Security On-Ramp for Variable Data Printing

Steven J Simske, Marie Vans; Hewlett-Packard Labs; Fort Collins, CO, USA

Abstract

Security printing jobs incorporate variable data into explicit regions, called deterrents, which can be read later, usually by a visible spectrum imager (e.g. scanner or camera). In order to initiate a security printing job, a number of authorization steps must be performed. These include the press operator entering the username, password and, possibly, biometric information. Other identification information includes the press identifier (serial number, MAC address, etc.), timestamp, job identification number, lot number (if appropriate), SKU, manufacturer ID, etc. All of these physical security data can be used to drive aspects of the variable data printing (VDP) associated with the security VDP job. This physical security data is digested (trimmed to uniform lengths for each of the input fields, exceptions handled, etc.), concatenated into a single binary string, and digitally signed as necessary to produce the desired security and/or string length. This binary string now represents the physical security data. Next, the binary string is used to drive the security VDP data. This paper will elaborate on several approaches—scrambling, hashing, encryption, sequential XOR—used to convert physical security strings into hybridized security VDP data. The advantages of this on-ramp approach to preventing spoofing of security VDP jobs, and its advantages in brand protection and anti-counterfeiting, are then discussed.

Security Variable Data Printing (SVDP)

Variable data printing (VDP), usually through the use of digital printing technologies—provides the capability of making every printed item the carrier of *explicitly* unique information. An example of explicitly unique information is a set of mass serialized barcodes, with a different barcode on each printed item. This is different from the carrying of *implicitly* unique information, such as the generally forensically-friendly interaction of ink and substrate during printing—which can be read using specialized high-resolution imagers [1].

Security VDP (SVDP) therefore writes an explicit stream of variable data into defined regions, called deterrents [2]. Deterrents are organized to be read later, usually by a visible spectrum imager (e.g. scanner or camera). The effect of the printing and scanning can be modeled and the deterrents altered to anticipate this effect, an approach called pre-compensation [3]. In order to determine what data goes into these one or more deterrents, an input stream of binary information is required. This stream can be derived cryptographically, serially, or by other means, and serves as the input for the information in one of the security deterrents.

Next, in order to initiate the overall security printing job, a number of authorization steps are performed. These steps can include the following: (1) the press operator entering the username, password and, possibly, biometric information; (2) the press identifier (serial number, MAC address, etc.); (3) a timestamp; (4) a job identification number; (5) a lot number (if appropriate); (6) SKU; and (7) manufacturer ID.

All of these physical security data can be used to drive aspects of the variable data printing (VDP) associated with the security VDP job. The security printing ecosystem requires security VDP data for multiple roles. Variable 1D and 2D barcodes, preferably GS1 GTIN compliant, are used for point of sale and mobile commerce applications. Complex security features—such as copy-evident deterrents, guilloches, microtext, void pantographs, and color barcodes—can be used for additional security purposes, such as mass serialization, authentication and forensics. In turn, these deterrents can be used for investigations, evidence gathering and subsequent prosecution. We now cover these concepts in more depth.

GS1 SGTIN (Serialized Global Trade Identification Number)

GS1 global traceability standards specify how data is encoded in RFID and barcodes to allow for automated supply chain workflows, including product recall [4]. Accordingly, prominent SVDP deterrents should carry Serialized Global Trade Identification Number (SGTIN) data. The GS1 SGTIN-96 standard consists of:

(1) The header, which is 8 bits.
(2) The filter, which is 3 bits, specifying if the tagged object is an item, case or pallet.
(3) The partition, which is 3 bits, indicating how the remaining fields are partitioned, allowing their data to be recovered and interpreted correctly.
(4) The company prefix, which is 20-40 bits (depending on the partition bit specification), containing the company's EAN.UCC Company Prefix.
(5) The item reference, which is 4-24 bits (depending on the partition bit specification), containing the item's GTIN item reference number.
(6) The serial number, which is 38 bits, and contains the item's unique serialized data, often designed for randomness in the mass (mass serialization).

The URI (Uniform Resource Identifier) representation of such an SGTIN may be, for example, urn:epc:tag:sgtin-96:3.0037000.06542.837201171, which decodes with the following meaning: the tag is an SGTIN-96 tag that has a Filter value of 3 (shipping unit), a Company Prefix of 0037000 (Proctor & Gamble), an Item Reference of 06542 (Bounty ® Paper Towels 15 Pack) and a Serial Number of 837201171, which uniquely disambiguates that item from others of the same type.

The SGTIN is described here as it is readily deployed in advanced security VDP workflows, as described in the next sections.

Guilloche and Color Tile SVDP Deterrents

Two deterrents we will use in the examples below are shown in Figure 1. On the left is an example of the guilloche deterrent used, which consists of families of polar curves in cyan, yellow and magenta and encodes 64 bits of information along with a checkbit (max capacity 65 bits). These bits are encoded from the type of curve, the color, and the location. On the right is a color tile based deterrent [3], which uses 8 non-data carrying, color and orientation calibrating, tiles in the upper left and lower right corners (Figure 2) along with 56 data-carrying bits. The deterrent thus holds a maximum capacity of 144 bits of data, or more than twice that of the guilloche.

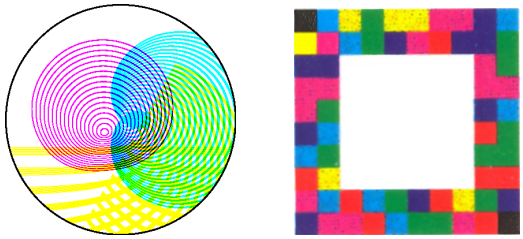

***Figure 1**.Sample SVDP marks. The guilloche mark (left) contains up to 65 bits of information and the tile deterrents mark (right) contains up to 145 bits of information.*

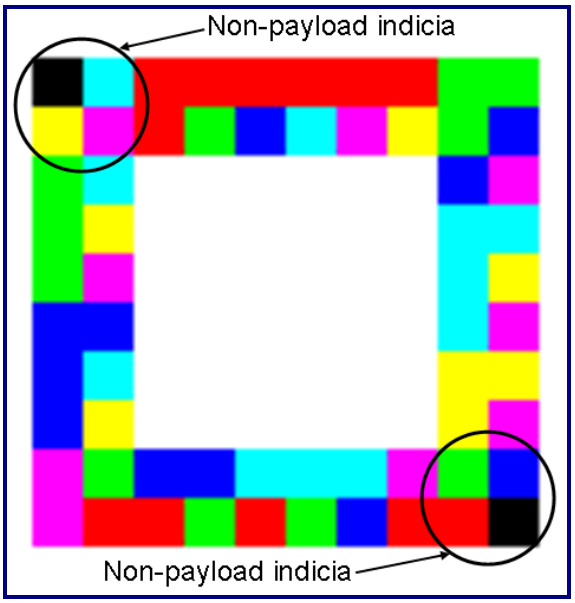

***Figure 2**. Color tile deterrent as described in the examples. Note that there are 8 non-payload indicia (tiles with no data carrying capacity) in the upper left {Black, Cyan, Yellow and Magenta, in reading order} and in the lower right {Green, Blue, Red and Black, in reading order} used for orientation and color calibration.*

Distributed SGTIN

The simplest manner in which to incorporate security information is to distribute the security data among several SVDP marks. For the SGTIN, this may correspond to the use of the above two deterrents in coordination as described here (note, of course, that the color tile itself can carry the complete 96 bits in the SGTIN described above, along with 50% error-correcting code, if so desired):

(1) A guilloche mark (Figure 1, left), which contains 64 bits and a check bit. The 8 bits in the header field are represented 8 times through the use of a scrambling algorithm. The easiest possible scrambling technique can use a 3-bit signal to decide on which digit in the original 8 bits in the header field to begin a cycle. For example, if the 8 bit header is {11010001}, and the 3-bit signal is {011}, then the eight sequential cycles represented in the 64-bit guilloche are {10001110}, {00011101}, {00111010}, {01110100}, {11101000}, {11010001}, {10100011}, and {01000111}.
(2) The 3-bit filter specifies which of 8 different scrambling approaches to use on the guilloche mark. In the above example, {011} indicates to start at the "3" index (where "0" is the first index), underlined here {110<u>1</u>0001}.
(3) The partition is also 3 bits, and may be stored in the carrier frequencies of the magenta, cyan and yellow channels (1 bit/color channel) of the color tile marks shown in Figure 1 (right). This is tied to a copy-prevention effect.
(4) The company prefix, which is in this example 28 bits, is encoded in some of the payload indicia (tiles) of the color tile mark in Figure 2. This color tile deterrent stores $ln(6)/ln(2) = 2.585$ bits/tile, so that in 56 tiles, it holds 144 bits maximum. We could use 14 tiles to directly encode these 28 bits using a reduced [4-color] palette. Alternatively, we can use the information in the guilloche encoding to determine which set of four colors to use, sequentially, for each tile. Regardless, even with this reduced approach, we have 42 tiles remaining for the last two parts of the SGTIN.
(5) The item reference, which is in this example 16 bits, is encoded in another 8 tiles as described for (4).
(6) The serial number, which is 38 bits, is encoded in another 19 tiles as described in (4).

After using up 41 tiles for the company prefix, the item reference, and the serial number, another 15 tiles still remain. We could use these, for example, to perform "power of 2" checkbits to the latter three fields—e.g. 5 for the company prefix, 3 for the item reference and 7 for the serial number. In this example, we used rudimentary encoding approaches to print the 96-bit SGTIN into two independent printed deterrents. In the next step, we build on this to incorporate physical security information into the VDP approach used.

Physical Security

Previously, we have performed threat analyses of security printing systems. Among our recommendations were the need for real-time logging and the need for relating the physical security to the security VDP job performed. Both of these require the physical security information to be securely stored. For the latter, however, *hybridized* security VDP—the establishing of a secure relationship between the data embedded in two or more security features—is recommended.

Important physical security fields are shown in Figure 3. These include information associated with the print job—username and password of press operator, machine ID, timestamp, product

information (stock keeping unit, lot identification, manufacturing identifier, print job number, and even biometric information tied to the press operator. Several of these fields are also part of the SGTIN described above.

***Figure 3**. Physical security field examples. Associated with the print job are the username and password of the press operator, the machine ID, the time stamp of when the job was initiated, product information (stock keeping unit [SKU], lot ID and/or manufacturing number), print job number and even biometrics such as fingerprint validation.*

Linking Physical Security to VDP: Security VDP

Hybridization of the physical security to the variable data printing proceeds as follows. The physical security data is digested (trimmed to the appropriate lengths for each of the input fields, exceptions handled, etc.), concatenated into a single binary string, and digitally signed as necessary to produce the desired security and/or string length. This binary string now represents the physical security data, and is denoted BS_{PSD}. If digitally signed, the original fields can only be recovered if the creator's private key is available. This is the preferred security approach, since it introduces no new security risk over the risk also incumbent with the existence of private keys.

Next, the BS_{PSD} is used to drive the security VDP data. As an example, suppose the barcode data comprises {10111000} and the first eight bits of the binary string are {11101001}. Then, a second deterrent—for example microtext characters—could be defined by the XOR of the first eight bits of the string with the barcode data, or {01010001}. If a simple hexadecimal set of characters are printed, this translates into a microtext string of "51". The next eight bits of the binary string, say {00111010}, can then be XOR'ed with the microtext-generating string to produce the data in another deterrent, or {01101011}. This mechanism—chained XORing, is generally useful for preserving the entropy in the original BS_{PSD}.

In general, any of dozens of encoding approaches can be used with the BS_{PSD} to produce the printed security information. If the encoding approach is followed by encryption, then the output string used to write to the security deterrents will have high entropy (i.e., randomness commensurate with the strength of the encryption method). Why does encoding occur before encryption? Because in some cases the amount of variability (usually expressed as entropy) in the input fields (Figure 3) is low enough that successful attacks can be made on the encrypted data otherwise.

The information in the BS_{PSD} can be used to link the deterrents together in a slightly more complicated manner. In this scenario, the first deterrent is encoded with raw data (or otherwise scrambling data) from the BS_{PSD}, and the second deterrent is derived from the data in the first deterrent using the subsequent information in the BS_{PSD} to derive one or more of the following:

(1) As the nonce for the XOR of the previous deterrent with the next deterrent (as described above for the microtext). This is described in shorthand by $N_{XOR}(L)$; that is, use length L of bits as a nonce for XOR the previous L bits.
(2) As a key on appropriate length L_K for a CSA (Common Switching Algorithm) to encode the next deterrent of length L. This is described in shorthand $Key_{CSA}(L_K,L)$.
(3) As coefficients to a shift register used to encode the previous deterrent. This is defined in shorthand as $SR(L_p,L)$, where L is the length of the next deterrent derived from the shift register, and L_p is the length of previous bits used for the settings of the shift register.
(4) As a specific code for switching between any/all of the previous three approaches on the fly.

When used in this way, the binary (bit) stream BS_{PSD} can thus provide any desired level of complexity to the relationship between deterrents in the print job. Importantly, however, the approach is readily conveyed in shorthand: e.g. if we choose method (4), we might describe the encoding of 4 deterrents, with 38, 144, 10 and 48 variable bits, as:

$$BS_{PSD}38 \rightarrow N_{XOR}(38) \rightarrow Key_{CSA}(64,144) \rightarrow SR(6,10) \rightarrow N_{XOR}(48)$$

This scheme requires 198 bits of data to encode 240 bits of data in the deterrents. An agent in the field can check a package, label or document solely by checking that:

$$N_{XOR}(38) \rightarrow Key_{CSA}(64,144) \rightarrow SR(6,10) \rightarrow N_{XOR}(48)$$

occurs correctly, even if she has no access to the original BS_{PSD}. Note that this shorthand would need to be accompanied by information on interpreting the settings and on which deterrents each step is applied to. This does not, however, limit its applicability to an off-line check of a printed item.

Extension to Mass Serialization

Mass serialization (Figure 4) is the process by which each item in a set of printed items (labels, packages, documents, etc.) is assigned a unique identifier, designated a unique binary string ID, or BS_{IDU}. The simplest means to achieve this is to assign the numbers {0,1,…,N-1} where N=the number of items to be assigned a BS_{IDU} to the set S. Let L=length of each binary string. For security purposes, N << L (generally, • L/2) so that the expected value of the Hamming Distance (HD) between any two mass serialized items (Equation 1, where BS(A,*) is location * in the array of BS(A)) is sufficient to prevent guessing of legitimate mass serialized strings.

$$\text{HD} = \sum_{i=0}^{N-1} XOR(BS(A,i), BS(B,i)) \qquad (1)$$

Generally, this requirement is easily achieved. With SGTIN, the serialization length L=38, and so for N • L/2 to hold we can use N=19, which allows us to use 2^{19}, or more than half a million, printed items with the odds of guessing a correct sequence also less than one in half a million. The mean HD should also be roughly L/2, or 19.

Figure 4*. Example section of a mass serialized set of binary strings. Note that the Hamming Distance (HD) between each string can be determined simply from the sum in Equation 1. HD is 10 in comparing the partial strings in the first and second row.*

Generating a mass serialized data set, then, requires no more than applying the following steps to each of the items printed at one time using the physical security to create the SVDP information:

(1) Create the original binary string for the physical security information, BS_{PSD}.
(2) Write the current number (starting with index=0) in series for the printed item in the desired number of bits for the mass serialized; e.g. for the 1,777th item in a 38 bit serialization:

{00000000000000000000000000011011110001}

(3) Apply the appropriate encoding algorithm to the string, e.g. $BS_{PSD}38$ where the first 38 bits of BS_{PSD} are:

{10001101011010100111011100100100111110}

(4) Create the mass serialized string from the appropriate encoding shorthand; e.g. $BS_{PSD}38 \rightarrow N_{XOR}(38)$.

{10001101011010100111011100111111001111}

(5) Encode the mass serialized data in (4) into the correct security printing deterrent.
(6) Print the deterrent as part of the VDP job.

Note that when using this approach, the entropy of the mass serialized identifier is primarily dependent on the entropy of the BS_{PSD} (i.e. the first 27 of 38 bit positions simply repeat BS_{PSD}). If the entropy of the BS_{PSD} strings is insufficient, this will be reflected in the mean HD of the mass serialized set.

Conclusions

This paper describes how physical security information can be incorporated into a variable data printing job to provide the desired level of per-item and per-job security. This is advantageous as shown not only for authentication of individual items, but also for validation of items when the original binary string is not available to the agent (e.g. off-line, outside secured environment, etc.) In this case, the agent can check if the relationship between the security deterrents is correct, which can provide a confidence level nearly that of having the original bit string—i.e. 202 of the 240 bits in the example given above—without compromising any of the original bit string.

This paper shows how many different encoding (scrambling, nonce, encryption, etc.) approaches provide a more dynamic SVDP environment, allowing the same deterrents to be used in perpetuity while still providing a "moving target" for would-be counterfeiters. That is, the hybridization approaches and linkages can be changed on the fly without changing the layout or aesthetics of the printing. The advantages of this on-ramp approach to preventing spoofing of security VDP jobs, and its advantages in brand protection and anti-counterfeiting, are obvious. Training can be readily provided to all parties involved in authenticating products, and the training does not need to be completely overhauled when the security of the overall printing is compromised, since only the relationship between the variably printed items need be changed in response to this compromise.

Finally, the broad applicability of the approach to mass serialization is outlined. The link between physical security and mass serialization is obvious—the serialized codes can be directly traced back to their creation (creator, press, time and location) for auditing and other (repudiation, legal, etc.) purposes.

Acknowledgements

The authors gratefully acknowledge many gifted collaborators, especially Juan Carlos Villa and Pipo Caban.

References

[1] S. J. Simske and G. Adams, "High-resolution glyph-inspection based security system", Proc. IEEE ICASSP, pp. 1794-1797, 2010.
[2] S. J. Simske, J. S. Aronoff, M. M. Sturgill, and G. Golodetz, "Security Printing Deterrents: A Comparison of Thermal Ink Jet, Dry Electrophotographic, and Liquid Electrophotographic Printing," Jour. Imaging. Sci. and Technol., 52(5), pg. 50201, 2008.
[3] S.J. Simske, M. Sturgill, and J.S. Aronoff, "Effect of Copying and Restoration on Color Barcode Payload Density," Proc. ACM DocEng, vol. 9, pp. 127-130, 2009.
[4] GS1 Global Traceability Standard (GTS), http://www.gs1.org/traceability/gts, last accessed on 22 June 2010.

Author Biography

Steven Simske is a Distinguished Technologist in the Print Production Automation Lab (PPAL) in Hewlett-Packard Labs, and is the Director and Chief Technologist for the HP Labs Security Printing and Imaging program. Security printing and imaging provides brand differentiation, brand protection and anti-counterfeiting through novel security algorithms, printing approaches and image analysis technologies. Steven has 35 US patents and nearly 250 peer-reviewed conference and journal publications.