

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTERNATIONAL BUSINESS MACHINES CORPORATION,
Petitioner,

v.

SECURITY FIRST INNOVATION, LLC,
Patent Owner.

Case IPR2025-01201
Patent 8,904,194

PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

	Page
I. INTRODUCTION	1
II. BACKGROUND.....	5
A. Overview Of The '194 Patent And The Challenged Claims.....	5
B. Overview Of Dickinson	9
C. Overview Of Hardjono.....	14
III. PETITIONER’S GROUNDS FAIL.....	16
A. Petitioner Fails To Demonstrate That Dickinson Teaches “Receiving, At The Electronic Computing System/Primary Interface, A Request To Retrieve The Data Set” (All Claims, All Grounds).....	16
B. Petitioner Fails To Demonstrate That Dickinson And Hardjono Disclose “Generating . . . A Plurality Of Shares” (All Claims, All Grounds).....	22
1. Dickinson Does Not Generate Shares By “Performing A Cryptographic Operation.”.....	22
2. Dickinson Does Not Disclose “Distributing The Data Set In The Plurality Of Shares Such That The Data Set Can Be Reconstructed Using Any Subset Of The Shares That Includes At Least A Minimum Number Less Than All Of The Shares.”	25
C. A POSITA Would Not Be Motivated To Combine Dickinson And Hardjono To Achieve The Claimed Inventions, Including The Limitation Of “Sending The Data Set Responsive To The Request” (All Claims, All Grounds).	30
IV. CONCLUSION	41

TABLE OF AUTHORITIES

Page(s)

COURT DECISIONS

Arctic Cat Inc. v. Bombardier Rec. Prods.,
876 F.3d 1350 (Fed. Cir. 2017)40

Arctic Cat Inc. v. Polaris Indus.,
795 F. App'x. 827 (Fed. Cir. 2019)40

Henny Penny Corp. v. Frymaster LLC,
938 F.3d 1324 (Fed. Cir. 2019)40

In re Schweickert,
676 F. App'x. 988 (Fed. Cir. 2017)38

Ruiz v. A.B. Chance Co.,
357 F.3d 1270 (Fed. Cir. 2004)41

Virtek Vision Int'l ULC v. Assembly Guidance Sys., Inc.,
97 F.4th 882 (Fed. Cir. 2024) 34, 38

AGENCY DECISIONS

Hulu LLC v. Sound View Innovations LLC,
IPR2018-00582, Paper 34 (Aug. 5, 2019) (informative).....38

EXHIBIT LIST	
2001	Lowenstein Declaration In Support of Notice of Intent
2002	Woo Declaration In Support of Notice of Intent
2003	Complaint in <i>Sec. First Innovations, LLC v. Int’l Bus. Machs. Corp.</i> , 1-25-cv-00514 (E.D. Va. Mar. 24, 2025), ECF No. 1 [Complaint]
2004	“Security First Corp. Presentation,” (March 4, 2015) [3/4/15-O’Hare Presentation] (CONFIDENTIAL)
2005	“IBM and Security First Corp to co-develop security capability for cloud computing,” PROACTIVE (last updated Nov. 9, 2010), https://www.proactiveinvestors.co.uk/companies/news/74758/ibm-and-security-first-corp-to-co-develop-security-capability-for-cloud-computing-9918.html [Proactive-Investors]
2006	“SFC – IBM Status Report” (September 16, 2011) (slip sheet omitted) [9/16/11 SFC-IBM Status Report] (CONFIDENTIAL)
2007	“IBM and Security First Corp. to Develop Integrated Security Technology,” SECURITY TODAY (Aug. 1, 2011), https://securitytoday.com/articles/2011/08/01/ibm-and-security-first-corp.-to-develop-integrated-security-technology.aspx [Security-Today]
2008	[8/24/09 Email] (slip sheet omitted) (CONFIDENTIAL)
2009	[11/11/13 Presentation] (CONFIDENTIAL)
2010	[2/15 Presentation] (CONFIDENTIAL)
2011	[3/11 IBM Presentation] (slip sheet omitted) (CONFIDENTIAL)
2012	Excerpts from File History of U.S. Patent Application No. 16/197,275 [’275-FH]
2013	U.S. Patent No. 8,447,695 [’695 patent]

2014	Excerpts from File History of U.S. Patent No. 8,447,695 [’695-FH]
2015	Licensed Works Agreement [LWA] (CONFIDENTIAL)
2016	2013 Statement of Work [2013 SOW] (CONFIDENTIAL)
2017	2015 Statement of Work [2015 SOW] (CONFIDENTIAL)
2018	“IBM Completes Acquisition of Cleversafe,” PRITZKER GROUP (Nov. 6, 2015), https://www.pritzkergroup.com/ibm-completes-acquisition-of-cleversafe/
2019	Jing Cao, “IBM Paid \$1.3 Billion to Acquire Cleversafe in Hybrid-Cloud Push,” BLOOMBERG (Feb. 24, 2016), https://www.bloomberg.com/news/articles/2016-02-24/ibm-paid-1-3-billion-to-acquire-cleversafe-in-hybrid-cloud-push (slip sheet omitted)
2020	[10/9/06 Email] (slip sheet omitted) (CONFIDENTIAL)
2021	[12/6/16 Email] (slip sheet omitted) (CONFIDENTIAL)
2022	Reserved
2023	Order Granting Motion to Stay in <i>Sec. First Innovations, LLC v. Int’l Bus. Machs. Corp.</i> , 1-25-cv-00514 (E.D. Va. Aug. 20, 2025), ECF No. 88 [Stay Decision]
2024	Chart of Settled Expectations Decisions [Chart]
2025	Scheduling Order in <i>DivX, LLC v. Amazon.com, Inc.</i> , No. 1:24-cv-2061 (E.D. Va. July 1, 2025), ECF No. 69 [Recent Scheduling Order]
2026	Opposition to Motion to Stay in <i>Sec. First Innovations, LLC v. Int’l Bus. Machs. Corp.</i> , 1-25-cv-00514 (E.D. Va. July 29, 2025), ECF No. 71 [Stay Opposition] (CONFIDENTIAL)
2027	U.S. Patent No. 11,178,116

2028	U.S. Patent No. 11,068,609
2029	U.S. Patent No. 10,452,854
2030	U.S. Patent No. 7,187,771 [Dickinson-'771]
2031	Declaration of Aviel D. Rubin, PhD. [Rubin-Decl.]
2032	Excerpts from MICROSOFT COMPUTER DICTIONARY, (5th ed. 2002) [Microsoft-Computer-Dictionary]
2033	Excerpts from CAMBRIDGE ESSENTIAL ENGLISH DICTIONARY, (2nd Ed. 2011) [Essential-American-English-Dictionary]
2034	Kyle Chin, <i>Biggest Data Breaches in US History (Updated 2025)</i> , UPWARD (last updated June 30, 2025), https://www.upguard.com/blog/biggest-data-breaches-us [UpGuard]

I. INTRODUCTION

Before reaching the merits of the Petition, the Deputy Director should exercise her discretion and deny institution. *First*, the patent-at-issue has been in force for over a decade, and Petitioner has known about it for years. *Second*, the original examination was correct. In fact, substantially similar Dickinson reference and Hardjono—the key references raised by Petitioner—were considered by the Examiner during prosecution. Petitioner points to no error by the Examiner, and for the reasons discussed in this POPR, there was none.

The Petition should be denied as it fails to demonstrate a reasonable likelihood of prevailing as to at least one of the challenged claims of the '194 Patent for multiple independently sufficient reasons. For each limitation of the independent claims (claims 1, 7, and 14), Petitioner relies on either Dickinson alone or Dickinson combined with Hardjono to try to show obviousness under 35 U.S.C. § 103.¹ But neither Dickinson nor Hardjono (or their combination) discloses key limitations and Petitioner fails to establish a motivation to combine.

As discussed below, the '194 Patent claims and Petitioner fails to demonstrate that its proposed combination teaches, *inter alia*, “generating . . . a plurality of shares,” “receiving, at [the electronic computing system/a primary interface a]

¹ Petitioner relies on Moulton only for the dependent claims.

request to retrieve the data,” and “sending the data set responsive to the request.” *See* Ex. 1001 [’194], cls. 1, 7, 14. In accord, the ’194 Patent discloses a “secure data parser . . . that may be integrated into any suitable system for securely storing and communicating data.” *Id.*, Abstract. The secure parser comprises systems, methods, and computer instructions “for securely storing virtually any type of data from unauthorized access or use” by, among other things, “parsing, splitting and/or separating the data” into two or more shares. *Id.*, 2:33-51. The shares may be stored “in multiple locations” and subsequently retrieved to “reconstitut[e] or re-assembl[e]” the original data “for authorized access or use.” *Id.*, 2:45-48.

Dickinson, in contrast, discloses a system and “method for facilitating an authentication related to an electronic transaction” between a user and a vendor by verifying the user’s identify. Ex. 1003 [Dickinson], Abstract. Dickinson’s system simply compares securely retained “sensitive” data about a user (“enrollment authentication data”)—such as a fingerprint or mother’s maiden name—against authentication data provided by the user at the time of the transaction. *Id.* Based on that comparison, the system generally returns a binary response to the vendor—either the user’s identity is verified or it is not. *Id.*, 28:20-24. Dickinson does not receive a request for the retained “enrollment authentication data” (the alleged data set), nor does Dickinson retrieve that data to send it to the user or the vendor. In other words, no stored data is ever requested, and it does not ever leave Dickinson’s

system. Ex. 1003 [Dickinson], 28:30-31 (“the authentication result transmitted . . . does not include the sensitive data”).

But Petitioner fundamentally, and misleadingly, characterizes Dickinson as “a method for securely storing and retrieving data.” *See* Pet., 34 (quoting Ex. 1001 [’194], cl. 1[Pre]). To take just one example, Petitioner claims that Dickinson’s system “receives a request to write or store a data set.” Pet., 15 (citing Ex. 1003 [Dickinson], 19:16-34). It does not; the system receives requests to authenticate or verify a user’s identity—and nothing in Dickinson says otherwise. *See* Ex. 1003 [Dickinson], 3:3-5 (“The method comprises receiving a request for an authentication transaction from a vendor.”). Dickinson’s authentication process is designed to verify a user’s identity, not to send the authentication data used to perform the verification back to the user or vendor, and the ’194 Patent clearly distinguishes a request to retrieve the data set from an authentication request.

Because Dickinson discloses an authentication method and not a method for securing storing and retrieving data, it fails to disclose key limitations of the independent claims. In particular, Dickinson fails to disclose an electronic computer system that: (1) “receiv[es], at the electronic computing system, [a] request to retrieve the data set” (*id.*, cls. 1[C], 7[B-3]); (2) “generat[es] . . . a plurality of shares by performing a cryptographic operation” and “distributing [a] data set in the plurality of shares (Ex. 1001 [’194], cls. 1, 7[B-1]); or (3) “send[s] the data set

responsive to the request” back to the requestor (*id.*, cls. 1[G], 7[B-7]). And although Petitioner relies on the combination of Dickinson and Hardjono, for the reasons stated below, Hardjono does not provide what Dickinson lacks. *See* Section III.

Not only that, the Petition fails to show a plausible motivation to combine Dickinson and Hardjono to obtain the claimed invention. Each reference is directed to a different method and purpose, and those differences render the references incompatible. Dickinson is directed to authenticating users based on enrollment authentication data, and Hardjono is directed to encrypted data storage. Yet Petitioner contends that a POSITA would have been motivated to combine Dickinson and Hardjono in such a way that a “data set” is “sen[t]” in response to Dickinson’s *authentication* request. More specifically, Petitioner contends that the “enrollment authentication data” from Dickinson would be reconstructed and sent to the vendor as it contends is taught under Hardjono. But that makes no sense. It provides no benefit to the users/vendors that use Dickinson’s system, and it renders the enrollment authentication data insecure.

Nevertheless, contrary to everything in Dickinson, Petitioner asserts in conclusory fashion that Dickinson’s authentication “workflow” is somehow “incomplete” unless the enrollment data itself is sent back to the vendor. *See* Pet., 26, 49. The Petition provides no evidence for this assertion but the equally

conclusory testimony of its expert. *See id.*; *see also* Ex. 1002 ¶ 201. Based on this, Petitioner proposes adding Hardjono’s alleged teaching of sending the data set in response to the request. But the vendor in Dickinson gets precisely the information it needs without the data set itself—*i.e.*, confirmation that the user is or is not authentic. The vendor does not want or need the underlying enrollment authentication data, and Dickinson makes perfectly clear that such data should be “assembled only inside the authentication engine 215” of Dickinson’s system, and should not be made available outside the system. Ex. 1003 [Dickinson], 28:25-31. Petitioner fails to show why a POSITA would have been motivated to combine Dickinson with Hardjono when doing so would make Dickinson’s system less secure—which runs counter to Dickinson’s purpose of providing “security to transactions in electronic commerce.” *See* Section III.C.

II. BACKGROUND

A. Overview Of The ’194 Patent And The Challenged Claims

The ’194 Patent relates to, *inter alia*, a cryptographic system to secure data. As Dr. Rubin explains:

The ’194 Patent’s inventions and related disclosure generally concern securing “*any* data set” by encrypting the data set and splitting the encrypted data set into shares using “cryptographic splitting” or a “cryptosplit” process.” *See, e.g.*, Ex. 1001 [’194], 52:12-13 (teaching that the data set to be secured is “encrypt[ed], cryptographically split,

dispersed and securely stored in multiple locations”). Following encryption of the data set, a cryptosplit “partitions the data [set] into N number of shares” based on “any size unit of data” including “bits, bytes, kilobytes, megabytes or larger units.” *Id.*, 52:40-44. The units are “distributed (either randomly or by a predetermined set of values)” into the N shares. *Id.*, 52:51-53. This means that each share “can be viewed as a sequence of these units.” *Id.*, 52:46-47.

The '194 Patent also discloses “redundancy functionality,” which appends redundancy information to each share so that the data set may be recreated from fewer than all of the shares. *Id.*, 72:36-39.

The '194 Patent also discloses storing the shares on a plurality of different storage devices, as well as the ability to retrieve the shares of the data set from storage, reconstruct the data set by reversing the storage process, and return the reconstructed data set to the user. *See id.*, cls. 1, 7, 14. To facilitate an efficient share retrieval process, the '194 Patent teaches that when a user requests stored data, the system should (i) identify a “set” of fastest-responding storage devices that contain the minimum number of shares necessary to retrieve the data; (ii) reconstruct the data set; and (iii) send the reconstructed data set to the user. *Id.*

The overall system/process used by the secure parser is shown in Figures 31 and 32 (below). Figure 31 shows a block diagram of a process used to “write” a data set “to a storage device.” *Id.*, 70:43-45. At 3100, the user selects the data to be stored, and the Secure Data Parser Core is called. *Id.*, 70:46-48. The Secure Data Parser Core

“parses and splits” the data set, adds redundancy information, and places the shares in Split Data Buffers 3010. *Id.*, 70:54-57; *see also id.*, 69:49-52. The shares are then distributed to different locations for storage by module 3014. Figure 32 shows the reconstruction of the data set when it is “read” from a “storage device.” *Id.*, 70:64-66. At 3200, the data to be restored is identified, and Secure Data Parser Core 3000 is called. The Secure Data Parser Core reverses the process of Figure 31: The shares are collected from storage locations, redundant information is removed (or fewer than all of the shares are used), and the data set is assembled from the shares and transmitted to the requestor. *Id.*, 71:1-15.

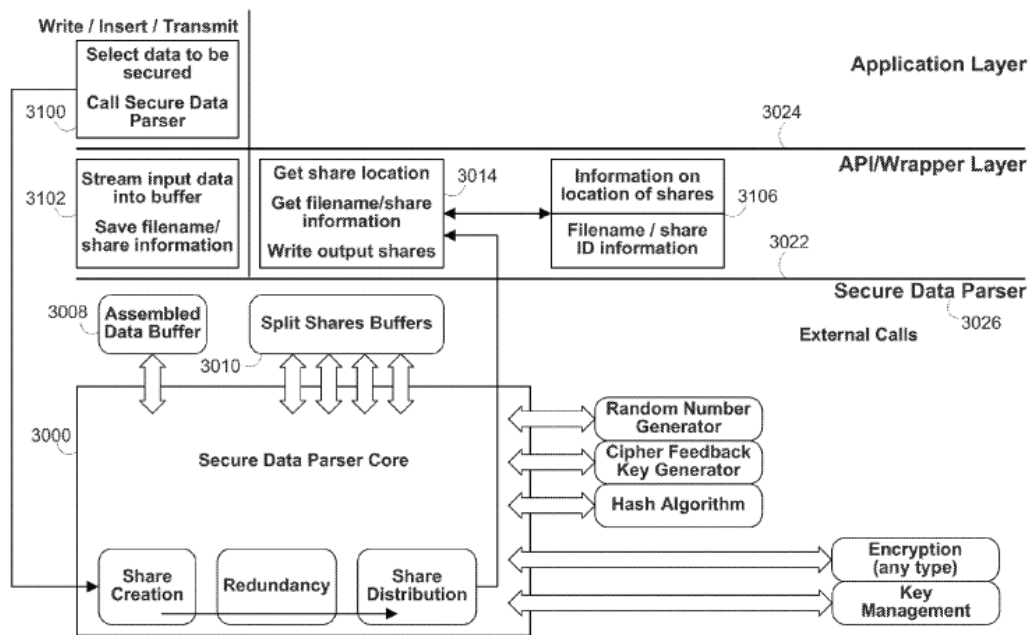


FIG. 31

Ex. 1001 [’194], Fig. 31

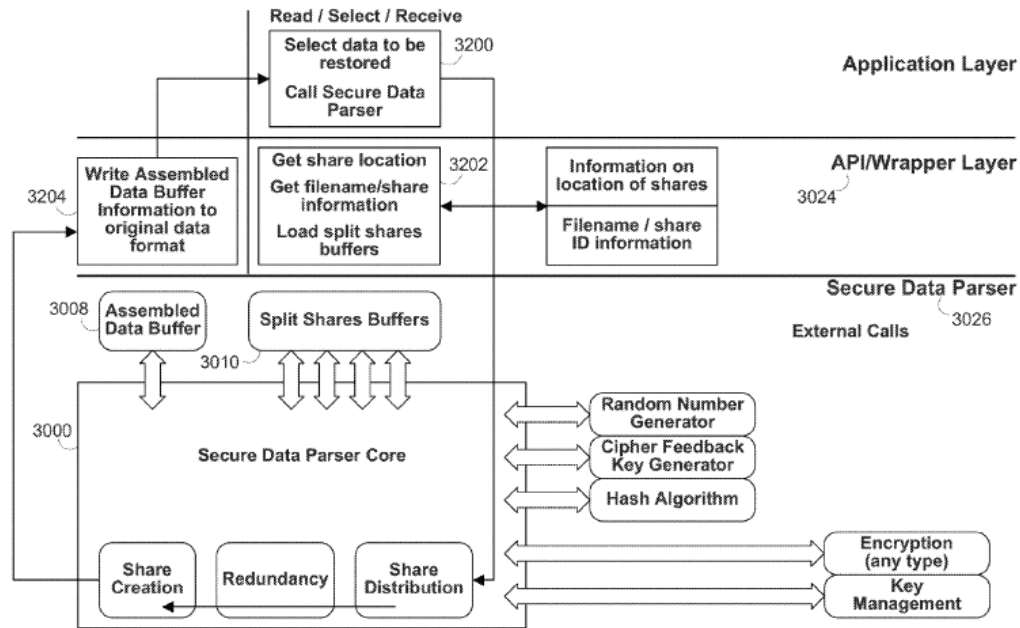


FIG. 32

Id., Fig. 32

The independent claims of the '194 Patent teach the above-described process. They require encrypting a data set by “performing a cryptographic operation;” parsing the encrypted data set into shares “such that the data set can be reconstructed using . . . less than all of the shares;” and storing the shares on a plurality of storage devices. *Id.*, cls. 1, 7, 14. Then, on receiving a “request to retrieve the data set,” identifying “a set of fastest-responding storage devices”; “retrieving from the set . . . the minimum number of shares” needed to reconstruct the data set; “reconstructing the data set from the minimum number of shares”; and finally “sending the dataset” in response to the request. *Id.*

Ex. 2031 [Rubin-Decl.] ¶¶ 35-39. For the reasons explained below, Petitioner fails to demonstrate that its Dickinson and Dickinson/Hardjono-based grounds teach multiple limitations of the '194 Patent.

B. Overview Of Dickinson

A substantially similar Dickinson reference was before the Examiner during prosecution of the '194 Patent and is listed on the face of the patent. Ex. 1001 ['194], 4.

Dickinson's goal is to meet the "need . . . for [an] authentication system[] which provide[s] security for electronic transactions" between a vendor and user that is "sufficient for the needs of a vendor without unnecessary inconvenience to the user." *See* Ex. 1003 [Dickinson], 2:8-11. Dickinson discloses a secure server or "trust engine" that stores "cryptographic keys and user authentication data."² *Id.* Dickinson's trust engine includes an "authentication engine" that compares the user's "enrollment authentication data," stored within Dickinson's secure server, with "current authentication data" provided by the user at the time of a transaction. *Id.*, 3:3-13. If the user's "current authentication data" matches their "enrollment authentication data," Dickinson will confirm for the vendor that the user is authentic. *Id.*, 5:24-25, 55:11-16. A positive comparison authenticating the user allows him or

² "Authentication data" as envisaged by Dickinson could be an "identification number, one or more biometrics, [or] a series of questions" about a user's "place of birth, address, anniversary, . . . mother's maiden name, favorite ice cream, or the like"). Ex. 1003 [Dickinson], 10:20-29.

her to carry out a number of cryptographic operations, including “authentication, authorization, digital signing and generations, storage and retrieval of certificates,” etc. *Id.*, 2:8-14.

In sharp contrast to the '194 Patent, Dickinson does not disclose either encryption or storage of a data set (other than the enrollment authentication data for purposes of comparison) or returning a previously stored data set to the user at the user or vendor's request. The enrollment authentication data is never provided to the user or vendor; in fact, once stored, it never leaves Dickinson's system—specifically, Dickinson's “Trust Engine 110.” A block diagram of the trust engine is shown in Figure 2 of the reference (below).

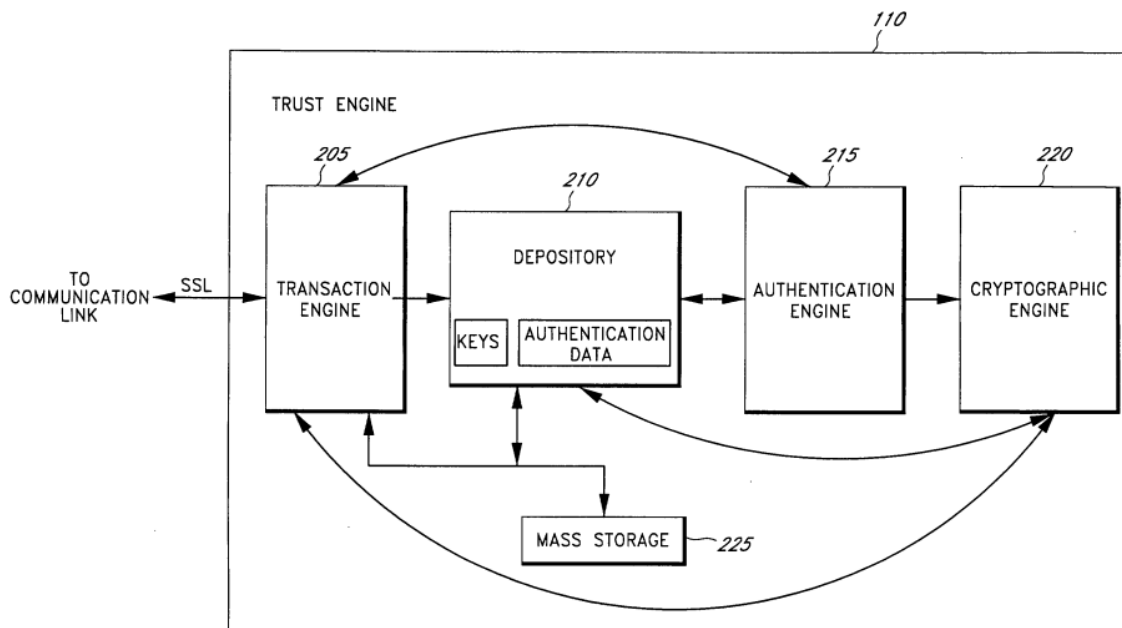


FIG. 2

Ex. 1003 [Dickinson], Fig. 2

Dr. Rubin explains the various modules within the trust engine:

As Dickinson explains, Trust Engine 110 comprises a “transaction engine 205,” a “depository 210,” “an authentication engine 215”, and a cryptographic engine 220. *Id.*, 13:5-7. The transaction engine 205 provides the sole external communication link with the Trust Engine 110. It provides front end security by “receiv[ing] incoming data,” including authentication requests, over the Communication Link and routing that data to the appropriate module of the trust engine. *Id.*, 13:15-28. Depository 210 comprises one or more storage facilities for storing private encryption keys (corresponding to users and vendors) and enrollment authentication data, none of which ever leaves Trust Engine 110. *Id.*, 14:17-19, 28:30-31 (“Moreover, the authentication result transmitted to the vendor does not include the sensitive data.”). Authentication engine 215 “comprises a data comparator” that compares authentication data from transaction engine 205 with enrollment authentication data from depository 210, to verify a user’s identity. *Id.*, 15:4-12. Finally, the cryptographic engine 220 is “configured to advantageously provide conventional cryptographic functions” to a user whose identity has been verified. *Id.*, 15:21-22. Dickinson suggests that these various functions could include “digital signing, encryption, decryption, hash creation, key generation,[] key destruction,” “logging into a portal,” or “unlocking a

password vault,” (*id.*, 55:33-56:9), but it does not teach methods to implement any of these cryptographic functions.^[3]

Dickinson’s trust engine works by receiving an authentication request from a vendor, which prompts the system to forward a transaction identifier (“TID”) and the user’s “enrollment authentication data” to its “authentication engine.” Ex. 1003 [Dickinson], 55:3-9. Next, Dickinson “quer[ies] the user for current authentication data and the TID,” which it also forwards to the “authentication engine.” *Id.*, 55:10-12. After comparing the “enrollment authentication data” and “current authentication data,” the “authentication engine” outputs a result that indicates whether the data match. *Id.*, 45:23-25, 55:15-16. This verifies the identity of the user and thus secures the transaction.

Central to Dickinson, however, is that the enrollment authentication data stored in depository 210 never leaves the trust engine. *See id.*, 28:30-31. This ensures that the private encryption keys and enrollment authentication data is provided “in an environment where they are not lost, stolen or compromised, thereby advantageously avoiding a need to continually reissue and manage new keys and authentication data.” *Id.*, 2:19-21. To that end, and contrary to

³ For example, Dickinson notes that the cryptographic engine 220 may generate public and private encryption keys on behalf of a user of the Trust Engine 110, such that “the private cryptographic keys are not available outside of the trust engine.” *Id.*, 15:20-30.

Petitioner’s characterization, Dickinson does not receive requests to “write or store a data set,” nor to retrieve or send data stored in depository 210 in response to such a request. *See* Pet., 15. Rather, Dickinson requires only a request for authentication and the authentication data as input, and it returns as output (1) a binary “YES/NO” result or (2) the result of a cryptographic operation (*i.e.*, a digital signature or certificate) if the user’s identify is verified. Ex. 1003 [Dickinson], 28:20-24, 31:17-21.

Dickinson also fails to disclose distributing a data set into “shares.” In fact, Dickinson uses the term “share” only three times in 54 pages of text and never to refer to a piece of a data set. *See generally id.* As explained below, there simply is no disclosure in Dickinson that partitions a data set into N shares by distributing units of that data set (*i.e.*, its bits, bytes, kilobytes, etc.) either randomly or by a “predetermined set of values” such that each share “can be viewed as a sequence of these units.” *See* Ex. 1001 [’194], 52:45-48.

Dickinson shares three inventors of the ’194 Patent: Orsini, O’Hare and Davenport. Petitioner claims that because Figures 1-20—and thus much of the discussions associated with those figures in the ’194 Patent—are the same as those in Dickinson, the reference “provides invalidating disclosures for most of the ’194 Patent’s claim limitations.” Pet., 12. That is simply wrong. Immediately *following* the portion of its specification that is also in Dickinson, the ’194 Patent states that “in a separate embodiment, the present invention comprises a complete system,” called the secure data parser, that operates “on any

data set.” Ex. 1001 [’194], 51:55-57. The limitations of the inventions claimed in the ’194 Patent are disclosed and enabled in the subsequent 15 figures and 25 columns of text that are not in Dickinson. These include “generating . . . a plurality of shares by “performing a cryptographic operation on a dataset and distributing the *data set* in the plurality of shares”; “receiving . . . a request to retrieve the data set”; and “sending the data set responsive to the request” to the requestor. *See id.*, 75:24-48.

Ex. 2031 [Rubin-Decl.] ¶¶ 43-47.

C. Overview Of Hardjono

Hardjono addresses a different concern than Dickinson: the “vulnerab[ility]” of data storage systems “comprising a single [encryption] key” that, if compromised, would allow “an unauthorized individual [to] access the data.” Ex. 1004 [Hardjono], 1:23-29. To address this concern, Hardjono teaches “a method and apparatus for data storage using distributed databases.” *Id.*, 1:46-47. As Dr. Rubin explains, its method includes, *inter alia*, generating a “plurality of shares” from an input data set such that “at least a subset” of the shares is required to recover the original data, then distributing the “plurality of shares . . . to a plurality of distributed databases.” *Id.*, 47-53:

Upon receipt of an input data block, Hardjono generates a password to associate with the data block “for use in later re-creation of the block of data.” Ex. 1004 [Hardjono], 9:48-51. Hardjono then

encrypts the data block with a key, generates a “first plurality of shares based on the block of data,” and “distribut[es] the first plurality of shares to a plurality of distributed databases.” *Id.*, 9:51-59. Next, Hardjono generates a “second plurality of shares based on the encryption key” and “distribut[es] the second plurality of shares to the plurality of distributed databases.” *Id.*, 9:59-65.

When Hardjono “receives a request to retrieve the block of data,” the system confirms that the request provided the correct password, accesses the required “subset of the plurality of databases to retrieve a second plurality of shares,” then reconstructs the input data block “using the second plurality of shares.” *Id.*, 10:7-14.

Ex. 2031 [Rubin-Decl.] ¶¶ 51-52.

Hardjono also teaches an “apparatus” to perform this method, comprising several connected modules that perform the various steps and communicate with other modules in the system. *See* Ex. 1004 [Hardjono], 10:23-54. These modules include (i) “a verification controller” that establishes the password, (ii) either a database or storage controller that stores the password, (iii) “a storage controller” that encrypts the data, (iv) “a share generator” that generates the first and second pluralities of shares, and (v) “a share distributor” that places the shares into the distributed databases. Ex. 1004, 10:23-54.

But Hardjono does not teach “generating . . . a plurality of shares by performing a cryptographic operation” as Hardjono merely teaches that shares are “create[ed]” or “generated.” *See, e.g.*, Ex. 1004 [Hardjono], 3:29-31, 3:39-40.

III. PETITIONER’S GROUNDS FAIL.

A. **Petitioner Fails To Demonstrate That Dickinson Teaches “Receiving, At The Electronic Computing System/Primary Interface, A Request To Retrieve The Data Set” (All Claims, All Grounds).**

Each of the independent claims of the ’194 Patent requires that a “request to retrieve the data set” is “received” at either the “electronic computing device” (independent claims 1, 14) or “the primary interface” (independent claim 7).⁴ Petitioner argues that Dickinson alone “discloses” this limitation for each of the independent claims. Pet., 44 (“Dickinson discloses 1[C].”), 60 (“Dickinson discloses [7B-3].”), 71 (“Dickinson discloses [14D] for the reasons provided in . . . [Ground I, 7B-3].”). In particular, Petitioner contends that this limitation is disclosed by Dickinson’s teachings that “the vendor system 120 . . . forwards . . . the

⁴ *See* Ex. 1001 [’194 Patent], cl. 1[C] (“receiving, at the electronic computing system, [a] request to retrieve the data set.”); 7[B-3] (“receiv[ing], via the primary interface, a request to retrieve the data set”); 14[D] (teaching that the “electronic computing device” is “cause[d]” to “receive a request to retrieve the data set.”).

authentication request to the trust engine 110,” and that “the transaction engine 205 [of trust engine 110] receives the [communication authentication request].” *E.g.*, Pet., 45 (quoting Ex. 1003 [Dickinson], 28:6-14). Not so.

The “request to retrieve the data set” in the ’194 Patent is a call to restore an identified data set, so that it may be sent back to the requester in its original format. This limitation is disclosed, for example, in Figure 32 of the ’194 Patent. That figure shows, as the patent explains, that a “request” to “read” an “identified” data set from a storage device may be “receive[d]” by the Secure Data Parser Core. Ex. 1001 [’194], 70:64-67; Ex. 2031 [Rubin-Decl.] ¶¶ 57. The ’194 Patent further explains that after “data to be restored is identified,” a “call to secure data parser 3000 is made from application layer 3024 [the requester]”; the request includes identification of the “data to be restored.” Ex. 1001 [’194], 70:67-71:2. The requested data set is then “assembled” by the secure parser and sent to the applications layer in its “original format” for transmission to the requester. *Id.*, 71:5-14. In other words, the Secure Data Parser Core, corresponding to the electronic computing system (or primary interface), “receives” a request from an external system which causes the Secure Data Parser Core to **reassemble the data** so that it is **converted to its “original format”** and may be sent back to the requester. *Id.*, 70:67-71:14.

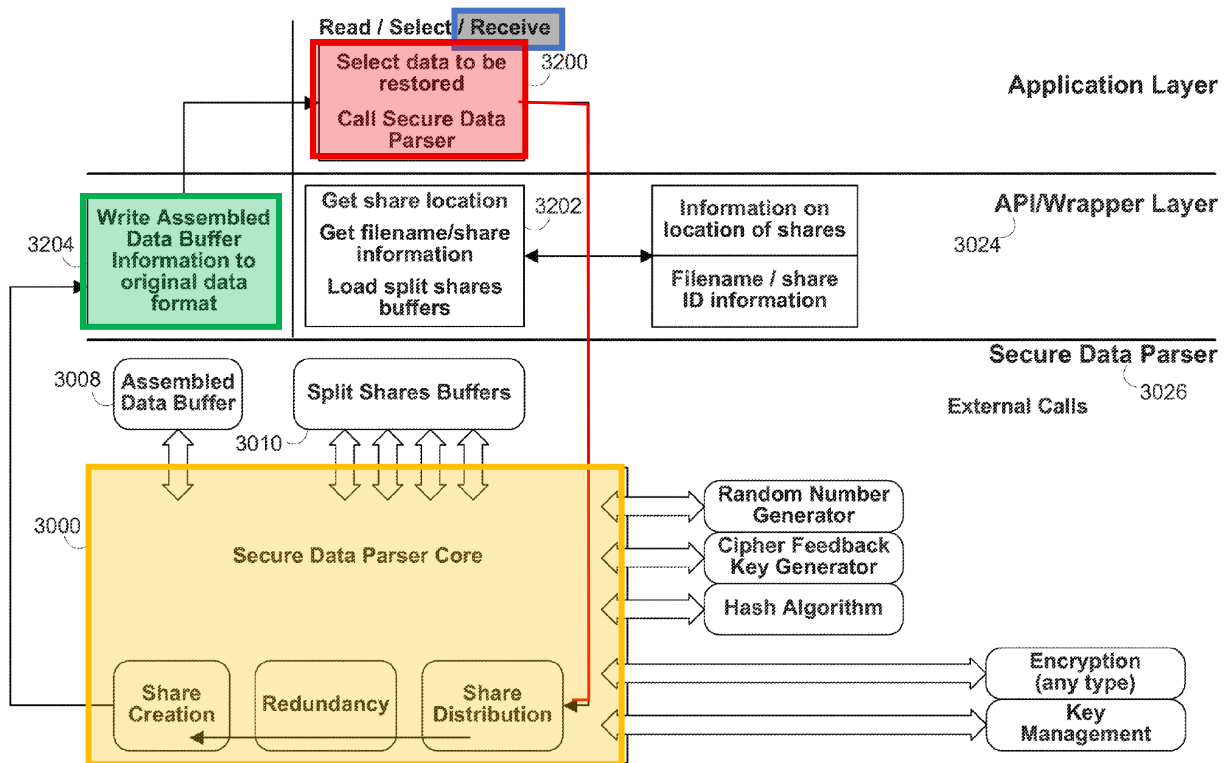


FIG. 32

Ex. 1001 [’194], Fig. 32 (annotated)

Dickinson fails to disclose this limitation because an “authentication request” is not the “request to retrieve a data set” disclosed and claimed in the ’194 Patent. Dickinson’s authentication requests are not directed to retrieving or restoring an identified data set. Rather, Dickinson’s requests are directed to authenticating a user, *i.e.*, by confirming that the current authentication data matches the enrollment authentication data. *See, e.g.*, Ex. 1003 [Dickinson], 27:15-16 (“[T]he authentication process 1000 includes gathering current authentication data from a user and **comparing** that to the enrollment authentication data of the user.”). The “authentication request” in Dickinson on which the Petition relies is simply a request

to compare information and provide a response based on the result of the comparison—*i.e.*, whether or not the user is verified. As Dickinson explains, “authentication is the process of proving that a user is who he says he is” and, “[g]enerally, authentication requires demonstrating some fact to an authentication authority.” *Id.*, 37:24-29. Notably, Dickinson’s authentication process does not need the vendor to receive the enrollment authentication data.

Petitioner relies solely on Dickinson’s “authentication request”—*i.e.*, a request to determine whether the user “is who he says he is” (*id.*, 37:24)—and not a request to retrieve a particular data set for the user or vendor. Pet., 49. But as Dickinson makes clear, an authentication request is not a request to retrieve a data set. Indeed, Dickinson expressly teaches that the response to an authentication request “does not include” the underlying enrollment authentication data—the only “data set” stored by the secure data parser. Ex. 1003 [Dickinson], 28:30-31 (“Moreover, the authentication result transmitted to the vendor does not include the sensitive data, and the user may not even know whether he or she produced valid authentication data.”).

This distinction—between an authentication request and a request to “retrieve the data set”—is important and is made expressly clear in the ’194 Patent itself. The ’194 Patent’s specification specifically distinguishes “requests to retrieve the data set” as discussed, *e.g.*, in connection with Figure 32 (*see supra* Section II.A; *see also*

Ex. 1001 [’194], 70:65-71:15), from “authentication requests.” *See, e.g., id.*, 26:24-27:52; *see also id.*, 36:21-22 (“authentication is the process of proving that a user is who he says he is”).

Relatedly, Petitioner also purports to rely on Dickinson’s disclosure that, in response to receiving an authentication request, the transaction engine of the trust engine generates a “request for the user’s enrollment authentication data to be assembled.” Pet., 45. But Petitioner’s reliance is misplaced because that request in Dickinson is not “receiv[ed]” at “the electronic computing system” or “the primary interface,” as required by the ’194 Patent’s independent claims. *See* Ex. 1001, cls. 1, 7, 14. Petitioner identifies Dickinson’s trust engine as the “electronic computing system” and the “primary interface.” *See, e.g.,* Pet., 35 (“A POSITA would have understood that the trust engine is an example of an electronic computing system.”). But the trust engine does not *receive* a “request for the user’s enrollment authentication data to be assembled”—it *generates* it. Dickinson makes clear that it is the transaction engine that generates the “request for the user’s enrollment authentication data.” Ex. 1003 [Dickinson], 28:9-11. And the *transaction engine* is a module of—*i.e.*, inside of—the *trust engine*:

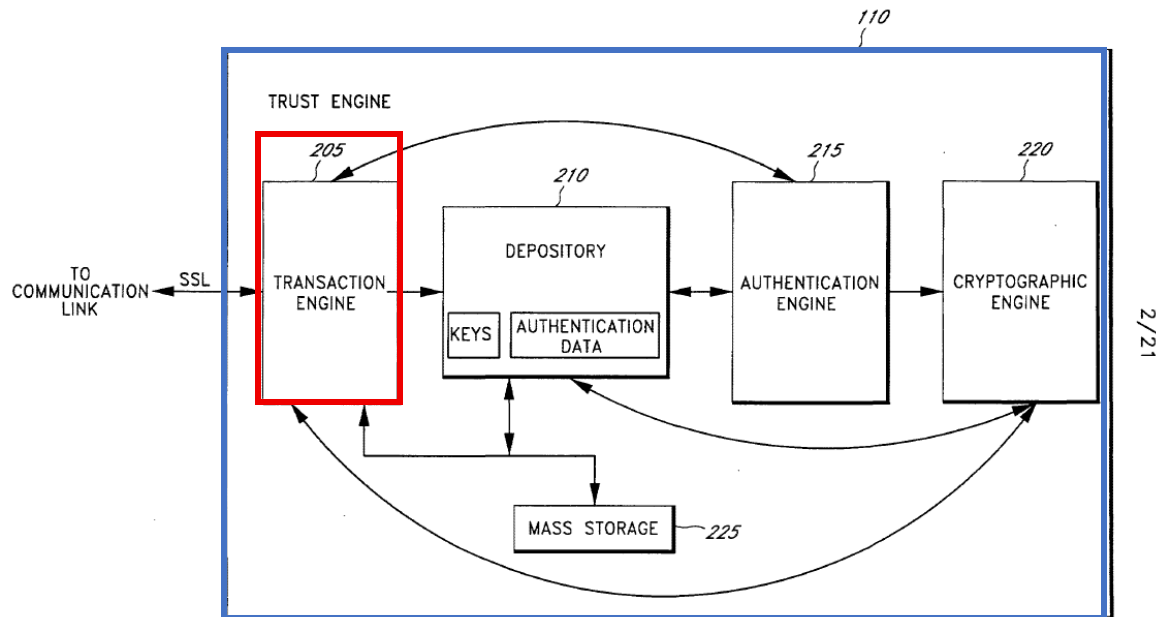


FIG. 2

Id., Fig. 2 (annotated); 13:6-8 (“[T]rust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220.”).

Petitioner offers no explanation for how Dickinson’s trust engine “receives” a request it generates internally. Certainly, the ’194 Patent does not suggest the electronic computing system would “receive” an internally generated request, confirming a prototypical receipt comes over a network. *See* Ex. 1001 [’194], 70:64-67 (disclosing that a request can be “receive[d] (e.g., from a network)”). In this, the ’194 Patent’s usage accords with the common understanding of “receive.” Ex. 2032 [Microsoft-Computer-Dictionary], 442 (“receive *vb.* To accept data from an *external* communications system”); Ex. 2033 [Essential-American-English-

Dictionary], (“to get something that *someone* has given or sent to *you*.”).

For these reasons, Petitioner has failed to show that the authentication requests disclosed in Dickinson constitute a “request to retrieve the data set” at an “electronic computing system” or “primary interface” as required by the independent claims of the ’194 Patent.

B. Petitioner Fails To Demonstrate That Dickinson And Hardjono Disclose “Generating . . . A Plurality Of Shares” (All Claims, All Grounds).

The ’194 Patent’s independent claims also require “generating a . . . plurality of shares” by “performing a cryptographic operation on a data set” and “distributing the data set in the plurality of shares” Ex. 1001 [’194] cls. 1[A], 7[B], 14[B]. Petitioner relies entirely on Dickinson for these limitations, but its reliance is unfounded and its arguments are misleading. Dickinson does not disclose the required limitations both because it does not disclose “performing a cryptographic operation” in creating the shares and because it does not disclose distributing the *data set* into the shares.

1. Dickinson Does Not Generate Shares By “Performing A Cryptographic Operation.”

The ’194 requires “generating . . . a plurality of shares” by first “performing a cryptographic operation on a data set.” Ex. 1001 [’194], cls. 1[A], 7[B], 14[B]. The ’194 Patent discloses that “cryptography,” in general, refers to “protecting data

by transforming, or encrypting, it into an unreadable format” such that “[o]nly those who possess the key(s) to the encryption can decrypt the data into a useable format.” *Id.*, 1:38-41. As a result, Dr. Rubin explains, a POSITA would have understood “cryptographic operation” as used in the ’194 Patent to mean operations that require key encryption, “such as encrypting a document with a particular key.” Ex. 2031 [Rubin-Decl.] ¶¶ 67-69.

The ’194 Patent discloses several embodiments in which shares are generated by first encrypting the data set and then distributing the data set into shares. For example, the ’194 Patent discloses an embodiment in which “encryption may occur prior to the splitting of the data set by the splitting module or secure data parser.” Ex. 1001 [’194 Patent], 52:29-31; *see also id.*, 52:12-13 (disclosing embodiment in which the data set is “encrypt[ed], cryptographically split, dispersed and securely stored in multiple locations.”), 71:32-33 (“In one suitable approach, original data 3306 may be encrypted prior to parsing, splitting or both.”); Figs. 21-24; Figs. 31-32 (disclosing “Encryption—Any Type” as a capability of the Secure Data Parser Core).

Petitioner relies entirely on Dickinson as disclosing this limitation, and points to three purported teachings. But none discloses the limitation.

First, Petitioner claims that “Dickinson Figure 5” discloses a data splitting module 520 that has “the ability to mathematically operate on various data so as to

. . . split the data into portions.” Pet., 37. But “mathematically operate” does not disclose that the portions are rendered undecipherable without a key, or any other cryptographic operation. See Ex. 2031 [Rubin-Decl.] ¶¶ 70-71.

Second, the Petition relies on Figure 8’s disclosure that the “data splitting module . . . generates a substantially random number value, or string or set of bits.” Pet., 37-38. Again, this does not establish that the random numbers are used to perform a cryptographic operation on the data set; the Petition does not explain, in connection with this limitation, whether or how the random number in Dickinson is used. *Id.*; see Ex. 2031 [Rubin-Decl.] ¶ 72.

Third, Petitioner relies on Dickinson’s purported disclosure that “‘the trust engine 110 may advantageously perform . . . data encryption and decryption’ as it generates the plurality of shares.” Pet., 38 (quoting Ex. 1003 [Dickinson], 16:8-10). This is misleading; nowhere does Dickinson disclose that encryption is used as the trust engine “generates the plurality of shares.” The passage on which Petitioner relies states in full: “For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions such as data encryption and decryption.” Ex. 1003 [Dickinson], 16:8-10. Contrary to Petitioner’s misleading contention, this is a reference to Dickinson’s disclosure that when a vendor requests a “cryptographic transaction” such as digital signature, the trust engine will “associat[e] a user . . . with one or more [encryption] keys”

stored on its server. *Id.*, 3:13-15, 5:28-30. Only after the trust engine completes its authentication method can the associated key be used to perform whichever “cryptographic function” the vendor requested. *Id.*, 3:13-15, 55:33-56:9. In context, Dickinson makes clear that the trust engine performs “cryptographic functions” in response to an authentication request, not in the process of generating shares. *See, e.g., id.*, 16:13-14 (“trust engine may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.”).

For these reasons, the Petition fails to show a reasonable likelihood that Dickenson discloses generating a plurality of shares by first performing a cryptographic operation on a data set, as required by independent claims 1, 7 and 14 of the '194 Patent.

2. Dickinson Does Not Disclose “Distributing The Data Set In The Plurality Of Shares Such That The Data Set Can Be Reconstructed Using Any Subset Of The Shares That Includes At Least A Minimum Number Less Than All Of The Shares.”

The independent claims of the '194 Patent each require “distributing the data set in a plurality of shares such that the data set can be reconstructed using any subset of the shares that includes at least a minimum number less than all of the shares.” Ex. 1001 [’194], cls. 1, 7, 14. The '194 Patent discloses “distributing the data set in the plurality of shares” by disclosing a “cryptographic split (cryptosplit)” that “partitions the data” into N shares based on “any size unit of data.” *Id.*, 52:40-43.

The cryptosplit process ensures that *the data set* is *distributed* among the plurality of shares. The '194 Patent provides an example of a cryptosplit distributing a 23-byte data set into four shares, with a data unit size of one byte:

Each byte would be distributed into one of the 4 shares.

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. 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. The distribution of the bytes of data into the four shares would occur by placing the first byte of the data into share number r1, byte two into share r2, byte three into share r3, through the 23rd byte of data into share r23.

Id., 52:61-53:7.

The '194 Patent extends this example to disclose how the cryptosplitting process can be used to distribute the data set into shares such that the data set can be reconstructed using *any subset of shares* that includes at least a minimum number less than all of the shares. The '194 Patent calls this an “M of N cryptosplit,” in which the data set can be reassembled from any subset M of the total N shares, where M is at least one less than N. *Id.*, 53:16-26. It discloses that, in one embodiment, each unit of data is stored in both a “primary” share and a “backup” share, providing sufficient redundancy to restore the data set if one share is missing. *Id.*, 53:28-39.

By extending the example discussed above, the '194 Patent discloses an “3 of 4” cryptosplit embodiment for distributing a 23-byte data set into four shares, using a data unit size of one byte, such that the data set can be reconstructed from any 3 of the 4 shares:

[A] set of random numbers (also referred to as primary share numbers) from 0 to 3 are generated equal to the number of data units. Then another set of random numbers is generated (also referred to as backup share numbers) from 1 to 3 equal to the number of data units. Each unit of data is then associated with a primary share number and a backup share number. . . . The primary share number is used to determine into which share the data unit is stored. The backup share number is combined with the primary share number to create a third share number [If] primary share number is between 0 and 3, and the backup share number is between 1 and 3 [that] ensures that the third share number [the backup share number] is different from the primary share number.

Id., 53:40-62. In this case, the result is that each of the four shares contains a portion of the data set and each share contains sufficient redundant data such that the data set can be reconstructed from any subset of three shares.

Petitioner relies entirely on Dickinson, and solely on the example shown in Dickenson’s Figure 8 (below), as disclosing this limitation. Pet., 40-42. But this example does not disclose “*distributing the data set* in the plurality of shares, such

that the data set can be reconstructed using any subset of shares that includes a minimum number less than all of the shares.”⁵

The Petition argues that in Dickinson’s Figure 8 example, two random numbers, “A” and “C”, are generated by the trust engine. Each of those numbers is then combined with the Sensitive Data (“S”) (*i.e.*, enrollment authentication data) using the XOR operation to create new numbers “B” and “D.” *See* Pet., 41. The A, B, C and D numbers are combined, in an unexplained way, to form the “pairings” AC, AD, BC and BD “such that any two [pairings] provide one of A and B, or, C and D,” sufficient to reconstruct S. Pet., 42 (quoting Ex. 1003 [Dickinson], 21:27-28).

⁵ Dickinson does not ever use the term “share” to refer to a piece or portion of a data set.

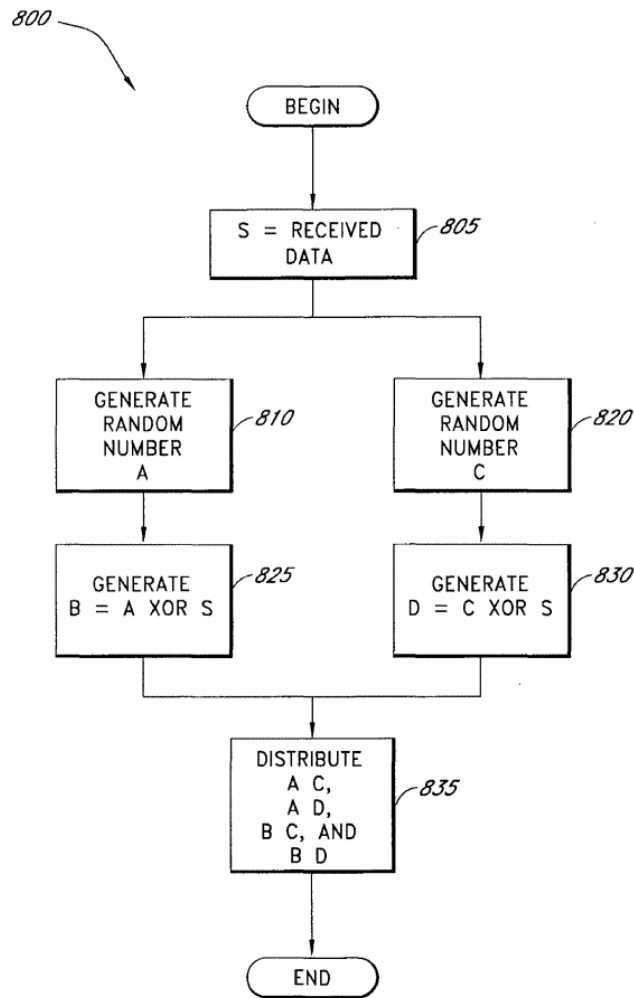


FIG. 8

Ex. 1003 [Dickinson], Fig. 8

While this example may relate to the generation and storage of different numbers (“pairings”) from which S may be recreated, it does not teach “*distributing*” the “*data set*” S into a plurality of shares in the manner disclosed and claimed in the ’194 Patent. For example, one of Dickinson’s Figure 8 pairings (AC) contains nothing from S; the AC pairing is simply the two “random number[s]” that were

generated by the trust engine. *See id.* The other pairings (AD, BC, BD)—as is evident from Figure 8 above—each include at least one copy of S *in its entirety* from B or D; in fact BD includes two copies. Ex. 2031 [Rubin-Decl.] ¶¶ 79-81.

Whatever may be said about the correspondence between the “pairings” disclosed in Dickinson, and the “shares” required by the ’194 Patent, it is clear that what is “distributed” in the pairings is not shares of *data set S*, as the pairings comprise either none of S or all of it.

For all these reasons, the Petition fails to show a reasonable likelihood that Dickinson discloses “generating . . . a plurality of shares” as required by the independent claims of the ’194 Patent.

C. A POSITA Would Not Be Motivated To Combine Dickinson And Hardjono To Achieve The Claimed Inventions, Including The Limitation Of “Sending The Data Set Responsive To The Request” (All Claims, All Grounds).

The ’194 Patent’s independent claims also require, following reconstruction of the data set, “sending the data set responsive to the request.” Ex. 1001 [’194], cls. 1[G], 7[B-6], 14[H]. Just as the claims require “receiving” the request from outside the Secure Data Parser, they require “sending” the data set that was reconstructed in response to the request outside the Secure Data Parser as well so that a user may access or use it.

For example, the '194 Patent discloses that “one aspect of the present invention is to provide a method for securing virtually any type of data from *unauthorized* access or use.” Ex. 1001 [’194], 2:33-35. The method also “comprises reconstituting or re-assembling the secured data into its original form for *authorized* access or use.” *Id.*, 2:46-48. Access to or use of the requested data by an authorized user requires sending the reconstituted data external to the Secure Data Parser. Likewise, Figure 32 discloses the operation of the secure data parser when it receives a “read” request to obtain data stored in the system. *Id.*, 70:64-71:14. In this example, the secure parser collects the shares, reconstitutes the data set, and sends it on for, *e.g.*, external “communicat[ion] o[ver] a network” to the requester. *Id.*, 70:63.

Petitioner does not suggest that Dickinson alone discloses this limitation. Pet., 49. Instead, the Petition relies upon “Dickinson in view of Hardjono” for the purported disclosure of limitations 1[G] and 7[B-6].⁶ But Petitioner’s argument fails

⁶ For limitation 14[H], Petitioner contends that “Dickinson discloses [14H]” but relies wholly upon its analysis in connection with limitation 7[B-6]. Pet., 72. In connection with limitation 7[B-6], however, Petitioner relies upon the combination with Hardjono. *Id.*, 63. No explanation is given for how Dickinson alone would satisfy the limitation requiring “send[ing] the data set responsive to the request.”

here, too. Neither Petitioner nor its expert offer any reason why a POSITA would be motivated to combine Dickinson and Hardjono to permit sending the highly sensitive data stored in Dickinson's trust engine, outside the system. Dickinson makes clear, the vendor has no need for this sensitive information, and it even explains that the enrollment authentication data should never leave the authentication engine. *See* Ex. 1003 [Dickinson], 28:30-31 (“Moreover, *the authentication result transmitted to the vendor does not include the sensitive data.*”). Indeed, modifying Dickinson to permit its data to be sent out of the trust engine contravenes one of the fundamental goals of Dickinson's invention—“to provide security to transactions in electronic commerce.” *Id.*, 1:7-8. Petitioner's combination offers no benefits and a significant, unnecessary, drawback that Dickinson expressly and unequivocally warns against. Accordingly, weighing the benefits against the drawbacks and given Dickinson's teachings, a POSITA would not be motivated to combine Dickinson and Hardjono in such a way.

In Dickinson, after a user initiates a transaction with a vendor, the vendor sends an authentication request to the trust engine, which ultimately determines whether the user is authenticated and reports the result back to the vendor or provides a cryptographic function such as for a digital signature. *Id.*, 28:20-24. This method makes perfect sense; a user attempts to initiate a transaction and the trust engine

authenticates the user before proceeding with verifying the transaction or providing the signature.

What Dickinson does not do, however, is “send[] the data set responsive to the request” as the claims require. Indeed, Dickinson expressly teaches that the authentication result returned to the vendor in response to the authentication requests does *not* include the sensitive data. *Id.*, 28:30-31 (“Moreover, *the authentication result transmitted to the vendor does not include the sensitive data*, and the user may not even know whether he or she produced valid authentication data.”).

In an attempt to bridge this gap, Petitioner argues that a POSITA would look to Hardjono—arguing that “[i]t would have been obvious to combine Dickinson with Hardjono so that Dickinson’s trust engine returns the assembled data set to the request, as taught by Hardjono.” Pet., 49. More specifically, Petitioner argues that in its proposed combination, “Dickinson’s trust engine assembles the stored segments and returns the reconstructed data ‘[u]pon subsequent receipt of a request for the data’ to the requester [(i.e., Dickinson’s vendor)], as taught by Hardjono.” *Id.*, 26, 45 (“the vendor system 120 ... forwards ... the authentication request to the trust engine” and “transaction engine 205 receives the [vendor request]”).⁷ In

⁷ To the extent that Petitioner argues that the Dickinson/Hardjono requester is the transaction engine, that argument also fails for the same reasons explained below.

other words, in Petitioner’s proposed combination, the vendor sends the trust engine an authentication request, the trust engine reconstructs the enrollment authentication data, and then the trust engine sends the enrollment authentication data to the vendor (*i.e.*, the requester).

As an initial matter, Petitioner’s argument about whether a POSITA would have been motivated to combine Dickinson and Hardjono in such a way is conclusory. Petitioner argues that a POSITA would have been motivated to add “Hardjono’s teaching of transmitting the assembled data in response to a request” to Dickinson because “Dickinson’s trust engine already reconstructs the data set upon receiving a retrieval request, and failing to forward that reconstructed data to the requester would leave the workflow incomplete and thereby use trust engine’s resources inefficiently.” Pet., 26-27. Petitioner’s only support for this argument is two wholly conclusory paragraphs in Dr. Zadok’s expert declaration. *See id.*; Ex. 1002 ¶¶ 203-04. Such conclusory testimony is insufficient to support Petitioner’s argument. *See Virtek Vision Int’l ULC v. Assembly Guidance Sys., Inc.*, 97 F.4th 882, 889 (Fed. Cir. 2024) (finding patent challenger failed to show obviousness where “conclusory [expert] testimony fails to address why or whether a skilled artisan would have been motivated to combine the camera disclosed in the ’094 Rueb with Keitler and Briggs”).

In any event, Petitioner’s argument is wrong. Dickinson’s workflow is not “incomplete,” and it does not “forward that reconstructed data to the requester” for good reason. As discussed above, the “request” from Dickinson’s vendor is simply a request to authenticate the user seeking to initiate a transaction with the vendor. *See* Ex. 1003 [Dickinson], 27:15-16 (“the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user.”). By comparing the user’s entered data with the enrollment authentication data, Dickinson’s trust engine verifies whether the user entered information matches that on file solely to verify the user. *Id.*, 28:20-24. The vendor gets precisely what it asked for, verification that the user is “who they say they are.”

As Dr. Rubin explains, far from being an “incomplete” workflow as Petitioner alleges, in Dickinson’s unmodified authentication process, the vendor gets precisely the information it needs and requested:

Dickinson’s workflow is not “incomplete.” Dickinson’s authentication request from the vendor is only asking the trust engine to authenticate user’s identity. Ex. 1003 [Dickinson], 37:24-29 (“authentication is the process of proving that a user is who he says he is. . . . The user must demonstrate to the trust engine 110 that he is who he says he is by either: knowing something that only the user should know (knowledge-based authentication), having something that only

the user should have (token-based authentication), or by being something that only the user should be (biometric-based authentication).”). When the authentication engine in Dickinson’s trust engine receives the portions of enrollment authentication data and then reconstructs the data set, the authentication engine “compares the enrollment authentication data to the current authentication data provided by the user.” *Id.*, 28:11-14.

Rather than returning the reconstructed enrollment authentication data as the ’194 Patent’s claims require, the trust engine only returns the result of the authentication comparison. *Id.*, 28:20-24 (“[T]he authentication engine 215 fills in the authentication request with the result of the comparison of STEP 1045. According to one embodiment of the invention, the authentication request is filled with a YES/NO or TRUE/FALSE result of the authentication process 1000. In STEP 1055 the filled-in authentication request is returned to the vendor for the vendor to act upon, for example, allowing the user to complete the transaction that initiated the authentication request.”). That “YES/NO” or “TRUE/FALSE” response is all the vendor needs to determine whether the user has been authenticated.

Ex. 2031 [Rubin-Decl.] ¶¶ 90-91.

Petitioner’s proposed modification, of Dickinson with Hardjono would send the “enrollment authentication data”—*e.g.*, username/password or biometric information (*see* Ex. 1003, 27:34-28:5)—directly to the vendor. *See* Pet., 26-27, 45 (admitting that its proposed combination has Dickinson’s vendor as the requester

sending an authentication request to the trust engine and, relying on Hardjono, the reconstructed authentication data is sent back to the requester). But Petitioner provides no colorable reason why Dickinson's system would benefit from sending the enrollment authentication data to the vendor as would be required by the claims and Petitioner proposes. As Dr. Rubin explains:

The vendor merely needs to confirm the user's identity in order to allow the user to perform a transaction such as "selecting a purchase option, requesting access to a restricted area or device of the vendor system 120, or the like." Ex. 1003 [Dickinson], 27:18-19. The vendor does not need any information other than that the user's current authentication data matches the enrollment authentication data, which is confirmed by the trust engine. *Id.*, 27:15-16 ("the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user."); 28:13-24 (authentication engine "compares the enrollment data to the current authentication data provided by the user" and "fills in the authentication request with the result of the comparison").

Indeed, the vendor never even receives the user's current authentication data and could not perform the comparison between the current and enrollment authentication data itself. *Id.*, 27:34-28:5 ("the user system 105 gathers the current authentication data, potentially including current biometric information, from the user. The user system 105 . . . transfers that data to the trust engine 110.").

Consequently, the vendor would not have any need for the reconstructed enrollment authentication data.

Ex. 2031 [Rubin-Decl.] ¶¶ 93-94. In other words, when the trust engine returns the “result of [that] comparison” to the vendor, (Ex. 1003 [Dickinson], 28:20-24), Dickinson’s workflow is complete because the vendor has received what it requested. Ex. 2031 [Rubin-Decl.] ¶ 95. There is no “incomplete” “workflow” in Dickinson that could be improved by sending the reconstructed data set. *Id.*

A POSITA would not have been motivated to combine Dickinson and Hardjono to reach the claimed invention because a POSITA would not be motivated to add a feature—sending the enrollment authentication data to the vendor—that provides no benefit. *See Virtek Vision Int’l ULC v. Assembly Guidance Sys., Inc.*, 97 F.4th 882, 887 (Fed. Cir. 2024) (“[T]here must exist a motivation to combine various prior art references in order for a skilled artisan to make the claimed invention.”); *Hulu LLC v. Sound View Innovations LLC*, IPR2018-00582, Paper 34, 20-21 (Aug. 5, 2019) (informative) (rejecting combination where not clear why it would be a “good idea” to modify); *In re Schweickert*, 676 F. App’x. 988, 995 (Fed. Cir. 2017) (remanding where unexplained how “Birrell would have ‘benefitted from the advantages of . . . Cunniff’s semaphore mechanism.’”) (non-precedential).

Even worse, Petitioner’s modification also unnecessarily discloses sensitive user information to the vendor that Dickinson makes clear should never be disclosed.

As Dr. Rubin explains, Dickinson teaches that the data set is secure because it never leaves the authentication engine within the trust engine:

Dickinson’s authentication process is designed to keep the data set secure by “only” allowing it to be “assembled ... inside the authentication engine”:

Based on the foregoing, the authentication process 1000 advantageously *keeps sensitive data secure* and produces results configured to maintain the integrity of the sensitive data. For example, *the sensitive data is assembled only inside the authentication engine 215*. For example, the enrollment authentication data is undecipherable until it is assembled in the authentication engine 215 by the data assembling module, and the current authentication data is undecipherable until it is unwrapped by the conventional SSL technology and the private key of the authentication engine 215. Moreover, *the authentication result transmitted to the vendor does not include the sensitive data*, and the user may not even know whether he or she produced valid authentication data.

Ex. 1003 [Dickinson], 28:25-31. Sending the reconstructed enrollment authentication data not only outside of the authentication engine, but outside the trust engine to the vendor, would compromise the data set’s security. Indeed, some of the largest data breaches in history have been due to authentication storage being breached. *See, e.g.,* Ex. 2034 [UpGuard] 1 (describing 26 “Biggest Data Breaches in US History”); 2

(“A team of Russian hackers targeted Yahoo’s database using backdoors, stolen backups, and access cookies to steal records from all user accounts”); 4-5 (“access to private information was allowed without needing verification or authentication procedures.”); 13 (“the entire Exactis database on a public network that was completely unsecured and accessible to everyone.”). Petitioner’s proposed combination thus would result in the significant drawback of failing to keep sensitive data secure.

Ex. 2031 [Rubin-Decl.] ¶ 97.

Accordingly, Petitioner’s argument fails as a matter of law. “The Board must weigh the benefits and drawbacks of the modification against each other, to determine whether there would be a motivation to combine.” *Arctic Cat Inc. v. Polaris Indus.*, 795 F. App’x. 827, 833 (Fed. Cir. 2019); *Henny Penny Corp. v. Frymaster LLC*, 938 F.3d 1324, 1331-32 (Fed. Cir. 2019) (similar); *Arctic Cat Inc. v. Bombardier Rec. Prods.*, 876 F.3d 1350, 1363 (Fed. Cir. 2017) (holding that “potential problems” with combination supported lack of motivation). Here, there are no benefits to sending the enrollment authentication data out of Dickinson’s authentication engine—only significant security drawbacks. *See* Ex. 2031 [Rubin-Decl.] ¶¶ 97-99 (explaining that risk of unauthorized access increases whenever data is sent outside of a system). Petitioner’s proffered modifications thus are precisely the sort of hindsight-driven analysis the Federal Circuit has rejected. *Ruiz v. A.B.*

Chance Co., 357 F.3d 1270, 1275 (Fed. Cir. 2004) (“This form of hindsight reasoning, using the invention as a roadmap to find its prior art components, would discount the value of combining various existing features or principles in a new way to achieve a new result—often the very definition of invention.”).

Thus, Petitioner fails to demonstrate that a POSITA would have been motivated to combine Dickinson and Hardjono such that Dickinson’s trust engine sends the reconstructed enrollment authentication data in response to an authentication request.

IV. CONCLUSION

Thus, Petitioner fails to demonstrate a reasonable likelihood of success as to any of the challenged claims, and the Petition should be denied.

Respectfully submitted,

/ Stephen J. Elliott /
Stephen J. Elliott (Reg. No. 52,858)
Andrei Iancu (Reg. No. 41,862)
SULLIVAN & CROMWELL LLP

Kenneth J. Weatherwax (Reg. No. 54,528)
Nathan Lowenstein, *pro hac vice*
Colette Woo, *pro hac vice*
LOWENSTEIN & WEATHERWAX LLP

Date: October 17, 2025

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITS

This Patent Owner Preliminary Response (the “POPR”) consists of 8,617 words, excluding table of contents, table of authorities, certificate of service, this certificate, or table of exhibits. The POPR complies with the type-volume limitation of 14,000 words as mandated in 37 C.F.R. § 42.24. In preparing this certificate, counsel has relied on the word count of the word-processing system used to prepare the paper (Microsoft Word).

Respectfully submitted,

/ Colette Woo /

Date: October 17, 2025

CERTIFICATE OF SERVICE

The undersigned hereby certifies that the following documents were served by electronic service, by agreement between the parties, on the date below:

PATENT OWNER'S PRELIMINARY RESPONSE

EXHIBITS 2031-2034

The names and addresses of the parties being served are as follows:

Taeg Sang Cho	tcho@desmaraisllp.com
Kurt Fredrickson	kfredrikson@desmaraisllp.com
Lindsey Miller	lmiller@desmaraisllp.com
Laura Avena	lavena@desmaraisllp.com
	IBM-SFI-IPR-Service@desmaraisllp.com

Respectfully submitted,

/ Colette Woo /

Date: October 17, 2025