(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
05 March 2020 (05.03.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/046282 A1

(51) International Patent Classification:
G06F 3/12 (2006.01) *G09C 3/00* (2006.01)
G09C 1/06 (2006.01)

(21) International Application Number:
PCT/US2018/048347

(22) International Filing Date:
28 August 2018 (28.08.2018)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 10300 Energy Drive, Spring, Texas 77389 (US).

(72) Inventors: **SCHIFFMAN, Joshua Serratelli**; Filton Road Stoke Gifford, Pt., Long Down Ave, Stoke Gifford, Bristol BS34 8QZ (GB). **LAING, Thalia**; Filton Road Stoke Gifford, Pt., Unit 700 The Quadrant, Bradley Stoke, Bristol BS34 8QZ (GB). **WATTIAU, Gaetan**; Filton Road Stoke Gifford, Pt., Long Down Ave, Stoke Gifford, Bristol BS34 8QZ (GB).

(74) Agent: **WOODWORTH, Jeffrey C.**; HP Inc., 3390 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:
— *as to the identity of the inventor (Rule 4.17(i))*
— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:
— *with international search report (Art. 21(3))*

(54) Title: ENCRYPTING DATA

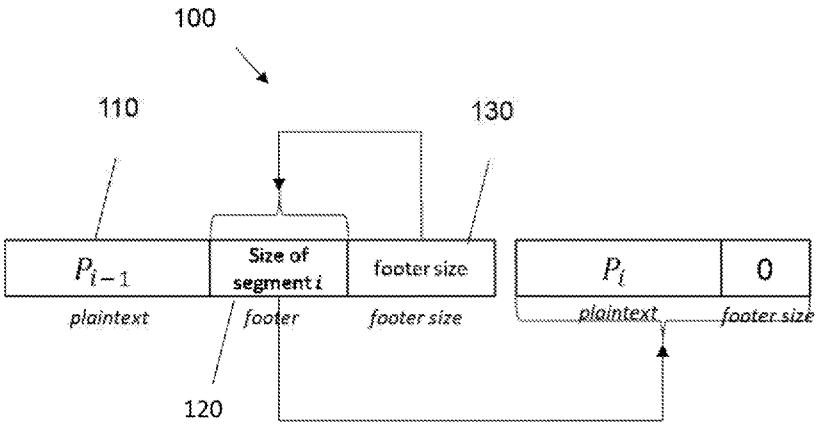


FIG. 1

(57) Abstract: There is disclosed a method for encrypting data representing a rendering task, the method comprising segmenting the data to form multiple variably sized segments, wherein each segment comprises a payload, and a footer portion comprising at least a footer size section indicating the size of a footer encoding the size of a subsequent segment, and encrypting each segment using data associated with that segment and the rendering task.

WO 2020/046282 A1

ENCRYPTING DATA

BACKGROUND

[01] Authenticated encryption is used to protect print job data confidentiality and integrity. Bulk encryption of a print job can be used, where a client can encrypt an entire job before sending it to a printer, then the printer can decrypt and verify the entire job before printing.

BRIEF DESCRIPTION OF THE DRAWINGS

[02] Various features and advantages of certain examples will be apparent from the detailed description which follows, taken in conjunction with the accompanying drawings, which together illustrate, by way of example, a number of features, and wherein:

[03] Figure 1 shows the structure of a PRinterMAC segment according to an example;

[04] Figure 2 shows PRinterMAC encryption according to an example;

[05] Figure 3 shows an encryption process according to an example;

[06] Figure 4 shows a whole encryption process according to an example;

[07] Figure 5 shows a decryption process according to an example;

[08] Figure 6 shows a method for encrypting data representing a rendering task according to an example; and

[09] Figure 7 shows a processor associated with a memory comprising computer readable instructions for executing a method for encrypting data representing a rendering task according to an example.

DETAILED DESCRIPTION

[010] In the following description, for purposes of explanation, numerous specific details of certain examples are set forth. Reference in the specification to "an example" or similar language means that a particular feature, structure, or characteristic described in connection with the example is included in at least that one example, but not necessarily in other examples.

[011] Authenticated encryption schemes both encrypt data for confidentiality and provide a message authentication code (MAC) to verify data integrity. A MAC tag can be computed over the whole decryption process before it can be validated. Thus, a receiver may wait to receive and process an entire ciphertext before checking the authenticity of the data. In a printing context, the delay to verify delays the time to print until the entire job is decrypted and verified. For large jobs, this delay can have significant impact on first page out (FPO) performance, which may lead to users disabling encryption or using a less secure mode. This problem is also relevant to any data flows to and from printers (e.g. scanning) that are processed immediately.

[012] A data segmentation scheme called interMAC is known that inserts MAC tags into a data stream to protect against adversaries that could re-order, truncate, or otherwise trick the recipient into receiving infinitely long messages. For example, interMAC uses a fixed segment size, agreed upon by both parties ahead of time, to achieve boundary hiding of the segment size from outside parties. However, this size may be discoverable since a common value may be used for performance. In addition, the interMAC scheme assumes that the data is a multiple of the segment size and can use padding of the data if it is not.

[013] This disclosure describes a construction called “PRinterMAC” that splits a print job into smaller segments that can be transmitted and decrypted independently and immediately. An encryption and decryption function are described. There is provided a method for applying authenticated encryption to large print jobs that enables immediate transmission at the sender and immediate processing at the receiver after the first segment is processed. The construction breaks the data into smaller segments (potentially of varying size) and applies authenticated encryption to the segments individually.

[014] Figure 1 shows the structure of a PRinterMAC segment according to an example. A segment may comprise a payload section 110 and a footer portion 120, 130. As shown, a segment 100 may comprise: the payload or plaintext section 110; a footer section 120; and a footer size section 130. The footer portion may comprise a footer section 120 and a footer size section 130. The plaintext section (P_{i-1}) 110 contains the data to be encrypted. The footer section 120, if present, contains the size of the next segment (l_i). The footer size section 130 contains the size of the footer section 120 or other metadata.

[015] The footer size section 130 can be fixed so that a recipient knows how much data to parse, for example, a 1-byte long section. This value is used to encode information such as: the size of the next segment (l_i); whether the stream is finished; whether a logical portion is finished; or whether the next segment (l_i) is a dummy value.

[016] According to an example, a first or initialization segment can be used to bootstrap the data stream. For example, the initialization segment can be a fixed size segment that contains no plaintext data but is used to indicate the size of the second segment.

[017] Each segment 100 comprises a footer 120 that indicates the size of the next segment to process or whether the data stream has finished. The sender can start the stream by sending a fixed size initialization segment to indicate to a recipient how large the first segment will be. As each segment 100 is received, the recipient may decrypt and authenticate the segment and immediately process (e.g. print) the data.

[018] According to an example, the segment size can be tuned to provide improved performance (e.g. smaller segments initially and larger later).

[019] According to an example, a strategy can be applied to obfuscate the boundary of the data's contents such as randomizing segment size, inserting dummy segments, or adjusting segments on a contextual level (e.g. per page, per job). This provides boundary hiding which can be useful when knowledge about the size of the data may be meaningful and where encryption alone cannot hide such metadata. According to an example, PRinterMAC enables the

security property of boundary-hiding for practical use cases. As such, segments can be variably sized and their respective sizes can be generated independently, i.e. that is for any i, j and $i \neq j$ the size of $segment_i$ is independent from the size of $segment_j$.

[020] PRinterMAC can generate segments from the plaintext and can encrypt and send these segments separately. Figure 2 describes PRinterMAC encryption according to an example. Each segment may be encrypted using the same key but with different initialisation vectors (IV) and/or different additional authenticated data (AAD). For example, the PRinterMAC construction may use a compound initialization vector construction to provide protection against an adversary that may attempt to re-order the segments, truncate the message, or replay the data.

[021] According to an example, initialization vectors may be generated from three values including a nonce, a message counter and a segment counter. A nonce may relate to a random value that does not change for a session. A message counter may relate to a counter that is incremented after each logical portion or message, i.e. the message counter is incremented after a logical portion is completed. A segment counter may relate to a counter that is incremented after each segment, i.e. each segment may increment the segment counter. A logical portion may be defined as an entire job. Thus, multiple jobs can be included in a single stream and the printer can increment the counter after each one is received. Alternatively, a printed page can act as a logical portion.

[022] According to an example, for segment i, the initialization vector can be generated as: $IV_i = nonce \| message\ counter \| segment\ counter_i$, where $\|$ is the concatenation operation.

[023] According to an example, additional authenticated data relates to a value that is authenticated but not encrypted. The additional authenticated data may be taken as an input to generate a MAC tag. Without the correct additional authenticated data, the ciphertext may not be authenticated. For example, for segment i the additional authenticated data can be generated by: $AAD_i = message\ counter \| segment\ counter_i$.

[024] Generating initialisation vectors and additional authenticated data mitigates attacks that replace one segment with another, since the initialisation vectors and additional authenticated data for two different segments are different. Consequently, the decryption would not otherwise succeed in such an attack.

[025] According to an example, the initialisation vectors and the additional authenticated data are implied by the PRinterMAC construction such that they are predictable by the receiver. As such, the initialisation vectors and the additional authenticated data may not be sent with the ciphertext.

[026] The encryption process will now be described according to the example of Figure 3. A single segment may be encrypted. The segment counter value i may be incremented by one for each segment. Inputs to the encryption process may comprise:

- plaintext data 310 (can be any size) $(P_i)_{i=0,\dots,N}$ for any $N \geq 0$
- the random nonce, n, 320,
- the message counter, m, 330,
- an authenticated encryption cipher AE, 340, and
- a random key for this cipher K, 350.

[027] A MAC tag 360, t_i is associated to the encrypted segment 370 or ciphertext segment c_i. When a segment is encrypted, the segment counter is incremented, then the next generated segment counter plaintext is taken and encrypted (Enc_{AE}) with the initialisation vectors (IV) and the additional authenticated data (AAD).

[028] The whole encryption process is shown in Figure 4 according to an example.

[029] The rendering task may be represented by plaintext 410. The plaintext is received as input and is split into N payloads or plaintext sections 420, 430, 440, 450 (of potentially differing sizes): P_1, P_2, ... P_N. Each of the plaintext sections to be used for constructing N respective segments.

[030] An initialization segment 410 (P_0) is formed first. The initialization segment 410 comprises the initialisation vectors. The initialisation segment may, for example, comprise an eight-byte footer portion 416 comprising a one-byte footer size section 417. As shown, this footer size section 417 is defined to be a "7" in the initialization segment 410, which defines the length of the footer in the current segment 422 as seven-bytes long. The initialization segment 410 is encrypted using authenticated encryption 418 with the message counter, *m*, and segment counter, *i*, 411 input as associated data. A MAC tag 412, (t_0) is associated to the encrypted segment 413 (C_0) or ciphertext segment.

[031] According to an example, the footer size section for each and every segment may be one-byte size.

[032] According to an example, the footer 419 (of the same segment) may be empty if the footer size section has an entry of "0", meaning that the next segment is the same size as the current segment. Similarly, if the footer size section has an entry of "255", the footer (of the same segment) is empty, meaning that the current segment is the final segment and there is no next segment. In other cases, if the footer size segment is any other value (from 1 to 254), the footer section is not empty.

[033] Now that the initialisation segment 410 has been encrypted, the segment counter value (*i*) is incremented. Then, the next (first) segment 422 for the generated segment counter value (i=1) is taken and encrypted (AE) with the initialisation vectors (IV) and the additional authenticated data (AAD).

[034] To construct the first segment 422, the length of the subsequent plaintext section (P_i) 430 (variable), the size of the footer 436 and the fixed-length footer size section (e.g. of one-byte) are considered. If P_i 430 is the same size as the current plaintext section (P_1) 420, then the footer size section 426 (which may itself have a size of one-byte) can be marked as "0" and the footer section 424

will be empty (which is the case in the example shown here). The current plaintext section (P_1) 420, the empty footer section 424 and the "0" footer size section 426 are encrypted using authentication encryption 427. The encryption uses the message counter, *m*, and segment counter value, *i*, 421 (incremented by 1) as associated data. A MAC tag 429, (t_1) is associated to the first encrypted segment 423 (C_1) or ciphertext segment.

[035] This encryption process continues to construct the subsequent, *i*th segment 432, considering the size of the subsequent N-1 segment 442. For example, if the subsequent plaintext section 440 (P_{N-1}) is the same size as the current plaintext section (P_i) 430, the footer size section 436 is "0" and the footer section 434 is empty. However, if the subsequent plaintext section (P_{N-1}) 440 is a different size to the current plaintext section (P_i) 430, the footer size section 436 (associated with P_{N-1}) is defined to show how many bytes the footer is (this is how many bytes are required to define the size of P_{N-1}). The plaintext P_{N-1} 440, the footer section 434 (defining how long P_{N-1} is) and the footer size section 436 (defining how long the footer section is) are then all encrypted using authenticated data 437 (with the message and segment counter 431 as associated data). The encryption uses the message counter, *m*, and segment counter value, *i*, 431 (incremented by 1) as associated data. A MAC tag 439, (t_i) is associated to the *i*th encrypted segment 433 (C_i) or ciphertext segment.

[036] This encryption process continues until, finally, the last segment 452, P_N, is reached and there will be no subsequent segment. In this case, the footer size section 456 value may be "255", indicating that this is the final segment 452 to be constructed, and the footer section 454 is empty. As before, the final segment 452 is encrypted using authenticated data with the message and segment counter 451 as associated data.

[037] Figure 5 shows a decryption process according to an example. Once the PRinterMAC encryption has been defined, the PRinterMAC decryption can occur where the receiver gets the initialization segment and decrypts it. For example, in block 500 the receiver reads the size of the next segment from the footer size section. Then, for each segment: at block 510 the receiver increments the segment counter and generates the corresponding AAD and IV; at block 520 the receiver takes the number of bytes indicated in the previous

segment and bytes for the MAC tag; and at block 530 the receiver examines the footer size field from the end of the segment. The receiver then determines any metadata for processing the next segment. For example, it could indicate that the length of the next segment, the size of the segment does not change, special handling behaviour (e.g. dummy segment) or this is the last segment. At block 540 all of the data bytes before the footer section is decrypted and outputted.

[038] Figure 6 shows a method for encrypting data representing a rendering task according to an example. The method may comprise segmenting the data to form multiple variably sized segments. Each segment may comprise a payload and a footer portion. The footer portion may comprise at least a fixed-length footer size section indicating the size of a footer encoding the size of a subsequent segment. The payload of a segment may be encrypted using data associated with that segment and the task.

[039] According to an example, the PRinterMAC may be implemented using any authenticated encryption scheme such as AES-CCM, AES-GCM, and ChaCha20-Poly1305. The application may extend beyond sending data to a printer, for example, other applications may comprise:

- Scanning data on a printer and sending it to a client for rendering
- 3D print data (3MF files)
- Cloud print workflows that render the data before printing

[040] According to an example, the logical portions may be defined as pages, paragraphs, 3D object layers, color channels, or other delineations that can be processed immediately upon receipt.

[041] Per-packet or per-message encryption schemes may use a combination of encryption and authentication schemes that are subject to manipulation or lack efficiency, where some examples include SSH encryptions, IPsec ESP and AH protection, and encrypted sessions like TLS. The present disclosure improves MAC tag construction and adapts it to a printing context. This is achieved by eliminating negotiated segment lengths and padding. Various strategies for boundary hiding are provided, including randomization and

context aware selection which allows for segments that can be fully processed on their own (such as a fully printable document page). The method provided defines an initialisation vector that works with the segment length strategy.

[042] The PRinterMAC construction can reduce the ciphertext size under some selection of parameters, which improves the efficiency over MAC tag. This improves user experience of authenticated encryption which can otherwise cause noticeable slowdown to the printing experience and otherwise affects the perceived "first page out" (FPO) time and may lead to users disabling encryption. For example, a print job can be encrypted, but the process slows the print process considerably and affects the perceived first page out (FPO) time. In order to maintain confidentiality and integrity whilst improving FPO time, a job may be segmented into multiple segments, each having different sizes from one another in order to prevent boundary observations. The payload or plaintext of each segment can be encrypted and the encoded segment is provided with data providing insight into the next segment of the job in question. The approach described herein enables the printer to readily retrieve a print job without pre-negotiated print segment length information. The methods described are applicable to 3D print jobs as well as 2D printing, i.e. there is provided an encryption scheme suitable for applications over and above a generic rendering task.

[043] Compared to encrypting and then decrypting the print job as one large object, the PRinterMAC construction described herein reduces the first page out time. The construction provides: boundary hiding of print job context, which makes it difficult for attackers to observe or alter the data in a meaningful way (e.g. randomized, jobs, pages, variable, dummy segment or footer to ignore segment); IV generation in the context of print (page and job level counters); and handling of unknown length inputs / payloads (DOS protection).

[044] According to an example, the methods described can be applicable to other rendering tasks, such as scanned jobs (i.e. scanning in the other direction client / cloud compared to printing a job), where the rendering time is reduced for the recipient due to the segmented data encryption scheme. The segmented data encryption scheme may be applied to other data streams beyond rendering tasks.

[045] PRinterMAC enables data security of a rendering task or print job without increasing the time to first page out as the data size grows. The first page out time being reduced enhances customer experience by providing a seamless, efficient printing experience where a customer does not wait at the printer to receive their print job for too long.

[046] Examples in the present disclosure can be provided as methods, systems or machine-readable instructions, such as any combination of software, hardware, firmware or the like. Such machine-readable instructions may be included on a computer readable storage medium (including but not limited to disc storage, CD-ROM, optical storage, etc.) having computer readable program codes therein or thereon.

[047] The present disclosure is described with reference to flow charts and/or block diagrams of the method, devices and systems according to examples of the present disclosure. Although the flow diagrams described above show a specific order of execution, the order of execution may differ from that which is depicted. Blocks described in relation to one flow chart may be combined with those of another flow chart. In some examples, some blocks of the flow diagrams may not be necessary and/or additional blocks may be added. It shall be understood that each flow and/or block in the flow charts and/or block diagrams, as well as combinations of the flows and/or diagrams in the flow charts and/or block diagrams can be realized by machine readable instructions.

[048] The machine-readable instructions may, for example, be executed by a general-purpose computer, a special purpose computer, an embedded processor or processors of other programmable data processing devices to realize the functions described in the description and diagrams. In particular, a processor or processing apparatus may execute the machine-readable instructions. Thus, modules of apparatus may be implemented by a processor executing machine readable instructions stored in a memory, or a processor operating in accordance with instructions embedded in logic circuitry. The term 'processor' is to be interpreted broadly to include a CPU, processing unit, ASIC, logic unit, or programmable gate set etc. The methods and modules may all be performed by a single processor or divided amongst several processors.

[049] Such machine-readable instructions may also be stored in a computer readable storage that can guide the computer or other programmable data processing devices to operate in a specific mode.

[050] For example, the instructions may be provided on a non-transitory computer readable storage medium encoded with instructions, executable by a processor.

[051] Figure 7 shows an example of a processor 710 associated with a memory 720. The memory 720 comprises computer readable instructions 730 which are executable by the processor 710. The instructions 730 comprise:

Instructions to segment the data to form multiple variably sized segments, wherein each segment comprises a payload, and a footer portion, the footer portion comprising at least a fixed-length footer size section indicating the size of a footer encoding the size of a subsequent segment; and

Instructions to encrypt the payload of a segment using data associated with that segment and the task.

[052] Such machine-readable instructions may also be loaded onto a computer or other programmable data processing devices, so that the computer or other programmable data processing devices perform a series of operations to produce computer-implemented processing, thus the instructions executed on the computer or other programmable devices provide an operation for realizing functions specified by flow(s) in the flow charts and/or block(s) in the block diagrams.

[053] Further, the teachings herein may be implemented in the form of a computer software product, the computer software product being stored in a storage medium and comprising a plurality of instructions for making a computer device implement the methods recited in the examples of the present disclosure.

[054] While the method, apparatus and related aspects have been described with reference to certain examples, various modifications, changes, omissions, and substitutions can be made without departing from the spirit of the present

disclosure. In particular, a feature or block from one example may be combined with or substituted by a feature/block of another example.

[055] The word "comprising" does not exclude the presence of elements other than those listed in a claim, "a" or "an" does not exclude a plurality, and a single processor or other unit may fulfil the functions of several units recited in the claims.

[056] The features of any dependent claim may be combined with the features of any of the independent claims or other dependent claims.

CLAIMS

1. A method for encrypting data representing a rendering task, the method comprising:

segmenting the data to form multiple variably sized segments, wherein each segment comprises:

a payload; and

a footer portion comprising at least a footer size section indicating the size of a footer encoding the size of a subsequent segment; and

encrypting each segment using data associated with that segment and the rendering task.

2. A method according to claim 1, wherein the footer portion further comprises a footer encoding the size of a subsequent segment.

3. A method according to claim 1, wherein the payload comprises plaintext.

4. A method according to claim 1, wherein the footer size section has a fixed-length of one-byte.

5. A method according to claim 1, wherein the footer portion defines an amount of data to parse for each respective segment.

6. A method according to claim 1, wherein the footer is empty when the footer size section indicates that the size of the footer encoding the subsequent section is 0 or 255 bytes.

7. A method according to claim 1, wherein each segment is variably sized such that the sizes of respective segments are generated independently of other segments.

8. A method according to claim 1, wherein each segment is generated from plaintext and subsequently encrypted.

9. A method according to claim 1, wherein each segment is encrypted using a different initialisation vector.

10. A method according to claim 9, wherein each initialisation vector generated from a nonce, a message counter, and a segment counter.

11. A method according to claim 1, wherein each segment is encrypted using a different authenticated data value.

12. A method according to claim 1, wherein each segment is sent to a receiver separately from other segments.

13. A non-transitory machine-readable storage medium encoded with instructions executable by a processor for encrypting data representing a

rendering task, the machine-readable storage medium comprising instructions to:

segment the data to form multiple variably sized segments, wherein each segment comprises:

a payload; and

a footer size section indicating the size of a footer encoding the size of a subsequent segment; and

encrypt each segment using data associated with that segment and the rendering task.

14. A storage medium according to claim 13, further comprising instructions to encrypt each segment using a different initialisation vector.

15. A storage medium according to claim 13, further comprising instructions to encrypt each segment using a different authenticated data value.

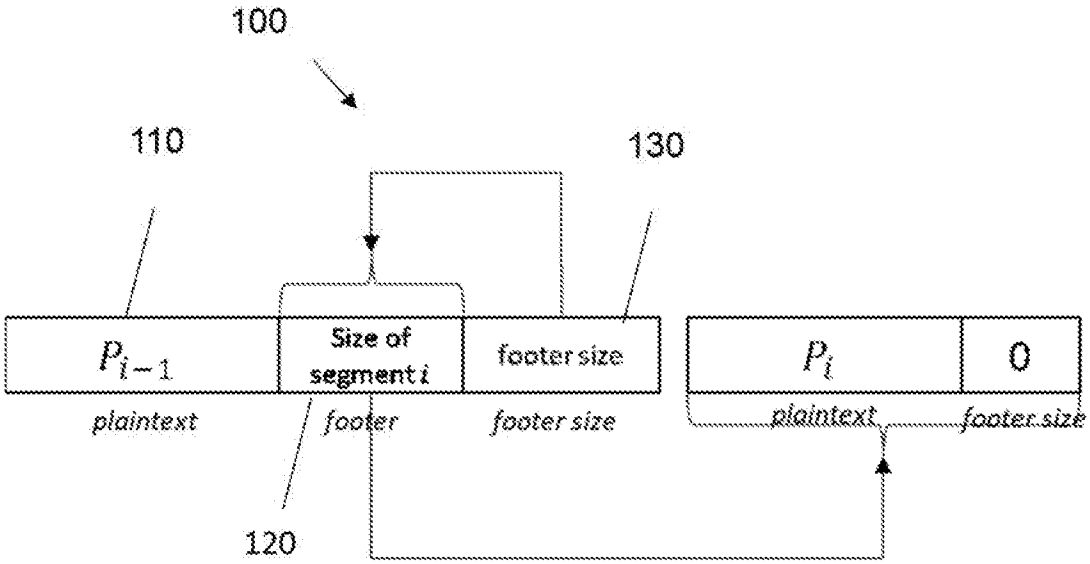

100
110
130
P_{i-1}
Size of segment i
footer size
P_i
0
plaintext
footer
footer size
plaintext
footer size
120

FIG. 1

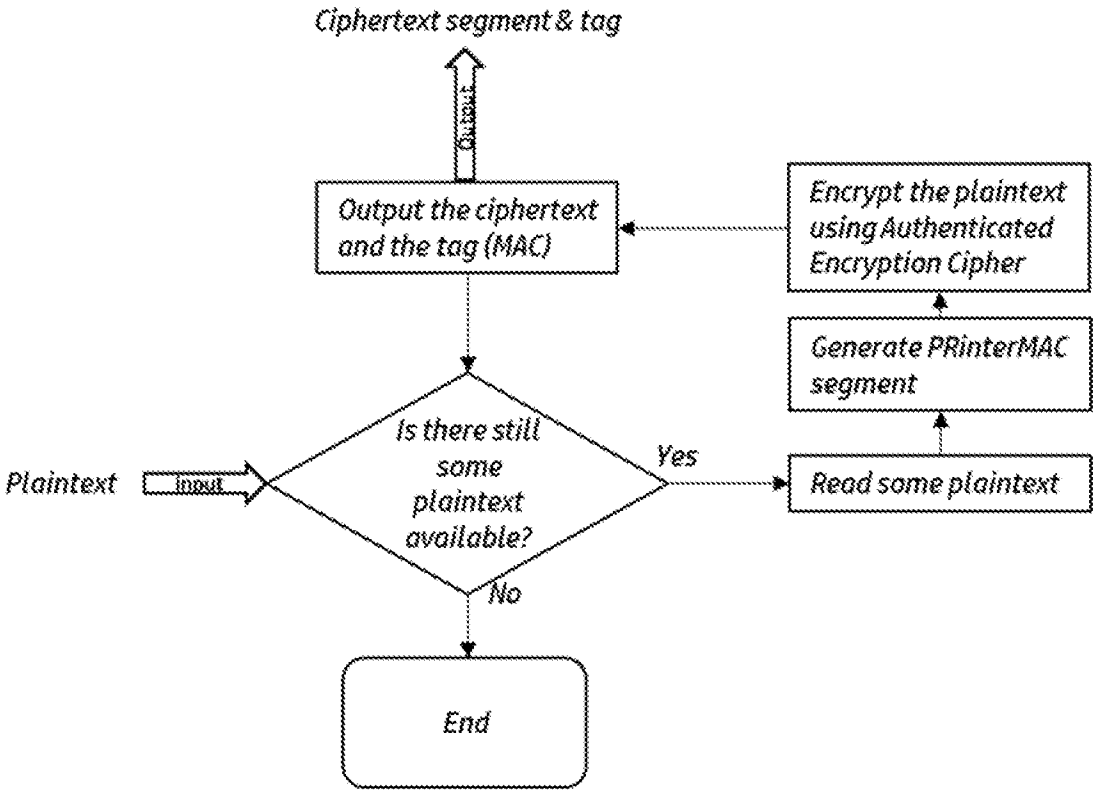

Ciphertext segment & tag
output
Output the ciphertext and the tag (MAC)
Encrypt the plaintext using Authenticated Encryption Cipher
Generate PRinterMAC segment
Is there still some plaintext available?
Yes
Plaintext
input
Read some plaintext
No
End

FIG. 2

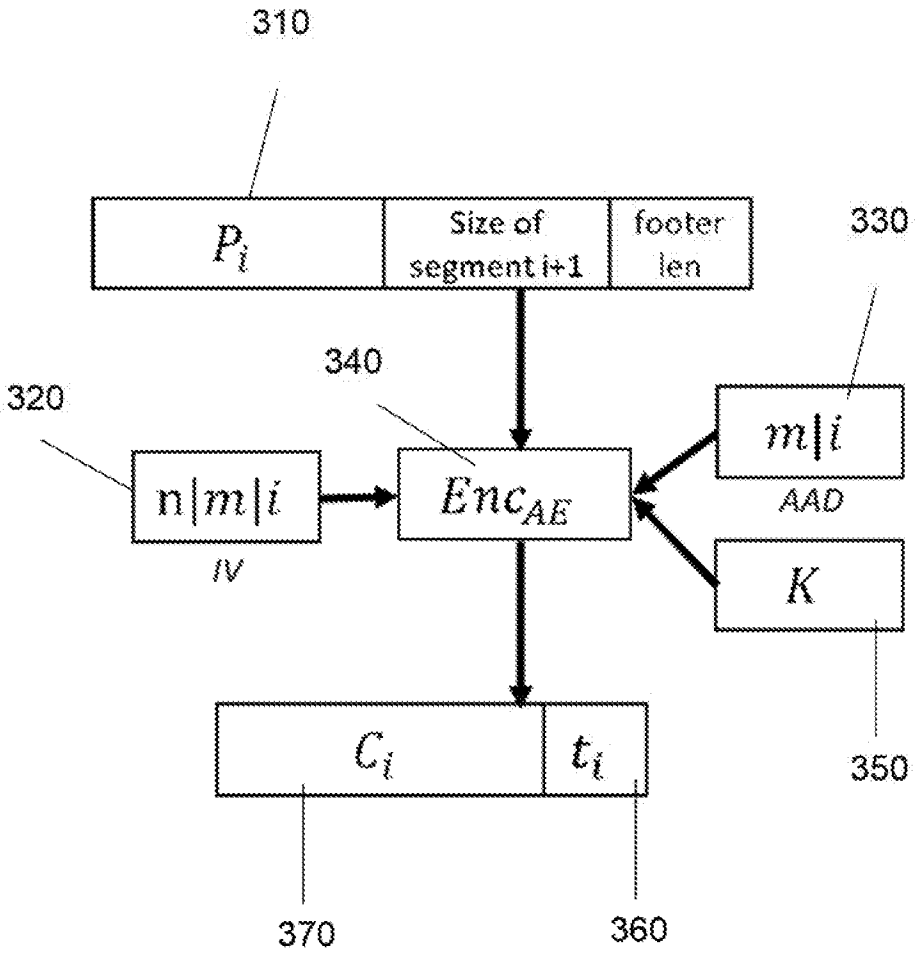

310
P_i
Size of segment i+1
footer len
330
340
320
$m|i$
AAD
n|m|i
IV
Enc_{AE}
K
C_i
t_i
350
370
360

FIG. 3

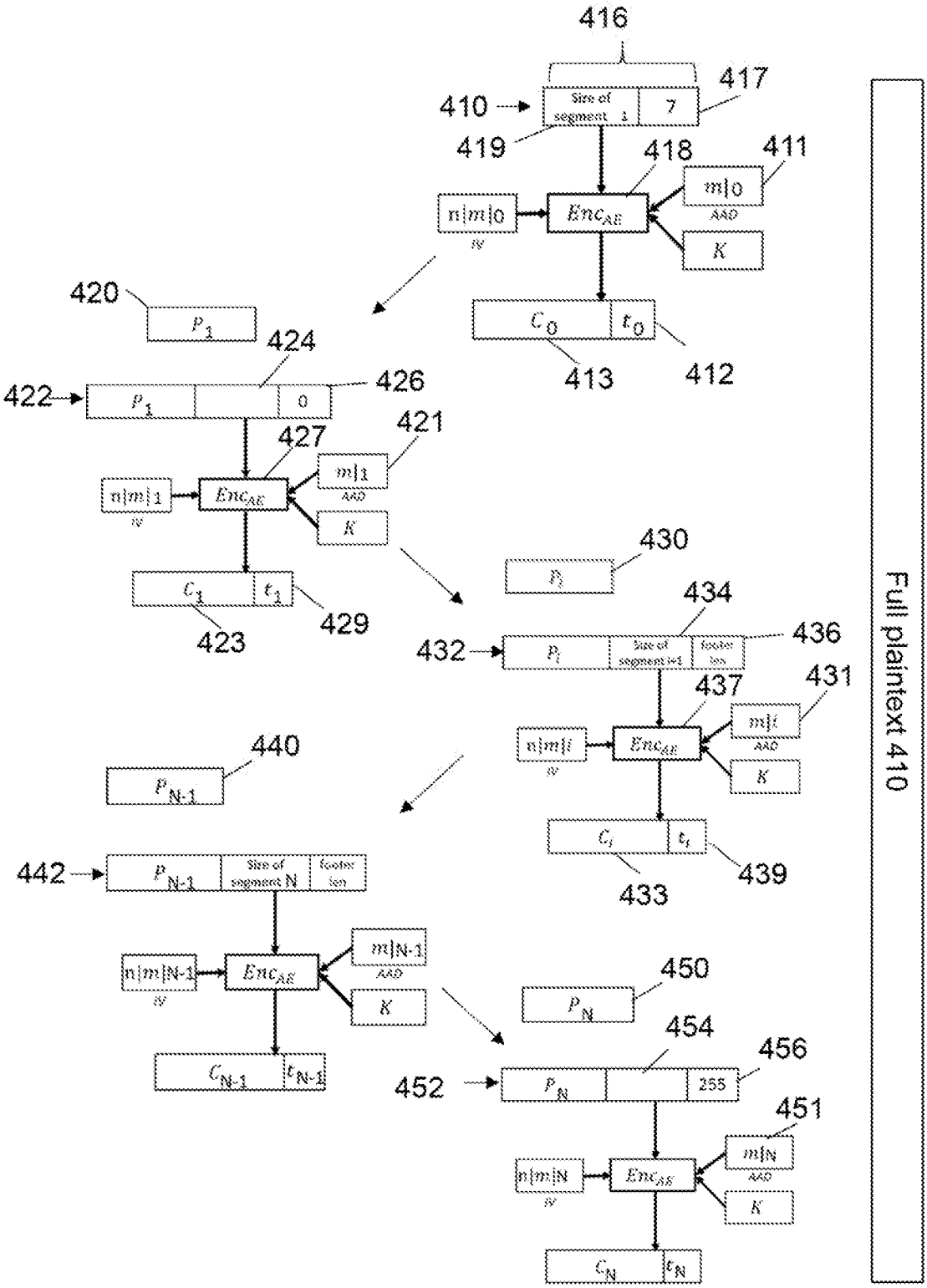

416
417
410
Size of segment 1
7
419
418
411
n|m|0
IV
Enc_{AE}
m|0
AAD
K
C_0
t_0
413
412
420
P_1
424
422
P_1
0
426
427
421
n|m|1
IV
Enc_{AE}
m|1
AAD
K
C_1
t_1
423
429
430
P_i
434
432
P_i
Size of segment i+1
Footer len
436
437
431
n|m|i
IV
Enc_{AE}
m|i
AAD
K
C_i
t_i
433
439
440
P_{N-1}
442
P_{N-1}
Size of segment N
Footer len
n|m|N-1
IV
Enc_{AE}
m|N-1
AAD
K
C_{N-1}
t_{N-1}
450
P_N
454
456
452
P_N
255
451
n|m|N
IV
Enc_{AE}
m|N
AAD
K
C_N
t_N
Full plaintext 410

FIG. 4

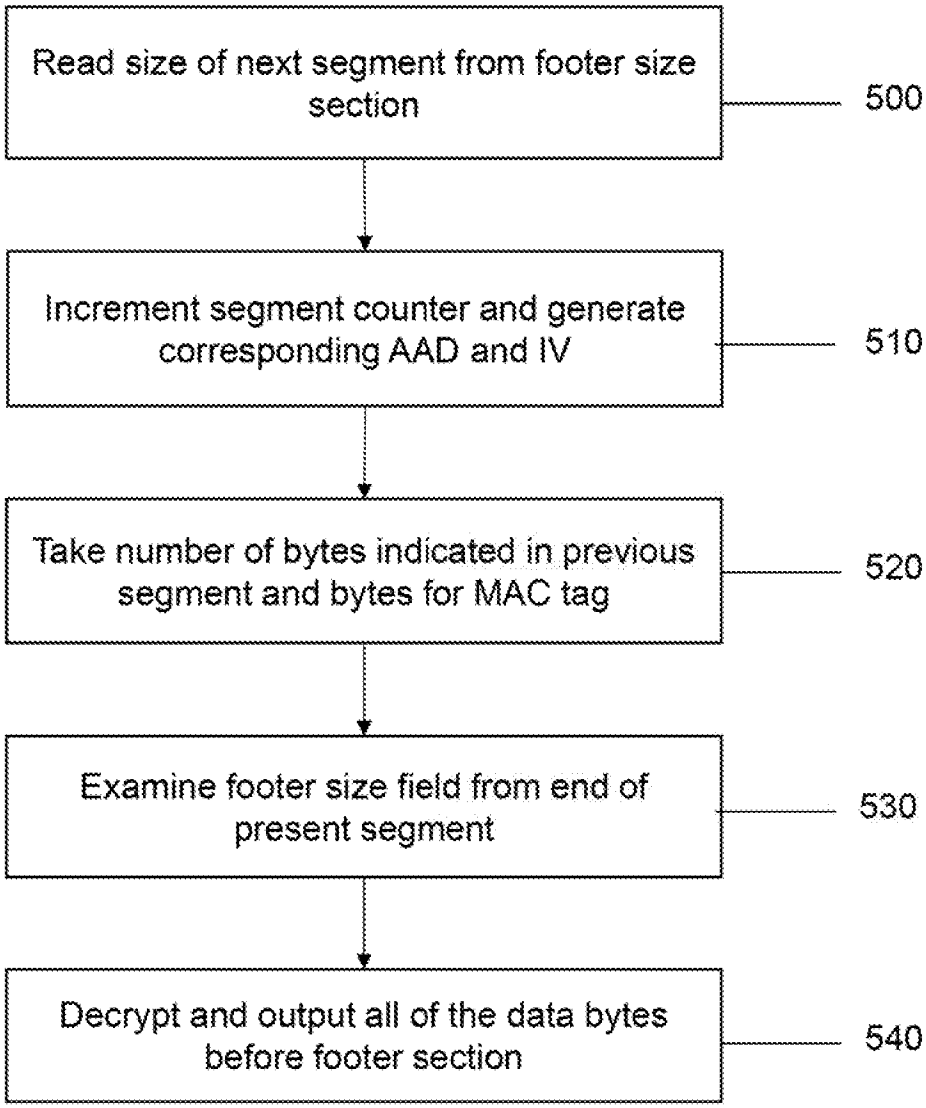

Read size of next segment from footer size section
500
Increment segment counter and generate corresponding AAD and IV
510
Take number of bytes indicated in previous segment and bytes for MAC tag
520
Examine footer size field from end of present segment
530
Decrypt and output all of the data bytes before footer section
540

FIG. 5

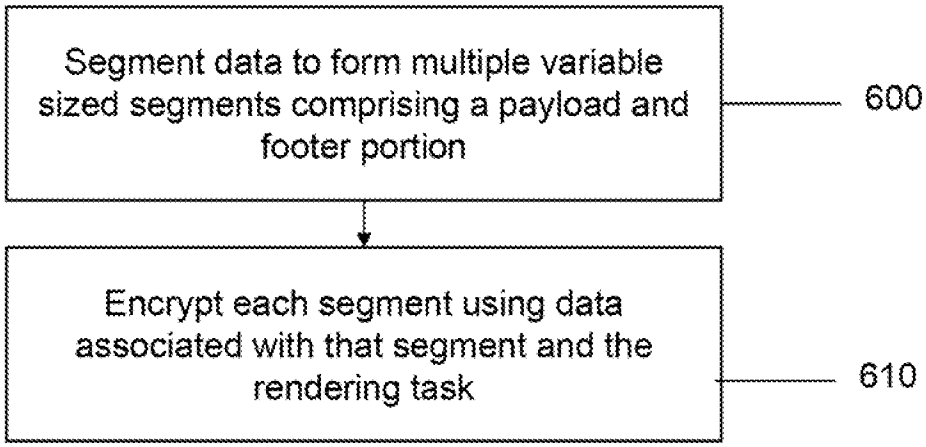

Segment data to form multiple variable sized segments comprising a payload and footer portion
600
Encrypt each segment using data associated with that segment and the rendering task
610

FIG. 6


710
Processor
Instructions
Memory
720
730

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2018/048347

A. CLASSIFICATION OF SUBJECT MATTER

G06F 3/12 *(2006.01)*
G09C 1/06 *(2006.01)*
G09C 3/00 (2006.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 3/00, 3/12, 7/00, 7/76, 15/00, 21/00, 21/10, 21/60, 21/62, G06K 15/00, G09C 1/00, 1/06, 3/00, 5/00, B41J 3/00, H04L 9/00, 9/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

PatSearch (RUPTO internal), USPTO, PAJ, K-PION, Esp@cenet, Information Retrieval System of FIPS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2004/0181661 A1 (SHARP LABORATORIES OF AMERICA, INC.) 16.09.2004, abstract, paragraphs [0017], [0025]-[0032], [0050]-[0057], claim 1, fig. 3, 9	1-15
Y	US 2006/0274355 A1 (SHARP LABORATORIES OF AMERICA, INC.) 07.12.2006, paragraph [0043]	1-15
Y	US 2018/0150619 A1 (RICOH COMPANY, LTD.) 31.05.2018, paragraph [0037]	3, 8
A	US 7561294 B2 (XEROX CORPORATION) 14.07.2009	1-15

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
01 April 2019 (01.04.2019)	04 April 2019 (04.04.2019)

Name and mailing address of the ISA/RU:
Federal Institute of Industrial Property,
Berezhkovskaya nab., 30-1, Moscow, G-59,
GSP-3, Russia, 125993
Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37

Authorized officer

D. Starshinov

Telephone No. (495) 531-65-15