UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTERNATIONAL BUSINESS MACHINES CORPORATION
Petitioner

v.

SECURITY FIRST INNOVATIONS, LLC
Patent Owner

Case No. IPR2025-01202
U.S. Patent No. 9,135,456

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 9,135,456

CHALLENGING CLAIMS 1–30
UNDER 35 U.S.C. §312 AND 37 C.F.R. §42.104

TABLE OF CONTENTS

Page

I. Introduction....1
II. Mandatory Notices....1
A. Real Party-In-Interest (37 C.F.R. §42.8(b)(1)) 1
B. Related Matters (37 C.F.R. §42.8(b)(2))....1
C. Lead And Back-up Counsel (37 C.F.R. §42.8(b)(3))....1
D. Service Information (37 C.F.R. §42.8(b)(4)) 2
III. Fees 3
IV. Certification Of Grounds For Standing 3
V. Overview Of Challenge And Relief Requested....3
A. Prior Art References....3
B. Public Accessibility of Desai 4
C. Grounds For Challenge 5
VI. The '456 Patent....5
A. Prosecution History 8
VII. Level Of Ordinary Skill In The Art 9
VIII. Claim Construction....9
IX. Specific Grounds For Challenge....10
A. Ground I: Torre And Desai Render Obvious Claims 1-3, 10-15, 22-27, And 30....10
1. Torre (EX1003)....10
2. Desai (EX1004)....22

3. Torre In Combination With Desai ("Torre/Desai System").....27

4. Claim 1.....39

5. Claim 2: The method of claim 1, wherein each of the n shares further comprises at least some of the redundancy information and storing the plurality of n shares and the redundancy information comprises storing the redundancy information in the n shares.....66

6. Claim 3: The method of claim 2, wherein storing the redundancy information in the n shares comprises including in each of the n shares information redundant to information included in at least one other of the n shares.....69

7. Claim 10: The method of claim 1, wherein at least some of the separate storage locations are geographically separate from one another.....72

8. Claim 11: The method of claim 1, wherein logically combining comprises logically combining using an exclusive-OR operation.....73

9. Claim 12: The method of claim 1, wherein the at least a portion of the encrypted data set comprises all of the encrypted data set.....74

10. Claim 13.....74

11. Claim 25.....75

12. Claims 14, 15, 22–24, 26, 27, 30.....77

B. Ground II : Torre, Desai And Watanabe Render Obvious Claims 1–3, 10–12, 25–27, And 30.....78

1. Watanabe (EX1005).....78

2. Torre/Desai In Combination With Watanabe ("Torre/Desai/Watanabe System").....79

3. Claim 1 83

4. Claim 25 85

5. Claims 2, 3, 10-12, 26, 27, 30 85

C. Ground III: Ground I Or Ground II In Combination With Orsini Renders Obvious Claims 4–9, 16–21, And 28–29. 86

1. Orsini (EX1006) 86

2. Grounds I or II In Combination With Orsini 87

3. Claim 4: The method of claim 1, wherein producing the plurality of n shares comprises causing the n shares to comprise a substantially random distribution of units of data of the second data set. 98

Claim 5: The method of claim 4, wherein causing the n shares to comprise a substantially random distribution comprises using a key. 98

Claim 6: The method of claim 5, wherein substantially randomly distributing units of data comprises using a key. 98

Claim 7: The method of claim 1, wherein producing the plurality of n shares comprises substantially randomly distributing units of data of the second data set into the plurality of n shares. 98

4. Claim 8: The method of claim 1, wherein producing the plurality of n shares comprises distributing units of data into the plurality of n shares and substantially randomly shuffling the units of data in the shares. 102

5. Claim 9: The method of claim 8, wherein substantially randomly shuffling the units of data comprises using a key. 104

6. Claims 16–21, 28–29 104

X. Conclusion 105

TABLE OF AUTHORITIES

Page(s)

Cases

Keynetik, Inc. v. Samsung Elecs. Co., Ltd.,
No. 2022-1127, 2023 WL 2003932 (Fed. Cir. Feb. 15, 2023)..... 37, 93, 97

Statutes

35 U.S.C. § 102 3

35 U.S.C. § 103 .....3, 5

35 U.S.C. § 314(a) 5

Rules

37 C.F.R. § 42.104(b)(1)-(2).....3

37 C.F.R. §§ 42.22(a)(1).....3

PETITIONER'S TABLE OF EXHIBITS

Exhibit No.	DESCRIPTION
1001	U.S. Patent No. 9,135,456 ('456 Patent)
1002	Declaration of Dr. Erez Zadok
1003	U.S. Patent Publication No. 2003/0065656 ("Torre")
1004	Anand Desai, "The Security of All-Or-Nothing Encryption: Protecting Against Exhaustive Key Search," Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, LNCS 1880, 359-375, 2000 ("Desai")
1005	U.S. Patent Application Publication No. 2003/0091191 Al ("Watanabe")
1006	U.S. Patent Publication No. 2004/0049687 ("Orsini")
1007	Declaration of Dr. Sylvia Hall-Ellis ("Hall-Ellis Decl.")
1008	File History of U.S. Patent No. 9,135,456 ("'456 File History")
1009	File History of U.S. Application No. 13/371,363 ("'363 App.")
1010	Ronald L. Rivest, "All-or-Nothing Encryption and the Package Transform," *Fast Software Encryption '97*, Lecture Notes in Computer Science Vol. 1267, E. Biham ed., Springer-Verlag, 1997 ("Rivest")
1011	Doug Burger, "Memory Systems," ACM COMPUTING SURVEYS, Vol. 28, No. 1, March 1996 ("Burger")
1012	"Understanding Computers: An Overview for Records and Archives Staff," INTERNATIONAL RECORDS MANAGEMENT TRUST, 1999 ("Understanding Computers")
1013	*Google, LLC v. Security First Innovations, LLC*, IPR2024-00212, Exhibit 1043 (Patent Owner's Proposed Claim Constructions In *Security First Innovations, LLC v. Google, LLC*, No. 2:23-cv-00097 (E.D. Va.)) ("PO's Google Litigation Construction")
1014	U.S. Provisional Application No. 60/622,146 ("'146 App.")
1015	U.S. Provisional Application No. 60/718,185 ("'185 App.")
1016	Microsoft Computer Dictionary, 5th ed., 2002, excerpts ("Microsoft Computer Dictionary")
1017	Bruce Schneier, Applied Cryptography, 2nd ed., 1996, excerpts ("Schneier")
1018	Yitzhak Birk, "Random RAIDs with Selective Exploitation of Redundancy for High Performance Video Servers," IEEE 1997 ("Birk")

Exhibit No.	DESCRIPTION
1019	Costello et al., "Some Thoughts on the Equivalence of Systematic and Non-Systematic Convolutional Encoders," (August 2, 2001) ("Costello")
1020	P. Venkat Rangan et al., "Designing File Systems for Digital Video and Audio," ACM SIGOPS Operating Systems Review, Volume 25, Issue 5, 81–94 (1991) ("Rangan")
1021	Comparison of the '456 Patent's Specification and Orsini
1022	"Primer: Reed-Solomon Error Correction Codes (ECC)," AHA PRODUCTS GROUP (Aug. 8, 2004) ("Primer: Reed-Solomon ECC")
1023	Norman Matloff, "Introduction to Digital Logic," September 4, 2003 ("Matloff")
1024	File History of U.S. Patent Application No. 11/258,839
1025	U.S. Patent Publication No. 2003/0016596A1 to Chiquoine et. al, Published Jan. 23, 2003, titled "Data Transmission from RAID Devices." ("Chiquoine"))
1026	Tsai et al., "Dynamic Buffer Management for Near Video-On-Demand Systems," Multimedia Tools and Applications 6, 61–83 (1998) ("Tsai")
1027	John Kubiatowicz, et al., "OceanStore: An Architecture for Global-Scale Persistent Storage," ACM 2000 ("Kubiatowicz")
1028	Curriculum Vitae of Dr. Erez Zadok

Throughout the Petition, all annotations, coloring, and emphases have been added unless indicated otherwise.

LISTING OF CLAIMS

Claim 1

[1Pre] A method performed by a computer system for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the method comprising:

[1A] reading the first data set from a memory that stores the first data set,;

[1B] processing the first data set to produce a second data set, the processing comprising:

[1B-1] encrypting the first data set using an encryption key to produce an encrypted data set, and

[1B-2] logically combining at least a portion of the encrypted data set with the encryption key to produce a resultant,

[1B-3] wherein the second data set comprises the encrypted data set and the resultant, and wherein all of the at least a portion of the encrypted data set is required to recover the encryption key based on the resultant;

[1C] producing redundancy information based on information in the second data set;

[1D] producing a plurality of n shares from the second data set, wherein each of the n shares comprises at least some of the second data set; and;

[1E] storing the plurality of n shares and the redundancy information, wherein at least some of the n shares are stored in separate storage locations and the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares

Claim 2

[2] The method of claim 1, wherein each of the n shares further comprises at least some of the redundancy information and storing the plurality of n shares and the redundancy information comprises storing the redundancy information in the n shares.

Claim 3

[3] The method of claim 2, wherein storing the redundancy information in the n shares comprises including in each of the n shares information redundant to information included in at least one other of the n shares.

Claim 4

[4] The method of claim 1, wherein producing the plurality of n shares comprises causing the n shares to comprise a substantially random distribution of units of data of the second data set.

Claim 5

[5] The method of claim 4, wherein causing the n shares to comprise a substantially random distribution comprises using a key.

Claim 6

[6] The method of claim 5, wherein substantially randomly distributing units of data comprises using a key.

Claim 7

[7] The method of claim 1, wherein producing the plurality of n shares comprises substantially randomly distributing units of data of the second data set into the plurality of n shares:

Claim 8

[8] The method of claim 1, wherein producing the plurality of n shares comprises distributing units of data into the plurality of n shares and substantially randomly shuffling the units of data in the shares.

Claim 9

[9] The method of claim 8, wherein substantially randomly shuffling the units of data comprises using a key.

Claim 10

[10] The method of claim 1, wherein at least some of the separate storage locations are geographically separate from one another:

Claim 11

[11] The method of claim 1, wherein logically combining comprises logically combining using an exclusive-OR operation.

Claim 12

[12] The method of claim 1, wherein the at least a portion of the encrypted data set comprises all of the encrypted data set.

Claim 13

[13Pre] A computer system for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the system comprising:

one or more processors configured to:

[13A] produce a second data set from the first data set by:

[13A-1] encrypting the first data set using an encryption key to produce an encrypted data set, and

[13A-2] logically combining at least a portion of the encrypted data set with the encryption key to produce a resultant,

[13A-3] wherein the second data set comprises the encrypted data set and the resultant, and wherein all of the at least a portion of the encrypted data set is required to recover the encryption key based on the resultant;

[13B] produce redundancy information based on information in the second data set;

[13C] produce a plurality of n shares from the second data set, wherein each of the n shares comprises at least some of the second data set; and

[13D] store the plurality of n shares and the redundancy information in a plurality of storage locations, wherein at least some of the n shares are stored in separate storage locations and the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares.

Claim 14

[14] The system of claim 13, wherein the one or more processors is configured to store the plurality of n shares and the redundancy information by storing the redundancy information in the n shares.

Claim 15

[15] The system of claim 14, wherein the one or more processors is configured to store the redundancy information in the n shares by including in each of the n shares information redundant to information included in at least one other of the n shares.

Claim 16

[16] The system of claim 13, wherein the one or more processors is configured to produce the plurality of n shares by causing the n shares to comprise a substantially random distribution of units of data of the second data set.

Claim 17

[17] The system of claim 16, wherein the one or more processors is configured to cause the n shares to comprise a substantially random distribution by using a key.

Claim 18

[18] The system of claim 13, wherein the one or more processors is configured to produce the plurality of n shares by substantially randomly distributing units of data of the second data set into the plurality of n shares.

Claim 19

[19] The system of claim 18, wherein the one or more processors is configured to substantially randomly distribute units of data by using a key.

Claim 20

[20] The system of claim 13, wherein the one or more processors is configured to produce the plurality of n shares by distributing units of data into the plurality of n shares and substantially randomly shuffling the units of data in the shares.

Claim 21

[21] The system of claim 20, wherein the one or more processors is configured to substantially randomly shuffle the units of data by using a key.

Claim 22

[22] The system of claim 13, wherein at least some of the separate storage locations are geographically separate from one another.

Claim 23

[23] The system of claim 13, wherein the one or more processors is configured to logically combine by logically combining using an exclusive-OR operation.

Claim 24

[24] The system of claim 13, wherein the at least a portion of the encrypted data set comprises all of the encrypted data set.

Claim 25

[25] A non-transitory computer-readable medium comprising instructions that, when executed by one or more processors, cause a computer system to carry out a method for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the method comprising:

[25A] reading the first data set from a memory that stores the first data set,

[25B] producing a second data set from the first data set, said producing comprising:

[25B-1] encrypting the first data set using an encryption key to produce an encrypted data set, and

[25B-2] logically combining at least a portion of the encrypted data set with the encryption key to produce a resultant,

[25B-3] wherein the second data set comprises the encrypted data set and the resultant, and wherein all of the at least a portion of the encrypted data set is required to recover the encryption key based on the resultant;

[25C] producing redundancy information based on information in the second data set;

[25D] producing a plurality of n shares from the second data set, wherein each of the n shares comprises at least some of the second data set; and

[25E] storing the plurality of n shares and the redundancy information, wherein at least some of the n shares are stored in separate storage locations and the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares.

Claim 26

[26] The non-transitory computer-readable medium of claim 25, wherein storing the plurality of n shares and the redundancy information comprises storing the redundancy information in the n shares.

Claim 27

[27] The non-transitory computer-readable medium of claim 26, wherein storing the redundancy information in the n shares comprises including in each of the n shares information redundant to information included in at least one other of the n shares.

Claim 28

[28] The non-transitory computer-readable medium of claim 25, wherein producing the plurality of n shares comprises causing the n shares to comprise a substantially random distribution of units of data of the second data set.

Claim 29

[29] The non-transitory computer-readable medium of claim 28, wherein causing the n shares to comprise a substantially random distribution comprises using a key.

Claim 30

[30] The non-transitory computer-readable medium of claim 25, wherein the at least a portion of the encrypted data set comprises all of the encrypted data set.

I. Introduction

Petitioner respectfully requests cancellation of all claims in U.S. Patent No. 9,135,456 ("'456 Patent").

II. Mandatory Notices

A. Real Party-In-Interest (37 C.F.R. §42.8(b)(1))

Petitioner, IBM, identifies itself as a real party-in-interest ("RPI").

B. Related Matters (37 C.F.R. §42.8(b)(2))

The '456 Patent is currently asserted against Petitioner by Patent Owner in *Security First Innovations, LLC v. International Business Machines Corporation.*, No. 1:25-cv-00514 (EDVA) ("Litigation"). Petitioner will file petitions for *inter partes* review of U.S. Patent Nos. 8,904,194 and 8,271,802, each of which is being asserted against Petitioner in the Litigation.

C. Lead And Back-up Counsel (37 C.F.R. §42.8(b)(3))

Petitioner is filing a Power of Attorney appointing the practitioners associated with Customer Number 132,593. Petitioner designates the following lead and back-up counsel:

Lead Counsel	First Back-up Counsel
Taeg Sang Cho (Reg. No. 69,618) Desmarais LLP 230 Park Ave New York, NY 10169 Telephone: (212) 351-3400 Email: tcho@desmaraisllp.com	Benjamin Rodd (Reg. No. 79,158) Desmarais LLP 230 Park Ave New York, NY 10169 Telephone: (212) 351-3400 Email: brodd@desmaraisllp.com
Back-up Counsel	**Back-up Counsel**
Lindsey Miller (*pro hac vice* forthcoming) Desmarais LLP 230 Park Ave New York, NY 10169 Telephone: (212) 351-3400 Email: lmiller@desmaraisllp.com	William Yau (*pro hac vice* forthcoming) Desmarais LLP 230 Park Ave New York, NY 10169 Telephone: (212) 351-3400 Email: wyau@desmaraisllp.com
Back-up Counsel	
Laura Avena (Reg. No. 78,920) Desmarais LLP 230 Park Ave New York, NY 10169 Telephone: (212) 351-3400 Email: lavena@desmaraisllp.com	

D. Service Information (37 C.F.R. §42.8(b)(4))

Post and hand delivery: Desmarais LLP

230 Park Ave, New York, NY 10169

Telephone: 212-351-3400

Email: IBM-SFI-IPR-Service@desmaraisllp.com

Please address all correspondence to counsel identified above. Petitioner consents to electronic service by email at:

tcho@desmaraisllp.com

brodd@desmaraisllp.com

lmiller@desmaraisllp.com;

wyau@desmaraisllp.com;

lavena@desmaraisllp.com; and

IBM-SFI-IPR-Service@desmaraisllp.com

III. Fees

Petitioner is concurrently electronically submitting the required fees for this Petition. The Board is authorized to charge Desmarais LLP's deposit account, No. 50-6822, for any fee deficiency.

IV. Certification Of Grounds For Standing

Petitioner certifies that the '456 Patent is available for *inter partes* review and that Petitioner is not barred or estopped from requesting *inter partes* review.

V. Overview Of Challenge And Relief Requested

Under 37 C.F.R. §§42.22(a)(1) and 42.104(b)(1)-(2), Petitioner requests cancellation of claims 1–30 ("Challenged Claims") of the '456 Patent.

A. Prior Art References

This Petition applies pre-AIA provisions of 35 U.S.C. §§ 102 and 103.

The following references are pertinent to the grounds of unpatentability explained below:

Pat. Pub. No. or Title	Publication/Priority Date	Prior Art Under At Least (35 U.S.C.)
US 2003/0065656 ("Torre") (EX1003)	published April 3, 2003	102(b)
"The Security of All-Or-Nothing Encryption: Protecting Against Exhaustive Key Search" ("Desai") (EX1004)	published as early as August 20, 2000	102(b)
US 2003/0091191 ("Watanabe") (EX1005)	published February 12, 2004	102(b)
US 2004/0049687 ("Orsini") (EX1006)	published March 11, 2004	102(a),(b)

B. Public Accessibility of Desai

Desai was publicly available at least since August 20, 2000—more than one year before the earliest possible priority date of the '456 Patent (September 16, 2005). EX1007, ¶¶44–50. EX1002, ¶165.

For instance, as explained by Dr. Sylvia Hall-Ellis, Desai was part of volume 1880 of the Lecture Notes in Computer Science (LNCS) series, titled *Advances in Cryptology: CRYPTO '00*, found at the Karl F. Wendt Engineering Library at the University of Wisconsin–Madison, which was "publicly available … as of August 7, 2000." EX1007, ¶47. Indeed, the print version of this book "had been received, cataloged, and indexed at the Library of Congress, made part of the OCLC bibliographic database, and received in the Karl F. Wendt Engineering Library at the University of Wisconsin–Madison" "on or shortly after August 7, 2000." EX1007, ¶50. *See also* EX1007, ¶¶46–49.

Desai is also available online from the publisher Springer-Verlag and in the Association for Computing Machinery (ACM) Digital Library, which indicates that the Desai reference was published on August 20, 2000. EX1007, ¶45.

C. Grounds For Challenge

Petitioner requests cancellation of claims 1–30 of the '456 Patent under 35 U.S.C. §103 based on the following Grounds.

Grounds	35 U.S.C.	Claims	Prior Art Reference
I	§103	1–3, 10–15, 22–27, 30	Torre, Desai
II	§103	1–3, 10–12, 25–27, 30	Torre, Desai, Watanabe
III	§103	4–9, 16–21, 28–29	References in Ground I or II, and Orsini

This Petition demonstrates that there is a reasonable likelihood that Petitioner would prevail with respect to at least one of the challenged claims. *See* 35 U.S.C. §314(a).

VI. The '456 Patent

The '456 Patent is directed to a "secure data parser" that "parses data and then splits the data into multiple portions that are stored or communicated distinctly." EX1001, Abstract. Relevant to the '456 Patent's claims, Figure 21 (below) "shows the steps of the process performed by the secure data parser on data to store the session master key with the parsed data." EX1001, 55:22–25. For example, Figure

21 of the ’456 Patent discloses: (1) generating a “session master key” (shown in magenta); (2) encrypting a data set using the session master key (shown in lime green); (3) “[s]eparating the resulting encrypted data [set] into four shares or portions of parsed data” (shown in blue); (4) similarly separating the session master key into “key share[s]” (shown in red); and (5) “append[ing] the key data [shares] to the encrypted parsed data” (shown in orange and magenta) to form data shares. EX1001, 55:22–39. The data shares with key data are further processed and stored in storage devices. *See* EX1001, 58:15–45, 58:57–64. EX1002, ¶¶160–161.

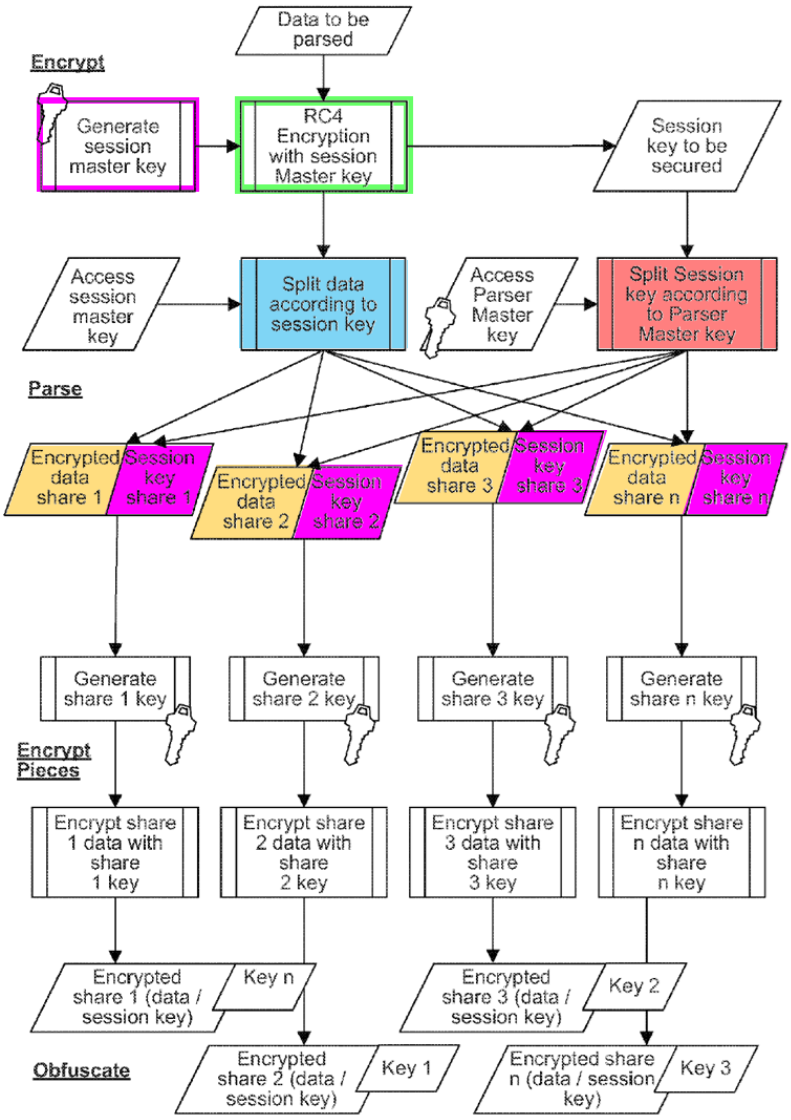


FIG. 21

EX1001, FIG. 21.

The ’456 Patent refers to the process of splitting the data into shares as “[a] cryptographic split (cryptosplit).” EX1001, 52:40–41. Its cryptosplit process “provide sufficient redundancy in the shares such that only a subset of the shares are needed to reassemble or restore the data to its original or useable form.” EX1001, 53:15–19. *See also* EX1001, 53:28–54:40. For example, “each unit of data is stored

in two shares, the primary share and the backup share," such that even if one of the shares is later missing, the remaining shares are guaranteed to "reassemble or restore the original data with no missing data units." EX1001, 53:28–34. *See also* EX1001, 53:34–54:40. The '456 Patent further explains that "an All or Nothing Transform (AoNT), such as a Full Package Transform, may be used to further secure data by transforming … any [] suitable data elements." EX1001, 74:45–49. EX1002, ¶¶162–163.

A. Prosecution History

The '456 Patent originated from U.S. Application No. 14/473,813 ("'813 App."), filed August 29, 2014, which is a continuation of U.S. Application No. 13/371,363 ("'363 App."), filed February 10, 2012, which is a continuation of U.S. Application No. 11/258,839 ('839 App."), filed October 25, 2005, which, in turn, claims priority to Provisional Applications 60/622,146 (filed October 25, 2004) and 60/718,185 (filed September 16, 2005). EX1001, Cover; EX1009. EX1002, ¶164.

On November 24, 2014, the Examiner issued a Non-Final Office Action rejecting pending claims as being anticipated by U.S. Patent Publication No. 2004/0049687 and as being rendered obvious over U.S. Patent Publication No. 2004/0003272 in view of U.S. Patent No. 6,807,649. EX1008, 300–320. The Examiner further rejected the claims because the term "substantially" rendered the

claims indefinite and for non-statutory double patenting. EX1008, 300–320. EX1002, ¶166.

Subsequently, the applicant had an interview with an Examiner and filed a terminal disclaimer, and that led to the allowance of the '456 Patent. EX1008, 373, 422, 426, 466. EX1002, ¶167.

VII. Level Of Ordinary Skill In The Art

A person of ordinary skill in the relevant field or art ("POSITA") as of the earliest claimed priority date of the '456 Patent would have had a Bachelor's degree in Computer Science, Computer Engineering, Electrical Engineering, or an equivalent field, and about 2–3 years of experience in the fields of data storage and security. Less professional experience can be substituted by additional education, and vice versa. EX1002, ¶168.

VIII. Claim Construction

Petitioner submits that no terms require construction for the purposes of this IPR.[1] EX1002, ¶¶170–172.

[1] Petitioner reserves all rights to raise claim construction arguments in other forums.

IX. Specific Grounds For Challenge

A. Ground I: Torre And Desai Render Obvious Claims 1-3, 10-15, 22-27, And 30.

1. Torre (EX1003)

Torre discloses "[a] system and method for shredding and deshredding [data] … to store and retrieve [the] data while emphasizing considerations such as redundancy, integrity, and security." EX1003, [0049]. Specifically, Torre teaches transforming the input data "into smaller pieces of data called shreds" (EX1003, [0051]) and "writing the shredded input data … [in] multiple storage units" (EX1003, [0052]). "Shredding is done in a manner that adds redundancy to the shreds so that the input data can be recovered from a fewer number of shreds than the total number stored." EX1003, [0051]. EX1002, ¶176.

Torre explains that its "shredder" shreds input data using a series of processing steps, as shown in Figure 4 below. EX1003, [0063]. *See also* EX1003, [0055]–[0057]. For example, when the shredder receives input data (step 122), it "performs first transformations (step 132)." EX1003, [0061]. The shredder then "allocates data transformed by the first transformations into various different blocks of data (step 133)" called "pre-shred[s]" (EX1003, [0061], [0080]) by "split[ting] up the [transformed] data." (EX1003, [0062]). Torre subsequently performs a second transformation "(step 134)" and "further splits the units of data into smaller sized units of data and generates additional redundant data (step 135) so that fewer pieces

of stored data are necessary to restore the stored input data." EX1003, [0061]. The shredder then completes the shredding process by performing a third transformation (step 136). EX1003, [0061]–[0062]. EX1002, ¶¶177–181.

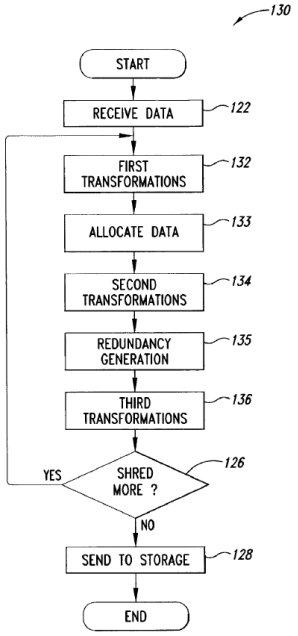


Fig. 4

EX1003, FIG. 4.

As the shredder processes input data using various steps, the shredder appends "tags" to the processed data "to keep track of what [has] been done to the pieces of the input data 139 as the pieces of the input data move through the shredder." EX1003, [0078]–[0079]. "Information [in the tags] describing how the shreds were created is used to reassemble the shreds through processes inverse to those used to create the shreds." EX1003, [0053]. EX1002, ¶¶182–183.

Torre provides implementations of a shredder and deshredder in Figures 32 and 33, respectively. Figure 32 illustrates a shredder in which: (1) the first transformer 174 "receives the input data 139 consisting in this example as a data file 460" and "generate[s] a signature associated with the received data to generate a pre-shred 466 containing the received data"; (2) the redundancy generator 468 generates six pre-shreds 470 "us[ing] a 4 input-6 output Reed Solomon redundancy generation function"; and (3) a third transformer "forms shreds 474, which includes the pre-shred data." EX1003, [0131]. Torre explains that "[t]he shreds 474 are then sent to the storage pool 148 where in this instance is made up of six servers 476." EX1003, [0132]. EX1002, ¶184.

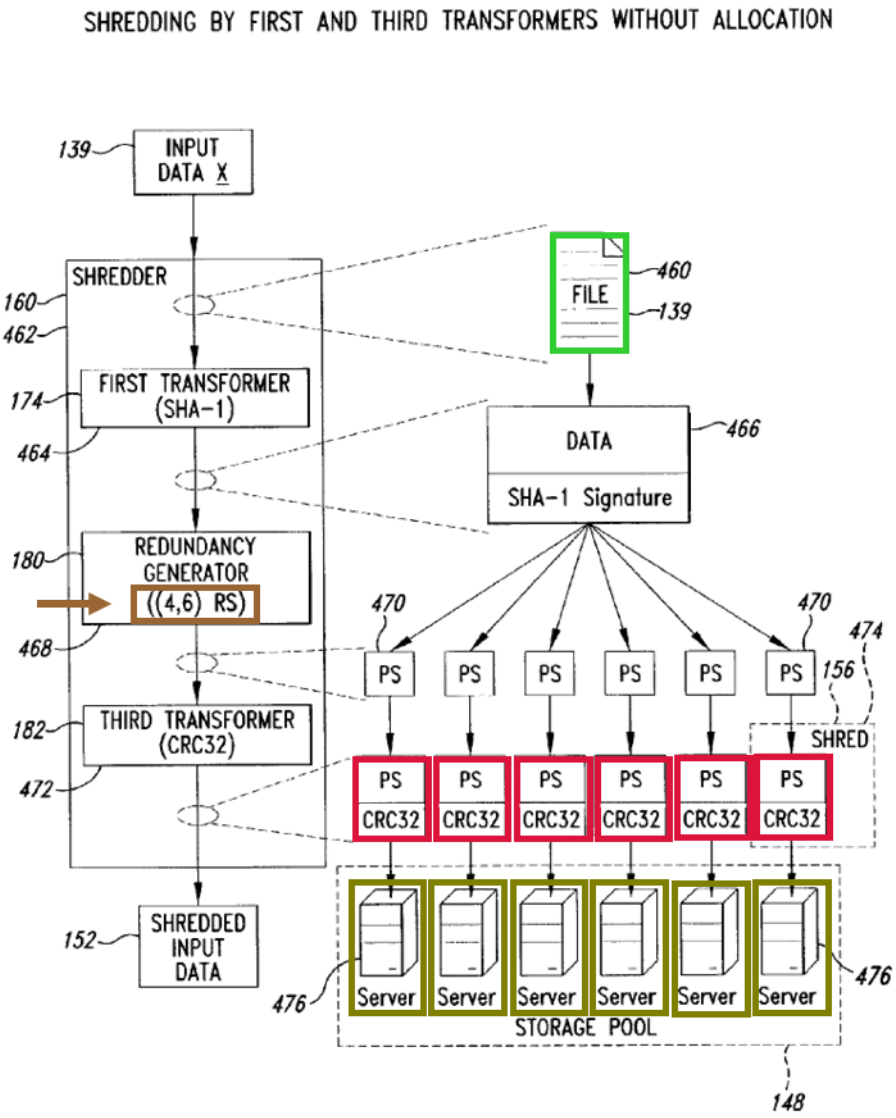


Fig. 32

EX1003, FIG. 32.

Figure 33, on the other hand, contains a deshredder that is "[c]omplementary to the shredder" from Figure 32. EX1003, [0133]. In the deshredder, (1) the third inverse transformer(s) "receive the four shreds 474 from four of the [six] servers 476"; (2) the inverse redundancy generator "perform[s] a[n] inverse Reed Solomon

redundancy function on the four pre-shreds to reconstruct the pre-shred 466 containing data"; and (3) the first inverse transformer outputs the data file 460 as the input data 139." EX1003, [0133]–[0134]. EX1002, ¶185.

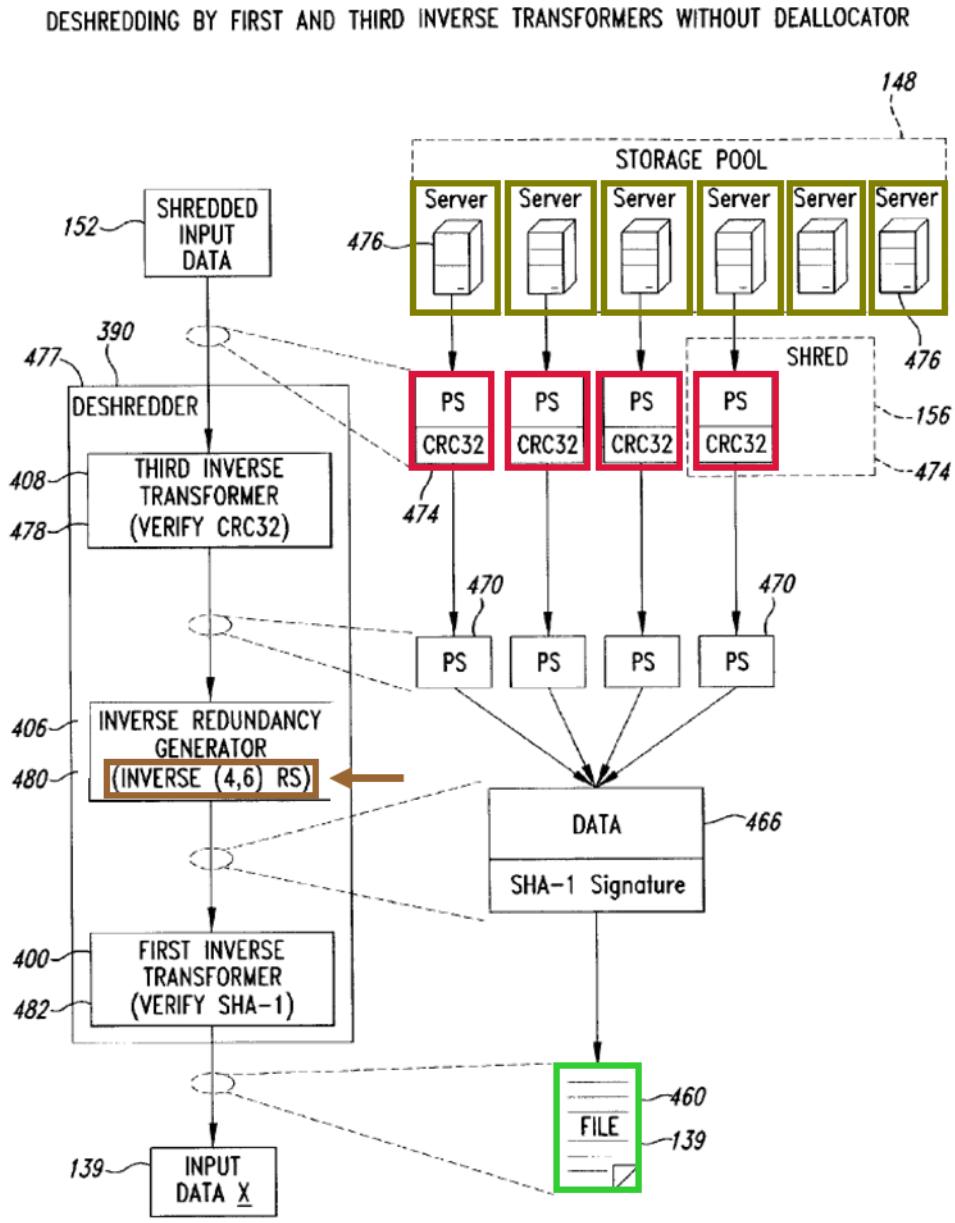


Fig. 33

EX1003, FIG. 33.

i. Torre Figure 21 Embodiment ("Torre System-1")

Torre further explains that the redundancy generator and the inverse redundancy generator—like the redundancy generator 468 in Figure 32 and the inverse redundancy generator 480 in Figure 33—can implement different types of redundancy schemes. For example, Torre teaches that the redundancy generator may use systematic encoding. EX1003, [0116]. Figure 21 shows "[a]n example of systematic encoding" "where the redundancy generator 180 receives four inputs of data and passes through the four inputs of received data to the output of the redundancy generator." EX1003, [0117]. "The redundancy generator 180 also generates two outputs of redundant data from the received data and outputs the two outputs of redundant data along with the four outputs of the received data." *Id.* EX1002, ¶186.

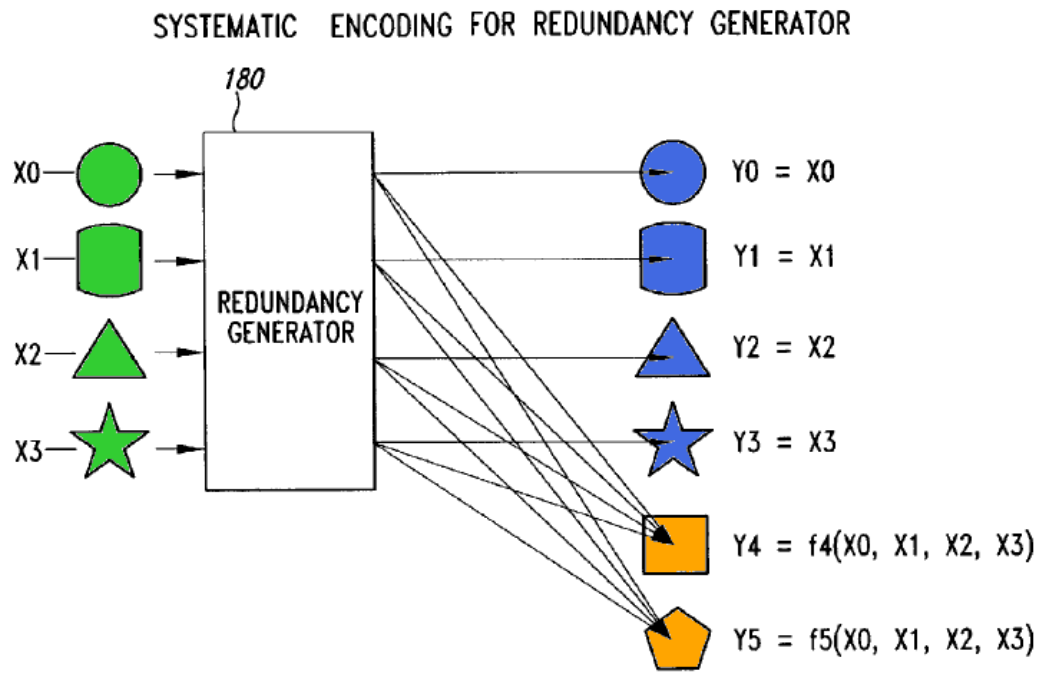
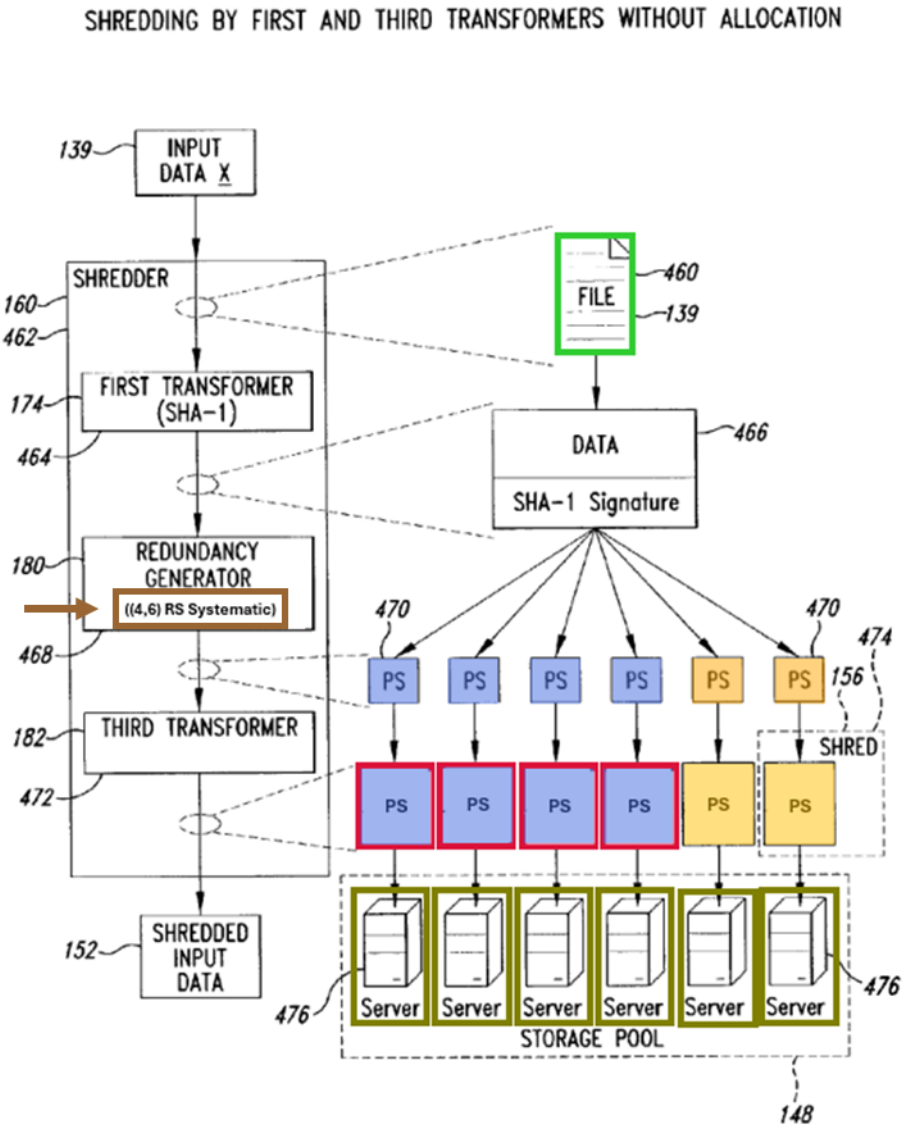


Fig. 21

EX1003, FIG. 21.

A POSITA would have understood that such implementation of the redundancy generator is applicable to the embodiment in Torre's Figure 32 because Torre's Figure 32 also labels the redundancy generator with the item number "180." *See* EX1003, Figure 32. The following figure illustrates the features that Torre describes in connection with the shredder in Figure 32 but is modified—as taught by Torre—so that the redundancy generator applies Reed-Solomon ***systematic*** encoding (Torre's Figure 21) to generate pre-shreds 470 that include the pass-through data blocks of the received data and redundant data. Such system is herein referred to as "Torre System-1." EX1002, ¶187.

Torre System-1

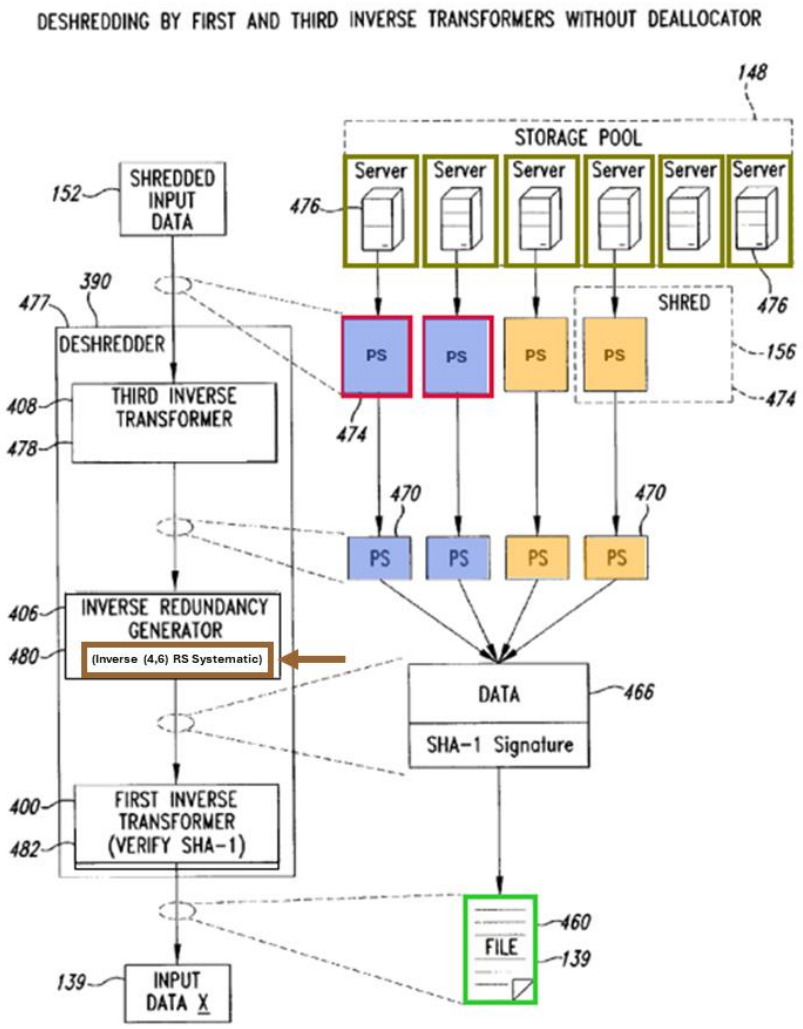


EX1003, FIG. 32.

Torre teaches that such variation is also applicable to the inverse redundancy generator in Torre's Figure 33. The following figure illustrates the features of Figure 33 but is modified—as taught by Torre—so that the inverse redundancy generator

applies inverse Reed-Solomon ***systematic*** encoding. Such system is also herein referred to as "Torre System-1." EX1002, ¶188.

Torre System-1



EX1003, FIG. 33.

ii. Torre Figure 22 Embodiment ("Torre System-2")

In addition to systematic encoding, the redundancy generator may also use ***non-systematic*** encoding. EX1003, [0116]. Figure 22 of Torre shows "[a]n

example of ***non-systematic*** encoding" "where the redundancy generator 180 generates six outputs of redundancy-generated data from four inputs of received data." EX1003, [0117]. EX1002, ¶189.

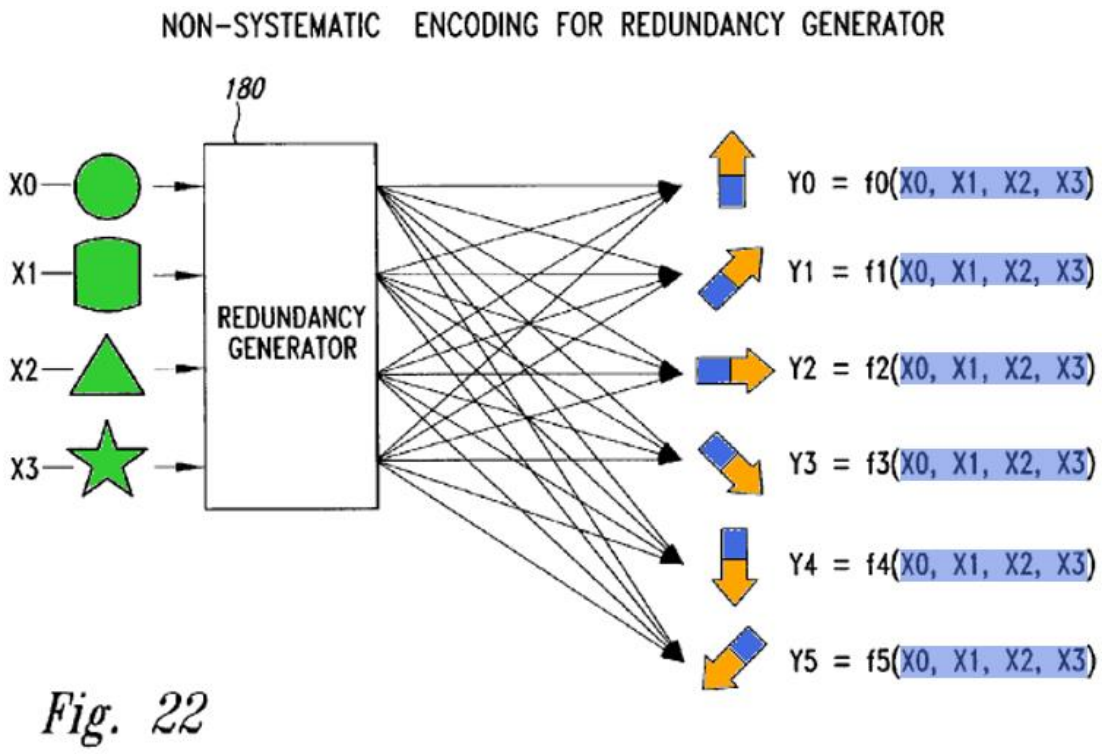


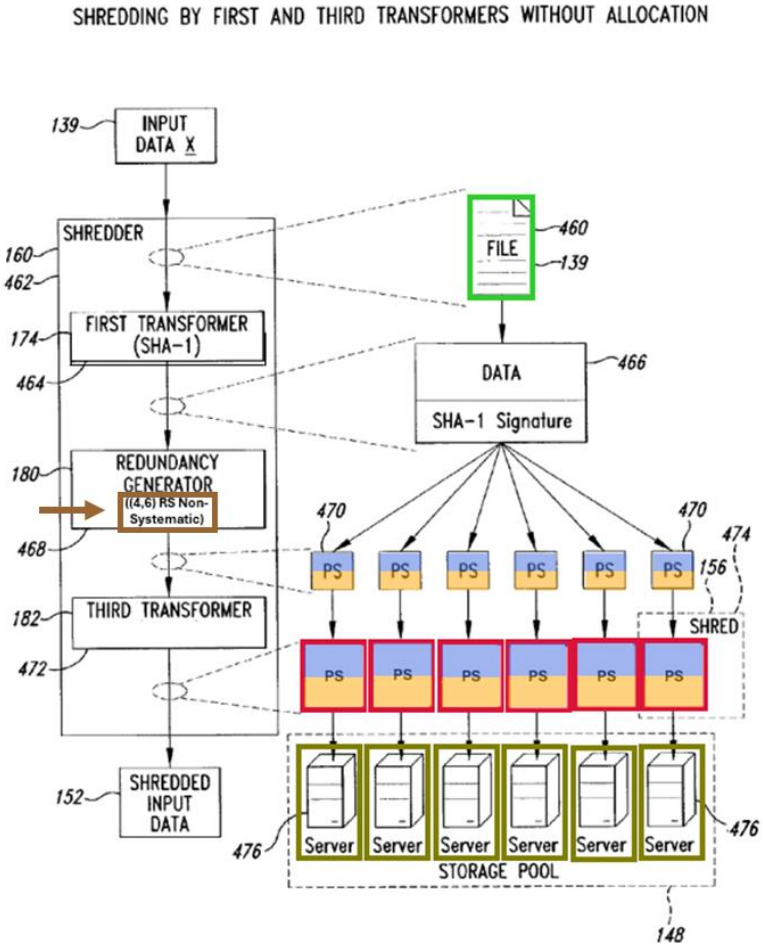
Fig. 22

EX1003, FIG. 22.

Again, a POSITA would have understood that such implementation of the redundancy generator is applicable to the embodiment in Torre's Figure 32 because Torre's Figure 32 also labels the redundancy generator with the item number "180." *See* EX1003, Figure 32. The following figure illustrates the features that Torre describes in connection with the shredder in Figure 32 but is modified—as taught by Torre—to apply Reed-Solomon (RS) ***non-systematic*** encoding to generate pre-

shreds 470 that include pass-through data blocks of the received data and redundant data. Such system is herein referred to as "Torre System-2." EX1002, ¶190.

Torre System-2

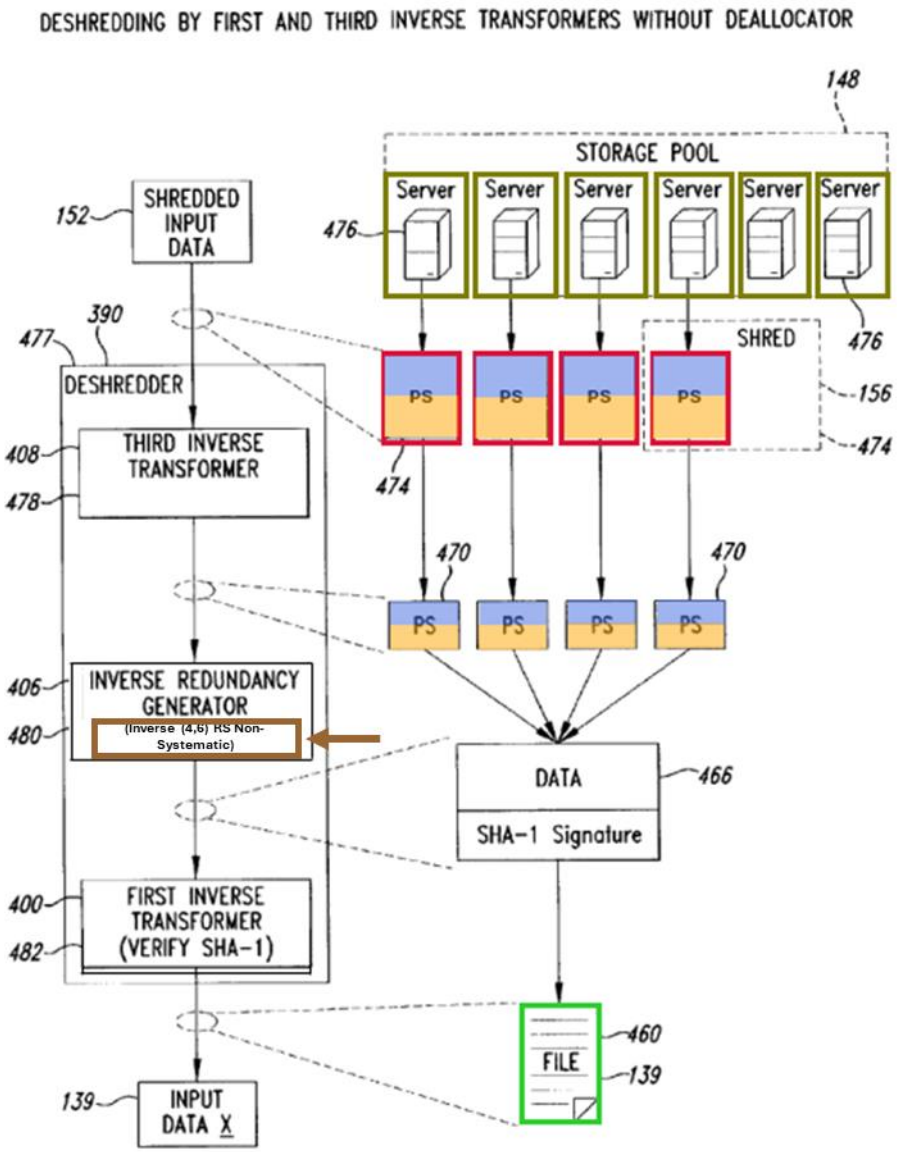


EX1003, FIG. 32.

Torre teaches that such variation is also applicable to the inverse redundancy generator in Torre's Figure 33. The following figure illustrates the features of Figure 33 but is modified—as taught by Torre—so that the inverse redundancy generator

applies inverse Reed-Solomon ***non-systematic*** encoding. Such system is herein also referred to as "Torre System-2." EX1002, ¶191.

Torre System-2



EX1003, FIG. 33.

* * *

Torre System-1 and Torre System-2 are identical except for the specific implementation of the redundancy generator. Thus, in the forthcoming sections, when the analysis does not implicate the specific operation of the redundancy generator, the analysis refers to Torre System-1 and Torre System-2 collectively as "Torre System." EX1002, ¶192.

2. Desai (EX1004)

Desai is directed to a variation of a technique called "all-or-nothing transform," or AONT. AONT was first introduced by Dr. Rivest in a seminar paper called "All-or-Nothing Encryption and the Package Transform," *Fast Software Encryption '97*, Lecture Notes in Computer Science Vol. 1267, E. Biham ed., Springer-Verlag, 1997 ("Rivest") (EX1010). As Dr. Zadok explains, AONT involves transforming a given data set (e.g., message sequence m_1, m_2, …, m_s) into a modified data set (e.g., pseudo-message sequence m'_1, m'_2, …, $m'_{s'}$). *See* EX1010, 3. This process results in AONT's defining characteristic where a user (or would-be attacker) must first have access to ***all*** pieces of the transformed data set (e.g., pseudo-message sequence m'_1, m'_2, …, $m'_{s'}$) before the AONT can be reversed to obtain the original data (e.g., message sequence m_1, m_2, …, m_s). *See* EX1010, 3; EX1004, 7. EX1002, ¶¶193–194.

Desai discloses "a simplified version of Rivest's 'package transform' construction of an AONT" that is based on "an efficient block-cipher-based

construction of an AONT." EX1004, 9, 16. *See also* EX1004, 7 ("[A]n AONT is an efficiently computable transformation, mapping sequences of blocks to sequences of blocks"), 22. This simplified version enables "all-or-nothing encryption modes that cost only two times the cost of normal CBC encryption" in contrast to Rivest's "package transform, [where] the resulting modes had cost about three times the cost of CBC." EX1004, 9. EX1002, ¶195.

Desai teaches its construction of AONT that is "based on the CTR [i.e., counter] mode of encryption"—referred to as "CTRT"—which comprises two algorithms: the "encoding algorithm" "*E*-CTRT"; and the "decoding algorithm" "*D*-CTRT." EX1004, 15–16. EX1002, ¶196.

Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\dots x[n])$	Algorithm $\mathcal{D}$-CTRT$^{F,F^{-1}}(x')$
$K' \leftarrow \{0,1\}^l$	Parse x' as $x'[1]\dots x'[n+1]$
$K = K' \bmod 2^k$ // $(\|K\| = k)$	$K' = x'[1] \oplus \cdots \oplus x'[n+1]$
for $i = 1,\dots,n$ do	$K = K' \bmod 2^k$ // $(\|K\| = k)$
$\quad x'[i] = x[i] \oplus F_K(i)$	for $i = 1,\dots,n$ do
$x'[n+1] = K' \oplus x'[1] \oplus \dots \oplus x'[n]$	$\quad x[i] = x'[i] \oplus F_K(i)$
return $x'[1]\dots x'[n+1]$	return $x[1]\dots x[n]$

EX1004, 16.

Focusing first on the encoding algorithm, the encoding algorithm receives, as input, a sequence of data blocks $x[1]$ … $x[n]$, and ultimately generates, as output, a sequence of encrypted data blocks $x'[1]$ … $x'[n]$, $x'[n+1]$. EX1004, 15–16. The

algorithm also takes as a parameter "*F*," which is a "cipher with key length *k* and block length *l*." EX1004, 16–17. EX1002, ¶197.

Desai describes the encoding algorithm as follows. EX1002, ¶198.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

At **step E1**, the algorithm generates a key *K′* as a random series of bits of length *l*. EX1004, 16–17. EX1002, ¶199.

At **step E2**, the algorithm computes *K′* modulo 2^k to obtain the truncated encryption key *K*, where *k* is the key length accepted by the cipher *F*. EX1004, 16 ("We describe the transform CTRT = (***E*-CTRT**, *D*-CTRT) … using an ideal cipher *F* with key length *k*…"). EX1002, ¶200.

At **step E3**, the algorithm encrypts each of the data blocks *x*[1] … *x*[*n*] by applying the cipher *F* with the truncated key *K* to obtain encrypted data blocks *x′*[1] … *x′*[*n*]. EX1004, 16 ("The message *x* to be transformed is regarded as a sequence of *l*-bit blocks, *x* = *x*[1] … *x*[*n*]."). EX1002, ¶201.

At **step E4**, the algorithm computes the exclusive-OR operation ("XOR," denoted by the symbol "⊕") of the key K' with each of the encrypted data blocks $x'[1] \ldots x'[n]$ to produce a new data block $x'[n+1]$. EX1004, 16. EX1002, ¶202.

At **step E5**, the algorithm "return[s] $x'[1] \ldots [x'[n],] x'[n+1]$." EX1004, 16. EX1002, ¶203.

Focusing next on the decoding algorithm, the decoding algorithm receives, as input, x', and generates, as output, the original sequence of data blocks $x[1] \ldots x[n]$. EX1004, 16. EX1002, ¶204.

Desai describes the decoding algorithm as follows. EX1002, ¶205.

	Algorithm $\mathcal{D}\text{-CTRT}^{F,F^{-1}}(x')$
Step D1	Parse x' as $x'[1] \ldots x'[n+1]$
Step D2	$K' = x'[1] \oplus \cdots \oplus x'[n+1]$
Step D3	$K = K' \bmod 2^k$ // $(\lvert K \rvert = k)$
Step D4	for $i = 1, \ldots, n$ do $x[i] = x'[i] \oplus F_K(i)$
Step D5	return $x[1] \ldots x[n]$

EX1004, 16.

At **step D1**, the algorithm "[p]arse[s] x' as $x'[1] \ldots [x'[n],] x'[n+1]$." EX1004, 16. EX1002, ¶206.

At **step D2**, the algorithm recovers K' by computing the XOR (denoted by the symbol "⊕") of $x'[1] \ldots x'[n]$ and $x'[n+1]$. EX1004, 16. EX1002, ¶207.

At **step D3**, the algorithm computes K' modulo 2^k to obtain the truncated encryption key K, where k is the key length accepted by the cipher F. EX1004, 16 ("We describe the transform CTRT = (E-CTRT, ***D*-CTRT**) … using an ideal cipher F with key length k…"). EX1002, ¶208.

At **step D4**, the algorithm decrypts the encrypted data set $x'[1] \ldots x'[n]$ by applying the cipher F with the key K to recover the original input data blocks $x[1] \ldots x[n]$. EX1004, 16. EX1002, ¶209.

At **step D5**, the algorithm "return[s] the data set $x[1] \ldots x[n]$." EX1004, 16. EX1002, ¶210.

Desai teaches that its "CTRT is strong enough that when used to realize the all-or-nothing encryption paradigm, the resulting mode will have the properties we desire": "key-search resistance" and "non-separability of keys." EX1004, 12–13, 16–17. Specifically, Desai explains that although "AONT is distinct from an encryption scheme," "if the output of an AONT is encrypted … then the resulting scheme will not only be secure as an encryption scheme but also be strongly non-separable"—meaning "it should be infeasible to determine even one message block (or any property of a particular message block) without decrypting all [encrypted] blocks." EX1004, 7. EX1002, ¶211.

Desai further explains that when CTRT is applied, "[a]s long as at least one block of the [AONT] output is missing, the key K used to 'encrypt' the message

blocks is information theoretically hidden," and that "CTRT would also have been secure in the sense given by Rivest [20]." EX1004, 11–12, 17. EX1002, ¶212.

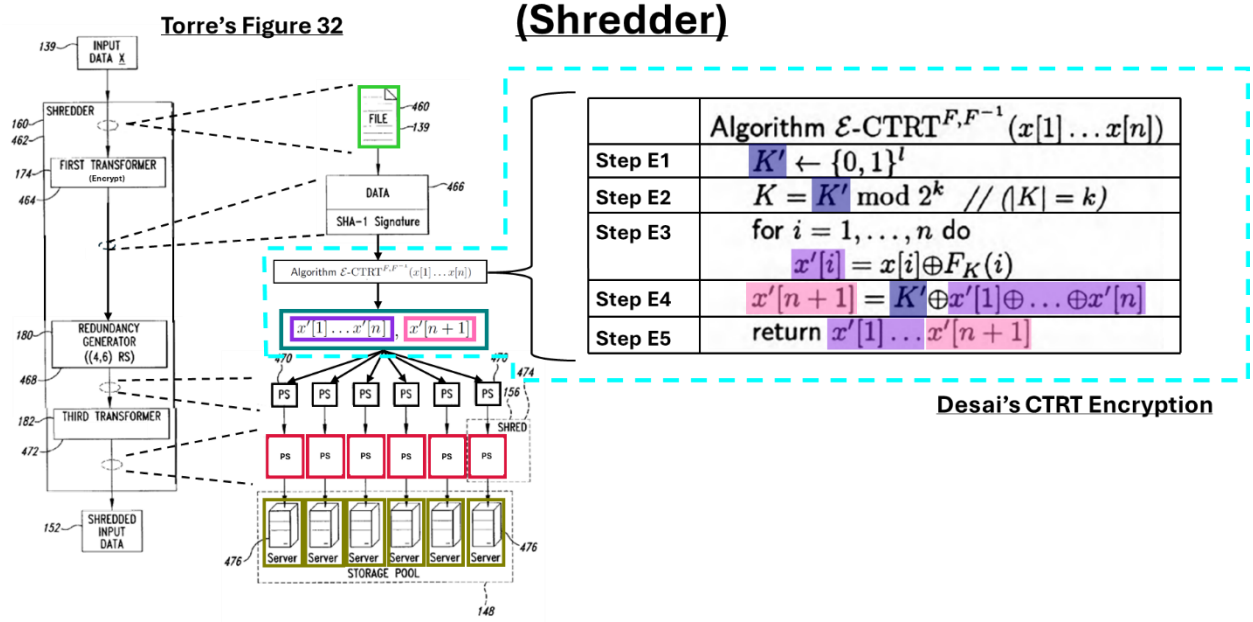
3. Torre In Combination With Desai ("Torre/Desai System")

Torre is directed to securely storing input data by parsing and splitting the input data into shreds. *See* Section IX.A.1 [Torre]. It would have been obvious to augment Torre's shredder (in both Torre System-1 and Torre System-2) with Desai's CTRT technique to further improve the security of the stored data. EX1002, ¶213.

Specifically, it would have been obvious to implement Desai's E-CTRT$^{F,F^{-1}}$ transformation algorithm in Torre's shredder, specifically between Torre's first transformer and redundancy generator so that Desai's CTRT algorithm transforms the data 466 (represented by data blocks $x[1]$ ... $x[n]$) into a data set consisting of encrypted data blocks $x'[1]$ … $x'[n]$ and data block $x'[n+1]$. The below figure shows the Torre/Desai System's shredder. EX1002, ¶214.

Torre/Desai System

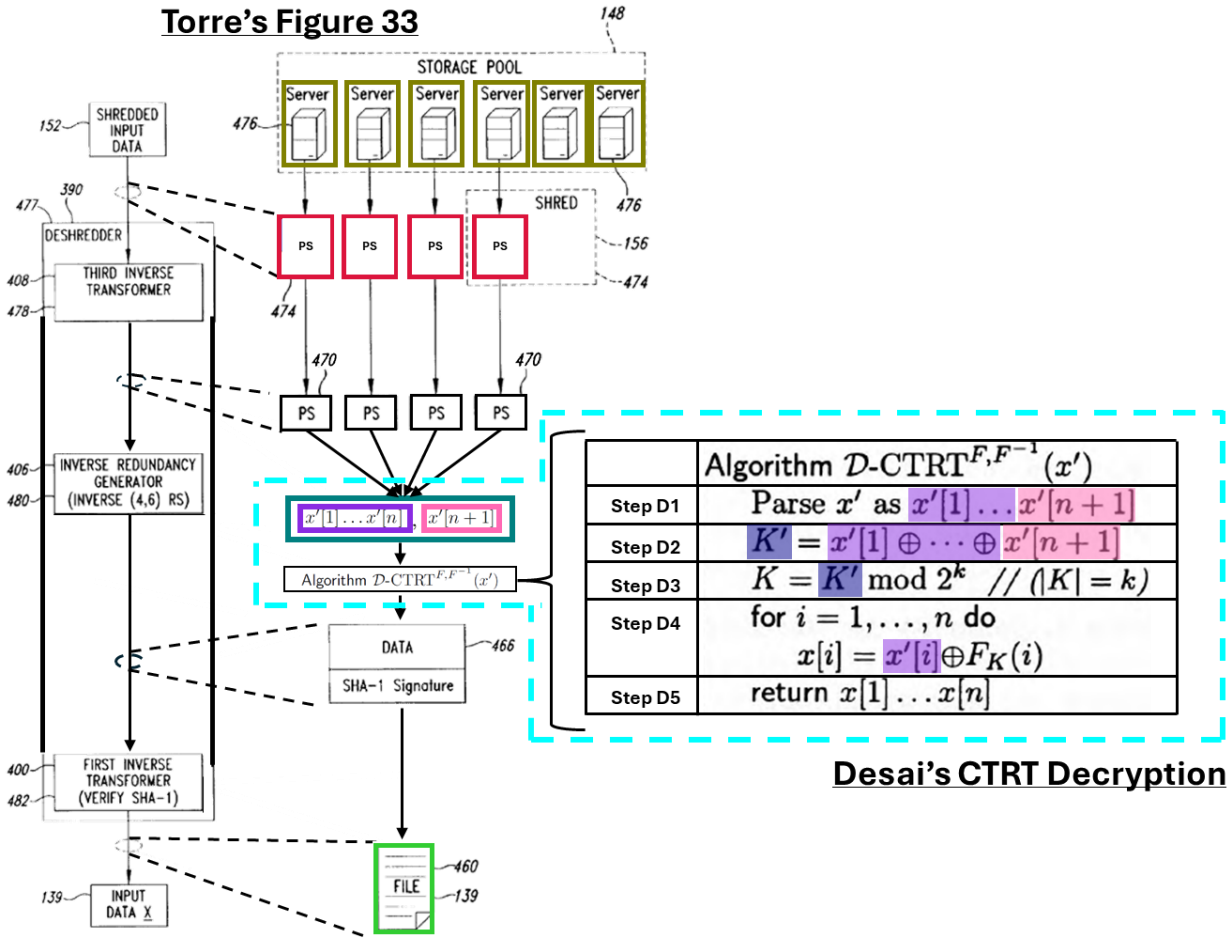
(Shredder)



EX1003, FIG. 32; EX1004, 16.

Likewise, it would have been obvious to augment Torre's deshredder in a corresponding way. Specifically, it would have been obvious to implement Desai's *D*-CTRT$^{F,F^{-1}}$ transformation algorithm Torre's deshredder, specifically in between Torre's first inverse transformer and inverse redundancy generator so that the algorithm uses encrypted data blocks $x'[1] \dots x'[n]$ and the data block $x'[n+1]$ to recover the data 466 (represented by the data blocks *x[1] ... x[n]*). The below figure shows the Torre/Desai System's deshredder. EX1002, ¶215.

Torre/Desai System
(Deshredder)



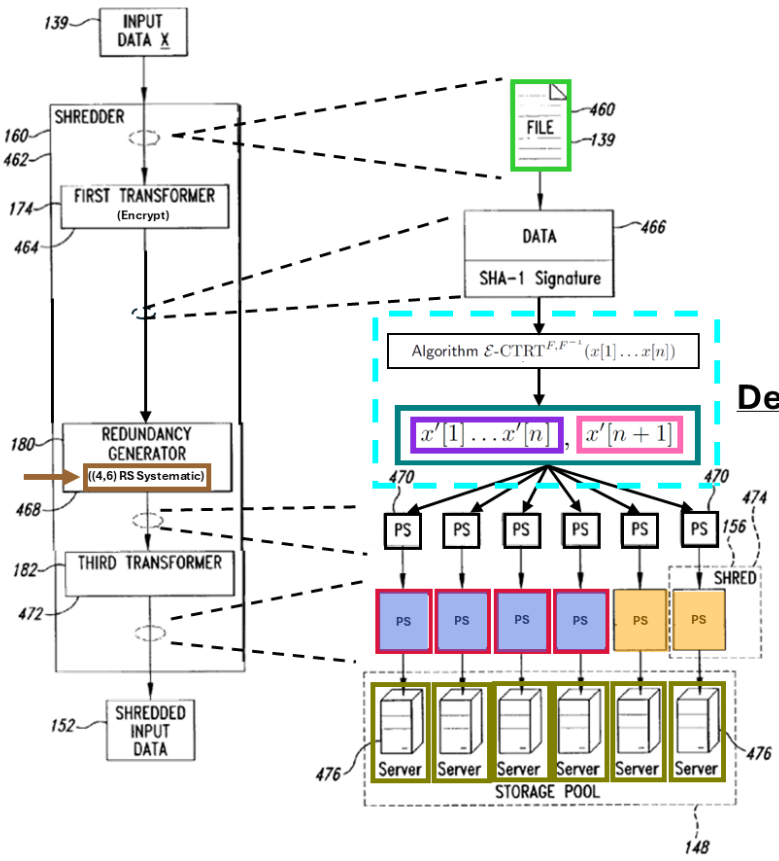
EX1003, FIG. 33; EX1004, 16.

i. Torre/Desai System-1

For the same reasons, it would have been obvious to augment Torre System-1 with Desai's CTRT technique to further improve the security of the stored data. Such system is herein referred to as "Torre/Desai System-1." EX1002, ¶216.

Torre/Desai System-1 (Shredder)

Torre's Figure 32

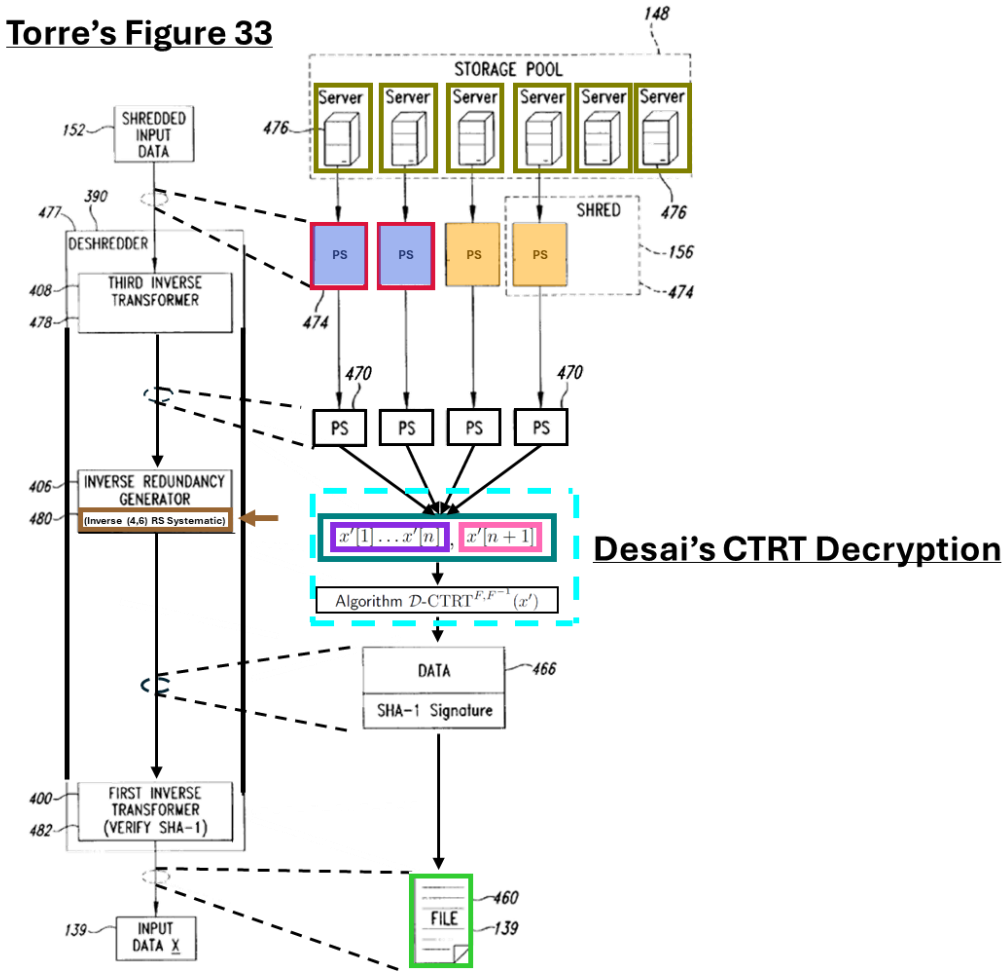


Desai's CTRT Encryption

EX1003, FIG. 32; EX1004, 16.

Torre/Desai System-1
(Deshredder)

Torre's Figure 33

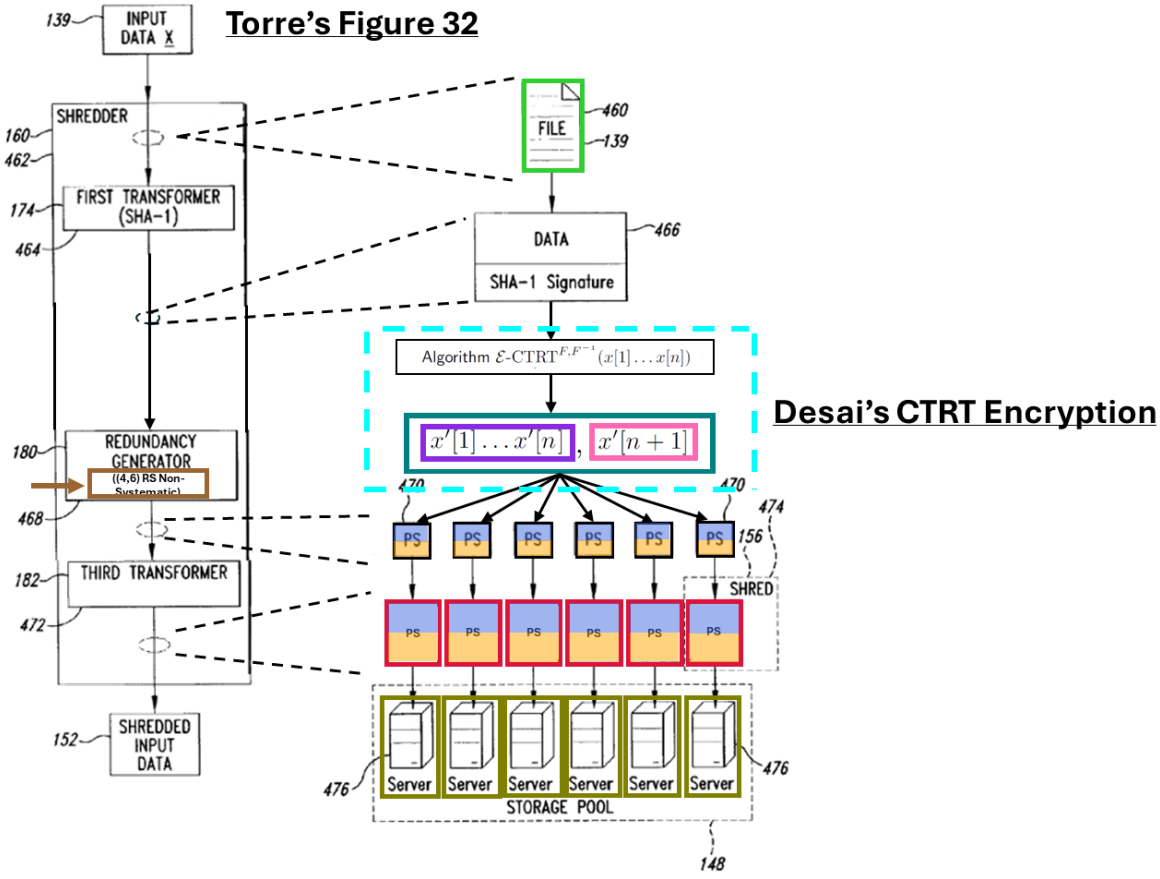


EX1003, FIG. 33; EX1004, 16.

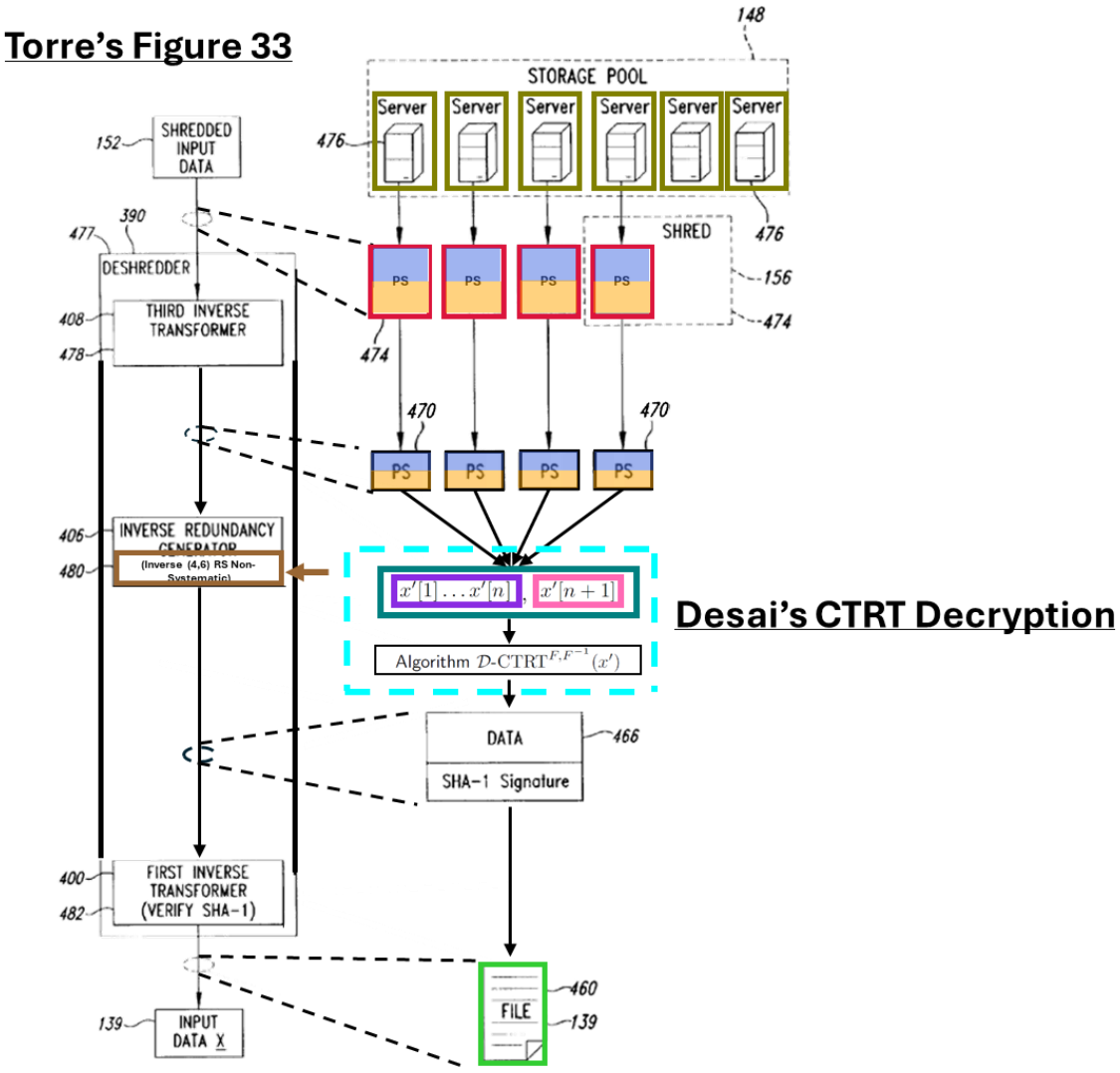
ii. Torre/Desai System-2

For the same reasons, it would have been obvious to augment Torre System-2 with Desai's CTRT technique to further improve the security of the stored data. Such system is herein referred to as "Torre/Desai System-2." EX1002, ¶217.

Torre/Desai System-2
(Shredder)



EX1003, FIG. 32; EX1004, 16.



EX1003, FIG. 33; EX1004, 16.

iii. Reasons To Combine Torre and Desai

A POSITA would have been motivated to combine Torre with Desai as shown in the Torre/Desai System. ***First***, both Torre and Desai are directed to solving a substantially similar problem in a substantially similar way. For example, Torre and Desai are both directed to securely encrypting a data file as blocks of data such that a threshold number of blocks of data can subsequently be decrypted to obtain the

original data file. *See* EX1003, [0049] ("A system and method for shredding and deshredding [data] … to store and retrieve [the] data while emphasizing considerations such as redundancy, integrity, and security."), [0051], [0053]; EX1004, 6 ("[T]he preferred approach might be to squeeze a little more security out of the existing encryption modes using some efficient pre-processing techniques"), 7 ("[I]t should be infeasible to determine even one message block … without decrypting all the [encrypted] blocks … Given the output sequence of [encrypted] blocks, one can easily obtain the original sequence of blocks."), 16 (explaining that encryption "Algorithm E-CTRT$^{F,F^{-1}}$(x[1] … x[n])" will "return x'[1] … x'[n+1]," while decryption "Algorithm D-CTRT$^{F,F^{-1}}$ (x')" will "parse x' as x'[1] … x'[n+1]" and "return x[1] … x[n]"). *See also* Sections IX.A.1, IX.A.2. EX1002, ¶¶218–221.

Second, a POSITA would have been motivated to implement Desai's CTRT algorithms in Torre because Torre expressly motivates that combination. Torre discloses that its "shredding and the shredding system and method discussed herein is envisioned to be used with ***any other encryption method that provides a desired level of security*** …." EX1003, [0085] (emphasis added). Torre further teaches that "[s]ecurity is also becoming more necessary for conventional systems to prevent unauthorized disclosure of stored or transmitted data." EX1003, [0005]. *See also* EX1003, [0055]. Accordingly, a POSITA desiring a level of security that would prevent unauthorized disclosure would have been motivated to seek suitable

methods for encrypting data as blocks, and would have understood that Desai teaches such a method that—through the addition of an AONT pre-processing step—provides an improved level of security. EX1002, ¶¶222–223.

Third, a POSITA would have been motivated to implement Desai's CTRT algorithms in Torre because Desai also motivates that combination. As Desai explains, block cipher encryptions are vulnerable to brute-force key search attacks. EX1004, 6–7. And Desai explains that its CTRT technique—when used as a "pre-processing step" to existing data security techniques—can address such vulnerability. EX1004, 6–7. So, a POSITA would have readily understood that Desai's CTRT technique can be applied to Torre's system to improve the security of shreds stored in storage devices. EX1002, ¶224.

Fourth, a POSITA would have been motivated to implement Desai's CTRT technique in Torre's shredder so that even if a third party gets access of sufficient number of shreds 156 to recover the pre-shred 466, the pre-shred 466 would still be obfuscated by CTRT. Without knowing how the CTRT was applied (e.g., the cipher *F*, key length *k* and block length *l*), a POSITA would not be able to recover any one of the original data block x[i] thanks to the non-separable characteristic of AONT techniques like Desai's CTRT technique. That further improves security. EX1002, ¶225.

Fifth, a POSITA would have been motivated to use Desai's CTRT technique over other variations of AONT, like Rivest's original AONT, because Desai teaches that the CTRT technique is just as secure, but is computationally more efficient. *See* EX1004, 9 ("[W]e can now get all-or-nothing encryption modes that cost only two times the cost of normal CBC encryption, while with the package transform, the resulting modes had cost about three times the cost of CBC"), 21. EX1002, ¶226.

Further, Desai's CTR technique would have been particularly attractive in Torre's system. Alternative methods of improving resistance to key-search attacks—such as using longer keys for encryption to increase the number of possible key permutations an attacker would have to test —generally "necessitate changing the existing cryptographic hardware and software implementing these encryption modes." EX1004, 6. In contrast, Desai's CTRT algorithm is able to successfully "increase the difficulty of an exhaustive key search on symmetric encryption schemes, while keeping the key size the same and not overly burdening the legitimate users." EX1004, 6–7. EX1002, ¶227.

Accordingly, a POSITA would have been motivated to implement a known technique (Desai's CTRT algorithms) to improve the security of a known device (Torre's System) in the same way (transforming the input data using Desai's CTRT algorithm to further obfuscate the input data and ensure non-separability) to obtain the predictable result of improved security. EX1002, ¶228.

A POSITA would have had a reasonable expectation of success in implementing Desai's CTRT algorithms in Torre. EX1002, ¶229.

First, Desai teaches that AONT, such as its CTRT algorithm, is a "pre-processing step" that improves existing data security systems. EX1004, 7. Accordingly, a POSITA would have understood that implementing Desai's CTRT algorithms in Torre would have merely entailed performing a known CTRT transformation to preprocess Torre's input data in the shredding process (and to perform the reverse operations in the deshredding process) to yield the predictable result of enabling a non-separable data processing and obfuscation of input data to improve security. EX1004, 15–16. EX1002, ¶230.

Second, Torre explains that "the embodiments disclosed herein … can be equivalently implemented … as one or more ***computer programs*** running on one or more computers." EX1003, [0141]. Accordingly, implementing Desai's CTRT technique in Torre System would have merely involved writing a program code. That was well within the skill of a POSITA. EX1003, [0141]; *Keynetik, Inc. v. Samsung Elecs. Co., Ltd.*, No. 2022-1127, 2023 WL 2003932, at *2 (Fed. Cir. Feb. 15, 2023) ("Normally, once the function to be performed by software has been identified, writing code to achieve that function is within the skill of the art."). EX1002, ¶231.

At a minimum, it would have been obvious to try implementing Desai's CTRT algorithm in Torre's system. As Desai explains, block cipher encryptions are vulnerable to brute-force key search attacks—a vulnerability addressed by implementing AONT as a pre-processing step to encryption. EX1004, 6–7. Moreover, AONT can be applied as a pre-processing step to any existing block cipher encryption without any changes to the encryption algorithm. EX1004, 6–7. Accordingly, it would have been obvious for a POSITA to include AONT as a pre-processing step to the block cipher encryption functions performed in Torre's shredder in order to improve the security of Torre's encryption. EX1002, ¶232.

* * *

The reasons for combining the Torre System and Desai described above are equally applicable to combining Torre System-1 and Desai; and Torre System-2 and Desai because the combination does not implicate the operation of a redundancy generator. EX1002, ¶233.

Torre/Desai System-1 and Torre/Desai System-2 are identical except for the specific implementation of the redundancy generator. Thus, in the forthcoming sections, when the analysis does not implicate the specific operation of the redundancy generator, the analysis refers to Torre/Desai System-1 and Torre/Desai System-2 collectively as "Torre/Desai System." EX1002, ¶234.

4. Claim 1

Torre/Desai System renders claim 1 obvious. EX1002, ¶235.

i. [1Pre] A method performed by a computer system for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the method comprising:

Torre/Desai System teaches [1Pre]. EX1002, ¶¶236, 247.

First, Torre/Desai System teaches "*[a] method … for securing a first data set*."[2] Torre describes "[a] system and method for shredding and deshredding … to ***store and retrieve data*** while emphasizing considerations such as redundancy, integrity, and ***security of data***." EX1003, [0049]. Torre's "shredding system and method … is envisioned to be used with any other encryption method that provides a ***desired level of security*** dependent upon such other factors as proper key management." EX1003, [0085]. *See also* EX1003, [0095]–[0096], [0109]. Also, "during allocation, certain particular bytes or other groupings of data may be rearranged based upon particular processing to occur in subsequent steps or for certain portions of the data to be secretly mapped to certain storage units for reasons such as ***enhanced security***." EX1003, [0109]. In the context of Torre's Figure 32,

[2] The quoted claim language is identified using *unbolded italics*.

Torre teaches receiving and processing the input data 139 so it can be securely stored. EX1003, [0131]–[0132]. EX1002, ¶¶237–238.

The input data stored by Torre (and hence the Torre/Desai System) is a "*data set*." Torre explains that "[t]he input data can be a ***file or another form***" and can be generated "by operating software programs and by receiving or recording data such as live audio or ***video***, or audio or ***video*** broadcast streams." EX1003, [0050]. Thus, the input data may be a video comprising a set of video frames. EX1016 (Microsoft Computer Dictionary), 144 ("data set" definition 1); EX1020, 2. The set of video frames satisfies the claimed "*data set*" because it is a set of data (frames). EX1002, ¶239.

Indeed, in Patent Owner's litigation against Google, Patent Owner broadly construed "*data set*" to mean "collection of information for storage." EX1013, 8. The "input data" in Torre/Desai satisfies Patent Owner's construction of "data set" because Torre's input data can be a video file with a collection of video frames, and each frame is "information for storage"—accordingly, the video file being input to Torre's shredding method is a "collection of information for storage." EX1002, ¶¶240–241.

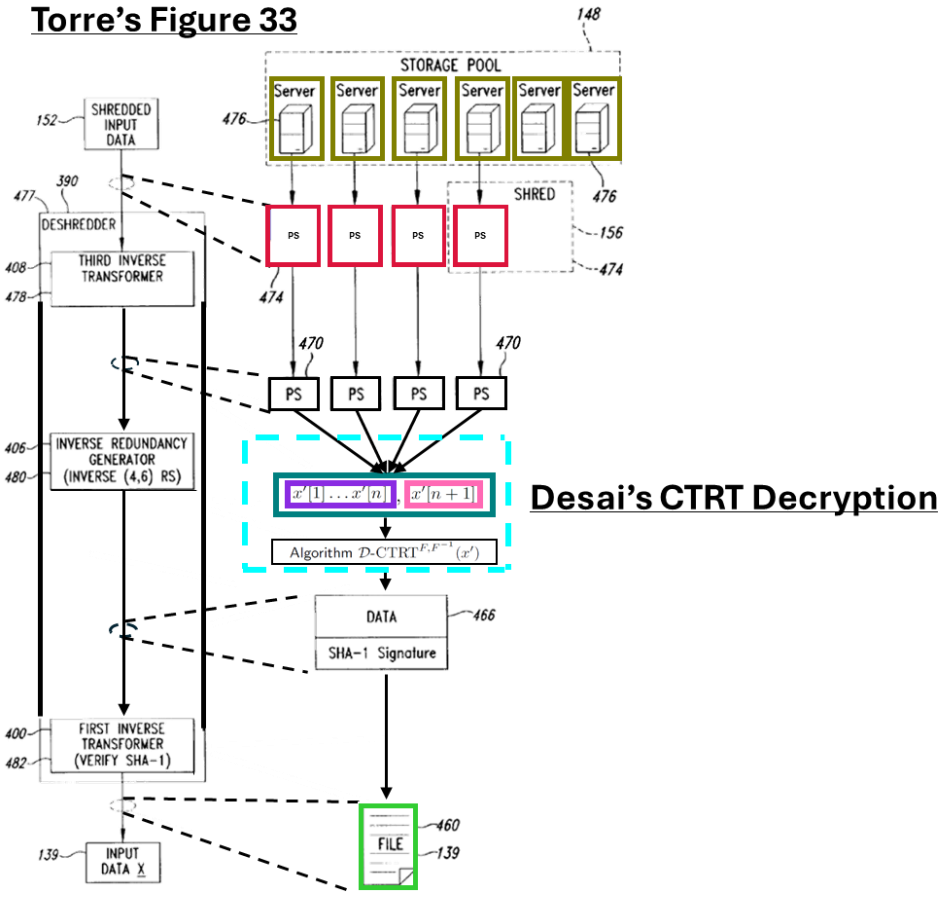
Second, Torre/Desai System teaches that the method also "*provid[es] recoverability of the first data set in the event of unavailability of data at one or more storage locations*." Torre explains that "[i]n the process of shredding, input

data is generally transformed into many smaller pieces of data called shreds." EX1003, [0051]. "Shredding is done in a manner that adds redundancy to the shreds so that the ***input data*** can be recovered from a ***fewer number of shreds*** than the total number stored." EX1003, [0051]. *See also* EX1003, [0131]. EX1002, ¶¶242–243.

For example, Torre's deshredder only requires a subset of the shreds to recover the original data (e.g., the shreds from four of the six servers shown in Figure 33). EX1002, ¶¶244.

Torre/Desai System (Deshredder)

Torre's Figure 33



Desai's CTRT Decryption

EX1003, FIG. 33; EX1004, 16.

Torre explains that recreating the original data using less than all of the shreds has the benefit of enabling recovery of the original data even if one of the servers is nonfunctional. "[A] general purpose for the redundancy generator 180 is to make the shredded input data 106 ***fault-tolerant***. The end result of the redundancy generator is to generate data redundant to the input data 139 so that ***some of the shredded input data 106 that is stored in storage pool 148 could be lost without preventing the input data from being recovered***." EX1003, [0111]. As a result, the storage pool 148 storing the shredded data "could be viewed as a ***giant highly fault-tolerant drive*** in which some of the shreds in some of the storage could be lost without losing function of the giant drive." EX1003, [0069]. EX1002, ¶245.

Third, Torre/Desai System teaches that the "*method [is] performed by a computer system*." Torre's teachings may be implemented "as one or more ***computer programs running on one or more computers***." EX1003, [0141]. *See also* EX1003, [0142]. EX1002, ¶246.

ii. [1A] reading the first data set from a memory that stores the first data set;

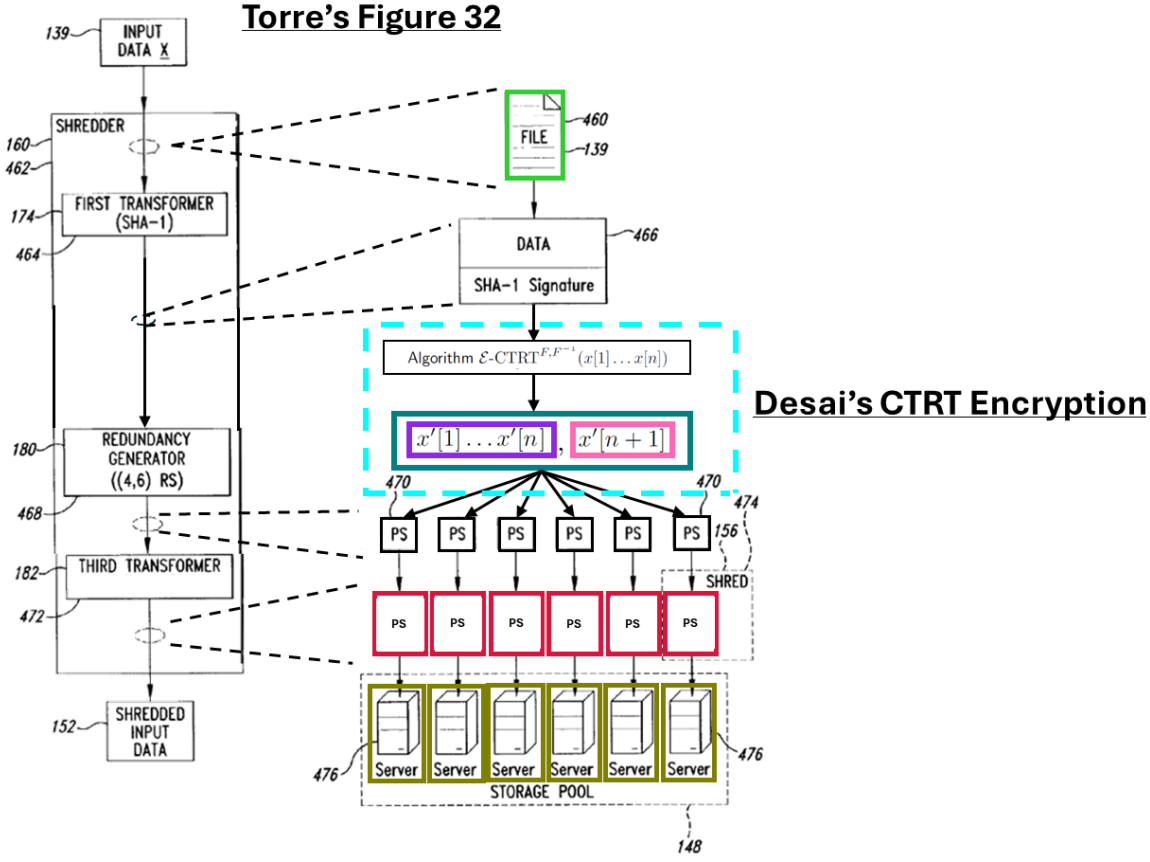
Torre/Desai System teaches [1A]. EX1002, ¶¶248, 252.

Torre/Desai System teaches a shredder that receives the first data set. For example, in Torre/Desai System's shredder, "a first transformer 464" "receives the

input data 139 consisting in this example as a data file 460" ("*first data set*"). EX1003, [0131]. *See also* EX1003, [0050]. Torre explains that "[t]he input data can be a file or another form that has been generated directly or by other means" and "[g]eneration of data includes, but is not limited to, generation by operating software programs and by receiving or recording data such as live audio or video, or audio or video broadcast streams." EX1003, [0050]. EX1002, ¶249.

Torre/Desai System (Shredder)

Torre's Figure 32



EX1003, FIG 32; EX1004, 16.

A POSITA would have understood that Torre's shredder can receive the first data set by, in part, retrieving it from a memory because a processing circuitry like the shredder typically stores the received data—such as a video stream—in a memory before it subsequently processes the received data. *See, e.g.*, EX1026, 2 ("For a video server, buffers are used as an intermediate to cache and deliver video data from the storage to the network interface."); EX1011, 1 ("A computer's memory system is the repository for all the information the computer's central processing unit (CPU, or processor) uses and produces."); EX1012, 26 ("[M]emory refers to the temporary internal storage areas within a computer … The main type of memory and the most familiar to users is random access memory (RAM). RAM is the same as main memory. A computer can both write data into RAM and read data from RAM."); EX1005, [0053] ("Data prepared by the user or provided by the information processing unit itself can be stored in the memory device for use…"). Indeed, Torre already teaches a memory that can maintain the input data. *See, e.g.*, EX1003, [0068] ("[If] a computer has system ***memory***, various portions of this ***memory*** can be used as the storage units 150."). EX1002, ¶¶250–251.

iii. [1B] processing the first data set to produce a second data set,

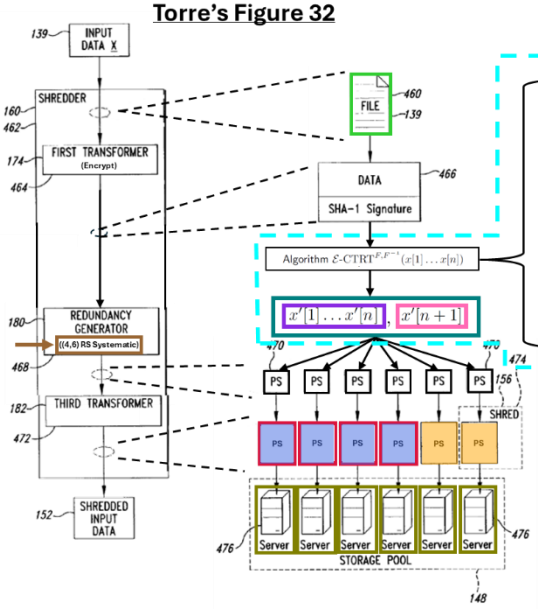
Torre/Desai System teaches [1B]. EX1002, ¶¶253, 256.

In Torre/Desai System, "a first transformer 464" "receives the input data 139 consisting in this example as a data file 460" ("*a first data set*"). EX1003, [0131]. The first transformer 464 then "generate[s] a pre-shred 466 containing the received data and a tag of the SHA-1 signature." EX1003, [0131]. EX1002, ¶254.

Next, the shredder of the Torre/Desai System processes the pre-shred 466 using Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm. Specifically, the pre-shred 466 containing the input data (the "*first data set*") "is regarded as a sequence of *l*-bit blocks, $x = x[1] \ldots x[n]$." EX1004, 16. Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm transforms the *x* blocks to produce $x' = x'[1] \ldots x'[n+1]$ (the "*second data set*"). EX1002, ¶255.

Torre/Desai System-1
(Shredder)

Torre's Figure 32

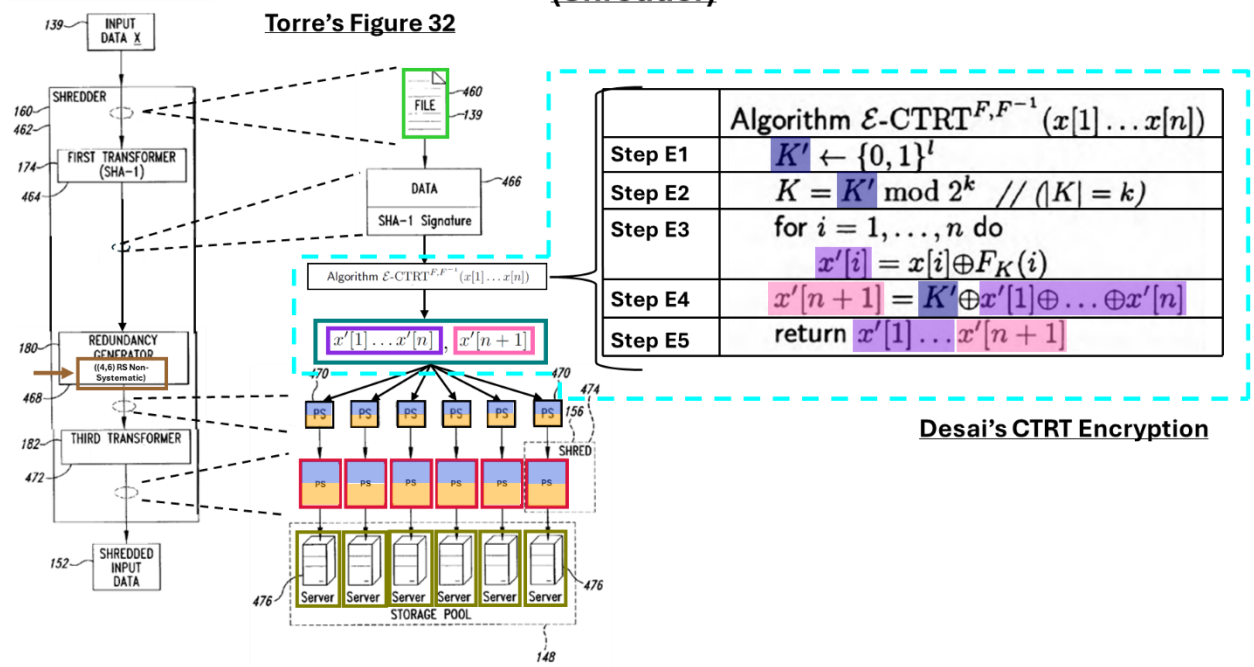


	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1, \ldots, n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

Desai's CTRT Encryption

EX1003, FIG. 32; EX1004, 16.

Torre/Desai System-2
(Shredder)



EX1003, FIG. 32; EX1004, 16.

a. [1B-1] the processing comprising … encrypting the first data set using an encryption key to produce an encrypted data set, and

Torre/Desai System teaches [1B-1]. EX1002, ¶¶257, 263.

As explained above, the shredder of Torre/Desai System has a first transformer 464 that "generate[s] a pre-shred 466 containing the input data" ("*the first data set*"). EX1003, [0131]. Next, Desai's *E*-CTRT$^{F,F^-}$ processes the pre-shred 466 containing the input data, as shown below. EX1002, ¶258.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm "*encrypt[s] the first data set* [the input data] *using an encryption key* [*K'*] *to produce an encrypted data set* [*x'*[1] … *x'*[*n*]]." For example, Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm processes the pre-shred 466 containing the input data "as a sequence of *l*-bit blocks, *x* = *x*[1] … *x*[*n*]." EX1004, 16. EX1002, ¶259.

At **step E1**, the algorithm generates a key *K'* (the "*encryption key*") as a random series of bits of length *l*. EX1004, 16. EX1002, ¶260.

At **step E2**, the algorithm computes *K'* modulo 2^k to obtain the truncated encryption key *K*, where *k* is the key length accepted by the cipher *F*. EX1004, 16. And at **step E3**, the algorithm encrypts each of the data blocks *x* = *x*[1] … *x*[*n*] by applying the cipher *F* with the key *K* on each ith block to obtain encrypted data blocks *x'*[*1*] ... *x'*[*n*] ("*an encrypted data set*"). EX1004, 16. EX1002, ¶¶261–263.

Accordingly, Torre/Desai teaches "*encrypting the first data set* [the input data represented as the data blocks x[1]…x[n]] *using an encryption key* [*K'*] *to produce an encrypted data set* [x'[1] … x'[n]]." EX1002, ¶263.

b. [1B-2] the processing comprising … logically combining at least a portion of the encrypted data set with the encryption key to produce a resultant,

Torre/Desai System teaches [1B-2]. EX1002, ¶¶264, 268.

For example, after step E3 described in Section IX.A.4.iii.a, in Step E4, Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm computes the exclusive-OR operation ("XOR," denoted by the symbol "⊕") of the encryption key *K'* with each of the encrypted data blocks *x'[1] … x'[n]* (*"at least a portion of the encrypted data set"*) to produce a new data block *x'[n+1]* ("*a resultant*"). EX1004, 16. EX1002, ¶265.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

This XOR operation of step E4 discloses the claimed "*logically combining*" operation because XOR is a logical operator. *See, e.g.,* EX1023, 3 ("The term combinational **logic** refers to circuitry that transforms bits, as opposed to storing bits."), 5 ("Here we have inputs X and Y, output Z, with Z being equal to 1 if the statement 'X = 1 or Y = 1 but not both' is true. The term used for this is 'exclusive-or,' abbreviated to **XOR**."). EX1002, ¶266.

Indeed, claim 11 of the '456 Patent states that "logically combining comprises logically combining using an exclusive-OR operation." EX1001, Claim 11. EX1002, ¶267.

c. [1B-3] wherein the second data set comprises the encrypted data set and the resultant, and wherein all of the at least a portion of the encrypted data set is required to recover the encryption key based on the resultant;

Torre/Desai System teaches [1B-3]. EX1002, ¶¶269, 275.

First, Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm discloses "*wherein the second data set comprises the encrypted data set and the resultant*." For example, **Step E5** of Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm returns $x' = x'[1] \ldots x'[n+1]$ (the "*second data set*") that comprises $x'[1] \ldots x'[n]$ (the "*encrypted data set*") and $x'[n+1]$ (the "*resultant*"). Below is an annotated version of Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm. EX1002, ¶270.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // ($\|K\| = k$)
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

Second, Desai's *D*-CTRT$^{F,F^{-1}}$ transformation algorithm discloses "*wherein all of the at least a portion of the encrypted data set is required to recover the encryption key based on the resultant*." Below is an annotated version of Desai's *D*-CTRT$^{F,F^{-1}}$ transformation algorithm. EX1002, ¶271.

	Algorithm $\mathcal{D}$-CTRT$^{F,F^{-1}}(x')$
Step D1	Parse x' as $x'[1]\ldots x'[n+1]$
Step D2	$K' = x'[1] \oplus \cdots \oplus x'[n+1]$
Step D3	$K = K' \bmod 2^k$ // ($\|K\| = k$)
Step D4	for $i = 1,\ldots,n$ do $x[i] = x'[i] \oplus F_K(i)$
Step D5	return $x[1]\ldots x[n]$

EX1004, 16.

At **step D1**, the algorithm "[p]arse[s] *x'* as *x'*[1] … *x'*[*n*+*1*] (the "*second data set*"). EX1004, 16. At **step D2**, the algorithm recovers *K'* ("*the encryption key*") by

computing the XOR (denoted by the symbol "⊕") of *x′*[1] … *x′*[*n*] ("*at least a portion of the encrypted data set*") and *x′*[*n*+1] (the "*resultant*"). EX1004, 16. EX1002, ¶272.

Here, the algorithm requires that all of the encrypted data blocks *x′*[1] … *x′*[*n*] and the resultant block *x′*[*n*+1] to reverse the XOR operation and obtain the encryption key *K′*. EX1004, 7 ("AONT is an efficiently computable transformation, mapping sequences of blocks to sequences of blocks" where "***[g]iven all but one of the output sequence of blocks, it is computationally infeasible to determine any function of any input block***."). This is the very characteristic of an all-or-nothing transform. *See* EX1010, 3, 5. EX1002, ¶¶273–274.

Accordingly, Desai teaches "*wherein all of the at least a portion of the encrypted data set* [*x′*[1] … *x′*[*n*]] *is required to recover the encryption key* [*K′*] *based on the resultant* [*x′*[*n*+1]]." EX1002, ¶275.

iv. [1C] producing redundancy information based on information in the second data set;

Both Torre/Desai System-1 (using systematic encoding for redundancy generation) and Torre/Desai System-2 (using non-systematic encoding for redundancy generation) satisfies [1C]. EX1002, ¶276.

a. Torre/Desai System-1

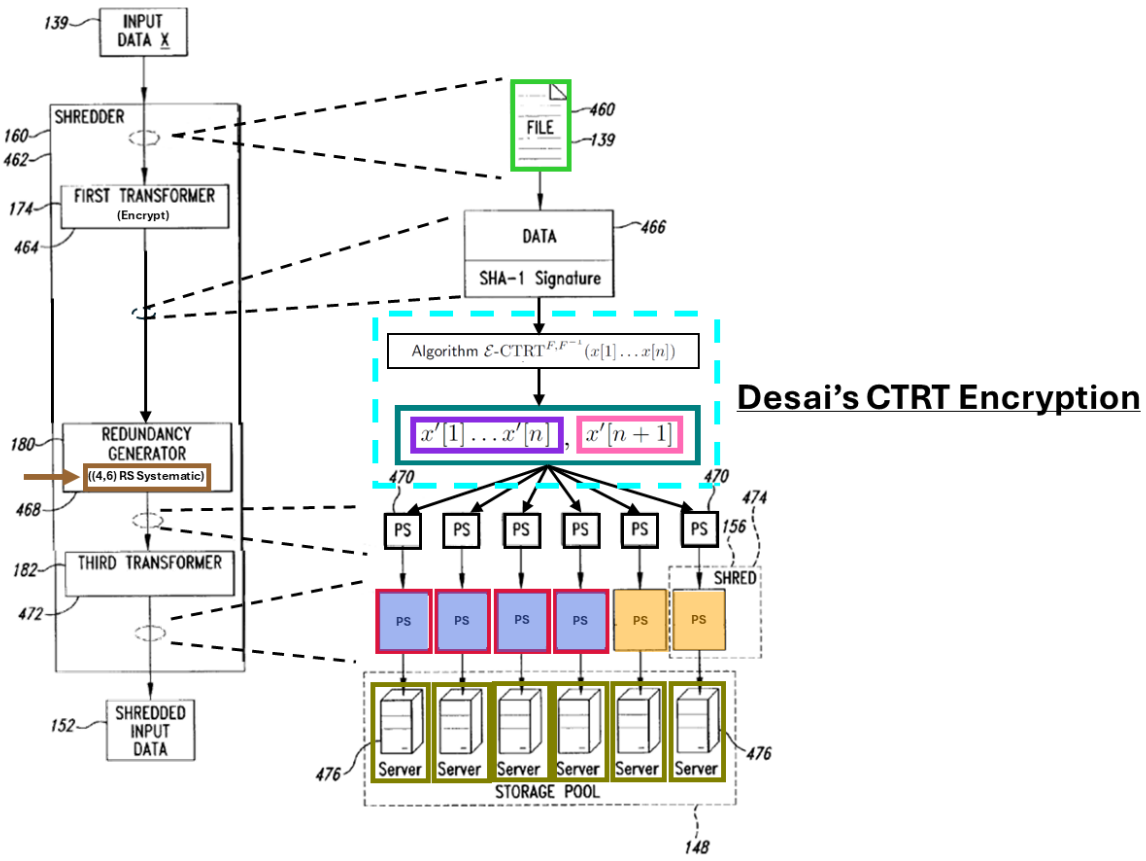
Torre/Desai System-1 satisfies [1C]. EX1002, ¶¶277, 280.

Torre/Desai System-1 teaches "*producing redundancy information based on information in the second data set.*" For example, Torre/Desai System-1's shredder uses Desai's *E*-CTRT$^{F,F-1}$ to process pre-shred 466 containing the input data to produce $x' = x'[1] \ldots x'[n+1]$ (a "*second data set*"). The redundancy generator 468 in the shredder then receives the "*second data set*" $x' = x'[1] \ldots x'[n+1]$ and produces redundancy information based on the "*second data set*" using systematic encoding. EX1003, [0117] ("The redundancy generator 180 also generates two outputs of redundant data from the received data and outputs the two outputs of redundant data along with the four outputs of the received data."); EX1003, [0131] ("The redundancy generator 468 … uses a 4 input-6 output Reed Solomon redundancy generation function with systematic encoding to generate six pre-shreds 470 with four of the pre-shreds containing direct pass-through data blocks of the data file and two of the pre-shreds being generated from combinations of data blocks from the data file ***for redundancy purposes***."); Figure 21. *See also* Section IX.A.1.i [Torre System-1] (describing redundancy generation using systematic encoding). EX1002, ¶¶277–278.

As shown below in an annotated Figure 32 of Torre, the redundancy information corresponds to two of the six shreds generated by the 4 input-6 output Reed Solomon redundancy generator using systematic encoding. EX1002, ¶279.

Torre/Desai System-1
(Shredder)

Torre's Figure 32



Desai's CTRT Encryption

EX1003, FIG. 32; EX1004, 16.

b. Torre/Desai System-2

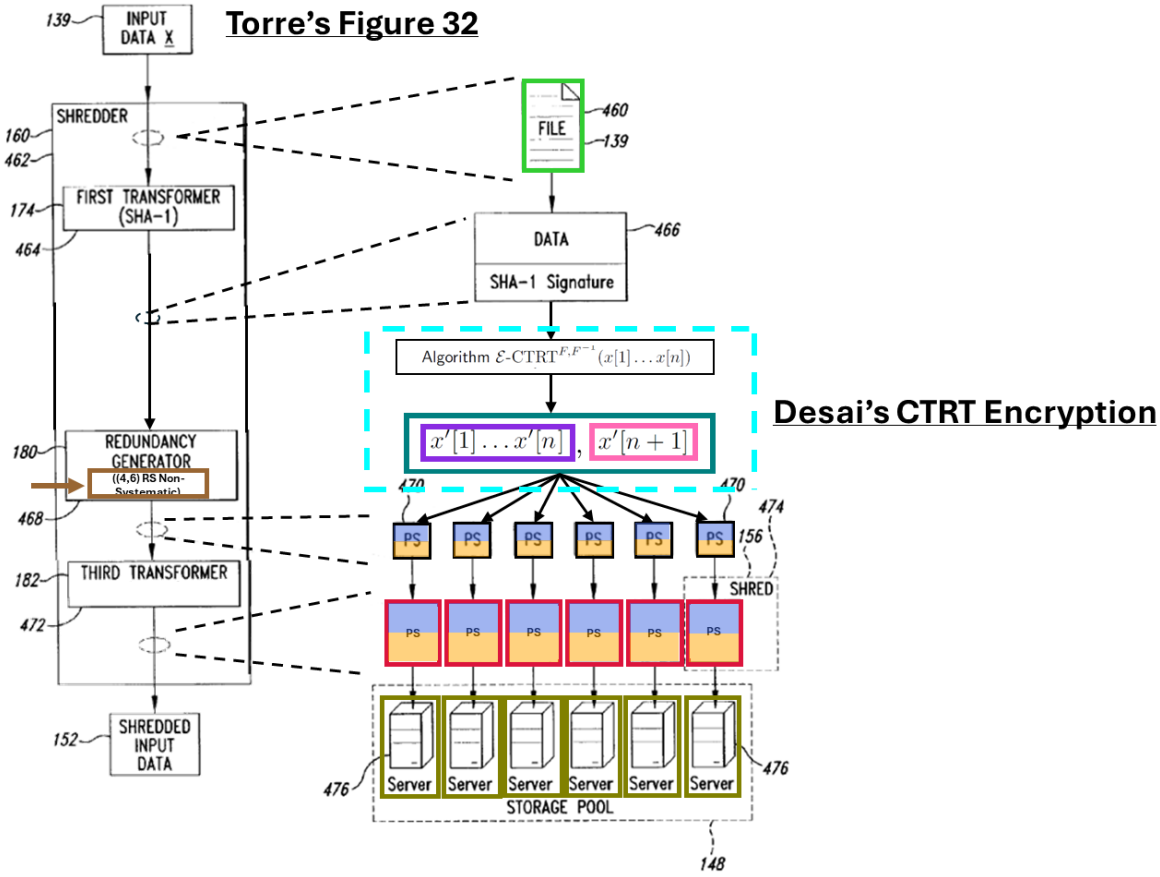
Torre/Desai System-2 satisfies [1C]. EX1002, ¶¶281, 283.

The Torre/Desai System-2 also meets "*producing redundancy information based on information in the second data set*." Torre/Desai System-2's shredder processes the input data using Desai's algorithm to produce $x' = x'[1] \ldots x'[n+1]$ (the "*second data set*"). Here, however, the redundancy generator 468 produces the

redundancy information based on the "*second data set*" *x′*[1] … *x′*[*n*+*1*] using ***non-systematic*** encoding. Torre explains that the redundancy generator implementing non-systematic encoding "generates six outputs of redundancy-generated data from four inputs of received data." EX1003, [0117]; *see also id.* FIG. 22. EX1002, ¶281.

As shown below in an annotated Figure 32 of Torre, the redundancy information is ***part of*** the six shreds generated by the 4 input-6 output Reed Solomon redundancy generator using non-systematic encoding. *See* EX1022; EX1019, 8. EX1002, ¶282.

Torre/Desai System-2
(Shredder)



EX1003, FIG. 32; EX1004, 16.

v. [1D] producing a plurality of n shares from the second data set, wherein each of the n shares comprises at least some of the second data set; and

Both Torre/Desai System-1 (using systematic encoding for redundancy generation) and Torre/Desai System-2 (using non-systematic encoding for redundancy generation) satisfies [1D]. EX1002, ¶284.

a. Torre/Desai System-1

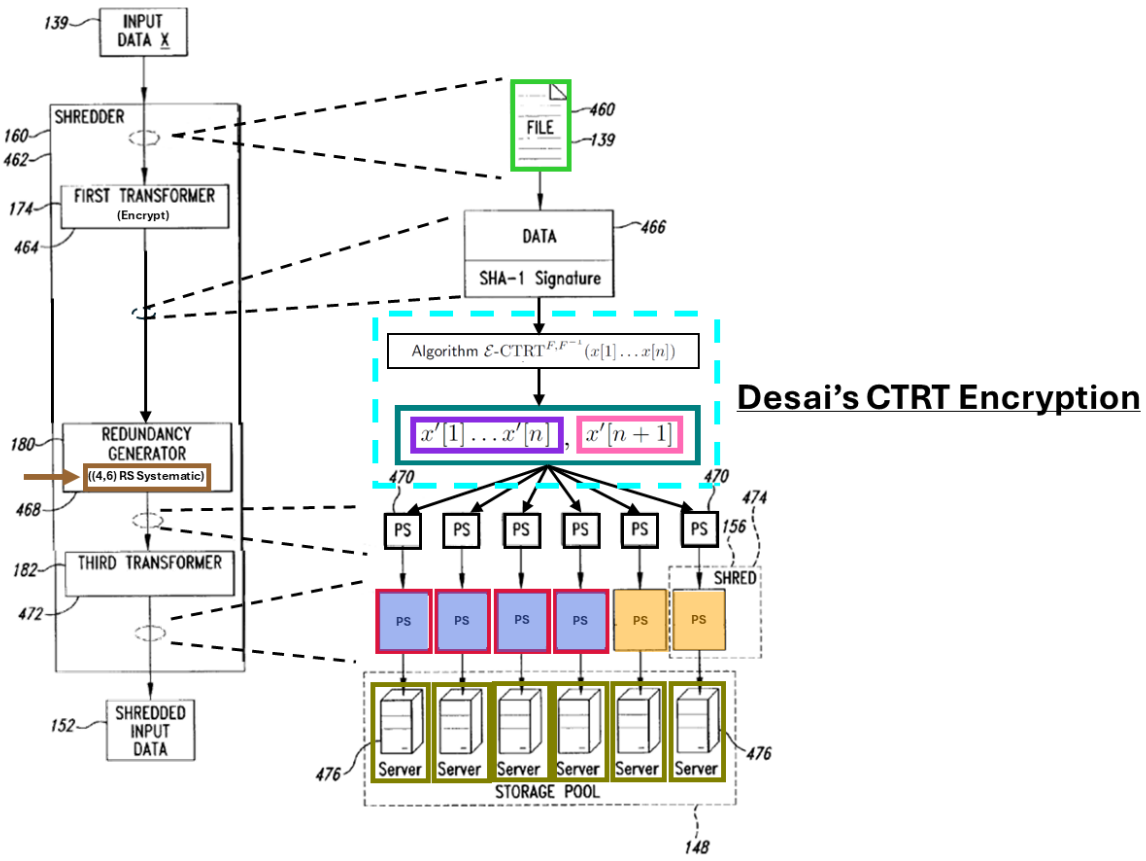
Torre/Desai System-1 satisfies [1D]. EX1002, ¶¶285, 288.

First, Torre/Desai System-1's shredder "*produc[es] a plurality of n shares from the second data set*." For example, as explained in Section IX.A.3.i, Torre/Desai System-1's shredder includes a redundancy generator 468 implementing systematic encoding that processes the "*second data set*" $x' = x'[1] \ldots x'[n+1]$ "to generate six pre-shreds 470 with four of the pre-shreds containing direct pass-through data blocks of the data file." EX1003, [0131]. EX1002, ¶285.

The "four of the pre-shreds" are further processed by the shredder's "third transformer(s) 472" to "form shreds 474, which includes the pre-shred data." EX1003, [0132]. As shown in Figure 32 of Torre (below), each shred includes pre-shred data from a corresponding one of the pre-shreds. Therefore, the third transformer produces ***four*** shreds 474 that each include "direct pass-through data blocks" of the "*second data set*" $x' = x'[1] \ldots x'[n+1]$. EX1003, [0132]. EX1002, ¶286.

Torre/Desai System-1
(Shredder)

Torre's Figure 32



EX1003, FIG. 32; EX1004, 16.

Accordingly, Torre/Desai System-1's shredder "*produc[es] a plurality of n shares* [four shreds] *from the second data set*." EX1002, ¶¶285, 288.

Second, in Torre/Desai System-1, "*each of the n shares comprises at least some of the second data set*." As explained above in this Section, the four shreds include pre-shred data from a corresponding one of the pre-shreds, which, in turn, contain data from the "*second data set*" $x' = x'[1] \ldots x'[n+1]$. EX1003, [0131] ("The

redundancy generator 468 … uses a 4 input-6 output Reed Solomon redundancy generation function with systematic encoding to generate six pre-shreds 470 with ***four of the pre-shreds containing direct pass-through data blocks of the data file***."). EX1002, ¶287.

b. Torre/Desai System-2

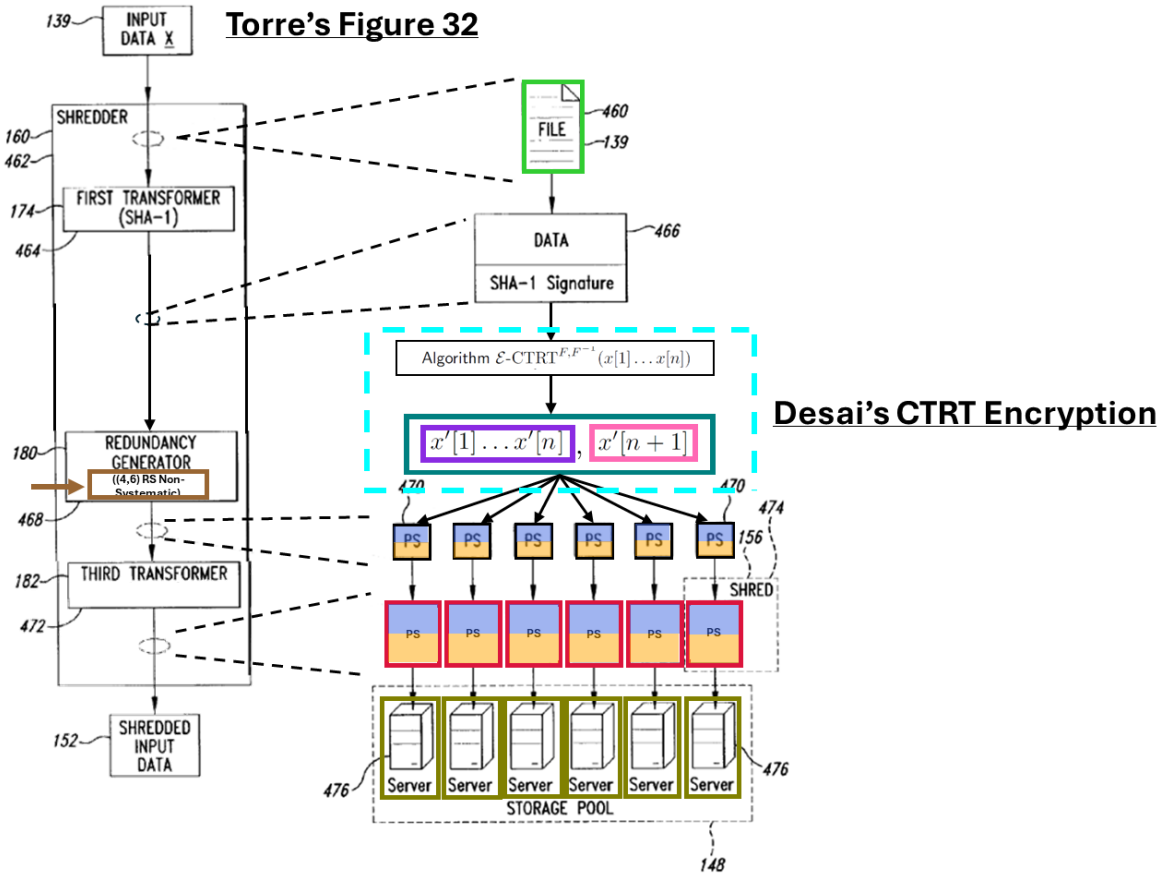
Torre/Desai System-2 satisfies [1D]. EX1002, ¶¶289, 292.

First, Torre/Desai System-2's shredder "*produc[es] a plurality of n shares from the second data set*." As explained in Section IX.A.3.ii, Torre/Desai System-2's shredder includes a redundancy generator 468 implementing non-systematic encoding that processes the second data set $x' = x'[1] \ldots x'[n+1]$ "to generate six pre-shreds 470" that ***each*** contain both pass-through data blocks of the received data and redundancy information. EX1003, [0131], [0117]. *See* EX1022; EX1019, 8 ("In the case of a non-systematic encoder, every one of the *n* columns of the generator matrix generates a 'parity' sequence."). EX1002, ¶289.

The "six pre-shreds" are further processed by the shredder's "third transformer(s) 472" to "form shreds 474, which includes the pre-shred data." EX1003, [0132]. As shown in Figure 32 of Torre (below), each shred includes pre-shred data from a corresponding one of the pre-shreds. Therefore, the third transformer produces ***six*** shreds 474 that each include both pass-through data blocks

of the received data (which is pre-processed to form the "*second data set*" $x' = x'[1] \ldots x'[n+1]$) and redundancy information. EX1003, [0132]. EX1002, ¶290.

Torre/Desai System-2 (Shredder)



EX1003, FIG. 32; EX1004, 16.

Accordingly, the Torre/Desai System-2's shredder "*produc[es] a plurality of n shares* [***six*** shreds] *from the second data set* [$x' = x'[1] \ldots x'[n+1]$]." EX1002, ¶¶289, 292.

Second, in Torre/Desai System-2, "*each of the n shares comprises at least some of the second data set*." As explained above in this Section, the six shreds include pre-shred data from a corresponding one of the pre-shreds, which, in turn, contain data from the "*second data set*" $x' = x'[1] \ldots x'[n+1]$. EX1003, [0131], [0117] ("An example of non-systematic encoding is shown in FIG. 22 where the redundancy generator 180 generates six outputs of redundancy-generated data from four inputs of received data."). EX1002, ¶291.

vi. [1E] storing the plurality of n shares and the redundancy information, wherein at least some of the n shares are stored in separate storage locations and the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares

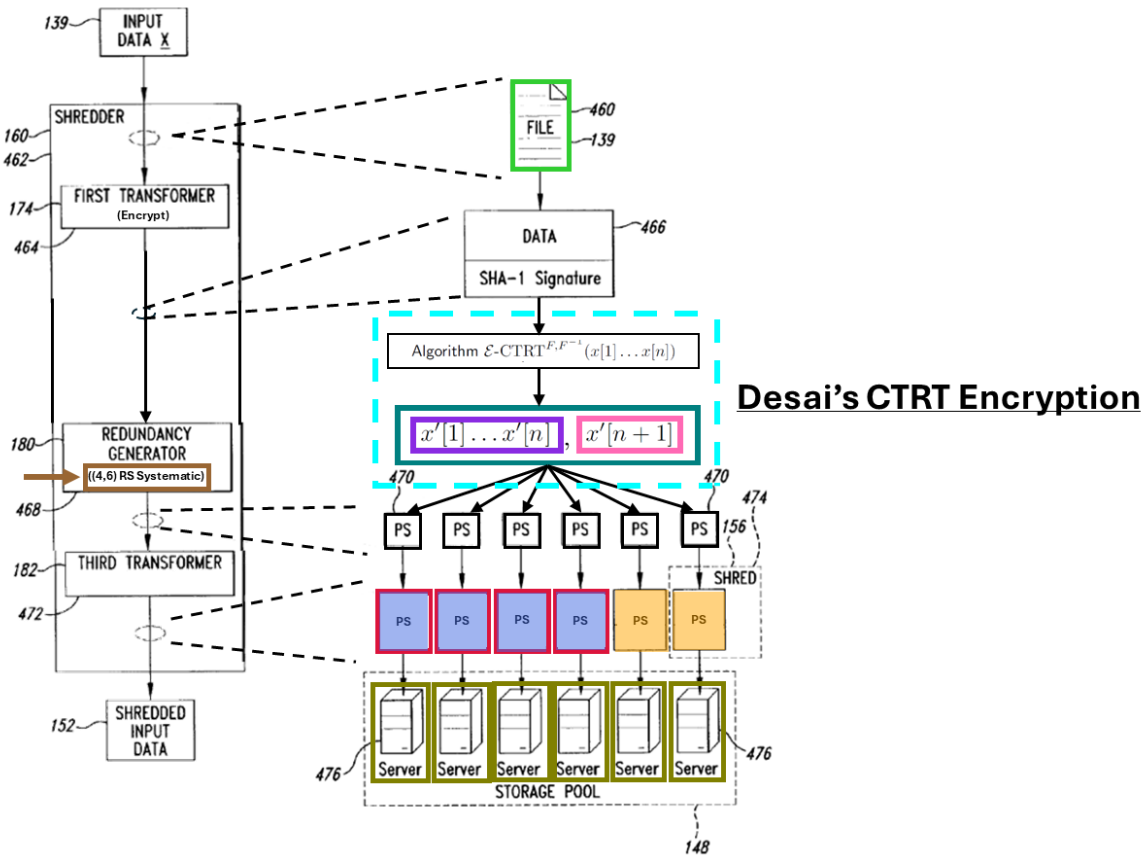
a. Torre/Desai System-1

Torre/Desai System-1 satisfies [1E]. EX1002, ¶¶293, 297.

First, Torre/Desai System-1 teaches "*storing the plurality of n shares and the redundancy information, wherein at least some of the n shares are stored in separate storage locations*." Torre/Desai System-1's shredder stores each of the N = 4 shreds containing pass-through data of the data set ("*plurality of n shares*") and the two shreds containing redundancy information ("*redundancy information*") in "the storage pool 146 where in this instance is made up of six servers 476." EX1003, [0132]. EX1002, ¶294.

Torre/Desai System-1
(Shredder)

Torre's Figure 32



Desai's CTRT Encryption

EX1003, FIG. 32; EX1004, 16.

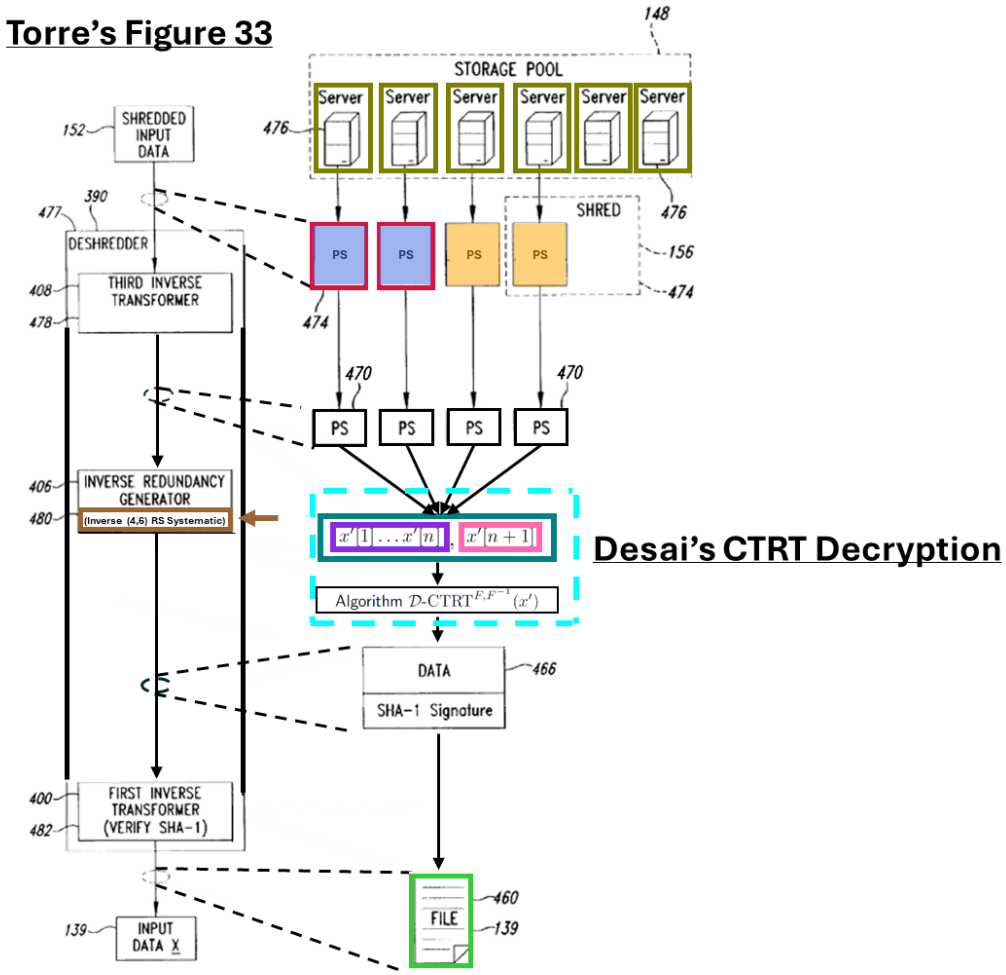
"For higher reliability, the storage units [such as the servers 476 in the storage pool 148] could be ***physically located separately from one another***, typically linked to the shredder by one of the interconnects 144 that is a communication network." EX1003, [0069]. *See also* EX1003, [0052] ("Storage of the shredded input data … may involve sending it … to ***multiple storage units*** such as … ***multiple servers***."). Accordingly, the Torre/Desai System-1 causes the plurality of shares to be stored in

respective servers that are "physically located separately from one another." EX1003, [0069]. EX1002, ¶295.

Second, Torre/Desai System-1 teaches that "*the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares*." Torre/Desai System-1's deshredder can recover the input data ("*first data set*") from any four of the six shreds stored in the servers. EX1003, [0133] ("Complementary to the shredder 462, an exemplary depiction of a deshredder 477 that deshreds without de-allocation, shown in FIG. 33, receives four of the shreds 474 from four of the servers 476."). Accordingly, Torre/Desai System-1's deshredder requires ***at least two*** of the four shreds ("*at least a threshold number* [=2]*, less than n* [=4]*, of the plurality of n shares*") containing pass-through data of the data set and the two additional shreds containing redundancy information to recover the pre-shred 466 containing the original input data, as shown in the below annotated figure. EX1003, [0133]. EX1002, ¶296.

Torre/Desai System-1 (Deshredder)

Torre's Figure 33



EX1003, FIG. 33; EX1004, 16.

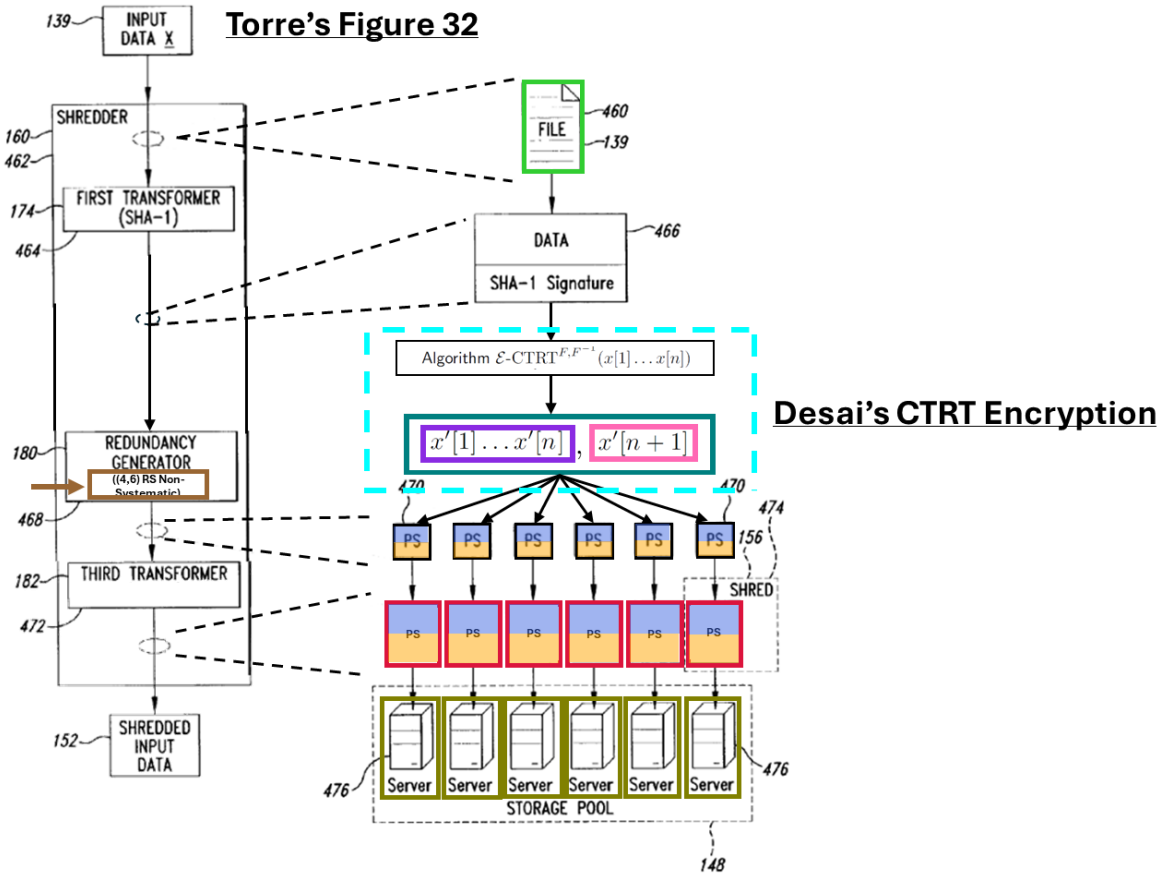
b. Torre/Desai System-2

Torre/Desai System-2 teaches [1E]. EX1002, ¶¶298, 302.

First, Torre/Desai System-2 teaches "*storing the plurality of n shares and the redundancy information, wherein at least some of the n shares are stored in separate storage locations*." Torre/Desai System-2's shredder stores each of the N = 6 shreds containing both pass-through data of the data set and redundancy information in six

different servers 476, as shown in the below annotated figure. EX1003, [0132]. EX1002, ¶299.

Torre/Desai System-2
(Shredder)



EX1003, FIG. 32; EX1004, 16.

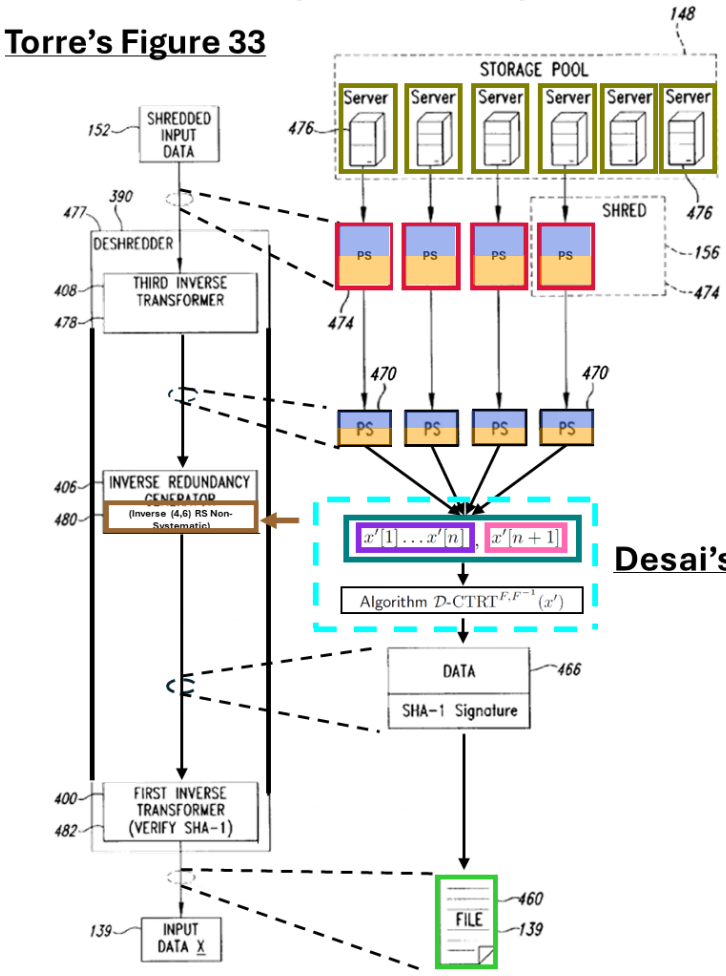
As explained in Section IX.A.4.vi.a [1E, Torre/Desai System-1], "[f]or higher reliability, the storage units [such as the servers 476 in the storage pool 148] could be ***physically located separately from one another***." EX1003, [0069]. *See also* EX1003, [0052]. Accordingly, for the same reasoning, the Torre/Desai System-2

causes the plurality of shares (the six shreds) to be stored in respective servers that are "physically located separately from one another." EX1003, [0069]. EX1002, ¶300.

Second, Torre/Desai System-2 teaches that "*the first data set is recoverable using at least a threshold number, less than n, of the plurality of n shares*." Torre/Desai System-2's deshredder can recover the input data ("*first data set*") from four of the six shreds stored in the servers. EX1003, [0133]. Accordingly, Torre/Desai System-2's deshredder requires ***at least four*** of the six shreds ("*at least a threshold number* [=4]*, less than n* [=6]*, of the plurality of n shares*") containing pass-through data of the data set and redundancy information to recover the pre-shred 466 containing the original input data, as shown in the below annotated figure. EX1003, [0133]. EX1002, ¶301.

Torre/Desai System-2 (Deshredder)

Torre's Figure 33



Desai's CTRT Decryption

EX1003, FIG. 33; EX1004, 16.

5. Claim 2: The method of claim 1, wherein each of the n shares further comprises at least some of the redundancy information and storing the plurality of n shares and the redundancy information comprises storing the redundancy information in the n shares.

Torre/Desai System-2 renders Claim 2 obvious because Torre/Desai System-2 teaches the additional limitation of Claim 2. EX1002, ¶¶303, 307.

First, Torre/Desai System-2 teaches "*wherein each of the n shares further comprises at least some of the redundancy information*." For example, Torre/Desai System-2 uses a redundancy generator implementing ***non-systematic encoding*** that "generates six outputs of redundancy-generated data from four inputs of received data." EX1003, [0131], [0116]–[0117]. A non-systematic encoding generates outputs that each include redundancy information. *See* EX1022; EX1019, 8. EX1002, ¶304.

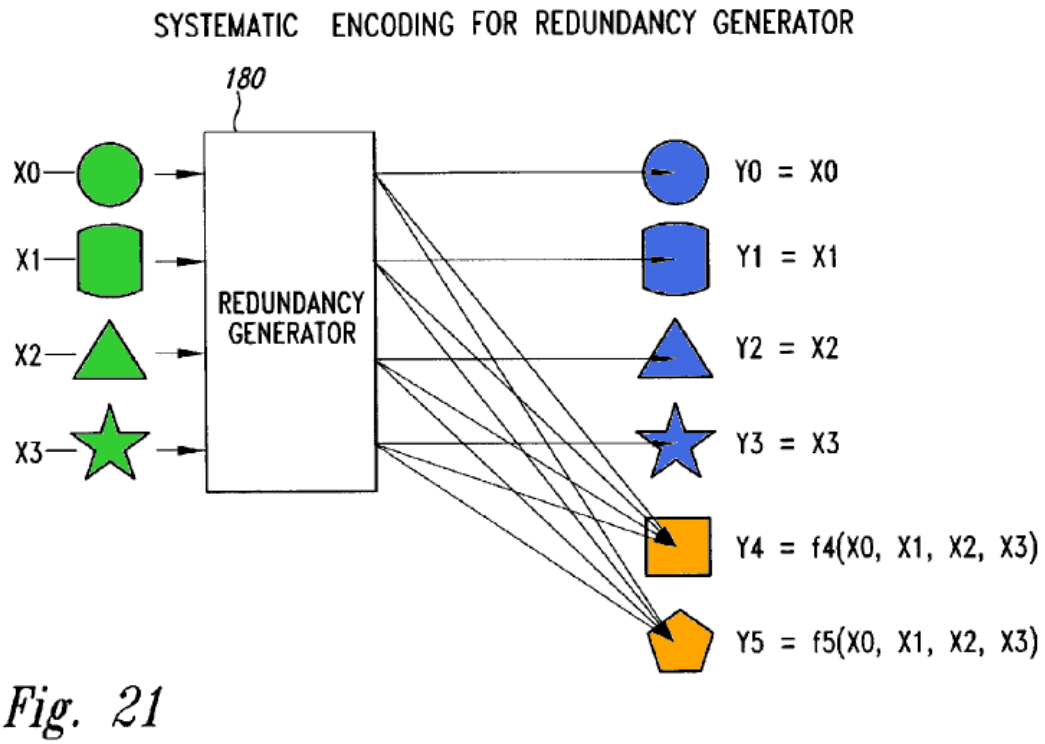


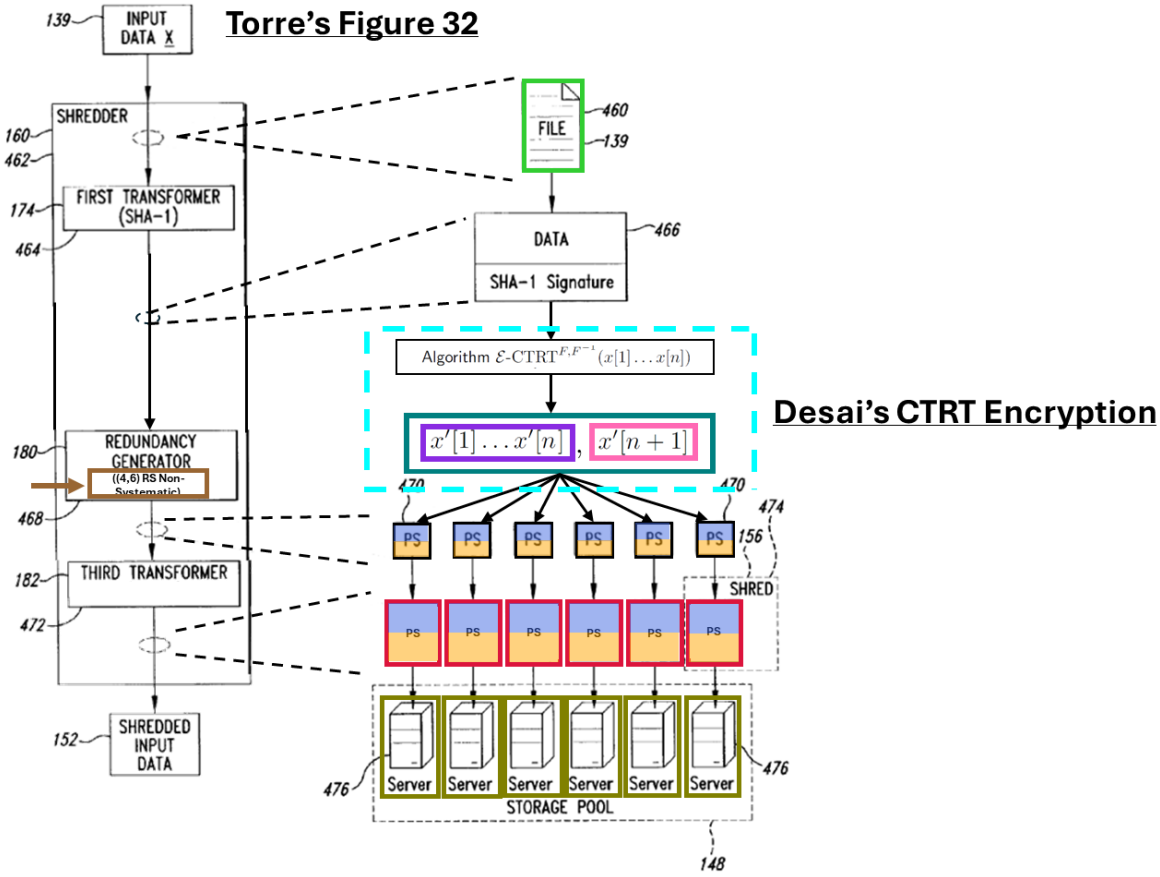
Fig. 21

EX1003, FIG. 22.

Second, Torre/Desai System-2 further teaches that "*storing the plurality of n shares and the redundancy information comprises storing the redundancy*

information in the n shares." For example, Torre/Desai System-2's shredder contains a redundancy generator that generates six pre-shreds 470 each containing redundancy information. EX1003, [0117], [0131]. *See* EX1022; EX1019, 8. The third transformer then uses the pre-shreds 470 (each including at least a portion of the redundancy information) to form the six shreds 474, thereby "*storing the redundancy information in the n shares* [six shreds]." EX1003, [0132] ("The third transformer(s) … form shreds 474, ***which includes the pre-shred data***."). EX1002, ¶¶305–306.

Torre/Desai System-2
(Shredder)



EX1003, FIG. 32; EX1004, 16.

6. Claim 3: The method of claim 2, wherein storing the redundancy information in the n shares comprises including in each of the n shares information redundant to information included in at least one other of the n shares.

Torre/Desai System-2 renders Claim 3 obvious because Torre/Desai System-2 teaches the additional limitation of Claim 3. EX1002, ¶¶308, 313.

For example, Torre/Desai System-2's shredder uses a redundancy generator implementing non-systematic encoding to "generate[] six outputs of redundancy-

generated data from four inputs of received data." EX1003, [0117], [0131]. The non-systematic encoding is illustrated in Figure 22, below. EX1002, ¶309.

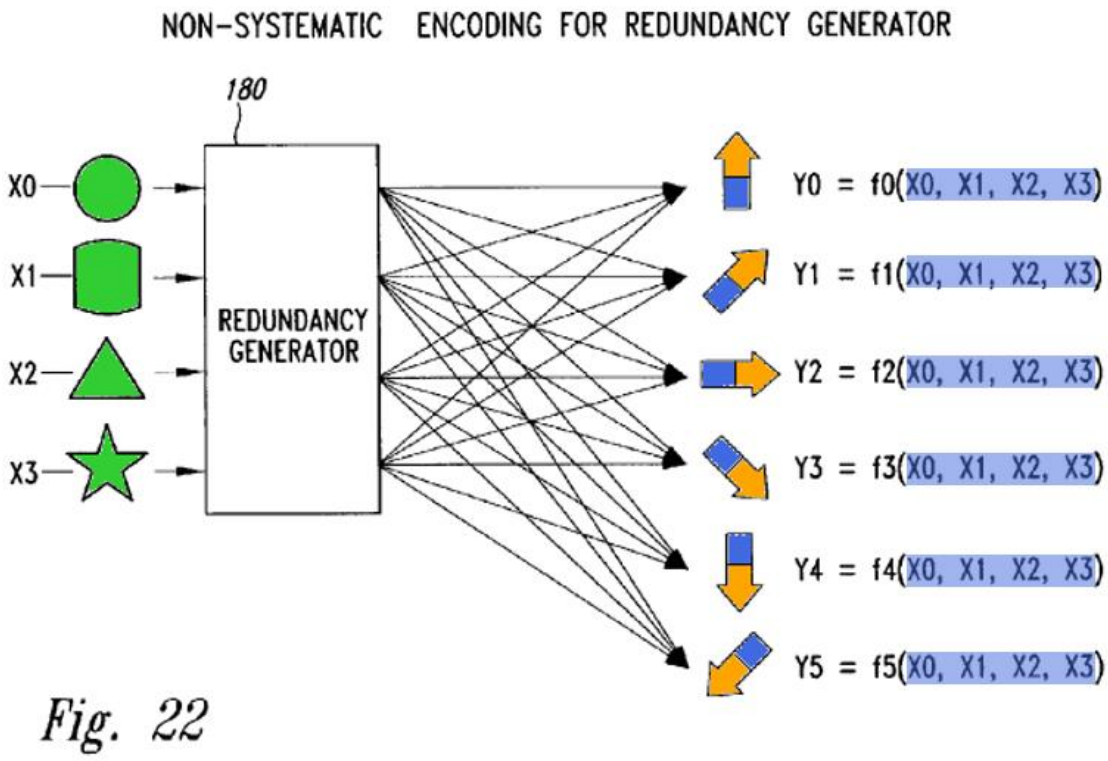


Fig. 22

EX1003, FIG. 22.

Figure 22 shows that each of the pre-shreds (Y0-Y5) includes "*information redundant to information included in at least one other of the n shares*." For example, one pre-shred Y0 is a function of inputs X0, X1, X2, and X3. Similarly, another pre-shred Y1 is also a function of inputs X0, X1, X2, and X3. Accordingly, both Y0 and Y1 include information on X0, X1, X2, and X3, and thus, Y0 and Y1 each include information that are redundant to each other. EX1022, 8 ("Nonsystematic codes add redundancy and transform the coded message such that no part of the original message is recognizable from the un-decoded message.").

That is further illustrated by Figure 22. In Figure 22, the directional vectors associated with pre-shreds Y0 and Y1 (and) each point upward, indicating that each pre-shred contains information redundant to the other pre-shred. EX1003, FIG. 22. EX1002, ¶310.

As another example, one pre-shred Y2 is also a function of inputs X0, X1, X2, and X3. Accordingly, both Y1 and Y2 include information on X0, X1, X2, and X3, and thus, Y1 and Y2 each include information that are redundant to each other. EX1022, 8. That is further illustrated by Figure 22 in which the directional vectors associated with pre-shreds Y1 and Y2 (and) each point sideways, indicating that each pre-shred contains information redundant to the other pre-shred. EX1003, FIG. 22. EX1002, ¶311.

The same analysis suggests that the following pairs of pre-shreds include information redundant to each other: Y2 and Y3 (and , both pointing sideways); Y3 and Y4 (and , both pointing downwards); and Y4 and Y5 (and , both pointing downwards). Accordingly, Torre's redundancy generator implementing non-systematic encoding includes, in pre-shreds Y0, … Y4, "*information redundant to information included in*" pre-shreds Y1, …, Y5, respectively, and conversely, includes, in pre-shreds Y1, … Y5, "*information*

redundant to information included in" pre-shreds Y0, …, Y4, respectively. EX1002, ¶312.

7. Claim 10: The method of claim 1, wherein at least some of the separate storage locations are geographically separate from one another.

Torre/Desai System renders obvious Claim 10 because Torre discloses the additional limitation of Claim 10. EX1002, ¶¶313, 317.

For example, Torre explains, in the context of Figure 32, that "[e]ach of the three data blocks 494 results in six shreds 156 being stored on the six servers 476" forming the "storage pool 148." EX1003, [0132], [0135]. Torre also teaches that the "storage pool 148 contain[s] multiple storage units 150," such as servers 476, and "[f]or higher reliability, the storage units 150 could be ***physically located separately from one another***." EX1003, [0068]–[0069]. For example, the storage devices may be connected by a "network," and hence "*geographically separate from one another*." EX1003, [0052]. *See also* EX1003, FIG. 5, [0132]. Furthermore, the '456 Patent does not disclose a minimum distance for a location to be considered "geographically separate." *See, e.g.*, EX1001, 3:28–32, 5:7–11, 5:51–53. EX1002, ¶¶315–316.

8. Claim 11: The method of claim 1, wherein logically combining comprises logically combining using an exclusive-OR operation.

Torre/Desai System renders obvious Claim 11 because Desai teaches the additional limitation of Claim 11. EX1002, ¶¶318, 320.

As explained in Section IX.A.4.iii.b [1B-2], Desai's *E*-CTRT$^{F,F^{-1}}$ transformation algorithm includes **step E4** in which the algorithm computes the exclusive-OR operation (denoted by the symbol "⊕") of the key K' and all of the encrypted data blocks $x'[1]$ … $x'[n]$ to produce the resultant $x'[n+1]$. This XOR operation of step E4 discloses the claimed "*logically combining*" operation, and thus teaches "*logically combining using an exclusive-OR operation*." EX1002, ¶319.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

9. Claim 12: The method of claim 1, wherein the at least a portion of the encrypted data set comprises all of the encrypted data set.

Torre/Desai System renders obvious Claim 12 because Desai discloses the additional limitation of Claim 12. EX1002, ¶¶321, 323.

As explained in Section IX.A.4.iii.b [1B-2], Desai discloses an *E*-CTRT$^{F,F^{-1}}$ transformation algorithm includes **step E4** in which the algorithm computes the XOR of ***all*** of the encrypted data blocks *x'*[1] ... *x'*[*n*] (the "*encrypted data set*") and *K'* (the "*encryption key*") to obtain *x'*[*n*+1] (the "*resultant*"). Thus, Desai teaches logically combining "*all of the encrypted data set*" *x'*[1] ... *x'*[*n*]. EX1002, ¶322.

	Algorithm $\mathcal{E}$-CTRT$^{F,F^{-1}}(x[1]\ldots x[n])$
Step E1	$K' \leftarrow \{0,1\}^l$
Step E2	$K = K' \bmod 2^k$ // $(\lvert K\rvert = k)$
Step E3	for $i = 1,\ldots,n$ do $x'[i] = x[i] \oplus F_K(i)$
Step E4	$x'[n+1] = K' \oplus x'[1] \oplus \ldots \oplus x'[n]$
Step E5	return $x'[1]\ldots x'[n+1]$

EX1004, 16.

10. Claim 13

Claim 13 varies from claim 1 for the preamble [13Pre], but otherwise has substantially the same limitations as claim 1. Torre/Desai System renders obvious

[13A]–[13D] obvious for the same reasons provided above for [1B]–[1E]. *See* Sections IX.A.4.iii–IX.A.4.vi. EX1002, ¶¶324–325.

i. [13Pre] A computer system for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the system comprising one or more processors configured to:

Torre discloses [13Pre]. EX1002, ¶¶326, 329.

For the reasons provided in Section IX.A.4.i [1Pre], Torre discloses "*securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations*." EX1002, ¶327.

Torre also discloses that "*[a] computer system … comprising one or more processors*" is configured to perform the method. For example, Torre explains implementing its teachings "as one or more computer programs running on one or more ***computers*** (e.g., as one or more programs running on one or more data processing ***systems***)" ("*[a] computer system*"). EX1003, [0141]. Torre further explains that the programs can "run[] on one or more ***processors*** e.g., ***microprocessors***" ("*comprising one or more processors*"). EX1003, [0141]–[0142], [0064]. EX1002, ¶328.

11. Claim 25

Claim 25 varies from claim 1 for the preamble [25Pre], but otherwise has the same limitations as claim 1. Torre/Desai System renders obvious [25A]–[25E] for

the reasons provides above for [1A]–[1E]. *See* Sections IX.A.4.ii–IX.A.4.vi. EX1002, ¶¶330–332.

i. [25Pre] A non-transitory computer-readable medium comprising instructions that, when executed by one or more processors, cause a computer system to carry out a method for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations, the method comprising:

Torre discloses [25Pre]. EX1002, ¶¶333, 337.

For the reasons provided in Section IX.A.4.i [1Pre], Torre discloses "*a method for securing a first data set while providing recoverability of the first data set in the event of unavailability of data at one or more storage locations*." EX1002, ¶334.

Torre also discloses that the method is implemented as "*[a] non-transitory computer-readable medium comprising instructions that, when executed by one or more processors, cause a computer system to carry out*" the method. For example, Torre states that its teachings "can be implemented … by … ***software***." EX1003, [0141]. Torre explains that its software is implemented as "code" (i.e., "*instructions*") (EX1003, [0141]), and that the software "*when executed by one or more processors, cause[s] a computer system*" to perform the method. EX1003, [0141] (explaining that its teachings are implemented "as one or more ***computer programs*** running on ***one or more computers*** …, as one or more ***programs*** running on ***one or more processors e.g., microprocessors***"), [0142], [0064]. Torre also

teaches that the software code is stored in "signal bearing media" such as "floppy disks, hard disk drives, CD ROMs, digital tape, and computer memory" (*i.e.,* "[a] *non-transitory computer-readable medium*"). EX1003, [0141], [0064]. EX1002, ¶¶335–336.

12. Claims 14, 15, 22–24, 26, 27, 30

- Torre/Desai System-2 teaches the additional limitations of claims 14 and 26 as provided in Section IX.A.5 [Claim 2].
- Torre/Desai System-2 teaches the additional limitations of claims 15 and 27 as provided in Section IX.A.6 [Claim 3];
- Torre/Desai System teaches the additional limitations of claim 22 as provided in Section IX.A.7 [Claim 10];
- Torre/Desai System teaches the additional limitations of claim 23 as provided in Section IX.A.8 [Claim 11]; and
- Torre/Desai System teaches the additional limitations of claims 24 and 30 as provided in Section IX.A.9 [Claim 12].

EX1002, ¶338.

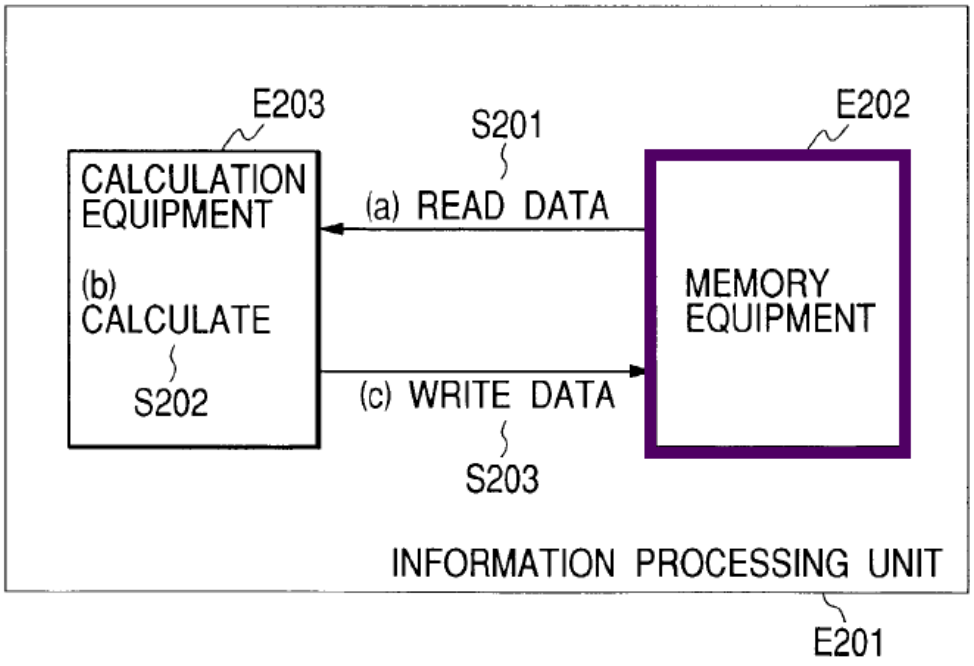
B. Ground II : Torre, Desai And Watanabe Render Obvious Claims 1–3, 10–12, 25–27, And 30.

1. Watanabe (EX1005)

Watanabe relates to "a method of encrypting and decrypting data used within an information processing unit." EX1005, [0003]. Watanabe discloses an "information processing unit" that "encrypts the contents of data exchanged between a memory device and calculation equipment." EX1005, [0025]. EX1002, ¶340.

Watanabe's "First Exemplary Embodiment" "relates to the encryption of data to be processed." EX1005, [0049]. "As shown in FIG. 2, data processing by an information processing unit E201 basically comprises: (a) ***the reading of data from a memory device E202*** (step S201); (b) calculation by calculation equipment E203 (step S202); and (c) the writing of data into the memory device E202 (step S203)." EX1005, [0049]. EX1002, ¶341.

FIG. 2

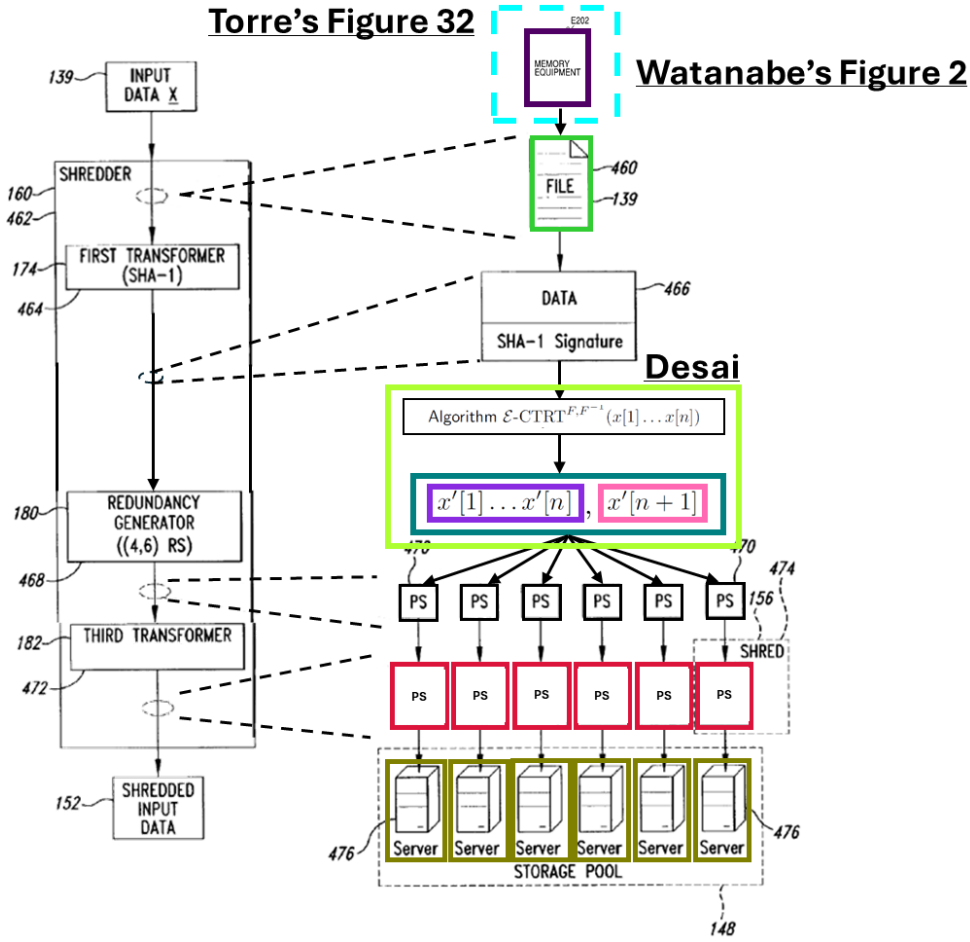


EX1005, FIG. 2.

2. Torre/Desai In Combination With Watanabe ("Torre/Desai/Watanabe System")

Torre/Desai System is directed to encrypting (and performing a variation of AONT on) the input data and subsequently splitting the input data into shreds and storing the shreds. It would have been obvious to augment the Torre/Desai System with Watanabe so that the Torre/Desai System retrieves the data to be processed from memory, as taught by Watanabe. EX1002, ¶342.

Torre/Desai/Watanabe System (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16.

i. Reasons To Combine Torre/Desai System With Watanabe

A POSITA would have been motivated to augment the Torre/Desai System with Watanabe's memory device so that the Torre/Desai System receives the input data from a memory device, as taught by Watanabe for two reasons. EX1002, ¶343.

First, Watanabe explains that an encryption system—like the calculation equipment E203—can read the data for processing from a memory device E202. EX1005, [0049]. And Watanabe specifically teaches that "various types of data items are read from the memory device E202 depending on the purpose or status of the unit." EX1005, [0049]. Thus, a POSITA would have understood that a memory device—like Watanabe's—can maintain and provide "input data" in Torre. EX1003, [0050] ("The input data can be a file or another form that has been generated directly or by other means. Generation of data includes, but is not limited to, generation by operating software programs and by receiving or recording data such as live audio or video, or audio or video broadcast streams."). Accordingly, a POSITA would have been motivated to use Watanabe's memory device with Torre/Desai System so that the Torre/Desai System can receive information from a variety of sources, such as a memory device. To the extent the Torre/Desai System does not already contemplate retrieving input data from a memory device (as explained in Section IX.A.4.ii [Ground I, [1A]]), that would have improved the versatility of the Torre/Desai System. EX1002, ¶¶344–345.

Second, a POSITA would have been particularly motivated to combine the Torre/Desai System with Watanabe because the combination would have entailed a simple addition of a memory device and using the memory device in a conventional way—to store and provide data to a data processing system, like the Torre/Desai

System. Accordingly, the combination would have merely amounted to combining prior art elements (Torre/Desai System and Watanabe's memory device) according to known methods (using the Watanabe's memory device to provide input data to Torre/Desai System, as taught in Watanabe) to yield predictable results (providing input data to Torre/Desai System for further data processing). EX1002, ¶346.

A POSITA would have had a reasonable expectation of success in using Watanabe's memory device to provide input data to the Torre/Desai System. EX1002, ¶347.

First, Torre discloses that its shredder "receiv[es] the input data (step 122)" and then "performs first transformations (step 132)," which "could include one or more of compression, encryption, or signature generation functions." EX1003, [0061]. Similarly, Watanabe specifically teaches that in the "embodiment relate[d] to the encryption of data to be processed" includes "the reading of data from a memory device" and "calculation by calculation equipment." EX1005, [0049]. Furthermore, a POSITA would have understood that the data files—such as the input data file disclosed in Torre (Section IX.A.1 [Torre]) and the message/data blocks disclosed in Desai (Section IX.A.2 [Desai])—can be stored on and read from well-known data storage memory structures, such as Watanabe's memory device. Accordingly, the combination—which simply uses Watanabe's memory device as the memory source of the input data without any substantive change to the

shredding/deshredding function of Torre/Desai System—would have yielded predictable results of storing the input data and key information in a memory device to improve overall data storage security. EX1002, ¶¶348–349.

Second, the combination would have merely amounted to using a known technique (using a memory device to store and provide data for a data processing system, as disclosed by Watanabe) to improve similar devices (to improve the versatility of a similar device like the Torre/Desai System to process a variety of information from various sources) in the same way (to provide the data as input to the Torre/Desai System for further processing). EX1002, ¶350.

3. Claim 1

The Torre/Desai/Watanabe System renders obvious [1Pre] and [1B]–[1E] for the same reasons discussed above in Ground I. *See* Section IX.A.4 [Ground I, Claim 1]. EX1002, ¶351.

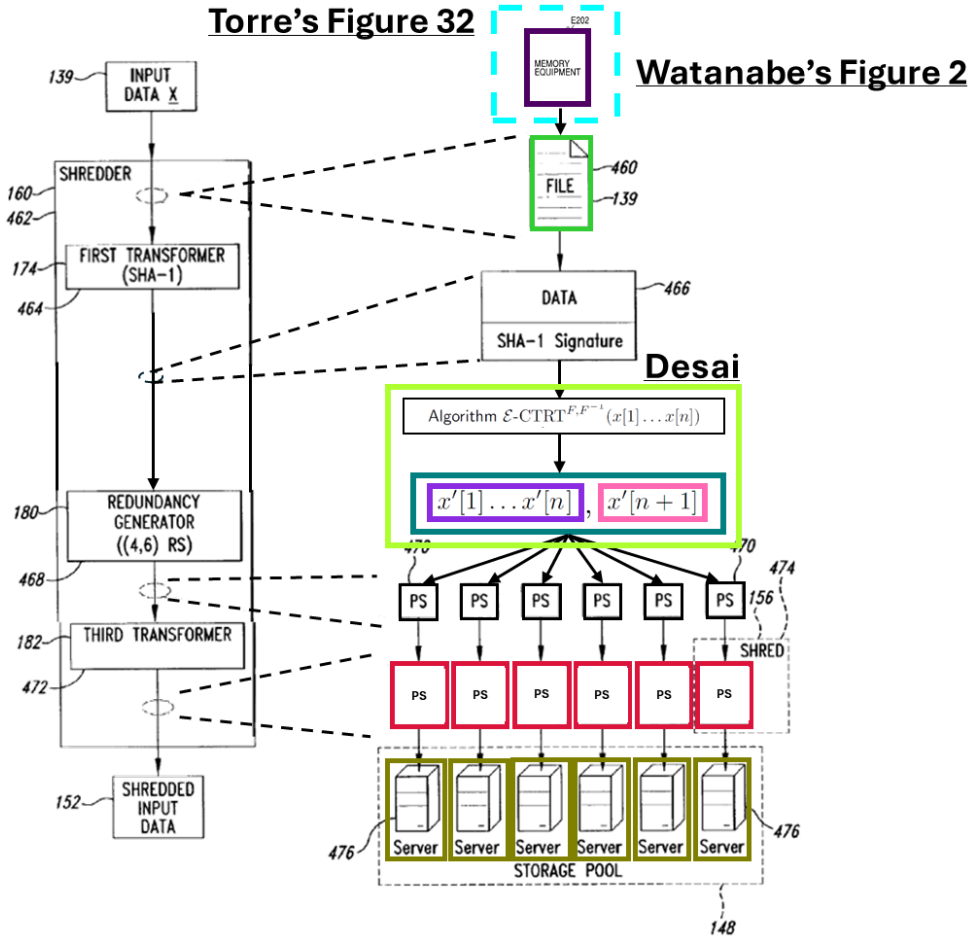
i. [1A] reading the first data set from a memory that stores the first data set;

To the extent Patent Owner argues that the combination of Torre and Desai does not disclose [1A], that limitation would have been obvious in view of Watanabe. EX1002, ¶¶352, 354.

Specifically, Watanabe teaches that "data processing by an information processing unit E201 basically comprises: (a) ***the reading of data from a memory***

device E202 (step S201) … ." EX1005, [0049]. As explained in Section IX.B.2 [Torre/Desai/Watanabe], it would have been obvious to combine the Torre/Desai System with Watanabe so that the Torre/Desai system "read[s] … [input data] from a memory device" (EX1005, [0049]), as shown below. EX1002, ¶353.

Torre/Desai/Watanabe System (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16.

4. Claim 25

The Torre/Desai/Watanabe System renders obvious [25Pre] and [25B]–[25E] for the same reasons discussed above in Ground I. *See* Section IX.A.11. The Torre/Desai/Watanabe System renders obvious [25A] for the same reasons discussed in Ground II for [1A]. *See* Section IX.B.3.i [1A]. EX1002, ¶355.

5. Claims 2, 3, 10-12, 26, 27, 30

- Torre/Desai/Watanabe System teaches the additional limitations of claims 2 and 26 as provided in Sections IX.A.5 [Ground I, Claim 2] and IX.A.12 [Ground I, Claim 26];
- Torre/Desai/Watanabe System teaches the additional limitations of claims 3 and 27 as provided in Sections IX.A.6 [Ground I, Claim 3] and IX.A.12 [Ground I, Claim 27];
- Torre/Desai/Watanabe System teaches the additional limitations of claim 10 as provided in Section IX.A.7 [Ground I, Claim 10];
- Torre/Desai/Watanabe System teaches the additional limitations of claim 11 as provided in Sections IX.A.8 [Ground I, Claim 11]; and
- Torre/Desai/Watanabe System teaches the additional limitations of claims 12 and 30 as provided in Sections IX.A.9 [Ground I, Claim 12] and IX.A.12 [Ground I, Claim 30].

EX1002, ¶356.

C. Ground III: Ground I Or Ground II In Combination With Orsini Renders Obvious Claims 4–9, 16–21, And 28–29.

1. Orsini (EX1006)

Orsini includes disclosures that are substantially identical to those in the '456 Patent. In fact, Orsini's Figures 1–25 and the associated descriptions are substantially identical to the '456 Patent's Figures 1–25 and the associated descriptions. *See* EX1021 (showing redlines between Orsini and the '456 Patent's specification). EX1002, ¶358.

Orsini states that "[i]n one embodiment of the present invention, [a] data splitting module comprises a parser program or software suite which comprises data splitting." EX1006, [0278]. Orsini explains that the data splitting module performs a "cryptographic split." EX1006, [0281]. "A cryptographic split (cryptosplit)" refers to a process that "partitions [] data into N number of shares." EX1006, [0281]. As part of the partitioning process, "[t]he units are [] distributed (either randomly or by a predetermined set of values) into the N shares. This distribution could also involve a shuffling of the order of the units in the shares." EX1006, [0281]. EX1002, ¶359.

Orsini explains that the parsing process may involve "splitting the data according to the contents of a generated random number, or key." EX1006, [0280]. In one example, Orsini describes a cryptosplit/data shuffling process where the data

is "23 bytes in size" and each "data unit" measures "one byte." EX1006, [0282]. In this example, "a key would be obtained to create a sequence of 23 random numbers (r1, r2, r3 through r23)"; "each of the units of data (in this example 23 individual bytes of data) is associated with one of the 23 random numbers corresponding to one of the four shares"; and each unit of data is distributed into the shares based on the associated random number. EX1006, [0284]. EX1002, ¶360.

Orsini further explains that "the data units in each share could be shuffled utilizing a different algorithm. This data unit shuffling may be performed as the original data is split into the data units, or after the data units are placed into the shares, or after the share is full, for example." EX1006, [0286]. EX1002, ¶361.

2. Grounds I or II In Combination With Orsini

Torre/Desai System in Ground I and Torre/Desai/Watanabe System in Ground II are both directed to securely storing input data by parsing and splitting the input data into shreds and storing the shreds in different storage locations. *See* Sections IX.A.3, IX.B.2. It would have been obvious to augment the Torre/Desai System in Ground I or Torre/Desai/Watanabe System in Ground II with Orsini's cryptosplit or data shuffling process to increase the security of the stored data in the Torre/Desai

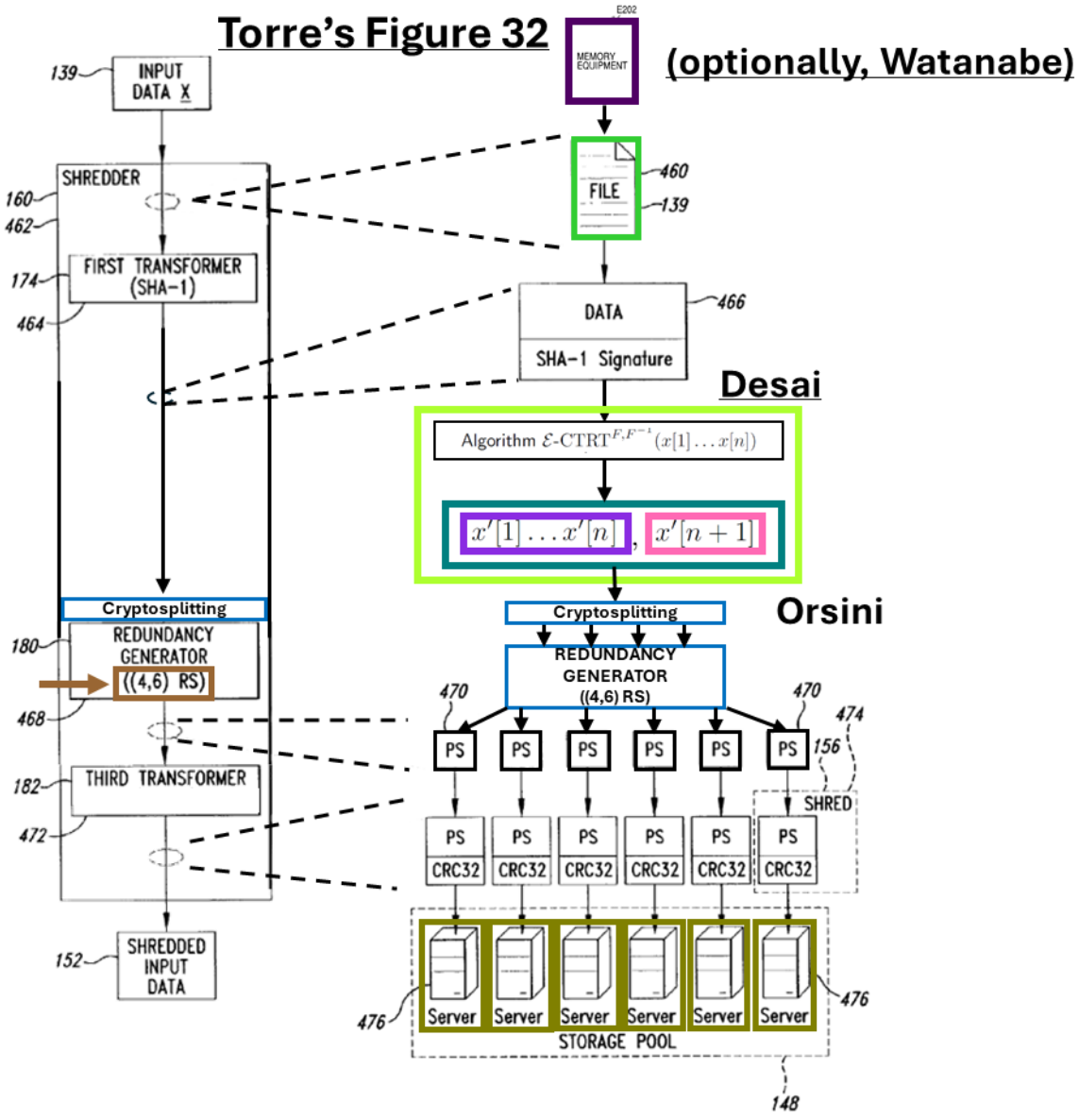
System in Ground I or Torre/Desai/Watanabe System in Ground II—collectively Torre/Desai(/Watanabe)[3] System, as explained below. EX1002, ¶¶362–363.

i. Torre/Desai(/Watanabe) System In Combination With Orsini's Cryptosplit Process ("Torre/Desai (/Watanabe)/Orsini System-1")

It would have been obvious to augment the Torre/Desai(/Watanabe) System with Orsini so that the output of Desai's CTRT process is subject to Orsini's cryptosplit process. Specifically, the output of Desai's *E*-CTRT algorithm—$x' = x'[1] \ldots x'[n+1]$—is subject to Orsini's cryptosplitting process to generate four pre-shreds that are then fed into the "4 input-6 output Reed Solomon redundancy generation function," as shown below. EX1006, [0281], [0286]; EX1003, [0131]. EX1002, ¶364.

[3] The parenthetical around Watanabe denotes that Watanabe is relevant only to Ground II.

Torre/Desai(/Watanabe)/Orsini System-1 (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16; EX1006, [0281], [0286].

a. Reasons for Combining Torre, Desai, Orsini (and, optionally, Watanabe) As Provided In Torre/Desai (/Watanabe)/Orsini System-1

A POSITA would have been motivated to combine Torre/Desai(/Watanabe) with Orsini as provided in Torre/Desai (/Watanabe)/Orsini System-1 for two reasons. EX1002, ¶365.

First, Torre's "4 input-6 output Reed Solomon redundancy generation function" receives input data as four segments and generates six pre-shreds. *See* EX1003, FIGS. 21–22; EX1003, [0131]. Accordingly, a POSITA would have looked to prior art to find ways to provide the output of Desai's *E*-CTRT algorithm as four segments to the "4 input-6 output Reed Solomon" redundancy generator 180. And a POSITA would have understood that Orsini's cryptosplit process provides a process for splitting the input data—such as the output of Desai's *E*-CTRT algorithm—into four segments. EX1002, ¶¶366–367.

Indeed, Orsini states that its data splitting module implementing the cryptosplit process can be used "as an add-on to ***any*** application which requires ***the ultimate security*** of its data elements" and "[t]his parsing process operates on ***any type of data set***, and on any and all file types, or in a database on any row, column or cell of data in that database." EX1006, [0278]. *See also* EX1006, [0115]. Accordingly, a POSITA would have been motivated to use Orsini's cryptosplit process to divide the output of Desai's CTRT algorithm into four segments to obtain

the predictable results of improving the security of the shredder process. EX1002, ¶¶366–367.

Second, a POSITA would have been motivated to implement a known technique (Orsini's cryptosplit process) to improve security of a known device (Torre/Desai(/Watanabe) System) in the same way (randomly shuffling and splitting the output of Desai's CTRT algorithm to provide an input to the "4 input-6 output Reed Solomon" redundancy generator 180). Orsini's cryptosplit process would have provided an additional layer of security to the Torre/Desai(/Watanabe) System—as Orsini itself explains (EX1006, [0278])—because for a third-party attacker to recover the input data from shreds generated by the Torre/Desai(/Watanabe)/Orsini System-1, the third-party attacker would have had to undo the additional splitting/data shuffling performed by Orsini's cryptosplit process. EX1002, ¶¶368–369.

A POSITA would have had a reasonable expectation of success in combining Torre/Desai(/Watanabe) System with Orsini as provided in Torre/Desai (/Watanabe)/Orsini System-1 for four reasons. EX1002, ¶370.

First, Orsini teaches that the cryptosplit process can be used "as an add-on to ***any*** application" and "operates on ***any type of data set***, and on any and all file types, or in a database on any row, column or cell of data in that database." EX1006, [0278]. Accordingly, a POSITA would have understood that Orsini's cryptosplit

process is readily applicable to the Torre/Desai(/Watanabe) System, particularly to the output of Desai's *E*-CTRT algorithm. EX1002, ¶371.

Second, in the Torre/Desai(/Watanabe)/Orsini System, the output of the cryptosplit process—the four data segments—is provided to "the "4 input-6 output Reed Solomon" redundancy generator 180 that receives four segments of data as input. EX1006, [0131]. Thus, a POSITA would have understood that each data segment generated by Orsini's cryptosplit process would be directly provided to the redundancy generator without further data manipulation so that the redundancy generator can directly process them as shown in Torre's FIG. 21 (systematic encoding) or FIG. 22 (non-systematic encoding). EX1002, ¶372.

Third, the combination would have merely amounted to using a known technique (e.g., Orsini's cryptosplit process) to improve the security of a similar device (e.g., Torre/Desai(/Watanabe) System) in the same way (by increasing the randomness of data units in the input of the redundancy generator, and hence, the shreds generated by the redundancy generator and the third transformer of the Torre/Desai(/Watanabe) System). EX1002, ¶373.

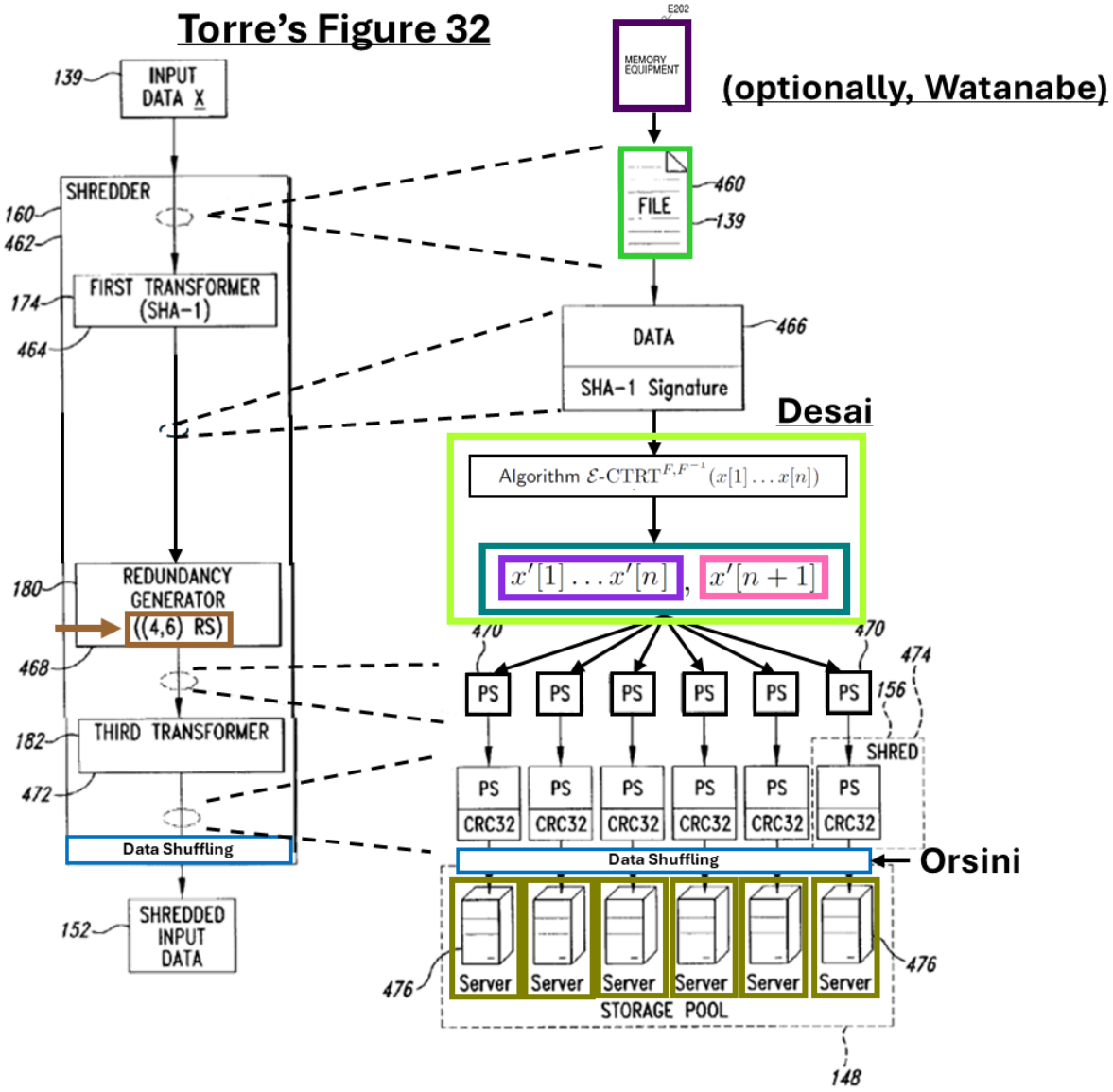
Fourth, Torre explains that its teachings can be implemented "as one or more ***computer programs***." EX1003, [0141]. Accordingly, implementing Orsini's teachings in the Torre/Desai(/Watanabe) System would have merely involved writing a program code. That was well within the skill of a POSITA. EX1003,

[0141]. *Keynetik*, 2023 WL 2003932, at *2 (Fed. Cir. Feb. 15, 2023). EX1002, ¶374.

ii. Torre/Desai(/Watanabe) System In Combination With Orsini's Data Shuffling Process ("Torre/Desai (/Watanabe)/Orsini System-2")

It would also have been obvious to combine the Torre/Desai(/Watanabe) System with Orsini to add Orsini's data shuffling process after the third transformer of the Torre/Desai(/Watanabe) System generates the shreds. Orsini explains that "the data units in each share could be shuffled utilizing a different algorithm. This data unit shuffling may be performed … ***after*** the data units are placed into the shares." EX1006, [0286]. It would also have been obvious to combine the Torre/Desai(/Watanabe) System with Orsini to add Orsini's data shuffling process ***after*** the third transformer of the Torre/Desai(/Watanabe) System generates the shreds, as shown below. This combination is referred to as "Torre/Desai (/Watanabe)/Orsini System-2." EX1002, ¶¶374–375.

Torre/Desai(/Watanabe)/Orsini System-2 (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16; EX1006, [0281], [0286].

a. Reasons for Combining Torre, Desai, Orsini (and, optionally, Watanabe) As Provided In Torre/Desai (/Watanabe)/Orsini System-2

A POSITA would have been motivated to combine Torre/Desai(/Watanabe) with Orsini as provided in Torre/Desai (/Watanabe)/Orsini System-2. EX1002, ¶377.

First, Orsini teaches that its data splitting module implementing the cryptosplit process and the data shuffling process can be used "as an add-on to ***any*** application which requires ***the ultimate security*** of its data elements." EX1006, [0278]. *See also* EX1006, [0115]. Accordingly, a POSITA would have been motivated to use Orsini's shuffling process in the Torre/Desai(/Watanabe) System to further increase the security of the shreds generated by the Torre/Desai(/Watanabe) System. EX1002, ¶378.

Second, a POSITA would have been motivated to implement a known technique (Orsini's data shuffling process) to improve security of a known device (Torre/Desai(/Watanabe) System) in the same way (randomly shuffling the data units in the shreds generated by the third transformer). Orsini's data shuffling process would have provided an additional layer of security to the Torre/Desai(/Watanabe) System—as Orsini itself explains (EX1006, [0278])—because for a third-party attacker to recover the input data from shreds generated by the Torre/Desai(/Watanabe)/Orsini System-2, the third-party attacker would have

had to undo the additional data shuffling performed by Orsini's process. EX1002, ¶379.

A POSITA would have had a reasonable expectation of success in combining Torre/Desai(/Watanabe) System with Orsini as provided in Torre/Desai (/Watanabe)/Orsini System-2 for three reasons. EX1002, ¶380.

First, Orsini teaches that the data splitting module that implements the data shuffling process can be used "as an add-on to ***any*** application" and "operates on ***any type of data set***, and on any and all file types, or in a database on any row, column or cell of data in that database." EX1006, [0278]. And Orsini specifically teaches that "[the] data unit shuffling may be performed … ***after*** the data units are placed into the shares," i.e., the shreds in the Torre/Desai(/Watanabe) System. EX1006, [0286]. Accordingly, a POSITA would have understood that Orsini's data shuffling process is readily applicable to the Torre/Desai(/Watanabe) System. EX1002, ¶¶381–382.

Second, the combination would have merely amounted to using a known technique (e.g., Orsini's data shuffling process) to improve the security of a similar device (e.g., Torre/Desai(/Watanabe) System) in the same way (by increasing the randomness in data units inside the shreds generated by the third transformer of the Torre/Desai(/Watanabe) System). EX1002, ¶383.

Third, Torre explains that "the embodiments disclosed herein … can be equivalently implemented … as one or more ***computer programs*** running on one or more computers." EX1003, [0141]. Accordingly, implementing Orsini's teachings in the Torre/Desai(/Watanabe) System would have merely involved writing a program code. That was well within the skill of a POSITA. EX1003, [0141]. *Keynetik*, 2023 WL 2003932, at *2 (Fed. Cir. Feb. 15, 2023). EX1002, ¶384.

3. **Claim 4: The method of claim 1, wherein producing the plurality of n shares comprises causing the n shares to comprise a substantially random distribution of units of data of the second data set.**

Claim 5: The method of claim 4, wherein causing the n shares to comprise a substantially random distribution comprises using a key.

Claim 6: The method of claim 5, wherein substantially randomly distributing units of data comprises using a key. [4]

Claim 7: The method of claim 1, wherein producing the plurality of n shares comprises substantially randomly distributing units of data of the second data set into the plurality of n shares.

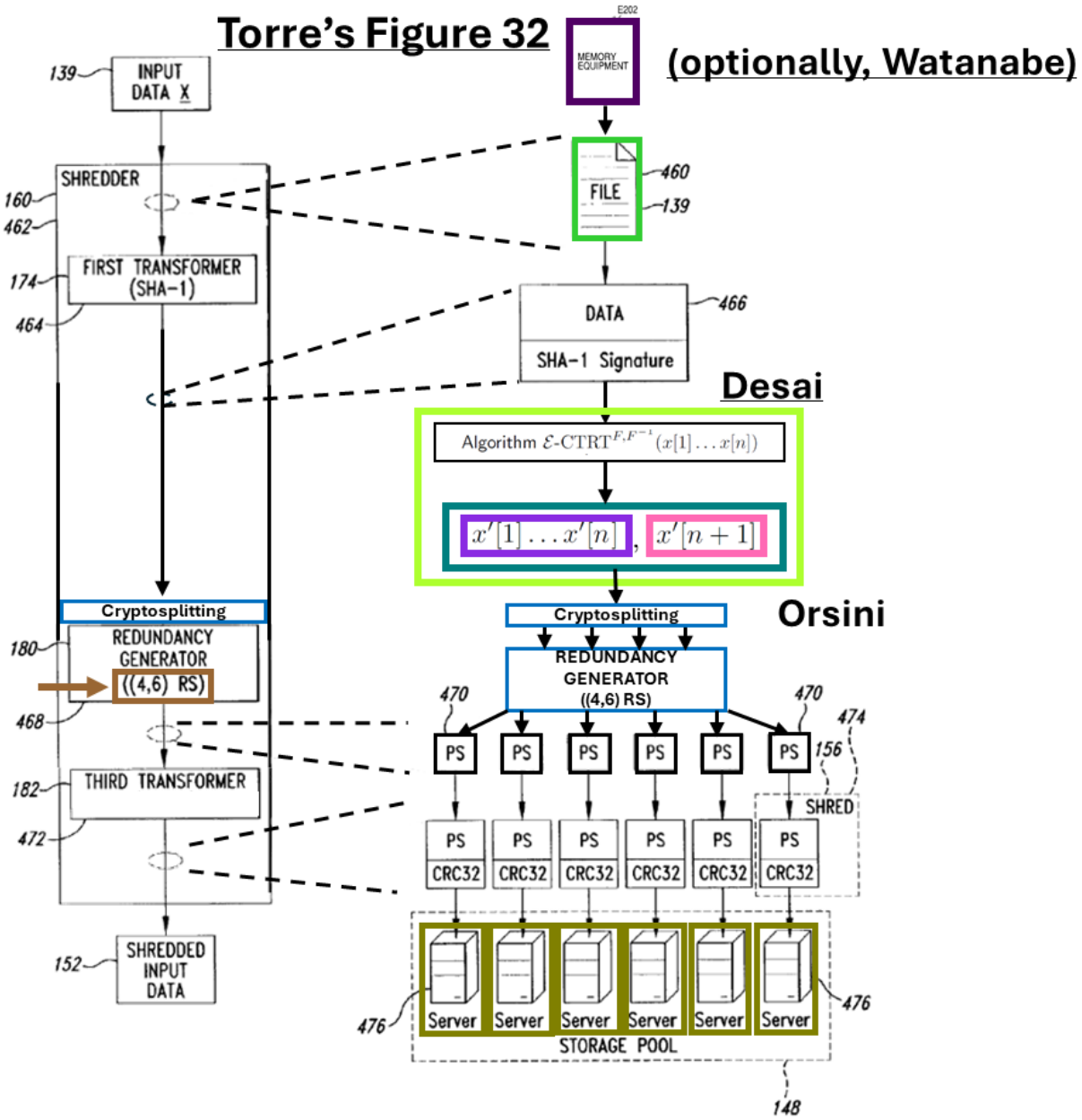
Ground I or Ground II in view of Orsini renders obvious Claims 4–7 because Torre/Desai(/Watanabe)/Orsini System-1 teaches the additional limitations of Claims 4–7. EX1002, ¶¶385, 389–392.

As explained in Section IX.C.2.i [Torre/Desai(/Watanabe)/Orsini System-1], the Torre/Desai(/Watanabe)/Orsini System-1 uses Orsini's cryptosplit process on the output of Desai's E-CTRT algorithm (i.e., $x' = x'[1] \ldots x'[n+1]$) to generate four data segments provided to the redundancy generator. Those data segments are provided to the redundancy generator and the third transformer that then generate

[4] Petitioner reserves the right to challenge claim 6 as being indefinite because the phrase "substantially randomly distributing units of data" lacks antecedent basis.

the shreds ("*plurality of n shares*"), as shown in the illustration below. EX1002, ¶386.

Torre/Desai(/Watanabe)/Orsini System-1 (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16; EX1006, [0281], [0286].

Orsini teaches "splitting the data according to the contents of a generated ***random*** number, or ***key***" (EX1006, [0280]) so that "[t]he units are [] distributed … ***randomly***" into output data segments (EX1006, [0281]). For example, Orsini discloses an example of the "cryptographic split process" in which 23 byte-sized units of data are split among 4 shares. EX1006, [0282]. "Assuming a random distribution, a key would be obtained to create a sequence of 23 ***random numbers*** (r1, r2, r3 through r23), each with a value between 1 and 4 corresponding to the four shares," "[e]ach of the units of data … is associated with one of the 23 random numbers corresponding to one of the four shares," and the units of data are distributed into the shares based on their associated random numbers. EX1006, [0282]. *See also* EX1006, [0284] (splitting data using a ***key*** with random numbers). EX1002, ¶387.

Because Orsini's cryptosplit process generates the four data segments provided to the redundancy generator by randomly distributing the data units of the output of Desai's *E*-CTRT algorithm (i.e., $x' = x'[1] \ldots x'[n+1]$), the shreds subsequently generated by the redundancy generator and the third transformer include randomly distributed data units of the output of Desai's *E*-CTRT algorithm. EX1002, ¶388.

Accordingly, with respect to Claim 4, Orsini's data splitting process in Torre/Desai(/Watanabe)/Orsini System-1 "*caus[es] the n shares* [the shreds] *to*

comprise a substantially random distribution of units of data of the second data set [output of Desai's *E*-CTRT algorithm]." EX1002, ¶389.

With respect to Claim 5, Orsini's data splitting process in Torre/Desai(/Watanabe)/Orsini System-1 "*us[es] a key* [Orsini's random key]" to "*caus[e] the n shares* [the shreds] *to comprise a substantially random distribution*." EX1002, ¶390.

With respect to Claim 6, Orsini's data splitting process in Torre/Desai(/Watanabe)/Orsini System-1 "*substantially randomly distribut[es] units of data* [in the output of Desai's *E*-CTRT algorithm] *using a key* [Orsini's random key]." EX1002, ¶391.

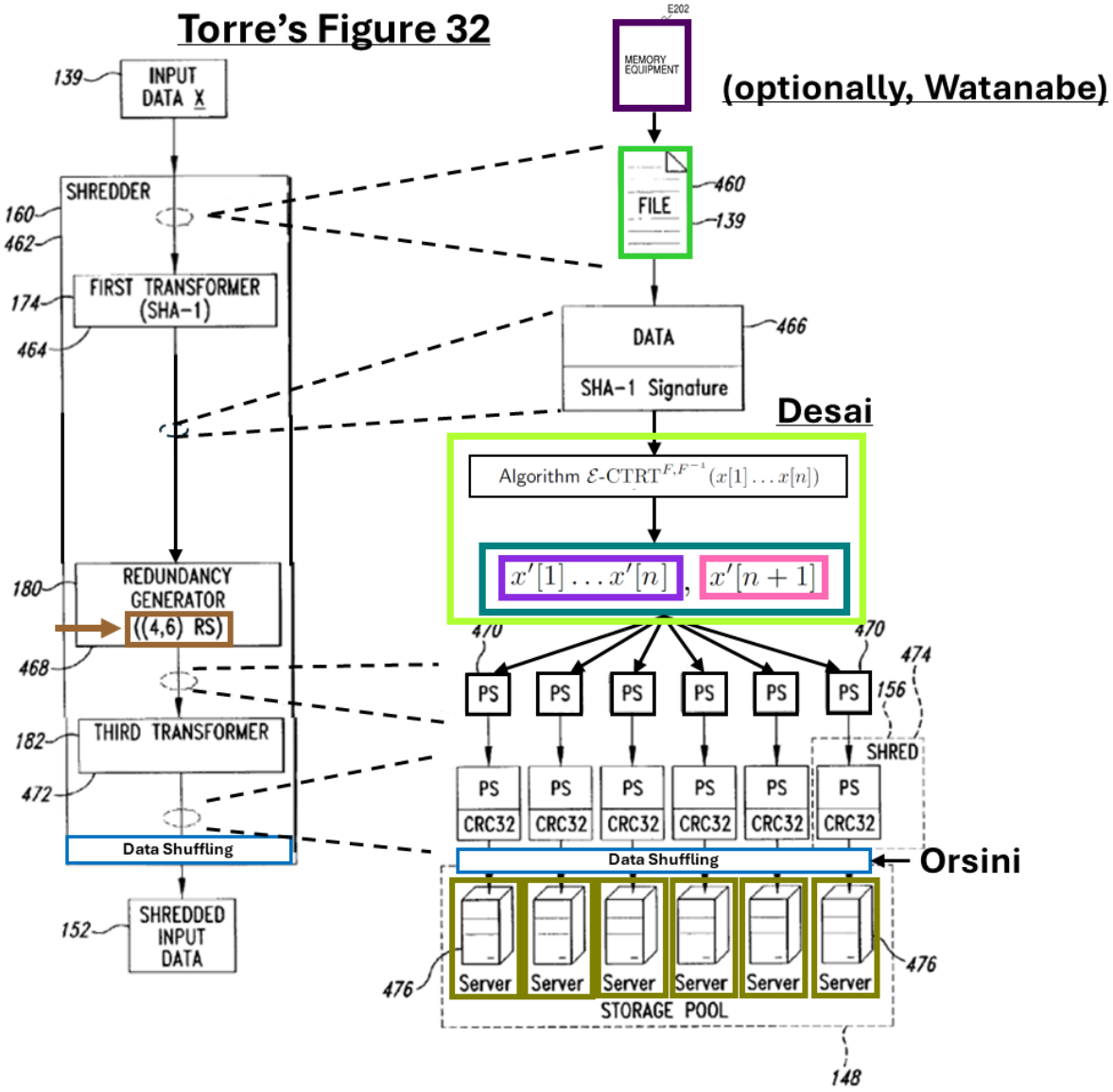
With respect to Claim 7, Torre/Desai(/Watanabe)/Orsini System-1 "*produc[es] the plurality of n shares* [shreds] *[by] substantially randomly distributing units of data of the second data set* [the output of Desai's *E*-CTRT algorithm] *into the plurality of n shares* [shreds]" using Orsini's cryptosplit process, Torre's redundancy generator, and Torre's third transformer. EX1002, ¶392.

4. Claim 8: The method of claim 1, wherein producing the plurality of n shares comprises distributing units of data into the plurality of n shares and substantially randomly shuffling the units of data in the shares.

Ground I or Ground II in view of Orsini renders obvious Claim 8 because Torre/Desai(/Watanabe)/Orsini System-2 teaches the additional limitation of Claim 8. EX1002, ¶¶393, 395.

As explained in Sections IX.A.3 [Torre/Desai] and IX.C.2.ii [Torre/Desai(/Watanabe)/Orsini System-2], Torre/Desai(/Watanabe)/Orsini System-2's redundancy generator generates six preshreds 470 that are then processed to form shreds. EX1003, [0131]. Then Orsini's data shuffling process further shuffles the shreds to add further randomness in the distribution of data units in the shreds. EX1006, [0281] ("The units are then distributed (either randomly or by a predetermined set of values) into the N shares. This distribution could also involve a ***shuffling of the order of the units in the shares***."), [0284], [0286]. EX1002, ¶394.

Torre/Desai(/Watanabe)/Orsini System-2 (Shredder)



EX1003, FIG. 32; EX1005, FIG. 2; EX1004, 16; EX1006, [0281], [0286].

Accordingly, Torre/Desai(/Watanabe)/Orsini System-2 "*produc[es] the plurality of n shares* [shreds] *comprises distributing units of data into the plurality of n shares* [using Torre's redundancy generator and third transformer] *and*

substantially randomly shuffling the units of data in the shares [using Orsin's data shuffling process]." EX1002, ¶395.

5. Claim 9: The method of claim 8, wherein substantially randomly shuffling the units of data comprises using a key.

Ground I or Ground II in view of Orsini renders obvious Claim 9 because Torre/Desai(/Watanabe)/Orsini System-2 teaches the additional limitation of Claim 9. EX1002, ¶¶396, 398.

Orsini explains that the cryptosplit/data shuffling process uses a "key" to randomly distribute the data units. EX1006, [0282], [0284]. Specifically, Orsini teaches generating a "key" with a sequence of random numbers, and using the key to randomly distribute the data units between shares. *See* EX1006, [0282] ("Assuming a random distribution, a ***key*** would be obtained to create a sequence of 23 random numbers (r1, r2, r3 through r23)."). *See also* EX1006, [0281]. EX1002, ¶397.

6. Claims 16–21, 28–29

- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claims 16 and 28 as provided in Section IX.C.3 [Claim 4];

- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claims 17 and 29 as provided in Section IX.C.3 [Claim 5];
- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claim 19 as provided in Section IX.C.3 [Claim 6];
- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claim 18 as provided in Section IX.C.3 [Claim 7];
- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claim 20 as provided in Section IX.C.4 [Claim 8]; and
- Torre/Desai/(Watanabe)/Orsini System teaches the additional limitations of claim 21 as provided in Section IX.C.5 [Claim 9].

EX1002, ¶399.

X. Conclusion

Petitioner respectfully requests that the Board institute *inter partes* review, hold the challenged claims unpatentable, and cancel the challenged claims.

Dated: July 9, 2025

Respectfully submitted,

/Taeg Sang Cho/
Taeg Sang Cho (Reg. No. 69,618)
DESMARAIS LLP
230 Park Ave
New York, NY 10169
212-808-1060 (telephone)
212-351-3401 (facsimile)

Counsel for Petitioner
International Business Machines Corporation

CERTIFICATION UNDER 37 C.F.R. § 42.24(D)

I hereby certify that this Petition for *Inter Partes* Review of U.S. Patent No. 9,135,456 has, excluding the portions exempted under 37 C.F.R. § 42.24(a), 13,896 words as counted by the word-processing system used to prepare this document, in compliance with 37 C.F.R. § 42.24(d).

Dated: July 9, 2025

Respectfully submitted,

/Taeg Sang Cho/
Taeg Sang Cho (Reg. No. 69,618)
DESMARAIS LLP
230 Park Ave
New York, NY 10169
212-808-1060 (telephone)
212-351-3401 (facsimile)

Counsel for Petitioner
International Business Machines Corporation

CERTIFICATE OF SERVICE

Under 37 C.F.R. §§ 42.6(e) and 42.105, the undersigned certifies that on July 9, 2025, complete copies of the foregoing and any accompanying exhibits were caused to be served by sending them via Federal Express Priority Overnight shipping, which is at least as fast and reliable as U.S. Priority Mail Express, to the correspondence address of record for U.S. Patent No. 9,135,456 as indicated in Patent Center:

Security First Innovations, LLC
c/o Farjami & Farjami LLP
26522 La Alameda Ave., Suite 360
Mission Viejo, CA

Dated: July 9, 2025

Respectfully submitted,

/Taeg Sang Cho/
Taeg Sang Cho (Reg. No. 69,618)
DESMARAIS LLP
230 Park Ave
New York, NY 10169
212-808-1060 (telephone)
212-351-3401 (facsimile)

Counsel for Petitioner
International Business Machines Corporation