SDP-RTCP Combine Solution With Security For Multi Port Session

Tariq Ahamad
College Of Computer Engineering & Sciences
Salman Bin Abdulaziz University, KSA

Abstract

Session Description Protocol (SDP) is used for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. When a session requires multiple ports, SDP assumes that these ports have consecutive numbers. However, when the session crosses a network address translation device that also uses port mapping, the ordering of ports can be destroyed by the translation. SDP provides a standard representation for such information, irrespective of how that information is transported. SDP is purely a format for session description -- it does not incorporate a transport protocol, and it is intended to use different transport protocols as appropriate, including the Session Announcement Protocol, Session Initiation Protocol, Real Time Streaming Protocol, electronic mail using the MIME extensions, and the Hypertext Transport Protocol. To handle this, we propose an extension attribute to SDP.

Keywords

Session Description Protocol, Real Time Control Protocol, Session Initiation Protocol.

Introduction

The Session Description Protocol (SDP) was originally conceived as a way to describe multicast sessions carried on the Mbone. The Session Announcement Protocol (SAP) was devised as a multicast mechanism to carry SDP messages. Although the SDP specification allows for unicast operation, it is not complete [1]. Unlike multicast, where there is a global view of the session that is used by all participants, unicast sessions involve two participants, and a complete view of the session requires information from both participants, and agreement on parameters between them.

As an example, a multicast session requires conveying a single multicast address for a particular media stream. However, for a unicast session, two addresses are needed - one for each participant. As another example, a multicast session requires an indication of which codecs will be used in the session. However, for unicast, the set of codecs needs to be determined by finding an overlap in the set supported by each participant[2].

The Session Initiation Protocol (SIP) is an application-layer control protocol for creating, modifying, and terminating sessions such as Internet multimedia conferences, Internet telephone calls, and multimedia distribution. The SIP messages used to create sessions carry session descriptions that allow participants to agree on a set of compatible media types[3]. These session descriptions are commonly formatted using SDP. When used with SIP, the offer/answer model provides a limited framework for negotiation using SDP[4].

The session invitation protocol is often used to establish multi-media sessions on the Internet[5]. There are often cases today in which one or both ends of the connection are hidden behind a network address translation device . In this case, the SDP text must document the IP addresses and UDP ports as they appear on the "public Internet" side of the NAT. In this memo, we will suppose that the host located behind a NAT has a way to obtain these numbers.

The SIP messages use the encoding defined in SDP to describe the IP addresses and TCP or UDP ports used by the various media. Audio and video are typically sent using RTP , which requires two UDP ports, one for the media and one for the control protocol (RTCP) [6]. SDP carries only one port number per media, and states that "other ports used by the media application (such as the RTCP port) should be derived algorithmically from the base media port." RTCP port numbers were necessarily derived from the base media port in older versions of RTP , but now that this restriction has been lifted, there is a need to specify RTCP ports explicitly in SDP[7]. Note, however, that implementations of RTP adhering to the earlier specification may not be able to make use of the SDP attributes specified in this document.

When the NAT device performs port mapping, there is no guarantee that the mappings of two separate ports reflects the sequencing and the parity of the original port numbers; in fact, when the NAT manages a pool of IP addresses, it is even possible that the RTP and the RTCP ports may be mapped to different addresses. In order to successfully establish connections despite the misordering of the port numbers and the possible parity switches caused by the NAT, we propose to use a specific SDP attribute to document the RTCP port and optionally the RTCP address. An SDP session description includes the following:

- Session name and purpose
- Time(s) the session is active
- The media comprising the session
- Information needed to receive those media (addresses, ports, formats, etc.)
- Information about the bandwidth to be used by the session
- Contact information for the person responsible for the session

In general, SDP must convey sufficient information to enable applications to join a session (with the possible exception of encryption keys) and to announce the resources to be used to any non-participants that may need to know. (This latter feature is primarily useful when SDP is used with a multicast session announcement protocol.)

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.

Proposed Solution

To make it more easy to understand and implement we declare an attribute of SDP for documenting the port used by RTCP.

RTCP Attribute

The RTCP attribute is used to document the RTCP port used for media stream, when that port is not the next higher (odd) port number following the RTP port described in the media line. The RTCP attribute is a "value" attribute, and follows the general syntax: "a=<attribute>:<value>". For the RTCP attribute:

- The name is the ascii string "rtcp" (lower case),
- The value is the RTCP port number and optional address.

rtcp-attribute = "a=rtcp:" port [nettype space addrtype space connection-address] CRLF

Example encodings could be:

```
m=audio 49170 RTP/AVP 0
a=rtcp:53020

m=audio 49170 RTP/AVP 0
a=rtcp:53020 IN IP4 126.16.64.4

m=audio 49170 RTP/AVP 0
a=rtcp:53020 IN IP6 2001:2345:6789:ABCD:EF01:2345:6789:ABCD
```

The RTCP attribute MAY be used as a media level attribute; it MUST NOT be used as a session level attribute. Though the examples below relate to a method that will return only unicast addresses, both unicast and multicast values are valid.

How do we Discover Port Numbers?

The proposed solution is only useful if the host can discover the "translated port numbers", i.e., the value of the ports as they appear on the "external side" of the NAT. One possibility is to ask the cooperation of a well connected third party that will act as a server according to STUN .We thus obtain a four step process:

1. The host allocates two UDP ports numbers for an RTP/RTCP pair,
2. The host sends a UDP message from each port to the STUN server,
3. The STUN server reads the source address and port of the packet, and copies them in the text of a reply,
4. The host parses the reply according to the STUN protocol and learns the

external address and port corresponding to each of the two UDP ports.

This algorithm supposes that the NAT will use the same translation for packets sent to the third party and to the "SDP peer" with which the host wants to establish a connection. There is no guarantee that all NAT boxes deployed on the Internet have this characteristic. Implementers are referred to the STUN specification for an extensive discussion of the various types of NAT.

Why not Expand the Media Definition?

The RTP ports are documented in the media description line, and it would seem convenient to document the RTCP port at the same place, rather than create an RTCP attribute[8]. We considered this design alternative and rejected it for two reasons: adding an extra port number and an option address in the media description would be awkward, and more importantly it would create problems with existing applications, which would have to reject the entire media description if they did not understand the extension. On the contrary, adding an attribute has a well defined failure mode: implementations that don't understand the "a=rtcp" attribute will simply ignore it; they will fail to send RTCP packets to the specified address, but they will at least be able to receive the media in the RTP packets.

Considerations

The RTCP attribute in SDP is used to enable establishment of RTP/RTCP flows through NAT. This mechanism can be used in conjunction with an address discovery mechanism such as STUN . STUN is a short term fix to the NAT traversal problem, which requires thus consideration of the general issues linked to "Unilateral self-address fixing"[9].

The RTCP attribute addresses a very specific problem, the documentation of port numbers as they appear after address translation by a port-mapping NAT. The RTCP attribute SHOULD NOT be used for other applications[10].

We expect that, with time, one of two exit strategies can be developed. The IETF may develop an explicit "middlebox control" protocol that will enable applications to obtain a pair of port numbers appropriate for RTP and RTCP. Another possibility is the deployment of IPv6, which will enable use of "end to end" addressing and guarantee that the two hosts will be able to use appropriate ports. In both cases, there will be no need for documenting a "non standard" RTCP port with the RTCP attribute.

Security

This SDP extension is not believed to introduce any significant security risk to multi-media applications. One could conceive that a malevolent third party would use the extension to redirect the RTCP fraction of an RTP exchange, but this requires intercepting and rewriting the signalling packet carrying the SDP text; if an interceptor can do that, many more attacks are available, including a wholesale change of the addresses and port numbers at which the media will be sent.

In order to avoid attacks of this sort, when SDP is used in a signalling packet where it is of the form application/sdp, end-to-end integrity using S/MIME is the technical method to be implemented and applied. This is compatible with SIP.

Conclusion

In this research article we discussed how SDP and RTCP can be involved to improve security and reliability of the network and the data that is to be transmitted over it. As session requires multiple ports, SDP assumes that these ports have consecutive numbers. However, when the session crosses a network address translation device that also uses port mapping, the ordering of ports can be destroyed by the translation an in this article we have shown and proved that after we apply our solution for this we can overcome this issue and can improve security and reliability. Even after inclusion of SIP it can be made more secure.

References

1. Westerlund, M., "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", RFC 3890, September 2004.
2. Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, July 2003.
3. Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "Key Management Extensions for Session Description Protocol (SDP) and Real Time Streaming Protocol (RTSP)", RFC 4567, July 2006.
4. Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.
5. 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.
6. Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.
7. Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
8. Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, October 2003.
9. Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC3711, March 2004.
10. Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.