MALFORMED_OPTION if the option contents are malformed, or UNSUPP_OPTION if the option is unrecognized, unimplemented, or disabled, or if the client is not authorized to use the option.

A PCP Proxy MUST follow the rules mentioned in section of 7 of [I-D.ietf-pcp-proxy] when the processing the ACCESS_TOKEN option.

7. Usage with PCP Authentication mechanism

The following steps MUST be followed when PCP third party authorization is used with PCP authentication mechanism.

- o PCP client MUST send the access token after successful EAP authentication. This provides integrity protection for ACCESS_TOKEN option.
- o If PCP Auth session lifetime expires before the authorization token expires and the PCP client, PCP server fail to trigger re-authentication then dynamic mappings created because of third party authorization MUST be deleted.

8. Security Considerations

Security considerations discussed in [RFC6887] and PCP authentication [I-D.ietf-pcp-authentication] are to be taken into account. If left unprotected the Authorization server could present a means for an attacker to poll a series of possible token values, fishing for a valid token. Therefore, the Authorization Server SHOULD issue special credentials to PCP server to access it and the communication between PCP server and Authorization server MUST be protected using TLS.

A PCP server will delete explicit dynamic mappings after the lifetime of the cryptographic token expires. The PCP client must obtain a new cryptographic token from the authorization server before the current token becomes invalid or expires. The PCP client must propagate the new cryptographic token to the PCP server to refresh lifetime of mappings before the current token becomes invalid or expires. The PCP server in addition to timestamp checking can also maintain a cache of used kid as an effective countermeasure against replay attacks.

Discussion: If the additional latency needs to be avoided and it is permissible to create a PCP mapping briefly for PCP clients, an implementation could create PCP mappings while the token is being validated. The PCP server could create a mapping immediately, send a PCP response and in parallel start verification of the token. If the verification request times out or returns a failure response, the PCP mapping can be destroyed and a PCP mapping update is sent to the PCP client. The PCP server while waiting for the validation response to arrive from Authorization server can either drop or buffer the traffic matching the mapping created.

9. IANA Considerations

We request IANA register the PCP option ACCESS_TOKEN and the result code AUTHORIZATION_REQUIRED in [pcp-registry].

10. Acknowledgements

Authors would like to thank Dave Thaler, Charles Eckel, Paul Jones, Dacheng Zhang, Anca Zamfir, Parthasarathi R and Suresh kumar for their comments and review.

11. References

11.1. Normative References

- [I-D.ietf-pcp-authentication]
Wasserman, M., Hartman, S., and D. Zhang, "Port Control Protocol (PCP) Authentication Mechanism", draft-ietf-pcp-authentication-02 (work in progress), October 2013.
- [I-D.ietf-pcp-proxy]
Perreault, S., Boucadair, M., Penno, R., Wing, D., and S. Cheshire, "Port Control Protocol (PCP) Proxy Function", draft-ietf-pcp-proxy-05 (work in progress), February 2014.
- [I-D.reddy-pcp-auth-req]
Reddy, T., Patil, P., Wing, D., and R. Penno, "PCP Authentication Requirements", draft-reddy-pcp-auth-req-04 (work in progress), July 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.
- [RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, October 2011.
- [RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.
- [RFC6887] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, April 2013.

[pcp-registry]

IANA, , "Port Control Protocol (PCP) Parameters", May 2013, <<http://www.iana.org/assignments/pcp-parameters/pcp-parameters.xml>>.

11.2. Informative References

[I-D.ietf-rtcweb-overview]

Alvestrand, H., "Overview: Real Time Protocols for Brower-based Applications", draft-ietf-rtcweb-overview-08 (work in progress), September 2013.

[I-D.ietf-rtcweb-use-cases-and-requirements]

Holmberg, C., Hakansson, S., and G. Eriksson, "Web Real-Time Communication Use-cases and Requirements", draft-ietf-rtcweb-use-cases-and-requirements-12 (work in progress), October 2013.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, April 2010.

[RFC5853] Hautakorpi, J., Camarillo, G., Penfield, R., Hawrylyshen, A., and M. Bhatia, "Requirements from Session Initiation Protocol (SIP) Session Border Control (SBC) Deployments", RFC 5853, April 2010.

[RFC6342] Koodli, R., "Mobile Networks Considerations for IPv6 Deployment", RFC 6342, August 2011.

[RFC6819] Lodderstedt, T., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", RFC 6819, January 2013.

Authors' Addresses

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tiredy@cisco.com

Prashanth Patil
Cisco Systems, Inc.
Bangalore
India

Email: praspati@cisco.com

Reinaldo Penno
Cisco Systems, Inc.
170 West Tasman Drive
San Jose 95134
USA

Email: repenno@cisco.com