VIDEO-DNA: LARGE-SCALE SERVER-SIDE WATERMARKING

Ivan Kopilovic[†]*, Virginie Drugeon*[††]*, and Marcel Wagner*[†]

[†] Nokia Siemens Networks GmbH & Co. KG
Otto-Hahn-Ring 6, 81739 Munich, Germany
phone: + (49) 89 636 41460, fax: + (49) 89 636 51115, email: {ivan.kopilovic|marcel.wagner}@nsn.com
web: www.nokiasiemensnetworks.com

[††] Panasonic Frankfurt Laboratory, Panasonic R&D Center Germany GmbH
Monzastrasse 4c, 63225 Langen, Germany
phone: + (49) 6103 766 240, fax: +(49) 6103 766 144, email: virginie.drugeon@eu.panasonic.com

ABSTRACT

We introduce a new server-side watermarking method for video-on-demand systems which we call "VideoDNA". The method is session based, i.e., each video stream receives an individual watermark (in some references also called fingerprint). The method is generic, since it can be integrated in any video-on-demand system without essential performance reduction, without increase in the transmission bandwidth, and without requiring changes in the network and in the receivers. There is no need for decoding or decrypting the video files on the server, which provides additional security. Finally, the method is large scale, since it does not significantly increase the complexity of streaming servers and, thus, the number of parallel streaming sessions that can be watermarked is only limited by the server architecture.

1. INTRODUCTION

Protecting the delivered content against piracy is crucial for video-on-demand (VoD) applications. The problem of efficient and reliable content protection is especially difficult for large scale systems that deliver videos to hundreds of thousands of users round the clock.

The conventional way of protection is to keep the videos encrypted during the whole content processing and delivery chain. However, several possibilities remain for a pirate to access and capture the video once it has been decoded for display, or to break the encryption once the stream has been stored on a local storage device. A possible countermeasure is to use watermarks. A watermark is a signal directly embedded into the video or audio data that carries a piece of copyright or usage information [1]. It has to be invisible and it has to survive common video processing and format conversion. Though severe signal processing can destroy virtually any watermark, the resulting quality and user experience are essentially reduced. In order to identify the origin of the piracy, several different watermarks may be embedded into a single video along the consecutive stages of the content delivery chain, including a terminal user or session specific watermark, in some references also referred to as fingerprint.

The above concept faces a serious bottleneck: the "on-the-fly" insertion of a user or session specific watermark with conventional approaches break down for a large scale VoD system:

- Conventional server-side watermarking methods need to decrypt and partially decode then re-encode end re-encrypt the video files for each streaming session. This approach becomes especially infeasible with advanced video compression methods like MPEG-4 AVC/H.264 [2] due to their complexity.
- Conventional client-side watermarking methods embed the watermark during decoding and display, which besides being a security issues, binds memory and computational resources, a serious problem for low-cost consumer devices (set-top-boxes).

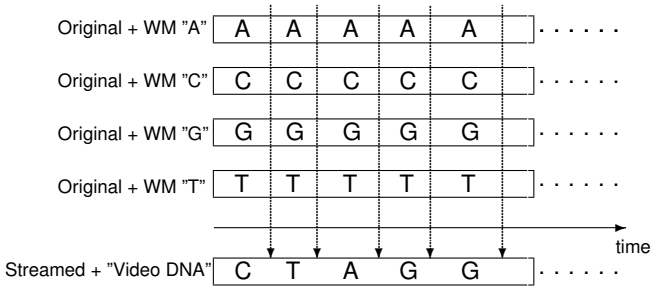

Figure 1: The streaming server embeds the individual, session specific pattern ("Video DNA") by switching between the different watermarked versions of the video file.

The scalability problem has been addressed in [3] for the multimedia broadcast scenario. In the proposed solution, the server sends two different copies of each multimedia packet to the multicast channel. The two copies are first watermarked with two different watermarks, then encrypted with two different keys. Each client receives only one of the both keys, so it has access to only one of the packets. The user or session specific information is described by the sequence of decryption keys sent to the receiver. The resulting decoded multimedia stream is a concatenation of packets with different watermarks. The embedded information is retrieved by analyzing the temporal variation of the watermark pattern. Similar methods based on decryption key distribution have been proposed for broadcast encryption and traitor tracing [4]. The main challenge addressed by these methods is the key distribution scheme, which is designed to be able to trace down the compromised devices and to disable them if necessary. However, the main drawback of these methods is that they multiply the bit rate of the multimedia stream, which is especially infeasible for unicast scenarios like VoD. Additionally, they require trusted clients and complex key management.

In this paper, we propose a system that solves the scalability problem for the unicast scenario without increasing the streaming bit rate and that is fully transparent to the network and to the receiver. Our main focus is to solve the technical challenge involved with a transparent server based solution.

2. METHOD DESCRIPTION

The method is demonstrated in Fig. 1. During the production, the original video is marked with $n \geq 2$ different watermarks resulting in n copies. The figure shows an example with $n = 4$. The videos are partitioned into M segments along the temporal axis. The length of each of these temporal segments may change randomly, but it must be identically chosen for all n copies. Finally, the n videos are

encoded in such a way that each temporal segment starts with a random access point. These steps are done only once in the production phase. The actual user or session specific information is embedded during the streaming by switching between the n compressed video files at the specified temporal segment boundaries. The number of bits that can be embedded in this way is $M \log_2 n$.

The advantages of this approach are:

- To embed the watermark into the n copies of the video, we can use any existing watermark embedding method. The embedding time is of no concern, since the procedure is offline. The robustness properties of the embedding method are inherited by VideoDNA.
- The embedding of the actual session-based watermark by stream switching has a very low complexity. The scalability properties of the server in terms of the number of parallel streaming sessions are inherited by VideoDNA.
- The resulting bit rate will not increase, since we can encode the n copies with the same bit rate (we discuss this issue in Section 3.2).
- The method is fully transparent for the system, no changes are needed in the network or in the receivers.
- The files on the server can even be stored encrypted, this remains completely transparent to the switching procedure. The only requirement is that the encryption keys are identical for each of the n files, otherwise the decryption would be desynchronized.

One disadvantage of this method is that the storage space required by the server increases by the factor of n. However, this may compensated by the fact that making a change in the server is probably easier than requiring changes in or exchanging thousands of receivers or network components in an existing system infrastructure.

3. IMPLEMENTATION ISSUES

To test the VideoDNA approach, we chose the watermark embedding method called JAWS (Just Another Watermarking Method) [5, 6, 7], because it is well known, well documented, competitive, and often used as reference in the watermarking research [1]. We would like to address the main challenges in the implementation of VideoDNA, namely how to design an appropriate representation for the session information for the embedding with JAWS, and how to ensure that the stream switching procedure does not affect the VoD network and receiver operation.

3.1 Watermark embedding and detection

We have seen that for $n \geq 2$ copies of the video stream, n different watermark signals are needed. However, consider the following situation. We are given a string of symbols $a_1 a_2 \ldots a_M$ with $a_i = a_{i+1}$ for some $1 \leq i < M$. If no additional information regarding the temporal segment boundaries is provided to the watermark detector, it will assume that a_i and a_{i+1} are a single symbol. To avoid this problem, we encode the position of the symbol within the string along with the symbol itself, i.e., the watermark signal has to carry an information of $\log_2 M + \log_2 n$ bits within each temporal segment.

Let us now discuss how to use JAWS for our purposes. The basic JAWS watermarking technique [5] marks a video frame by adding a "secret" noise pattern to it. To detect the presence of the mark, each frame is correlated with this pattern. More precisely, let us first assume that the picture size is $N \times N$. We denote the luminance value of a picture at the pixel position i, j by $I(i, j)$ and the value of the noise pattern at the same position by $W(i, j)$. The watermarked image is obtained as

$$I_{WM}(i, j) = I(i, j) + s \cdot \Lambda(i, j) \cdot W(i, j), \quad (1)$$

where s is the watermark embedding strength and Λ is a scaling matrix, which scales the values of W according to the visibility of the pattern at the given pixel position. During the detection, I_{WM} is correlated with W. If the peak correlation value exceeds a certain threshold, the picture is considered to be marked. Detection is impossible without knowing W. Finally, in order to handle arbitrary picture sizes, the picture is tiled with the watermark pattern of a fixed size N.

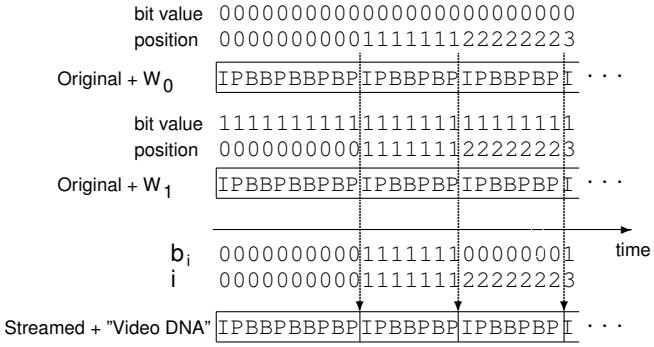

Figure 2: VideoDNA with JAWS for $n = 2$ copies. W_0 and W_1 denote the noise patterns according to JAWS that carry the indicated bit value and bit position information.

The above method embeds a single bit of information in each picture. To embed more than one bit, the authors of [5] propose to take the following modified noise pattern instead of the original noise pattern W:

$$W_{\sigma,k,l}(i, j) = W(i, j) + \sigma \cdot W(i+k, j+l),$$

where $\sigma \in \{-1, +1\}$ is a sign variable, and k and l are vertical and horizontal cyclical shift parameters (the addition is "modulo N"). The detection is done by correlating the picture with the noise pattern W. If the watermark is present, the correlation will result in two peak values. It is possible to retrieve information on the shift parameters by considering the relative distance of the two peaks, and it is possible to retrieve σ by comparing their signs. In this way, the watermark $W_{\sigma,k,l}$ can carry several bits of information, which is discussed in [7].

In our implementation of the VideoDNA method, we used $n = 2$ copies of the video, i.e., the user or session specific information is represented by a binary string $b_1 b_2 \ldots b_M$. We use the sign parameter σ of the pattern $W_{\sigma,k,l}$ to embed the bit values into the frames of the corresponding temporal segments. To encode the position i of the symbol within the binary information string, we used the shift parameters k and l. In our test configuration, the positions are encoded with 9 bits, so we can embed binary strings of maximal length equal to $M = 512$. We used a tile size $N = 128$ in our tests, which we found to be a good choice for standard definition PAL sequences.

The VideoDNA implementation with JAWS is depicted in Figure 2. Within each temporal segment, the bit value and the bit position are embedded into the pictures of the segment using the JAWS method. Finally, the whole sequence is compressed. The video streams in the figure are shown in compressed format reflecting the GOP structure. Note that each temporal segment starts with a key frame.

The detection is done in two steps. First, the standard JAWS detection procedure is applied for each video frame separately to decode the corresponding bit value and position. If the decoding fails, an erasure is reported. This is followed by a majority decision: for the frames with the same decoded position information, the original bit value is inferred to be 1 if the majority of the decoded values is 1, otherwise it is inferred to be 0.

At low compression bit rates, a single video frame is likely to result in an erasure. This can be avoided by defining a temporal sliding window over the video frames and by summing the intensity values within the window to strengthen the watermark signal

before applying the JAWS detector [5]. Nevertheless, erasures will still happen at the boundaries of the temporal segments due to the superimposition of different watermark signals.

3.2 Transparent stream switching

There are two main issues to make the stream switching transparent for the streaming client. The first is to make sure to create a standard compliant video elementary stream. The second is to assure that there are no problems at the transport and systems level. The first point can be solved easily as follows. The n watermarked copies of the video are compressed under the constraint that each temporal segment terminates with a closed GOP. The closed GOP requirement is necessary to keep the decoding consistent and to avoid prediction mismatch at the border of the segments.

The transport and systems level imposes a problem on our approach which is not obvious at first sight. To explain this, we assume in the following that the video streams are sent over the network with a constant bit rate R, which is often the case in high quality and highly scalable VoD-systems. Because of the variable bit rate nature of video, a buffer model has to be applied to shape the traffic. In case of H.264 this is done by the so called hypothetical reference decoder (HRD) [2] which makes sure that the buffer at the streaming client does neither overflow nor underflow.

However, the HRD model can be broken if the n streams are encoded independently. To understand the consequences of the independent encoding, note that the encoded length of the corresponding temporal segments will be slightly different for each stream, since they are marked with different watermarks. Let us now consider the following two extreme cases. In the first extreme, we can construct a user or session ID such that VideoDNA will always switch to the temporal segment with the shortest encoded length. Obviously, the resulting video has a smaller average bit rate than the streaming bit rate R. The video frames will be arriving faster than specified by the HRD, which leads to overflows in the receiver buffer. In the other extreme, we can construct a message that makes VideoDNA switch always to the temporal segment with the largest encoded length. The resulting video has a higher average bit rate than R, which leads to underflows at the receiver, since the frames of the video will be arriving slower than specified by the HRD model.

We solve the above problem by jointly encoding the n watermarked streams. The n encoders encoding the streams communicate with each other and exchange rate control information. This is done to make sure that the corresponding temporal segments have the same encoded lengths, and that the HRD models are synchronized at the segment borders. Using this method, the streaming bit rate can always be taken to be equal to R, no underflows or overflows will occur at the receiver. The stream switching will be transparent to the system and the required transmission bandwidth does not increase due to the watermarking.

4. RESULTS

The goal of this section is to present tests on how to use JAWS within VideoDNA. The aim was not to test JAWS itself, but to see how to configure VideoDNA to achieve the best detection. We rather discuss the strengths and weaknesses of VideoDNA regarding watermark removal and collusion attacks.

The JAWS method has been implemented according to [5] and [6]. Standard definition (PAL, 720x576 pixels, 25 frames per second) test sequences have been used, which were compressed with MPEG-4 AVC/H.264. Currently, an acceptable compression quality for full resolution PAL sequences is achieved between 1000 and 2500 kbits/s peak bit rate, depending on the content type and the quality expectation, using state-of-the-are H.264 encoders. This was also our target scenario. We have modified the open source software x264 [8] to randomly select appropriate switching points and to perform the joint rate control for the watermarked streams.

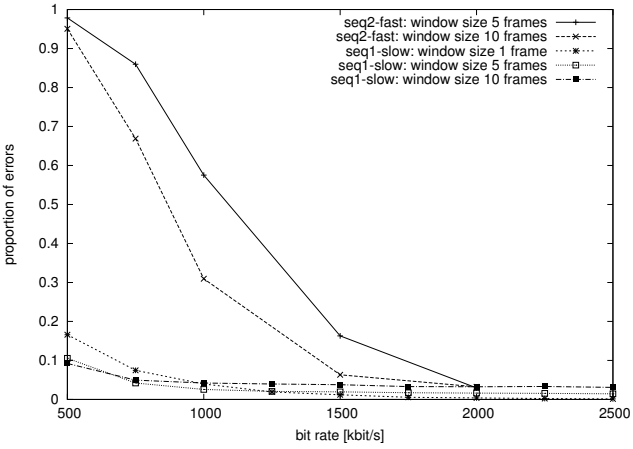

Figure 3: The performance of the JAWS detector with different sliding window sizes. The embedding strength was $s = 350 \cdot s_0$. The length of the temporal segments was randomly chosen between 50 and 250 frames.

4.1 Tuning the parameters of JAWS

The size of the noise pattern W was 128×128 pixels in all our experiments. The other free parameter is the embedding strength s. The authors of [5] give a method to compute the lower bound s_0 for this factor for each picture to remain under the specified false positive detection rate. However, we found it useful to modify the proposed value with a user defined multiplication factor, the choice of which will be discussed later.

We measure the performance of the JAWS watermark detector by computing the following error rate value:

$$\text{Proportion of errors} = \frac{\text{erasures} + \text{wrong detections}}{\text{total number of frames}}.$$

The detection result are shown for two 5 minutes long test sequences representing typical movie content, the first one ("seq1-slow") consisting of mostly static or slow scenes, the second one ("seq2-fast") being an especially difficult scene with lots of motion. Figure 3 shows the proportion of errors for different bit rates when using sliding window method (see Section 3.1). For window sizes greater that one, we will always have detection errors at the boundaries of the temporal segments even though there are no wrong detections. The corresponding curves will therefore not tend to zero as it is the case for window size one. However, with larger window size, the number of detection errors remains low even at low bit rates.

Regarding the choice of the multiplication factor, we consider the results from Figure 4. These show that the useful user defined modification factors lie between 100 and 350. In this range, the number of detection errors remain acceptable, and the artifacts introduced by the watermark remain visually indistinguishable from the usual encoding artifacts.

4.2 Tuning the parameters of VideoDNA

After finding the appropriate detection parameters for JAWS, we can test the reliability of the majority decision procedure. The number of errors in the user or session ID as a binary string is expressed as the corresponding bit error rate:

$$\text{BER} = \frac{\text{number of wrong bits}}{\text{total number of bits}}.$$

The tests were done for "seq1-slow", "seq2-fast", and in addition for "seq3-mixed", a 10 minutes scene with both low and high activity parts.

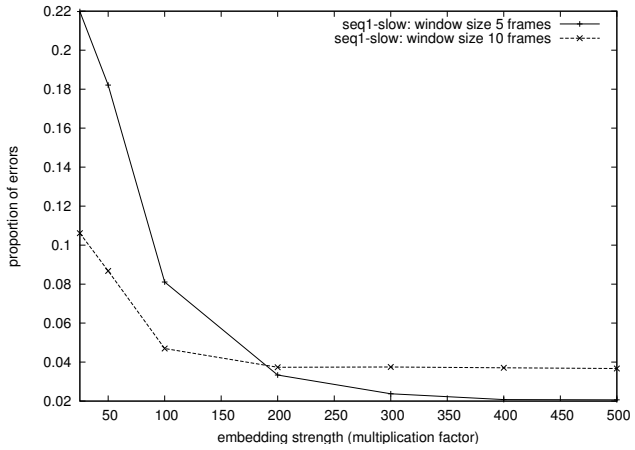

Figure 4: JAWS performance versus embedding strength ($s =$ "multiplication factor" $\cdot s_0$) at 1000 kbit/s.

The results for "seq1-slow" in Figure 5a show that although JAWS results in some errors (compare Figure 3), the overall BER for VideoDNA after the majority decision reconstruction remains zero, i.e., we can reconstruct the user or session ID perfectly even at relatively low bit rates.

The sequence "seq2-fast" is a more difficult case, the proportion of errors for the JAWS decoding was large at low bit rates, the BER is larger accordingly. However, by increasing the sliding window size for the JAWS detector, the BER will be reduced. In fact, by choosing an appropriately large sliding window, the BER can be largely reduced: this is shown in Figure 5b for "seq3-mixed". Of course, the lengths of the temporal segments have to be larger as well, because of the boundary effects. Indeed, as long as the proportion of errors is smaller than 0.5, we can always chose a large enough temporal segment size to arbitrarily reduce the BER. This is illustrated in Figure 5c for "seq2-fast".

4.3 Discussion

The result of the above tests show how to reduce the amount of decoding errors: If the JAWS detector had a large proportion of errors, we could reduce this by taking a larger sliding window. Similarly, due to the majority decision procedure, if the JAWS detector had a proportion of errors that was under 0.5 BER, we could increase the temporal segment size in order to reduce the BER of the VideoDNA. For typical and very difficult standard definition movie content, we could achieve zero or very low BER when using a JAWS detection window of 10 frames and temporal segment length between 5 and 10 seconds. This enabled us to embed 180 to 360 bits into 30 minutes of video content.

Now, we address the robustness of VideoDNA against typical attacks aiming to destroy the watermark. Regarding removal attacks, we have mentioned that VideoDNA profits from the robustness properties of the underlying watermark embedding method. JAWS is for example resistant to picture shifts, cropping, scaling, and aspect ratio changes [5, 7]. In our context, a removal attack manifests itself in the increase of the proportion of errors in the JAWS decoding. We can therefore apply the two countermeasures mentioned above to increase the robustness of the detection.

Collusion attacks constitute a more serious problem for VideoDNA. There are several possibilities for an attacker:

Intra-video collusion: The attacker combines the frames of a single video sequence to estimate the watermark with statistical methods. This attack is successful if the same watermark signal is embedded in a video scene with lots of motion, since the watermark can be easily estimated by averaging the frames. In the VideoDNA implementation, the temporal segments are chosen adaptively for

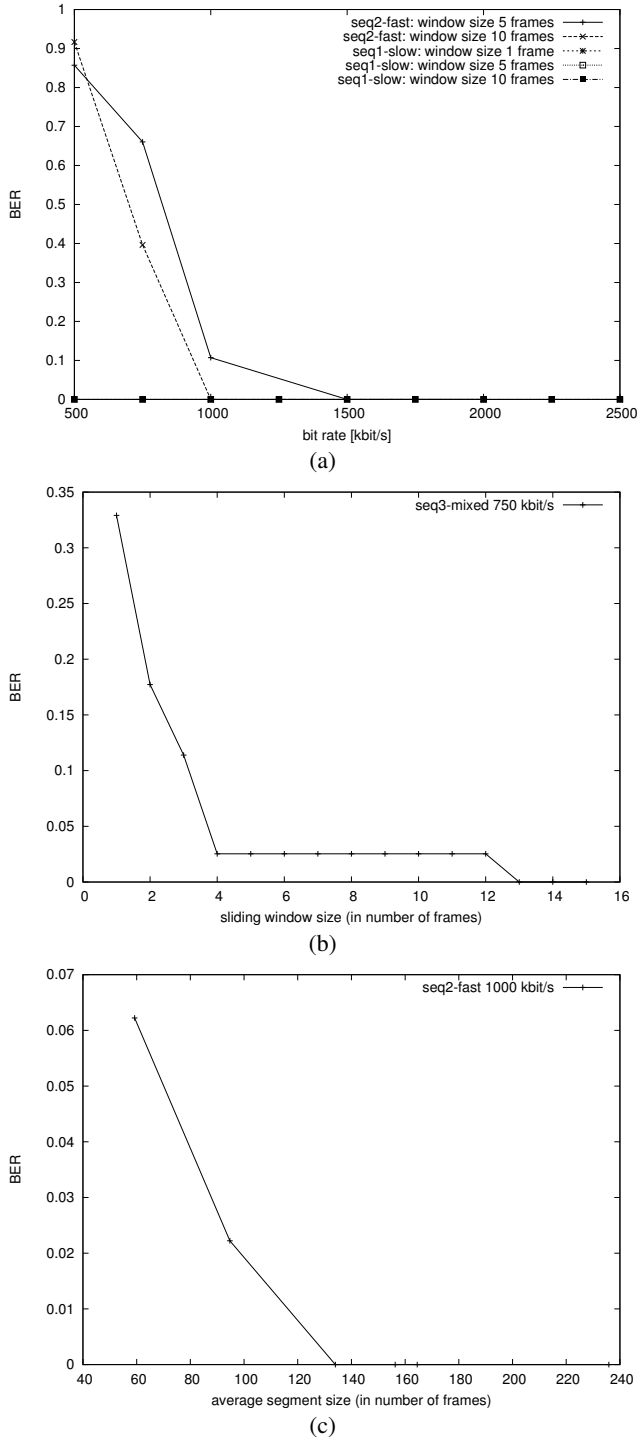

Figure 5: BER versus a.) compression bit rate, b.) detection window size, c.) average temporal segment size (for a detection window of 10 frames)

sake of the compression efficiency. The temporal segments start mostly at scene change points, which avoids spreading the same watermark signal over a highly changing portion of the video.

Frame-by-frame inter-video collusion: Two or more attackers combine their versions of the same video. The videos are combined frame-by-frame by averaging, by mosaicing, or by any other method. Since this is a very difficult situation, it is better to prevent

it in the following way [9, 10]: The n parallel streams are slightly modified before watermarking each one being distorted in a slightly different but non-disturbing way. This can be done by randomly warping the frames, or by a random temporal resampling of the video. It will be very difficult for the attackers to obtain a colluded video with acceptable quality.

Inter-video collusion by interleaving: The attackers combine segments of their video files together to obtain an uninterpretable session ID. This kind of attack is counteracted by the fact that the segment borders are chosen randomly and that the watermark contains the bit position along with the bit value. Provided that the number of attackers is small, the user or session ID can be at least partially reconstructed, by using error correction codes. However, further research is needed to understand how far is this possible. Note that any watermarking method (e.g., [3]) based on the temporal variation of watermarks will suffer from the same problem.

5. CONCLUSION

We have introduced a new very fast session based watermarking method for VoD systems called "VideoDNA". VideoDNA embeds the user or session ID into the video on-the-fly during the streaming session by switching between several watermarked copies of the same video file. We have shown that the VideoDNA approach is scalable and that it can be transparently integrated in a VoD system by considering several implementation issues.

We have considered a prototype using the watermarking scheme JAWS for our proof of concept: We have given a method for embedding a binary user or session ID string into the streamed video by using the JAWS watermark embedder. We have established rules for choosing the embedding parameters to ensure that the used ID is recoverable.

There is still a lot of room for improvement. The idea of stream-switching can be combined with any watermarking method other than JAWS capable of embedding several bits of information within short temporal segments. The embedding and decoding of the user or session ID may be improved by using error correction codes and soft decision decoding procedures.

Finally, the VideoDNA approach is not restricted to the VoD scenario. The stream switching method can also be used with any multimedia or appropriately organized data, for example in a download server.

REFERENCES

[1] G. Doerr, J.-L. Dugelay *A Guide of Video Watermarking* Signal Processing: Image Communications 18, pp. 263 - 282, 2003

[2] ITU-T Recommendation H.264, also ISO/IEC 14496-10, 2005

[3] R. Parviainen, P. Parnes. *Large scale distributed watermarking of multicast media through encryption.* In: Proceedings of the CMS 2001, Darmstadt, Germany, May 2001

[4] H. Jin, J. Lotspiech, S. Nusser *Traitor tracing for prerecorded and recordable media*. In: Proc. of the 4th ACM workshop on Digital rights management, pp. 83 - 90, 2004

[5] M. Maes, T. Kalker, J.-P. Linnartz, J. Talstra, G. Depovere, J. Haitsma *Digital Watermarking for DVD Video Copy Protection* IEEE Signal Processing Magazine, September 2000

[6] M. Maes, T. Kalker, J. Haitsma, G. Depovere *Exploiting Shift Invariance to Obtain a High Payload in Digital Image Watermarking* ICMCS, Vol. 1, pp. 7 - 12, 1999

[7] P. Termont, L. De Strycker, J. Vandewege, M. Op de Beeck, J. Haitsma, T. Kalker, M. Maes, G. Depovere *How to Achieve Robustness Against Scaling in a Real-time Digital Watermarking System for Broadcast Monitoring* ICIP 2000

[8] http://developers.videolan.org/x264.html

[9] M. U. Celik, G. Sharma, A. M. Tekalp *Cullusion-resilient fingerprinting using random pre-warping.* In: Proc. IEEE Int. Conf. on Image Processing (ICIP), pp. 509 - 512, 2003

[10] Y. Mao, M. K. Mihcak *Collusion-resistant intentional desynchronization for digital video fingerprinting.* In: Proc. IEEE Int. Conf. on Image Processing (ICIP), 2005