# Evaluation of an Optimal Watermark Tampering Attack Against Dirty Paper Trellis Schemes

Patrick Bas, Gwenaël Doërr

# Evaluation of an Optimal Watermark Tampering Attack Against Dirty Paper Trellis Schemes

Patrick Bas
Gipsa-lab CNRS
961, Rue de la Houille Blanche BP 46
38042 St. Martin d'Hères Cédex, France
Patrick.Bas@inpg.fr

Gwenaël Doërr
University College London
UCL Adastral Park – Ross Building 2
Martlesham IP5 3RE, United Kingdom
g.doerr@adastral.ucl.ac.uk

## ABSTRACT

Benchmarking watermarking systems now goes beyond only evaluating the ability of the embedded watermark to withstand common signal primitives such as filtering, resampling, lossy compression, D/A-A/D conversions, etc. Evaluation procedures have to consider how much information leaks from a watermarking system since such knowledge could prove most helpful to design very powerful attacks. This paper further refines an attack on dirty paper watermarking schemes which relies on security weaknesses i.e. information leakage. In particular, additional constraints are introduced to be able to handle 'complex' trellises. Moreover, the efficiency of this attack has been evaluated for different trellis configurations. Quite counter-intuitively, increasing the number of states in the trellis seems to enhance both the robustness and the security of the system.

## Categories and Subject Descriptors

H.4.m [**Information Systems Applications**]: Miscellaneous—*Watermarking*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance Evaluation (efficiency and effectiveness)*; K.6.m [**Management of Computing and Information Systems**]: Miscellaneous—*Security*

## General Terms

Security, Algorithms

## Keywords

Dirty paper trellis watermarking , security, Attack.

## 1. INTRODUCTION

Watermarking security is defined today as the possibility to estimate all or a part of the secret parameters of a watermarking system [1]. In contrast with robustness attacks, security attacks inherently involve two steps: (i) the adversary devise some strategy to estimate part or all the secret of the attacked watermarking system and (ii) the adversary exploit this acquired knowledge to perform some action. The first step of any security attack is commonly categorized depending on the kind of material available to the adversary e.g. Known Message Attack (KMA), Constant Message Attack (CMA), Known Original Attack (KOA) and Watermarked content Only Attack (WOA) setups [1, 2]. Additionally the classification of security attacks can be further refined by incorporating the objective of the adversary during the second step. Depending on the acquired knowledge in the first step, attacks which goes beyond the usual `erase-only` might be possible.

A first effort in that direction has been reported in [3] where the authors proposed three different attack objectives: watermark removal, unauthorized watermark estimation and/or detection, and unauthorized watermark writing which encompasses the copy attack [4]. However, this early classification fails to address a couple of issues. First, the denomination "watermark removal" is well fit for zero-bit watermarking schemes [5] but possibly inappropriate for multi-bit schemes. Indeed, in that case, it is not always necessary to erase the watermark to defeat the system. Forcing the detector to output a message which is different from the embedded one might prove critical in some application. Second, the classification in [3] does not reflect the fact that the knowledge acquired in the first step of the attack will be used if necessary to tailor modifications which achieve the desired objectives while preserving at most the fidelity of the attacked content.

Based on these observations, a new terminology is adopted to describe more accurately the potential objectives of an adversary:

- *Unauthorized watermark copying.* From one or more contents watermarked with the same message, the goal is to design a procedure which copies with high probability the watermark into a 'blank' content. Depending on the number of available watermarked contents, preserving fidelity may be an issue.
- *Unauthorized watermark detection/reading.* The objective is to detect whether a given content contains a watermark or not, and, in the case of a multi-bit scheme, what is the embedded message.
- *Unauthorized watermark tampering.* The objective is to modify the watermarked content while preserving fidelity so that the detector no longer outputs the embedded message with high probability. This includes for instance the Oracle attack [6, 7] when the detector is available.


*MM&Sec'08,* September 22–23, 2008, Oxford, United Kingdom.

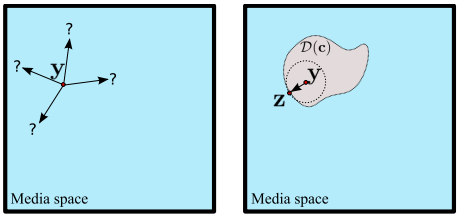


**Figure 1: Geometrical comparison of a blind and an optimal watermark tampering attack. In the blind case, the adversary goes in any random direction since she has no knowledge about the secret, here the decoding region $\mathcal{D}(\mathbf{c})$. On the other hand, relying on some knowledge acquired during a security analysis, the optimal attack finds the shortest way outside the decoding region where is the watermarked content y.**

- *Unauthorized watermark writing.* The objective it to modify a content so that the detector outputs a given message with high probability. In contrast with a copy attack, it does not require that watermarked contents with this specific message embedded are available during the first phase. Ideally, the distortion induced by a this attack should be of the same order than if the message was embedded with the official embedder.

Each objective involves a different knowledge of the secret parameters of the watermarking system to be performed e.g. the feature space being watermarked, the detection regions of the used set of codewords, the message encoded by each codeword, etc.

This paper focuses on devising an optimal watermark tampering attack against Dirty Paper Trellis (DPT) watermarking schemes in the WOA scenario. As illustrated in Figure 1, in contrast with a blind attack, such optimal tampering attacks aims at identifying the most damaging direction for modification i.e. the direction which indicates the shortest way outside the decoding region where the watermarked content lies. Such knowledge can be acquired because DPT watermarking schemes have been proven to leak information.

## 2. PREVIOUS WORKS

### 2.1 Dirty Paper Trellis Watermarking

Dirty Paper Trellis codes [8] have been introduced as an alternative implementation of digital watermarking when it is modeled as communications with side information [9]. In comparison with the popular lattice codes, DPT codes are significantly more robust against requantization (e.g. lossy compression) and by construction immune to valuemetric scaling (e.g. contrast enhancement for an image, volume change for a song). DPT watermarking schemes mostly rely on two fundamental components: a high-dimensional trellis-structured dirty paper code and a computationally intensive iterative embedding procedure.

DPTs rely on a small modification to conventional binary trellises design in order to provide a one-to-many mapping between messages and codewords. A trellis is basically a

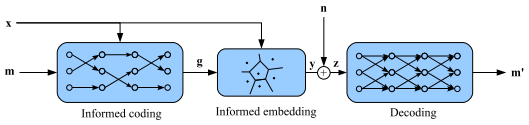


**Figure 2: Main principles of DPT watermarking. In this example $N_b = 3$, $N_s = 3$, $N_a = 2$. Three alternative codewords are available in the expurgated trellis to represent the message m. The codeword g with the highest correlation with x is identified using the Viterbi algorithm. Afterward, the watermarked vector y is computed taking into account g. On the receiver side, the detector uses the whole DPT to retrieve the embedded payload.**

graph which can be fully described with its number of *states* $N_b$, its number of *arcs per state* $N_a$, and its number of *steps* $N_s$. For each step, the arcs defines how the incoming states are connected to outgoing states. In a conventional binary trellis, there are only two arcs entering/reaching each state, one encoding 1 and the other 0. Therefore, there is a one-to one mapping between messages and codewords. In order to obtain the desired dirty paper property, a DPT allows several arcs entering/reaching each state, half of them encoding 1 and the other half encoding 0. As a result, multiple paths through a DPT encode the same message. In addition to a binary label, a symbol $\mathbf{s}_a$, taken from an alphabet $\mathcal{A}$ consisting of $N_a$ pseudo-randomly generated $N_v$-dimensional unit norm symbols (also referred to as patterns), is attached to each arc of the trellis. Thus, each path through the trellis can be seen as a message or, alternatively, as a succession of symbols emitted by the DPT communications system. Concatenating these symbols altogether defines one of the DPT codewords, and the set of all possible codewords defined by a DPT trellis results in a spherical dirty paper code.

As depicted in Figure 2, the first step of a DPT watermarking scheme is referred to as *informed coding*. It consists in finding the codeword $\mathbf{g}$ which (i) encodes the desired message $m$ and (ii) is the closest to the original cover content $\mathbf{x}$. To do so, the DPT is first simplified by eliminating all the arcs which do not encode the desired message. Therefore, any path through this simplified DPT encodes the desired message and informed coding is only a matter of finding which of them is the closest to the cover. In other words, it is necessary to compute some *distance* between the cover and each one of the codewords, and to retain the one with the smallest distance. Hopefully, thanks to the trellis structure of the code, this search can be performed efficiently using the Viterbi decoding algorithm[10].

The next step of DPT watermarking is *informed embedding* which is tightly related to the detection procedure applied on the receiver side. Indeed the goal of this step is to bring the cover content inside the decoding region $\mathcal{D}(\mathbf{g})$ of the identified codeword $\mathbf{g}$, possibly with some level of robustness to be able to withstand distortions onto the watermarked content during transmission. This decoding region is fully defined by the detection procedure which is a basic nearest codeword search performed once again using the Viterbi decoding algorithm. However, since the receiver does not know the hidden message $\mathbf{m}$, it uses the full non-simplified trellis. Once the best path through the trellis

is identified, the hidden message can easily be retrieved by looking at the bits encoded by the arcs along that path. Unfortunately, 'bringing the content inside the desired decoding region' is a complex problem which cannot be easily solved. Today, informed embedding relies on an iterative embedding algorithm [8], which basically simulates the communication channel for some specified noise level and terminates when there is no longer any decoding error.

## 2.2 Security Analysis and Tampering Attack

Although DPT watermarking schemes have resisted for quite a long time to attacks from hostile adversaries, a recent study [11] has reported that it is feasible to acquire valuable secret information from such systems through security analysis, which then allows to perform powerful watermark tampering attacks. The secret parameters of a DPT consist of:

1. the pseudo-randomly generated alphabet $\mathcal{A} = \{\mathbf{s}_a\}$ containing the symbols $\{\mathbf{s}_a\}$ attached to the arcs,
2. the connectivity of the trellis i.e. the incoming and outgoing states for each arc (which can be assimilated to a symbol),
3. the binary labels attached to the arcs.

The mentioned study has revealed that, in a WOA setup, it is possible to estimate the secret alphabet and, subsequently, the connectivity of the trellis. This gained knowledge about the secret parameters of the trellis can then be used as a keystone to design an optimized distortion-preserving watermark tampering attack. It could be noted that the estimation of the third secret parameter would require to be in a KMA setup and would open avenues to perform unauthorized read/write attacks.

The estimation of the secret alphabet $\mathcal{A}$ is made possible by the fact that DPT watermarking schemes tend to cluster watermarked contents around the different codewords in order to guarantee some level of robustness. Due to the trellis structure of the code, the same observation is also valid to some extent at the step level. In other word, the segments of watermarked content, a segment of content corresponding to one step in the trellis, tends to be clustered in the direction of the different symbols of the alphabet. Moreover, the stronger the specified robustness constraint is, the more valid this observation is. As a result, it is possible to estimate the collection of $N_s.N_a$ symbols $\mathcal{A} = \{\mathbf{s}_a\}$ by applying clustering techniques, such as the K-means algorithm [12], to a large set $\mathcal{Y}$ of segments of watermarked contents[1].

The estimation of the alphabet permits to infer the connectivity of the trellis by looking at the sequences of emitted signals, and more particularly at couple of successive symbols. A state in the trellis always outputs the same subset of symbols. Consequently, when two symbols are followed by the same subset of symbols, it clearly indicates that the emission of these two symbols lead to the same state in the trellis. The identification of such similarities allows to estimate the outgoing state of each symbol. Moreover, since the outgoing state of the first symbol is also the incoming state of the second symbol, it is then possible to also assign

[1]Due to the curse of dimensionality, the clustering process is more efficient when working on segments of contents than directly on the contents themselves [11].

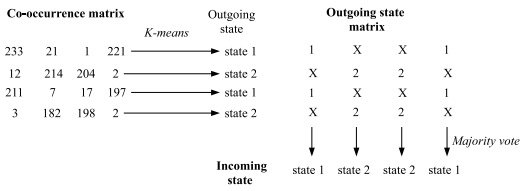


**Figure 3: Trellis connectivity estimation. Similar rows in the co-occurrence matrix indicate that the same state has been reached after the emission of the symbols associated to these rows. Such similarities can be automatically identified with clustering techniques, e.g. the K-means algorithm. Once the outgoing state has been identified for each symbols, it is then possible to build an outgoing state matrix. A majority vote along the columns of this matrix then indicate the incoming state for the symbols associated to the columns.**

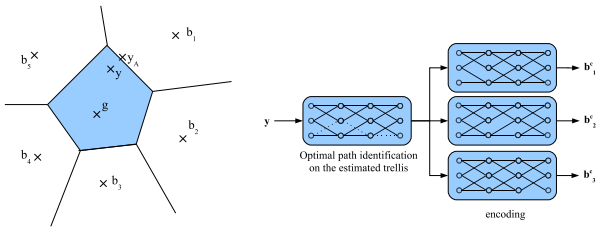


**Figure 4: DPT watermark tampering attack. To be optimal with respect to distortion, the adversary needs to find the closest point to y outside the decoding region $\mathcal{D}(\mathbf{g})$ of the embedded codeword g. To do so, she needs to identify the second nearest codeword from y ($\mathbf{b}_1$ in the Figure). To do so, the adversary simply needs to run $N_b$ Viterbi decoders in parallel, a single arc of the best path g being forbidden each time. This gives $N_b$ eligible candidates and the one which is the closest to y is retained.**

an incoming state for all the symbols in the alphabet. As depicted in Figure 3, [11] relies on clustering techniques applied to the rows of a co-occurrence matrix to estimate the outgoing states, and subsequently performs a majority vote to identify the incoming state of each arc.

Once the adversary has access to this secret information, she can easily find the codeword $\mathbf{g}$ embedded in a watermarked content by running the Viterbi decoder with the estimated trellis and the content itself. In the context of a tampering attack, the goal is then to find the shortest path outside this detection region. This requires finding the boundary of the decoding cell which is the closest to the watermarked content. Figure 4 clearly shows that this can be approximated by finding the second nearest codeword from $\mathbf{y}$. This can be done by running $N_b$ Viterbi decoders in parallel, each one using a slightly different trellis. The idea is to remove a single arc of $\mathbf{g}$ in the original DPT trellis to identify a codeword which is close enough to $\mathbf{y}$ to be a potential candidate. Out of the $N_b$ eligible codewords, the one which is the closest to $\mathbf{y}$ is retained as the second nearest codeword $\mathbf{b}_1$. Subsequently, running the embedding procedure for this codeword with minimal robustness constraint permits to leave the original decoding region while main-

taining fidelity. Note that by selecting the closest codeword to the one used during the embedding, the adversary is able to devise an attack which is close to optimal. Nevertheless since the watermarked content in not located exactly on a codeword, the direction toward the closest codeword may not be the direction toward the closest boundary hence the optimality is not guarantied in this case. However in practice the trellis produces distances between closest codewords that are of the same order of magnitude and the attack is very close to optimal.

# 3. EXTENSION TO REAL-LIFE DPT CONFIGURATIONS

Although the experimental results reported in [11] clearly demonstrated the proof of concept of this attack, they were mostly considering toy example trellis configurations which would probably never be used in practice. In particular, the alphabet $\mathcal{A}$ was designed using only orthogonal or opposite symbols, thus limiting the cardinality of the alphabet to $2.N_v$. Moreover, the study was limited to a single trellis configuration, namely a trellis using $N_v = 12$-dimensional symbols, $N_s = 6$ states and $N_a = 4$ arcs per state.

In practice, the alphabet is usually significantly larger and the symbols, which are independently pseudo-randomly generated [8], exhibit some cross-correlation. Additionally, several alternative trellis configurations would be investigated since it has been shown that robustness performances can be significantly be affected depending on which trellis configuration is used [13]. To deal with such real-life DPT configurations it is necessary both to define a new metric to assess the quality of the alphabet estimation and to improve the trellis connectivity estimation procedure to be able to handle discrepancies appearing when the complexity of the trellis is increased.

## 3.1 Alphabet Estimation Assessment

Although an adversary will never have access to the secret parameters of the watermarking system in practical cases, it is commonly admitted that, during the *design phase* of the attack, these parameters are available in order to be able to assess the quality of the estimation performed by the adversary. For instance, in the proposed attack, it is important to evaluate how close the estimated alphabet $\hat{\mathcal{A}} = \{\hat{\mathbf{s}}_a\}$ output by the K-means procedure is from the actual secret one $\mathcal{A}$. This is of utmost importance since the follow-up estimation of the trellis connectivity will rely on the estimation of the secret alphabet. Therefore, the quality of this alphabet will have a tremendous impact on the efficiency of the attack.

In [11], the following metric has been proposed:

$$\Delta = \frac{1}{N_s N_a} \sum_{\hat{\mathbf{s}}_a \in \hat{\mathcal{A}}} \left[ \max_{\mathbf{s}_a \in \mathcal{A}}{}_1 \, \mathrm{nc}(\hat{\mathbf{s}}_a, \mathbf{s}_a) - \max_{\mathbf{s}_a \in \mathcal{A}}{}_2 \, \mathrm{nc}(\hat{\mathbf{s}}_a, \mathbf{s}_a) \right] \quad (1)$$

where the $\max_1$ and $\max_2$ respectively output the highest and second highest values of some set, and $\mathrm{nc}(.,.)$ denotes the normalized correlation operator. The score $\Delta$ basically indicates how close, in average, each estimated symbol $\hat{\mathbf{s}}_a$ is from any of the original one $\mathbf{s}_a$. If a symbol given by the K-means algorithm is not a good estimate, the $\max_1$ and $\max_2$ values will be similar and their difference close to 0. On the other hand, in case of a good estimate, the $\max_1$ value will be close to 1 and the $\max_2$ value close to 0 *since the symbols in $\mathcal{A}$ are opposite or orthogonal* in [11], thus leading to a difference close to 1. In summary, this metric returns a value close to 0 when the estimation of the secret alphabet is very poor and a value close to 1 in case of a nearly perfect match.

Now, in the case of a generic randomly generated alphabet, the situation is slightly different. In the case of a good estimation, the second term in the difference would no longer be close to 0 since symbols are not orthogonal, and subsequently, the resulting difference would not be close to 1. To address this issue, the metric is modified as follows:

$$\Delta = \frac{1}{N_s N_a} \sum_{\hat{\mathbf{s}}_a \in \hat{\mathcal{A}}} \frac{\max_{\mathbf{s}_a \in \mathcal{A}} \mathrm{nc}(\hat{\mathbf{s}}_a, \mathbf{s}_a) - \max_{\mathbf{r}_a \in \mathcal{R}} \mathrm{nc}(\hat{\mathbf{s}}_a, \mathbf{r}_a)}{1 - \max_{\mathbf{r}_a \in \mathcal{R}} \mathrm{nc}(\hat{\mathbf{s}}_a, \mathbf{r}_a)} \quad (2)$$

where $\mathcal{R}$ is a randomly generated alphabet. In other words, the modified metric measures the difference between (i) the distance between the estimated alphabet and the secret one and (ii) the distance between the estimated alphabet and a random one. The denominator is simply used as a normalization factor to guarantee that the metric varies between 0 and 1. It is immediate to check that, in the case of a bad estimation, this metric will have a value close to 0. Conversely, for a good estimation, the metric value will be close to 1.

## 3.2 Improved Trellis Connectivity Estimation

As described in Subsection 2.2, the estimation of the trellis connectivity basically relies on processing the co-occurrence matrix $\mathbf{C}(i,j)$ defined as follows:

$$\mathbf{C}(i,j) = \mathrm{occ}(\mathbf{o}_s \in \mathcal{C}\updownarrow_i, \mathbf{o}_{s+1} \in \mathcal{C}\updownarrow_j), 1 \leq i,j \leq N_s.N_a \quad (3)$$

where $\mathbf{o}_s$ is a segment of watermarked content observed at step $s$, $\mathcal{C}\updownarrow_i$ is the set of segments of watermarked content associated with the $i^{\text{th}}$ symbol in the estimated alphabet $\hat{\mathcal{A}}$, and $\mathrm{occ}(A,B)$ is an occurrence function that counts the number of times both $A$ and $B$ are true. The test $(\mathbf{o}_s \in \mathcal{C}\updownarrow_i)$ is performed using the classification results of the K-means algorithm used for the alphabet estimation. The estimation of the outgoing state of each symbol can then be determined by applying the K-means algorithm onto the rows of the co-occurrence matrix.

At this point, the original design of the attack relied on a majority vote along the columns of the outgoing state matrix to assign an incoming state to each symbol. Unfortunately, when the number of arcs $N_s.N_a$ is increased, empirical observations have revealed that this strategy could get confused e.g. some states could end up not being assigned as an incoming state for any symbol. The direct consequence is that the estimated trellis becomes erroneous thus leading to a less efficient tampering attack. To fix this problem, the estimation of the incoming states has been revisited along the following two axis:

- keep the voting weights when building the outgoing state matrix;
- inject some a priori information during the incoming state assignment e.g. each state should be assigned exactly $N_a$ symbols.

The first point is simply a matter of combining the information of the co-occurrence matrix $\mathbf{C}(i,j)$ together with the estimation of the outgoing state. More practically, looking at Figure 3 and reading the first column from top to bottom, the matrix is now changed to: 233 votes for state 1,

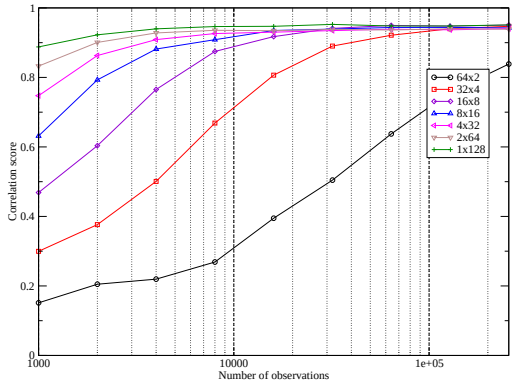


**Figure 5: Assessment of the quality of the alphabet estimation for different trellis configurations (represented by $N_s \times N_a$).**

12 votes for state 2, 211 votes for state 1 and 3 votes for state 2. Pooling the votes for this column, state 1 gets 444 votes and state 2 gets 15. States are then considered one after the other and the $N_a$ symbols with the highest voting score for this state are assigned this state as incoming state. The cumulated voting score for this state assignment is also recorded e.g., when state 1 is assigned to symbols 1 and 4 in Figure 3, the score $444 + 418 = 862$ is recorded. That state can then no longer be considered as a potential incoming state for any other symbol. The order in which states are scanned has a strong influence on the outcome of this process. This is why it is repeated times with different scanning order. the incoming state assignment which gives the highest average cumulated voting score is retained.

## 4. EXPERIMENTAL RESULTS

As mentioned earlier, the main objective of this paper is to verify if the proof of concept reported in [11] for a toy example trellis configuration can be extended to more realistic ones. To this end, we have considered an alphabet $\mathcal{A}$ composed of 128 $N_v$-dimensional symbols. This alphabet can be arranged into seven alternative trellises i.e. 1 (resp. 2, 4, 8, 16, 32, 64) state and 128 (resp. 64, 32, 16, 8, 4, 2) arcs per state. We also have generated a collection of 256.000 synthetic cover contents. Each cover is 120 samples long, each sample following a normal distribution. This implies that the trellises used for watermarking will have $120/12 = 10$ steps. Moreover, the robustness parameter used during the embedding procedure has been adjusted for each trellis configuration so that the resulting average Watermark to Content Ratio (WCR) is equal to -5 dB.

Figure 5 depicts how well the secret alphabet $\mathcal{A}$ is estimated using the metric defined in Equation 2 for different trellises configurations and for a varying number of observations. Keeping in mind that the K-means algorithm is performed using 12-samples long segments of watermarked content, it should be noted that each watermarked content provide 10 observations. The estimation of the alphabet is repeated 10 times and the resulting $\Delta$ scores averaged. First of all, it is easy to verify that, as expected, the quality of the estimation improves when more observations are considered. However, one can also notice that the configuration of the

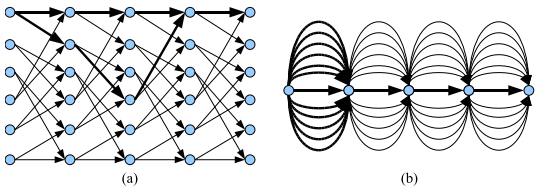


**Figure 6: Relationship between the memory of a trellis and its configuration. A $6 \times 2$ trellis has a 3-step long memory i.e. it takes a t least 3 steps to come back to the same state as a given path after deviating from it. In contrast, a $1 \times 12$ trellis has a one step long memory. Even if we deviate from the path at some point, the same state is reached after the step.**

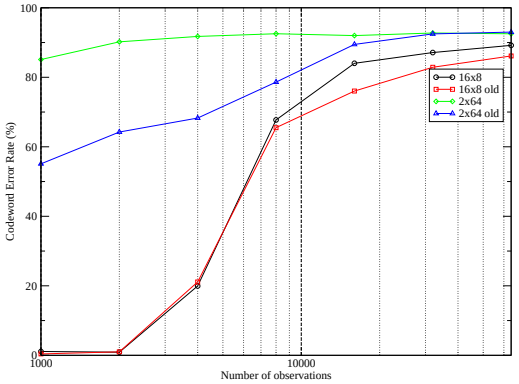


**Figure 7: Performances of the DPT watermark tampering attack for the two different trellis connectivity estimation procedures.**

trellis seems to have a strong impact on the quality of the estimation. With 10.000 observations, the estimation for the $64 \times 2$ trellis is quite poor whereas it is almost perfect for the $1 \times 128$ trellis. This could be quite surprising at first since, after all, the size of the secret is the same in all cases i.e. an alphabet of 128 symbols. Nevertheless, it should be kept in mind that the configuration of a trellis is directly related with its *memory* as illustrated in Figure 6. In a $1 \times 128$ trellis, all the arcs are parallel. As a result, for the embedding process to be successful, it needs to individually modify each segment of content so as to avoid any confusion with competing arcs. This results in well-clustered observations, thus resulting in a good estimation. On the other side of the spectrum, a $64 \times 2$ trellis has a 7-steps long memory. In other words, the embedding algorithm can distribute its distortion budget along 7 steps so as to avoid the confusion between two competing paths. This results in significantly less well separated observations, thus making the task of the K-means algorithm more difficult and leading to a poorer estimation of the alphabet.

Using the estimated alphabet, the trellis connectivity estimation process is run and the optimal DPT watermark tampering attack applied. The success of the attack is evaluated with respect to the Codeword Error Rate (CER) i.e. the probability of moving the watermarked content outside

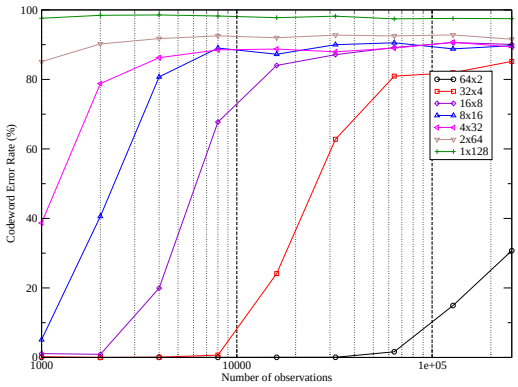


**Figure 8: Efficiency of the optimal watermark tampering attack for different trellis configurations (represented by $N_s \times N_a$).**

its decoding region[2]. Figure 7 depicts the performance gain achieved with the new incoming state assignment technique (cf. section 3.2) compared to the original one for two different trellis configurations. Two observations can be made:
- there is no noticeable gain when the alphabet estimation is either very poor or very good;
- the potential gain seems to decreases when the number of states increases, possibly because the increased sparsity of the co-occurrence matrix in that case helps the assignment process.

The interplay between these two aspects is not yet well understood and will require further investigations. Finally, the performances of the attack with the proposed incoming state assignment process are reported in Figure 8 for the different trellis configurations under investigation. It clearly highlights a strong correlation between the quality of the alphabet estimation and the efficiency of the attack. For two different trellis configuration, if the quality of the estimated alphabet as measured by the $\Delta$ metric is similar, then subsequent CERs are comparable. This tends to suggest that the number of states involved in the trellis connectivity estimation process only has a marginal impact on performances.

# 5. CONCLUSION AND PERSPECTIVES

The main contribution of this paper is to extend the security attack against DPT watermarking schemes presented in [11] to more realistic configurations e.g. using a larger alphabet of symbols which are not necessarily orthogonal or opposite. Moreover, the estimation of the trellis connectivity has been tested for various trellis configurations. The experimental results reported in Section 4 clearly indicate that DPT watermarking schemes leaks enough information about the secret parameters to allow adversaries to devise an effective watermark tampering attacks. In comparison with blind attacks, these attacks make the best use of the gained information to design a procedure which brings the watermarked content outside the decoding region where it lies while preserving fidelity as much as possible.

[2]It should be noted that the attack does not guarantee that the decoded message will be different from the embedded one. Still, empirically we observed that it is the case with high probability.

Next, this study provided further insight regarding the different parameters having an impact on the security level of DPT watermarking schemes, the security level being defined as the minimum number of observations required to perform a successful attack. It was already reported in [11] that the security level decreases when the embedding distortion increases. Indeed, increasing the distortion simply reduces to rejecting more host interference, hence producing more well separated observations. The clustering operation is subsequently more accurate and leads to a better attack. In this study, we have highlighted that the *memory* of the trellis has a strong impact on the performances of the proposed watermark tampering attack. For a fixed alphabet size, the memory of the trellis increases with the number of states. The reported experimental results clearly showed that trellis with larger memory were more difficult to estimate. This could be explained by the fact that the embedding algorithm has then more flexibility to distribute its distortion budget, hence producing less clearly isolated clusters.

# 6. ACKNOWLEDGEMENTS

Patrick Bas thanks the National French projects ACI-SI Fabriano, RIAM Estivale and ARA TSAR for their financial support. Gwenaël Doërr's research is supported in part by the Nuffield Fundation through the grant NAL/32707.

# 7. REFERENCES

[1] P. Comesaña, L. Pérez-Freire, and F. Pérez-González. Fundamentals of data hiding security and their application to spread-spectrum analysis. In *7th Information Hiding Workshop, IH05*, LNCS, Barcelona, Spain, June 2005. Springer Verlag.

[2] F. Cayre, C. Fontaine, and T. Furon. Watermarking security: theory and practice. *IEEE Trans. Signal Processing*, 53(10), oct 2005.

[3] T. Kalker. Considerations on watermarking security. In *Proc. of MMSP*, pages 201–206, Cannes, France, October 2001.

[4] M. Kutter, S. Voloshynovskiy, and A. Herrigel. Watermark copy attack. *EI'2000: Security and Watermarking of Multimedia Content II*, volume 3971, San Jose, California USA, 23–28 jan 2000.

[5] T. Furon. A constructive and unifying framework for zero-bit watermarking. *IEEE Trans. on Information Forensics and Security*, 2(2), June 2007.

[6] T. Kalker. A security risk for publicly available watermark detectors. In *Proc. Benelux Inform. Theory Symp.*, Veldhoven, The Netherlands, May 1998.

[7] P. Comesaña, L. Pérez Freire, and F. Pérez-González. Blind newton sensitivity attack. *IEE Proceedings on Information Security*, 153(3):115–125, September 2006.

[8] M. L. Miller, G. J. Doërr, and I. J. Cox. Applying informed coding and embedding to design a robust, high capacity watermark. *IEEE Trans. on Image Processing*, 6(13):791–807, 2004.

[9] M. H. M. Costa. Writing on dirty paper. *IEEE Transactions on Information Theory*, 29(3):439, 1983.

[10] A. J. Viterbi. *CDMA: Principles of Spread Spectrum Communication*. Addison-Wesley, pub-AW:adr, 1995.

[11] P. Bas and G. Doërr. Practical security analysis of dirty paper trellis watermarking. *Information Hiding: 9th international workshop*, LNCS, Saint-Malo, France, 2007. Springer Verlag, Berlin, Germany.

[12] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symp. on Mathematics Statistics and Probability*, 1967.

[13] C. Wang, G. Doërr, and I. J. Cox. Toward a better understanding of dirty paper trellis codes. In *IEEE Proc. Int. Conf. Acoust, Speech, Signal Processing*, volume 2, pages 233–236, 2006.

[14] C. Wang, G. Doërr, and I. J. Cox. Trellis coded modulation to improve dirty paper trellis watermarking. In *Proc. SPIE*, January 2007.