# Watercasting: Distributed Watermarking of Multicast Media

Ian Brown, Colin Perkins, and Jon Crowcroft

Department of Computer Science, University College London
Gower Street, London WC1E 6BT, UK
{I.Brown, C.Perkins, J.Crowcroft}@cs.ucl.ac.uk

**Abstract.** We outline a scheme by which encrypted multicast audiovisual data may be watermarked by lightweight active network components in the multicast tree. Every recipient receives a slightly different version of the marked data, allowing those who illegally re-sell that data to be traced. Groups of cheating users or multicast routers can also be traced. There is a relationship between the requirements for the scheme proposed here, the requirements for reliable multicast protocols, and proposed mechanisms to support layered delivery of streamed media in the Internet.

## 1 Introduction

When discussing multicast in the Internet, the loosely coupled model often causes information providers some disquiet. There are a number of reasons for this, but the one of interest to us is the relative anonymity of the receivers: traffic forwarding and group membership are local issues, and at no point is the complete group membership known unless an application level protocol is used to provide an approximate list of members (for example RTCP [26]). This anonymity implies that it is a simple matter to eavesdrop on, and record, a media stream in an undetectable manner.

There have been various proposals for protecting multicast data using public key authentication and encryption techniques, together with some suggestions for providing scalable key distribution for such services [2, 3]. These schemes rely on the cost of re-transmitting media data being large enough to deter paying customers from re-selling on content they have received. However, the Internet provides an environment where miscreants can easily re-transmit data without detection and on a large scale. Even if illegal copies are found, it may be difficult to determine which receiver was the source of those copies. It is therefore necessary to provide a means for tracing illegal copies, to deter would-be thieves.

One way to provide an audit trail of the origin of a copy is to use a technique known as *watermarking*. This entails embedding additional, virtually undetectable, information in the data to distinguish one copy from another. There is, however, a problem with using watermarking with broadcast or multicast media: the addition of distinguishing marks to the data stream is contrary to the bandwidth saving of being able to distribute the same data efficiently to multiple recipients.

We propose to use lightweight active network components at the branch points in an IP multicast network to modify a media stream such that recipients are delivered unique

versions of the sourced data in an efficient manner. Our method builds upon some of the mechanisms proposed to provide support for reliable multicast data delivery and for delivery of scalable, layered, streamed multicast data (such as video and audio), and is part of a family of end-to-end services that can make use of a distributed, heterogeneous and dynamic set of filters in a multicast distribution tree.

We consider here the typical content provider case: one source sending data to many recipients.

In the next section we describe related work on watermarking and multicast security. Following that, we outline the salient points of our approach and analyse its efficiency and possible threats to its effectiveness. Finally, we note directions for further study.

## 2 Background

### 2.1 Watermarking

Typically, in broadcast networks, cryptographic techniques are sufficient to give a sufficient level of protection against dishonest reception. The major problem is legitimate users illegally selling their keys to others. Schemes have been designed to make keys effectively as large as the content they protect, and hence uneconomic to sell [9], or to identify the source of stolen keys [21] even when legitimate users collude to try and cover their trail. These schemes are no protection against retransmission of content rather than keys, but this is not typically a problem since retransmission of content is difficult and expensive.

In the Internet environment, retransmission of content is simple and cheap so further protection is essential. Watermarking allows content providers to trace any illegally redistributed data they discover back to a subscriber by subtly marking the data sent to each user in a way that is difficult to reverse.

The simplest watermarking schemes use the low bits in an audio or image file to embed information such as subscriber ID. These are easily defeated by altering these bits. More complex schemes select a subset of bits to alter using a secret key, or use spread spectrum techniques or complex transformations of the data to make removal of the watermark more difficult.

Anderson and Manifavas describe an ingenious scheme that allows a single broadcast ciphertext to be decrypted to slightly different plaintexts by users with slightly different keys. Unfortunately the scheme is extremely vulnerable to collusion between users. Five or more users can together produce plaintext (or keys for installation in pirate decoders) that cannot be traced. Shamir has pointed out that increasing collusion resistance in all of these schemes requires exponential work from the defender to cost the attacker linearly more effort [1].

Regrettably this is the only scheme proposed so far that allows efficient marking of data being supplied to large numbers of users. While others are not hugely computationally intensive, they would not scale well. A content provider would need huge computing resources to be able to watermark data being sent to typical live event audiences. An enormous amount of bandwidth would also be used up sending a different version of the data to each viewer. This motivates our approach, which distributes the processing needed throughout the multicast tree used to efficiently deliver data.

### 2.2 Multicast Security

The IP multicast model allows for *any* receiver to become a sender, subject to their ISP's policy and pricing mechanisms. Unlike traditional broadcast networks, the effort needed to re-multicast data is small. Further, any host can receive multicast traffic, with the decision to route data to that host being made close to that host with no reference to the original source of the data. It becomes clear that, in addition to marking data to trace those who illegally redistribute it, we also need to protect data in transit to prevent unauthorised access by eavesdroppers.

By encrypting packets, we ensure only those possessing the necessary key can access content. This encryption is typically performed at the application level, although in the future it may be integrated into the forwarding mechanisms [15]. Whilst the problem of key management for multicast data is not yet completely solved, proposals to provide this functionality [12] are moving forward and could build on content providers' systems for authenticating users. With rapid re-keying, content providers could remove pirates from groups even quicker than current pay-per-view systems.

It is possible to limit multicast traffic to a specific region of the network, using administratively scoped addressing [19]. This relies on border routers of the administrative region being correctly configured to prevent traffic sent to certain address ranges leaking out of the region. It provides an effective means of limiting the flow of traffic if correctly configured, but does not prevent unauthorized reception of data by hosts within the region. It is also difficult to configure and use, although future protocol developments may ease these problems [11].

To summarise, we note that it is almost impossible to limit access to multicast data. We must rely on encryption and good key management to prevent intercepted traffic being decoded, and watermarking to trace authorised users who illegally redistribute content.

### 2.3 Multicast Loss Characteristics

A number of studies have been conducted into the performance and loss characteristics of the Mbone [10, 30]. These have shown a large amount of heterogeneity in the reception quality for multiple receivers in a single session, posing a challenge to designers of resilient multicast streaming protocols [8, 22] and reliable multicast transport protocols [16].

The *loss signature* of a receiver is used by a number of protocols to identify subsets of receivers which belong, at least symptomatically, to shared subtrees. This signature has two components: the temporal pattern of packet loss, and the correlation between the position of a receiver in the multicast distribution tree and the observed loss.

The temporal correlation of packet loss has been noted by a number of authors. Bolot [5] noted that packet loss is not independent (it is more likely that a packet is dropped if the previous packet was also lost) and derived a simple Bernoulli model for such loss. More recent work [31, 20] notes that this model is not sufficient in many cases, and that higher-order Markov models are more accurate.

Correlation is also noticable at longer time-scales. For example, Handley [10] and Bolot [5] have noted bursts of loss with a 30 second period (possibly due to router bugs) and the authors have noted similar effects.

Packet loss is also correlated between receivers, such that many receivers see the same patterns of loss [30]. This is clearly due to lossy links within the distribution tree which cause loss for all leaf nodes below them. Packet loss correlation is a therefore a good predictor for the shape of the multicast distribution tree [17].

One significant drawback of these techniques is that whilst loss signatures match the network topology to a fairly high accuracy, they do not allow the topology to be discovered directly, although recent work has shown that it is possible to infer the *logical* network topology based on packet loss measurements [7, 24].

### 2.4 Reliable Multicast: Router Support

Reliably multicasting a packet to a large group of receivers becomes more efficient if the network acts to ensure reliability. Recently, a number of proposals have been made to add such reliability into the network. One of these, PGM [27], provides a close fit for our requirements for a watermarking scheme.

PGM is "a reliable transport protocol for applications that require ordered, duplicate-free, multicast data delivery from multiple sources to multiple receivers". To achieve reliability, receivers send negative acknowledgements (NAKs) which are reliably propogated up the multicast distribution tree towards the sender, with the aid of the routers. Retransmissions of lost data are provided by the sender or by designated local retransmittors.

Two mechanisms are incorporated to prevent NAK implosion: on detecting loss, receivers employ a random backoff delay before sending a NAK with suppression if a NAK is received from another receiver. In addition, routers which receive duplicate NAKs from multiple downstream links eliminate the duplicates, sending only a single NAK up towards the source.

The result is a timely and efficient means by which NAKs can be returned to the source of multicast data, allowing either retransmission of that data or addition of FEC to the stream to ensure reliable delivery.

In addition to providing an efficient NAK delivery and summarisation service, PGM offers a number of end-to-end options to support fragmentation, sequence number ranges, late joins, time-stamps, reception quality reports, sequence number dropout and redirection.

Of interest to us is the sequence number dropout option. This allows placement of "intermediate application-layer filters" in routers. Such filters allow the routers to selectively discard data packets and convey the resulting sequence number discontinuity to receivers such that sequencing can be preserved across the dropout, and to suppress NAKs for those packets intentionally discarded. They act as lightweight active network elements, modifying data streams passing through them. The operation of these filters is not defined by PGM. In later sections of this paper, we describe semantics for these filters suitable for watermarking multicast streams.

### 2.5 Reliable Multicast: Layering and FEC

The use of packet-level forward error correction data to recover from loss is well-known. For every $k$ data packets, $n - k$ FEC packets are generated, for the transmission of $n$

packets over the network. For every transmission group of $n$ packets it is necessary to receive only a subset to reconstruct the original data.

There are a number of means by which these FEC packets may be transmitted. The three primary means are by piggy-backing them onto previous packets, sending them as part of the same stream but with a different payload type indicator or sending them as a separate stream.

Sending FEC packets within the same stream as the original data has the advantage of reducing overheads (routers only need keep state for a single stream), but forces all receivers to receive the FEC data in addition to the original data. If a receiver is not experiencing loss, this is clearly wasteful.

Sending the FEC data on a different stream has greater overhead (because routers need keep state for multiple flows), but allows for greater flexibility. Those receivers which are not experiencing loss do not join the multicast group transporting the FEC stream, and hence do not receive the FEC data; varying amounts of FEC can be supplied, layered over a range of groups, giving different levels of protection; or enhancement layers can be provided – not FEC but additional data to improve the quality of the stream for those on high capacity links who are not experiencing loss.

This use of layered transmission to provide either FEC or differing quality has been studied by a number of authors [28, 18] and shown to perform well.

## 3 Protocol Overview

Given that the loss signature of a receiver corresponds to its position in the network, it should be possible to use this as a simple form of digital watermark. The pattern of degradation in a stream will likely be different for each receiver provided there is a non-zero packet loss rate in the network (see section 2.3). There are four problems with this:

1. A receiver may neglect to send a loss signature back to the sender, escaping notice by the watermarking scheme.
2. Lost packets cause degradation of the delivered stream. A network which drops enough packets to make this watermarking technique successful will likely provide insufficient quality for most uses.
3. A receiver may collude with another receiver to repair the loss, hence defeating the watermarking scheme.
4. A receiver may easily defeat the watermarking scheme by dropping additional packets (possibly transforming the stream to match that received by another receiver).

Ensuring that a receiver returns its loss signature to the sender is clearly an impossible task in the traditional Internet environment with smart end-points and dumb routers. However, if an active network is assumed it becomes possible for the last-hop router to return a loss signature to the sender. If the installation of this active element forms part of the multicast tree setup procedure, we may ensure that the loss signature of each receiver is returned to the source.

The active network elements can also conspire to ensure that all receivers see unique loss patterns, rather than leaving this to chance. Instead of relying on the loss signature of a particular branch in the multicast forwarding tree being unique, the position of a node

in the tree can be used to determine which packets to drop in order to ensure a unique loss pattern for each node.

The proposed use of active network components is not unique to our scheme; a number of reliable multicast protocols have been developed which would benefit from support within the network. This support typically takes the form of filtering, summarisation and subcasting abilities: exactly the requirements for our scheme (see section 2.4).

The assumption of an active network leaves three barriers to the development of an effective watermarking solution: degradation of the stream by packet loss, collusion attacks by multiple receivers to repair the stream, and the ease of breaking the protection by dropping additional packets. These three problems are related, and have a common solution.

A typical counter to the problem of packet loss in a multicast network is to add forward-error correction data to a media stream (section 2.5). This allows a stream to be repaired if some fraction of the packets are lost. We modify this approach by sending FEC data which is subtly different to the original data, such that a stream repaired using this FEC data will differ based on the observed loss pattern, but will not be noticably degraded.

This altering of the media stream is typically straightforward, although content specific. It is vital that the set of transformed packets resulting from one packet cannot be used to recreate the original, otherwise a collusion attack could produce a non-watermarked version of the data. Likewise, the watermark must be resistant to a wide range of transforms, such as the introduction of jitter or re-sampling [23].

The active network elements therefore *subtract* FEC packets. Rather than ensuring a unique loss pattern at each receiver, they ensure a unique pattern of packets is received. This may be implemented using the PGM sequence number dropout option and application layer filters as noted in section 2.4.

This solves the quality degradation problem: since some version of each packet is received by each participant the reception quality is no worse than that provided by the underlying network, although each receiver sees a slightly different stream. Receivers can no longer collude to repair a stream (the result will simply be a combination of their watermarks, enabling identification of the conspirators). Finally, discarding additional packets simply results in a degraded stream with the watermark still present.

The result is a relatively simple means of watermarking multicast data: the source sends multiple subtly different copies of each packet. Routers at the branch points in the network discard packets, such that the stream delivered to each receiver is unique.

## 4 Implementation Strategy

We have designed an initial protocol to implement watercasting using PGM and slight modifications to multicast tree setup. These could be made using active network code in routers. We hope to refine this protocol after gaining implementation experience.

A client wishing to receive content from a server first performs a unicast authentication with that server. After convincing the server it is a valid subscriber, the client is given a receiver identification key.

This key is supplied to the last hop router when the receiver joins the session. The last hop router passes this key, its address and the time the receiver joined back up the

multicast tree to the source. Each router in the tree adds its address, encrypted with the public key of the server to keep the topology secret. This is a slight variation on the current Internet Group Management Protocol MTRACE packet. When the source receives a valid RIK and topology report, it unicasts the current session key(s) for the requested media to the client.

This server is therefore able to validate and store the entire tree topology. This information is necessary to allow it to later determine the correct watermark for each receiver, in the event that it becomes necessary to trace an illegal copy of the content.

For a multicast distribution tree with maximum depth $d$, the source generates a total of $n$ differently watermarked copies of each packet such that $n \geq d$. Each group of $n$ alternate packets is termed a transmission group.

On receiving the packets which form a transmission group, a router forwards all but one of those packets out of each downstream interface on which there are receivers. The choice of which packet to discard is made at random on a per-interface basis, with the pseudo-random sequence keyed by the position of the router in the distribution tree and the interface address.

Each last hop router in the distribution tree will receive $n - d_r$ packets from each transmission group, where $d_r$ is the depth of the route through the distribution tree to this router. Exactly one of these packets will be forwarded onto the subnet with the receiver(s). The choice of which packet is forwarded is determined pseudo-randomly, keyed with the position of the router in the tree, the interface address and the receiver identification key.

The filtering process is illustrated in figure 1. In this example, the receiver furthest from the source is R1 and the maximum depth of the distribution tree, $d = 5$. The source will generate $n \geq 5$ distinct versions of each packet to form a transmission group; label these ABCDE. At router 0 these are filtered, passing ABDE to router 00 and ACDE to router 01. At router 01 the packets are filtered again, with ACE being passed to router 010. Since this is the last hop router before receiver R2, router 010 does not just filter out a single packet, it pseudo-randomly selects one packet, E, to pass to the receiver. A similar process occurs to filter the packets destined for the other receivers.

The media payload is encrypted. The receiver also receives a decryption key from the source by the same means that it receives the receiver identification key. Media packet headers are not encrypted, since routers need to use the sequence numbers in the packets to determine which packets to discard.

The encryption of the media payload prevents unauthorised receivers from snooping on the packets. The watermark may be used to detect illegal redistribution of the decrypted payload by legitimate receivers.

In effect, the combination of tree topology and receiver identification key is the secret used by the source to do the watermarking. Participating routers should therefore refuse requests to reveal any part of that topology. Even if some routers and clients collude, they would need a conspiracy from a client right up to the source to discover anything useful.

The selective discard function aims to provide the multicast routers and their clients the minimum degree of freedom possible in order to facilitate the later tracing of cheating routers or users. Every router in the tree indelibly affects the stream by dropping

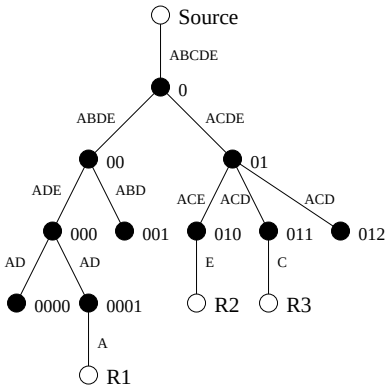


**Fig. 1.** Filtering transmission groups to obtain a unique watermark

certain packets. A cheating downstream router therefore cannot produce a stream seemingly originating from an upstream router, as it does not have access to all of the packets passing through that router. It would need to collude with all other branches of that router to get copies of all data passing through it. The higher up the tree, the less likely this is, the lower down the tree, the easier to eliminate targets from an investigation. An upstream router could attempt to impersonate a downstream router or client, but would need to know the topology of the tree to that point to do so effectively. This becomes increasingly difficult as the distance of the point from the cheating router increases, which may be sufficient protection in that points upstream of a suspected router could be included in any investigation of that router. Alternatively, each router could be given a shared secret by the source at the time it joins the multicast tree, and include that secret in the initialisation of its pseudorandom number generator.

We keep as much of the processing as possible at the source to simplify the router protocol. For each watercast stream a router is processing, it only needs to store the sequence state to allow it to drop the appropriate packet in each transmission group. We place a heavier burden on the source: it needs to appropriately modify the outgoing stream and inject sufficient redundant packets to maintain quality for all clients whilst allowing enough packets to be dropped to watermark each client's stream uniquely.

The source also needs to store enough information to enable it to later reconstruct the path to a watermark found in a recovered media clip. Because the watermarking algorithms, and the pseudo-random sequence generator, operate in a deterministic manner, this comprises the original data, the topology of the distribution tree, and details of when receivers join and leave.

Given a recovered clip, the server can determine how many redundant packets it was sending out at that time and hence which transformations were being applied. It can then reduce the clip to a series of packet labels (such as AECDBBABC using the example in section 4). By simulating the operation of various network components from the start of the transmission of the original broadcast through to the end of the clip, it can aggres-

sively rule out nodes that could not have produced the clip because they did not have access to any of the packets present in it.

The completion of this process will result in a set of nodes that could have produced the given clip. The length of clip required for this result depends on the multicast tree topology and pseudo-random sequence generator used. This can vary continuously according to the topology of the multicast tree at the time of transmission of the marked data, something known only to the source. This makes it difficult even for conspiracies of users and routers to remove the watermark or alter it to implicate someone else. The probability that the media clip must have originated with a particular receiver increases with the length of the clip.

## 5 Analysis

The two important effects of watercasting are reducing the number of unique watermarked copies of data required, and distributing the task of selecting a different sequence of watermarked packets to each recipient. The results of these gains are considered below.

The probability that a particular version, $v$, of a packet is received, given that the transmission group size is $n$ and the receiver is $d$ hops from the source, $p$, is simply

$$p_v = \frac{1}{n-d+2} \cdot \prod_{i=1}^{d-2} \frac{n-i}{n-i+1} = \frac{1}{n} \tag{1}$$

The probability that multiple receivers receive the same version of a packet depends on the position of those receivers in the distribution tree. The closer those receivers are, that is the longer the shared path from the source, the more likely they are to receive the same version. This can be offset by increasing the number of packets in a transmission group.

Introducing redundant data into the multicast stream increases the size of the stream by the number of packets per transmission group at the first hop, then one less at the second hop, and so on until the last hop where the traffic size is the same as for a non-watermarked stream. If $n$ is the group size and $d$ the maximum depth of the tree, this increases the amount of traffic by a factor of

$$\frac{2n-d+3}{2} - \frac{n}{d} \tag{2}$$

This is still far less than the traffic that would be generated by unicasting unique versions of each stream to every receiver. If we set $n = d$, which creates the minimum extra traffic but makes the watermark sequence longer, this factor is

$$\frac{d+1}{2} \tag{3}$$

At the cost of greater complexity at routers, this figure could be decreased still further. If each router knows the maximum depth of each of its interface's subtrees, it need only send the minimum number of redundant packets necessary to each child node. Rather

than choosing one packet from each transmission group to drop at random, it selects $n - d_i$ packets to drop, where $d_i$ is the depth of the subtree on that interface. This reduces the extra bandwidth consumed on all subtrees that are shallower than the maximum depth of the tree. It also reduces the scope of attacks by cheating downstream routers, which have fewer versions of each packet to use.

Using a value of $n > d$ allows a tree to grow more easily. At the initial setup of a tree, this is particularly important: as many new members join, continually altering the depth of the tree, the source would otherwise need to constantly increase $n$. By setting $n$ to the likely depth of the tree after this phase, the source reduces this update complexity, at the cost of greater bandwidth requirements over the (initially small) tree as it is set up. The source may use knowledge of other likely tree changes, or react to a rapidly-changing depth, by discontinuously varying $n$ in the same manner.

There is therefore a tradeoff: larger values of $n$ allow greater flexibility in tree setup and reduce the length of the watermark sequence required to trace the originator of data, but require greater processing and bandwidth to create and distribute.

An obvious optimisation would seem to be only watermarking 1 in every $x$ packets, reducing bandwidth and computation requirements by a factor close to $x$. But because every router in the multicast tree would need to know the location of the watermarked packets to run the watercasting algorithm, it would be impossible to keep this information secret.

Unfortunately, if an attacker knew the position of the watermarked packets, she could simply remove them and redistribute the resulting degraded data. While this would be fatal to data such as executable code, it may be acceptable for lossy information such as an audiovisual signal. An information provider must consider the quality of data they are effectively prepared to give away before using this technique.

When the last-hop network is a multi-access subnet, such as an Ethernet, any host on that network can receive the same packet with no extra effort. Non-subscribers on a network can intercept such packets, but do not have the decryption key needed to read them. But two legitimate subscribers on the same sub-network will receive the same watermarked data. We contend that this is a small problem: multi-access sub-networks are typically under the administrative control of a single agency. If one of the users of such a network illegally resold data, it would be traceable to the agency controlling that network, which is sufficient in most cases.

Collaborative conspiracies are always a difficult problem for watermarking schemes. Groups of users can attempt to combine their different watermarked versions of the same piece of data in a way that removes or at least damages the watermark. The simplest way to do this is perform 'bit voting': set each bit in the reconstructed piece of data to be that which is most prevelant in the same bit in the set of watermarked files. This is usually fatal to simple schemes, and can damage more sophisticated watermarks.

The watermarking techniques we use must therefore be able to defeat collaborative and individual attacks such as introducing jitter and re-sampling [23]. But the main contribution of our paper is that an active network can perform part of the watermarking function. Even if the specific transforms we use can be defeated, we hope that it should be possible to simply plug in others more resistant to attacks, preserving the validity of our approach.

## 6 Conclusions and Future Work

We have outlined a general idea for watermarking media data using active network components, and a specific method to perform that task. Our method leverages schemes such as PGM that are being developed to provide other services such as reliable multicast and filtering in active networks.

Traditional communications security systems protect data *en route* to recipients by encrypting it with a key known only to authorised users. This is effective at preventing eavesdropping of the data in transit, but can do nothing to stop authorised recipients redistributing the data. This is a major problem for 'secrets' such as live sporting events shared between millions of paying customers. As the cost of redistributing data continues to plummet, watermarking is likely to become as essential to information security as cryptography is today.

In conventional watermarking schemes, each recipient gets a different version of the watermarked data to allow any cheating recipients who illegally redistribute the data to be traced. This is computationally expensive for the source, which needs to calculate a large number of unique versions of the data, and requires unicast transmission of the resulting data to clients. Our scheme reduces both loads. The source needs to calculate a far smaller number of versions of each packet of the data, and is relieved of the task of deciding which packet goes to which user by routers. This load is spread thinly throughout the distribution tree. The resulting data can be transmitted efficiently through the network at a cost in bandwidth over pure multicast related to the depth of the multicast tree rather than the number of recipients. This is particularly important for very large multimedia streams.

Watercasting requires a considerable amount of network complexity compared to content control schemes such as Nark [6]. It uses a trusted smartcard to provide the keys needed to decrypt data according to an access policy, thus scaling excellently even with a constantly changing membership. Watercasting is more appropriate for protecting high value content where sufficient incentive exists for an attacker to compromise a smartcard.

We are now performing simulations to model the performance of our method. Factors such as the size of transmission groups and complexity of filtering algorithms will have large effects on the performance of the system, and we intend to use our simulations to fine tune these parameters. We are also investigating optimisations such as increasing capacity near the multicast root for given sizes of receiver sets and the depth and breadth of the tree. Finally, we intend to build small test networks and distribute audiovisual and other data through them to experimentally verify our scheme and determine its implementation complexity, using these results to further develop the protocol.

The central task of watercasting is to provide *evidence.* Computer security systems often claim to reduce or obviate the need for legal solutions to a problem by removing it through technical means. Our design instead aims to provide an audit trail through which the illegal distributors of a given piece of data can be traced and prosecuted. The strength of the evidence provided by watercasting is crucial to the ability to mount a successful trial, particularly if no other evidence is available. We are working with legal researchers to determine how to best fine tune our scheme to meet this aim. We are also

investigating the requirements of content providers: 'fair use' provisions in copyright laws, for example, may reduce their need for the tracing of very short clips.

Watercasting has wide applicability to the protection of any data that is distributed to a large number of people via a network. While we have focussed on audiovisual data, other information such as software could be equally covered with the development of appropriate transforms. Indeed, an appropriate software architecture would allow small pieces of transform code to be plugged-in to our system to extend it with minimum effort.

The authenticity of such data may be more important to clients than that of a video broadcast. It would be trivial to put a public-key Authentication Header in each packet [14] and so assure clients of the information's integrity and origin. Servers authenticating large amounts of data could use a hybrid scheme combining public-key certificates and k-time signature schemes that allows offline pre-computation of expensive parameters so that even bursty, lossy and low-latency streams can be authenticated [25].

Watermarking technology is still in its infancy. Petitcolas *et al*. [23] hoped that their attacks on first-generation algorithms would lead to an improved second generation, and so on. We hope our system is reasonably resistant to the attacks they designed, but we will no doubt see further ones developed.

Our design criteria are slightly less robust than those of, for example, the International Federation for the Phonographic Industry (IFPI), who required a watermark that could not be removed or altered "without sufficient degradation of the sound quality as to render it unusable" [13]. Our definition of unusable is not unlistenable or even unsellable, but simply perceptually intrusive enough to justify paying for the original rather than a pirated copy. We plan to use tools developed for measuring subjective audio quality [29] to assess the impact of removing the watermarks we develop.

We also believe that the best use for our system is the transmission of 'live' data. As Barlow observed [4], the value of such transmissions drop rapidly as they age. The live TV rights to a popular sporting event are worth a considerable amount, but this drops dramatically once the game is finished and the result is known.

Therefore, even if our watermarking scheme can be defeated, as long as it takes a reasonable amount of time to do so it will have achieved its main objective — to prevent large profits being available from the illegal re-distribution of content.

## 7 Acknowledgements

Thanks to the IRTF Reliable Multicast Research Group and the network multimedia research group at UCL for discussion that led to some of these ideas. Thanks also to Bob Briscoe and the anonymous referees for their useful comments.

## References

1. R.J. Anderson and C. Manifavas. Chameleon – A New Kind of Stream Cipher. Fourth Workshop on Fast Software Encryption, pp. 107-113, January 1997.
2. A. Ballardie. Scalable Multicast Key Distribution. RFC 1949, May 1996.
3. A. Ballardie and J. Crowcroft. Multicast-Specific Security Threats and Counter-Measures. The Internet Society Symposium on Network and Distributed System Security, San Diego, California, February 1995.

4. J. P. Barlow. The economy of ideas. Wired 2(3) pp. 85, March 1994.
5. J.-C. Bolot and A. Vega-García. The Case for FEC-Based Error Control for Packet Audio in the Internet. ACM Multimedia Systems, 1997.
6. Bob Briscoe and Ian Fairman. Nark: Receiver-based Multicast Non-repudiation and Key Management. ACM Conference on Electronic Commerce (EC-99), Denver, Colorado, November 1999.
7. R. Cáceres, N. G. Duffield, J. Horowitz, D. Towsley and T. Bu. Multicast-Based Inference of Network-Internal Characteristics: Accuracy of Packet Loss Estimation. IEEE Infocom, New York, March 1999.
8. G. Carle and E. Biersack. A Survey of Error Recovery Techniques for IP-Based Audio-Visual Multicast Applications. IEEE Network, November/December 1997.
9. C. Dwork, J. Lotspiech and M. Naor. Digital Signets: Self Enforcing Protection of Digital Information. 28th Annual ACM Symposium on the Theory of Computing, Philadelphia, pp. 489, May 1996.
10. M. Handley. An Examination of Mbone Performance. USC/ISI Research Report ISI/RR-97-450, April 1997.
11. M. Handley and D. Thaler. Multicast-Scope Zone Announcement Protocol. IETF work in progress, October 1998.
12. T. Hardjono, B. Cain and N. Doraswamy. A Framework for Group Key Management for Multicast Security. IETF work in progress, July 1998.
13. International Federation of the Phonographic Industry. Request for proposals — Embedded signalling systems issue 1.0. June 1997.
14. S. Kent and R. Atkinson. IP Authentication Header. RFC 2402, November 1998.
15. S. Kent and R. Atkinson. IP Encapsulating Security Payload. RFC 2406, November 1998.
16. B. Levine and J. J. Garcia-Luna-Aceves. A Comparison of Reliable Multicast Protocols. ACM Multimedia Systems Journal, August 1998.
17. B. Levine, S. Paul and J. J. Garcia-Luna-Aceves. Organizing Multicast Receivers Deterministically by Packet-Loss Correlation. Preprint, University of California, Santa Cruz.
18. S. McCanne, V. Jacobson and M. Vetterli. Receiver-driven Layered Multicast. ACM SIGCOMM'96, Stanford, CA, August 1996.
19. D. Meyer. Administratively Scoped IP Multicast. RFC 2365, July 1998.
20. S. Moon, J. Kurose, P. Skelly and D. Towsley. Correlation of Packet Delay and Loss in the Internet. Technical Report 98-11, Department of Computer Science, University of Massachusetts, Amherst, MA 01003, USA.
21. M. Naor and B. Pinkas. Threshold Traitor Tracing. Advances in Cryptology – CRYPTO '88, Lecture Notes in Computer Science 403, August 1988.
22. C. Perkins, O. Hodson and V. Hardman. A Survey of Packet Loss Recovery Techniques for Streaming Audio. IEEE Network, pp. 40–49, September/October 1998.
23. F. A. P. Petitcolas, R. J. Anderson and M. G. Kuhn. Attacks on copyright marking systems. Second Workshop on Information Hiding, Portland, Oregon, April 1998.
24. S. Ratnasamy and S. McCanne. Inference of Multicast Routing Trees and Bottleneck Bandwidths using End-to-end Measurements. IEEE Infocom, New York, March 1999.
25. P. Rohatgi. A Hybrid Signature Scheme for Multicast Source Authentication. IRTF work in progress, June 1999.
26. H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson. RTP: A Transport Protocol for Real-time Applications. RFC 1889, January 1996.
27. T. Speakman, D. Farinacci, S. Lin and A. Tweedly. PGM Reliable Transport Protocol Specification. IETF work in Progress, February 1999.
28. L. Vicisano, L. Rizzo and J. Crowcroft. TCP-like congestion control for layered multicast data transfer. IEEE Infocom, San Francisco, March 1998.

29. A. Watson and M. A. Sasse. Measuring Perceived Quality of Speech and Video in Multimedia Conferencing Applications. ACM Multimedia '98, Bristol, England, pp. 55-60, September 1998.
30. M. Yajnik, J. Kurose and D. Towsley. Packet Loss Correlation in the Mbone Multicast Network. IEEE Global Internet Conference, November 1996.
31. M. Yajnik, S. Moon, J. Kurose and D. Towsley. Measurement and Modelling of the Temportal Dependence in Packet Loss. IEEE Infocom, New York, March 1999.