Reflections on Security Options for the Real-time Transport Protocol Framework

Colin Perkins
University of Glasgow, UK
Email: csp@csperkins.org

***Abstract*—The Real-time Transport Protocol (RTP) supports a range of video conferencing, telephony, and streaming video applications, but offers few native security features. We discuss the problem of securing RTP, considering the range of applications. We outline why this makes RTP a difficult protocol to secure, and describe the approach we have recently proposed in the IETF to provide security for RTP applications. This approach treats RTP as a framework with a set of extensible security building blocks, and prescribes mandatory-to-implement security at the level of different application classes, rather than at the level of the media transport protocol.**

I. Introduction

Real-time multimedia comprises a large, and rapidly growing, fraction of the traffic on the Internet. This traffic can be broadly divided into two categories: streaming and interactive. Streaming multimedia is typically delivered using some variant of dynamic adaptive streaming over HTTP [1], building on the web content distribution infrastructure for scalability. Interactive multimedia, including mobile telephony, voice over IP, video conferencing, telepresence, and the emerging WebRTC standards [2] overwhelmingly uses the Real-time Transport Protocol (RTP) [3] for media transport. RTP has proven to be an effective basis for such applications, supporting a wide range of use cases, media codecs, error correction and concealment mechanisms, and performance monitoring tools. However, the base RTP specification offers only minimal security features.

The limited provisions for security in the base RTP standards were not due to lack of interest or awareness by the Internet Engineering Task Force (IETF) or the research community. Indeed, the Mbone multicast video conferencing tools [4], [5], [6], [7], some of the earliest research prototypes of RTP, incorporated strong (for the time) media path encryption, and experimented with various key distribution schemes. Rather, the limited security options in the baseline protocol are due to conflicting security requirements, making it unclear what features should be provided as standard.

In this paper, we show that no single media path security or key exchange mechanism is sufficient to secure the RTP protocol. Rather, we recognise that a framework approach must be adopted, using security building blocks that can be composed to suit the requirements of particular classes of application, rather than trying to secure the protocol as a whole. Our contribution has been to identify that such a framework is needed, and to achieve consensus on the approach in IETF, ensuring that future application classes will be required to describe a clear security architecture around how they use RTP. We review and justify this approach, and discuss how RTP security extensions are developing to support emerging applications, such as WebRTC. We conclude with a discussion of the applicability of these ideas to new media transport protocols.

This paper builds on two recent proposals we made to the IETF [8], [9]. We review the arguments for treating RTP as a framework with pluggable security building blocks suited to different use cases, and extend them with additional motivation, history, and discussion of the relevant issues. The RTP protocol is novel in that it supports a much broader range of transport topologies and reliability modes than other transport-layer protocols, such as TCP, DCCP [10], and SCTP [11]. This leads to wide-ranging, and potentially conflicting, security requirements. The issues explored here in the context of RTP are therefore potentially applicable to future transport protocols that offer flexibility to support many different application scenarios, topologies, and classes of reliability. With the resurgence in interest in new transport protocols, such as TCP Minion [12], QUIC, etc., and the recent Transport Services (TAPS) BoF in IETF, these issues potentially have broad applicability.

We structure the remainder of this paper as follows. We begin by reviewing security requirements for Internet standards in Section II, and the security features offered by RTP in Section III. Use case and applications of RTP are outlined in Section IV, and Section V describes the implications of these for securing the RTP framework. Section VI discusses mandatory to implement security mechanisms for RTP. Finally, Section VII concludes.

II. Security for Internet Standards

The Internet standards community has long considered the need for strong security in network protocols. The so-called "Danvers Doctrine", named after the location of the meeting where it was formulated in 1995, states that IETF should standardize the best available security for network protocols, regardless of national policies.

This policy led the Internet Architecture Board (IAB) and the Internet Engineering Steering Group (IESG) to issue a Statement on Cryptographic Technology and the Internet [13] in 1996, noting the need for strong encryption to protect the privacy of Internet users and to secure commercial transactions on the network, and the adoption of a formal policy on Strong Security Requirements for IETF Standard Protocols [14] in 2002. This policy states that the IETF "MUST implement strong security in all protocols to provide for the all too frequent day

978-1-4799-6204-4/14$31.00 ©2014 IEEE

when the protocol comes into widespread use in the global Internet". It is, however, important to note that the requirement is that strong security must be implemented, not that it must be used. The goal is that users will have the option of using those protocols in a secure manner. The IETF cannot, and does not try to, enforce the use of strong security.

More recently, the IETF has considered the issue of pervasive network monitoring. It was concluded that "Pervasive monitoring is a technical attack that should be mitigated in the design of IETF protocols, where possible" [15]. That is, the IETF policy is that pervasive monitoring is to be treated as any other man-in-the-middle attack on the protocols, and should be prevented by the same means used to prevent other such attacks. Taken together, these policies point to a clear requirement that protocol standards, such as the RTP standards, need to provide strong, mandatory to implement, security.

III. Security Features of the RTP Standard

The previous section demonstrated the clear consensus in the IETF that all Internet standard protocols need to provide strong security. The first version of the RTP standard was published in 1996 [16], pre-dating publication of the IETF policy documents in this area, and the majority of protocol development work was completed before the adoption of the Danvers Doctrine. Despite this, the RTP protocol and its early implementations did provide confidentiality through encryption of RTP packets using the Data Encryption Standard (DES) in cipher block chaining (CBC) mode. Authentication and message integrity were not defined, "since these services would not be directly feasible without a key management infrastructure" [16].

The authors of [16] were aware of the limitations of the security service they proposed. Indeed, the protocol specification states that "other services, other implementations of services and other algorithms may be defined for RTP in the future if warranted. The selection presented here is meant to simplify implementation of interoperable, secure applications and provide guidance to implementers. No claim is made that the methods presented here are appropriate for a particular security need."

A revision to RTP specification was published in 2003 [3] that deprecates the confidentiality mechanism in the base specification. This revision suggests the Secure RTP (SRTP) profile (still under development at that time, but now published as [17]) as a likely replacement security mechanism. The SRTP profile has indeed seen wide adoption, and is appropriate and provides suitable strong security for many applications. However, it has become clear that the SRTP profile is not suitable for all applications using RTP. In Section IV we discuss use cases of RTP, highlighting where SRTP is not appropriate. Then, in Section V, we describe available security options.

IV. RTP Applications and Use Cases

RTP is used by a wide range of applications, supporting different topologies and use cases. To understand the security requirements, we must first review how the protocol is used.

A. RTP Topologies

RTP is inherently a group communication protocol, that can be used in a wide range of different topologies. These include point-to-point unicast communication, group communication using unicast to some middlebox such as a media mixer, transcoders, and other protocol translator, any source multicast, or source-specific multicast [18].

In the simplest case, RTP sessions are point-to-point between two endpoints. In these cases, RTP security is primarily concerned with providing confidentiality from third-parties, integrity protection, and authentication of the endpoint identity to prevent spoofing.

RTP is not solely designed for point-to-point communication, however, and also supports group communication as a fundamental part of the protocol. This leads to challenges with source identity, authentication, and access control that are inherently different to those with point-to-point sessions. In some cases, the group communication is provided by a multicast group, adding the challenge of unreliable UDP-based communication to the problems of group security. In other cases, group communication is supported by incorporating central middleboxes such as audio mixers, video switchers, or media transcoders into the RTP session. The extent to which such middleboxes are trusted to access the media varies, with many conferencing applications granting them full access, while some TV distribution applications do not trust the middleboxes in the content distribution network with access to the media data, and only grant access to the header metadata.

B. RTP Application Scenarios

RTP is also used for a range of different applications, and has been deployed in numerous different scenarios. Examples include, but are not limited to:

- Point-to-point telephony and voice-over-IP;
- Point-to-point video conferencing and telepresence;
- Centralised group video conferencing and telepresence, using a Multipoint Conference Unit or similar middlebox to perform audio/video mixing or video switching;
- Mbone-style video conferencing [4] using Any Source Multicast and the light-weight sessions model;
- Point-to-point streaming audio and/or video (e.g., on-demand streaming, often with RTSP [19] control);
- Cable TV replacement services offered by residential ISPs, or in the mobile space as the 3GPP Multimedia Broadcast Multicast Service, using Source-Specific Multicast (SSM) streaming to large groups of receivers;
- Replicated unicast (peer-to-peer) streaming to a group of receivers;
- Interconnecting components in music production studios and video editing suites;
- Interconnecting components of distributed simulation systems; and
- Streaming real-time sensor data, for example in some e-VLBI radio astronomy test-beds.

The key point is that these use cases vary in size from two participant point-to-point sessions to multicast groups with tens of thousands (or, perhaps, even millions) of participants, vary

between interactive to non-interactive, and from low bandwidth (kilobits per second) telephony to high bandwidth (multiple gigabits per second [20], [21]) video and data streaming.

Most applications of RTP run over unreliable UDP streams, but some build on TCP or DCCP [10] transport for NAT traversal, reliability, or congestion control reasons. Some use of RTP run solely on highly reliable optical networks, while others use low rate unreliable wireless networks. Some applications of RTP operate entirely within a single trust domain, others run inter-domain, with untrusted (and, in some cases, potentially unknown) users. The range of scenarios is wide, and increasing in both number and in heterogeneity.

V. Securing the RTP Protocol Framework

As discussed in Section II, the IETF requires all protocols to provide strong, mandatory-to-implement, security. This promotes the overall security of the network by ensuring that all implementations of a particular protocol are capable of interoperating in a secure manner. The wide range of topologies and use cases of RTP, described in Section IV, significantly complicate the security options for RTP, however. In the following we describe the requirements, then some of the media security and session establishment protocols used.

A. Requirements

Considering different application use cases and topologies, several different security requirements emerge. These include:

- **Confidentiality** of the media. Who has access to the media streams, and for how long are they granted access. This is particularly complex in group communication scenarios where group membership can change, since this will involve revoking the credentials of the leaving member, or updating the credentials of the other group members, so that the leaving member has no access once they have left the group. The use of centralised conferencing servers to distribute the media can simplify this problem, but there are still issues with scaling sessions that grow too large for a single server to support. Controlling membership of multicast sessions is a subset of the reliable multicast problem, and is subject to many challenges as a result.
- **Integrity protection** to detect corruption in the network and to prevent third-parties from unauthorised modification of content. Heavily compressed audio-visual content is sensitive to corruption because errors disrupt the codec state and propagate (if they don't stop coding entirely), so effective integrity protection is a must if the underlying network is not reliable. This becomes an issue in group communication scenarios using middleboxes for content distribution, since media manipulation by the middlebox is, in many ways, indistinguishable from media manipulation by an attacker; they key difference being that it is wanted modification, rather than unwanted. Applications need to consider whether hop-by-hop security, with some evidence that the desired middlebox is present in the session, is sufficient; or if true end-to-end integrity protection is needed. If the latter, to what extent can be middlebox perform media mixer or other manipulation?
- Integrity is closely tied with **source authentication**, that seeks to determine who sent a particular RTP packet, and **identity**. Again, use of group communication complicates the issues, leading to consideration whether it is necessary to authenticate individual participants, or just to authenticate them as a valid member of the group. If middleboxes are being used to support the group, the question of whether the middlebox or the participants are being authenticated applies, especially if the goal is to authenticate data as being from a valid group member.
- Finally, **privacy** is an issue. Applications that communicate directly between peers expose the IP address of the peer, but this may be sensitive information for some applications, because it can expose the geographic location of the user. This can be addressed by the use of relays, provided the relay is trusted, or by the use of anonymity services such as Tor. These expose issues with integrity, confidentiality, etc., as discussed earlier.

Much of the challenge in securing RTP comes because different applications that use the protocol have widely differing requirements for these security properties.

B. Media Path Security

The range of scenarios and use cases for RTP has led to multiple media security protocols being developed, each addressing different requirements.

The most widely applicable of these media security protocols is the secure RTP (SRTP) profile [17]. SRTP is an application-level security solution, that encrypts RTP payload data running over UDP to provide confidentiality, and provides source origin authentication as an option. SRTP was designed to be low overhead, and to support the group communication and third-party performance monitoring features of RTP. It leaves RTP protocol headers in the clear to simplify operation through middleboxes and to support header compression for use of low-speed links. SRTP provides strong support for cryptographic algorithm agility. The mandatory-to-implement transforms are the Advanced Encryption Standard (AES) in counter mode, with 160-bit keyed HMAC-SHA-1 and an 80-bit authentication tag. Alternative transforms can be negotiated, and the protocol currently also support AES in f8 mode, AES in Galois Counter Mode with CBC MAC, SEED, ARIA, and other transforms.

Datagram TLS [22] is an extension to Transport Layer Security (TLS) [23] that works over an unreliable datagram transport, such as UDP. It can be used with RTP, encapsulating RTP packets inside DTLS, to provide security analogous to TLS but for point-to-point datagram traffic. DTLS offers confidentiality, integrity protection, and can offer source origin authentication if the client or server certificates can be verified and provide a usable identity. DTLS operates in point-to-point mode only, and so requires the use of a middlebox for multiparty applications. In this case it offers security for each connection to the middlebox, but the middlebox itself has to be trusted with the media.

RTP is most commonly used over UDP, to give the application visibility into, and some control over, the delivery, error recovery, and congestion control processes. There is

nothing in the protocol that requires operation over UDP however, and some implementations run over TCP, accepting reliable delivery and TCP congestion control, often to simplify NAT traversal. When RTP is run over a point to point TCP connection, TLS can be used to protect that connection [24]. This offers confidentiality, integrity protection, and possible source origin authentication, much as was just described for RTP over DTLS, with the same limitations regarding trusted middleboxes and point-to-point communication.

As an alternative to the transport layer mechanisms, RTP flows can be secured at the network layer using IPSec [25] to protect the RTP packets in transit between network interfaces on the sender and receiver hosts. IPSec can be used in either transport or tunnel mode, depending on needs, but perhaps most common is tunnel mode as a virtual private network (VPN), protecting RTP and all other traffic across a particular network path. A concern with this approach is that the VPN often terminates at some intermediate node, rather than at the final destination, and so leaves traffic unprotected for some part of the path. This may be sufficient in many cases, however, depending on the security risks deemed important. For example, users within the network belonging to a particular organisation might be considered trusted, with the VPN being used to tunnel traffic from roaming users back to the home organisation. The increased use of pervasive monitoring increases the risk of this approach, but it may be acceptable nonetheless.

There are also various media content security and digital rights management solutions that provide security for media content transported in RTP. Some of these solutions encrypt only the media content, and operate within the RTP payload data, leaving payload and RTP headers unprotected. A common rationale for this is streaming media where the streaming server is not trusted, and has to work with pre-encrypted content (i.e., content that is encrypted by the media producer, and can be decrypted by the end user, but not by the intermediate content distribution infrastructure). One example of this approach is the Internet Streaming Media Alliance (ISMA) content Encryption and Authentication standard [26]. This describes how content stored as MPEG elementary streams within an ISO media file can be encrypted and conveyed within the RTP payload format for generic MPEG-4 streams, leaving the RTP and payload headers in the clear.

The different approaches highlighted, and others that undoubtedly exist, show the range of media security protocols. Some support group communication; some do not. Some are optimised for low-overhead, header compression, and low speed links; others are not. Some require reliable transport; others do not. Some allow intermediate nodes to access the media; some do not trust such nodes. There are a range of options.

C. Session Establishment

To complement the variety of media security protocols, there are a range of session establishment protocols. These are primarily designed to support negotiation of media codecs, transport protocols, network addresses, etc., to be used for the session, but can also be used to bootstrap security for the media security protocols that cannot do so themselves.

The secure RTP profile has no integrated key management protocol, and instead relies on external key management. There are four commonly used key management protocols: DTLS-SRTP, MIKEY, SDP security descriptions, and ZRTP:

- As noted in Section V-B, DTLS can be used directly to protect RTP session, sending RTP packets inside DTLS packets. The **DTLS-SRTP** protocol [27], [28] is an optimisation to this, that uses DTLS key exchange but sends SRTP packets without DTLS encapsulation, hence reducing the per-packet header overhead. This is important for low-rate applications, such as voice over IP. As with regular DTLS, it can support perfect forward secrecy, cipher suite agility, source authentication with the aid of an identity service, and so on. Since it builds on DTLS, this is a point-to-point session establishment protocol, and requires middleboxes to be trusted. DTLS-SRTP is the preferred key management protocol for WebRTC [29].
- **MIKEY** [30] is a keying protocol with a range of modes that is designed to support not just point-to-point applications, but also broadcast and multicast applications, and centralised group communication with a shared security context for the group. MIKEY can be embedded into existing session establishment protocols, such as SIP, or sent directly over UDP.
- **SDP security descriptions** [31] can be used to exchange keys in the context of an SDP offer/answer exchange for example as part of a SIP call. It assumes the signalling channel and any signalling servers are securely protected, and sends the media keys in plain text within that channel. As such, it has questionable security outside walled garden networks with trusted operators. Despite that, it is widely used.
- **ZRTP** [32] is an alternative to DTLS-SRTP that provides best-effort encryption with key continuity. It uses either a short authentication string exchanged over the media path (i.e., read out and compared by users) or public key infrastructure for authentication.

Content security and digital rights management protocols used with RTP, such as ISMACryp [26], also require key management protocols to pass media access keys to the end users. These are typically coupled with subscription management, billing, and identity services. We do not consider details of such protocols further here.

As with media security protocols, the key point from this discussion is not the details of the specific key exchange and session establishment protocols, but rather the range of protocols used, and their different properties.

VI. Mandatory-to-implement Security for RTP

To ensure that all users of the RTP protocol have access to strong security, it would be desirable if a single media security protocol and a single session establishment protocol could be developed to suit the various application scenarios and topologies where RTP is used. As the discussion in Section V has shown, however, this will be a challenge. Unifying security mechanisms across a particular topology (e.g., for all unicast RTP flows, or for all multicast RTP sessions) is difficult because different applications use those RTP topologies in different ways. Unifying across application scenarios may be

possible in closely related areas, but fails when considering the range of use cases.

We are not aware of any protocols that meet the needs of all applications, and think it unlikely that any will be developed, since the appropriate security mechanism for one scenario does not work for some other scenarios where RTP is used. It is difficult to meet the IETF requirement for strong mandatory-to-implement security [14] for all users of RTP. The requirements of the various use cases are simply too different for a common solution to work.

A. Mandatory-to-implement Security for Framework Protocols

For a framework protocol, such as RTP, it appears therefore that the only reasonable way to provide strong security is to develop security building blocks that can be combined to meet the needs of particular classes of application; then to mandate particular combinations of building blocks for particular classes of application. In essence, the requirement for strong mandatory-to-implement security is moved from the low-level protocol component, to a higher-level application class.

In the RTP context, existing media security building blocks are outlined in Section V-B and signalling building blocks in Section V-C. A security architecture document needs to be written for a particular class of application, specifying the media path security and keying protocols that are mandatory-to-implement for those applications. An example of such an application class is Web Real-time Communication (WebRTC) system specified by the W3C and IETF [33], where the IETF has mandated the combination of the SRTP profile for media path security with DTLS-SRTP as the key exchange protocol as the mandatory to implement security building blocks [29].

The case of WebRTC is illustrative of the requirement for different media security and keying requirements for different application scenarios. For example, the set of security building blocks using in WebRTC differs from that used in SIP-based voice telephony applications. The requirement in the telephony community for interworking between SIP-based applications and the legacy PSTN drives a strong need for middleboxes to be trusted, and able to access the media in certain cases so they can gateway it into non-RTP environments. Both WebRTC and the telephony community have chosen SRTP as the media path security, but the telephony community generally makes use of SDP security descriptions in place of DTLS-SRTP for key exchange, trusting the middleboxes.

Another illustrative example application scenario is RTP-based cable TV replacement services using IP multicast media distribution. These often source content from traditional TV broadcasters as MPEG Transport Streams, with media content security being provided at the payload level, leaving no need for RTP-layer media path security, and couple key distribution to use digital rights management meaning that they cannot use protocols such as DTLS-SRTP or SDP security descriptions. Their security architecture is therefore quite different to that for WebRTC or for telephony, although all use RTP.

When new classes of application arise for a framework protocol such as RTP, they need to be studied to determine if the existing protocols and building blocks can provide acceptable security, satisfying the requirements of the new application domain. If so, then a mandatory-to-implement specification for security of this application class needed to be written. If not, new security building blocks must be defined, then incorporated into such a mandatory-to-implement security specification. To maximise interoperability, and reduce specification effort, it is desirable to use common building blocks for applications with similar requirements, where possible.

B. Relation to RTP Extension Points

The RTP protocol framework can be extended in various ways, and a common misconception is that security can be enforced at the various extension points. Some types of extension can significantly change the behaviour of the protocol, so that applications using those extensions form a different application class, with different security requirements. However, other types of extension have little or no impact on the security requirements and properties of the protocol, and can be used independently by different classes of application.

Two key extension points for RTP are RTP Payload Formats and RTP Profiles. An RTP Payload Format describes how a particular media codec can be used with RTP. There are over 70 standard RTP payload formats defined, along with numerous proprietary formats. In addition to describing how codec data is to be encoded in RTP packets, an RTP payload format specification should describe the security implications of using that codec with RTP (e.g., if the codec has non-uniform computational complexity when decoding, and can potentially be forced to cause a denial of service due to excess resource consumption). RTP payload formats do not, however, specify interoperable classes of application using RTP since media codecs and their associated RTP Payload Formats can be generally used with many different classes of application. As such, an RTP Payload Format is neither secure in itself, nor something to which the requirement for strong security defined in [14] applies.

An RTP Profile is a larger extension to the RTP protocol that adapts the RTP framework in some more extensive manner. Depending on the features provided, an RTP profile might form the basis for an interoperable class of applications that share common security requirements. An example is the secure RTP profile itself, which provides one of the main security options for RTP suitable for applications that require low overhead. Not all RTP profiles denote particular classes of application that share security requirements, however. For example, the RTP/AVPF profile [34] provides more extensive reception quality feedback and codec control options, but is used by a range of applications from source-specific multicast TV distribution to point-to-point telepresence, with distinct security requirements. Each RTP profile needs to indicate whether or not it makes sense to mandate particular security options for all implementation of the profile. However, there is no expectation that all RTP profiles will mandate particular security options.

For both extension points, RTP payload formats and RTP profiles, RTP acts as a framework, allowing flexible extensions that are suitable for a wide range of applications. Where particular security requirements and mandates are defined is at a higher level than RTP: at the level of the complete system design. That is, one would design a security architecture for a particular class of applications, and specify a set of

RTP extensions plus a signalling and session establishment protocol as a whole, defining security properties for that whole. An example is the WebRTC security architecture [29], that provides strong security features for the interactive conferencing implementations in web browsers.

VII. Discussion and Conclusions

We have shown that providing effective strong security for RTP requires the adoption of a framework approach, using security building blocks that can be composed to meet the needs of different classes of application. We have taken these ideas to IETF [8], [9] and achieved consensus in that community that the Danvers doctrine and the requirement that Internet standard protocols include strong security [14] can be addressed at the level of particular application classes for RTP. The WebRTC system is the first to adopt a security architecture [29] based on these ideas.

The issues we identify with securing RTP are illustrative of a broader problem that will emerge as transport protocols evolve more flexibility. When a transport protocol has limited scope, with well defined behaviour across the range of usage scenarios, then the security requirements are constrained, and can naturally be implemented at the transport layer. For example, TLS [23] provides a widely adopted security framework for TCP, that relies on reliable ordered delivery. Extensions to other environments are possible, witness DTLS [22], but not trivial. RTP as a transport covers a broader range of scenarios, topologies, and use cases than many protocols, and as a result needed a much broader range of security building blocks. Efforts are under way in IETF (e.g., the TAPS BoFs at the 89th and 90th IETF meetings) to develop flexible and pluggable transport protocols, that broaden the reach of transports accessible to non-real-time applications. These protocols such must aware of the issues with securing RTP, incorporate the appropriate security building blocks into their design, and be prepared to consider non-traditional policies for where in the protocol stack they enforce mandatory-to-implement security requirements.

VIII. Acknowledgements

This work has benefited from many discussions with Magnus Westerlund, the other members of the IETF Audio/Video Transport working group, and the IETF security area directors over many years.

References

[1] T. Stockhammer, "Dynamic adaptive streaming over HTTP – standards and design principles," in *Proceedings of the Conference on Multimedia Systems (MMSys)*. San Jose, CA, USA: ACM, February 2011.

[2] C. Jennings, T. Hardie, and M. Westerlund, "Real-time communications for the web," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 20–26, April 2013.

[3] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF, July 2003, RFC 3550.

[4] M. R. Macedonia and D. P. Brutzman, "MBone provides audio and video across the Internet," *IEEE Computer Magazine*, vol. 27, no. 4, pp. 30–36, April 1994.

[5] S. McCanne and V. Jacobson, "vic: A flexible framework for packet video," in *Proc. ACM Multimedia*, Nov. 1995.

[6] V. Hardman, M. A. Sasse, M. Handley, and A. Watson, "Reliable audio for use over the Internet," in *Proceedings of INET*. Honolulu, HI, USA: Internet Society, June 1995.

[7] M. Handley, J. Crowcroft, C. Bormann, and J. Ott, "Very large conferences on the Internet: the Internet multimedia conferencing architecture," in *Computer Networks*, vol. 31, no. 3, Feb. 1999.

[8] M. Westerlund and C. S. Perkins, "Options for securing RTP sessions," IETF, April 2014, RFC 7201.

[9] C. S. Perkins and M. Westerlund, "Securing the RTP framework: Why RTP does not mandate a single media security solution," IETF, April 2014, RFC 7202.

[10] E. Kohler, M. Handley, and S. Floyd, "Datagram congestion control protocol (DCCP)," IETF, March 2006, RFC 4340.

[11] R. Stewart, "Stream control transmission protocol," IETF, September 2007, RFC 4960.

[12] M. F. Nowlan *et al.*, "Fitting square pegs through round pipes: Unordered delivery wire-compatible with TCP and TLS," in *Proc. NSDI*. San Jose, CA, USA: USENIX, April 2012.

[13] B. Carpenter and F. Baker, "IAB and IESG statement on cryptographic technology and the Internet," IETF, August 1996, RFC 1984.

[14] J. Schiller, "Strong security requirements for IETF standard protocols," IETF, August 2002, RFC 3365.

[15] S. Farrell and H. Tschofenig, "Pervasive monitoring is an attack," IETF, May 2014, RFC 7258.

[16] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF, Jan. 1996, RFC 1889.

[17] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman, "The secure real-time transport protocol," IETF, March 2004.

[18] J. Ott, J. Chesterfield, and E. Schooler, "RTCP extensions for SSM sessions w/unicast feedback," IETF, Feb. 2010, RFC 5760.

[19] H. Schulzrinne, A. Rao, and R. Lanphier, "Real time streaming protocol (RTSP)," IETF, April 1998, RFC 2326.

[20] L. Gharai, C. S. Perkins, G. Goncher, and A. Mankin, "RTP payload format for Society of Motion Picture and Television Engineers (SMPTE) 292M video," IETF, March 2003, RFC 3497.

[21] P. Bellows, J. Flidr, L. Gharai, C. S. Perkins, P. Chodowiec, and K. Gaj, "IPsec-protected transport of HDTV over IP," in *Proceedings of the International Conference on Field Programmable Logic and Applications*, ser. Lecture Notes in Computer Science, no. 2778. Lisbon, Portgual: Springer, September 2003.

[22] E. Rescorla and N. Modadugu, "Datagram transport layer security version 1.2," IETF, Jan. 2012, RFC 6347.

[23] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2," IETF, April 2008, RFC 5246.

[24] J. Lennox, "Connection-oriented media transport over the transport layer security (TLS) protocol in the session description protocol (SDP)," IETF, July 2006, RFC 4572.

[25] S. Kent and K. Seo, "Security architecture for the Internet Protocol," IETF, December 2005, RFC 4301.

[26] Internet Streaming Media Alliance, "ISMA encryption and authentication, version 2.0," November 2007.

[27] J. Fischl, H. Tschofenig, and E. Rescorla, "Framework for establishing a SRTP security context using DTLS," IETF, May 2010, RFC 5763.

[28] D. McGrew and E. Rescorla, "DTLS extension to establish keys for SRTP," IETF, May 2010, RFC 5764.

[29] E. Rescorla, "WebRTC security architecture," IETF, July 2014, work in progress.

[30] J. Arkko, E. Carrara, F. Lindholm, M. Naslund, and K. Norrman, "Multimedia internet keying," IETF, August 2004, RFC 3830.

[31] F. Andreasen, M. Baugher, and D. Wing, "SDP security descriptions for media streams," IETF, July 2006, RFC 4568.

[32] P. Zimmerman, A. Johnston, and J. Callas, "ZRTP: Media path key agreement for unicast secure RTP," IETF, April 2011, RFC 6189.

[33] H. Alvestrand, "Overview: Real time protocols for browser-based applications," IETF, July 2014, work in progress.

[34] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Rey, "RTP profile for RTCP-based feedback," IETF, July 2006, RFC 4585.