Forensic Analysis and Anonymisation of Printed Documents

Timo Richter*, Stephan Escher*, Dagmar Schönfeld, Thorsten Strufe
TU Dresden
Dresden, Germany
<firstname>.<lastname>@tu-dresden.de

ABSTRACT

Contrary to popular belief, the paperless office has not yet established itself. Printer forensics is therefore still an important field today to protect the reliability of printed documents or to track criminals. An important task of this is to identify the source device of a printed document. There are many forensic approaches that try to determine the source device automatically and with commercially available recording devices. However, it is difficult to find intrinsic signatures that are robust against a variety of influences of the printing process and at the same time can identify the specific source device. In most cases, the identification rate only reaches up to the printer model. For this reason we reviewed document colour tracking dots, an extrinsic signature embedded in nearly all modern colour laser printers. We developed a refined and generic extraction algorithm, found a new tracking dot pattern and decoded pattern information. Through out we propose to reuse document colour tracking dots, in combination with passive printer forensic methods. From privacy perspective we additional investigated anonymization approaches to defeat arbitrary tracking. Finally we propose our toolkit *deda* which implements the entire workflow of extracting, analysing and anonymisation of a tracking dot pattern.

KEYWORDS

Printer identification, Multimedia forensics, Digital Forensics, Laser Printer, Yellow Dots, Tracking Dots

ACM Reference Format:
Timo Richter*, Stephan Escher*, Dagmar Schönfeld, Thorsten Strufe. 2018. Forensic Analysis and Anonymisation of Printed Documents. In *Proceedings of 6th ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '18).* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3206004.3206019

1 INTRODUCTION

Still today, in our digitalised world, printed documents are used everywhere. Contracts, tickets, money, letters, invoices or analogue archives are just a small selection of examples. As a result printed documents are often an issue in crimes, like Fake IDs, copyright theft or as evidence in a criminal case. Hence, identifying the source printer of such documents is an important feature for evaluating their reliability or for tracking criminals. The research field of printer forensics provides solutions for this. Tools and algorithms developed in this area can basically be distinguished between active and passive methods [3].

Active forensic methods focuses on hidden information, called extrinsic signatures, that has been explicitly added to the document before or at the printing process. These information, e.g. the serial number of the printer device or a secure hash of the document, can then be used to identify the printer or to detect forgery. Examples in this field are the intentional adding of banding frequencies [20], colour-tile deterrents [13] or tracking dots [7].

In contrast, passive forensics does not require any explicitly added features. The quality of print outs is influenced by the corresponding printer mechanism and its components. This as well as several imperfections of such components produces artifacts within the printed document. Passive printer forensic methods try to find such artifacts or individual printing characterisics which are stable over several iterations, distinguishable among different printers and robust against influences. These artifacts can be used as identification features, called intrinsic signatures, of a specific printer technology, brand, model or the device itself. Traditional technologies in this area, such as physical [12], chemical [28] or microscopic [22, 23] methods, can give good results but are slow, require specialized equipment, educated employees and may destroy the document itself. Digital forensic science aims to improve the analysis in such a way that it can be carried out cost-effectively and automatically with standard commercial scanners. Depending on the type of document different signatures are important or extractable in order to be able to make relevant statements. Methods focusing on text documents mainly analyse the differences of texture and structure of printed characters (e.g. microtexture within the character, edge roughness, etc.). These features can be used to identify the source printer technology [5, 14, 25, 26] as well as the specific printer brand and model [8, 11, 29, 34, 36]. Geometrical distortion is another artifact which could be used as intrinsic signature for text [15, 35] as well as for image prints [1, 2, 16]. For images, the different implementations of the halftoning process [17, 18, 26] as well as the different colour representations [4, 30] are important features for an intrinsic analysis. Furthermore, there are methods that analyse the paper itself or extract traces left on the paper by the paper feed construction.

However, the complex printing process not only produces usable distinguishable signatures but even could change these signatures itself. Many variable parameters like different driver settings (e.g. toner save modi or resolution), age of the toner, change of components, used paper (plain vs. recycled), different font types and many others could potentially influence the intrinsic signatures (e.g. [10]). Furthermore these methods can differentiate at most

Thanks to BMWi for funding.
*equal contribution.

IH&MMSec '18, June 20–22, 2018, Innsbruck, Austria

ACM ISBN 978-1-4503-5625-1/18/06…$15.00
https://doi.org/10.1145/3206004.3206019

up to the printer model (including technology and brand) but not between printers of the same model. After all, a database with such an intrinsic signature of all existing printers is necessary for real forensic use, as otherwise misallocations may occur.

Active methods on the other side give clear results but are only usable for documents where the printing process can be controlled. An exception are methods implemented directly in the printer, like tracking dots which are used in nearly all colour laser printers. Through the constant existence of this signature, we propose to reuse these forensic patterns, e.g. in combination with passive printer forensic algorithms.

While the characteristics and information content of these patterns is chiefly unknown, we describe our analysis of these patterns in the following for reusability. Additionally we explore anonymisation approaches against this extrinsic signature to defeat arbitrary tracking. Finally we present our toolkit which implements the entire workflow of extracting, analysing and anonymisation of such a tracking dot pattern.

2 DOCUMENT COLOUR TRACKING DOTS

Many colour laser printer models print tiny and systematic yellow dots on each page. These are being generated at the firmware level [9] and represent encoded information such as the serial number of the printer or the date of the print [7]. This information can be read and decoded automatically. On the one hand, such tracking data is a helpful way of active forensics e.g. as a counterfeit protection system for bank notes. On the other hand, the tracking data is a lack of privacy. Theoretically it can not only be used by official authorities but also by any third party.

Since the origin and content of these yellow dots is largely unknown, we looked for answers from some printer manufacturers. We only received an official statement from one manufacturer, in which they called the yellow dots "Document Colour Tracking Dots". Unfortunately, they were not able to give any answer and referred us to the Central Bank Counterfeit Deterrence Group (CBCDG), which also could not answer our request because it is "not a CBCDG product/technology". Finally we worked on decoding the data by ourselves and found 4 distinct *tracking dot pattern* (TDP). Patterns 2, 3 and 4 had been mentioned in previous literature [33]. Pattern 4 had also been decoded by the Electronic Frontier Foundation (EFF) [7]. Here we introduce pattern 1 to the public for the first time, analyse the code and structure for each TDP and explain the information in one further code word from pattern 4.

2.1 Definitions

The *tracking dot matrix* (TDM) is one prototype of tracking dots in a matrix of $n_i \times n_j$ cells which is printed repeatedly over the whole sheet of paper with a cell distance of Δ_i inches horizontally and Δ_j inches vertically. Each cell of the matrix stores one bit where a yellow dot represents "1" and an empty space represents "0". A *tracking dot pattern* (TDP) is a format of storing tracking information. It uses a certain code, produces a TDM of a certain size and may include marking dots and a mask of empty cells. The TDP of a printer can be described as $(n_i, n_j, \Delta_i, \Delta_j)$.

A code could be algebraic or non-algebraic. In *algebraic* codes, A is the set of all code words. An information word a^* can be encoded in a code word $a \in A$ so that it is possible to detect or even correct a certain amount of erroneous bits. This is helpful when transferring data via a distorted channel such as yellow dots on a sheet of paper. A code described by the parameters (n, l, d_{min}) encodes information words of length l and adds $k = n-l$ redundant bits to the code words of length n. The amounts of ones in a code word a is called weight and noted as $w(a)$. The minimal weight among $2^l - 1$ nonzero code words is the minimal Hamming distance d_{min}. Binary codes can detect $f_e = d_{min} - 1$ errors in a distorted word $b = a \oplus e$ where e is the error word. A distorted word can only be reconstructed if the error word e has a weight of $f_k = \lfloor (d_{min} - 1)/2 \rfloor$ or less.

Figure 1: Two interleaved (4,3,2) even parity codes

000	0
011	0
111	1
100	1

An even parity code $(n, l = n - 1, d_{min} = 2)$ is a systematic code where the parity bit k is calculated by $k = \oplus_{i=1}^{l} u_i, u_i \in \{0, 1\}$. The weight of such code words is always even. An odd parity code is a parity code where the parity bit is $k = k \oplus 1$.

A product code (n, l, d_{min}) with an interconnected block interleaver (see fig. 1) is a code chain that consists of an outer code $(n_1, l_1, d_{min,1})$, an interleaver and an inner code $(n_2, l_2, d_{min,2})$ where $n = n_1 \cdot n_2$, $l = l_1 \cdot l_2$ and $d_{min} \geq d_{min,1} \cdot d_{min,2}$. An outer code writes code words of length n_1 row by row into a matrix before an inner code reads the words column by column of length l_2 and encodes them [19].

A "one hot encoding" $(n, n, 2)$ is a type of *non-algebraic* constant-weight code. It consists of one "1" and $n - 1$ zeros so that the weight of a code word a is $w(a) = 1$. Any error word e can certainly be detected with $w(e) \in [1..n] \backslash \{2\}$.

Example. Let $a = (01000)$ be a code word of a "one hot encoding" $(5, 5, 2)$, then the error word $e_1 = (01100)$ with weight 2 produces another code word: $b = a \oplus e_1 = (00100) \in A$. But the error word $e_2 = (00110)$ produces a word $b = a \oplus e_2 = (01110)$ with $b \notin A$.

2.2 Dataset

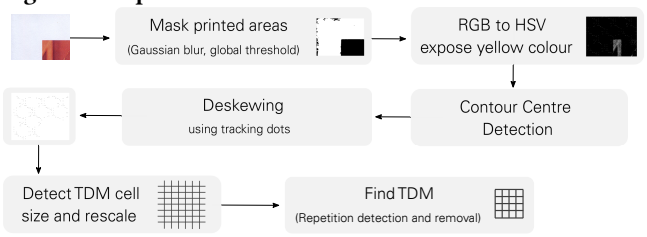
The tracking dots of 1286 prints by 106 printer models from 18 different manufacturers have been analysed (tab. 1), a total of 141 printers. This covers the majority of the world's most successful printer manufacturers [21]. For each printer model we considered up to three different printers. The data and prints were obtained from an archive by the DFKI [6] and from printers in the department of computer science of the TU Dresden. The DFKI data set contains prints from 132 printers. It provides the printer's manufacturer, model and serial number. Our TU Dresden data set additionally contains the date of the printing, information about the used driver, resolution and toner. Each print out has been digitalised with a common scanning device (Epson Perfection V30, 800dpi). The content of the documents consists of either images or text.

Table 1: Printer manufacturer in data set

Manufacturer	Analysed printers	Dots found
Brother	1	no
Canon	10	yes
Dell	4	yes
Epson	8	some models
Hewlett-Packard	43	some models
IBM	1	yes
Konica Minolta	21	some models
Kyocera	4	yes
Lanier	1	yes
Lexmark	6	some models
NRG	1	yes
Okidata	9	some models
Ricoh	6	yes
Samsung	5	no
Savin	1	yes
Tektronix	4	no
Unknown	1	yes
Xerox	15	some models

Canon, Brother, Hewlett-Packard, Konica Minolta, Ricoh and Xerox have signed an agreement of the Angloamerican Secret Service to fulfill "document identification requests" [32] which might have caused the tracking dots

Figure 2: Steps of the extraction workflow



2.3 Tracking Dot Extraction

This section will describe the method on reading arbitrary TDP and transforming a sheet into a list of TDM for further analysis of previously unknown TDP. The entire workflow can be practically tested with our deda toolkit (see section 6).

In comparison to previous work by van Beusekom et al. [33], our method maps the tracking dots into a grid and therefore transforms them into a matrix. For each two prints, van Beusekom et al. aimed at deciding whether they come from the same printer or not. To achieve this, they did not extract the TDM as a matrix but as an image – using a pixel threshold to match a TDM's repetitions on the sheet. For later decoding though, numerical TDMs are needed. Furthermore our extraction algorithm is independent regarding the content of the printed document.

First, the empty areas of the document must be detected, as the yellow dots in these areas are visible. Therefore we mask the printed areas using Gaussian Blur and a global threshold. After a colour space conversion to HSV and exposure of the yellow colour range, the set D of all recognised yellow dots can be created by a contour detection algorithm [27].

Table 2: Automatically detected TDP

Pattern	n_i	n_j	Δ_i	Δ_j
1	32	32	0.02 in	0.02 in
2	18	23	0.03 in	0.03 in
3	24	48	0.02 in	0.02 in
4	16	32	0.04 in	0.04 in

Next, the page needs to be aligned so that the tracking dots can be separated by straight lines into a grid. Because of the manual scanning process, the sheet might have been skewed by $\alpha \in \mathbb{R}$ degrees and must be corrected by a rotation of $-\alpha°$. It is possible to correct a skew up to 45° by taking advantage of the fact that on the sheet a TDM is being repeated many times in a straight line. Remember that the set D of yellow dots might be distorted. To approximate α, we calculate the angles between each two dots from D, quantise them and find the most occurring value.

When mapping the dots into a matrix, the cell separating grid might be shifted due to inaccuracies caused by a limited scan resolution and therefore skip a column or row each few centimetres. To prevent this, we segment the sheet into overlapping blocks of 7.5 cm per page. Each block is then processed separately and the block with the most dots is selected.

Afterwards the tracking dots are mapped from the page block into a grid. In a matrix, all cells of the same column are exactly one below the other. Due to imperfections in the scanning and/or printing process, the x coordinates of dots from the same column vary slightly. We call this bias. Let's assume Δ_1 and Δ_2 are the two smallest local maxima of the neighbouring dots' horizontal distances' frequency with $\Delta_1 < \Delta_2$. Δ_1 typically is the biased distance between dots of the same column and Δ_2 is the distance between dots of neighbouring columns: $\Delta_i = \Delta_2$. The vertical dot distance Δ_j can be calculated analogously. The grid shall be placed in a way so that the most occurring x coordinate of D is in the center of a cell and the most occurring y coordinate is in the center of a row. The cells have the size $\Delta_i \times \Delta_j$ inches. The tracking data yields "1" where there is a yellow dot in the grid cell and "0" everywhere else.

The TDM has been printed repeatedly over the whole sheet. Its dimensions should be detected given the matrix of all yellow dots. Let's assume a function that calculates the likelihood of two columns being identical. Then for each column c we calculate the median distance to each column $\hat{c}$ where the likelihood that the content of c and $\hat{c}$ is above a given threshold. The most occurring distance is assumed to be horizontal separation distance n_i cells. The vertical separation distance n_j can be calculated analoguously. If the sheet is being cut into pieces where each contains $n_i \times n_j$ cells, a list of (possibly distorted) TDMs result. For TDPs using a redundancy code (see 2.1), all TDMs shall be removed from that list where the redundancy check fails. Otherwise a TDM prototype can be estimated by overlapping all found TDMs and setting the value "0" or "1" by a majority decision.

3 FORENSIC ANALYSIS

Four different TDP tuples were detected in our dataset using the proposed extraction algorithm (see tab. 2). The patterns may appear rotated (90° steps) and/or flipped. The companies Lanier and Savin

Table 3: Patterns by manufacturer

Manufacturer	Pattern
Lanier	1
NRG	1
Ricoh	1
Savin	1
Hewlett-Packard	2
Kyocera	2
Lexmark	2
Okidata	2
Ricoh	2
Epson	3
Konica Minolta	3
Dell	4
Epson	4
Xerox	4

belong to the company of Ricoh [24] which use pattern 1 together with the company NRG. Pattern 2 is being used by 5 different independent manufacturers (tab. 3). The patterns 1, 2 and 3 are constant for each printer and do not vary by each print. Hence we assume that they contain fixed information like the printer's serial number but not the date. Many Canon printers showed a pattern that is not constant but seems to repeat its TDM in a rotated transformation. This pattern has not been analysed further because of its unusual irregularity.

All other detected TDPs were analysed and some matrices decoded. The patterns were evaluated according to their information density, capacity, error detection rate and conspicuousness. We also analysed the number of yellow dots generated by each pattern in the best, worst and average case, depending on the content of the TDM.

For a TDM, let *col* be the column and *row* the row index number. All patterns use a kind of repetition code because their matrices are spread over the whole sheet of paper. Therefore forward error correction can be achieved using the repetitions of a matrix. The amount of repetitions depends on the size of the printed area and the size of the matrix.

Each section relates to only one prototype of the matrix. The same statements always apply to its repetitions.

3.1 Analysis Methods on TDP

The structures of all patterns have been determined by analysing a bigger amount of TDMs.

Due to the repetitions of a pattern over the entire printout, each pattern is likely to contain marker of its beginning. To find possible markers, we overlapped all TDMs of the same pattern from different printers such that the resulting matrix shows only a dot where all matrices show a dot. Dots that appear in all TDM samples do not contain information and can be used as orientation markers therefore (red in figures).

Furthermore a TDP may contain empty cells, rows or columns that need to be skipped when reading the data. To determine them, all TDMs of one pattern were overlapped so that the resulting matrix shows a dot where at least one matrix shows a dot (fig. 3). Thereby the cells become visible that are empty on all TDMs. They

Figure 3: All matrices of pattern 3 united

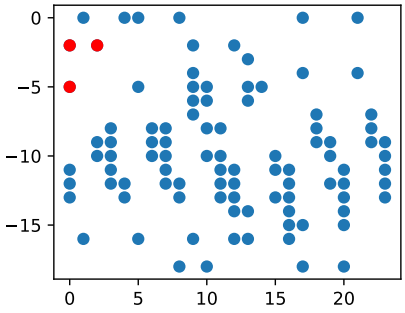
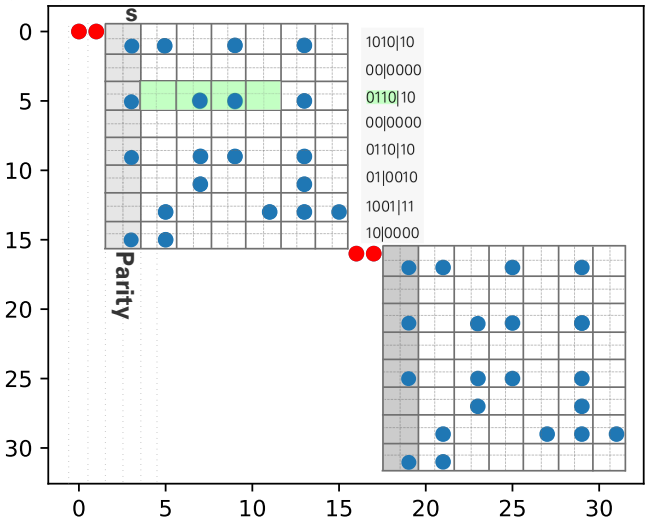


Figure 4: Pattern 1: Marking (red) and other tracking dots (blue)



may mark spaces between data blocks. If there is only one dot in each of these blocks of size n, then its data may be stored in a "one hot encoding" of length n – written row by row or column by column.

The information stored in a matrix was attempted to reveal by analyzing the inference of metadata (e.g. the printer's serial number) to the matrix. For the printers, for which both the serial number and the TDP were known, a known-plaintext attack could be achieved.

3.2 Pattern 1

The first pattern is printed offset (fig. 4). Therefore its dimension is detected as 32×32 cells although the unique matrix with the spacing uses $32 \times 16 = 512$ cells. This section deals with the prototype of the matrix in rows 0-15 and columns 0-15. The pattern marks its beginning with two neighbouring dots (red in figure) and stores information in every second column in every second row. All even rows do not contain any dot except the marking ones. Each row is one code word. Let s be the index of the first column that contains

the first code word bit in the row. s is either 2 or 3 depending on the printer. In our figure s is 3. This pattern has been discovered on 7 different devices.

Redundancy check. The pattern uses a (7,6,2) even parity code. It stores 8 code words row by row which contain $8 \cdot 6 = 48$ information bits in total. A TDM is considered as valid if the amount of dots is even in all rows. Error words with an even weight produce code words. To detect them, all valid TDMs have to be compared and chosen by a majority decision.

Example. Row 31 contains the code word (1100000) and passes the parity check. The correct word might have been (0000000) as well united with the error word $e = (1100000)$. This error with $w(e) > 2$ cannot be detected.

To improve the error detection one could check the condition that on the one hand every second column from $s-1$ to 15 is empty and on the other hand that each even row from 2 to 14 is empty as well. Using this condition the probability of decoding a word wrong due to burst errors, e.g. through printing/scanning artifacts, is much lower.

Decoding. The pattern contains the printer's serial number as 4 binary bit blocks in the said (7,6,2) even parity code. Being a systematic code makes it easily readable. For the rows $1, 3, 5, \ldots, 15$, column $s = 3$ contains the parity bit and the information bits can be found in every second column from $s + 2$ to 15. A binary chain has to be read from left to right starting with the bottom row. Each 4 bits of this chain represent a binary number. The 11 binary numbers before the last one are the printer's serial number. The first and fifth number may represent letters, where "9" stands for "P" and "0" stands for "W" or "Q".

Example. Figure 4 contains the words (1101010), (0000000), (1011010), (0000000), (1011010), (0010010), (0100111) and (1100000). The information bits without the leading parity bit are (101010), (000000), (011010), ..., (100000). Splitting this chain into 4 bit chunks results in (1010), (1000), (0000), (0110), ..., (0000). Reading the chain as well as the chunks backwards and transforming them into decimal numbers gives us the string "0**7949601605**1".
The serial number is W**794**P**601601**.

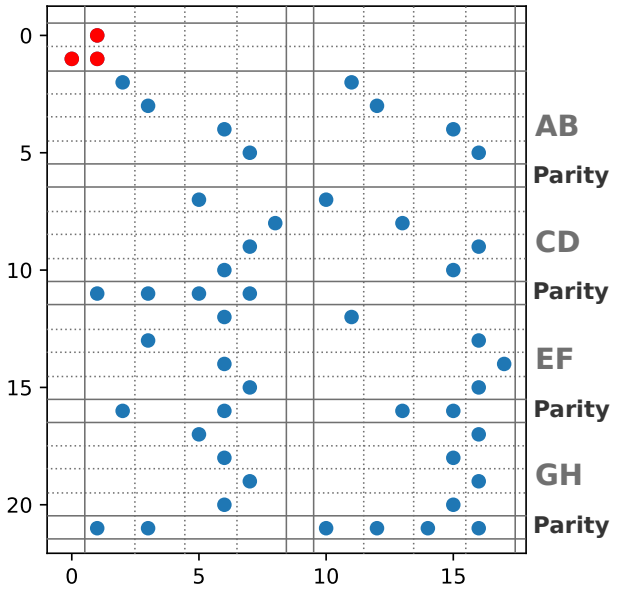
Conspicuousness. The amount of dots per code word a is determined by its even weight $w(a) \in \{0, 2, 4, 6\}$. From 64 code words with an equal probability of occurrence, 35 have a weight of 4 (54,7%)[1]. This makes 8 code words · 4 dots + 2 marking dots = 34 dots per matrix at average (0.67 dots per bit).

3.3 Pattern 2

Pattern 2 uses 18×23 = 414 cells (fig. 5) and was found on 51 devices. Depending on the printer, each dot in the figure is represented by one or two printed dots. Three dots in the first two rows mark the beginning (red in figure). The TDM consists of eight blocks named A to H (from left to right) situated in rows 2-6, 7-11, 12-16, 17-21 and columns 1-8 and 10-17. Row 22 as well as columns 0 and 9 are separators. The pattern considers every second column in rows with an even index and every first column otherwise, so all cells are

[1]Calculated with binomial distribution

Figure 5: Pattern 2: Marking (red) and other tracking dots (blue) aligned into blocks A-H with parity. The estimated grid has been added to this figure for readability



being considered where $(col \bmod 9 + row) \bmod 2 = 0$. This pattern stores 4 signs from a (4,4,2) "one hot encoding" and interleaves it in a (5,4,2) odd parity code. This results in a (20,16,4) product code where each code word can differentiate between 4^4 different states. The pattern consists of 8 code words, so it stores $4^{4 \cdot 8}$ different states in total. This is equivalent to storing 64 bits.

Example. Block G contains

0010
0010
0001
0010
1100

the last row is the parity.

Redundancy Check. Each of the first four rows of each block contains information bits as a "one hot encoding". The fifth row contains the parity bits of the outer encoding which make an odd amount of dots in each column of each block. This helps to detect errors e with $w(e) = 2$ which are not detected by the "one hot encoding". Each inner code word contains exactly one "1", so 1, 3 or 4 faulty bits can be detected. The product code can detect any number of faulty bits that is 2 or odd. Moreover, error correction is possible.

Example. If $a = (1000\,0100\,0010\,0001\,0000)$ from block A is being distorted with an error word $e = (1100\,1100\,0000\,0000\,0000)$ then e is one of the few error words with weight 4 that produces a code word which matches the parity bits with
$b = a \oplus e = (0100\,1000\,0010\,0001\,0000) \in A$.

Table 4: TDM's block A by manufacturer

Manufacturer	HP	Kyocera	Lexmark	Okidata	Ricoh	Ricoh
Block A	3021	0123	0213	3210	2310	0132

Decoding. Block A equals block B in all samples. They correlate to the printer's manufacturer. Printers using this pattern have serial numbers like CNBB002529, CNBC55MOPR, JPGMC52527, etc. Blocks C and D correlate to the 2nd, 3rd and 4th letter from the serial number. But these blocks are ambiguous: one can conclude them from the serial number but not vice versa. The information of blocks E-H is uncertain. It may contain encrypted digits from the serial number or some other data. To obtain the information part from the matrix, the first four rows of each block can be interpreted as a number in $\mathbb{Z}_4$ (translate "0001" into "0", "0010" into "1" etc.).

Example. Block A from our figure contains the information bits $(1000\ 0100\ 0010\ 0001)$. These represent the numbers 3, 2, 1, 0 and signify an Okidata printer.

From the information in block A, the printer manufacturer can be concluded (tab. 4). Obviously there has been a preference for chains of distinct digits to identify the manufacturer. The advantage of these numbers is that they produce less dots. Only if a block consists of all distinct numbers, the amount of added parity dots is minimal.

Conspicuousness. The amount of dots per code word ranges from 4 to 8. If one word produces 4 dots, it consists of 4 different inner code words and all parity bits are 0. In the worst case one word produces 8 dots: This occurs when all parity bits are being set to 1. The average amount of dots from all code words is 6. There are $4^4 = 256$ different code words per block. 192 of these either contain three identical numbers and one different one (example 3.1) or two identical numbers and two other numbers of which both differ (example 3.2). Both cases produce exactly 2 parity bits which are 1. All code words contain exactly 4 dots from the information bits. This sums up to 4 information dots + 2 parity dots = 6 dots. For the whole pattern this means $8 \cdot 6 + 3$ marking dots = 51 dots at average (0.80 dots per bit).

Example 3.1. The inner code words (1000), (1000), (1000), (0100) produce an outer code word where the parity bits are (0011). Two parity bits are being set.

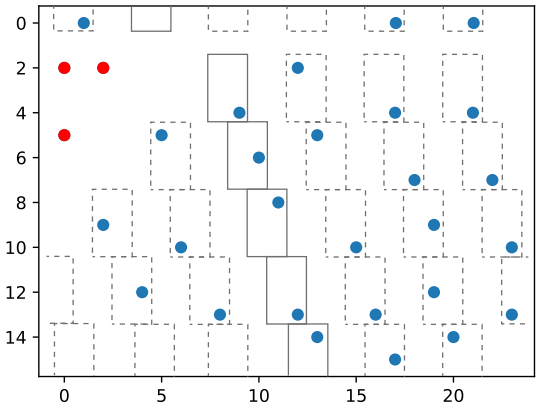
1000
1000
1000
0100
0011

Example 3.2. The inner code words (1000), (1000), (0100), (0010) produce the parity bits (1001). Again two parity bits are being set.

3.4 Pattern 3

Pattern 3 (fig. 6) consists of 27 blocks of 6 bits where each block uses 2 columns and 3 rows. The pattern's beginning is marked by three dots (red in figure) which do not fit into one block. The pattern is being repeated over the whole sheet. Its detected shape is 24×48 cells because each vertical repetition of the pattern is being shifted by +8 columns. The unique pattern consists of $24 \times 16 = 384$ cells.

Figure 6: Pattern 3: Marking (red) and other tracking dots (blue). Rectangles have been added to this figure to mark our detected code word blocks. The solid boxes indicate the pattern's offset.



Redundancy Check. The source alphabet contains 6 elements which are being encoded with a (6,6,2) "one hot encoding". The pattern consists of 27 blocks where each block stores one code word. A code word weights "1" and therefore produces exactly one tracking dot.

Decoding. Currently we found no correlation with any of the printer's know properties.

Conspicuousness. Considering the markers, there are 27 + 3 = 30 tracking dots in total (0.43 dots per bit). The pattern can differentiate between 6^{27} states in total. This is equivalent to storing $\ln(6^{27})/\ln(2) \approx 69.8$ bits.

3.5 Pattern 4

Pattern 4 (fig. 7) is being used by Dell, Epson and Xerox printers. According to a research fellow at Xerox, the U.S. government and his company have a „good relationship" [31] which might be the origin of this pattern. Pattern 4 uses $16 \times 16 = 256$ cells and is being repeated offset (16×32 cells in total). This section relates to the matrix in rows 0-15 and columns 0-7. There are 3 or 7 marking dots (sometimes called "separators") in row 6 although they are missing on some printers. The pattern encodes words with a (8,7,2) odd parity code and interleaves 14 code words in a (15,14,2) odd parity code. This results in a (120,98,4) product code. The parity bits are in row 15 as well as in column 0. For some printers the outer parity does not cover the inner parity bits (e.g. see fig. 7 col 0, row 15). The pattern stores 98 information bits in total. Overall 16 devices in our dataset use this pattern.

Figure 7: Pattern 4: Marking (red) and other tracking dots (blue)

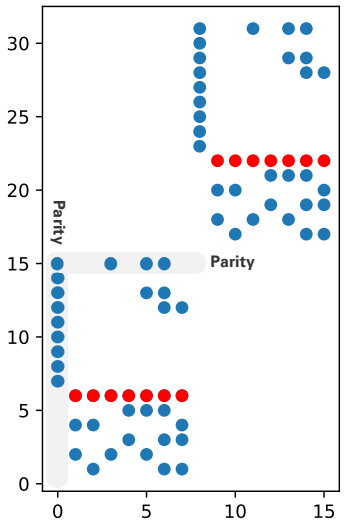


Table 5: Number in TDM's row 12 by manufacturer

Manufacturer	Dell	Epson	Xerox	Xerox
Row 12	20	3	0	4

Redundancy Check. Code words in rows 1 to 14 as well as each of columns 1 to 7 must show an odd amount of dots. The product code allows error correction.

Decoding. Each row has to be transcribed into a binary number excluding the leading parity column. The resulting number can be transformed into the decimal system. The TDM contains the date and time of the print. The manufacturers Epson and Xerox add 6 digits of the serial number as well. Dell's TDMs do not include them. The minutes can be found in row 14, the hour in row 11, the day in row 10, the month in row 9 and the year in row 8. The middle of the serial number is a concatenation of the numbers from rows 3, 4, 5. Row 12 correlates with the manufacturer (tab. 5) and row 7 has been constantly empty (except parity bit). The meaning of the information in rows 1, 2 and 13 does not correlate with any of the printer's known features. Row 15 contains parity bits and row 0 is always empty.

Conspicuousness. If we assume that the pattern stores an arbitrary serial number and a date where the hour ranges from 0-23, the minutes from 0-59, the day from 1-31, the month from 1-12 and the year from 0-127 then it produces between 12 and 76 tracking dots. At average it makes 46 tracking dots (0.47 dots per bit).

Table 6: Pattern Comparison

Pattern	Δ_i	Size		Capacity	Density		Dots/in²
		Cells	in²		Bits/cell	Bits/in²	avg / max
1	0.02 in	512	0.21	48 bits	0.09	234.38	166 / 244
2	0.03 in	414	0.37	64 bits	0.15	171.77	137 / 180
3	0.02 in	384	0.15	69 bits	0.18	454.43	195 / 195
4	0.04 in	256	0.41	98 bits	0.38	239.02	112 / 186

Table 7: Code parameters

Pattern	Dots per word avg / max	(n, l, d_{min})	f_e	f_k	f_e
1	34 / 50	(7,6,2)	1	0	14%
2	51 / 67	(20,16,4)	3	1	15%/100%[1),2)]
3	30 / 30	(6,6,2)	6[(1)]	0	100%[1)]
4	46 / 76	(120,98,4)	3	1	14%[2)]

1) An error of two bits per code word might not be detected in any case. Presence of parity bits has not been revealed.
2) For the inner code of the code chain

3.6 Evaluation of the Patterns

Table 6 gives an overview over the detected patterns. It notes the amount of cells of one unique matrix including the spacing to its closest repetition. The capacity for storing information bits is given as well as the amount of bits that one table cell and one square inch can store. The density is the quotient of the capacity and the size. The amount in bits per cell shows the efficiency of the patterns regardless of the cell distance whereas the number in bits per square inch does consider the cell distance. The more dots per square inch are printed the more visually conspicuous the matrix is on the paper. The amount of dots per matrix depends on the encoded information. Its minimum, average and maximum are given in dots/in², divided by the pattern's size. The (n, l, d_{min}) code description (tab. 7) follows with the amounts f_e and f_k of faulty bits that can be detected and/or restored correctly (forward error correction).

Comparison. Comparing the patterns leads to the following observations: Pattern 1 encodes information per in² very densely because it uses the binary system and produces few redundancy so it only detects single faulty bits and cannot correct errors at all. Pattern 4 is similar to pattern 1 but adds error correction and therefore displays information less densely. The density per cell of pattern 4 is higher than of pattern 1 because pattern 1 leaves nearly every second column and row empty, assuming a cell distance of $\Delta_i = 0.02$ in. Pattern 2 uses the "one hot encoding" and a parity code. The "one hot encoding" with length 4 still can display information quite densely and can detect distributed erroneous bits quite well. On the other hand it only detects at maximum 4 faulty bits whereas a "one hot encoding" with a higher length has a lower information density but can detect errors of a higher weight. The parity bits lead to a high redundancy.

Pattern 3 uses a "one hot encoding" with length 6. The presence of parity bits has not been found so we assume that all bits are information bits. A "one hot encoding" with length 6 stores little

Table 8: Redundancy check for known patterns from 300 dpi scans of each printer from our dataset

Pattern	Passed	Out of
1	7	7
2	47	50
3	17	23
4	15	15
-	0	46

information per sign but the absence of additional redundancy helps this pattern to create a higher density than pattern 2. The "one hot encoding" allows 6 faulty bits to be detected. Pattern 1 can be the least conspicuous one because it produces the lowest amount of dots per in² in the best case. Though the worst case is likely to occur if the pattern is being used to store a big variety of information. Pattern 2 produces at maximum 180 yellow dots per in² which is the lowest maximum for all patterns. Pattern 3 produces just 15 dots more but stores 2.6 times of the information of pattern 2 per in². Pattern 4 does not use an explicit marker. It can be aligned by the definition of the free space of its offset pattern but this is a lot more computationally expensive than finding the three marking dots of pattern 2. Code words of Pattern 1 can be created by the random distortion on the paper. This makes it difficult to find unambiguous information and also to determine whether a sheet does or does not contain this pattern.

Conclusion. The most efficient pattern is pattern 3. Storing more than 454 bits per in², it has the highest density. Because error correction can be achieved using the repetitions anyway, it is reasonable to focus on a high error detection capability rather than on forward error correction. This pattern has the highest error detection capability. A small cell distance of 0.02 in is useful to allow many cells per area. This is possible due to the few dots produced by the "one hot encoding". In the worst case 195 tracking dots are being produced which is similar to the other patterns. Pattern 3 has enough capacity to store the same information as in all other patterns without time and date information. Pattern 2 minimises its dots because it sets the parity to an odd amount of ones. Blocks where all four inner code words are different occur a lot more often than a block where all code words are the same. Therefore the outer parity code deals with odd amounts of "1" more often and sets "0" as the parity bit in this case.

4 REFINED EXTRACTION ALGORITHM

This is a very resilient method for classifying tracking information and comparing tracking information of different printers. In the code comparison in [33], each two prints from the same or different printers were analysed to classify a common or a distinct origin. To achieve this, all dots that did not appear in many of the TDM's repetitions have been removed greedily. This may allow false positives: The prints from two different printers might be detected as from the same origin if significant dots were removed. In contrast, this method uses the code's redundancy check. Comparing only valid TDMs, a classification of two different printers as identical is very unlikely.

To find a valid TDM, the tracking dots must be extracted. Different yellow colour ranges may be tried. Then any repetition of the TDM must be selected from the sheet. Initially it must be shifted so that the marking dots are on the desired place (top left corner). If an offset TDP does not provide marking dots, the matrix can be shifted according to its empty space. An offset repetition has to be removed from the extracted matrix if it contains any. Next, markers and spaces have to be removed from the matrix so that it only contains cells that belong to the code word. The result's redundant bits can be checked according to the code's description. If the check fails, the algorithm has to be repeated using another prototype from the TDM repetitions on the sheet until a valid TDM has been found.

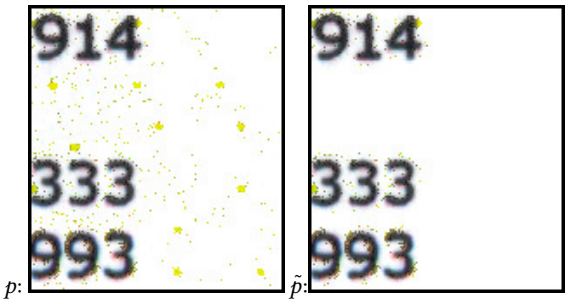
The table 8 shows the amount of printers where the prints' tracking information passed the redundancy check successfully at least once on at least one sheet out of 10. It shows the amount of printers belonging to a pattern as well. Each of the printer's sheets has been tried a redundancy check on for all patterns. The last row tells us that pages without any known or without any TDP at all did not pass redundancy checks. From pattern 1 the extraction of a TDM often is ambiguous. A randomly distorted matrix could be valid according to the redundancy check by pattern 1. Especially the markers from pattern 2 are valid markers for pattern 1 and can therefore be interpreted as a pattern 1 matrix more easily. The spots around valid dots have to be checked carefully before deciding on pattern 1 for a sheet. The prints of which the tracking information could not be decoded showed sparse matrices or were scanned unluckily. Especially pattern 3 prints were hardly distorted on the scan.

Furthermore, we evaluated the extraction of the printers' manufacturer and serial number from a valid TDM. 100% of the extracted serial numbers are part of the printer's actual serial number. The manufacturers for all but one printer could be decoded correctly. The erroneous printer[2] has possibly been labelled wrong in the dataset.

A comparison to the evaluation in [33] might not be helpful because the used test scenario cannot reveal the advantages of our method. There, only printers were considered where the tracking dots differ heavily. Van Beusekom et al. did not find the patterns' codes and therefore could not use the redundancy check to extract a distortionless TDM. Instead they compare only the most certain dots in a TDM which are being concluded from a scan using the TDM's repetitions and a threshold to separate true dots from distortion. In our tests, this often lead to a distorted TDM that contains only few dots. The results of the evaluation by van Beusekom et al. is quite good because their method is sufficient to spot printers with a very different TDM and their dataset does not include many TDMs that differ only slightly. In a huge dataset, printers with a similar TDM might appear which could be detected as identical. Our method makes it possible to spot the difference between printers that produce very similar TDMs. A false detection of two printers as identical is very unlikely for our method because we require any true TDM to pass the redundancy check. This extraction method is also included in the deda toolkit (section 6).

[2] Sample 99 in the DFKI dataset

Figure 8: A scan before (p) and after ($\tilde{p}$) automatic tracking dot removal. Yellow colours are darkened equally on p and $\tilde{p}$ for better visibility



5 AN ANONYMISATION APPROACH

Tracking dots on a sheet reveal information about the printer and are therefore a lack of privacy. Tracking dots information have no controlled access and can theoretically be read and decoded by anyone. It is possible to conclude the printer of a sheet which is often owned by the author. This would be a disaster e.g. in case the sheet is a critical leaflet about the government in a dictatorship. For this reason we introduce methods for removing tracking data from scanned prints and for masking tracking data on prints. Each anonymisation method was successfully tested using our deda toolkit.

5.1 Removing Tracking Dots on Scans

When scanned documents are being sent via the internet, they might contain tracking information. Tracking dots may have a strong effect: In 2017, a document by the NSA has been published without authorisation[3]. The most probable reason for having identified the publisher is the tracking dots. Tracking dots can mostly be removed from scans (fig. 8) by clearing the original document's empty areas as detected in section 2.3.

5.2 Masking Tracking Dots on Prints

A custom TDM shall be added as a mask on top of the printer's TDM to prevent restoring a word b correctly. The ambiguity of correcting b must be high enough to allow many possibilities on restoring a masked TDM: Some code words can be detected as wrong by their content, e.g. if a decoded word contains a month greater than 12 or an invalid serial number. Wesselman et al. [9] have mentioned to print a full unit matrix on the sheet to prevent an unambiguous decoding of a pattern 4 matrix. Their mask puts a dot on all possible cells. Though, printing a yellow dot in all cells makes the TDM very conspicuous and uses a lot more toner. This might not be necessary. Because we know the codes used by the different patterns, for a given TDM we want to find a mask that has as few dots as possible but makes the decoding ambiguous when it is being united with the original TDM. The mask has the same size as the printer's TDM and must cover all of the TDM's repetitions on the sheet constantly. For each pattern there is a different algorithm to create a mask. The

[3] https://qz.com/1002927/computer-printers-have-been-quietly-embedding-tracking-codes-in-documents-for-decades

Figure 9: Pattern 1 mask example (green)

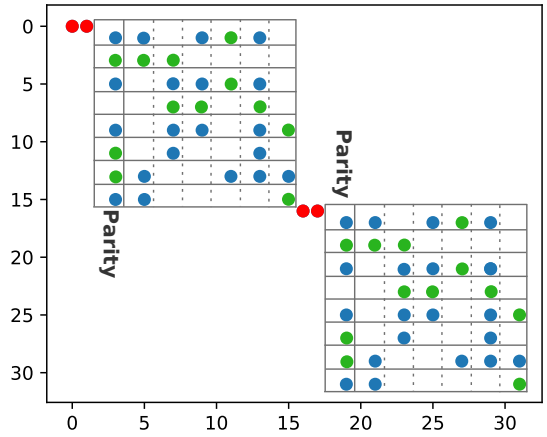
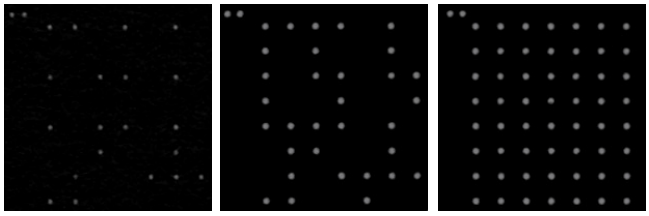


Figure 10: Pattern 1 practical mask example. From left to right: original TDM, masked TDM, fully dotted TDM

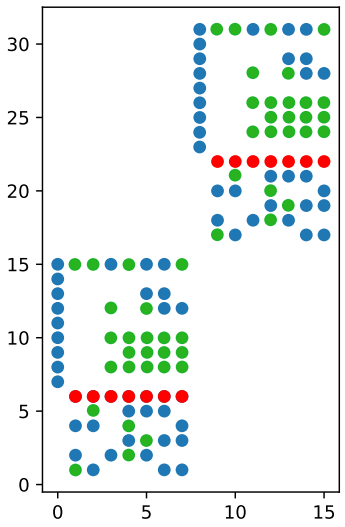


offset patterns 1 and 4 must not apply the mask on the empty areas because otherwise the mask could be concluded and therefore be removed by subtracting it from its union with the original TDM.

5.2.1 Pattern 1. The mask (fig. 9 and 10) is being created as follows. Let $s = 3$ if the information dots are placed in the odd columns and $s = 2$ otherwise. On each of the rows 1, 3, 5, 7, 9, 11, 13 and 15, one empty cell has to be chosen at random where the cell number must be one of $s, s+2, s+4, s+6, s+8, s+10, s+12$. These cells must carry a dot on the mask. The parity will be broken and the adversary does not know which dot has been added. From rows containing exactly one dot, it is unambiguous which dot we added. If we added two dots to each row that was empty on the original TDM, the parity would reveal that we have added an even amount of dots which must be smaller or equal than two. Therefore empty rows must be added three dots. Exactly the same mask starting at column 0, row 0 has to be repeated from column 16, row 16. This adds at least 8 dots to each TDM.

If our algorithm is known to the adversary and he wants to restore the masked TDM, there are $\prod_{r=1,3,\ldots,15} d(r) \geq 3^8 \approx 2^{12.7}$ possible code words where $d(r)$ is the amount of dots in row r with $d(r) \geq 3$. If the parity bit in row r is not being set, the adversary will remove one of each of the dots. On rows where the parity bit is being set, itself might as well be considered as the flipped bit from our mask.

Figure 11: Pattern 4 mask example (green)



Using this interpretation, an additional possible code word can be concluded from the unchanged information bits. The ambiguity can be low if each of the the original TDM's rows contain very few dots. To increase it, to each row a dot can be added at a randomly chosen cell. The parity in these rows will be correct.

5.2.2 Patterns 2 and 3. Because the information in the code words is not completely known to the public and might use a further error correcting code, filling all blocks completely with dots is a safe option to anonymise the TDM (see fig. 10 right TDM).

5.2.3 Pattern 4. The mask (fig. 11) for columns 0-7, rows 1-5 shall carry a dot on a randomly chosen empty cell for each row and fill the outer parity row 15 completely with dots between columns 0 and 7. To obscure the manufacturer, row 12 has to be modified. All known manufacturers use one of the numbers 3, 4 or 20 resp. binary numbers 11, 100 and 10100. From their disjunction 10111, none of the original number can be concluded. So it is sufficient to fill columns 7, 6, 5 and 3 of row 12. To hide the date of the printing process, rows 8, 9 and 10 have to be treated in a specific way: Filling columns 3-7 in row 8 hides the domain of the year, filling columns 4-7 in row 9 hides the domain of the month and filling columns 3-7 in row 10 hides the domain of the day. The hour and minute without the date are irrelevant and not being considered any further here. If rows 3, 4 and 5 are filled so that each row contains at least s dots, then at least s^3 different serial numbers can be concluded from the TDM. A copy of the mask shall be placed on columns 8-15, rows 17-30. This adds between 6 and 8+5+4+5+2+12=36 dots to each TDM.

Proposition 5.1. *Let's assume a parity code that makes the weight of a code word odd. If a distortion flips at least one "0" to a "1" but never vice versa and the error word e is unknown, then for a distorted word b with weight $w(b) \in [3..\infty)$ there are $s(w)$ different possible code words a such that $a \oplus e = b$ with*

$$s(b) = \sum_{w(e)=1,3,\ldots,w-1-w'} \binom{w(b)}{w(e)+w'(b)} \text{ where } w'(b) = w(b) \bmod 2.$$

Proof. For a word b_1 with an odd weight $w(b_1)$, the parity is being satisfied and $w'(b_1)$ is 1. Because the code word a also satisfies the parity condition, an even amount of "1" has to be flipped to create another code word. $w(e)$ is one of $\{2, 4, \ldots, w(b_1) - 1\}$. There are $\binom{w(b_1)}{w(e)}$ possibilities for choosing e and restoring $a_1 = b_1 \oplus e$. For a word b_2 of an even weight $w(b_2)$ (with $w'(b_2) = 0$), the amount $w(e)$ of changed bits must be odd for the word to originate from a valid code word. So $w(e) \in \{1, 3, \ldots, w(b_2) - 1\}$. For any word $b = a \oplus e$ with an even or odd weight, $w(e)$ is one of $\{1 + w'(b), 3 + w'(b), \ldots, w(b) - 1\}$. Considering all possible $w(e)$ the result is

$$s(b) = \sum_{w(e)=1+w'(b),3+w'(b),\ldots,w(b)-1} \binom{w(b)}{w(e)} = \sum_{w(e)=1,3,\ldots,w(b)-1-w'(b)} \binom{w(b)}{w(e)+w'(b)}$$

□

Example. A distorted word is $b = (1110)$. It has three "1" so $w(b) = 3$. The weight is odd so b is a code word. There are $s(3) = 3$ different code words that might have caused b: $a_1 = 0010$, $a_2 = 0100$ or $a_3 = (1000)$. Note that (1110) cannot have caused b because according to the proposition the error word has a weight of at least 1 but $w((1110)) = w(b)$. (1100) and (0000) would not be code words because they do not satisfy the parity condition.

The serial number is separated into the rows 3, 4 and 5. Let's assume each of these rows has been added at least one dot but enough dots to contain min_d dots in total. Let b_r be the word in row r with $w(b_r) \geq min_d$. The amount of different possible serial numbers an adversary can conclude from a masked TDM is $\hat{s}(b_3) \cdot \hat{s}(b_4) \cdot \hat{s}(b_5)$ where

$$\hat{s}(b) = \begin{cases} w(b) & \text{if } w(b) > min_d \\ & \vee w(b) \leq 2 \\ \sum_{w(e)=1,3,\ldots,w(b)-1-(w(b) \bmod 2)} \binom{w(b)}{w(e)+(w(b) \bmod 2)} & \text{otherwise} \end{cases}$$

If for a word $b = a \oplus e$, $w(b)$ is bigger than min_d and the adversary does not know e, then he can conclude that e contains one "1" which could be any of the $w(b)$ dots giving $w(b)$ possibilities to find it. If $w(b)$ is 1 or 2 then there are only 1 resp. 2 possibilities for determining a. Otherwise $w(a)$ is not shown to the adversary and the amount of possibilities for choosing a is a sum (prop. 5.1). The date is totally being hidden in the TDM, though if the date can be concluded from the printed content and the outer parity bits were not masked, in the worst case the outer parity could help to separate the original TDM from the mask.

Example. If the masked TDM is

	01234567
15	11111111
14	00110000
13	00110000
12	00110000
11	00110000
10	00110000
9	00110000
8	00110000
7	00110000
6	00110000
5	00100100
4	00100010
3	00100001
2	00110000
1	00110000

then the parity bits in the top row inform that there are errors in columns 5, 6 and 7. The parity bits in column 0 point at a distortion in rows 1-14. Moreover the adversary knows that one "0" per row has been changed to a "1". Therefore the ones in rows 3-5, columns 5-7 must have been changed by the mask. It is possible to restore rows 3, 4 and 5:

	01234567
5	00100000
4	00100000
3	00100000

These rows contain the printer's serial number (see section 3.5). To prevent the support by the outer parity bits, they are all being set to "1" to remove information.

6 DEDA TOOLKIT

The entire workflow of TDM extraction, retrieving the known information content and the generation of an anonymisation pattern could be obtained with our provided Dot Extraction, Decoding and Anonymisation toolkit *deda*. The code is freely available at dfd.inf.tu-dresden.de[4]. For further guidance have a look at our toolkit's README file. For the print anonymisation part a calibration sheet has to be printed and scanned. The printer's TDM is being read from this scan and an anonymisation mask is being created on top. This calibration step is described in the following. After generation of the anonymisation pattern it can be merged with the document to be printed. Note that it has to be printed borderless. If this is not possible, the margins have to be cut off, otherwise TDMs might be reconstructed.

Calibrating the TDM's location. The mask needs to be printed in a way to join the printer's native tracking dots although their position on the sheet is unknown. If a test image has been printed that contains markers in each edge on the Cartesian points A, B, C, D, then its scan can be aligned so that its coordinate system matches the test image's coordinates. An additional marker O is located in one edge of the page and used to maintain the page orientation. A contour detection algorithm [27] is used to find the markers in the scan. It is recommended to use the printer toner's colours cyan and magenta. Yellow is already used by the tracking dots and black is being displayed at the border of the scanned page. All other colours are not recommended because they would be printed using halftones.

First, the scan needs to be rotated so that point O is in the same edge as in the test image. Then all other points A', B', C', D' need to be matched against the markers at A, B, C, D in the test image by filtering markers according to coordinate ranges and focusing on the smallest x and y coordinate for each marker. After applying a perspective transform mapping A' to A, B' to B etc., any point $Z \in \{A, ..., D\}$ on the scan would represent Z' on the test image so that $Z = Z'$. The process of scanning a document causes geometrical distortions. These distortions are minimal near the alignment markers A, B, C, D. Let's assume the printer's TDP has the parameters $n_i, n_j, \Delta_i, \Delta_j$. Now we need to find the offset coordinates x_o, y_o near the top left corner where the first TDM begins. If four valid TDMs have been found at points $\hat{A}, \hat{B}, \hat{C}, \hat{D}$ where $\hat{A} = \left(x_{\hat{A}}, y_{\hat{A}}\right)$ is close to A, $\hat{B} = \left(x_{\hat{B}}, y_{\hat{B}}\right)$ is close to B, etc. then the best estimation of x_o is the average of the TDM's offsets at $\hat{A}, \hat{B}, \hat{C}, \hat{D}$. We use modulo to calculate the offset given the coordinate. If the pattern's offset is close to 0, a constant $c \in \mathbb{R}$ must be chosen so that all x values can be mapped into a range $(x + c)\text{mod}(n_i \cdot \Delta_i)$ that allows us to calculate a meaningful average.

$$x_o = \text{average}((x_{\hat{A}} + c)\text{mod}(n_i \cdot \Delta_i), (x_{\hat{B}} + c)\text{mod}(n_i \cdot \Delta_i), (x_{\hat{C}} + c)\text{mod}(n_i \cdot \Delta_i), (x_{\hat{D}} + c)\text{mod}(n_i \cdot \Delta_i))$$

y_o can be calculated analogously. If a page shall be printed and anonymised, the calculated mask shall be transformed into an image having cell distances of Δ_i, Δ_j. It must be printed at $(x_o|y_o)$ and at all points where a repetition of the TDM begins: $\{(x_o + x \cdot \Delta_i \cdot n_i | y_o + y \cdot \Delta_j \cdot n_j) \mid x, y \in \mathbb{Z}\}$. Now the tracking dots information has been made ambiguous. Note that the page margin must be identical for printing the test image and the anonymised document. If the margin has not been zero, there might remain unmasked tracking dots on it.

7 CONCLUSION

In this work we analysed document colour tracking dots, an extrinsic signature embedded in nearly all colour laser printers. From printer forensics point of view we propose to reuse these forensical patterns in combination with existing passive printer forensic algorithms. Since the properties and information content is chiefly unknown, we have researched methods of analysis for reusability. In total, we discovered 4 patterns and succeeded in automatically extracting and decoding the structure of the patterns as well as interpreting patterns 1, 4 and partially pattern 2. Further investigations are required to interpret the total information content for Patterns 2 and 3, e.g. on printers firmware level[5]. From a privacy point of view we explored anonymisation approaches to prevent arbitrary tracking. The whole workflow is provided with our freely available toolkit *deda*.

REFERENCES

[1] Gazi N Ali, Aravind K Mikkilineni, Jan P Allebach, Edward J Delp, Pei-Ju Chiang, and George T Chiu. 2003. Intrinsic and extrinsic signatures for information hiding and secure printing with electrophotographic devices. In NIP & Digital

[4] or https://github.com/dfd-tud/deda

[5] https://events.ccc.de/congress/2011/Fahrplan/events/4780.en.html; 08/02/2018

Fabrication Conference, Vol. 2003. Society for Imaging Science and Technology, 511–515.
[2] Orhan Bulan, Junwen Mao, and Gaurav Sharma. 2009. Geometric distortion signatures for printer identification. In Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on. IEEE, 1401–1404.
[3] Pei-Ju Chiang, N. Khanna, A. Mikkilineni, M.V.O. Segovia, Sungjoo Suh, J. Allebach, G. Chiu, and E. Delp. 2009. Printer and scanner forensics. IEEE Signal Processing Magazine 26, 2 (March 2009), 72–83. https://doi.org/10.1109/MSP.2008.931082
[4] Jung-Ho Choi, Hae-Yeoun Lee, and Heung-Kyu Lee. 2013. Color laser printer forensic based on noisy feature and support vector machine classifier. Multimedia Tools and Applications 67, 2 (Nov. 2013), 363–382. https://doi.org/10.1007/s11042-011-0835-9
[5] M. Uma Devi, C. Raghvendra Rao, and M. Jayaram. 2014. Statistical Measures for Differentiation of Photocopy from Print technology Forensic Perspective. International Journal of Computer Applications 105, 15 (2014). http://search.proquest.com/openview/5acf04697ebccb072c71b71e5ab30c47/1?pq-origsite=gscholar
[6] DFKI. 2015. Datasets for Document Analysis. (2015). http://madm.dfki.de/downloads 03/08/2017.
[7] Electronic Frontier Foundation. 2005. DocuColor Tracking Dot Decoding Guide. (2005). https://w2.eff.org/Privacy/printers/docucolor/index.php#program 21/07/2017.
[8] Sara Elkasrawi and Faisal Shafait. 2014. Printer Identification Using Supervised Learning for Document Forgery Detection. IEEE, 146–150. https://doi.org/10.1109/DAS.2014.48
[9] Maya Embar, Louis F. McHugh IV, and William R. Wesselman. 2014. Printer watermark obfuscation.. In RIIT, Becky Rutherfoord, Lei Li, Susan Van de Ven, Amber Settle, and Terry Steinbach (Eds.). ACM, 15–20. http://dblp.uni-trier.de/db/conf/riit/riit2014.html#EmbarMW14; http://doi.acm.org/10.1145/2656434.2656437
[10] Stephan Escher and Thorsten Strufe. 2017. Robustness Analysis of a passive printer identification scheme for halftone images. In IEEE International Conference on Image Processing (ICIP).
[11] Anselmo Ferreira, Luiz C. Navarro, Giuliano Pinheiro, Jefersson A. dos Santos, and Anderson Rocha. 2015. Laser printer attribution: Exploring new features and beyond. Forensic Science International 247 (Feb. 2015), 105–125. https://doi.org/10.1016/j.forsciint.2014.11.030
[12] Lukas Gal, Michaela Belovičová, Michal Ceppan, Michal Oravec, and Miroslava Palková. 2013. Analysis of Laser and Inkjet Prints Using Spectroscopic Methods for Forensic Identification of Questioned Documents. In Symposium on Graphic Arts, Vol. 10.
[13] Matthew D. Gaubatz and Steven J. Simske. 2009. Printer-scanner identification via analysis of structured security deterrents. In Information Forensics and Security, 2009. WIFS 2009. First IEEE International Workshop on. IEEE, 151–155. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5386463
[14] Johann Gebhardt, Markus Goldstein, Faisal Shafait, and Andreas Dengel. 2013. Document Authentication Using Printing Technique Features and Unsupervised Anomaly Detection. IEEE, 479–483. https://doi.org/10.1109/ICDAR.2013.102
[15] Hardik Jain, Gaurav Gupta, Sharad Joshi, and Nitin Khanna. 2017. Passive Classification of Source Printer using Text-line-level Geometric Distortion Signatures from Scanned Images of Printed Documents. arXiv preprint arXiv:1706.06651 (2017).
[16] Weina Jiang, Anthony TS Ho, Helen Treharne, and Yun Q Shi. 2010. A novel multi-size block Benford's law scheme for printer identification. In Pacific-Rim Conference on Multimedia. Springer, 643–652.
[17] Do-Guk Kim, Jong-Uk Hou, and Heung-Kyu Lee. 2017. Learning deep features for source color laser printer identification based on cascaded learning. arXiv preprint arXiv:1711.00207 (2017).
[18] Do-Guk Kim and Heung-Kyu Lee. 2014. Color laser printer identification using photographed halftone images. In Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European. IEEE, 795–799. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6952258
[19] Herbert Klimant, Rudi Piotraschke, and Dagmar Schönfeld. 2006. Informations- und Kodierungstheorie (3 ed.). Teubner.
[20] Aravind K. Mikkilineni, Pei-Ju Chiang, George TC Chiu, Jan P. Allebach, and Edward J. Delp. 2007. Channel model and operational capacity analysis of printed text documents. In Electronic Imaging 2007. International Society for Optics and Photonics, 65051U–65051U. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=1298973
[21] CNN Money. [n. d.]. Fortune Global 500. ([n. d.]). http://fortune.com/global500/list 31/08/2017.
[22] Thong Q. Nguyen, Yves Delignon, Lionel Chagas, and François Septier. 2014. Printer identification from micro-metric scale printing. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 6236–6239.
[23] John Oliver and Joyce Chen. 2002. Use of signature analysis to discriminate digital printing technologies. In NIP & Digital Fabrication Conference, Vol. 2002. Society for Imaging Science and Technology, 218–222.
[24] Ricoh. [n. d.]. Going global, Company History about Ricoh. ([n. d.]). http://www.ricoh.com/about/company/history/2000_2009; http://www.ricoh.com/about/company/history/1985_1999 31/08/2017.
[25] Marco Schreyer, Christian Schulze, Armin Stahl, and Wolfgang Effelsberg. 2009. Intelligent Printing Technique Recognition and Photocopy Detection for Forensic Document Examination.. In Informatiktage, Vol. 8. 39–42. https://www.researchgate.net/profile/Sebastian_Magnus/publication/221388729_Ein_Rahmenwerk_fur_Genetische_Algorithmen_zur_Losung_erweiterter_Vehicle_Routing_Problems_VRPSPDMUTW/links/0a85e52fca0dcedffb000000.pdf#page=40
[26] Shize Shang, Nasir Memon, and Xiangwei Kong. 2014. Detecting documents forged by printing and copying. EURASIP Journal on Advances in Signal Processing 2014, 1 (2014), 1–13. http://link.springer.com/article/10.1186/1687-6180-2014-140
[27] S. Suzuki and K. Abe. 1985. Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30, 1 (1985), 32–46.
[28] Małgorzata Szafarska, Renata Wietecha-Posłuszny, Michał Woźniakiewicz, and Paweł Kościelniak. 2011. Application of capillary electrophoresis to examination of color inkjet printing inks for forensic purposes. Forensic Science International 212, 1-3 (Oct. 2011), 78–85.
[29] Min-Jen Tsai, Chien-Lun Hsu, Jin-Sheng Yin, and Imam Yuadi. 2016. Digital forensics for printed character source identification. In Multimedia and Expo (ICME), 2016 IEEE International Conference on. IEEE, 1–6. http://ieeexplore.ieee.org/abstract/document/7552892/
[30] Min-Jen Tsai, Jung Liu, Chen-Sheng Wang, and Ching-Hua Chuang. 2011. Source color laser printer identification using discrete wavelet transform and feature selection algorithms. In Circuits and Systems (ISCAS), 2011 IEEE International Symposium on. IEEE, 2633–2636. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5938145
[31] Jason Tuohey. 2004. Government Uses Color Laser Printer Technology to Track Documents. PCWorld (nov 2004). https://www.pcworld.com/article/118664/article.html 25/09/2017.
[32] United States Secret Service. 2012. Re: Freedom of Information Act Appeal. (Feb. 2012). https://www.scribd.com/doc/81897582/microdots-pdf 25/09/2017.
[33] Joost van Beusekom, Faisal Shafait, and Thomas M. Breuel. 2013. Automatic authentication of color laser print-outs using machine identification codes. Pattern Analysis and Applications 16, 4 (Nov 2013), 663–678.
[34] Changyou Wang, Xiangwei Kong, Shize Shang, and Xin'gang You. 2013. Photocopier forensics based on arbitrary text characters, Adnan M. Alattar, Nasir D. Memon, and Chad D. Heitzenrater (Eds.). 86650G. https://doi.org/10.1117/12.2005524
[35] Yubao Wu, Xiangwei Kong, Xin'gang You, and Yiping Guo. 2009. Printer forensics based on page document's geometric distortion. In Image Processing (ICIP), 2009 16th IEEE International Conference on. IEEE, 2909–2912.
[36] Luo Xiao, Qinghu Chen, and Yuchen Yan. 2015. Printed Characters Texture Identification Based on Two-factor Analysis. Journal of Computational Information Systems 14, 11 (2015), 5199–5207. https://doi.org/10.12733/jcis14760