2019-2192, -2258

UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

TECSEC, INC.

Plaintiff-Appellant,

v.

ADOBE INC.

Defendant-Cross Appellant,

and

SAS INSTITUTE, INC., SAP AMERICA, INC., SAP, AG, CISCO SYSTEMS, INC., SYBASE, INC., SOFTWARE AG, SOFTWARE AG, INC., PAYPAL, INC., ORACLE CORPORATION AND ORACLE AMERICA, INC.

Defendants.

Appeal from the United States District Court for the Eastern District of Virginia in 1:10-CV-115, Judge Liam O'Grady

---

CORRECTED OPENING BRIEF FOR PLAINTIFF-APPELLANT

Michael A. Oakes
Ozzie A. Farres
Steven L. Wood
HUNTON ANDREWS KURTH LLP
2200 Pennsylvania Avenue, N.W.
Washington, D.C. 20037
Telephone: (202) 955-1500
Facsimile: (202) 778-2201

October 7, 2019

Form 9. Certificate of Interest **Form 9**
Rev. 10/17

UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

TecSec, Inc. v. Adobe Inc.

No. 19-2192, -2258

CERTIFICATE OF INTEREST

Counsel for the Appellant TecSec, Inc. certifies the following (use "None" if applicable; use extra sheets if necessary):

1. Full name of party represented by me is:	2. Name of real party in interest (Please only include any real party in interest NOT identified in Question 3) represented by me is:	3. Parent corporations and publicly held companies that own 10% or more of stock in the party
TecSec, Inc.	TecSec, Inc.	None

4. The names of all law firms and the partners or associates that appeared for the party or amicus now represented by me in the trial court or agency or are expected to appear in this court **(and who have not or will not enter an appearance in this case)** are:

Law Firm	Attorney
Hunton Andrews Kurth LLP (formerly Hunton & Williams LLP)	Brian M. Buroker Thomas J. Cawley Kevin Gaunt Torsten Kracht Jeffrey S. Leaning Harry Manion Michael O'Shea Ryan P. Phair Brian Saunders Stephen M. Sayers Gregory N. Stillman Richard L. Wyatt, Jr. Rachael R. Yocum

Law Firm	Attorney
DiNovo Price Ellwanger & Hardy LLP	Andrew G. DiNovo Jay D. Ellwanger Raymond W. Mort III Adam G. Price

5. The title and number of any case known to counsel to be pending in this or any other court or agency that will directly affect or be directly affected by this court's decision in the pending appeal. *See* Fed. Cir. R. 47.4(a)(5) and 47.5(b). (The parties should attach continuation pages as necessary).

None

Date: October 7, 2019 /s/ Michael A. Oakes

TABLE OF CONTENTS

JURISDICTIONAL STATEMENT 6

STATEMENT OF THE ISSUES 7

STATEMENT OF THE CASE 8

I. FACTUAL BACKGROUND 8

1. TecSec And The DCOM Patents 8

B. The Prior Proceedings 10

C. The Accused Adobe Acrobat Products 12

D. The Adobe Proceedings 13

SUMMARY OF THE ARGUMENT 17

ARGUMENT 20

I. The District Court Erred In Converting a Motion in Limine into a Motion for Summary Judgment and Resolving a Dispositive, Factual Issue Regarding Adobe's Intent as a Matter of Law 21

A. Preventing TecSec From Presenting Evidence on its Induced Infringement Claim Was the Equivalent of a Grant of Summary Judgment 22

B. Whether Adobe Possessed the Requisite Intent to Induce Infringement from 2011-2013 Was A Fact Issue For the Jury. 28

II. The District Court's Improper Exclusion of Evidence Requires a New Trial on Inducement 32

A. The District Court Excluded TecSec's Evidence of Direct Infringement by a Third Party 33

B. The District Court Abused its Discretion in Excluding Mr. Johnson's Testimony 38

C. A New Trial on Inducement is Necessary to Remedy the District Court's Exclusion of Evidence 40

III. The District Court's Misleading Jury Instructions Require a New Trial on Inducement 41

1. Jury Instruction No. 25 Misled the Jury by Implying that the 2009 Blog is Irrelevant to Inducement 42

2. Jury Instruction No. 28 Misled the Jury into Finding No Inducement of any Asserted Claim 44

3. A New Trial is Necessary to Remedy the Prejudice to TecSec Caused by Jury Instructions Nos. 25 and 28 46

IV. The District Court Erred By Vacating The Jury's Damages Award For Direct Infringement And Awarding Tecsec No Damages 49

A. The Damages Statute Precludes Awarding Zero Damages When There is an Underlying Finding of Infringement.....50

B. The Jury's Damages Award is Supported by Substantial Evidence.....54

C. TecSec Did Not Waive Pursuit of Reasonable Royalty Damages for Direct Infringement 61

CONCLUSION.....64

TABLE OF AUTHORITIES

Cases

ACCO Brands, Inc. v. ABA Locks Mfr. Co.,
501 F.3d 1307 (Fed. Cir. 2007) 33

Advanced Cardiovascular Sys., Inc. v. Medtronic, Inc.,
265 F.3d 1294 (Fed. Cir. 2001) 20

Advanced Display Sys., Inc. v. Kent State Univ.,
212 F.3d 1272 (Fed. Cir. 2000) 21

Allen Organ Co. v. Kimball Int'l, Inc.,
839 F.2d 1556 (Fed. Cir. 1988) 30

Apple, Inc. v. Motorola, Inc.,
757 F.3d 1386 (Fed. Cir. 2014) 51, 52, 57

Bowers v. Nat'l Collegiate Athletic Ass'n,
563 F. Supp. 2d 508 (D.N.J. 2008) 22

Bradley v. Pittsburgh Bd. Of Educ.,
913 F.2d 1064 (3d Cir. 1990) 22, 23

Brobst v. Columbus Servs. Int'l,
761 F.2d 148 (3d Cir. 1985) 23

Brooktree Corp. v. Advanced Micro Devices, Inc.,
977 F.2d 1555 (Fed. Cir. 1992) 55

Buckley v. Mukasey,
538 F.3d 306 (4th Cir. 2008) 38

Campbell v. Boston Sci. Corp.,
882 F.3d 70 (4th Cir. 2018) 20

Del Mar Avionics, Inc. v. Quinton Instrument Co.,
836 F.2d 1320 (Fed. Cir. 1987) 50, 52

DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.,
567 F.3d 1314 (Fed. Cir. 2009) 50

Dow Chem. Co. v. Mee Indus., Inc.,
341 F.3d 1370 (Fed. Cir. 2003) 49, 51, 57

Ecolab Inc. v. Paraclipse, Inc.,
285 F.3d 1362 (Fed. Cir. 2002) 21, 48

Elbit Sys. Land & C4I Ltd. v. Hughes Network Sys., LLC,
No. 2:15-CV-00037-RWS-RSP, 2017 WL 2651618
(E.D. Tex. June 20, 2017) 54

Global-Tech Appliances, Inc. v. SEB SA,
563 U.S. 754 (2011) 33

Graco Children's Prods. Inc. v. Kids II, Inc.,
No. 1:13-CV-1183-TWT, 2018 WL 1509192 (N.D. Ga. Mar. 26, 2018) 53

Gustafson, Inc. v. Intersystems Indus., Prods., Inc.,
897 F.2d 508 (Fed. Cir. 1990) 30

Hana Fin., Inc., v. Hana Bank,
735 F.3d 1158 (9th Cir. 2013) 23

Hilton Davis Chem. Co. v. Warner-Jenkinson Co.,
62 F.3d 1512 (Fed. Cir. 1995) 21

Holbrook v. Lykes Bros. Steamship Co.,
80 F.3d 777 (3d Cir. 1996) 28

Johnson v. Hugo's Skateway,
974 F.2d 1408 (4th Cir. 1992) 49, 54, 60

Johnson v. Parrish,
827 F.2d 988 (4th Cir. 1987) 55

K-2 Corp. v. Salomon S.A.,
191 F.3d 1356 (Fed. Cir. 1999) 20

Keg Techs., Inc. v. Laimer,
436 F. Supp. 2d 1364 (N.D. Ga. 2006) 62

Lindemann Maschinenfabrik GmbH v. Am. Hoist & Derrick Co.,
895 F.2d 1403 (Fed. Cir. 1990) 49

Louzon v. Ford Motor Co.,
718 F.3d 556 (6th Cir. 2013) 22

Malone v. Reliastar Life Ins. Co.,
558 F.3d 683 (7th Cir. 2009) (vacating 48

Meyer Intellectual Props., Ltd. v. Bodum, Inc.,
690 F.3d 1354 (Fed. Cir. 2012) 23, 27, 28, 32

Mid-America Tablewares, Inc. v. Mogi Trading Co.,
100 F.3d 1353 (7th Cir. 1996) 23

Mirror Worlds, LLC v. Apple, Inc.,
784 F. Supp. 2d 703 (E.D. Tex. 2011) 53

Molins PLC v. Quigg,
837 F.2d 1064 (Fed. Cir. 1988) 20

Monsanto Co. v. Ralph,
382 F.3d 1374 (Fed. Cir. 2004) 55

Power Integrations, Inc. v. Fairchild Semiconductor Int'l, Inc.,
843 F.3d 1315 (Fed. Cir. 2016) 33

Promega Corp. v. Life Techs. Corp.,
875 F.3d 651 (Fed. Cir. 2017) 49, 61, 62

RZS Holdings AVV v. PDVSA Petroleo S.A.,
506 F.3d 350 (4th Cir. 2007) 38

SmithKline Diagnostics, Inc. v. Helena Labs. Corp.,
926 F.2d 1161 (Fed. Cir. 1991) 57

SSL Servs., LLC v. Citrix Sys., Inc.,
769 F.3d 1073 (Fed. Cir. 2014) 38

TecSec, Inc. v. Adobe Sys. Inc.,
658 F. App'x 570 (Fed. Cir. 2016) 5, 14

TecSec, Inc. v. Int'l Bus. Mach. Corp.,
466 F. App'x 882 (Fed. Cir. 2012) 5

TecSec, Inc. v. Int'l Bus. Mach. Corp.,
731 F.3d 1336 (Fed. Cir. 2013) 5

TecSec, Inc. v. Int'l Bus. Mach. Corp.,
769 F. Supp. 2d 997 (E.D. Va. 2011) 11, 13

Tegal Corp. v. Tokyo Electron Co.,
248 F.3d 1376 (Fed. Cir. 2001) 45, 46

Toshiba Corp. v. Imation Corp.,
990 F. Supp. 2d 882 (W.D. Wisc. 2013) 46

U.S. ex rel. Drakeford v. Tuomey,
792 F.3d 364 (4th Cir. 2015) 38, 41

United States v. Gibson,
No. 2:17cr126, 2018 WL 4903261 (E.D. Va. Oct. 10, 2018) 22

United States v. Verges,
No. 1:13cr222 (JCC), 2014 WL 559573, (E.D. Va. Feb. 12, 2014) 28, 29

Viam Corp. v. Iowa Exp.-Imp. Trading Co.,
84 F.3d 424 (Fed. Cir. 1996) 20

Waterton Polymer Prods. USA, LLC v. Edizone, LLC,
No. 2:12-CV-17 TS, 2014 WL 12600697 (D. Utah Nov. 25, 2014) 53

WCM Indus., Inc. v. IPS Corp.,
721 F. App'x 959 (Fed. Cir. 2018) 29, 30

Williams v. Johnson,
747 F.Supp.2d 10 (D.D.C. 2010) 24, 32, 36

Winner Int'l Royalty Corp. v. Wang,
202 F.3d 1340 (Fed. Cir. 2000) 20

Statutes

28 U.S.C. § 1295(a)(1) 6

28 U.S.C. §§ 1331 and 1338(a) 6

35 U.S.C. §271(b) (2012) 33

35 U.S.C. § 284 ..... 49, 50, 51, 52, 54, 57

Rules

Fed. Cir. R. 47.4(a)(5) and 47.5(b) 3

Fed. R. Civ. P. 61 38

Fed. R. Civ. P. 62 39

Federal Circuit Rule 32(b) 3

Federal Rule of Appellate Procedure 32(a)(5) 3

Federal Rule of Appellate Procedure 32(a)(6) 3

Federal Rule of Appellate Procedure 32(a)(7)(B) 3

Federal Rule of Appellate Procedure 32(a)(7)(B)(iii) 3

Federal Rules of Civil Procedure 58 14

Other Authorities

75 Am. Jur.2d Trial § 44 (2017) 25

TABLE OF ABBREVIATIONS

Parties

TecSec	TecSec, Incorporated
Adobe	Adobe Inc.

Defined Terms

(___:____)	Column and line number in patent or page and lines in deposition transcript
‘702 Patent	U.S. Patent No. 5,369,702 to Shanton entitled “Distributed Cryptographic Object Method”
‘452 Patent	U.S. Patent No. 5,680,452 to Shanton entitled “Distributed Cryptographic Object Method”
‘755 Patent	U.S. Patent No. 5,717,755 to Shanton entitled “Distributed Cryptographic Object Method”
‘781 Patent	U.S. Patent No. 5,898,781 to Shanton entitled “Distributed Cryptographic Object Method”
DCOM	Distributed Cryptographic Object Method
DCOM Patents or Asserted Patents	‘702 Patent, ‘452 Patent, ‘755 Patent and ‘781 Patent collectively
Asserted System Claims	‘702 Patent claim 8 and ‘781 Patent claim 14
Asserted Method Claims	‘702 Patent claim 1, ‘452 Patent claim 1, ‘755 Patent claim 1, and ‘781 Patent claim 1
Asserted Claims	Asserted System Claims and Asserted Method Claims

Defined Terms cont.

Accused Product(s)	Adobe Acrobat Standard and Pro Versions, Versions 5 and up, as well as other Adobe products that include or are bundled with Acrobat
court *or* district court	United States District Court for the Eastern District of Virginia, Honorable Liam O'Grady presiding

INTRODUCTION

TecSec has already successfully appealed twice, the first time from an erroneous claim construction decision during the 2011 proceedings against co-defendant IBM, and the second time, from a grant of summary judgment of no infringement following a pre-discovery summary judgment motion filed by Adobe. On remand following fact discovery, Adobe was permitted to file a second summary judgment motion, and in it raised new arguments seeking a finding of no induced infringement, but TecSec was able to defeat the motion. Nearly ten years after filing its original complaint in 2010, TecSec was finally on the verge of having its infringement claims against Adobe presented to a jury. Just before trial, Adobe stipulated that it had, in fact, infringed every asserted claim of the TecSec patents.

But Adobe was not done trying to avoid liability. Although it had failed to raise the issue properly in either of its summary judgment motions, Adobe filed an omnibus motion—asking the district court, in part, to dispose of TecSec's induced infringement claims for the period beginning with the court's erroneous and later-reversed claim construction decision in 2011 and continuing until the expiration of the patents in 2013. Adobe had not identified any evidence and had not designated a single witness to show that it had relied upon the erroneous 2011 claim construction decision to form a good faith belief of no infringement, but argued that as a matter of law it must have lacked the necessary intent to induce infringement.

TecSec opposed, arguing that intent was a fact issue for the jury and that this Court's precedent precluded the district court from resolving dispositive, disputed factual issues through *limine* motions. The district court sided with Adobe, and in doing so abused its discretion by granting disposing of TecSec's induced infringement claims. This decision must be reversed.

The district court's improper *limine* ruling had a profound ripple effect on the trial. Adobe and the district court used the creation of the artificially-narrowed one-year infringement window as the justification for excluding relevant and probative evidence of Adobe's inducement and of third party direct infringement. For example, when TecSec sought to present compelling direct testimony from a third party witness deposition regarding his installation and use of the accused products in infringing ways, the district court excluded any mention of the evidence because one of the acts of infringement–but not all of the acts of infringement–occurred in April 2011, a month after the district court's claim construction decision. This was an abuse of discretion because the witness admitted to acts that constituted direct infringement prior to 2011, and, in any event, the witness's acts in April 2011 should have been admitted 1) because they were circumstantial evidence of Adobe's induced infringement and third party direct infringement, and 2) because the narrowed damages window was erroneous, as set forth above.

The district court's jury instructions further compounded these errors. Over TecSec's objections, the district court recharacterized Adobe's stipulation of infringement and instructed the jury that "the parties agree" that the one time Adobe admitted to infringing was prior to 2010, outside of the narrowed post-filing damages window. TecSec never agreed to that characterization, and it is contradicted by the parties' stipulation of undisputed facts. In connection with this improper characterization, the district court's instructions to the jury incorrectly suggested to the jury that Adobe's decision to post-infringing instructions on its website and then continue to post them after being put on notice of infringement amounted to mere inactivity, rather than active inducement. At closing, Adobe's counsel highlighted the mischaracterization of the stipulation and suggested to the jury that no indirect infringement could be found because Adobe's infringing conduct occurred outside of the district court's artificial damages window, and because Adobe's failure to take down the website post was insufficient to find inducement as a matter of law. As a result of these errors, the jury found no induced infringement. These errors materially and impermissibly affected the jury verdict, and TecSec should be entitled to a new trial on indirect infringement.

In spite of these errors, the jury still returned a verdict against Adobe, finding all of the patents valid and directly infringed by Adobe, and awarded TecSec $1.75 million in damages. The damage awarded fell squarely between the two competing

damages amounts proposed by the parties' experts. Nevertheless, in response to Adobe's renewed judgment as a matter of law, the district court vacated the damages award, finding that the amount was not supported by substantial evidence, and held that TecSec was entitled to zero damages for its proven infringement. This decision was in error, and the damages should be reinstated.

STATEMENT OF RELATED CASES

There have been three prior appeals in the same civil action before this court:

1) *TecSec, Inc. v. Int'l Bus. Mach. Corp.*, 466 F. App'x 882 (Fed. Cir. 2012) (Judges Linn, Dyk, and Prost).

2) *TecSec, Inc. v. Int'l Bus. Mach. Corp.*, 731 F.3d 1336 (Fed. Cir. 2013) (Judges Moore, Linn, and Reyna).

3) *TecSec, Inc. v. Adobe Sys. Inc.*, 658 F. App'x 570 (Fed. Cir. 2016) (Judges Prost, Linn and Taranto)

TecSec is not aware of any other case that would be affected by this appeal.

JURISDICTIONAL STATEMENT

This is an appeal from the United States District Court for the Eastern District of Virginia's Order entering judgment on the jury verdict as amended by the Court (Appx1, Appx2) and the Order granting Adobe's Renewed Motion for Judgment as a Matter of Law, or, in the Alternative, for a Remittitur (Appx52-60), as well as the district court's Orders on the Parties' Motions *in Limine* (Appx15-36, Appx37), and the district court's ruling on the Notice of Unresolved Deposition Designations (Appx61-83).

The district court had jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a). TecSec filed a timely notice of appeal and this Court has appellate jurisdiction pursuant to 28 U.S.C. § 1295(a)(1).

STATEMENT OF THE ISSUES

1. Whether the district court erred in deciding through a limine motion that Adobe lacked the specific intent to induce infringement based on an erroneous claim construction decision that was later reversed, when there was no evidence presented that Adobe believed that the claim construction ruling negated Adobe's intent to induce infringement.

2. Whether the district court compounded the error from its improper limine ruling by excluding evidence related to Adobe's inducement and third-party infringement based on the newly-created artificial damages period created by the district court.

3. Whether TecSec should be awarded a new trial on induced infringement based upon flawed instructions given to the jury and improper arguments made by Adobe's counsel in closing.

4. Whether the district court erred in vacating the jury's damages award for Adobe's infringement and awarding TecSec zero damages when Adobe stipulated to infringement and the jury found that Adobe directly infringed every asserted claim.

STATEMENT OF THE CASE

I. FACTUAL BACKGROUND

1. TecSec And The DCOM Patents

TecSec is a Virginia corporation founded in 1990 by Ed Scheidt after he served twenty-six years with the CIA, most recently as the Director of the CIA's Cryptographic Center. For the past twenty-nine years, TecSec has designed products that secure data using a combination of access control and encryption. TecSec owns more than 30 data security patents, and its customers and licensees have included Microsoft, Boeing, Lockheed, Martin Marietta, Motorola, Sterling Commerce, the IRS, and the United States Postal Service.

TecSec's early efforts to provide security solutions focused on using access control and encryption to protect entire messages or entire files. Realizing the limitations of file-level only security, Greg Shanton, a TecSec software developer, conceived of a novel solution to increase security and flexibility—the Distributed Cryptographic Object Method ("DCOM")—which provides for security at the ***object*** level. TecSec filed a patent application on this invention on October 18, 1993 that issued as the '702 Patent in 1994, the first of the four DCOM Patents asserted in the present case. The DCOM Patents disclose systems and methods for protecting sensitive information so that any digital object, such as portions of a document, could be labeled and then protected by a combination of access control and encryption.

When an object is labeled (e.g., "Top Secret"), the DCOM system can limit access to users with permission to view objects with certain labels (e.g., User A has rights to see "Top Secret" objects). Access control through labeling, therefore, provides one level of security on an object. To obtain access to an encrypted object, a user must enter the appropriate identifier, such as a password or security certificate, which is used to identify the user as one who indeed has permission to access the object that was labeled. If the user has permission, access to the object is obtained; if the user does not have rights to certain objects, the DCOM system precludes access. Once access has been obtained, if the object has been encrypted, the user must then have the appropriate key to enable decryption to actually see the object. Thus, encryption provides the second level of security creating a system of multilevel security. The DCOM patents allow for multilevel encryption of objects, or "nesting," through applying two layers of encryption to an object. Thus, using DCOM, objects (e.g., portions of a file), as opposed to entire files, are encrypted and embedded or "nested" into other objects (called container objects), which could be files, tables, or any other type of object. This allows for the encryption (and later decryption) of only the required sensitive information, rather than the entire file.

This technology has significant advantages for various types of documents, such as PDF files that are widely-used today. PDF files consist of multiple objects and can be viewed and manipulated using software, such as Adobe Acrobat, installed

on hardware. PDF files include documents, as well as container PDF files—portfolios and envelopes, into which other PDF files or objects can be embedded. These embedded files can be individually encrypted prior to embedding and have access control in the form of passwords or certificates applied. Likewise, the container file itself can be encrypted and have access control applied. Thus, multilevel security can be achieved. Once the container file is decrypted and accessed, the individual embedded files or objects remain protected until accessed. Through multilevel security, the DCOM system is able to locate, determine access permissions, and decrypt specific sensitive objects that are requested at a given time. This provides significant security advantages.

TecSec obtained three additional DCOM patents—the '452, '755 and '781 Patents—with additional claims directed at security systems and methods that employ a combination of both access control and encryption to protect digital data.

B. The Prior Proceedings

TecSec sued thirteen defendants in its original complaint. The district court severed TecSec's claims against International Business Machines Corp. ("IBM") from TecSec's Claims against Adobe Systems, Inc., Cisco Systems, Inc., Oracle Corporation, PayPal, Inc., SAS Institute, Inc., Software AG, Inc., Software AG, SAP America, Inc., SAP AG, Oracle America, Inc. (f/k/a Sun Microsystems), and Sybase,

Inc. (hereinafter, the "Defendants").[1] TecSec's claims against IBM then proceeded to dispositive motion practice while its claims against the Defendants were stayed. In 2011, the district court construed certain claim terms and granted summary judgment of non-infringement in IBM's favor on multiple grounds. *TecSec, Inc. v. Int'l Bus. Mach. Corp.*, 769 F. Supp. 2d 997 (E.D. Va. 2011). TecSec filed an appeal. This Court summarily affirmed the district court's judgment of no infringement under Rule 36.

Upon remand in 2012, TecSec stipulated, subject to its right to appeal the district court's prior claim construction ruling, that it could not demonstrate infringement by the Defendants for the DCOM Patents under the district court's 2011 claim constructions made in the IBM case.[2] Based on the stipulation, the district court entered judgment of no infringement in favor of the Defendants with respect to the DCOM Patents. On appeal, this Court reversed the district court on 17 of 18 claim construction rulings and remanded the case to the district court. Following remand, the defendants filed a petition for a writ of certiorari, which was denied and the mandate was returned to the district court. The district court then ordered TecSec to proceed against Adobe only.

[1] TecSec's claims against eBay, Inc. were dismissed without prejudice.

[2] TecSec entered into a settlement with IBM and no further issues are pending between IBM and TecSec at this time.

C. The Accused Adobe Acrobat Products

TecSec accused Adobe of infringing through the use, installation and sale of the well-known Adobe Acrobat Standard and Pro products, Versions 5 and higher, which can be used to create, protect and read PDF files. Appx12270, 34:12-25 (Preliminary Jury Instructions); Appx12725, 490:13-19 (Jones Direct). PDF files or documents are multimedia files that consist of multiple objects and can be viewed and manipulated using software, such as Adobe Acrobat, installed on hardware. Appx25771, Fig. 1; Appx12740-12741, 505:14-19, 506:9-17 (Jones Direct). Using Acrobat, a user can choose to encrypt the entire PDF document, the document but not the metadata, or only the document's attachments. Appx12730, 495:10-16 (Jones Direct); Appx24739. Acrobat also applies access controls, such as passwords, to restrict access to these encrypted objects. Appx12730, 495:22-25 (Jones Direct); Appx24737. Each option provides for an access control mechanism, an encryption process, and the ability to embed encrypted objects in other objects that are also encrypted. Appx12733-12734, 498:24-499:7 (Jones Direct); Appx24459-24460.

Acrobat provides functionality for the embedding of PDF documents or other objects into other PDF documents that serve as container objects —known as PDF portfolios and PDF envelopes. Appx12727, 492:4-24 (Jones Direct); Appx26703, 85:21-86:12 (Landwehr Dep.); Appx26668, 51:23-52:11, 52:19-22, 52:25-53:03

(Borstein Dep.). Any number of encrypted PDF documents can be embedded within other PDF documents. *See, e.g.*, Appx26693-26696 (Borstein Dep.). Likewise, the container file itself can be encrypted and have access control applied to it. *Id.* Thus, Acrobat can create a web of encrypted and password protected PDF documents within other encrypted and password protected PDF documents. *Id*. Beginning in 2009, Adobe posted instructions on its website teaching its customers how to perform multi-level encryption. Appx24459-24460. Adobe continued to post those instructions after the filing of TecSec's complaint, and those instructions remained available to Adobe's customers at all times relevant to these proceedings. Appx13029, 794:15-18 (Landwehr Direct); Appx13725-13785 (Summary of Blog Posting Access). At least a thousand Adobe's customers viewed the post containing Adobe's instructions during that period. *See* Appx13725-13785 (Summary of Blog Posting Access).

D. The Adobe Proceedings

Following remand, and over TecSec's objection, the district court permitted Adobe to file a pre-discovery motion for summary judgment of no infringement. After Adobe filed its motion, the Court set an abbreviated and accelerated schedule for TecSec to take discovery necessary to oppose Adobe's motion. At the hearing on Adobe's motion, the district court, without prior notice, suggested another possible non-infringement argument—an issue not raised or briefed by Adobe.

Adobe and TecSec declined the district court's invitation for further briefing on that ground, but the district court later issued an opinion granting summary judgment of no infringement to Adobe, based on the issue raised *sua sponte* by the district court at the hearing. The district court entered judgment in favor of Adobe under Federal Rules of Civil Procedure 58 and certified the judgment as final under Rule 54(b).

TecSec again appealed, and this Court again reversed the district court, finding that factual issues precluded a grant of summary judgment. *TecSec, Inc. v. Adobe Sys. Incorporated*, 658 F. App'x 570 (Fed. Cir. 2016). TecSec also asked this Court to reassign the case on remand, which this Court declined to do. However, following remand, the district court, *sua sponte*, reassigned the case to Judge Liam O'Grady.

The case proceeded through discovery, claim construction, and summary judgment. Adobe moved for summary judgment on nine separate grounds, including induced infringement, but the district court denied the motion on all nine grounds, finding that factual disputes precluded a grant of summary judgment. Appx3-13; Appx14. Shortly before trial was set to begin, Adobe stipulated that every version of the accused Acrobat products did, in fact, infringe every asserted claim of the DCOM patents. Appx11993-11997.

Although the district court had already denied Adobe's motion for summary judgment of no induced infringement, Adobe filed an omnibus motion *in limine*

asking the district court, in part, to prevent TecSec from arguing that Adobe had the requisite intent to induce infringement from the date of the district court's erroneous 2011 claim construction decision from the IBM phase of the case until the expiration of the DCOM patents in 2013. Appx10429-10432. TecSec opposed, but the district court granted the motion, disposing of TecSec's induced infringement claims for the period from 2011-2013. Appx15-36. TecSec filed a motion for reconsideration, arguing that the request was an improper and untimely summary judgment motion, but the district court indicated that it was unwilling to have a trial on Adobe's state of mind during the relevant infringement period. Appx12213-12214, 12:21-13:1.

The case proceeded to a jury trial on the remaining direct infringement claims from 2010-2013, the indirect infringement claims only for the period from 2010-2011, and Adobe's invalidity counterclaims. Based on its grant of Adobe's Motion *in Limine*, the district court precluded TecSec from presenting any evidence of any acts of infringement by Adobe or third parties outside of the 2010-2011 period. Appx26-29. Over TecSec's objection, the district court recharacterized Adobe's stipulation of infringement and instructed the jury that the stipulation only pertained to a period of time prior to 2010, outside of the damages window. Appx12076. At closing, Adobe's counsel highlighted the mischaracterization of the stipulation and suggested to the jury that no indirect infringement should be found because the infringing conduct occurred outside of the district court's artificial damages window.

Appx13540-13541, 1305:24-1036:2. The jury returned a verdict finding that all of the asserted claims were valid and infringed directly by Adobe, finding no indirect infringement, and awarding TecSec damages in the amount of $1.75 million. Appx38-44.

Adobe filed a post-trial motion for judgment as a matter of law on the jury's adverse findings. Appx12107-12109. The district court denied Adobe's motion with respect to direct infringement and validity, but vacated the damages award, and found that TecSec was entitled to zero damages, despite upholding the jury's infringement verdict. Appx52-60.

TecSec filed this timely appeal.

SUMMARY OF THE ARGUMENT

I. After correctly denying Adobe's motion for summary judgment of no induced infringement, the district court committed clear error by disposing of TecSec's induced infringement claims for the 2011-2013 period in response to Adobe's Motion *in Limine* #1. The district court's ruling was procedurally improper because it amounted to a grant of summary judgment, without affording TecSec the protections guaranteed by the Federal Rules of Civil Procedure.

Moreover, the district court decided as a matter of law that Adobe lacked the requisite intent to induce third parties to infringe during the 2011-2013 period. However, this Court has held consistently that intent is a factual issue that must be resolved by the trier of fact, and disputed factual issues regarding Adobe's state of mind prevented the district court from taking this issue away from the jury.

II. Based on the improper ruling regarding indirect infringement discussed in Section I above, the district court created an artificial infringement and damages window limited to just a one year period, and then precluded TecSec from presenting relevant evidence regarding third party infringement of the system and method claims, purportedly because the evidence related only to infringement outside of the newly-created window. For the reasons set forth above, this evidence should have been admitted because the district court's decisions limiting the induced infringement period was in error. Moreover, the excluded evidence had probative

value even if the limited damages window were proper, and should have been permitted as circumstantial evidence of Adobe's inducement of customers and of third party infringement of the DCOM patents.

III. Building upon the errors set forth in Sections I and II above, the district court provided the jury with flawed jury instructions regarding induced infringement, which materially prejudiced TecSec and caused the jury to return a verdict of no induced infringement, when the undisputed evidence should have led to a finding that Adobe induced customers to infringe. Specifically, TecSec demonstrated that the Adobe blog post instructing customers on how to use Acrobat in an infringing manner was posted in 2009, that Adobe continued to post the instructions after being notified of its infringement and throughout the entire period until the patents expired, and that customers continued to view the instructions during that period. Nevertheless, the district court's instructions suggested that the Adobe blog post amounted to "inactivity" that was insufficient to give rise to induced infringement. Adobe's counsel seized upon this misstatement and made the improper argument in closing that because the blog was *originally* posted before the complaint was filed, the jury could not find induced infringement.

IV. Despite the district court's multiple errors with respect to indirect infringement, the jury still found that Adobe directly infringed every claim of the asserted patents, and awarded TecSec $1.75 million in damages for Adobe's

infringement. The amount was squarely between the two competing damages amounts proposed by the parties' experts, and was never shown by Adobe to be excessive. Notwithstanding the jury's verdict, the district court vacated the damages award and held that TecSec was entitled to zero damages for Adobe's (admitted) infringement. The district court erred in vacating the damages award and the award should be reinstated.

ARGUMENT

This Court applies the law of the regional "circuit to which district court appeals normally lie, unless the issue pertains to or is unique to patent law," *Molins PLC v. Quigg*, 837 F.2d 1064, 1066 (Fed. Cir. 1988) (citations omitted), in which case, it applies its own law to both substantive and procedural issues "intimately involved in the substance of enforcement of the patent right." *Viam Corp. v. Iowa Exp.-Imp. Trading Co.*, 84 F.3d 424, 428 (Fed. Cir. 1996).

Because "[e]videntiary rulings . . . are not unique to our jurisdiction, . . . we review them under the law of the [regional circuit]." *Advanced Cardiovascular Sys., Inc. v. Medtronic, Inc.*, 265 F.3d 1294, 1308 (Fed. Cir. 2001). The Fourth Circuit reviews such evidentiary rulings for abuse of discretion. *Campbell v. Boston Sci. Corp.*, 882 F.3d 70, 77 (4th Cir. 2018). However, where a district court rules, as a matter of patent law, that a party is precluded from introducing evidence, the Court will apply Federal Circuit law and review the district court's ruling de novo. *K-2 Corp. v. Salomon S.A.*, 191 F.3d 1356, 1367 (Fed. Cir. 1999); *Winner Int'l Royalty Corp. v. Wang*, 202 F.3d 1340, 1345 (Fed. Cir. 2000). Here, because the district court ruled as a matter of patent law that a party cannot have the requisite intent to induce infringement where an erroneous and later-reversed claim construction order is in place, this decision should be reviewed de novo.

The question of whether a jury instruction on an issue of patent law is erroneous is a matter of Federal Circuit law and is reviewed de novo. *Advanced Display Sys., Inc. v. Kent State Univ.*, 212 F.3d 1272, 1282 (Fed. Cir. 2000). A jury verdict will be set aside, based on erroneous jury instructions, if the movant can establish that "those instructions were legally erroneous," and that "the errors had prejudicial effect." *Id*. at 1281; *see also Ecolab Inc. v. Paraclipse, Inc*., 285 F.3d 1362, 1373 (Fed. Cir. 2002). In reviewing jury instructions, the full trial record and the jury instructions in their entirety must be examined because "instructions take on meaning from the context of what happened at trial, including how the parties tried the case and their arguments to the jury." *Hilton Davis Chem. Co. v. Warner-Jenkinson Co.*, 62 F.3d 1512, 1522 (Fed. Cir. 1995), rev'd on other grounds, 520 U.S. 17 (1997).

I. The District Court Erred In Converting a Motion *in Limine* into a Motion for Summary Judgment and Resolving a Dispositive, Factual Issue Regarding Adobe's Intent as a Matter of Law

Through a *limine* motion, the district court decided as a matter of law that Adobe lacked the specific intent to induce infringement during the period from 2011-2013, even though it had previously denied Adobe's motion for summary judgment on induced infringement. By resolving a substantive, factual dispute through a motion *in limine*, the district court committed an abuse of discretion. TecSec should

have been permitted to present this disputed, factual issue to the jury, and the district court's decision should be reversed.

A. Preventing TecSec From Presenting Evidence on its Induced Infringement Claim Was the Equivalent of a Grant of Summary Judgment

A motion *in limine* is designed to permit the trial court "to rule in advance of trial on the relevance of certain forecasted evidence, as to issues that are definitely set for trial, without lengthy argument at, or interruption of, the trial." *United States v. Gibson*, No. 2:17cr126, 2018 WL 4903261, at *2 (E.D. Va. Oct. 10, 2018) (quotations omitted). The purpose of a motion *in limine* is to bar "irrelevant, inadmissible, and prejudicial" issues from being introduced at trial. Black's Law Dictionary 1013 (6th ed. 1990); *see also Bradley v. Pittsburgh Bd. Of Educ*., 913 F.2d 1064, 1069 (3d Cir. 1990) ("a motion in limine is designed to narrow the evidentiary issues for trial and to eliminate unnecessary trial interruptions.").

It is well settled that motions *in limine* "are inappropriate devices for resolving substantive issues, such as the sufficiency of the evidence to support a claim or defense." *Bowers v. Nat'l Collegiate Athletic Ass'n*, 563 F. Supp. 2d 508, 532 (D.N.J. 2008) (internal quotation marks and citation omitted). Courts of appeal, including this one, have held consistently that substantive issues, especially factual inquiries reserved for the trier of fact, cannot be resolved through a motion *in limine*. *See, e.g.*, *Louzon v. Ford Motor Co.,* 718 F.3d 556, 566 (6th Cir. 2013) (reversing

grant of motion *in limine* on non-evidentiary issues and remanding); *Hana Fin., Inc., v. Hana Bank*, 735 F.3d 1158, 1162 n.4 (9th Cir. 2013) ("motion in limine is not the proper vehicle for seeking a dispositive ruling on a claim, particularly after the deadline for filing such motions has passed"); *Meyer Intellectual Props., Ltd. v. Bodum, Inc.*, 690 F.3d 1354, 1378 (Fed. Cir. 2012) (same); *Brobst v. Columbus Servs. Int'l*, 761 F.2d 148, 154 (3d Cir. 1985) (vacating decision that amounted to a grant of summary judgment pursuant to a *limine* motion); *Mid-America Tablewares, Inc. v. Mogi Trading Co.*, 100 F.3d 1353, 1363 (7th Cir. 1996) (argument that plaintiff could not prove its lost profits with reasonable certainty goes to the sufficiency of the evidence, and while "this might be a proper argument for summary judgment or for judgment as a matter of law, it is not a proper basis for a motion to exclude evidence prior to trial."). The reason is that deciding substantive issues during the *limine* phase deprives the non-moving party of the protections afforded by Federal Rule's summary judgment process. *See, e.g.*, *Brobst*, 761 F.2d at 154 ("In effect, the district court's procedure converted the in limine motion into one for summary judgment, but without the procedural protections of notice which the federal rules require before judgment on the merits may be granted."); *Bradley*, 913 F.2d at 1070 ("the district court's procedure converted the in limine motion into one for summary judgment, but without the procedural protections of notice which the federal rules require before judgment on the merits may be granted.") (internal

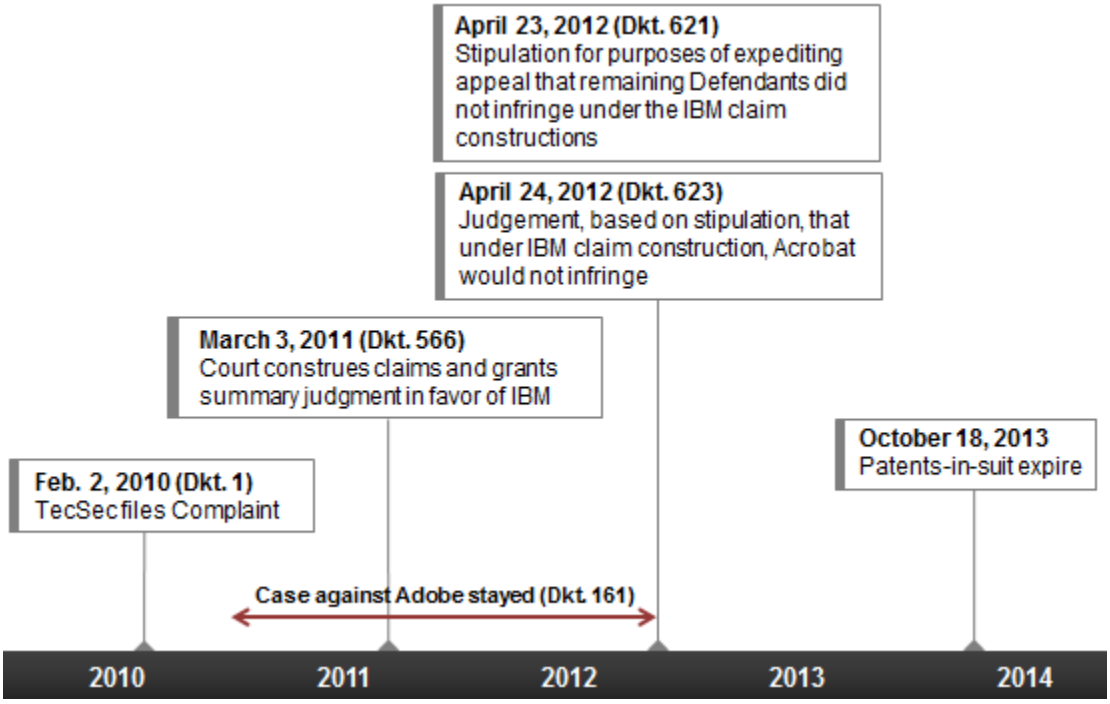
quotation marks omitted); *Williams v. Johnson*, 747 F.Supp.2d 10, 14 (D.D.C. 2010) ("In light of their limited purpose, motions in limine should not be used to resolve factual disputes, which remains the function of a motion for summary judgment, with its accompanying and crucial procedural safeguards.") (internal quotation marks omitted).

The present case demonstrates exactly why district courts are prohibited from resolving dispositive issues through *limine* motions. Here, Adobe did file a motion for summary judgment of no induced infringement. The district court, however, correctly denied Adobe's motion for summary judgment, finding that there were "material fact issues in dispute . . . which must be resolved by a factfinder." Appx10299. In that motion for summary judgment, Adobe either intentionally or inadvertently failed to raise the argument that the 2011 claim construction decision in the IBM case negated, as a matter of law, any possibility that Adobe could have had the intent to induce infringement. Instead, Adobe filed an omnibus motion *in limine* raising eleven discrete purportedly evidentiary issues. Motion *in Limine* #1, however, was a disguised, untimely summary judgment motion. In that motion, Adobe asked the Court to either: 1) preclude TecSec from presenting any evidence on the core issue of whether Adobe had the specific intent to induce infringement during the entire period from the Court's claim construction decision in 2011 that was subsequently reversed and the expiration of the patents-in-suit in 2013, or 2)

allow Adobe to present the claim construction order, TecSec's stipulation of non-infringement, and the Court's judgment as evidence of its good faith belief of non-infringement. *See, e.g.*, Appx10432 ("if TecSec insists on arguing to the jury that Adobe intended to infringe during this period, Adobe is entitled to rebut that argument."). Adobe concluded its argument in its Reply Brief by asking to "be allowed to present this highly probative evidence." Appx11687-11688. If Adobe believed that it was entitled to a dispositive decision on induced infringement based on a lack of intent, then it needed to raise that issue in its prior summary judgment motion or seek leave to file another summary judgment motion. *See* 75 Am. Jur.2d Trial § 44 (2017) (stating that motions *in limine* should not be used as "unnoticed motions for summary judgment or motions to dismiss....Clearly, a motion in limine may not properly be used as a vehicle to circumvent the requirements of rules of procedure."); 21 Charles Alan Wright & Kenneth Graham, Jr., Federal Practice and Procedure § 5037.10 (2d ed. 2005) (observing that "case law provides ammunition against those who would use the motion in limine as a substitute for a motion for summary judgment or other peremptory ruling"). Adobe did neither.

Nevertheless, the district court granted Adobe's summary judgment motion in disguise, and precluded TecSec from presenting its induced infringement claim for the period from 2011 until the patents expired in 2013. In the Court's Opinion and Order granting Adobe's Motion *in Limine* No. 1, the Court indicated that "[a]s a

matter of law, Defendant Adobe lacked the requisite intent to induce infringement or willfully infringe during the time period where the Court had reasonably, though erroneously, ***ruled in Adobe's favor on infringement*** and TecSec had entered a stipulation of non-infringement." Appx27 (emphasis added). This was clear error. The IBM claim construction order was not a ruling in Adobe's favor on non-infringement at all; it had nothing to do with Adobe. It does not mention Acrobat, and it did not find that Acrobat would not infringe under the court's erroneous construction.[3] The relevant timeline is as follows:



[3] In fact, the case against Adobe was stayed shortly after this action was filed. Appx1836. The Court confirmed on March 15, 2011—less than two weeks after it issued the claim construction order and granted summary judgment to IBM—that the case against Adobe remained stayed. Appx1837.

TecSec moved for reconsideration of the district court's decision, emphasizing that the district court's decision amounted to an improper grant of summary judgment, and that a retrial would almost certainly be required. Appx11881. At the hearing, the district court repeated its prior error, stating that Adobe "had an order of the Court that cleared them to continue to sell whatever products they felt were endangered by the patent litigation." Appx12213-12214, 12:24-13:1. To be clear, in 2011 there was no order pertaining to Adobe products whatsoever. And while Adobe could have argued at summary judgment that the 2012 stipulation removed its intent as a matter of law, it failed to do so. Nonetheless, the district court maintained its decision, stating: "I'm not going to have a mini-trial on what, you know, what Adobe sat back in their corporate headquarters and said, what should be do about this or what should we do about that." Appx12213, 12:21-24. But that is exactly what was required—the relevant inquiry for the jury was Adobe's state of mind during the entire infringement period—and TecSec was entitled to demonstrate at trial that the erroneous claim construction ruling had no impact whatsoever on Adobe's belief as to whether or not it was infringing.

In a case very similar to this one, this Court reviewed a district court's decision to address the sufficiency of an inequitable conduct defense on an evidentiary motion, finding that "the district court essentially converted [plaintiff's] motion *in limine* into a motion for summary judgment." *Meyer Intellectual Properties*, 690

F.3d at 1378. "In doing so, the court did not allow for full development of the evidence and deprived [defendant] of an opportunity to present all pertinent material to defend against the dismissal of its inequitable conduct defense." *Id.* In reversing and remanding, this Court concluded "that it was procedurally improper for the court to dispose of [defendant's] inequitable conduct defense on a motion in limine." *Id.* The Court here should similarly find that the district court's *de facto* grant of summary judgment at the limine phase was improper, and remand this issue for trial.

B. Whether Adobe Possessed the Requisite Intent to Induce Infringement from 2011-2013 Was A Fact Issue For the Jury

Even if it were permissible for the district court to resolve substantive, factual disputes through a motion in limine, and it is not, the district court still erred in taking a disputed factual issue away from the jury. "The Federal Rules of Evidence embody a strong and undeniable preference for admitting any evidence having some potential for assisting the trier of fact." *Holbrook v. Lykes Bros. Steamship Co.*, 80 F.3d 777, 780 (3d Cir. 1996) (quotations omitted). A court should only grant a motion *in limine* to exclude evidence if the evidence is obviously inadmissible on all possible grounds. *United States v. Verges*, No. 1:13cr222 (JCC), 2014 WL 559573, at *3, (E.D. Va. Feb. 12, 2014). "This principle applies because a court is almost always better situated during the actual trial to assess the value and utility of evidence." *Id.* (quotations and citation omitted).

Here, in Motion in Limine #1, Adobe did not seek to exclude any evidence proffered by TecSec that Adobe contends is irrelevant or inadmissible. Instead, in seeking to preclude relevant evidence of Adobe's intent, Adobe argued, without citation to any evidence, that it "***reasonably relied*** on the Court's Opinion to confirm its belief that it did not infringe." Appx10429 (emphasis added). The district court is not permitted to accept Adobe's attorneys' representation of disputed facts as a matter of law at the limine phase. Whether Adobe had the requisite intent to induce infringement and whether its belief that it did not infringe was "reasonable" is a classic fact issue; Adobe admitted as much in both its Memorandum and its Reply Brief. Specifically, in its Memorandum, Adobe argued that "TecSec defeated Adobe's summary judgment motion of no willful or induced infringement by arguing that this evidence 'raises factual questions that ***need to be addressed by a jury***.'" Appx10432 (emphasis in original). "***Adobe agrees.***" *Id.* (emphasis added).

Adobe repeated this position in its Reply Brief arguing:

> TecSec contends that Adobe could not have a good-faith belief of non-infringement because it has no (i) opinion of counsel or (ii) witnesses to testify on these issues. Dkt. 1234 at 3-4. ***That is for the jury to decide, based on 'all the circumstances,'*** including the Court's exculpatory order and TecSec's exculpatory stipulation. *WCM Indus., Inc. v. IPS Corp.*, 721 F. App'x 959, 970 (Fed. Cir. 2018) ('Whether an act is 'willful' is by definition a question of the actor's intent, the answer to which must be inferred from all the circumstances.').

Appx11686-11687 (emphasis added).

This Court has repeatedly held that a party's intent is a factual issue that must be resolved by the trier of fact. *See Allen Organ Co. v. Kimball Int'l, Inc.*, 839 F.2d 1556, 1567 (Fed. Cir. 1988) ("Intent is a factual determination particularly within the province of the trier of fact."); *WCM Indus*, 721 F. App'x at 970 (*citing Gustafson, Inc. v. Intersystems Indus., Prods., Inc.*, 897 F.2d 508, 510-11 (Fed. Cir. 1990) ("Whether an act is 'willful' is by definition a question of the actor's intent, the answer to which must be inferred *from all the circumstances*.")).

The district court's decision appears to be premised on the court's conclusion that Adobe *must have* unilaterally believed that it did not infringe under the erroneous claim construction order entered in favor of IBM. Appx28. But Adobe failed to identify any witness or evidence that it intended to produce at trial to demonstrate that it had a good faith belief of non-infringement based on the later-reversed 2011 claim construction decision. Further, TecSec could have presented evidence that Adobe was on notice of infringement at least as of the filing of the complaint, that Adobe stipulated to infringement, that it was aware that its communications to customers caused others to infringe, and that its actions continued unabated during the entire period from 2010-2013. Moreover, all of the evidence provided to date indicates that Acrobat ***is*** capable of encrypting multimedia files, even under the erroneous claim construction. *See, e.g.,* Appx25747-25748 (explaining that Acrobat is able to "[a]ssemble documents, spreadsheets,

presentations, e-mail, and web pages in a single compressed PDF file to easily share with others."); Appx25749-25752 (describing the ability to insert video), Appx26703, 85:24-86:07 (Landwehr Dep.) (Q. Okay. And once you have the individual files encrypted using Acrobat, you can then attach them to a PDF envelope using Acrobat, correct? A. You can attach any file format to PDF acting as an envelope whether or not it's encrypted.); Appx26668, 51:23-52:11, 52:19-22, 52:25-53:03 (Borstein Dep.) (Q. And with respect to a Portfolio, could it also contain encrypted files? A. Portfolio is a single PDF which can be combined or made up of other files. Potentially a video clip or anything or a Word file could be encrypted or a PDF could be encrypted."). Thus, it is unclear how Adobe could have ever believed that it had a good faith belief of non-infringement based on the Court's construction of the term "multimedia," at least for the period of time between March 3, 2011 and April 23, 2012.

Arguably, the 2012 stipulation filed by TecSec to get the case up on appeal could have impacted Adobe and caused them to have a good faith basis of non-infringement. But having failed to raise that issue through a summary judgment motion, Adobe's belief and intent became a fact issue for the jury to decide. At a minimum, there was a factual dispute between the parties as to the reasonableness of Adobe's continued provision of installation instructions through its manuals and encryption instructions through the blog post to its customers, which would result in

customers creating infringing systems and performing the claimed method steps. It was "procedurally improper" for the Court to preclude TecSec from offering evidence on a central claim in its case, and the district court's decision must be reversed so that this issue can be presented to the jury. *See Meyer Intellectual Props.*, 690 F.3d at 1378; *see* also, *Williams*, 747 F Supp 2d at 20 (holding that whether a party has met its burden regarding a disputed matter is "a question for the trier of fact.").

II. The District Court's Improper Exclusion of Evidence Requires a New Trial on Inducement

After creating the artificial one-year infringement and damages window through its limine decision, the district court compounded that error by using that decision as the basis for excluding additional probative evidence of Adobe's inducement and of third party direct infringement. Specifically, over TecSec's objection, the district court excluded the first-hand testimony of an Acrobat user that comprised an essential part of the inducement case against Adobe, as it constituted circumstantial evidence of direct infringement by a third party resulting from Adobe's inducing acts. By precluding TecSec from presenting the jury with critical evidence of direct third party infringement, the district court irreparably prejudiced TecSec, and all but guaranteed the jury's verdict of no inducement. The only remedy for the district court's error is for this Court to vacate the jury's verdict of no inducement and order a new trial on that issue for all asserted claims.

A. The District Court Excluded TecSec's Evidence of Direct Infringement by a Third Party

The Patent Act provides that "[w]hoever actively induces infringement of a patent shall be liable as an infringer." 35 U.S.C. §271(b) (2012). To prove inducement TecSec must show: (1) that Adobe's inducement act led to direct infringement by a third party; (2) that Adobe intended to cause the third party's infringing acts; and (3) Adobe knew that the induced acts would constitute patent infringement. *Power Integrations, Inc. v. Fairchild Semiconductor Int'l, Inc.*, 843 F.3d 1315, 1331 (Fed. Cir. 2016); *Global-Tech Appliances, Inc. v. SEB SA*, 563 U.S. 754, 766 (2011); *ACCO Brands, Inc. v. ABA Locks Mfr. Co.*, 501 F.3d 1307, 1312 (Fed. Cir. 2007).

TecSec presented evidence from which the jury could have concluded that during all relevant times Adobe had the intent to induce infringement and knew that the induced acts would constitute infringement. For example, TecSec presented undisputed evidence that installation of Acrobat on a computer infringes the asserted system claims. Appx12753-12755, 518:18-520:2; Appx12755, 520:13-16; Appx12768, 533:6-14; Appx12769, 534:4-5. TecSec also presented evidence that Adobe instructed its customers on how to install Acrobat, and posted a blog in 2009 instructing customers how to perform multi-level security. Appx24459-24460; Appx12769-12771, 534:6-536:1 (Jones); Appx19884; Appx16433; Appx17009. TecSec also presented evidence that the blog post first appeared on February 15,

2009, and remained publicly accessible throughout all times relevant to this law suit. Appx24459-24460. Evidence was presented to the jury that the blog post had been viewed at least 1,000 times by the public. Appx13725-13785; Appx13027, 792:9-17 (Landwehr).

To show that Adobe's inducement led to direct infringement by a third party, TecSec was prepared to present the jury with the testimony of Mr. George Johnson ("Mr. Johnson"), an independent consultant and long-time Acrobat user who fields and responds to online forum posts and inquiries regarding the features and functionality of Acrobat. Appx26685, 23:9-20 (Johnson Dep.).[4] Mr. Johnson is not employed by Adobe, nor is he paid by Adobe in any way. Appx26685, 26:13-27:03 (Johnson Dep.).

Specifically, TecSec was going to play for the jury Mr. Johnson's testimony concerning a forum post from another Acrobat user named "tpothen" titled "Double encrypting a pdf" to which Mr. Johnson responded. Appx24451-24452; *see also* Appx70-71, 69:02-70:10 (Johnson Dep.). The forum post, which was first appeared on an Adobe online forum at 2:35pm on April 15, 2011, reads as follows:

> I have a security audit in PDF format that is password protected from editing, you can open and view the PDF, which was encrypted by the auditing company and the password is unknown. My company does not allow sending unencrypted attachments. I need to send the PDF to

[4] Mr. Johnson did not attend trial. Rather, the district court permitted TecSec to present the jury with video clips of Mr. Johnson's testimony. Appx12716, 481:13-17.

> a client who does not allow .ZIP attachments. I am looking for another idea on how to completely encrypt the PDF with a separate password that I can give to the client. Any ideas? Thanks.

Appx24451.

Mr. Johnson responded to the post six minutes later (at 2:41pm) by proposing the following solution:

> Include the file as an attachment to another PDF that you create, and apply the security you want to it. The recipient can then open the file and extract the attachment.

Id.

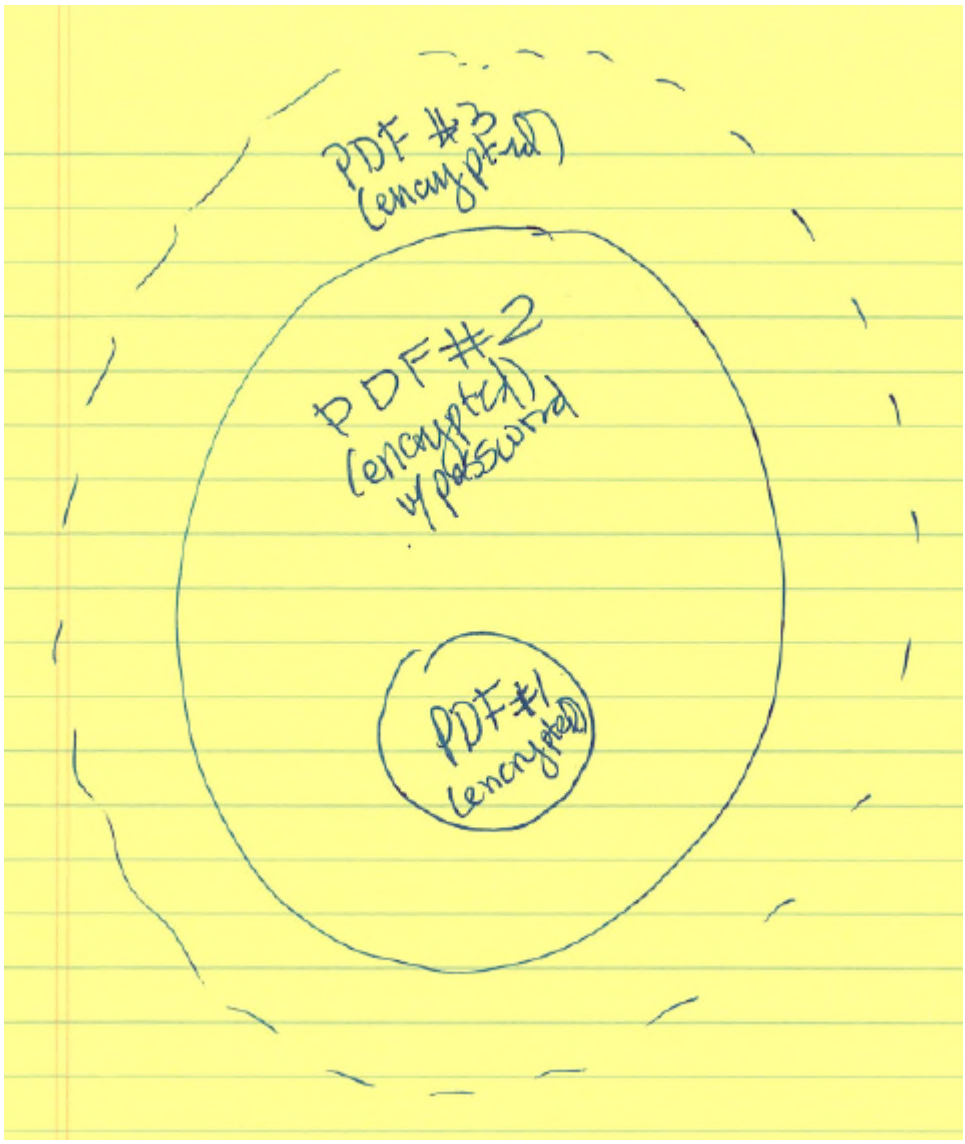
tpothen responded to Mr. Johnson at 2:55pm as follows:

> Thank you George, this works perfect. I did not know that this could be done in Acrobat.

Id.

TecSec was also going to show the jury Mr. Johnson's testimony that his proposed solution to tpothen is depicted in the hand-drawn image shown below.[5]

[5] The hand-drawn image was created in real-time during Mr. Johnson's deposition. Appx72, 74:4-9 (Johnson Dep.)



Appx24458.

Specifically, Mr. Johnson explained that pdf #1, which is encrypted, corresponded to tpothen's "security audit in pdf format." Appx72, 72:21-25 (Johnson Dep.). Pdf #2 corresponds to a pdf document that Mr. Johnson recommended that tpothen create, attach pdf #1 to, and then encrypt. *Id*., 73:14-74:9. Mr. Johnson further testified that in order to access pdf #1 (which is attached

to pdf#2), a recipient would first need to decrypt pdf #2. Appx72-73, 73:14-74:9 (Johnson Dep.). Mr. Johnson also testified that tpothen's comments that "this works perfect" told him that tpothen followed through on his advice. Appx73-74, 77:9-18 (Johnson Dep.).

TecSec's plan at trial was to present Mr. Johnson's testimony concerning the April 15, 2011 post, the hand-drawn image, and his exchange with tpothen as probative of direct infringement by a third party—a cornerstone element of its inducement case against Adobe. Adobe, however, objected to Mr. Johnson's testimony in its entirety, claiming that it referred to events that took place outside of the district court's newly created artificial damages window. Appx12426, 191:1-20. TecSec countered that—even if the blog post occurred outside of the damages window—Mr. Johnson's testimony is at least circumstantial evidence of direct infringement by Mr. Johnson and tpothen. Appx12430, 195:8-17. The district court agreed that Mr. Johnson's testimony was probative evidence of Adobe's "inducement," but sided with Adobe and struck the evidence as prejudicial and confusing because it occurred outside the artificially-narrowed damages window. Appx12429-12430, 194:23-195:5.[6]

[6] The district court also excluded testimony by Mr. Johnson acknowledging that Adobe instructs its customers on how to use Adobe Acrobat's security features [Appx70, 68:23-24, 69:02], as well Mr. Johnson's discussion of his use of multi-level security as part of his invoicing practice to his customers. *Id*., 65:15-20.

B. The District Court Abused its Discretion in Excluding Mr. Johnson's Testimony

The Federal Circuit reviews a district court's exclusion of evidence under the law of the regional circuit. *SSL Servs., LLC v. Citrix Sys., Inc.*, 769 F.3d 1073, 1092 (Fed. Cir. 2014). In reviewing a district court's decision to exclude evidence, the Fourth Circuit applies an abuse of discretion standard. *U.S. ex rel. Drakeford v. Tuomey*, 792 F.3d 364, 375 (4th Cir. 2015); *Buckley v. Mukasey*, 538 F.3d 306, 317 (4th Cir. 2008). "By definition, a district court abuses its discretion when it makes an error of law." *RZS Holdings AVV v. PDVSA Petroleo S.A.*, 506 F.3d 350, 356 (4th Cir. 2007). A reversal is in order if the district court's evidentiary error affects a party's substantial rights. *Buckley*, 538 F.3d at 317; *see also* Fed. R. Civ. P. 61; *Drakeford*, 792 F.3d at 375 (a new trial warranted if the exclusion results in "a high probability that the error ... affect[ed] the judgment.").

The district court abused its discretion in excluding Mr. Johnson's testimony. The district court's sole reason for excluding the testimony was because the events discussed took place outside the damages window. But as discussed above in Section I of the Argument, the district court's decision to limit the damages period to February 5, 2010 to March 10, 2011 was improper and constitutes legal error. The proper damages window should have been from February 5, 2010 until the expiration of the patents on October 18, 2013, which of course includes the April 15, 2011 post.

More importantly, however, the district court committed legal error in conflating the issue of damages with the issue of infringement liability. The Johnson evidence constitutes direct and circumstantial evidence of infringement liability, but the district court erroneously excluded it on the premise that it was outside of the damages window. By excluding Mr. Johnson's testimony that Adobe's third party customers performed multi-level encryption, it deprived the jury of key evidence that undoubtedly could have affected the jury's finding of no indirect infringement. This is the essence of TecSec's substantial rights under Fed. R. Civ. P. 62, and necessitates a new trial on this issue.

The same result must hold even if the district court's one-year damages window ruling is upheld as proper. That is, the jury should still have been permitted to see and hear Mr. Johnson's characterization of the April 15, 2011 post within the context of his other testimony which was presented to the jury, such as, for example, that prior to (and after) the 2011 tpothen post Mr. Johnson had performed multi-level security [Appx26687, 64:5-7, 64:11], and that Mr. Johnson had tested Acrobat security features on every version of Acrobat since version 5 (except version 8) [Appx26686, 40:17-25]. At a minimum, the full scope of Mr. Johnson's testimony, alongside the evidence showing how many times Adobe's 2009 website blog post was viewed, constitutes circumstantial evidence of inducement by Adobe. The district court erred by excluding such evidence from the jury.

C. A New Trial on Inducement is Necessary to Remedy the District Court's Exclusion of Evidence

A new trial is necessary because the district court's exclusion of Mr. Johnson's testimony directly affected TecSec's substantial rights and severely prejudiced TecSec's inducement case against Adobe. In preventing the jury from hearing Mr. Johnson's testimony regarding the circumstances surrounding the April 15, 2011 post, his awareness that Adobe instructs customers how to perform Acrobat's security features, his testing of Acrobat's security features, and his own personal use of multi-level security in his invoicing practices, the district court took away essential components of TecSec's inducement case against Adobe.

Further, there is a high probability that the district court's error in excluding the evidence affected the verdict. Indeed, taking away first-hand, probative evidence of direct infringement by at least one Adobe customer—a cornerstone requirement for inducement liability—all but guaranteed a jury finding of no inducement. Adobe's counsel repeatedly argued during closing that there is no evidence of direct infringement during the relevant damages period. Appx13508-13509, 1273:24-1274:4; Appx13540-13541, 1305:19-1306:2 ("There is no evidence that anybody did multi-level encryption during the period that the Court is going to talk about. The Court defined that period of February 2010 to March 2011. I don't get to pick that period, Mr. Oakes doesn't get to pick it, the judge does. Mr. Landwehr admitted that he must have done this when he created his blog in February 2009. That is

outside the Court's period. That doesn't count, according to the Court's instructions."). There is a high likelihood that the perceived lack of evidence during the damages period directly led to its finding of no inducement for all asserted claims. Under these circumstances, a new trial on inducement is TecSec's only recourse. *Drakeford*, 792 F.3d at 376 (granting a new trial on the basis that a witness's excluded testimony "was a relevant, and indeed essential, component of the government's evidence on [an] element" of its claim.)

III. The District Court's Misleading Jury Instructions Require a New Trial on Inducement

TecSec's inducement case against Adobe was based on three foundational contentions: (1) Adobe induces infringement by instructing customers how to install Acrobat to create infringing systems and how to infringe the method claims by performing multi-level security using Acrobat (through its 2009 blog post); (2) Adobe was aware that its inducing acts resulted in infringement by no later than the filing of the complaint; and (3) Adobe's intent to induce infringement is demonstrated by its instructing customers on how to install Acrobat and how to perform multi-level security through its continued provision of the website blog post after the lawsuit was filed.

However, TecSec's inducement case was critically hampered when the district court gave erroneous jury instructions combined with an inaccurate characterization of the evidence that, in the context of the district court's narrow

damages window and exclusion of TecSec's evidence of direct infringement by a third party, virtually ensured a finding of no inducement. Specifically, the district court's jury instructions (and Adobe's exploitation thereof) misled the jury by implying: (1) that the inducement case was focused solely on the February 2009 blog post, rather than on the installation instructions, (2) that the blog post was irrelevant to inducement during the narrowed damages window because it was created before TecSec's lawsuit was filed, and (3) that Adobe's continued provision of instructions through the blog post during the litigation amounted to inactivity, rather than continuous, active inducement. The prejudice to TecSec caused by the district court's misleading jury instructions, both of which were proposed by Adobe over TecSec's objections, can only be remedied by vacating the jury's verdict of no inducement and ordering a new trial on this issue as to all asserted claims.

1. Jury Instruction No. 25 Misled the Jury by Implying that the 2009 Blog is Irrelevant to Inducement

Immediately prior to trial, the parties entered into a Stipulation of Uncontroverted Facts ("Stipulation"). Appx11993-11997. However, near the close of trial, Adobe requested a final jury instruction (No. 25) to clarify that the Stipulation "is to a certain time frame." Appx13306, 1071:7-11. TecSec objected to the requested instruction and instead proposed giving the jury the stipulation so that they could see what the stipulation read word for word, without any risk of mischaracterizing the stipulated facts. TecSec also objected to the jury instruction

as argumentative, confusing, prejudicial and incomplete for leaving out key aspects of the stipulation, such as, for example, Adobe's stipulation that it internally installed each version of Acrobat. Appx13305-13306, 1070:13-1071:1; Appx11994 at ¶13.

The district court, over TecSec's objections, gave a modified version of Adobe's requested instruction, which stated in relevant part:

> The parties agree that the one time for which Adobe has admitted infringement, that is, Adobe's use in connection with creating a February 2009 blog post, occurred before TecSec filed its lawsuit in February of 2010.

Appx12076.

However, the district court's instruction was misleading for at least two reasons. First, TecSec never agreed that Adobe only admitted to a single act of infringement that occurred prior to the filing of the lawsuit. To the contrary, Adobe stipulated that every version of the accused Acrobat products infringed every asserted patent. Appx11994 at ¶13 (Adobe stipulates that "[o]n at least one occasion, Adobe has installed for its own internal use at least one copy of ***each version*** of the accused products on a computer having system memory.") (emphasis added). [7] Several accused versions were not even released until after 2009, and therefore the

[7] Adobe did not contest TecSec's evidence that installation of Acrobat on a computer infringes the asserted system claims. Appx12753-12755, 518:18-520:2; Appx12755, 520:13-16; Appx12768, 533:6-14; Appx12769, 534:4-5.

district court's characterization that the admission of infringement was limited to a pre-complaint time frame cannot be accurate. Appx23450-23451.

Second, the instruction is misleading because it implies that, in the context of the district court's narrow damages window, Adobe's 2009 blog is irrelevant for infringement purposes because it occurred *before* the lawsuit was filed. But, as noted above, Adobe's stipulation of infringement does not mention the blog, nor does it indicate that anything in the stipulation is limited to 2009.

As is further explained below, the instruction had a particularly deleterious effect on TecSec's inducement case because it was exploited by Adobe's counsel during closing to make the legally incorrect and misleading argument to the jury that TecSec's failure to take down the blog cannot be a basis for finding Adobe liable for induced infringement.

2. Jury Instruction No. 28 Misled the Jury into Finding No Inducement of any Asserted Claim

The parties agreed that Instruction No. 26, which comes straight out of the AIPLA model instructions, sets forth the proper standard for inducement. Appx12077; Appx13309, 1074:3-7. Adobe, however, requested an additional follow-on instruction—No. 28—purportedly for the purpose of clarifying the requirements for inducement. Appx13311, 1076:3-11. TecSec objected to

Instruction No. 28 as unnecessary, argumentative, confusing, prejudicial and structured to bait or mislead the jury.[8] Appx13309-13310, 1074:8-1075:11.

Instruction No. 28 states, in part, that "[e]vidence of mere inaction, or a failure to stop or prevent infringement, does not constitute inducement." Appx12079. TecSec never alleged, however, that Adobe induced infringement from failing to act. Rather, the active inducement stems from Adobe *actively* instructing customers, through its 2009 blog post, which Adobe continued to post through 2013 and even to this date, as to how to infringe by performing multi-level security using Acrobat.

Adobe's counsel, however, exploited the improper instruction during closing by improperly telling the jury that, ***as a matter of law***, it ***cannot*** find Adobe liable for inducement based on the district court's misstatement:

> The jury instruction that the Court read you said: Mere inaction or failure to stop isn't enough for inducement. That's the Court's instruction, and so this on its own, the not taking it down, is as a matter of law based on the Court's instruction not enough.

Appx13544, 1309:6-10. This is a rank mischaracterization of the law, and a clear attempt by Adobe to mislead the jury. In the Parties' Proposed Final Jury Instructions, Adobe cited *Tegal Corp. v. Tokyo Electron Co.*, 248 F.3d 1376 (Fed. Cir. 2001) as justification for its inclusion of the phrase "[e]vidence of mere inaction, or a failure to stop or prevent infringement, does not constitute inducement."

[8] The district court requested that the parties address Adobe's three proposed instructions collectively. Appx13308, 1073:21-23.

Appx11834. *Tegal* stands for the proposition that being aware of another's infringement but doing nothing to stop it cannot constitute *active* inducement. *Tegal*, 248 F.3d at 1378 ("Actively inducing" . . . requires an affirmative act of some kind.")

But that is not what TecSec accused Adobe of doing. Rather, TecSec alleges that Adobe *actively* induced infringement by posting a blog that teaches customers how to perform multi-level security. TecSec presented evidence that Adobe continued to post the blog throughout the damages window of the litigation, and that it had been accessed by Acrobat users nearly 1,000 times during even the limited, artificial damages window. Appx13725-13785. Therefore, Adobe's continued posting of the blog after being put on notice of infringement once the lawsuit commenced constitutes an affirmative act for inducement purposes. *See, e.g.*, *Toshiba Corp. v. Imation Corp.*, 990 F. Supp. 2d 882, 906, 909-910 (W.D. Wisc. 2013) (finding the taking down of a statement from defendant's FAQ page on its website was relevant to whether defendant induced infringement of the patent claims). The district court's improper instruction misled the jury, and enabled Adobe to misrepresent the facts and law during closing argument to mislead the jury into finding no inducement.

3. A New Trial is Necessary to Remedy the Prejudice to TecSec Caused by Jury Instructions Nos. 25 and 28

In the context of the district court's narrowing of the damages window and exclusion of critical evidence of direct infringement by a third party, the district

court's jury instructions Nos. 25 and 28 provided the framework exploited by Adobe to marginalize TecSec's infringement evidence and mislead the jury into a finding of no inducement. Instruction No. 25 is legally erroneous because it incorporates elements that were never agreed to by the parties, and also improperly diminishes the relevance and significance of Adobe's 2009 blog—Adobe's alleged inducing act. Instruction No. 28 provided Adobe with a platform to mischaracterize the law and mislead the jury into believing it could not, as a matter of law, find inducement.

The prejudice to TecSec resulting from the narrowed damages window, exclusion of critical evidence, and improper instructions could not be more glaring. Had the district court not improperly narrowed the damages window or excluded TecSec's evidence of direct infringement by a third party, there would have been additional unrebutted evidence in the record for the jury to find Adobe liable for induced infringement on all asserted claims. For example, given Mr. Johnson's testimony and the blog viewership statistics, there was at least circumstantial evidence that at least one customer read the 2009 blog and performed multi-level security during all relevant periods. There was also evidence Adobe continued to make the blog post available up until the expiration of the patents in 2013, which goes to Adobe's intent to induce infringement since at least the filing of the complaint. Likewise, there was ample evidence that customers installed Acrobat during the damages window based on Adobe instructions. Adobe presented no non-

infringement expert or defense of any kind as to infringement, and never disputed or contested TecSec's evidence that installation of Acrobat on a computer infringes the asserted system claims.

The improper instructions, however, were exploited by Adobe to mislead the jury into believing that it could not, as a matter of law, find inducement. Adobe's ability to frame the infringement issue as one lacking evidence was made possible because the district court's errors in limiting the damages window, excluding probative relevant evidence, and giving improper jury instructions that in the context of trial misled the jury into finding no inducement. The only remedy is for this court to vacate the jury's verdict of no inducement and order a new trial on all asserted claims. *See, e.g., Ecolab*, 285 F.3d at 1377-78 (vacating the jury verdict and remanding for a new trial: "Because we find sufficient evidence to support a jury verdict of infringement under the correct interpretation of claim 16, we conclude that Ecolab suffered prejudice by the erroneous jury instruction."); *see also*, *Malone v. Reliastar Life Ins. Co.*, 558 F.3d 683, 694 (7th Cir. 2009) (vacating judgment and remanding for a new trial where erroneous jury instruction, while independently harmless, combined with language on a special verdict form, taken together impermissibly caused jury bias).

IV. The District Court Erred By Vacating The Jury's Damages Award For Direct Infringement And Awarding Tecsec No Damages

Under 35 U.S.C. § 284, a finding of infringement "establishes the fact of damage because the patentee's right to exclude has been violated." *Lindemann Maschinenfabrik GmbH v. Am. Hoist & Derrick Co.*, 895 F.2d 1403, 1406 (Fed. Cir. 1990). The statute is unequivocal that, upon a finding of patent infringement, "*the district court must award damages* in an amount no less than a reasonable royalty." *Dow Chem. Co. v. Mee Indus., Inc.*, 341 F.3d 1370, 1381 (Fed. Cir. 2003) (emphasis added); *see also Promega Corp. v. Life Techs. Corp.*, 875 F.3d 651, 660 (Fed. Cir. 2017) (citing cases). Although the statute notes that "[t]he court ***may*** receive expert testimony as an aid to the determination of damages or of what royalty would be reasonable under the circumstances," 35 U.S.C. § 284 (emphasis added), such expert testimony is "not necessary to the award of damages." *Dow Chem.*, 341 F.3d at 1382. Expert testimony aside, a jury is within its province to weigh the evidence—including circumstantial evidence—and draw rational inferences therefrom in support of its verdict, and an appeals court must be "mindful of the Seventh Amendment's commission of factual questions to the jury and of [its] limited role of review." *Johnson v. Hugo's Skateway*, 974 F.2d 1408, 1412 (4th Cir. 1992).

Here, by granting Adobe's motion for judgment as a matter of law, the district court impermissibly vacated the jury's damages determination and awarded TecSec zero damages for Adobe's adjudged infringement. Stated another way, despite

upholding the jury's direct infringement verdict—finding all six asserted claims of the four DCOM Patents directly infringed[9]—the district court vacated the jury's damages award for being "inherently speculative and unsupported by the record," thereby substituting its own assessment of the weight and credibility of the record evidence in place of the jury's assessment. Appx59.

The district court's decision to vacate and nullify the jury's damages verdict contravenes the plain meaning of § 284 and usurps the jury's constitutional role as factfinder under the Seventh Amendment.

A. The Damages Statute Precludes Awarding Zero Damages When There is an Underlying Finding of Infringement

Giving plain and ordinary meaning to the language of § 284, an award of zero damages cannot withstand scrutiny. Indeed, in the event of infringement, the district court is *compelled* by the statute to award damages to the patentee in the event that the jury fails to do so. *See* 35 U.S.C. § 284 ("When the damages are not found by a jury, the court shall assess them."); *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 567 F.3d 1314, 1334 (Fed. Cir. 2009) (finding 0% jury royalty award inconsistent with finding of patent infringement and jury instruction that the jury choose a rate between 6% and 15%); *see also Del Mar Avionics, Inc. v. Quinton Instrument Co.*, 836 F.2d 1320, 1327 (Fed. Cir. 1987) ("The requirement to

[9] The district court disagreed with Adobe's argument that there was no direct infringement, and therefore denied JMOL on that issue. Appx54-55.

determine actual damages is not diminished by difficulty of determination."). So too does the opposite hold true; namely, where the jury has found damages for patent infringement, the district court may not subsequently reduce them to zero. *Dow Chem.*, 341 F.3d at 1382 ("The district court's conclusion that no damages could be awarded, in light of the presumption of damages when infringement is proven, was in error.").

The language of the statute unequivocally refutes the notion that zero damages may be awarded upon a finding of infringement—not once, but *twice*:

> Upon finding for the claimant the court shall award the claimant damages adequate to compensate for the infringement, but *in no event less than a reasonable royalty* for the use made of the invention by the infringer, together with interest and costs as fixed by the court.
>
> *When the damages are not found by a jury, the court shall assess them.*

35 U.S.C. § 284 (emphasis added).

In support of its decision to nullify the jury's damages award, the district court's JMOL opinion cites favorably to a single sentence from this Court's decision in *Apple, Inc. v. Motorola, Inc.*, where the Federal Circuit stated in dicta that "in a case completely lacking any evidence on which to base a damages award, the record may well support a zero royalty award." 757 F.3d 1386, 1328 (Fed. Cir. 2014).[10] Yet

[10] In *Apple*, the Federal Circuit reversed and vacated the district court's entry of summary judgment of no damages, holding that Motorola as the moving party failed to meet its burden of showing that "zero is the only reasonable royalty." *Id.* at 1329.

despite raising the specter of such a hypothetical situation, and in the very same paragraph of *Apple* cited by the district court, the Federal Circuit questioned such a notion, saying "it seems unlikely that a willing licensor and willing licensee would agree to a zero royalty payment in a hypothetical negotiation, where both infringement and validity are assumed." *Id.*; *see also*, *id.* at 1328 n.7 ("We note that we know of no case where we found that the record supported an infringement award of a zero royalty.").

In this case, the jury was fully instructed on the law regarding reasonable royalty damages, including the *Georgia-Pacific* factors that would have been considered by the hypothetically negotiating parties, pursuant to which infringement and validity of the patent claims are presumed. *See* Appx12097-12098 (jury instructions pertaining to reasonable royalty damages). While applying the law and weighing the record evidence from trial, the jury deliberated for nearly two full days before rending its verdict, which found direct infringement of all claims and awarded TecSec damages falling in between the experts' ranges. Appx44. The district court overstepped its bounds by upending the jury verdict—which, itself, was supported by substantial evidence (*see* infra SECTION IV.B.)—but even more egregiously, in granting JMOL, the district court contravened § 284 by assessing the same evidence the jury considered, rejecting the evidence, and reducing the damages to zero. Appx59 (JMOL order).

In support of its decision to nullify damages in this case, the district court cited favorably to the *Mirror Worlds* case, but this non-binding case from the Eastern District of Texas is immediately distinguishable because in addition to vacating the jury's damages award, the *Mirror Worlds* court contemporaneously granted the defendant's JMOL of no patent infringement as to all claims of the three asserted patents. *Mirror Worlds, LLC v. Apple, Inc.*, 784 F. Supp. 2d 703, 715, 720 (E.D. Tex. 2011). Accordingly, there was no underlying infringement upon which the district could would need to assess damages of its own, pursuant to the statute. *See id.* at 731.

The issue of weighing evidence and awarding damages for patent infringement falls squarely within the province of the jury, and district courts are reluctant to usurp the jury's fact-finding role in substitution of their own judgment. *See, e.g.*, *Waterton Polymer Prods. USA, LLC v. Edizone, LLC*, No. 2:12-CV-17 TS, 2014 WL 12600697, at *1 (D. Utah Nov. 25, 2014) (denying JMOL of no damages where the record evidence "on a variety of the *Georgia-Pacific* factors" supported the jury's damages award); *Graco Children's Prods. Inc. v. Kids II, Inc.*, No. 1:13-CV-1183-TWT, 2018 WL 1509192, at *10 (N.D. Ga. Mar. 26, 2018) (denying summary judgment of no damages "because the Plaintiff will still be entitled to some reasonable royalty amount, to later be determined by the Court, if the Generation One Products are found to infringe."); *Elbit Sys. Land & C4I Ltd. v. Hughes Network*

Sys., LLC, No. 2:15-CV-00037-RWS-RSP, 2017 WL 2651618, at *5 (E.D. Tex. June 20, 2017), report and recommendation adopted, 2017 WL 4693971 (E.D. Tex. July 31, 2017) (denying summary judgment where the record evidence "does not suggest that zero is the only reasonable royalty for the [asserted] patent."). Indeed, the Fourth Circuit—whose procedural law controls this issue—has so held in no uncertain terms. *See Johnson*, 974 F.2d at 1412 (declining to disturb the jury's damages verdict, which was based on reasonably weighing the record evidence and drawing inferences therefrom, because appeals courts must exercise "limited role of review" in issues of fact-finding and weighing evidence).

Here, the district court's grant of JMOL vacating damages in the case was an improper substitution of its own judgment for that of the jury. What's more, the district court's further decision to nullify all damages against TecSec contravenes the plain language of § 284 and is unsupported by any of the legal authority cited for support in the district court's opinion. On this basis alone, the district court's decision must be reversed.

B. The Jury's Damages Award is Supported by Substantial Evidence

In addition to the district court's decision constituting legal error (*see* supra Section IV.A.), the court erred in finding that there was no evidence upon which the jury could properly render a damages award for direct infringement without

complete speculation. *See* Appx55. To the contrary, the record evidence at trial fully supported the jury's damages award.

Underlying the district court's error was its belief that there can be no overlap in evidence for assessing reasonable royalty damages in connection with both direct and indirect infringement. *See* Appx59 (JMOL opinion describing one as an "alternative theory" of damages from the other). During the course of the five-day trial, the jury received significant evidence regarding both Adobe's own (direct) infringement of the patent claims, as well as regarding the infringement attributable to Adobe from use of the patent claims by Adobe's customers (indirect infringement). Although the jury ultimately decided that Adobe did not indirectly infringement the patent claims, it found that Adobe did infringe each asserted claim directly, and accordingly awarded TecSec damages in the form of a reasonable royalty. Appx40; Appx44.

A jury's award of damages is entitled to deference and "must be upheld unless the amount is grossly excessive or monstrous, clearly not supported by the evidence, or based only on speculation or guesswork." *Monsanto Co. v. Ralph*, 382 F.3d 1374, 1383 (Fed. Cir. 2004) (citing *Brooktree Corp. v. Advanced Micro Devices, Inc.*, 977 F.2d 1555, 1580 (Fed. Cir. 1992)); *Johnson v. Parrish*, 827 F.2d 988, 991 (4th Cir. 1987) (holding that vacating the jury's verdict is only when it is "against the clear

weight of the evidence, or is based upon evidence which is false, or will result in a miscarriage of justice.").

Like the underlying infringement verdict, which was upheld by the district court on JMOL, the damages awarded by the jury for Adobe's infringement were proper and based on substantial evidence. There can be no credible argument that the damages amount was grossly excessive or speculative because the amount awarded fell between the damages numbers presented by TecSec and Adobe during the trial, and indeed the awarded amount was less than half of the amount proposed by TecSec's expert in connection with the court's narrowed damages window. Specifically, the $1.75 million awarded fell squarely between the reasonable royalty amount proposed by Adobe's expert Dr. Sullivan of $142,735, Appx13212-13213, 977:22-978:3 (Sullivan), and the reasonable royalty amount proposed by TecSec's expert Mr. Wagner of $4,310,704, Appx12834-12835, 599:23-600:4 (Wagner). While the reasonable royalty rate used by the jury is not known because that specific jury question was not included on the verdict form (at Adobe's request), the jury was presented evidence of the royalty rates proposed by the parties' experts, as well as the rate that TecSec would have been looking for at the outset of a hypothetical negotiation. *Compare* Appx12863, 628:18-22 (Wagner) (proposing a royalty rate of 1.25 percent); and Appx12858-12859, 623:9-624:6 (discussing TecSec's prior

licensing offers of 1 percent per patent); *with* Appx13212-13213, 977:22-978:3 (Sullivan) (proposing a royalty rate of 0.53 percent).

The jury's award was perfectly logical given that it found that Adobe infringed each of the asserted claims directly, but not indirectly. The district court takes issue with what it believes was a lack of *specific* direct infringement damages testimony from TecSec's damages expert, but this concern is explicitly allayed by the language of § 284. Plainly stated, there is no requirement under the law to present expert testimony in order for a patent owner to be awarded damages. 35 U.S.C. § 284 ("The court *may* receive expert testimony as an aid to the determination of damages or of what royalty would be reasonable under the circumstances") (emphasis added); *and see Apple*, 757 F.3d at 1330 ("Even if Apple had not submitted expert evidence, this alone would not support a finding that zero is a reasonable royalty.").[11] It would have been absolutely permissible for the jury to have accepted one of the rates proposed by the competing experts or to have arrived at a different rate, in its own judgment, based on the evidence presented at trial. *See SmithKline Diagnostics, Inc. v. Helena Labs. Corp.*, 926 F.2d 1161, 1167 (Fed. Cir. 1991) ("[T]he factual

[11] In fact, even if the Court were to determine that TecSec failed to present any evidence of direct infringement damages, this still would not justify an award of no damages. *Dow Chem.*, 341 F.3d at 1382 ("Should Dow prove infringement of claims 23 and 24 the district court should consider the so-called *Georgia–Pacific* factors [] in detail, and award such reasonable royalties as the record evidence will support.").

determination of a reasonable royalty, however, need not be supported, and indeed, frequently is not supported by the specific figures advanced by either party.").

Nor is the jury's award speculative simply because the proposed hypothetical negotiation was based on sales of the accused products to Adobe's customers—a point of evidence that the district court inexplicably attributes to an "alternative" theory of infringement that it asserts has no bearing in the realm of direct infringement damages. Such a distinction is misplaced because the evidence of liability and damages adduced at trial regarding Adobe's direct and indirect infringement were *both* in connection with reasonable royalty damages. Indeed, the experts for both parties explained their damages rationale for *reasonable royalty damages* based on an analysis of the relevant *Georgia-Pacific* factors, and tellingly *both experts agreed* that sales of the accused Acrobat products would have been the proper consideration for the royalty base. [12] *See* Appx12840-12841, 605:6-606:4 (Wagner) (explaining appropriate form of royalty), Appx12841-12844, 606:24-609:2 (explaining calculation of baseline royalty rate), Appx12856-12862, 621:18-627:20 (describing *Georgia-Pacific* factor analysis); Appx26665 (TecSec's summary of damages showing sales of $344 million of accused products during the

[12] It is notable that, during trial, neither damages expert distinguished between an award for direct infringement versus indirect infringement. Yet, the district court drew a distinction of its own, attributed the experts' testimony entirely to indirect infringement damages, and concluded, improperly, that no evidence remained pertaining to direct infringement.

relevant period); Appx13213-13214, 978:14-979:15 (Sullivan) (describing areas of agreement with Wagner and use of *Georgia-Pacific* factors), Appx13238-13241, 1003:7-1006:13 (describing his calculation of baseline royalty).

The jury was free to accept the experts' testimony or reject it outright and arrive at a damages amount based on other factors, including the detailed testimony on the *Georgia-Pacific* factors or any other testimony presented. *See* Appx12058-12060. For example, the jury was presented with testimony indicating that Adobe employees installed Acrobat on their own computers, made presentations to customers using Acrobat, received feedback from customers based on their use of Acrobat, and modified the accused Acrobat products based on that customer feedback. *See*, *e.g.*, Appx12927, 692:14-17 (Ydens) (testifying he was employee 1544 when he joined Adobe in 1994); Appx12959, 724:15-20 (testifying Adobe tracks customer feedback on features in the product as well as development status of features in the product); Appx12963-12964, 728:23-729:7 (testifying on tracking of features going through engineering development); Appx12965, 730:6-20 (testifying on tracking of features moving towards completion); Appx12984, 749:20-22 (Landwehr) (testifying Adobe has about 200 employees in its Virginia office); Appx24913-24914; Appx26667, 43:16-43:23 (Borstein Dep.) (testifying that Adobe employees have Acrobat installed on computers); Appx13038-13039, 803:2-804:25 (Baartz-Bowman Dep.) (testifying that Adobe gathers customer feedback on

Acrobat); Appx26675-26678, 34:6-24, 35:14-15, 37:17-38:9, 41:22-42:11, 42:18-43:04, 48:10-49:15, 49:18-50:7, 50:15-21, 50:23-51:8, 51:19-20, 51:22-52:3, 52:05, 52:7-14, 52:16-18, 115:15-23, 149:12-20; Appx26681-26683, 151:7-12, 151:14-19, 151:21-152:14, 152:16-153:1 (Baartz-Bowman Dep.) (describing customer surveys and value propositions based on customer feedback for Acrobat); Appx15231. The jury could very well have reasonably concluded that Adobe could not have designed or developed new versions of Acrobat at all but for Adobe's direct infringement of at least the system claims by installing Acrobat internally for its own use and development. *See, e.g.*, Appx12783, 548:21-23 (Jones) ("if the claim is infringed, by installation, then that means that they're entitled to damages, is my understanding), Appx12824-12825, 589:24-590:1 ("Well, the system claims are infringed every time that the software is installed. So every installation is a making under 271(a).").

The evidence of record at trial, whether direct or circumstantial, was all fair game for the jury's consideration and deliberation in rendering a verdict. And indeed, the jury deliberated for two full days before reaching its final decision. A jury is within its province to weigh the evidence—including circumstantial evidence—and draw rational inferences therefrom in support of its verdict, and an appeals court must be "mindful of the Seventh Amendment's commission of factual questions to the jury and of [its] limited role of review." *Johnson*, 974 F.2d at 1412.

Even though the jury found that Adobe did not indirectly infringe, the jury still could have reasonably relied upon the substantial evidence of Adobe's direct infringement to arrive at a damages number that was based on the value of Adobe's own internal installation and use of Acrobat, including by considering, for example, that Adobe's ability to manufacture and sell the accused products related to its own direct infringement by Adobe's thousands of employees. Appx12927 (Ydens).

Based upon the substantial evidence adduced at trial, which the jury was *specifically instructed* to assess and weigh in arriving at its reasoned decision, the jury's damages award was far from speculation or guesswork, and it was error for the district court to vacate that award and substitute its own judgment.

C. TecSec Did Not Waive Pursuit of Reasonable Royalty Damages for Direct Infringement

The district court's JMOL opinion advances, as an additional basis for vacating the jury's damages and awarding nothing, that TecSec "deliberately [took] a risk by relying at trial exclusively on a damages theory that ultimately proved unsuccessful." Appx56-57 (citing *Promega Corp. v. Life Techs. Corp.*, 875 F.3d 651, 666 (Fed. Cir. 2017)). The district court's opinion suggests that because TecSec pursued damages "tied solely to Adobe's sales of Adobe Acrobat to customers," it has somehow waived its pursuit of direct infringement damages. But this notion of waiver is unsupported by both the case law and by the factual record.

In *Promega*, the plaintiff deliberately blurred the lines between infringing sales in the United States per § 271(a) and worldwide per § 271(f)(1) and elected throughout the case to pursue a single damages theory based solely on worldwide sales. 875 F.3d at 662. Promega failed to present damages figures at trial, or rebut argument in opposing the defendant's Rule 50(b) motion, advancing exemplary damages as a subset of total worldwide sales. *Id.* at 662-663. What's more, the district court "warned Promega during trial that it bore the burden to separately prove infringement under § 271(a) and § 271(f)(1)," yet it made a strategic decision that resulted in abandoning the alternative damages base. *Id. See also Keg Techs., Inc. v. Laimer*, 436 F. Supp. 2d 1364, 1360-71 (N.D. Ga. 2006) (finding plaintiff's sole pursuit of lost profits damages, forsaking a reasonable royalty recovery theory, waived pursuit of the latter upon failure of the former).

Unlike *Promega* and *Keg Techs.*, TecSec never abandoned pursuit of any damages base or theory of recovery that was later disproved or otherwise forsaken. First, TecSec did not pursue a lost profits theory of damage recovery. Second, TecSec did not strategically drop direct infringement in favor of indirect infringement (or vice versa), such that one would be the alternative recovery to the other. Indeed, the questions of direct and indirect infringement as to all asserted claims were separately put to the jury, and the jury returned in the affirmative for direct infringement as to all claims. Appx40-41. TecSec made clear that its damages

model and hypothetical negotiation was intended to encompass both direct and indirect infringement damages. Appx12155-12157. Accordingly, because the damages award was supported by substantial evidence, between the amounts proposed by the parties' experts, and never shown to be excessive or shocking to the conscience, the award should be reinstated.

CONCLUSION

For these reasons, the district court's limine and evidentiary rulings discussed above should be vacated, and TecSec should be afforded a new trial on indirect infringement using proper jury instructions. Moreover, the district court's decision vacating the damages award should be reversed, and the jury's damages award should be reinstated.

Respectfully Submitted

/s/ Michael A. Oakes

Michael A. Oakes
Ozzie A. Farres
Steven L. Wood
HUNTON ANDREWS KURTH LLP
2200 Pennsylvania Avenue, N.W.
Washington, D.C. 20037
Telephone: (202) 955-1894
Facsimile: (202) 857-3891

Counsel for TecSec, Inc.

October 7, 2019

ADDENDUM

IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division

TECSEC, INC., *Plaintiff*, v. ADOBE INC., *et al.*, *Defendants*.	Civil Action No. 1:10-cv-115 Hon. Liam O'Grady

ORDER

The Court, having ruled on all post-trial Motions in this case, the Court hereby **ORDERS** that pursuant to Rule 58 of the Federal Rules of Civil Procedure, judgment is **HEREBY ENTERED** on the jury verdict (*see* Dkt. 1353) as amended by the Court (*see* Dkt. 1402).

It is **SO ORDERED**.

June 18, 2019
Alexandria, Virginia

Liam O'Grady
United States District Judge

UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division

TECSEC, INC.,	)	
	)	
Plaintiff,	)	
v.	)	
	)	Case No. 1:10-cv-115-LO/TCB
ADOBE INC.,	)	
	)	
Defendant.	)	
	)	
	)	
	)	

ORDER

In consideration of TecSec's Unopposed Motion for Entry of Judgment under Fed. R. Civ. P. 54(b), it is hereby **ORDERED** that the Motion is **GRANTED**. The Judgment under Fed. R. Civ. P. 58 (Dkt. No. 1403) entered on the jury verdict, as amended by the Court, is hereby altered to be a final, appealable order under Fed. R. Civ. P. 54(b).

Entered this 3 day of July. 2019.

/s/
Liam O'Grady
United States District Judge

IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division

TECSEC, INCORPORATED, *Plaintiff,* v. ADOBE INC., *et al.* *Defendants.*	Civil Action No. 1:10-cv-115 Hon. Liam O'Grady

MEMORANDUM OPINION AND ORDER

This matter comes before the Court on Plaintiff TecSec Incorporated's Motions in Limine (Dkt. 1198), Defendant Adobe Inc.'s Motion for Leave to File Amended Witness List (Dkt. 1202), and Defendant Adobe Inc.'s Omnibus Motions in Limine (Dkt. 1207). The motions have been fully briefed, and the Court heard oral argument on November 9, 2018. As set forth below, for good cause shown, Plaintiff TecSec's Motions in Limine (Dkt. 1198) and Defendant Adobe's Omnibus Motions in Limine (Dkt. 1207) are both **GRANTED IN PART, DENIED IN PART, DEFERRED IN PART**, and Defendant Adobe's Motion for Leave to File Amended Witness List (Dkt. 1202) is **GRANTED**.

1. **TecSec's Motion in Limine #1: Motion to Preclude Adobe from Soliciting or Offering at Trial Any Testimony, Evidence, Statement or Argument Concerning Excluded and Withdrawn Prior Art References, Including PKZIP and TISPEM.**

For the following reasons, and for good cause shown, TecSec's Motion in Limine #1 is hereby **DENIED**.

a. PKZIP

TecSec argues that evidence, testimony, and argument regarding PKZIP should be excluded because (1) Adobe failed to disclose PKZIP in its June 22 supplemental disclosure as

required by the Court's June 8, 2018 Order and (2) TecSec would be prejudiced because Adobe's expert, Dr. Clark, does not explain how PKZIP individually renders the asserted claims invalid in his expert report. Adobe responds that the Court's June 8 Order required Adobe to identify only the combinations of prior art Dr. Clark would be relying upon, and therefore did not require Adobe to identify PKZIP because Dr. Clark would be relying upon PKZIP alone rather than in combination with other prior art. Adobe further argues that TecSec would not be prejudiced by Adobe's failure to identify PKZIP in its initial disclosure because TecSec has been on notice of Adobe's PKZIP contentions for over a year, both parties' experts have disclosed their opinions on PKZIP, and TecSec has had the opportunity to re-depose Dr. Clark at Adobe's expense.

While TecSec's Motion did seek to force Dr, Clark to identify the specific combinations that he would rely on for obviousness, the June 8 Order was not limited to requiring disclosure of the prior art *combinations* on which Dr. Clark would rely. The Order sought disclosure of the "prior art, *individually* or in combination, [Dr. Clark] believes renders any of the claims obvious" and ordered Adobe to accordingly "supplement its claim charts and analysis by specifically identifying *the* prior art Dr. Clark will rely on." Dkt. 1152 at 6 (emphasis added). The Court's Order did not speak directly to single art anticipation references that would have made it clear that Adobe was required to identify PKZIP in its June 22 disclosure. This lack of precision in the Court's Order should not be held against Adobe.

Further, the Court finds that TecSec was not prejudiced by Adobe's delayed disclosure because TecSec has been on notice of Adobe's PKZIP claims for years, TecSec's expert has analyzed PKZIP, and TecSec was given the opportunity to re-depose Dr. Clark at Adobe's expense but declined to do so. Under these circumstances, the Court determines that precluding testimony and evidence regarding PKZIP would be an unfair. As such, Adobe will not be

precluded from introducing evidence, testimony, and argument regarding PKZIP, but, as the Court indicated at the November 9 hearing, TecSec is granted permission to re-depose Dr. Clark at Adobe's expense regarding PKZIP if TecSec so chooses.

b. TISPEM

TecSec argues evidence and testimony concerning the TISPEM publications are precluded by the Court's June 8, 2018 Order and TecSec's view that Dr. Clark referenced only a TISPEM system, not TISPEM publications, as prior art in his expert report. The Court's June 8 Order precluded testimony about an alleged earlier iteration of the TISPEM *system* because there was no corroborating evidence of its existence; the Order did not preclude testimony about TISPEM *publications*. *See id.* at 5. Moreover, the TISPEM publications at issue were not only timely disclosed but also analyzed by both sides' experts. *See* Dkt. 1240-5 (Excerpts of Dr. Clark's Expert Report); Dkt. 1240-6 (Excerpts of Dr. Jones's Rebuttal Expert Report). As a result, while Adobe will be precluded from referencing a prior iteration of the TISPEM system pursuant to the June 8 Order, Adobe will be permitted to reference the disclosed TISPEM publications as prior art.

2. TecSec's Motion in Limine #2: Motion to Preclude Adobe from Soliciting or Offering at Trial Any Testimony, Evidence, Statement or Argument Concerning Computers, Software or Other Materials that Were Not Produced in an Accessible Form Prior to the Close of Fact Discovery.

a. Physical Exhibits

i. *PKZIP on Computers (DPX-001 & DPX-002)*

Adobe made DPX-001, a Grid computer running Windows 3.1 and PKZIP version 2.04g, available to TecSec's expert but did not disclose that PKZIP was on the computer or that Adobe intended to utilize the computer during trial to demonstrate PKZIP. *See* Dkt. 1240-10 (email to TecSec stating that the computers were available for inspection but making no reference to

PKZIP being on the computers). Because the presence of PKZIP on the computer was not properly disclosed to TecSec, TecSec's Motion in Limine #2 is **GRANTED** as to DPX-001.

TecSec's Motion in Limine #2 is also **GRANTED** as to DPX-002, another computer compatible with PKZIP, because the computer was never disclosed during discovery.

ii. *DPX-003, DPX-004, DPX-005 & DPX-008*

Pursuant to TecSec's representations in its reply brief, TecSec's Motion in Limine #2 is deemed **WITHDRAWN** as to DPX-003, DPX-004, DPX-005 and DPX-008.

iii. *DPX-006, DPX-007, & DPX-009–014*

Adobe did not produce DPX-006, DPX-007,[1] or DPX-009–014 during discovery, and argues now that these physical documents should be admissible at trial because they were publicly available. But parties are not excused from complying with discovery rules merely because documents are publicly available. *Godo Kaisha IP Bridge 1 v. Broadcom Ltd.*, 2017 WL 2869365, at *1–2 (E.D. Tex. May 18, 2017) (rejecting Defendants' argument that information did not need to be disclosed during discovery because it was publicly available). Additionally, Adobe's failure to properly disclose these exhibits has prejudiced TecSec by denying TecSec the opportunity to examine the documents and depose Adobe's witnesses on them. Because these documents were not properly disclosed during discovery and the failure to disclose was neither justified nor harmless, TecSec's Motion in Limine #2 is **GRANTED** as to these physical exhibits pursuant to Federal Rule of Civil Procedure 37(c)(1).

b. Documents Cited in Expert Reports: DX-169, DX-223, DX-224, DX-225, DX-282, DX-283, DX-363, DX-364, DX-367, DX-373, DX-374

[1] Adobe argues that DPX-007, the PKZIP 2.04g software, was made available to TecSec when Adobe gave TecSec the opportunity to examine two computers including DPX-001. As noted above, however, Adobe failed to properly disclose that the PKZIP software was on the computers. See Dkt. 1240-10 (email to TecSec stating that the computers were available for inspection but making no reference to PKZIP being on the computers). PKZIP 2.04g is also a different version of PKZIP than the version relied upon by Adobe's experts, and was neither disclosed in any expert reports nor shown to be an equivalent PKZIP version during discovery. As such, the PKZIP 2.04g software will be excluded.

The Court's Scheduling Order required experts to produce "all materials on which he or she relied . . . within five (5) days of service of the expert report." Dkt. 845 at 10. Adobe's expert, Dr. Sullivan, did not produce the documents subject to this Motion in Limine, but did cite the publicly available documents in his expert report. While Adobe's failure to produce the materials during discovery violated the Court's Scheduling Order, Dr. Sullivan's reference to the documents in his report put TecSec on notice of the documents, the documents were publicly available for TecSec's review, and TecSec's expert did in fact analyze and cite these documents in his expert report. The Court therefore finds that TecSec was not prejudiced by Adobe's failure to timely comply with the Court's Scheduling Order. As such, TecSec's Motion in Limine #2 is **DENIED** as to the documents cited in expert reports.

c. Dictionary Definitions of "Recursively": DX-10, DX-15, & DX-16

Claim construction in this case is complete and the dictionary definitions of "recursively" Adobe seeks to introduce were neither used during claim construction nor otherwise produced or cited by Adobe's experts. Therefore, TecSec's Motion in Limine #2 is **GRANTED** as to the dictionary definitions of "recursively."

d. Publicly Available Articles Dated After the Close of Discovery: DX-92 & DX-93

TecSec also seeks to exclude two publicly available articles that were published in June 2018, after the close of discovery, arguing that these articles were not timely produced. Adobe's failure to produce these articles during discovery is justified because the articles were not published until after the close of discovery. Moreover, the Court finds that Adobe's disclosure of the publications in its exhibit list on September 28, 2018, *see* Dkt. 1193-3, was sufficiently close in time to the publication of the June 2018 articles and early enough before trial to prevent undue prejudice to TecSec.

In a footnote in its reply brief, TecSec also suggests these articles should be excluded because they are irrelevant hearsay. The relevance and admissibility of the publications were not fully briefed, so the Court declines to rule on those issues at this time. Should Adobe seek to admit these articles at trial, TecSec may raise any relevancy or hearsay objections it deems appropriate at that time.

Accordingly, TecSec's Motion in Limine #2 is **DENIED WITHOUT PREJUDICE** as to these two publicly available articles dated after the close of discovery.

e. Other Undisclosed Publications

i. *Screenshots of the Accused Product Acrobat X: DX-8 & DX-9*

The screenshots at issue are screenshots of the accused product Acrobat X, which Adobe made available during discovery. As such, the screenshots will not prejudice TecSec and TecSec's Motion in Limine #2 is **DENIED** as to the screenshots.

ii. *Documents on Adobe's Website: DX-227, DX-228, DX-229, DX-232, DX-234, DX-235, DX-241, DX-242, DX-243, DX-245, DX-246, & DX-247*

TecSec seeks to exclude documents from Adobe's website that were not produced during discovery even though TecSec has itself offered as exhibits other documents from Adobe's website. While Adobe should have properly disclosed the documents during discovery, TecSec has not been prejudiced by Adobe's failure to disclose the objected-to documents because TecSec has searched Adobe's website and seeks to enter into evidence other portions of the same website. Accordingly, TecSec's Motion in Limine #2 is **DENIED** as to the documents on Adobe's website.

iii. *Exhibits from Third-Party Websites: DX-11, DX-13, DX-94, DX-122, DX-123, DX-124, DX-230, DX-231, DX-233, DX-236, DX-237, DX-238, DX-239, DX-240, DX-408, & DX-409*

As noted above, Adobe's failure to timely produce third-party documents is not excused merely because the documents are "publicly available" and TecSec has been prejudiced by Adobe's failure to disclose these third-party websites during discovery. As a result, TecSec's Motion in Limine #2 is **GRANTED** as to the documents from third-party websites that were not disclosed prior to the end of discovery.

f. Damages Summary Exhibits: DX-529–34

Lastly, TecSec seeks to exclude Adobe's damages summary exhibits. This is a complex case with a voluminous record, and the Court finds that the damages summary exhibits are likely highly probative, would help the jury better understand the evidence, and were timely produced trial exhibits. "It is well-established that a district court may admit summary testimony and summary charts, under Rule 611(a) of the Federal Rules of Evidence, if they will help the jury to better understand the evidence presented and to ascertain the truth." *United States v. Nolberto Pena*, 213 F.3d 634, 2000 WL 541087, at *3 (4th Cir. April 27, 2000) (citing *United States v. Johnson*, 54 F.3d 1150, 1159 (4th Cir. 1995)). Any prejudice caused to TecSec by allowing the summary exhibits "may be dispelled by giving [TecSec] an opportunity to cross-examine the individual who prepared the chart" and/or through "a cautionary jury instruction." *Id.* (quoting *United States v. Loayza*, 107 F.3d 257, 264 (4th Cir. 1997)). That being said, Adobe's failure to identify the sources of the summarized evidence in the summary exhibits hampers TecSec's ability to effectively cross-examine the producer of the exhibits and the Court's and jury's ability to assess the reliability of the summary exhibits.

For these reasons, TecSec's Motion in Limine #2 is **DENIED WITHOUT PREJUDICE** as to Adobe's damages summary exhibits but Adobe is **ORDERED** to identify the sources of the information contained in the summary exhibits no later than seven (7) days before trial.

3. **TecSec's Motion in Limine #3: Motion to Preclude Adobe from Soliciting or Offering at Trial Non-Infringement Related Opinion Testimony by John Landwehr, Leonard Rosenthol or Phil Ydens.**

John Landwehr, Leonard Rosenthol, and Phil Ydens were properly identified by Adobe as fact witnesses but later also identified as expert witnesses after the deadline for disclosing rebuttal experts had passed. In an effort to address this issue without the assistance of the Court, TecSec proposed the following stipulation:

> **Opinions from John Landwehr, Leonard Rosenthol or Phil Ydens**: Adobe will not elicit any opinion testimony from John Landwehr, Leonard Rosenthol or Phil Ydens on the ultimate issues of noninfringement or non-infringing alternatives. This agreement does not preclude Adobe from eliciting factual testimony from Messrs. Landwehr, Rosenthol, or Ydens regarding, for example, the features, functionality, or use of Acrobat, including but not limited to any changes that have been or could be made to Acrobat's features and functionality.

Dkt. 1240-22 at 7 (redlined changes in original accepted). Adobe agreed to this proposed stipulation, Dkt. 1240-23 at 3, but rather than agree to jointly file the stipulation, TecSec instead brought this Motion in Limine. Under these facts, TecSec's Motion in Limine #3 is hereby **GRANTED IN PART, DENIED IN PART** pursuant to the above excerpted proposed stipulation, which the Court hereby **ADOPTS** as binding.

4. **Adobe's Motion for Leave to File Amended Witness List & TecSec's Motion in Limine #4: Motion to Preclude Adobe from Soliciting or Offering at Trial Testimony by Russ Mundy.**

TecSec seeks to preclude testimony by Russ Mundy because Adobe inadvertently left Russ Mundy off its trial witness list. Adobe realized that it had inadvertently failed to include Russ Mundy in its trial witness list within two weeks, notified TecSec of the error and attempted to resolve the issue without Court intervention, and then promptly filed its Motion to File

Amended Witness List when its meet and confer efforts failed. Adobe has shown good cause to allow the amendment, as the inadvertent error was discovered within two weeks and the delay in disclosing Russ Mundy as a trial witness will not prejudice TecSec since both parties agree that Russ Mundy was identified as a potential witness over a year ago.

TecSec also argues that Russ Mundy should be excluded from testifying at trial because he does not have any relevant knowledge. Russ Mundy was previously identified as having personal knowledge of only "[p]rior art systems and methods that were publically available and which render the patents-in-suit invalid." Dkt. 1200-10 at 9. TecSec has argued in its Motion in Limine #1 that all the remaining prior art at issue should be excluded from trial, and therefore argues that Russ Mundy should also be excluded from testifying. As the Court has denied TecSec's Motion in Limine #1, TecSec's relevance objections to Russ Mundy also fail.

For these reasons and for good cause shown, Adobe's Motion for Leave to File Amended Witness List is hereby **GRANTED** and TecSec's Motion in Limine #4 is hereby **DENIED**.

5. **TecSec's Motion in Limine #5: Motion to Preclude Adobe from Soliciting or Offering at Trial Testimony by David Balenson or William Barker.**

Similar to TecSec's relevance objections to Russ Mundy, TecSec argues that both David Balenson and William Barker should be excluded at trial because they were identified as having factual knowledge of only "[p]rior art systems and methods that were publically available and which render the patents-in-suit invalid" and TecSec's Motion in Limine #1 argues that the only prior art still at issue should be excluded. As the Court has denied TecSec's Motion in Limine #1, TecSec's relevancy objections to David Balenson and William Barker also fail. As such, TecSec's Motion in Limine #5 is **DENIED**.

6. TecSec's Motion in Limine #6: Motion to Preclude Adobe from Soliciting or Offering at Trial Testimony by Doug Brotz.

Doug Brotz was identified by Adobe as a fact witness having knowledge of "[l]ack of notice of the patents-in-suit." Dkt. 1200-10 at 5. TecSec chose not to depose Mr. Brotz, and now seeks to exclude testimony regarding the results of an investigation Mr. Brotz conducted into whether Adobe was on notice of the patent-in-suit. Mr. Brotz's relevant knowledge in this case was properly disclosed, as his investigation gave Mr. Brotz personal knowledge regarding Adobe's lack of notice of the patents-in-suit. Accordingly, TecSec's Motion in Limine #6 is **DENIED**. Mr. Brotz will be permitted to testify about his personal knowledge regarding Adobe's lack of notice of the patents-in-suit, including his own lack of knowledge of the patents, the investigation he personally conducted to the extent permitted by the rules against hearsay, and the conclusions he drew from his investigation.

7. TecSec's Motion in Limine #7: Motion to Preclude Adobe from Soliciting or Offering at Trial Any Testimony, Evidence, Statement or Argument Concerning TecSec's Claims Against Other Parties in the Present Lawsuit.

TecSec seeks to exclude all reference to TecSec's claims against other parties in the present lawsuit as irrelevant. Adobe responds that it should be permitted to reference the other parties to (1) explain the delay in this case coming to trial so the jury will not blame Adobe for the delay, and (2) rebut TecSec's argument that Adobe intended to infringe the asserted patents while the Court's earlier claim construction was on appeal at the Federal Circuit. The Court finds that reference to the other parties in the present lawsuit has no relevance to TecSec's claims against Adobe. *See infra* at 12–15 (the Court's ruling on Adobe's Motion in Limine #1). The Court will notify the jury that the delay was caused by the Court's actions. TecSec's Motion in Limine #7 is therefore **GRANTED**.

8. TecSec's Motion in Limine #8: Motion to Preclude Adobe from Soliciting or Offering at Trial Any Testimony, Evidence, Statement or Argument Concerning the Pretrial Withdrawal or Dismissal of Patent Infringement Claims Not Being Asserted at Trial or Products that Are No Longer Being Accused of Infringement.

The Court finds that the withdrawn or dismissed infringement claims and products are not relevant to a determination of whether Adobe infringed the remaining products at issue. *See Corning Inc. v. SRU Biosystems*, 2005 WL 2465900, at *2–3 (D. Del. Oct. 5, 2005) (excluding evidence related to dismissed claims because the dismissed claims were irrelevant). Adobe unpersuasively cites two non-binding district court cases in support of its argument that the dismissed claims and products are relevant to "all the circumstances" the court must consider in determining willfulness: *Uniloc USA, Inc. v. Microsoft Corp.*, 640 F. Supp. 2d. 150 (D.R.I. 2009), and *Carnegie Mellon University v. Marvell Technology Group, Ltd.*, 2012 WL 5463669 (W.D. Pa. Nov. 7, 2012). In both cases, the district judge allowed evidence of dropped or dismissed claims or products as relevant evidence of the objectiveness prong of willful infringement under *Seagate*. *Uniloc*, 640 F. Supp. 2d at 176–77; *Carnegie Mellon*, 2012 WL 5463669, at *2. Both cases, however, predate the Supreme Court's decision in *Halo Electronics, Inc. v. Pulse Electronics, Inc.* rejecting the objectiveness prong of *Seagate*'s willfulness test. 136 S. Ct. 1923, 1932–33 (2016).

In *Halo*, the Supreme Court took issue with the fact that the *Seagate* objectiveness test "mak[es] dispositive the ability of the infringer to muster a reasonable . . . defense at the infringement trial" because "culpability is generally measured against the knowledge of the actor at the time of the challenged conduct." *Id.* at 1933. Allowing evidence of dropped or dismissed claims and products in this case would not only confuse the jury, but also violate the Supreme Court's determination in *Halo* that defenses to claims should not serve as a basis for determining willfulness. The dropped or dismissed claims and products are irrelevant to Adobe's culpability

at the time of the challenged conduct and they are also unduly prejudicial and confusing to the jury. For these reasons, and for good cause shown, TecSec's Motion in Limine #8 is hereby **GRANTED**.

9. **TecSec's Motion in Limine #9: Motion to Preclude Adobe from Soliciting or Offering at Trial Testimony by Roy Follendore.**

Adobe has indicated that it "may" call Roy Follendore at trial, either live or by deposition. Dkt. 1193-1 at 2. Adobe's designated portions of Mr. Follendore's deposition include discussions of Mr. Follendore's background, his personal familiarity with Adobe Acrobat, a now-dropped prior art called "Net Shield," and Mr. Follendore's opinion of the state of the art of "labels." Since Adobe has dropped Net Shield as an asserted prior art, the Court is unclear as to what other relevant testimony Mr. Follendore would provide at trial. It is also unclear at this stage whether the Court will ultimately need to rule on the admissibility of Mr. Follendore's designated deposition testimony, as Adobe has indicated it may choose to call Mr. Follendore live or not at all. As a result, the Court will **DEFER** ruling on TecSec's Motion in Limine #9 until Adobe seeks to admit Mr. Follendore's testimony at trial.

10. **Adobe's Motion in Limine #1: Motion to Preclude Argument, Evidence, or Testimony on Adobe's Intent to Induce or Willfully Infringe Between March 3, 2011 and October 18, 2013, Unless TecSec Concedes Admissibility of the March 3, 2011 Order (DX-28), Stipulation of Non-Infringement (DX-29), and Judgment (DX-30).**

On March 3, 2011, Judge Brinkema adopted a claim construction in a Memorandum Opinion and Order under which Adobe's allegedly infringing products would not have infringed TecSec's patents. *See* Dkts. 566–67. In light of Judge Brinkema's ruling, TecSec entered a stipulation on April 23, 2012 agreeing that under Judge Brinkema's claim construction, "the Defendants have not infringed and do not infringe the asserted claims of the DCOM Patents." Dkt. 621 ¶ 5. TecSec appealed Judge Brinkema's ruling to the Federal Circuit, staying the case, and the DCOM patents expired while the case was still at the Federal Circuit. Adobe seeks to

preclude TecSec from arguing that Adobe induced infringement or willfully infringed between the date of Judge Brinkema's claim construction Order in Adobe's favor and the expiration of the patents. Alternatively, should TecSec make such an argument at trial, Adobe seeks to enter into evidence Judge Brinkema's Order, TecSec's stipulation of non-infringement, and the Court's judgment that allowed the appeal to the Federal Circuit. The Court has considered the parties' original briefing, their arguments at the November 9 hearing, and their supplemental briefing submitted after the Court's ruling from the bench.

As a matter of law, Defendant Adobe lacked the requisite intent to induce infringement or willfully infringe during the time period where the Court had reasonably, though erroneously, ruled in Adobe's favor on infringement and TecSec had entered a stipulation of non-infringement. *Move, Inc. v. Real Estate All. Ltd.*, 221 F. Supp. 3d 1149, 1173 (C.D. Cal. 2016) (granting summary judgment of no willful infringement where the plaintiff "stipulated to a finding of non-infringement based on [a] reasonable, but erroneous, construction"); *Toshiba Corp. v. Imation Corp.*, 990 F. Supp. 2d 882, 912 (W.D. Wis. 2013) ("[N]o reasonable fact finder could find that defendants formed the requisite intent to induce infringement between the time this court ruled in their favor on summary judgment and the time the Federal Circuit reversed that decision. . . . Defendants simply could not have 'known' that they were inducing infringement when they had a judicial declaration telling them that they weren't."); *Chaffin v. Braden*, 2018 WL 1794766, at *5 (S.D. Tex. April 16, 2018) (holding that where the district court had entered a reasonable, though later over-turned, claim construction in favor of the defendants, the defendants "had an objectively reasonable understanding that their Accused Devices were non-infringing" and were therefore entitled to summary judgment on the induced infringement claim).

TecSec argues that the Court cannot preclude TecSec from arguing induced or willful infringement during that time period at the motion in limine stage because the Court's ruling would be tantamount to an impermissible "de facto summary judgment." "[T]he main purpose of a motion in limine is to streamline the trial by keeping irrelevant or improper evidence out of the courtroom," and it is therefore proper for a Court at the motion in limine stage to preclude arguments at trial, such as the one at issue, without factual or legal support. *Amdocs (Israel) Ltd. V. Openet Telecom, Inc.*, 2012 WL 12833699, at *1 (E.D. Va. April 3, 2012); *see also*, *Anchor Wall Sys. v. Rockwood Retaining Walls, Inc.*, 610 F. Supp. 2d 998, 1007–09 (D. Minn. 2009) (noting that the parties "clearly could have raised" – and should have raised – the same motion at issue here in a motion in limine). Indeed, the Federal Circuit has upheld such rulings on motions in limine even though the rulings amounted to de facto summary judgments. *Trading Techs. Int'l, Inc. v. eSpeed, Inc.*, 595 F.3d 1340, 1361 (Fed. Cir. 2010) (affirming order on motion in limine precluding a defense at trial); *Colassi v. Cybex Int'l. Inc.*, 221 F. App'x 973, 976–77 (Fed. Cir. 2007) (affirming order on motion in limine finding that the reverse doctrine of equivalents did not apply even though the order may have been a "de facto summary judgment").

Moreover, allowing TecSec to argue induced or willful infringement during the time period when Judge Brinkema's Order and TecSec's stipulation were in effect would taint the trial and any verdict with undue prejudice and juror confusion. *Anchor Wall Sys.*, 610 F. Supp. 2d at 1005–07. On the one hand, it would be substantially and unduly prejudicial to Adobe to allow TecSec to argue induced and willful infringement during that time period but not allow Adobe to show the jury the Court's Order and TecSec's stipulation that gave Adobe a reasonable belief that its products were not infringing. On the other hand, it would also be substantially and unduly

prejudicial and confusing for the jury to see a prior ruling in this case and TecSec's stipulation that Adobe's products did not infringe. *Id.*

Accordingly, for these reasons and for good cause shown, Adobe's Motion in Limine #1 is **GRANTED**. TecSec will be precluded from arguing at trial that Adobe induced infringement or willfully infringed the patents-in-suit after the March 3, 2011 Order was entered.

11. Adobe's Motion in Limine #2: Motion to Preclude TecSec from Presenting Argument, Evidence, or Testimony Regarding Non-Comparable Licenses Except for Limited Purpose of Establishing or Refuting "Marking" & Adobe's Motion in Limine #5: Motion to Preclude Argument, Evidence, or Testimony Regarding Secondary Considerations of Non-Obviousness that Lack a Nexus to the Merits of the Claimed Invention.

Adobe's Motions in Limine #2 and #5 both concern the admissibility of licenses of TecSec's products. The Court has reviewed the licenses at issue and has determined that all the licenses are relevant to secondary considerations of obviousness and, at least according to TecSec's experts, have a close nexus to the patents-in-suit.[2] It will be up to the jury to decide whether a sufficient nexus exists between the claimed technology and the licenses and other documents TecSec argues demonstrate commercial success. *See Pro-Mold & Tool Co., Inc. v. Great Lakes Plastics, Inc.*, 75 F.3d 1568, 1574 (Fed. Cir. 1996) ("It is within the province of the fact-finder to resolve these factual disputes regarding whether a nexus exists between the commercial success of the product and its patented features, and to determine the probative value of [Plaintiff's] evidence of secondary considerations . . . ."); *Morpho Detection, Inc. v. Smiths Detection, Inc.*, 957 F. Supp. 2d 655, 679 (E.D. Va. 2013) (holding the existence of a sufficient nexus is "a factual question for the jury").

[2] While two of the licenses (DX-110 and PX-156) are hybrids between a settlement agreement and a license, and settlement agreements are generally barred by Federal Rule of Evidence 403, the Court finds that because the agreements are also licenses, they should be admitted at trial as relevant to secondary considerations of obviousness.

The amounts paid are also relevant to secondary considerations of obviousness and will be admissible for that limited purpose. The amounts paid will not be admissible to suggest a reasonable royalty rate because both parties' experts have agreed that the licenses are not comparable.

In accordance with these rulings, Adobe's Motions in Limine #2 and #5 are **GRANTED IN PART, DENIED IN PART**.

12. Adobe's Motion in Limine #3: Motion to Preclude Expert Testimony Regarding Damages Based on Alleged Direct Infringement by Adobe.

Adobe seeks to prevent TecSec's damages expert, Mr. Wagner, from testifying about direct infringement damages because Mr. Wagner opines on a reasonable royalty based on Adobe's sales of Acrobat to customers, not based on Adobe's own use of Acrobat. TecSec has stated to the Court that Mr. Wagner's testimony at trial will be limited to the statements he made in his expert report. Accordingly, Mr. Wagner's testimony was timely and properly disclosed. Furthermore, Adobe's arguments regarding Mr. Wagner's methodology and conclusions go to weight, not admissibility, and are therefore proper topics for cross-examination but are not alone a sufficient basis to exclude Mr. Wagnor's testimony regarding direct infringement. Adobe's Motion in Limine #3 is therefore **DENIED**.

13. Adobe's Motion in Limine #4: Motion to Preclude Argument, Evidence, or Testimony Regarding the Lack of an Opinion of Counsel.

The Federal Circuit has held that "there is no affirmative obligation to obtain opinion of counsel" and "an accused infringer's failure to obtain legal advice does not give rise to an adverse inference with respect to willfulness." *In re Seagate Tech., LLC*, 497 F.3d 1360, 1370–71 (Fed. Cir. 2007), *abrogated on other grounds by Halo Elecs., Inc. v. Pulse Elecs., Inc.*, 136 S. Ct. 1923 (2016). At the same time, however, the Federal Circuit has also held that the failure to obtain the advice of counsel is a relevant factor to consider in analyzing inducement and

willfulness. *Suprema, Inc. v. Int'l Trade Comm'n*, 626 F. App'x 273, 282 (Fed. Cir. 2015); *Broadcom Corp. v. Qualcomm Inc.*, 543 F.3d 683, 699–700 (Fed. Cir. 2008).

For these reasons, and for good cause shown, Adobe's Motion in Limine #4 is hereby **GRANTED IN PART, DENIED IN PART**. TecSec will be permitted to introduce evidence or arguments regarding the absence of an opinion of counsel at trial to prove inducement or willfulness, but TecSec cannot indicate Adobe had an affirmative duty to seek an opinion of counsel nor indicate that any adverse inference should be drawn as to what the content of an opinion of counsel would have been. *Georgetown Rail Equip. Co. v. Holland L.P.*, 2014 WL 6467782 (E.D. Tex. Nov. 17, 2014) (considering a similar motion in limine and adopting a similar order).

14. Adobe's Motion in Limine #6: Motion to Preclude References to Pre-Suit Damages Unless and Until TecSec Makes a *Prima Facie* Showing that It Can Meet Its Burden Under 35 U.S.C. § 287(A).

Adobe seeks to prevent TecSec from referencing $41 million in alleged damages during its opening statement because Adobe believes TecSec will not meet its burden under 35 U.S.C. § 287(A) to establish marking. TecSec would not be prejudiced by the inability to refer to "$41 million" during its opening as TecSec could explain its damages in reasonable royalty terms during opening and, if TecSec meets its burden of proving marking, have its expert testify during trial that TecSec suffered $41 million in damages. By contrast, if TecSec tells the jury during opening that it suffered $41 million in harm but is not able to meet its burden to prove sixty percent of these damages, then the "damages horizon" in the case would be impermissibly skewed in the minds of the jurors and Adobe unfairly prejudiced as a result. *See Uniloc USA, Inc. v. Microsoft Corp.*, 632 F.3d 1292, 1320–21 (Fed. Cir. 2011) (finding that the improper disclosure that defendant made $19 billion in revenue "cannot help but skew the damages horizon for the jury" and could not be cured by cross-examination or jury instructions).

Therefore, Adobe's Motion in Limine #6 is hereby **GRANTED**: TecSec will be precluded from telling the jury that it has suffered $41 million in damages unless and until TecSec's experts sufficiently establish marking.

15. Adobe's Motion in Limine #7: Motion to Preclude TecSec from Arguing that Accused Third-Party Products Did Not Practice the Asserted Patents.

The doctrine of judicial estoppel appears to preclude TecSec from arguing that the referenced Microsoft products did not practice the asserted patents because TecSec asserted the opposite position when it previously sued Microsoft alleging infringement of the same patents at issue in this case. *Lowery v. Stovall*, 92 F.3d 219, 223 (4th Cir. 1996) ("Judicial estoppel precludes a party from adopting a position that is inconsistent with a stance taken in prior litigation." (quoting *John S. Clark Co. v. Faggert & Frieden, P.C.*, 65 F.3d 26, 28–29 (4th Cir. 1995))). Nevertheless, the Court hereby **DEFERS** ruling on Adobe's Motion in Limine #7 until this issue arises at trial, at which time the Court will further consider whether TecSec should be precluded from arguing that Microsoft's products did not practice the asserted patents.

16. Adobe's Motion in Limine #8: Motion to Preclude TecSec from Calling or Making Reference to (1) Witnesses Not Disclosed in Initial Disclosures and (2) Duplicative 30(B)(6) Testimony.

a. Mark Bartel & Dr. Charles Geschke

Mark Bartel and Dr. Charles Geschke are former Adobe employees whom TecSec identified in its supplemental responses to Adobe's interrogatories as "individuals knowledgeable concerning the factual and legal bases of TecSec's allegations." Dkt. 1233-7 at 4–5. Both witnesses are beyond the subpoena power of the court and have not been deposed by TecSec. While those facts may impede TecSec's ability to call the witnesses at trial, they do not serve as a basis for preventing TecSec from attempting to do so or from making relevant references to either individual. These facts do, however, require that, in the event the witnesses

do not testify, TecSec be precluded from making any argument that Adobe failed to present the witnesses or that the witnesses' testimony would favor TecSec. Thus, Adobe's Motion in Limine #8 is hereby **GRANTED IN PART, DENIED IN PART** as to Mr. Bartel and Dr. Geschke: TecSec will not be precluded from calling these witnesses at trial, but if the witnesses do not testify, TecSec will be precluded from implying Adobe failed to present the witnesses or that the witnesses' unknown testimony would favor TecSec.

b. John Petty

John Petty has been TecSec's CFO since 1996. Dkt. 1208 at 30. Mr. Petty was repeatedly identified as a person with relevant knowledge in Adobe's disclosures. Dkt. 1233-8 at 10; Dkt. 1233-9 at 10; Dkt. 1233-10 at 10. Mr. Petty has also been deposed in two lengthy depositions in this case that have produced over 600 pages of deposition testimony. Adobe's failure to supplement its initial disclosures with Mr. Petty's name is not fatal to calling him because his identity and knowledge were "made known to the other parties during the discovery process." Fed .R. Civ. Pro. 26(e)(1)(A). TecSec had the opportunity to extensively depose Mr. Petty and therefore would not be prejudiced by allowing Mr. Petty to testify at trial. As such, Adobe's Motion in Limine #8 is **DENIED** as to John Petty.

c. Eunha Ranft

Eunha Ranft, Adobe's Director of Finance, was identified as one of Adobe's Rule 30(b)(6) corporate representatives. Adobe seeks to exclude portions of Ms. Ranft's deposition testimony as cumulative because Adobe executive Julie Morgan was later deposed on the same topics and much of Ms. Ranft's testimony on those topics were non-responsive answers such as "I don't know." TecSec argues Ms. Ranft's testimony is not cumulative because most of the testimony was provided in Ms. Ranft's individual capacity, rather than in her capacity as a corporate representative. The Court has reviewed the identified portions of Ms. Ranft's

deposition testimony and finds that a substantial portion of the selected testimony was made in Ms. Ranft's capacity as corporate representative and contains substantially more information than mere "I don't know" answers. While Ms. Morgan may have been deposed on the same topics, TecSec is entitled to rely on Ms. Ranft's substantive testimony on those topics.

However, the Court is unable to make a determination about whether Ms. Ranft's testimony would be cumulative at this time without knowing what other evidence will be presented to the jury prior to Ms. Ranft's deposition testimony. Thus, the Court **DEFERS** ruling on Adobe's Motion in Limine #8 as to Ms. Ranft until the Court has had the opportunity to hear the testimony at trial and better determine if Ms. Ranft's testimony should be excluded as cumulative.

17. Adobe's Motion in Limine #9: Motion to Preclude Argument, Evidence, or Testimony Conflating the Value of "Security" Generally with the Value of TecSec's Claimed Technology.

The patents-in-suit in this case concern patented encryption technology that enhances security. The value of the patents-in-suit is not equal to value of security generally, but rather only the value of the added security benefits the patents-in-suit provide over other technology lacking the same safeguards. Thus, Adobe's Motion in Limine #9 is hereby **GRANTED IN PART, DENIED IN PART**. TecSec will be precluded from arguing or introducing testimony that *conflates* the value of security generally with the value of TecSec's claimed technology, but TecSec will not be precluded from presenting evidence or argument concerning the value of security generally to explain how TecSec's claimed technology's improvement to security is valuable.

18. Adobe's Motion in Limine #10: Motion to Preclude Argument, Evidence, or Testimony on Adobe's Size and Total Employees.

Adobe's Motion in Limine #9 is hereby **GRANTED IN PART, DENIED IN PART.** TecSec will be precluded from suggesting that a large jury award would be reasonable given Adobe's size and employees because such an argument would be substantially more prejudicial than probative under Federal Rule of Evidence 403 and the Federal Circuit "does not permit the use of a company's overall size or revenue as a check to confirm the reasonableness of a jury award." *HTC Corp. v. Tech. Props. Ltd.*, 2013 WL 4782598, at *6 (N.D. Cal. Sept. 6, 2013) (citing *Uniloc*, 632 F.3d at 1312). Adobe's size and total employees in the United States are, however, relevant to determining the magnitude of Adobe's alleged direct infringement, and TecSec will not be precluded from offering evidence of Adobe's size and total employees for that limited purpose.

19. Adobe's Motion in Limine #11: Motion to Preclude Argument, Evidence, or Testimony on Adobe's Alleged Specific Intent to Induce or Willfully Infringe Based on Conduct After Patent Expiration.

At issue in this final Motion in Limine is a blog post by Adobe through which Adobe allegedly "instruct[ed] its customers and the public at large on how to encrypt documents in a manner that infringes the asserted claims." Dkt. 1234 at 32. The parties do not seem to dispute the admissibility or relevancy of the blog post generally in this Motion in Limine, but instead dispute only TecSec's ability to reference or introduce evidence showing that the blog post is still available today or was available after the patents-in-suit expired. TecSec argues that the blog post's continued availability after the expiration of the patents is evidence of the value of TecSec's claimed technology. The Court agrees and further finds that if the blog post is admissible generally, the mere additional admission of the fact that the blog post is still available

despite the expiration of the patents-in-suit would not be unfairly prejudicial under Federal Rule of Evidence 403. Accordingly, Adobe's Motion in Limine #9 is hereby **DENIED**.

20. Duration of Trial & Timekeeping

The Court hereby **NOTIFIES** the parties that it believes this case can be tried within six (6) trial days evenly divided between the parties. The parties will be expected to keep and manage their own time accordingly.

November 21, 2018
Alexandria, Virginia

Liam O'Grady
United States District Judge

IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division

TECSEC, INCORPORATED, *Plaintiff*, v. ADOBE INC., *et al*., *Defendants*.	Civil Action No. 1:10-cv-115 Hon. Liam O'Grady

ORDER

This matter comes before the Court on Plaintiff TecSec's Motion for Reconsideration of Court's Ruling on Adobe's Motion in Limine No. 1 (Dkt. 1307). The Court reviewed the pleadings and the related portions of the record, and heard oral argument on December 7, 2018. For the reasons stated from the bench, and for good cause shown, TecSec's Motion for Reconsideration of Court's Ruling on Adobe's Motion in Limine No. 1 (Dkt. 1307) is **DENIED**.

It is **SO ORDERED**.

December 7 2018
Alexandria, Virginia

Liam O'Grady
United States District Judge

IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division

TECSEC, INC., *Plaintiff*, v. ADOBE INC., *et al.*, *Defendants*.	Civil Action No. 1:10-cv-115 Hon. Liam O'Grady

MEMORANDUM OPINION AND ORDER

This matter comes before the Court on Defendant Adobe Inc.'s Renewed Motion for Judgment as a Matter of Law or, in the Alternative, for a Remittitur (Dkt. 1376) and Plaintiff TecSec, Incorporated's Motion for Prejudgment Interest and Post-Judgment Interest (Dkt. 1378). The motions were fully briefed and the Court heard oral argument on May 2, 2019. For the reasons that follow, and for good cause shown, Defendant Adobe Inc.'s Renewed Motion for Judgment as a Matter of Law or, in the Alternative, for a Remittitur (Dkt. 1376) is hereby **GRANTED** to the extent it seeks judgment as a matter of law of no direct infringement damages and **DENIED** in all other respects; and Plaintiff TecSec, Incorporated's Motion for Prejudgment Interest and Post-Judgment Interest (Dkt. 1378) is hereby **DENIED** as moot.

I. BACKGROUND

During a five-day trial, Plaintiff TecSec, Incorporated, sought to prove that Defendant Adobe Inc. directly and indirectly infringed TecSec's patented method and system claims for multi-level multi-media encryption technology. After almost two full days of deliberation, the jury entered a verdict partially in favor of TecSec and partially in favor of Adobe. The jury found that TecSec owned the asserted patents and that the patents were valid. It also found that Adobe

directly infringed all the asserted claims during the relevant damages period, but that Adobe did not induce infringement on any of the asserted claims. The jury awarded TecSec $1.75 million for Adobe's direct infringement.

Adobe has now renewed its motion for judgment as a matter of law that (a) the patents are invalid, (b) TecSec has not proven direct infringement of the method claims during the damages period, and (c) TecSec has not proven direct infringement damages. TecSec has moved for pre- and post-judgment interest.

II. LEGAL STANDARD

Rule 50(b) of the Federal Rules of Civil Procedure allows a party to renew a prior Rule 50(a) motion for judgment as a matter of law ("JMOL") that was denied. "A Rule 50(b) motion should be granted if a district court determines, without weighing the evidence or considering the credibility of the witnesses, that substantial evidence does not support the jury's findings." *Konkel v. Bob Evans Farms Inc.*, 165 F.3d 275, 279 (4th Cir. 1999). "[T]he question is whether a jury, viewing the evidence in the light most favorable to [the prevailing party], 'could have properly reached the conclusion reached by this jury.' If reasonable minds could differ about the result in this case, [the Court] must affirm the jury's verdict." *Bryant v. Aiken Reg'l Med. Ctrs. Inc.*, 333 F.3d 536, 543 (4th Cir. 2003) (internal citation omitted).

III. ANALYSIS

A. The Jury's Finding that the Patents-in-Suit Are Valid.

Adobe has renewed its JMOL motion for a finding that the patents-in-suit are invalid. Issued patents are presumed valid, and that presumption can be overcome only through clear and convincing evidence that the patents are invalid. 35 U.S.C. § 282(a); *Microsoft Corp. v. i4i Ltd. P'ship*, 564 U.S. 91, 95 (2011). "At trial, the jury heard conflicting [expert] testimony" as to whether the patents-in-suit were invalid. *i4i Ltd. P'ship v. Microsoft Corp.*, 598 F.3d 831, 848

(Fed. Cir. 2010), *aff'd*, 564 U.S. 91 (2011). The jury's finding of validity had evidentiary support in the record and, "[i]n evaluating the evidence, the jury was free to disbelieve" Adobe's expert, Dr. Clark, and instead credit TecSec's expert, Dr. Jones. *Id.* Further, the jury's validity verdict was not tainted by TecSec's argument that the fact testimony of Mr. Mundy and Mr. Balenson lacked corroboration because corroboration, or lack thereof, is always a relevant factor the jury may consider in evaluating the credibility of a witness's testimony. Therefore, the Court will not overturn the jury's verdict that the patents-in-suit are valid.

B. The Jury's Finding that Adobe Directly Infringed the Asserted Method Claims During the Relevant Damages Period.

The Court will also not overturn the jury's verdict that Adobe directly infringed the asserted method claims during the relevant damages period, which ran from February 5, 2010 through March 3, 2011. A patentee may prove direct infringement during the relevant damages period through circumstantial evidence. *Liquid Dynamics Corp. v. Vaughan Co.*, 449 F.3d 1209, 1219 (Fed. Cir. 2006). At trial, TecSec introduced sufficient circumstantial evidence for the jury to have reasonably found that Adobe directly infringed the asserted method claims during the damages period.

Mr. Landwehr, Adobe's public sector chief technology officer, testified that he became interested in security at a young age and "[p]retty much [his] entire career" in computer science "has had some aspects intersecting with security." Tr. 755:7–23. Mr. Landwehr also testified that he is a Certified Information Systems Security Professional who frequently discusses security issues with customers, has twice testified before Congress on security topics while at Adobe, and has written "pretty much all" of his approximately one hundred Adobe blog posts on security issues. *Id.* at 756:2–14, 784:8–21. Mr. Landwehr described how to perform multi-level encryption using Adobe Acrobat in a 2009 blog post, which remained posted even at the time of

trial, and admitted that he had performed multi-level encryption in preparing that blog post. *Id.* at 787:2–788:7.

A reasonable jury could have inferred that Mr. Landwehr performed the asserted method claims at least once during the relevant damages period because he was a security specialist who promoted the security features of Adobe Acrobat and assisted customers in using those features, he admitted performing the patented methods the year before the damages period, and he taught customers how to perform the patented methods in his 2009 blog post, which was live for the duration of the damages period. *See Mirror Worlds, LLC v. Apple, Inc.*, 784 F. Supp. 2d 703, 715 (E.D. Tex. 2011) (granting JMOL of no direct infringement where the plaintiff failed to "introduce circumstantial evidence that would support a finding of [the defendant's] direct infringement," such as "evidence of [the defendant's] employees testing the method steps" or "documents containing instructions that teach all of the steps of the method claims"); *cf. Lucent Techs., Inc. v. Gateway, Inc.*, 580 F.3d 1301, 1317–19 (Fed. Cir. 2009) (holding that the jury could have reasonably found direct infringement by the defendant's customers from the circumstantial evidence of sales of the defendant's product with manuals teaching customers how to perform the method claims).

As a result, the Court will not overturn the jury's verdict on direct infringement.

C. The Jury's Award of $1.75 Million for Direct Infringement Damages.

While there was sufficient evidence for the jury to find that Adobe directly infringed during the relevant damages period, there was insufficient evidence to support the $1.75 million damages award for that direct infringement.

"Patent owners who prove infringement are typically awarded at least some amount of damages." *Promega Corp. v. Life Techs. Corp.*, 875 F.3d 651, 666 (Fed. Cir. 2017). Indeed, 35 U.S.C. § 284 states that upon finding infringement, "the court *shall* award" the plaintiff

"damages adequate to compensate for the infringement, but in no event less than a reasonable royalty for the use made of the invention by the infringer . . . ." 35 U.S.C. § 284 (2012) (emphasis added).

Yet, "the burden of proving damages falls on the patentee, and the patentee must show his damages by evidence." *Promega*, 875 F.3d at 660 (alterations, internal quotations, and citations omitted). "Damages 'must not be left to conjecture by the jury. They must be proved, and not guessed at.'" *Id.* (quoting *Philp v. Nock*, 84 U.S. 460, 462 (1873)). Thus, 35 U.S.C. § 284 does not require the award of a reasonable royalty "even in the absence of any evidence from which the court," or the jury, "may derive a reasonable royalty." *Devex Corp. v. Gen. Motors Corp.*, 667 F.2d 347, 363 (3d Cir. 1981) (cited with approval in *Promega*, 875 F.3d at 666). "In the absence of any evidence as to what would constitute a reasonable royalty in a given case, a fact finder would have no means of arriving at a reasonable royalty, and none could be awarded." *Id.* at 361; *see also Apple Inc. v. Motorola, Inc.*, 757 F.3d 1286, 1328 (Fed. Cir. 2014) ("[I]n a case completely lacking any evidence on which to base a damages award, the record may well support a zero royalty award."); *Gustafson, Inc. v. Intersys. Indus. Prods., Inc.*, 897 F.2d 508, 509–10 (Fed. Cir. 1990) (finding "no reversible error in the district court's . . . awarding no damages to [the plaintiff] because none were proven"); *Mirror Worlds*, 784 F. Supp. 2d at 724–27 (vacating the jury's direct infringement damages award because the damages award was not sufficiently supported by the record); *Keg Techs., Inc. v. Laimer*, 436 F. Supp. 2d 1364, 1370 (N.D. Ga. 2006) (denying the plaintiffs a monetary damages award because any award "would be an exercise in arbitrary speculation").

A patent owner is also "not entitled to any damages" when it "deliberately takes a risk by relying at trial exclusively on a damages theory that ultimately proves unsuccessful." *Promega*, 875 F.3d at 666. In *Promega*, the Federal Circuit held that a patent owner who sought damages at

trial based solely on lost profits for total worldwide sales, and proved infringement for only a subset of those sales, "waived its right to any award based on a reasonable royalty" or "any subset of total worldwide sales." *Id.* at 660, 662–63. It follows that a patent owner who pursues an all-or-nothing damages theory based on sales to customers and fails to present evidence of a reasonable royalty based solely on the defendant's own use, waives its right to recover direct infringement damages. *Mirror Worlds*, 784 F. Supp. 2d at 724–25 (vacating a jury's damages award where the plaintiff was only entitled to recover direct infringement damages, but the plaintiff's damages theory was based solely on the defendant's sales to customers and the plaintiff did not present any evidence of a reasonable royalty based solely on the defendant's own use of the patented technology).

At trial, TecSec pursued a damages theory tied solely to Adobe's sales of Adobe Acrobat to customers. Because Adobe Acrobat does not infringe TecSec's system claims until it is installed, the testimony of a reasonable royalty based on sales to customers alone cannot support any award of direct infringement damages.

In denying Adobe's earlier JMOL motion regarding failure to prove direct infringement damages, the Court opined that "while there was no direct testimony on direct infringement damages, the jury should be given the opportunity to determine whether" the testimony of TecSec's damages expert, Mr. Wagner, "could also serve as a basis to find direct infringement damages." Dkt. 1373 at 2. The Court believed that there may have been evidence adduced at trial that would have allowed the jury to apply a reasonable royalty rate derived from the damages experts' testimony to a royalty base tied solely to Adobe's own use of Adobe Acrobat. Now that the Court has had the opportunity to review the record and the more thorough arguments of counsel, the Court finds that the damages experts' testimony could not have served as a basis for an award of direct infringement damages as a matter of law.

For the jury to have non-speculatively found direct infringement damages based on the damages experts' testimony at trial, TecSec would have needed to introduce evidence of a royalty base tied solely to Adobe's own use of Adobe Acrobat against which the jury could have applied a reasonable royalty rate derived from the experts' testimony. But there was no evidence at trial to establish such a royalty base. For example, there was no testimony regarding the number of Adobe employees in the United States or the number of times Adobe Acrobat was installed by Adobe employees during the damages period. As a result, the jury could not have relied on the damages experts' testimony to arrive at $1.75 million in direct infringement damages without impermissible speculation.

There is also insufficient evidence in the record to support remitting the jury's damages award to $200,000 – the maximum cost Adobe would have incurred to design around the asserted patents in April 2001, the date of the hypothetical negotiation between Adobe and TecSec. As an initial matter, "damages awarded for patent infringement 'must reflect the value attributable to the infringing features of the product, and no more.'" *Commonwealth Sci. & Indus. Research Org. v. Cisco Sys., Inc.*, 809 F.3d 1295, 1301 (Fed. Cir. 2015) (quoting *Ericsson, Inc. v. D-Link Sys., Inc.*, 773 F.3d 1201, 1226 (Fed. Cir. 2014)). The cost to design around TecSec's patents is not tied to the value of TecSec's invention.[1] Furthermore, even if the $200,000 cost to design around the asserted patents could support a damages award, it cannot support an award for direct infringement damages alone because any apportionment of the cost between Adobe's own use of Adobe Acrobat and Adobe's sales of Adobe Acrobat to customers would be inherently speculative.

[1] Indeed, the evidence adduced at trial demonstrated, at most, a *de minimis* value of TecSec's invention to Adobe; there was only circumstantial evidence that one or two Adobe employees even knew the infringing capability existed prior to this litigation.

The only other evidence of an appropriate amount of direct infringement damages is Dr. Jones's testimony that an award of "a couple hundred dollars . . . would seem reasonable" to compensate TecSec for Adobe's direct infringement of the asserted method claims during the damages period. Tr. at 547:14–25. Yet, Dr. Jones's opinion regarding a reasonable damages award for Adobe's direct infringement during the relevant damages period was also inherently speculative. Dr. Jones twice stated that he "would have to defer to the damages experts" regarding a reasonable award for Adobe's direct infringement of the asserted method claims. *Id.* at 547:24–25; *see also id.* at 546:3. Dr. Jones also stated that he does not know how damages are calculated, and when asked if the proper damages award could be lower than a couple hundred dollars, he responded: "It could be, I just don't know." *Id.* at 546:3–6. As a result, the record is also devoid of qualified testimony regarding appropriate damages for Adobe's direct infringement of the asserted method claims during the relevant damages period.

Accordingly, the Court finds that the jury's damages award of $1.75 million must be vacated in its entirety as inherently speculative and unsupported by the record.

The Court also finds that TecSec is not entitled to a new trial to establish direct infringement damages above zero dollars. "[R]e-presenting this issue to the jury would [be] pointless because, as a matter of law," an award of zero damages is the maximum award supported by the evidence adduced at trial. *Tronzo v. Biomet, Inc.*, 236 F.3d 1342, 1351 (Fed. Cir. 2001).[2] "When a plaintiff deliberately takes a risk by relying at trial exclusively on a damages theory that ultimately proves unsuccessful, and, when challenged, does not dispute that it failed to present an alternative case for damages, a district court does not abuse its discretion

[2] The Fourth Circuit and Federal Circuit both have "a 'maximum recovery rule' which remits an excessive jury award to the highest amount the jury could 'properly have awarded based on the relevant evidence.'" *Shockley v. Arcan, Inc.*, 248 F.3d 1349, 1362 (Fed. Cir. 2001) (quoting *Unisplay, S.A. v. Am. Elec. Sign Co.*, 69 F.3d 512, 519 (Fed. Cir. 1995)).

by declining to give that plaintiff multiple chances to correct deficiencies in its arguments or the record." *Promega*, 875 F.3d at 666.

D. TecSec's Motion for Pre- and Post-Judgment Interest.

Having found that the jury's award of damages must be overturned in its entirety, TecSec's motion for interest on the damages award is moot.

IV. CONCLUSION

For the reasons stated above, and for good cause shown, Defendant Adobe Inc.'s Renewed Motion for Judgment as a Matter of Law or, in the Alternative, for a Remittitur (Dkt. 1376) is hereby **GRANTED** to the extent it seeks judgment as a matter of law of no direct infringement damages and **DENIED** in all other respects. Accordingly, the jury's award of damages is hereby **VACATED**.

It is further **ORDERED** that Plaintiff TecSec, Incorporated's Motion for Prejudgment Interest and Post-Judgment Interest (Dkt. 1378) is hereby **DENIED** as moot.

It is **SO ORDERED**.

June 18 2019
Alexandria, Virginia

Liam O'Grady
United States District Judge

U 7214101

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:
UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

November 25, 2009

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,369,702*
ISSUE DATE: *November 29, 1994*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

P. SWAIN
Certifying Officer

UNITED STATES PATENT AND TRADEMARK OFFICE · DEPARTMENT OF COMMERCE

Plaintiff's Trial Exhibit
1
TecSec. v. Adobe Case no. 1:10cv115

TEC00140757

US005369702A

United States Patent [19]

Shanton

[11] Patent Number: 5,369,702

[45] Date of Patent: Nov. 29, 1994

[54] **DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD**

[75] Inventor: **M. Greg Shanton**, Fairfax, Va.

[73] Assignee: **TECSEC Incorporated**, Vienna, Va.

[21] Appl. No.: **138,857**

[22] Filed: **Oct. 18, 1993**

[51] **Int. Cl.**[5] **H04L 9/00**

[52] **U.S. Cl.** **380/4**; 380/9; 380/21; 380/23; 380/25; 380/28; 380/49; 380/50; 340/825.31; 340/825.34

[58] **Field of Search** 380/4, 9, 21, 23, 25, 380/28, 30, 43, 49, 50; 340/825.31, 825.34

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,218,582	8/1980	Hellman et al.	380/30
4,405,829	9/1983	Rivest et al.	380/30
4,424,414	1/1984	Hellman et al.	380/30
4,713,753	12/1987	Boebert et al.	380/4 X
4,864,616	9/1989	Pond et al.	380/25
4,955,082	9/1990	Hattori et al.	455/33.3
4,962,533	10/1990	Krueger et al.	380/25
4,984,272	1/1991	McIlroy et al.	380/25
5,052,040	9/1991	Preston et al.	380/4
5,204,961	4/1993	Barlow	380/4 X

OTHER PUBLICATIONS

Massey, "An Introduction to Contemporary Cryptology," Proceeding of the IEEE, May 1988, pp. 533–549.

Schneier, "Untangling Public-Key Cryptography," Dr. Dobb's Journal, May 1992, pp. 16–28.

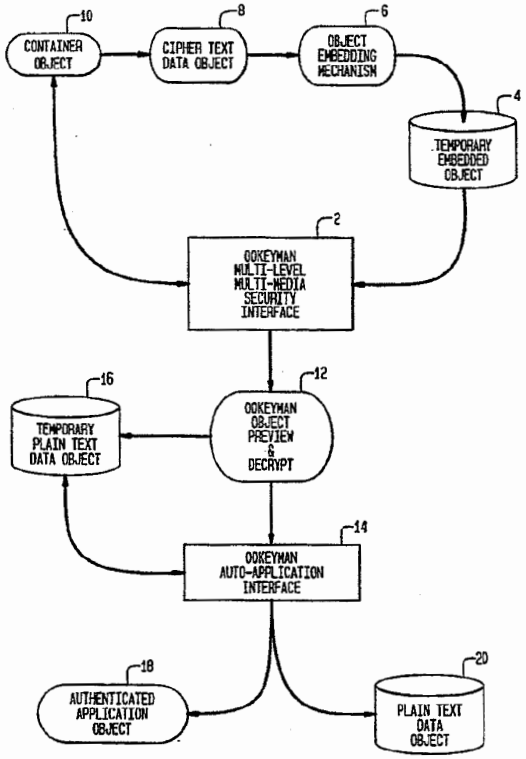
Primary Examiner—Bernarr E. Gregory

Attorney, Agent, or Firm—Thomas M. Champagne; Jon L. Roberts

[57] **ABSTRACT**

A system for increasing the security of a computer system, while giving an individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By effecting compartmentalization of every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

15 Claims, 8 Drawing Sheets



Copy provided by USPTO from the PIRS Image Database on 11/24/2009

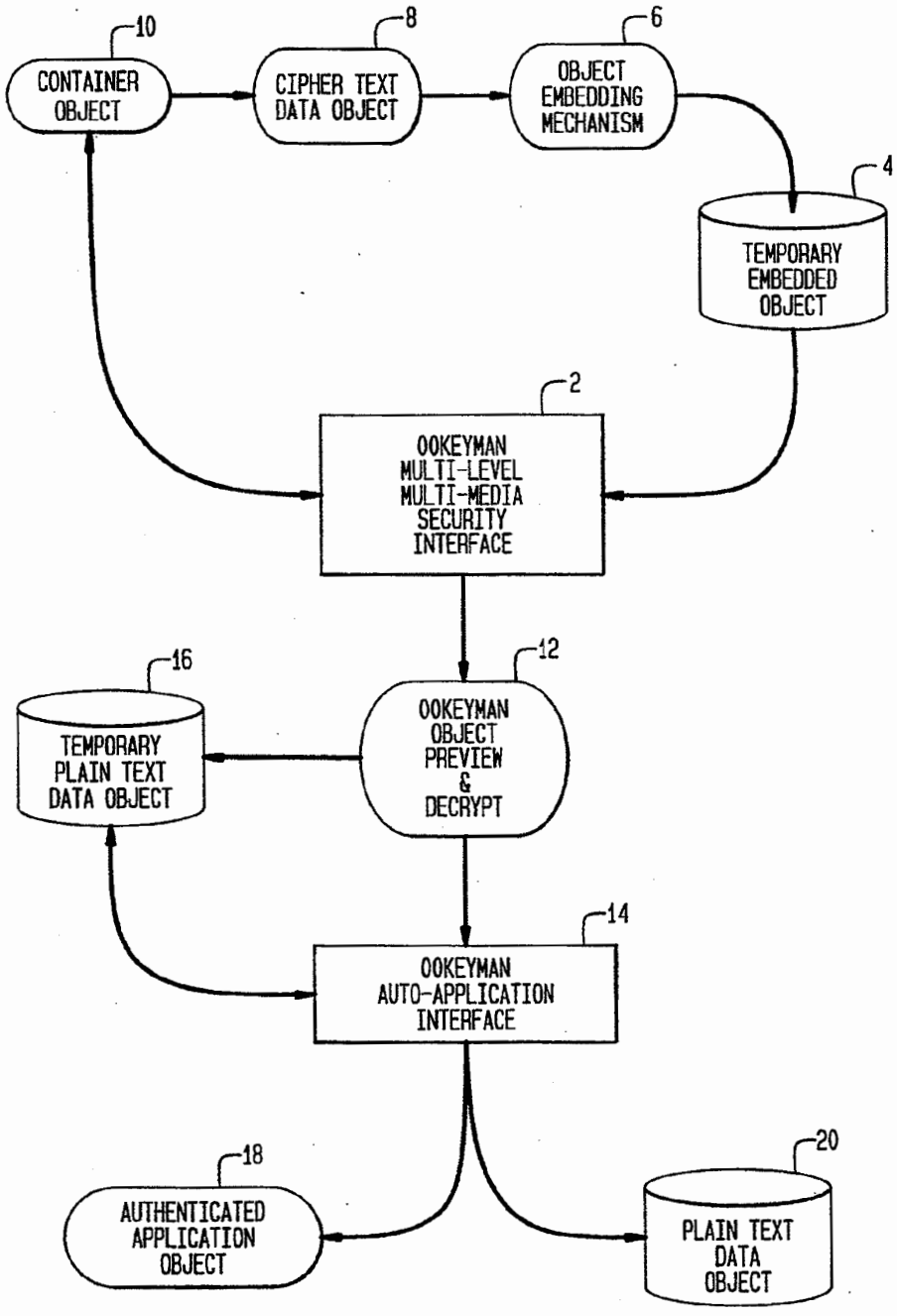


FIG. 1

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

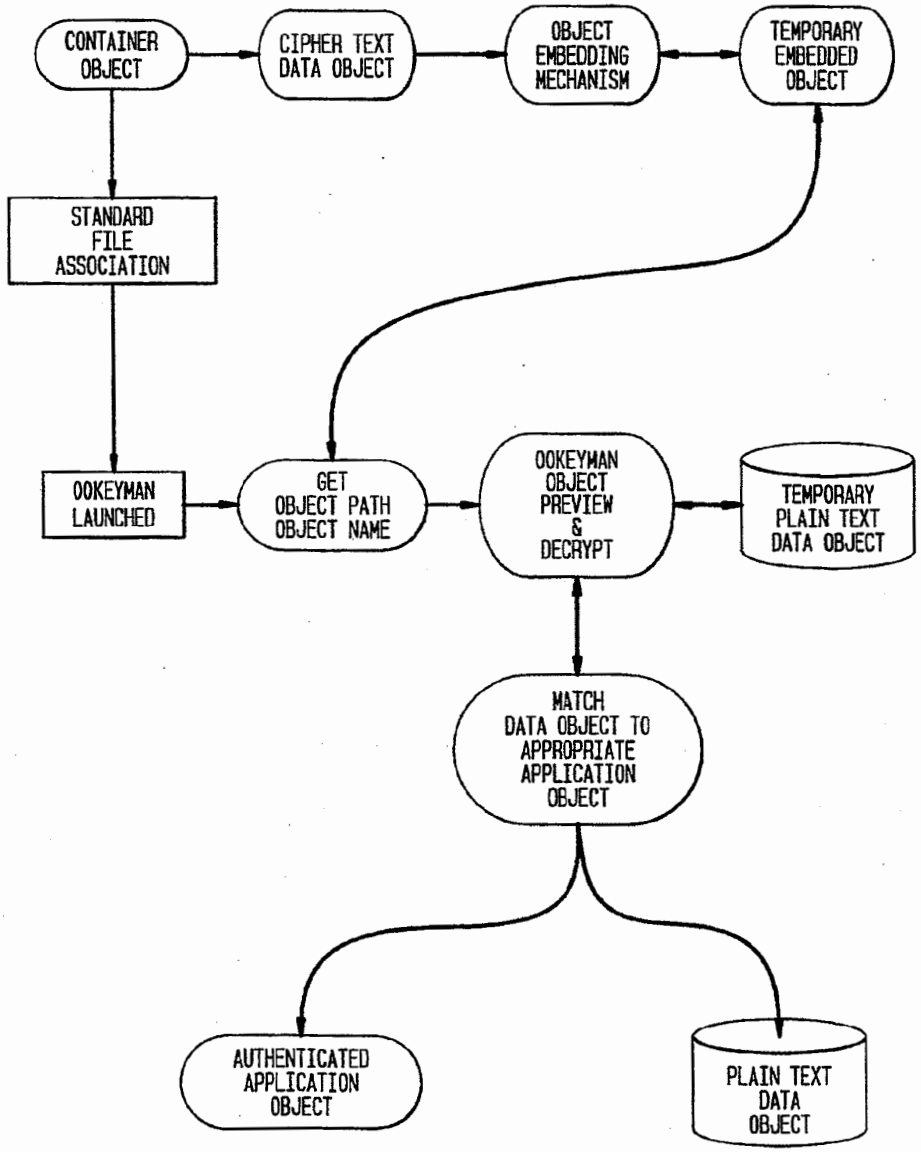


FIG. 2

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

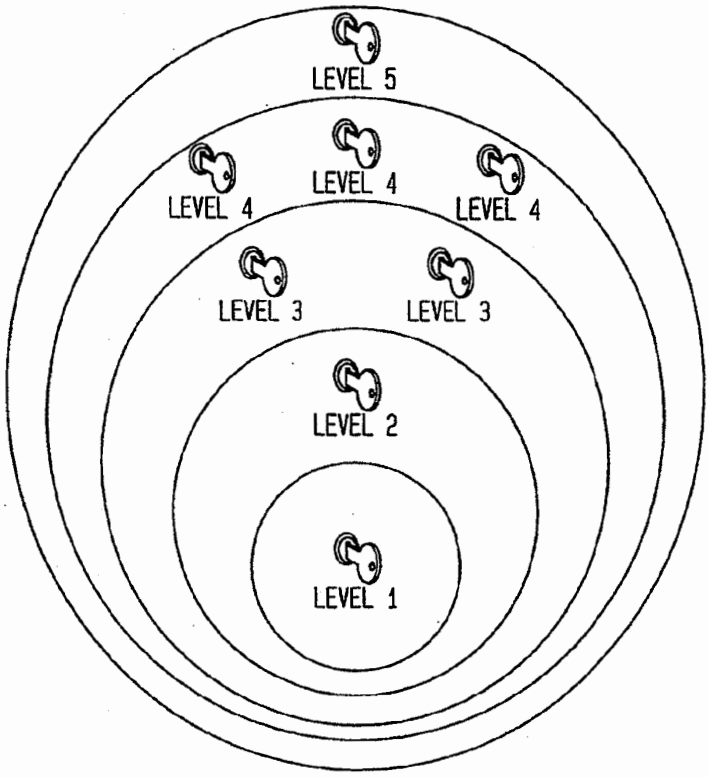


FIG. 3

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

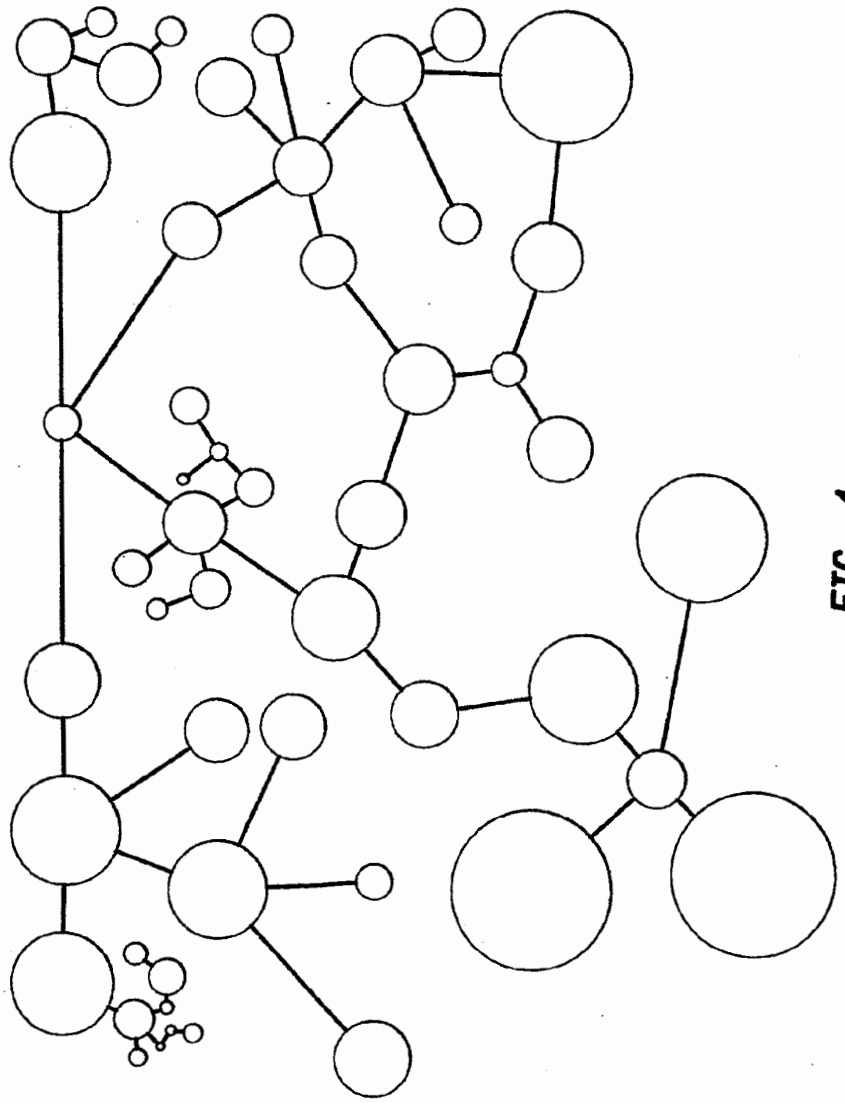


FIG. 4

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 119 Filed: 10/07/2019

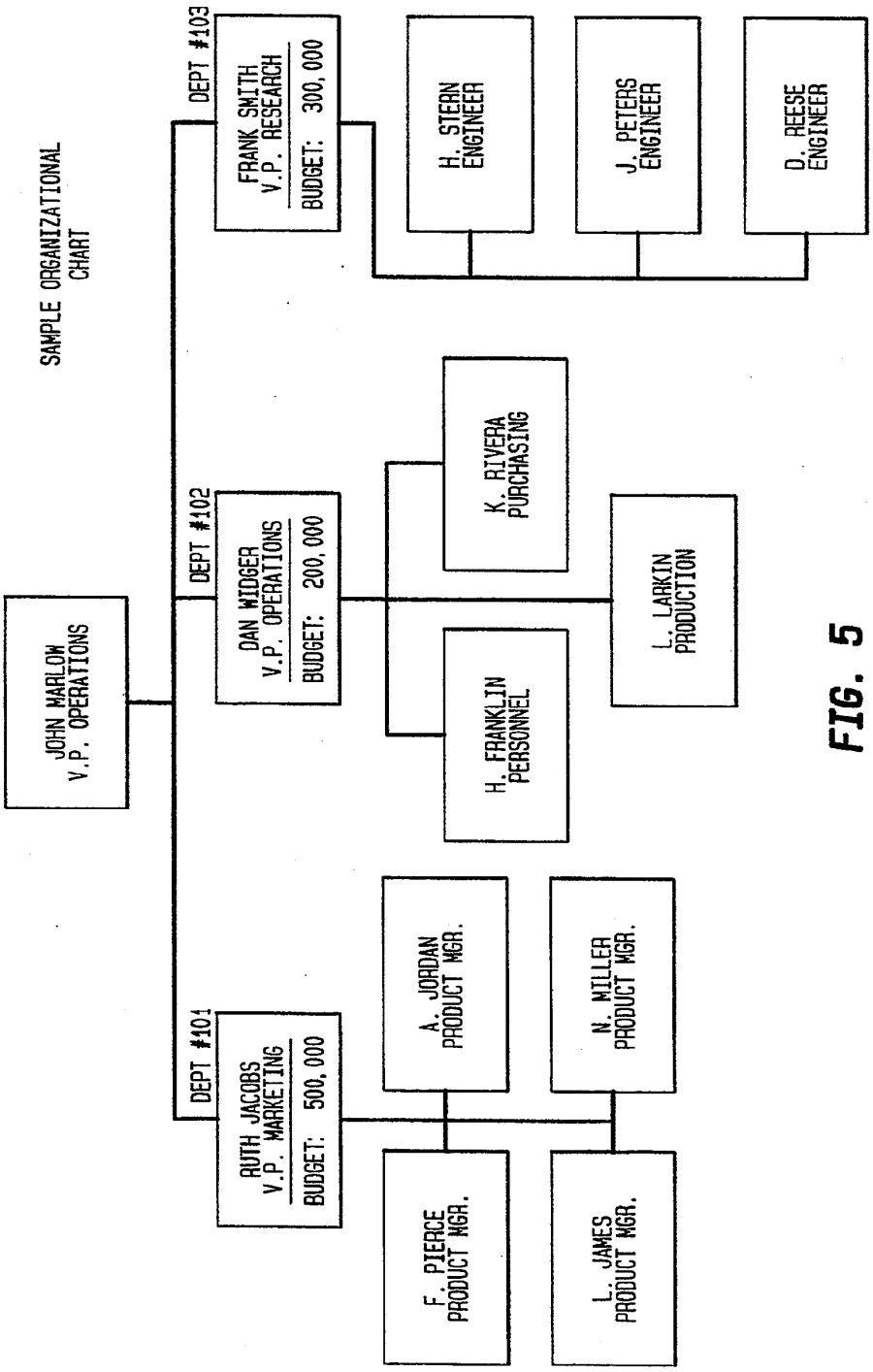


FIG. 5

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

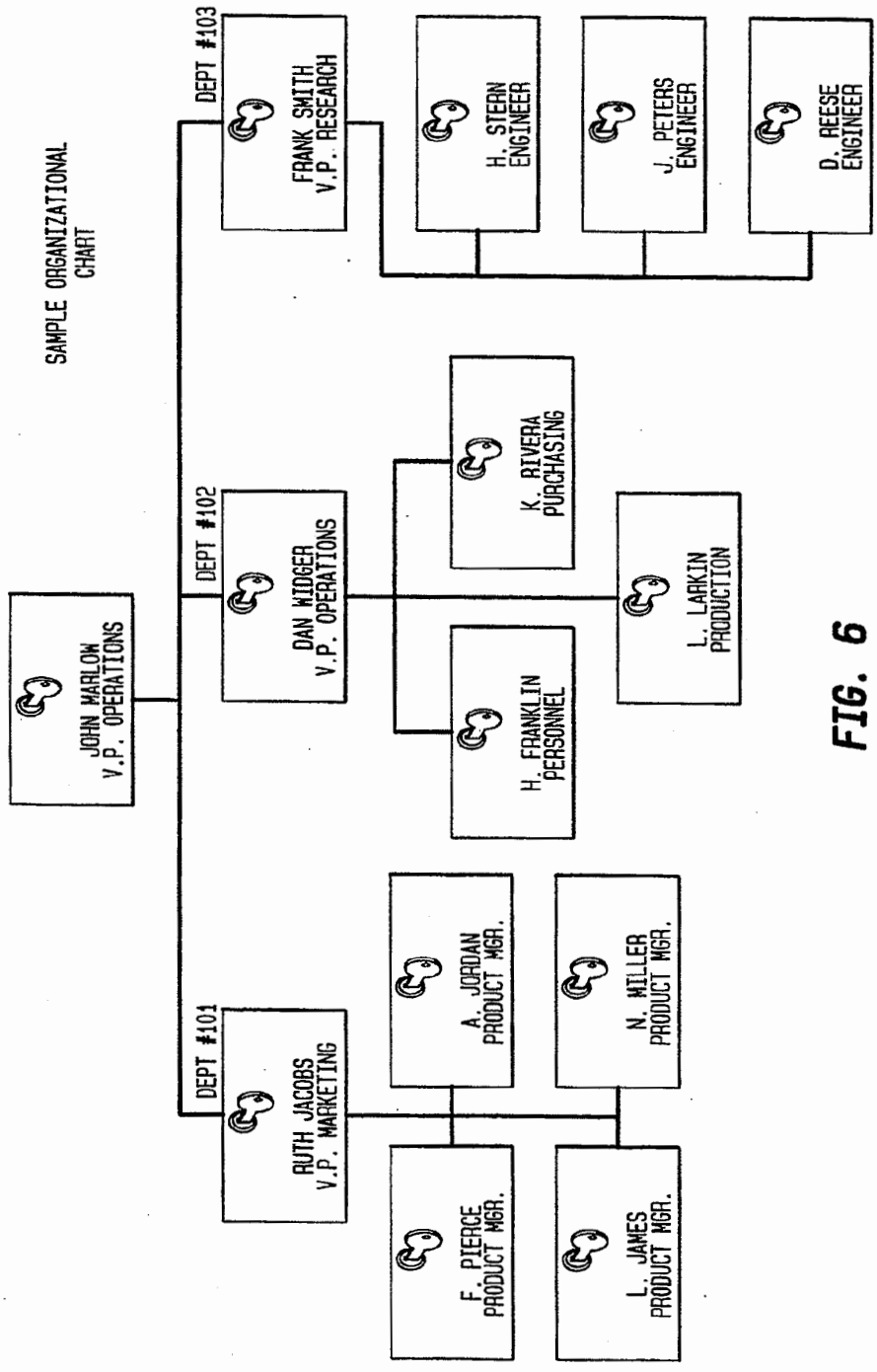


FIG. 6

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 120 Filed: 10/07/2019

Case: 19-2192 Document: 20 Page: 121 Filed: 10/07/2019

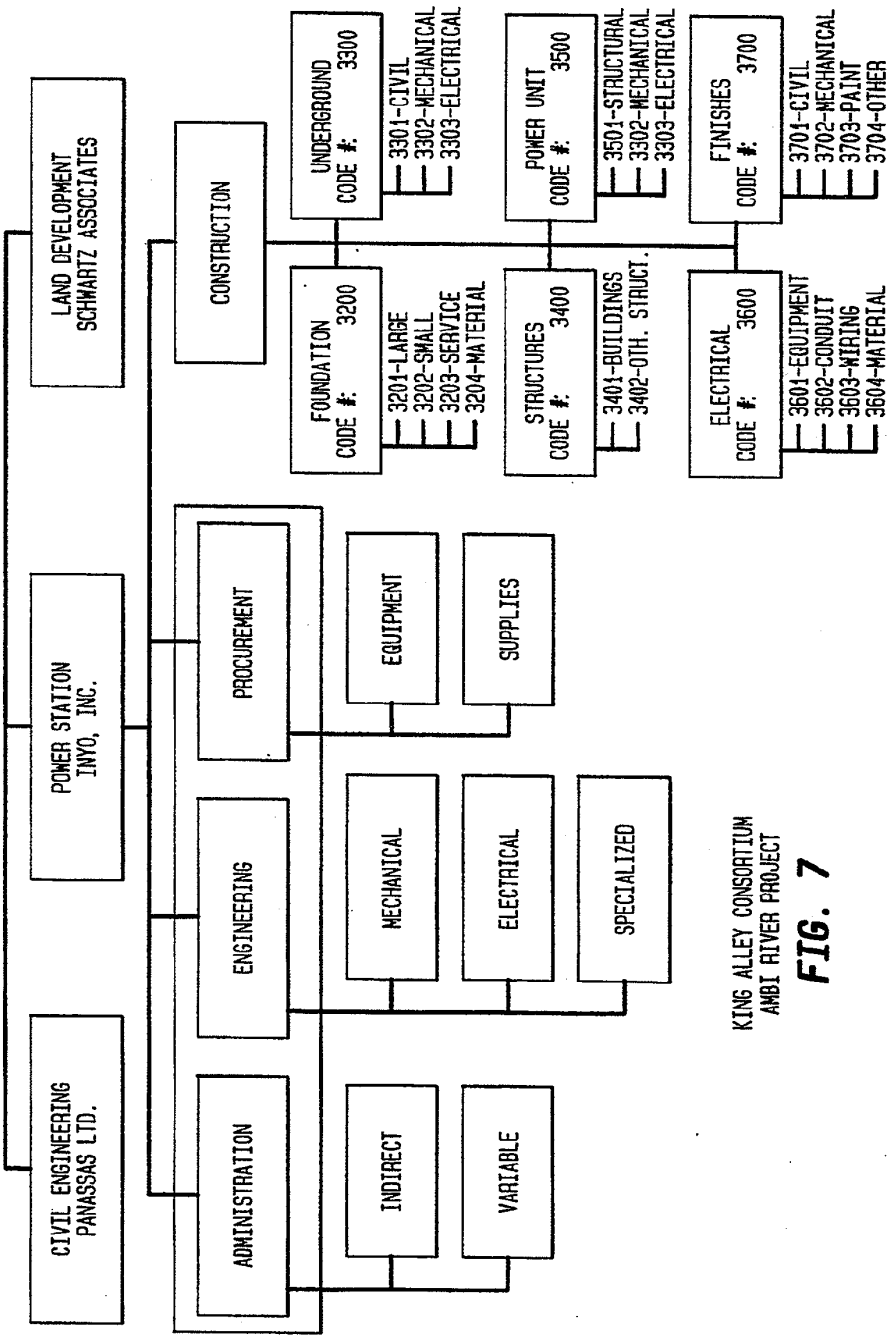


FIG. 7

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

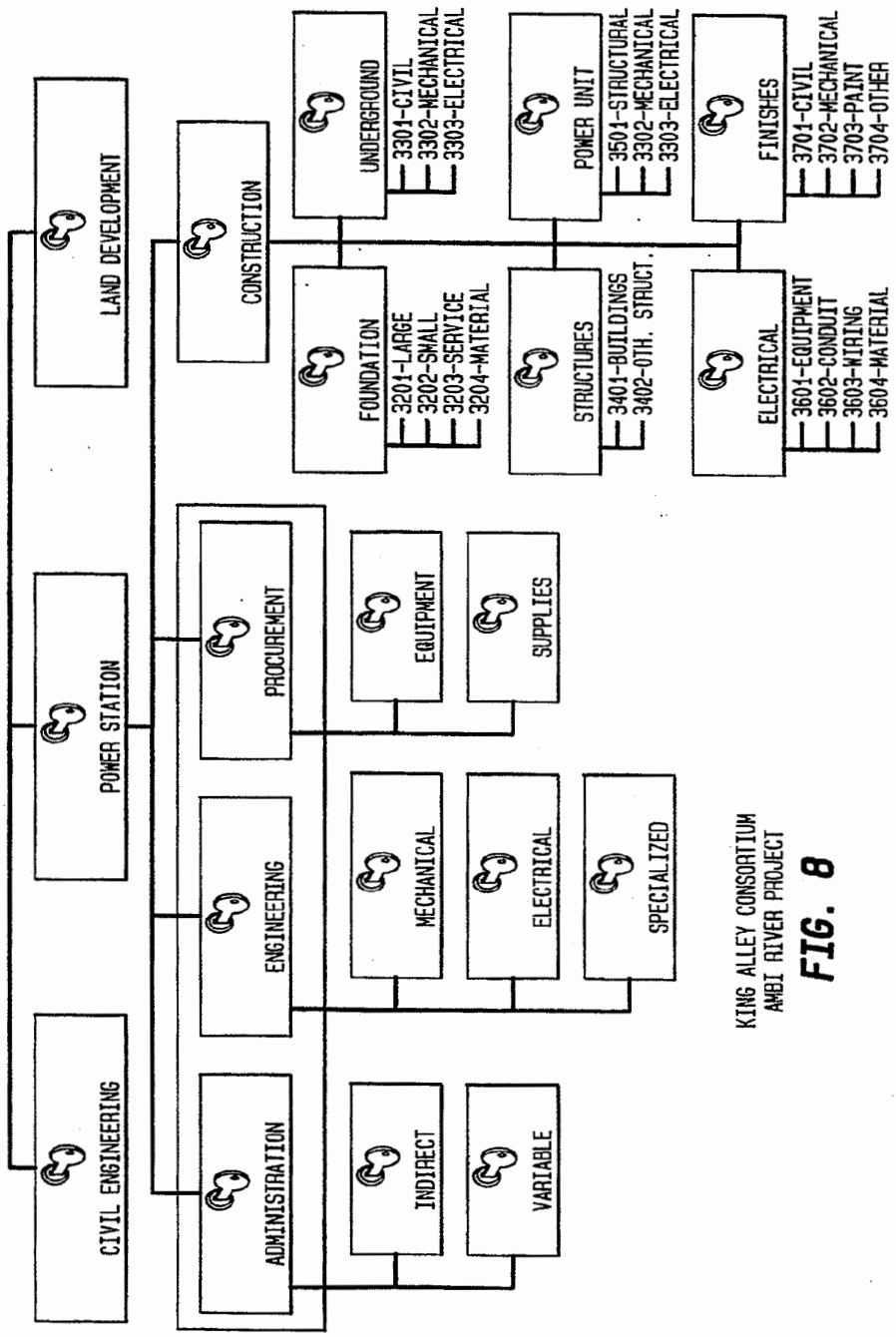


FIG. 8

Case: 19-2192 Document: 20 Page: 122 Filed: 10/07/2019

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD

FIELD OF THE INVENTION

The present invention relates generally to a system that can be used to restrict access to computer data. In particular, the system of the present invention restricts access in a flexible way, identifying objects for restriction and nesting restriction requirements through the use of embedded objects.

BACKGROUND OF THE INVENTION

While the specter of "spies" eagerly trying to obtain the defense information of various countries is very much still present in the defense and intelligence community, an equally massive threat now exists from technological or commercial "spies" who desire to obtain commercial and technical information from competing companies. These agents use sophisticated means similar to those used by the defense and intelligence community in order to obtain commercially valuable information that reveals the plans and commercial activities of competitors thereby allowing the aggressor company to obtain a competitive advantage in the marketplace. Theft of commercially valuable information is a very real and ever present threat.

To combat this type of commercial spying, various complex systems have evolved to protect company proprietary information. These systems involve physical controls over personnel as well as over the data flowing in and out of a company. For example, most computer systems used within companies require a password to be entered before the system can be accessed. It is frequently the case that confidential or company proprietary information must be passed electronically from one location to another in order to convey that information within the company in a timely fashion. Such electronic communication is easily susceptible to interception if not protected in some other form.

Cryptographic systems have evolved to fill the needs of companies and individuals wanting to protect the proprietary commercial information of a company from competitors and those who generally should not have that information.

Encryption of data is therefore a critical requirement in denying access to confidential information from those who are not so authorized. Cryptographic "keys" are an essential part of the information encryption process. The cryptographic key, or "key" for short, is a sequence of letters, numbers, or bytes of information which are manipulated by a cryptographic algorithm to transform data from plain (readable) text to a series of unintelligible text or signals known as encrypted or cipher text. The key is then used by the receiver of the cipher text to decrypt the message back to plain text. However, for two people to communicate successfully using keys, each must use the same key, assuming that the same encryption/decryption algorithm is used on both ends of the communication.

Various methods have evolved to manage the distribution of keys. Such methods of distribution are collectively referred to as "key management". The function of key management is to perform the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys. Under normal operational circumstances, the key manager begins and ends a cryptographic session by controlling access to the algorithm used to encrypt and decrypt plain text objects. Thus, a user who wants to encrypt an object or decrypt an object must first access the key manager so that an encryption algorithm may be chosen.

Simple encryption of data being communicated between two points only provides one level of security, however. Encryption limits data communication to those who have the key. Anyone who has the key is privy to any communication at any location. That is, if a group of people are working on a particular project, they will all presumably share a key for decrypting information relating to the project. Some of the project group may be working in one location, while the rest of the group may be located in a distant city. If one member of the group wants to send a communication to a particular member in the other city, the key will afford him no protection because everyone in the project shares the same key. Likewise, if someone wants to communicate a message to a subset of the group, for example, only to management personnel, this key would again provide her with no extra security. In another case, someone may want to send a message that is capable of being read only at a particular computer terminal, or of being printed only at a particular printer. In these and other cases, multilevel multimedia key access, or individual keys issued to each person, would provide a solution, albeit one that is quite unwieldy, inflexible, and difficult to manage by a security officer or key administrator.

A secure method of labelling files or messages that are sent from a sending user to a receiving user over a network can provide a level of protection in addition to cryptographic protection. A file "label" for purposes of this invention means a series of letters or numbers, which may or may not be encrypted, separate from but associated with the sending of a message, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated message. Using a secure labelling regimen, a network manager or user can be assured that only those messages meant for a certain person, group of persons, and/or location(s) are in fact received, decrypted, and read by the intended receiver. Thus, a sending user can specify label conditions that limit access to the transmitted message. For example, many people within a company may have the key necessary to read a data file that a sender may transmit from his computer terminal to other terminals at another site within his company. The sender may, however, wish to restrict reception to those persons present at a particular terminal. By employing a secure labelling technique in addition to encryption, the sender can be assured that people having the correct key to decrypt the message but working at different terminals will not receive or be allowed to access the communication. Access may be limited to particular people as well.

A system such as that described above is disclosed in U.S. patent application Ser. No. 08/009,741, filed Jan. 27, 1993, the specification of which is incorporated by reference herein.

A system that can limit access on an object level would be more flexible and would offer still more protection. Access could be specified on an object-by-object basis, and objects could be embedded within other objects, providing an access hierarchy for users.

The ability to cryptographically secure objects ensures the authentication and data integrity of the partic-

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

ular object or objects in question. If a device were able to cryptographically control an object(s) or nested object(s), then that device would have total control over the entire object and all other objects within it. This type of control over the knowledge/information flow would allow for clear data separation, and at some levels could become a transparent method. A system that is able to do this would be able to achieve multi-level multimedia security.

SUMMARY OF THE INVENTION

It is therefore an objective of the present invention to provide a system to insure that properly specified kinds of information in a network system flows only to designated locations and to further insure that such information is only read by those individuals who are designated to review that information.

It is a further objective of the present invention to provide a system that recognizes objects and permits or denies access on the object level.

It is an additional objective of the present invention to provide a system in which objects may be embedded within other objects, resulting in an access hierarchy for users of the system.

It is another objective of the present invention to provide a system in which access control is transparent to the user.

These and other objectives and advantages of the present invention will be apparent to those of ordinary skill in the art upon inspection of the detailed description, drawings, and appended claims.

The definition and concept of objects varies greatly depending on with whom you consult. Everything around you in your daily life is an object. Your car, your car keys, books, people, desks, etc. Objects are entities by themselves, but they may contain other objects, in either single or multiple configurations. Objects can change their make up dynamically by inheritance. Objects can inherit the attributes of other objects and the inheritance features can change dynamically "on the fly" during the operation of the objects.

In the context of the present invention, an object can come in a vast number of forms, shapes or sizes and can be either passive or active, dynamic or static. An object may stay dormant until it is acted upon, or it may be an active participant, dynamically auditing and verifying every transaction that occurs in a system. Examples of what an object can be include a bit of information, a byte of information, Sound Clips, Video Clips, Graphic Images, text, charts, tables, forms, controls, MDI-Forms, variables, executable files, video files, binary files, text files, data files, container files, graphic files, application file(s), Library files, a directory, a collection of directories, a hard disk, multiple hard disks, any hardware component, any software component, a complete computer system, a single network, multiple networks.

Thus, an object is any distinct, separate entity. In a computer or data communication context, entities that may be treated as objects include:

1) Program objects, representing applications such as word processors, spreadsheets, games, etc., as well as utilities and operating systems;
2) Folder objects, representing collections of other objects;
3) Data file objects, including information such as text, memos, letters, spreadsheets, video, and sound; and
4) Device objects, such as printers, fax modems, plotters, and CD-ROM drives.

In object linking and embedding, an object can be any user-selected group of data, such as a block of test, a set of spreadsheet cells, a chart, sounds, or a graphical image. This data can be embedded in or linked to another document created by a different application. For example, a folder may represent a directory and may contain a group of files, or it may represent a group of programs. Folders can also contain other folders.

In object-oriented programming, a program consists of a set of related but self-contained objects that can contain both code and data.

The present invention is able to increase the security of the system, while at the same time giving the individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By being able to compartment every object by label attributes and algorithm attributes, multi-level multimedia security is achieved. Multi-level security is achieved because encrypted objects may be nested within other objects which are also encrypted, possibly within other objects, resulting in multiple layers of encryption. Multimedia security is achieved because objects are encrypted. Where other encryption systems encrypt only files or other data, the system of the present invention encrypts any object, encompassing all forms of media. Thus, the nesting of individually encrypted objects provides security that is multi-level and multimedia.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 shows a block diagram of the system of the present invention.

FIG. 2 shows a block diagram of the system of the present invention when an embedded encrypted object is activated.

FIG. 3 shows an object containing ten embedded encrypted objects at five various levels.

FIG. 4 shows an encrypted object that contains a web of embedded encrypted objects nested within it.

FIG. 5 shows a sample organizational chart.

FIG. 6 shows the present invention used in conjunction with the dynamic structure of a sample organizational chart.

FIG. 7 shows a larger, more complicated sample organizational chart.

FIG. 8 shows the present invention used in conjunction with the dynamic structure of the larger, more complicated sample organizational chart.

DETAILED DESCRIPTION OF THE INVENTION

Definitions

OOKeyMan stands for the Object-Oriented Key Manager. OOKeyMan is a Microsoft Windows™ stand alone application.

The Auto Application Interface is an intelligent front end and back end interface between a standard Microsoft Windows 3.1 application and OOKeyMan.

An Encrypted Embedded object is an encrypted OOKeyMan object which can contain a single plain text object that has been encapsulated within the encrypted object, or it can contain an infinite web of encrypted

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

objects matched with plain text objects or other cipher text objects.

A container object is an object that contains other objects. These objects can be either cipher text or plain text. This is the transport vehicle for a standard object mechanism that embeds objects. A non-container object is an object that does not contain other objects.

Multi-Level Multimedia Security is defined as the ability to have simultaneous control over the knowledge/information flow of numerous media formats while allowing for clear data separation. At some levels the multi-level multimedia security becomes transparent. Examples of multi-media objects would include a file that contained two or more of the following: sound objects, video objects, graphic V, text objects, chart objects, table objects, and form objects.

Disclosure

The present invention, known as the Distributed Cryptographic Object Method ("DCOM"), is able to control which objects are visible to a specific user, which object attributes are inherited by other objects, which objects are available for use, and which level of system implementation can become transparent.

The main function of the DCOM is to securely manage and track encrypted objects. The DCOM can securely manage and track a single encrypted object, or it can securely manage and track encrypted objects embedded within other encrypted objects. The capability to securely manage and track encrypted objects within other encrypted objects is only limited by storage space.

Referring to FIG. 1, the DCOM system is described. The DCOM has a standard Multi-Level Security object interface 2 that interfaces with the plain text container object's encrypted embedded object(s) 4. It does this through a standard application 6 that has the capability to embed an object in a container object, such as Microsoft's Object Package for Windows. After the Encrypted object(s) is/are embedded in a standard container object(s) 10 and the container object(s) 10 is/are encrypted, the original encrypted object(s) and the new encrypted container object(s) is/are ready for transport.

The new encrypted object(s) can be easily transported/routed over any network that supports binary travel without modification. The original encrypted objects can be deleted because all information from the original encrypted objects is encapsulated in the embedded encrypted object. All of the nested embedded encrypted objects will appear to a user as a single encrypted object until extracted with a standard object embedding/extracting mechanism through the DCOM process. To activate an embedded encrypted object, the user simply selects the encrypted object to initiate the DCOM process, launching the OOKeyMan application, as shown in FIG. 2. The user/encrypted object authentication process is started and if the user/encrypted object is/are approved, the following encrypted object information can be returned and used by the user:

A. Plain Text Object Name
B. Plain Text Object Location
C. Plain Text Object Application
D. Plain Text Object Environment
E. Plain Text Object Date
F. Plain Text Object Time
G. Plain Text Object Digital Signature
H. Code word Object Tracking Label
I. Cluster Object Tracking Label
J. Device Object Tracking Label
K. Use Object Label
L. Algorithm Object Type

At this point, the authenticated user is given the option to decrypt the requested embedded encrypted object 12. After decryption, a check is done to match the encrypted object's plain text object application to the correct Intelligent Auto Application Interface 14. If the correct Auto Application Interface is not found, a notice is returned and the object is copied to a temporary location 16, otherwise the Auto Application Interface process is started. During this process the encrypted object is matched to the appropriate authenticated application object 18 according to the returned encrypted object information. The correct authenticated application object 18 is then activated with the plain text object 20. Due to the relative dynamic nature of objects, the DCOM is able to accomplish all tasks "on the fly".

The scope of the DCOM directly correlates to the level at which the DCOM was embedded into the system. The scope of the DCOM would cover the implemented embedded system level and all system levels above that, appearing transparent to all levels beneath the implemented embedded system level. For instance, if the DCOM were was embedded at the Open System Interconnection ("OSI") 7 Application layer, then the scope of the DCOM would cover objects on that level and above. In this scenario, the DCOM could run transparent to OSI levels 1 through 6. On the basis of current technology, this implementation would produce the most flexible DCOM. At this level and above, the DCOM is able to provide multi-level multi-media security while staying at the document level. This cross-application compatibility or document-level security is critical to the evolving component based document centered computer system desktop. The DCOM achieves cross-application multi-level multi-media security at the document level through its use of Object-Based Security.

The current implementation of the DCOM at the application layer is called the Object-Oriented Key Manager (OOKeyMan). Currently, OOKeyMan is a Microsoft Windows 3.1 stand alone application, but the DCOM can be applied to other environments. OOKeyMan provides Document-Level Security through its use of Object Based Security.

Some examples of where the DCOM can be applied to ensure the authentication and data integrity of objects include:

IBM's OS/2 2.X and above
IBM's System Object Method(SOM)
Microsoft's Object Package
Microsoft's Object Linking and Embedding 1.0(OLE 1.0)
Microsoft's Object Linking and Embedding 2.0(OLE 2.0)
Microsoft's Windows NT 3.1
Microsoft's Cairo (Future Operating System)
Microsoft's Chicago (Future Operating System)
Taligent (joint venture future Operating System of Macintosh and IBM)
Macintosh's Compound Document Standard
Macintosh Operating System
Novell
Novell Netware Directory Services(NDS)
Unix Object-Oriented Systems
Virtual/Alternate Reality Systems
Future Object-Oriented Operating Systems

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

By applying the DCOM to the above examples, the security of a system can be moved to a more abstract object level. By securing objects with cryptography, a level of security is achieved much higher than that of common access control mechanisms such as password or pass phrase protection.

The steps for embedding an Encrypted Embedded OOKeyMan Object(s) are as follows:

1. User Creates a plain text Object by using a standard application;
2. User Encrypts Object(s) with OOKeyMan;
3. User uses a standard Container Object;
4. Using Standard object to embed Encrypted Embedded OOKeyMan object(s) into Container Object;
5. Encrypt Container Object;
6. Repeat Steps 1 through 5 until all Objects are encrypted;
7. Multi-Level Multimedia Security achieved at the document level.

Examples of the Distributed Cryptographic Object Method

The following resources were used in the following examples:

Software:
MS-DOS 5.0
Microsoft Windows 3.1
Microsoft Word for Windows 2.0c
Standard Microsoft Object Package
WordPerfect 5.2 for Windows
OOKeyMan 1.Ob
Auto Application Interface for Word 1.0
Auto Application for WordPerfect 5.2 for Windows 1.0 **Hardware:**
486 50 MHz DX with 16 megabytes of RAM

The next two examples demonstrate some of the capabilities of the DCOM through the OOKeyMan implementation. The examples involve two of the most popular standard Microsoft Windows applications on the market today; Word TM for Windows ® and WordPerfect TM for Windows ®. The examples also use a standard object, Standard Microsoft Object Package, to embed the encrypted files in a standard container object. The interfaces used for these examples were the Auto Application Interface for Word for Windows 1.0 and the Auto Application for WordPerfect 5.2 for Windows 1.0.

The first example shows the ability for OOKeyMan to securely manage and track single or multiple embedded encrypted objects within other encrypted objects. This is done with a single application.

The second example shows the ability for OOKeyMan to securely manage and track single or multiple embedded encrypted objects within other encrypted objects. The embedded encrypted objects can even be part of encrypted objects from other applications. This example is performed in a cross-application manner between Word for Windows and WordPerfect for Windows.

Example 1: Document Level Multi-Level Multimedia Security

(with Microsoft Word for Windows and WordPerfect 5.2 for Windows)

OOKeyMan Process:

1. Lock Object
   - A. User creates an object(s) in Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Encrypt;
   - D. User Selects Labels for object;
   - E. User Selects an algorithm for encryption;
   - F. User Selects Lock Object;
   - G. OOKeyMan Object Manager performs Setup and Internal Checks;
   - OOKeyMan Object Manager Calls Key Management System object;
   - I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;
   - J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
   - K. Encrypted OOKeyMan Object Created;
     - i. Results in Encapsulation of
       - a. Plain Text Object
       - b. Plain Text Object Name
       - c. Plain Text Object Location
       - d. Plain Text Object Application
       - e. Plain Text Object Environment
       - f. Plain Text Object Date
       - g. Plain Text Object Time
       - h. Plain Text Object Digital Signature
       - i. Code word Object Tracking Label
       - j. Cluster Object Tracking Label
       - k. Device Object Tracking Label
       - l. Use Object Label
       - m. Algorithm Object Type
     - ii. Results in New Encrypted Object being created
     - iii. Results in Plain Text Object Being Delete if Requested
   - L. OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.
2. Unlock Object
   - A. User creates an encrypted object(s) using Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Decrypt;
   - D. User Selects Unlock object;
   - E. OOKeyMan Decrypt Object;
   - F. OOKeyMan Object Manager performs Setup and Internal Checks;
   - G. OOKeyMan Object Manager Calls Key Management System object;
   - H. OOKeyMan Object Manager Calls Algorithm object;
   - I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object;
   - J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.
3. Preview Object
   - A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Preview;
   - D. User selects Preview Object;
   - E. OOKeyMan Display Header Object;
   - F. OOKeyMan Object Manager performs Setup and Internal Checks;
   - G. OOKeyMan Object Manager Calls Key Management System object;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

Example 2: Cross-Application Multi-Level Multimedia Security at The Document Level

(Between Microsoft Word for Windows and WordPerfect 5.2 for Windows)

OOKeyMan Process:

1. Lock Object
   - A. User creates an object(s)in Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Encrypt;
   - D. User Selects Labels for object;
   - E. User Selects an algorithm for encryption;
   - F. User Selects Lock Object;
   - G. OOKeyMan Object Manager performs Setup and Internal Checks;
   - H. OOKeyMan Object Manager Calls Key Management System object;
   - I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;
   - J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
   - K. Encrypted OOKeyMan Object Created;
     - i. Results in Encapsulation of
       - a. Plain Text Object
       - b. Plain Text Object Name
       - c. Plain Text Object Location
       - d. Plain Text Object Application
       - e. Plain Text Object Environment
       - f. Plain Text Object Date
       - g. Plain Text Object Time
       - h. Plain Text Object Digital Signature
       - i. Code word Object Tracking Label
       - j. Cluster Object Tracking Label
       - k. Device Object Tracking Label
       - l. Use Object Label
       - m. Algorithm Object Type
     - ii. Results in New Encrypted Object being created
     - iii. Results in Plain Text Object Being Delete if Requested
   - L. OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.
2. Unlock Object
   - A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Decrypt;
   - D. User Selects Unlock object;
   - E. OOKeyMan Decrypt Object;
   - F. OOKeyMan Object Manager Setup and Internal Checks;
   - G. OOKeyMan Object Manager performs Calls Key Management System object;
   - H. OOKeyMan Object Manager Calls Algorithm object;
   - I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
   - J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.
3. Preview Object
   - A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Preview;
   - D. User selects Preview Object;
   - E. OOKeyMan Display Header Object;
   - F. OOKeyMan Object Manager Setup and Internal Checks;
   - G. OOKeyMan Object Manager Calls Key Management System object;
   - H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

Example 3: Standard Distributive Cryptographic Object Method Process (DCOMP)

OOKeyMan Process:

1. Lock Object
   - A. User creates an object(s);
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Encrypt;
   - D. User Selects Labels for object;
   - E. User Selects an algorithm for encryption;
   - F. User Selects Lock Object;
   - G. OOKeyMan Object Manager performs Setup and Internal Checks;
   - H. OOKeyMan Object Manager Calls Key Management System object;
   - I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;
   - J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
   - K. Encrypted OOKeyMan Object Created;
     - i. Results in Encapsulation of
       - a. Plain Text Object
       - b. Plain Text Object Name
       - c. Plain Text Object Location
       - d. Plain Text Object Application
       - e. Plain Text Object Environment
       - f. Plain Text Object Date
       - g. Plain Text Object Time
       - h. Plain Text Object Digital Signature
       - i. Code word Object Tracking Label
       - j. Cluster Object Tracking Label
       - k. Device Object Tracking Label
       - l. Use Object Label
       - m. Algorithm Object Type
     - ii. Results in New Encrypted Object being created
     - iii. Results in Plain Text Object Being Delete if Requested
   - L. OOKeyMan Object Manager Returns To Application Object.
2. Unlock Object
   - A. User creates an encrypted object(s);
   - B. User Initiates OOKeyMan sequence;
   - C. User Selects object(s) to Decrypt;
   - D. User Selects Unlock object;
   - E. OOKeyMan Decrypt Object;
   - F. OOKeyMan Object Manager performs Setup and Internal Checks;
   - G. OOKeyMan Object Manager Calls Key Management System object;
   - H. OOKeyMan Object Manager Calls Algorithm object
   - I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
3. Preview object
   - A. User creates an encrypted object(s);

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan Display Header Object;
F. OOKeyMan Object Manager performs Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

The DCOM process can be applied to a vast number of areas in the real world. Whether it be the physical topology of the local area network/wide area network environment or the dynamic structure of an organization, the DCOM process will change dynamically to reflect the current state of the object in question.

FIG. 3 and FIG. 4 show an encrypted object that contains a web of embedded encrypted objects nested within the other encrypted objects. The object shown in FIG. 3 contains ten embedded encrypted objects at five various levels. The encrypted object embedded in level 5 was embedded in an object in level four, level four objects in level 3 and so on. The plain text object containing the level 5 encrypted object can then be encrypted for further security. This single encrypted object encapsulates all of the data associated with the encrypted objects within it and therefore the entire encrypted object can then be sent out via any transport mechanism supporting binary file transfer.

FIG. 4 shows an encrypted object that contains a web of embedded encrypted objects nested within it. All of the attached embedded encrypted objects are fused together resulting in a single encapsulated encrypted object. The DCOM is powerful enough to dynamically adapt to accommodate N dimensional objects. In the very near future computing systems incorporating technology such as Virtual/Alternate Reality and Cyberspace, will need systems that can secure N dimensions.

The single encrypted objects shown in both FIGS. 3 and 4 can act as a secure package and can be sent out for distribution to an entire organization (e.g. E-mail). This single encrypted object can represent a branch(s), department(s), or even an entire company. Every employee would receive the single encrypted file, but they would only be able to unravel the portions that corresponded to them and acquire no knowledge of other existing embedded encrypted objects. For example, FIG. 5 displays a sample organization chart. When applied, the DCOM would control the knowledge/information flow of the organization and would allow for clear data separation, further compartmentalization through multiple algorithm use, and document-level security. With the improved communication paths, an organization would become more efficient. FIG. 6 demonstrates the use of the DCOM in conjunction with the dynamic structure of a sample organization. Since the DCOM is dynamic in nature, it can adapt to any organizational size or type (For example, see FIGS. 7 and 8).

Preferred and alternate embodiments of the present invention have now been described in detail. It is to be noted, however, that this description of these specific embodiments is merely illustrative of the principles underlying the inventive concept. It is therefore contemplated that various modifications of the disclosed embodiments will, without departing from the spirit and scope of the invention, be apparent to persons skilled in the art.

What is claimed is:

1. A method for providing multi-level multimedia security in a data network, comprising the steps of:
A) accessing an object-oriented key manager;
B) selecting an object to encrypt;
C) selecting a label for the object;
D) selecting an encryption algorithm;
E) encrypting the object according to the encryption algorithm:
F) labelling the encrypted object:
G) reading the object label;
H) determining access authorization based on the object label; and
I) decrypting the object if access authorization is granted.

2. The method of claim 1, wherein the object is an application document, and further comprising the steps of:
A) creating an object in an application prior to accessing the object-oriented key manager; and
B) returning the encrypted object to the application prior to reading the object label.

3. The method of claim 1, further comprising the step of embedding the encrypted object in a second object after labelling the encrypted object.

4. The method of claim 3, further comprising the steps of:
A) selecting a second label for the second object;
B) selecting an encryption algorithm;
C) encrypting the second object; and
D) labelling the second encrypted object with a second object label.

5. The method of claim 4, further comprising the steps of:
A) reading the second object label;
B) determining access authorization based on the second object label; and
C) decrypting the second object if access authorization is granted.

6. The method of claim 4, wherein the second label is a second plurality of labels.

7. The method of claim 1, wherein the label is a plurality of labels.

8. A system for providing multi-level multimedia security in a data network, comprising:
A) digital logic means, the digital logic means comprising:
1) a system memory means for storing data;
2) an encryption algorithm module, comprising logic for converting unencrypted objects into encrypted objects, the encryption algorithm module being electronically connected to the system memory means for accessing data stored in the first system memory;
3) an object labelling subsystem, comprising logic means for limiting object access, subject to label conditions, the object labelling subsystem being electronically connected to the system memory means for accessing data stored in the system memory means and the object labelling subsystem being further electronically connected to the encryption algorithm module to accept inputs from the encryption algorithm module;
4) a decryption algorithm module, comprising logic for converting encrypted objects into unencrypted objects, the decryption algorithm module being electronically connected to the

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

system memory means for accessing data stored in the system memory means; and

5) an object label identification subsystem, comprising logic for limiting object access, subject to label conditions, the object label identification subsystem being electronically connected to the system memory means for accessing data stored in the system memory means and the object label identification subsystem being further electronically connected to the decryption algorithm module to accept inputs from the decryption algorithm module;

B) the encryption algorithm module working in conjunction with the object labelling subsystem to create an encrypted object such that the object label identification subsystem limits access to an encrypted object.

9. The system of claim **8**, wherein the digital logic means further comprises means for embedding a first object within a second object.

10. The system of claim **9**, wherein the digital logic means further comprises means for accessing computer program applications stored in the system memory means.

11. The system of claim **8**, wherein the digital logic means further comprises means for accessing computer program applications stored in the system memory means.

12. A system for providing multi-level multimedia security in a data network, comprising:

A) means for accessing an object-oriented key manager;

B) means for selecting an object to encrypt;

C) means for selecting a label for the object;

D) means for selecting an encryption algorithm;

E) means for encrypting the object;

F) means for labelling the encrypted object;

G) means for reading the object label;

H) means for determining access authorization based on the label; and

I) means for accessing the object if access authorization is granted.

13. The system of claim **12**, wherein the object is an application document and the wherein the system further comprises:

A) means for creating an object in an application prior to accessing the object-oriented key manager; and

B) means for returning the encrypted object to the application prior to reading the object label.

14. The system of claim **13**, further comprising means for embedding an object within a second object.

15. The system of claim **14**, further comprising:

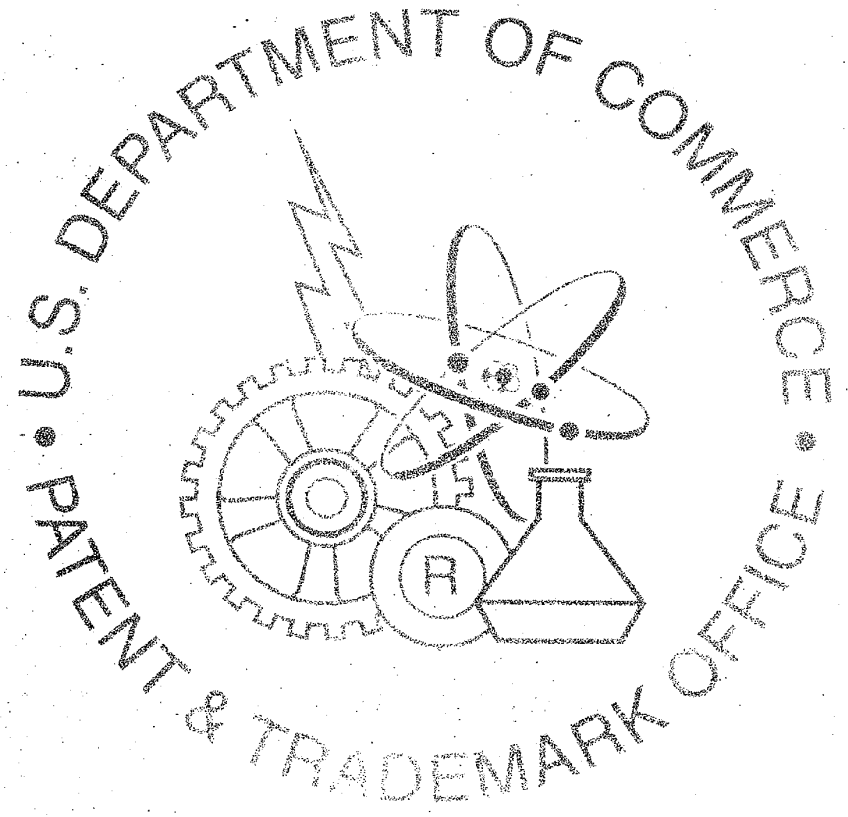
A) means for reading the second object label;

B) means for determining access authorization based on the second object label; and

C) means for decrypting the second object if access authorization is granted.

* * * * *

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



TEC00140774

PTO-1683
(Rev. 7-96)

TEC00140775

U 7214101

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:
UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

November 25, 2009

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,680,452*
ISSUE DATE: *October 21, 1997*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

P. SWAIN
Certifying Officer

Plaintiff's Trial Exhibit
2
TecSec. v. Adobe Case no. 1:10cv115

TEC00140776

US005680452A

United States Patent [19]

Shanton

[11] Patent Number: 5,680,452

[45] Date of Patent: *Oct. 21, 1997

[54] **DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD**

[75] Inventor: **M. Greg Shanton**, Manassas, Va.

[73] Assignee: **TECSEC Inc.**, Vienna, Va.

[*] Notice: The term of this patent shall not extend beyond the expiration date of Pat. No. 5,369,702.

[21] Appl. No.: **394,402**

[22] Filed: **Feb. 24, 1995**

Related U.S. Application Data

[63] Continuation-in-part of Ser. No. 304,867, Sep. 13, 1994, which is a continuation of Ser. No. 138,857, Oct. 18, 1993, Pat. No. 5,369,702.

[51] **Int. Cl.**[6] **H04L 9/00**

[52] **U.S. Cl.** **380/4**; 380/9; 380/21; 380/23; 380/25; 380/28; 380/49; 380/50; 340/825.31; 340/825.34

[58] **Field of Search** 380/3, 4, 5, 9, 380/10, 23, 24, 25, 21, 49, 50, 28, 30, 43; 340/825.31, 825.34; 455/33.3

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,218,582	8/1980	Hellman et al.	380/30
4,405,829	9/1983	Rivest et al.	380/30
4,424,414	1/1984	Hellman et al.	380/30
4,713,753	12/1987	Boebert et al.	380/4 X
4,864,616	9/1989	Pond et al.	380/25
4,955,082	9/1990	Hatoori et al.	455/33.3
4,962,533	10/1990	Krueger et al.	380/25
4,984,272	1/1991	McIlroy et al.	380/25
5,052,040	9/1991	Preston et al.	380/4
5,065,429	11/1991	Lang	380/25
5,191,611	3/1993	Lang	380/25
5,204,961	4/1993	Barlow	380/4 X
5,369,702	11/1994	Shanton	380/4
5,369,707	11/1994	Follendore III	380/25

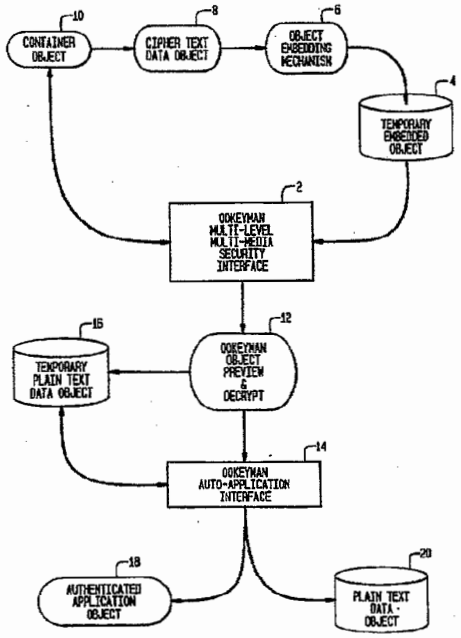
Primary Examiner—Bernarr E. Gregory

Attorney, Agent, or Firm—Rabin, Champagne & Lynt, P.C.

[57] **ABSTRACT**

A system for increasing the security of a computer system, while giving an individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute to other individuals multiple encrypted objects embedded in a single encrypted object. By effecting compartmentalization of every object by label attributes and algorithm attributes, multi-level multimedia security is achieved. Label attributes are used to restrict access to objects based on location, group, or other criteria and may specify personal access. Access type, such as read-only, write-only, and print-only may be specified. Nested embedded objects may be accessed directly through selection from a header array.

17 Claims, 8 Drawing Sheets



Copy provided by USPTO from the PIRS Image Database on 11/24/2009

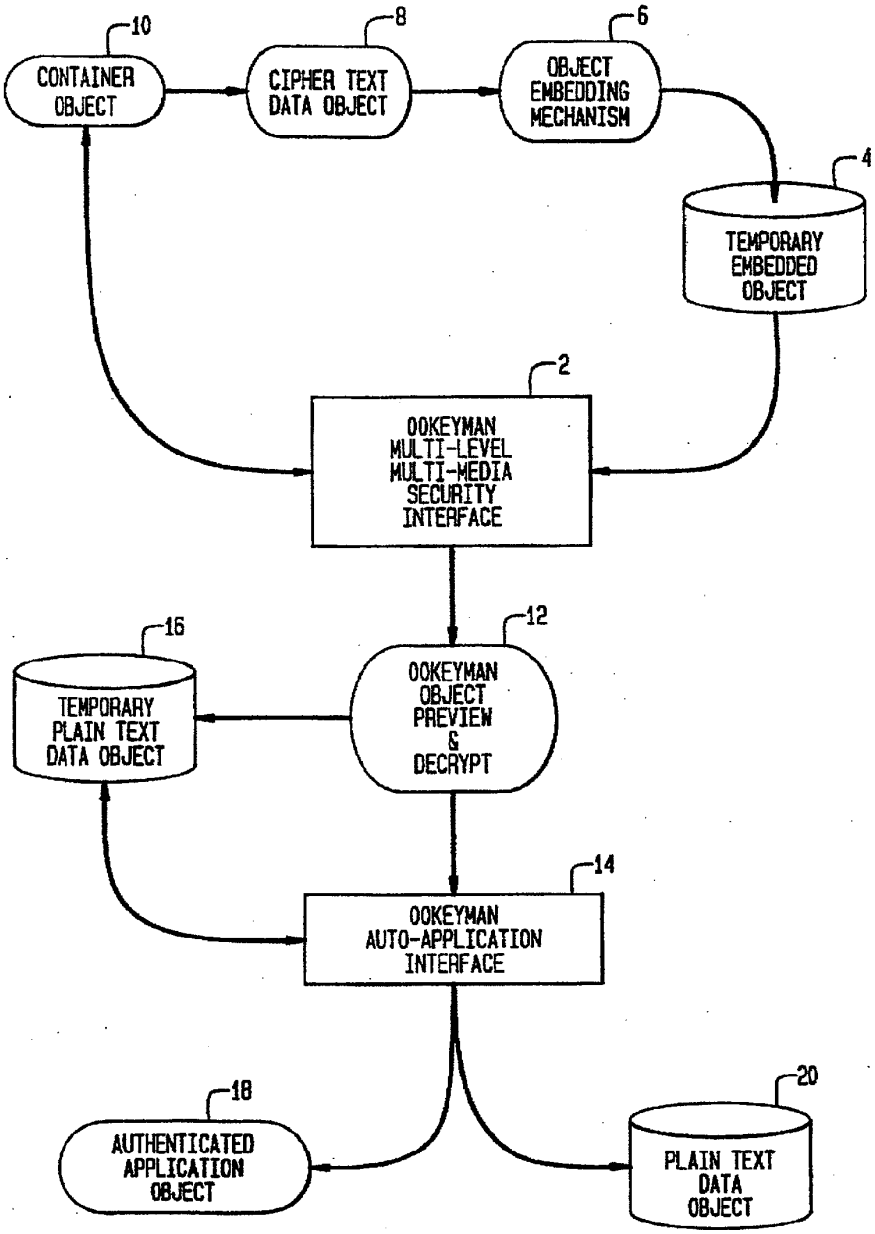


FIG. 1

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

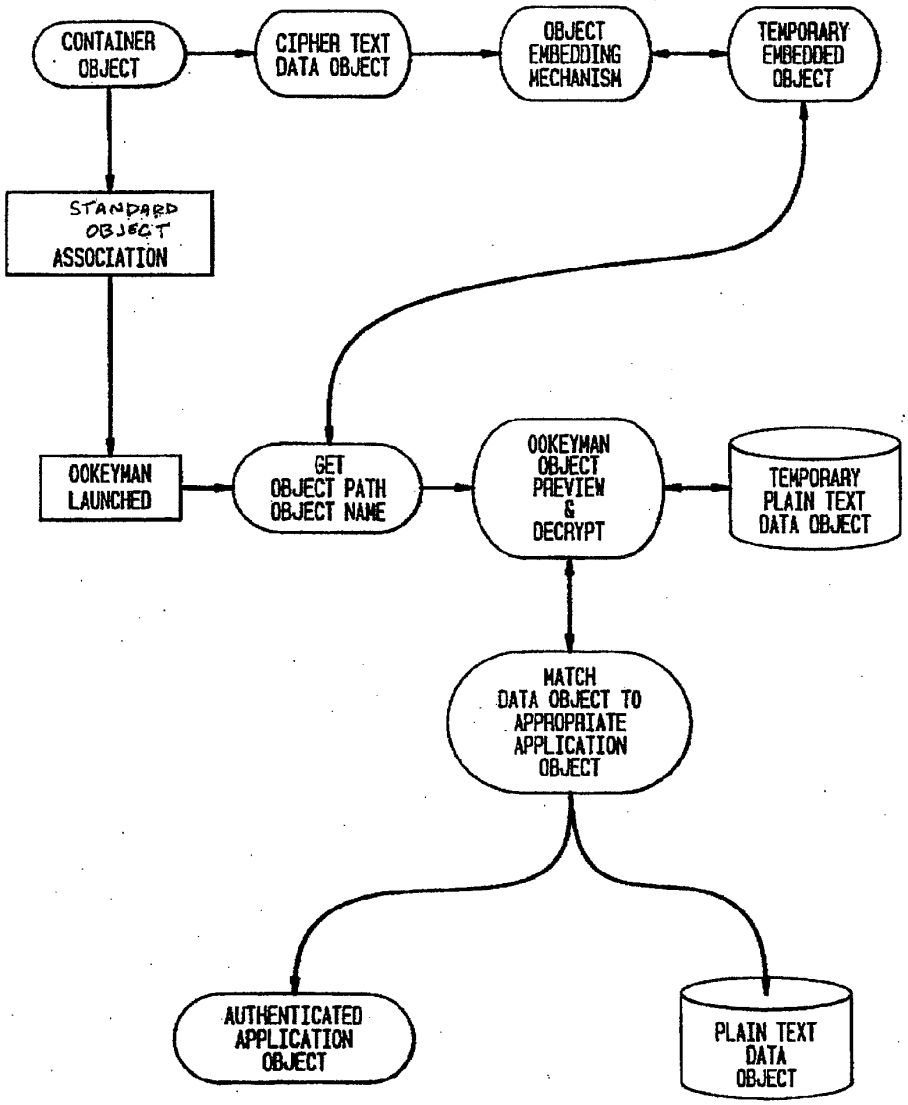


FIG. 2

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

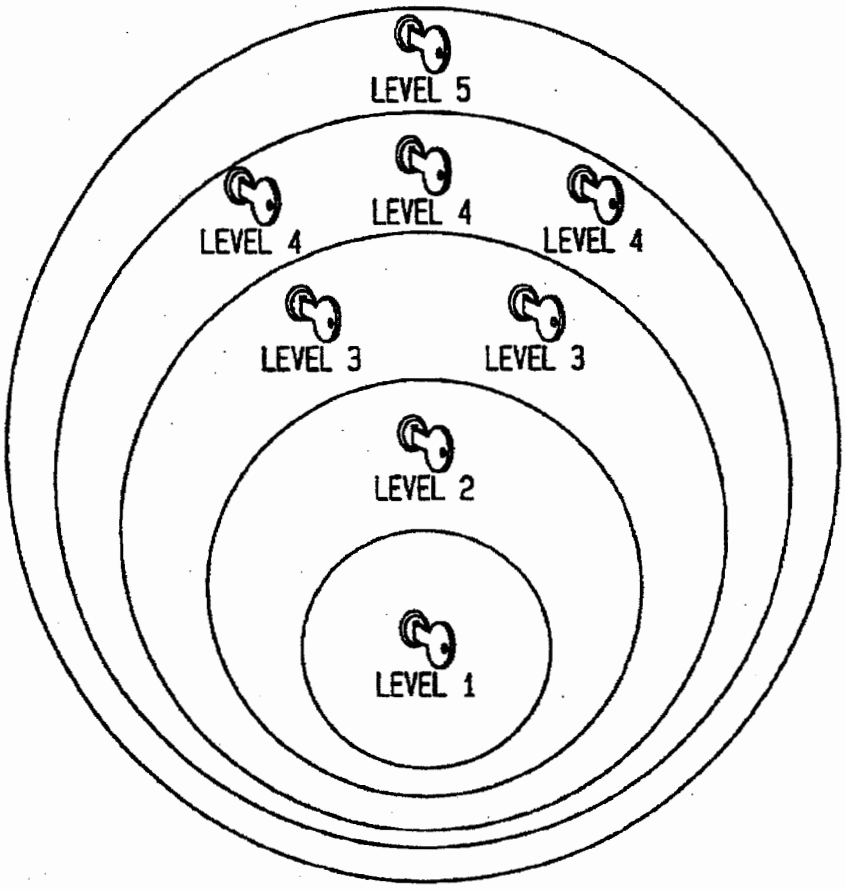


FIG. 3

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

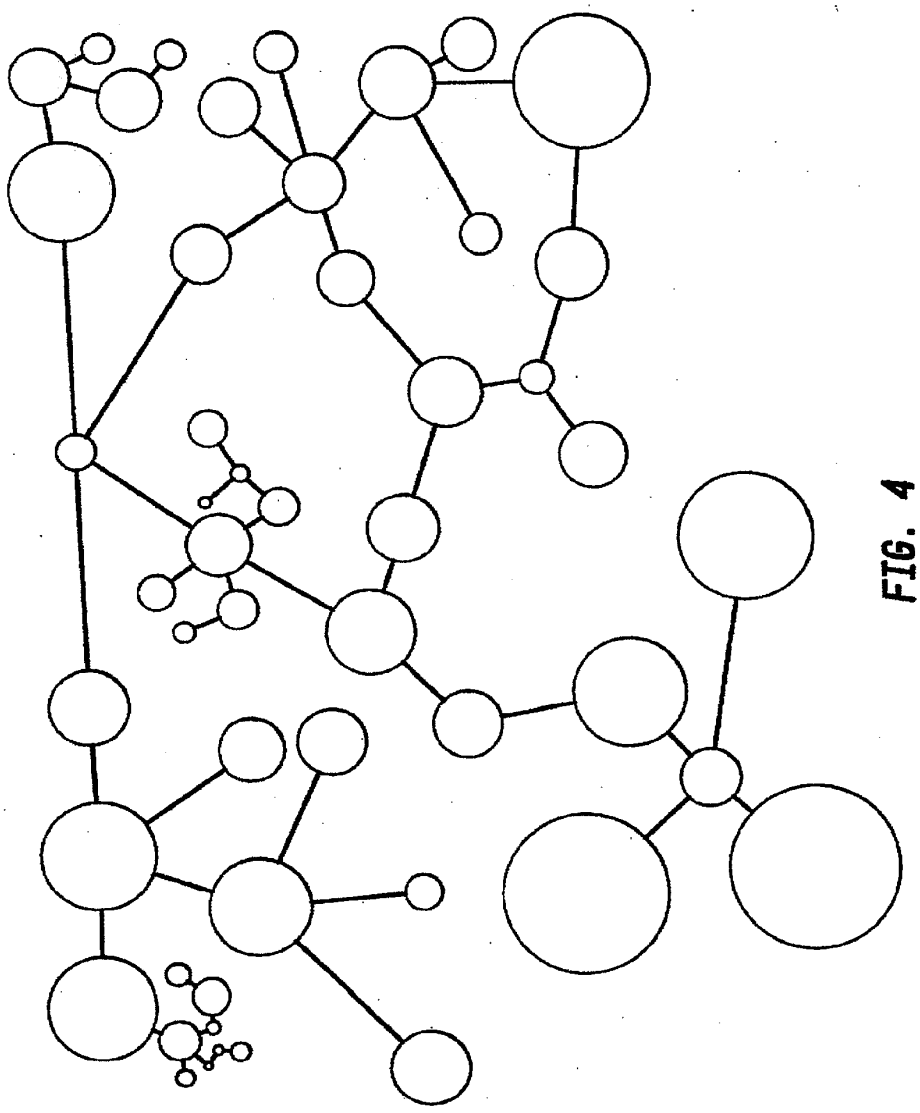


FIG. 4

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

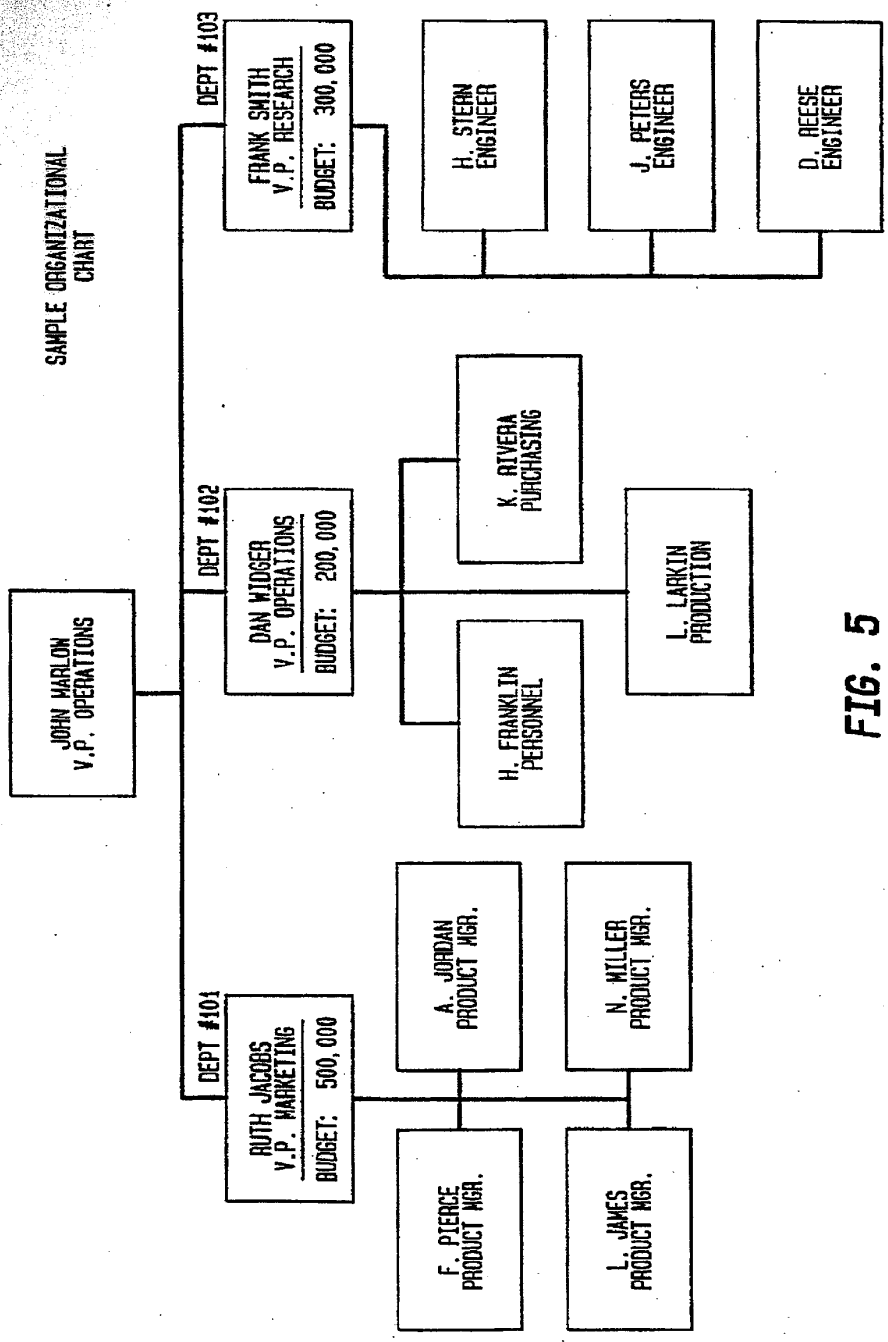


FIG. 5

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 138 Filed: 10/07/2019

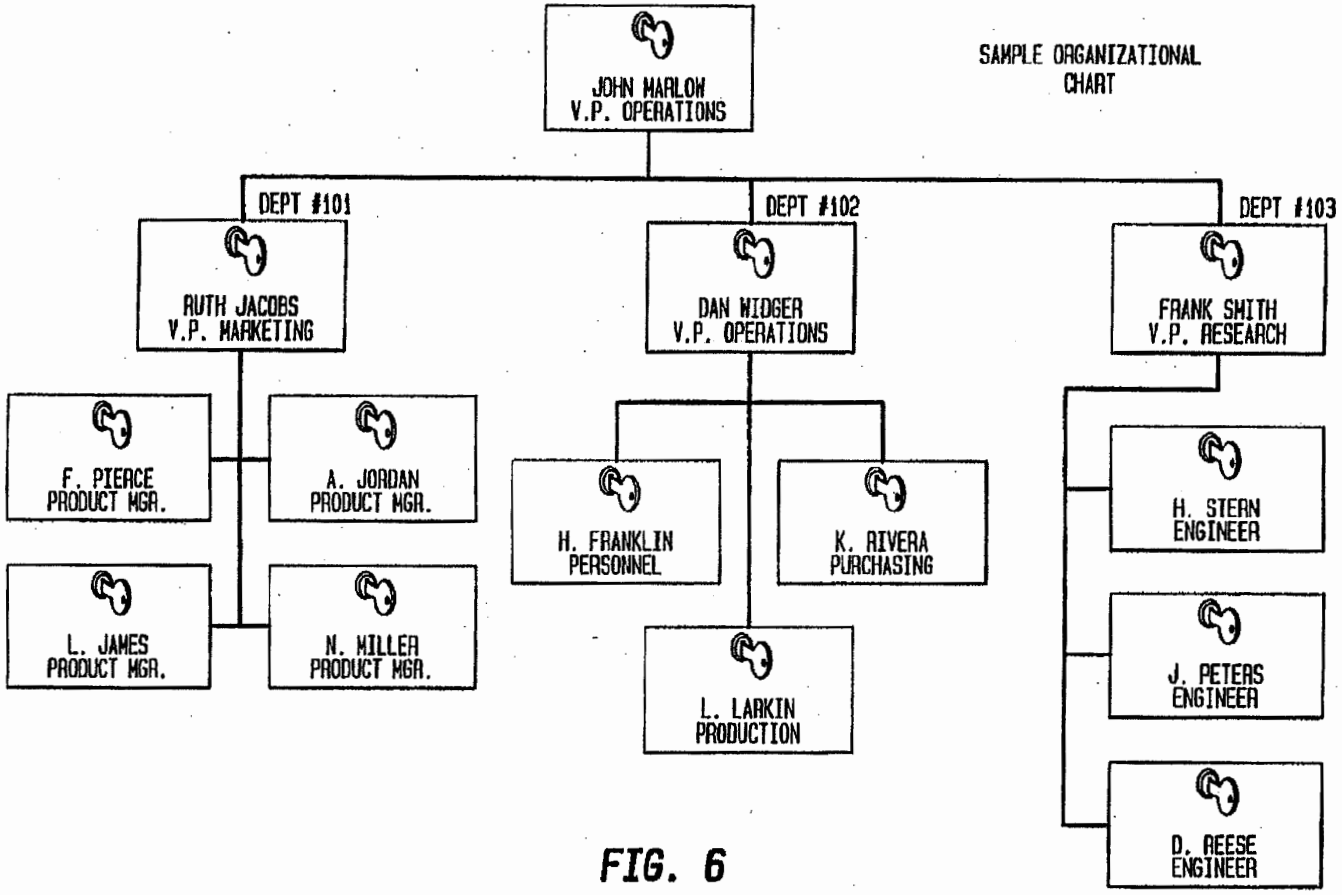
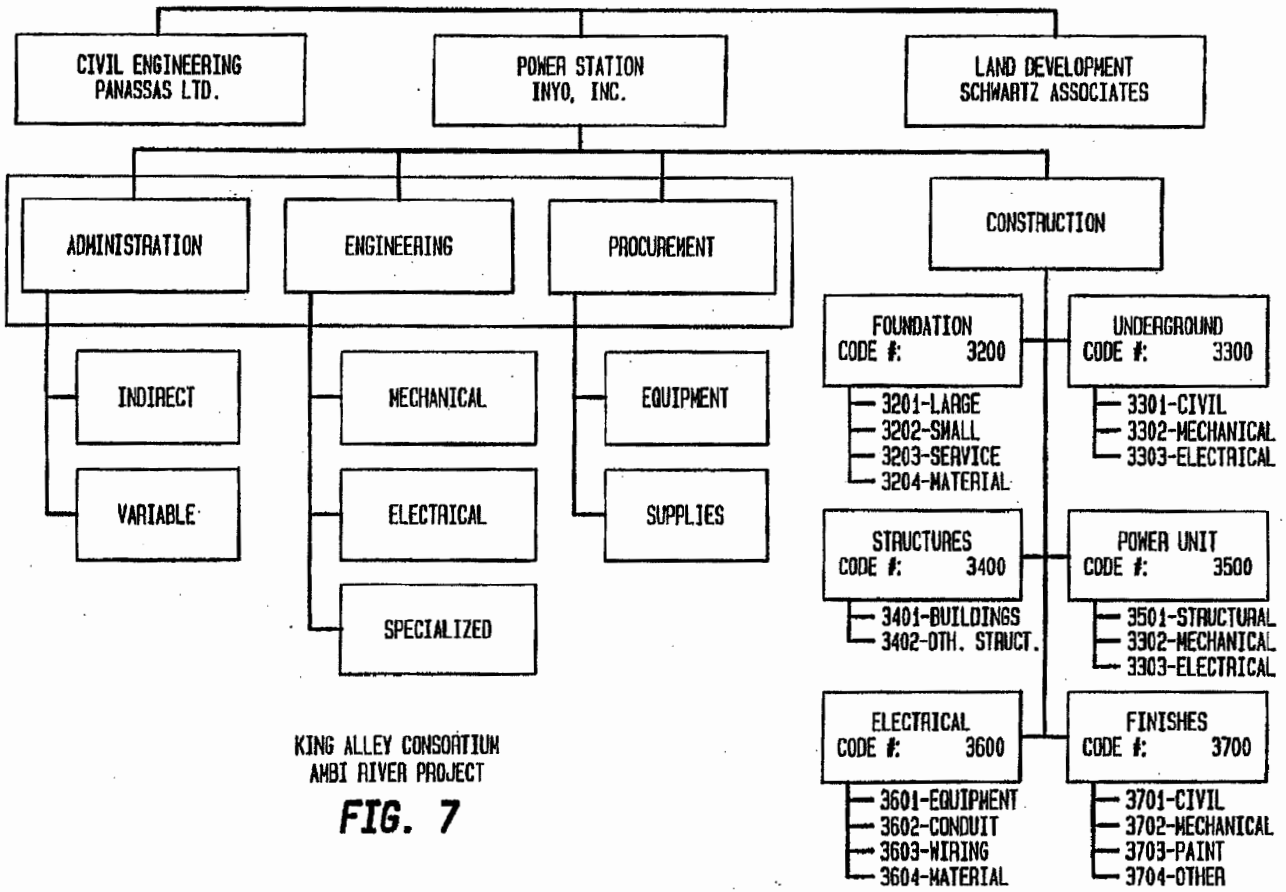


FIG. 6

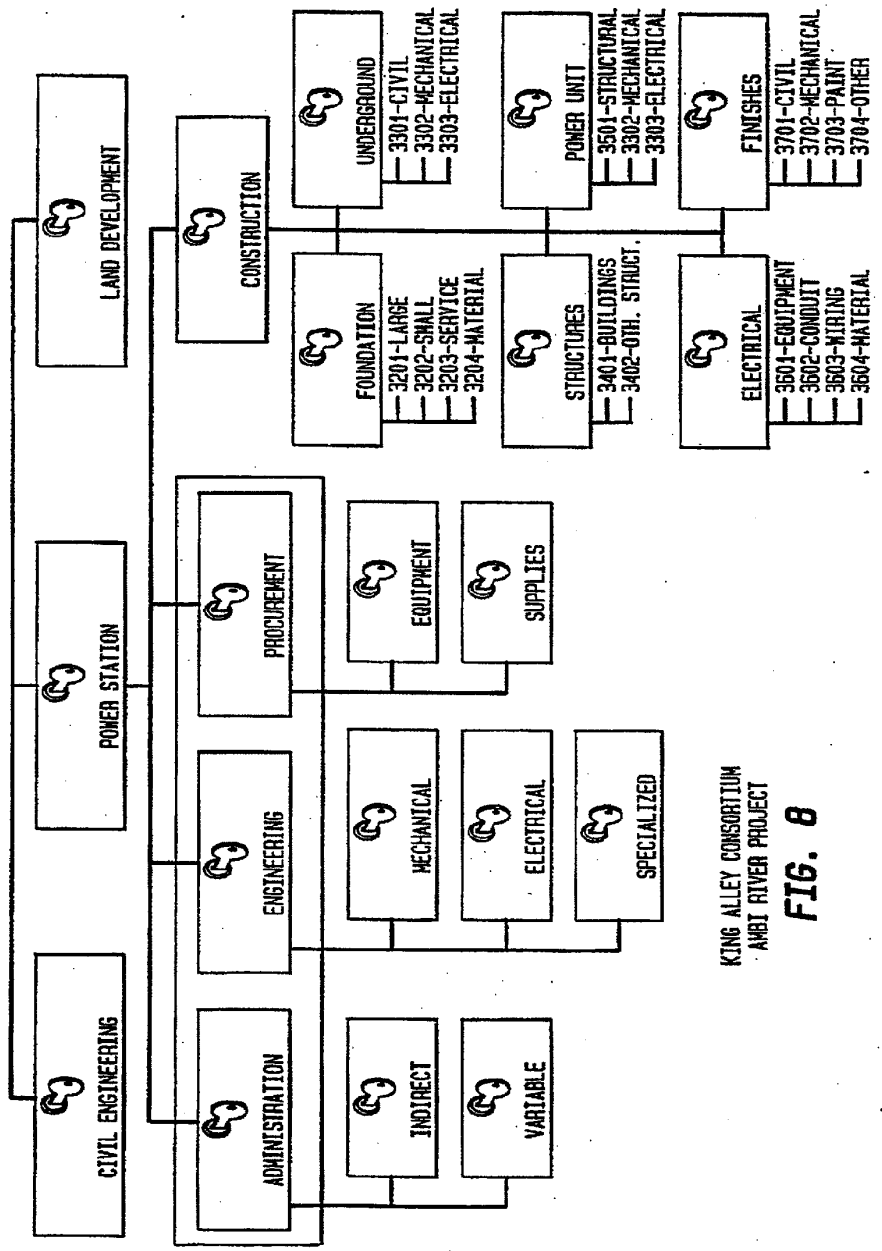
Copy provided by USPTO from the PIRS Image Database on 11/24/2009



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 7

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 141 Filed: 10/07/2019

DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD

This is a continuation-in-part of U.S. patent application Ser. No. 08/304,867, filed Sep. 13, 1994, which is a continuation of U.S. patent application Ser. No. 08/138,857, filed Oct. 18, 1993, which issued as U.S. Pat. No. 5,369,702 on Nov. 29, 1994.

FIELD OF THE INVENTION

The present invention relates generally to communications security systems. In particular, the system of the present invention relates to computer security systems and means of restricting access to data stored on or communicated by computers.

BACKGROUND OF THE INVENTION

The specter of spies eagerly trying to obtain the defense information of various countries is very much still present in the defense and intelligence community. An equally dangerous threat also exists in the form of technological or commercial spies who desire to obtain commercial and technical information from competing companies, and who may use industrial espionage in their efforts to gain access to this information. Such espionage may include the use of sophisticated means similar to those used by the defense and intelligence community. The data obtained may include commercially valuable information that reveals the plans and commercial activities of competitors, the acquisition of which allows the aggressor company to obtain a competitive advantage in the marketplace. Theft of commercially valuable information is a very real and ever present threat.

To combat this type of commercial spying and protect company proprietary information, various complex systems have evolved to protect proprietary information. These systems usually employ physical controls over personnel as well as over the data flowing in and out of a company. For example, most computer systems used within companies require a password or passphrase to be entered before the system can be accessed. It is frequently the case that confidential or company proprietary information must be passed electronically from one location to another in order to convey that information between employees within the company in a timely fashion. Such electronic communication is highly susceptible to interception if not protected in some manner. Passphrase protection is not adequate to protect against interception or corruption of data in transmission.

Cryptographic systems have evolved to fill the needs of companies and individuals wanting to protect proprietary commercial information from competitors and others who generally should not be privy to that information. Such systems ensure that data is not recognizable to unauthorized parties who intercept it. Only authorized persons would have the means for decrypting the data and returning it to readable form.

For example, U.S. Pat. No. 4,405,829 to Rivest et al. discloses a cryptographic communications system utilizing a particular manipulation of data to be transmitted in order to effect encryption. The data is first encoded as a number in a predetermined set, which is then raised to a predetermined power corresponding with the intended receiver. The result is then divided by the product of two predetermined prime numbers corresponding with the intended receiver. The remainder of this division is the cipher text. A reverse procedure is followed at the receiver to decrypt the cipher text.

Encryption of data is therefore a critical requirement in controlling access to confidential information. Cryptographic "keys" are an essential part of the data encryption process. The cryptographic key, or "key" for short, is a data stream that is manipulated with plain (readable) text data according to a cryptographic algorithm to transform the plain text data to a string of unintelligible text or signals known as encrypted text or cipher text. The key is then used by an authorized receiver of the cipher text to decrypt the message, returning it to plain text form. However, for two people to communicate successfully using such a cryptographic system, each must use the same key, assuming that the same encryption/decryption algorithm is used on both ends of the communication. Thus, a system must be put into place for the distribution and other management of these keys within the closed group of authorized users.

For example, U.S. Pat. No. 4,218,582 to Hellman et al. discloses a public key cryptosystem for providing secure data transmission. According to the Hellman et al. system, a receiver generates a secret key and a public key, such that the secret key is difficult to generate from the public key, which in turn is easy to generate but is difficult to invert without the secret key. The transmitter encodes the data using the public key. The encrypted message is then transmitted to the receiver, which decrypts the data using the secret key. U.S. Pat. No. 4,424,414 to Hellman et al. discloses a similar system which uses an exponentiation function rather than a public key.

Various methods have evolved to manage the distribution of keys. Such methods of distribution are commonly referred to as "key management". The function of key management is to perform the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys. Under normal operational circumstances, the key manager begins and ends a cryptographic session by controlling access to the algorithm used to encrypt and decrypt plain text objects. Thus, a user who wants to encrypt an object or decrypt an object must first access the key manager so that an encryption algorithm may be chosen.

Simple encryption of data being communicated between two points only provides one level of security, however. Encryption limits data communication to those who have the key. Anyone who has the key is privy to any communication at any location. That is, if a group of people are working on a particular project, they will all presumably share a key for decrypting information relating to the project. Some of the project group may be working in one location, while the rest of the group may be located in a distant city. If one member of the group wants to send a communication to be read only by a particular member in the other city, the key will afford him no protection because everyone in the project shares the same key, and therefore will have access to the communication. Likewise, if someone wants to communicate a message to a subset of the group, for example, only to management personnel, this key would again provide her with no extra security. In another case, someone may want to send a message that is capable of being read only at a particular computer terminal, or of being printed only at a particular printer. In these and other cases, multilevel multimedia key access or individual keys issued to each person, would provide a solution, albeit one that is quite unwieldy, inflexible, and difficult to manage by a security officer or key administrator.

A secure method of labeling files or messages that are sent from a sending user to a receiving user over a network can provide a level of protection in addition to cryptographic

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

protection. A file "label" relative to the present invention means a series of characters, which may or may not be encrypted, separate from the file or message but associated with the storing of a file or the sending of a message, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated message or to read or modify an associated file. Using a secure labeling regimen, a network manager or user can be assured that only those messages or files meant for a certain person, group of persons, and/or location(s) are in fact received, decrypted, and read by the intended receiver. Thus, a sending user can specify label conditions that limit access to the transmitted message. For example, many people within a company may have the key necessary to read a data file that a sender may transmit from his computer terminal to other terminals at another site within his company. The sender may, however, wish to restrict reception to those persons present at a particular terminal. By employing a secure labeling technique in addition to encryption, the sender can be assured that people having the correct key to decrypt the message but working at different terminals will not receive or be allowed to access the communication. Using such a scheme, access may be limited to particular people as well.

U.S. Pat. No. 5,369,707 to Follendore III discloses a system for encrypting and labeling files in order to limit access to them. Access is limited only to particular users or groups of users, however. Using the Follendore system, access cannot be limited to, for example, a particular printer.

Likewise, U.S. Pat. No. 5,052,040 to Preston et al. discloses a multiple-user stored data cryptographic labeling system that uses labels to control access to data files. Objects other than data files are not controlled by this system, and even these data file objects do not have the capability of being cryptographically embedded.

A system that can limit access on an object level would be more flexible and would offer still more protection. Access could be specified on an object-by-object basis, and objects could be embedded within other objects, providing an access hierarchy for users.

The ability to cryptographically secure objects ensures the authentication and data integrity of the particular object or objects in question. If a system were able to cryptographically control an object or nested objects, then that system would have total control over the entire object and all other objects within it. This type of control over the knowledge/information flow would allow for clear data separation, and at some levels could become a transparent method. A system that is able to do this would be able to achieve multi-level multimedia security.

SUMMARY OF THE INVENTION

It is therefore an objective of the present invention to provide a system that insures that properly specified kinds of information in a network system flows only to designated locations and to further insure that such information is only read by those individuals who are designated to review that information.

It is a further objective of the present invention to provide a system that recognizes objects and permits or denies access on the object level.

It is an additional objective of the present invention to provide a system in which objects may be embedded within other objects, resulting in an access hierarchy for users of the system.

It is another objective of the present invention to provide a system in which access control is transparent to the user.

These and other objectives and advantages of the present invention will be apparent to those of ordinary skill in the art upon inspection of the detailed description, drawings, and appended claims.

The present invention is a data processing system which can be used to select and encrypt objects. The encrypted objects, either individually or in groups, can be embedded in other objects called container objects, which may also be encrypted. This process is based on a technique called cryptographic encapsulation. The original representation of the object embedded within this container object can then be deleted, as all of the contained object information can be found in encrypted form within the container object. The container object can be further embedded within another container object. Container objects can only be "opened", that is, decrypted, by users having access authority in the form of a cryptographic key and who are using the proper cryptographic engine. Existence of the embedded object is not even known to users who don't have access to the container. The present invention can be utilized with all objects in the system, including standard software applications, utilities, operating systems, and devices. Device objects can be embedded within other device objects, or within a data file object. For example, a printer object may be embedded within a file object; only a user having cryptographic access to the data file will know of the accessibility of the system to the printer.

The definition and concept of objects varies greatly depending on with whom you consult. Everything around you in your daily life is an object. Your car, your car keys, books, people, desks, etc. are all objects in the ordinary sense of the word. Objects are entities by themselves, but they may contain other objects, in either single or multiple configurations. Objects can change their make up dynamically by inheritance. Objects can inherit the attributes of other objects and the inheritance features can change dynamically "on the fly" during the operation of the objects.

In the context of the present invention, objects can come in a vast number of forms, shapes, or sizes and can be either passive or active, dynamic or static. An object may stay dormant until it is acted upon, or it may be an active participant, dynamically auditing and verifying every transaction that occurs in a system. Examples of what an object can be include a bit of information, a byte of information, Sound Clips, Video Clips, Graphic Images, text, charts, tables, forms, controls, MDIForms, variables, executable files, video files, binary files, text files, data files, container files, graphic files, application file(s), library files, a directory, a collection of directories, a hard disk, multiple hard disks, any hardware component, any software component, a complete computer system, a single network, and multiple networks.

Thus, an object is any distinct, separate entity. In a computer or data communication context, entities that may be treated as objects include:

1) Program objects, representing applications such as word processors, spreadsheets, games, etc., as well as utilities and operating systems;

2) Folder objects, representing collections of other objects;

3) Data file objects, including information such as text, memos, letters, spreadsheets, video, and sound; and

4) Device objects, such as printers, fax modems, plotters, and CD-ROM drives.

In object linking and embedding processes, an object can be any user-selected group of data, such as a block of text,

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

a set of spreadsheet cells, a chart, sounds, or a graphical image. These data can be embedded in or linked to another document created by a different application. For example, a folder may represent a directory and may contain a group of files, or it may represent a group of programs. Folders can also contain other folders.

In object-oriented programming, a program consists of a set of related but self-contained objects that can contain both code and data.

As previously stated, the system of the present invention may be used to encrypt and embed objects. The encrypted objects are then not accessible to someone not having a proper key or cryptographic engine. Encrypted embedded objects are not even known to those who do not have access to the container object. Access to the file is further restricted through the use of labels. A label is a field of characters attached to the encrypted file. The label may define a group of people that may have access to the file. The label may define the device at which the file may be accessed. The device may define a single person who may have access. A label may also define the type of access, that is, read only, write only, read and write, print only, etc., that authorized persons may have. A label may also define any combination of authorized people, devices, objects, and/or access type. Thus, the label is a flexible, powerful way to set forth with great specificity all conditions that must be fulfilled in order to gain the defined access to the file. Multiple labels may be attached to an encrypted object. The maximum number of labels that may be used with an object is limited only by the capabilities of the host computer. The system of the present invention is contemplated for use with N labels per object, that is, with any number of labels deemed useful or necessary by the user.

It is important to note that the label is preferably attached to an encrypted object. Thus, an authorized person specified in the label must still have a key to decrypt the object. The system of the present invention can restrict access to an object using labels only, but much stronger protection is available if the object is encrypted as well. Further, passphrase protection is preferably included as a first line of protection. Thus, an authorized user would preferably need three elements to access an object: the correct passphrase, the correct key, and authorization in the label.

For example, a family may own a personal computer ("PC") having a CD-ROM drive, a printer, and a modem. Each member of the family may have a unique passphrase allowing basic access to the PC. The family may include an 11-year old boy whose friends also have been given passphrases so that they can all do homework or play computer games together. However, some files on the PC may contain personal family information; these files may be encrypted, with only family members having keys to decrypt these files. The parents may also wish to restrict the boy's access to the modem because they are afraid that he may run up expensive on-line charges. The modem object may be placed in a container object. This container object may then be encrypted and labelled such that the boy is not granted access to the container object. Thus, not only will the boy not have access to the modem, he will not be able to determine that the modem is embedded within the container object. The parents may also put restrictions on themselves. They may have stored certain important documents on their system which they must reference occasionally, but which must not be changed. They may use the label restriction to give themselves read only access to these documents, insuring that the text is never inadvertently changed. Alternatively, they could restrict themselves to print only access. Thus, when access is attempted, the system of the present invention would determine that they have print only access, would decrypt the file using their key, and would present only print options to them on the display, rather than an image of the document itself. Their only option regarding this document would be to print.

The label may appear as a header to authorized users. This header can include pertinent information about the label, such as a list of authorization attributes and a specification of object type. For example, the header may identify the object as a container object, and may further list the objects embedded in the container object, preferably in the form of an array, or tree structure. This form of presentation is preferable because it is more meaningful to a user when the container object contains other container objects as well as other embedded objects, although other header structures, such as lists, may also be utilized. These lower level container objects may also contain further container objects, and so on, for as many levels as needed by the users, limited only by the amount of memory available on the host computer. In the preferred embodiment, users who are not authorized access to the container object will not be able to read the header in decrypted form, and users who are authorized access to the container object but not to one or more objects embedded in the container object will not be able to read the corresponding header array element.

Thus, the header array will show a user all the objects to which he has access. Using this header, the user is allowed random access to any object embedded in the container object at any level, as long as that particular user is authorized at that location. The user simply selects an object embedded at a level lower than the one he has currently accessed, and the system will open all objects along the path between the highest level object and the selected object. Obviously, this will only be allowed if the user has access to all objects along the path. Thus, the user does not have to open the objects along the path one at a time; the system will open them automatically. Of course, the user has the option of opening the objects one at a time if this is desired.

As an alternative to having the labels attached to each encrypted object, the labels may be attached to an object called an access control list ("ACL"). The ACL will contain the labels corresponding to a number of different objects. When access to one of these objects is attempted, operation is directed to the ACL, which provides the system with the information needed to grant or deny access to the user. If access is granted, operation is directed back to the target object. If access is denied, operation is directed back to the system of the present invention, which preferably will display a message to the user saying that access has been denied.

The system of the present invention is able to increase the security of the system, while at the same time giving the individual user a large amount of power and flexibility. To give users the most power and flexibility, a standard object that has the capability to embed objects is preferably used. To allow users even more flexibility, a standard object tracking mechanism is preferably used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By being able to compartment every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

In addition to encryption, compression may be performed on objects prior to storage or transmission. Compression of the object can take place at any stage in the method of the present invention, but preferably takes place prior to encryp-

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

tion. According to the system of the present invention, a number of compression algorithms may be stored. The particular algorithm used to compress the object may be selected by the user, or may be chosen in a sequence or at random by the system.

Multi-level security is achieved because encrypted objects may be nested within other objects which are also encrypted, possibly within other objects, resulting in multiple layers of encryption. Multimedia security is achieved because objects of all types are encrypted. Where other encryption systems encrypt only files or other data, the system of the present invention encrypts any object, encompassing all forms of media. Thus, the nesting of individually encrypted objects provides security that is multilevel and multimedia.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a block diagram of the system of the present invention.

FIG. **2** shows a block diagram of the system of the present invention when an embedded encrypted object is activated.

FIG. **3** shows an object containing ten embedded encrypted objects at five various levels.

FIG. **4** shows an encrypted object that contains a web of embedded encrypted objects nested within it.

FIG. **5** shows a sample organizational chart.

FIG. **6** shows the present invention used in conjunction with the dynamic structure of a sample organizational chart.

FIG. **7** shows a larger, more complicated sample organizational chart.

FIG. **8** shows the present invention used in conjunction with the dynamic structure of the larger, more complicated sample organizational chart.

DETAILED DESCRIPTION OF THE INVENTION

Definitions

OOKeyMan® stands for the Object-Oriented Key Manager. OOKeyMan® is a Microsoft® Windows™ stand-alone application.

The Auto Application Interface is an intelligent front-end and back-end interface between a standard Microsoft® Windows™ application and OOKeyMan® or other key manager.

An Encrypted Embedded object is an encrypted OOKeyMan® object which can contain a single plain text object that has been encapsulated within the encrypted object, or it can contain an infinite web of encrypted objects matched with plain text objects or other cipher text objects.

A container object is an object that contains other objects (embedded objects). These embedded objects can be either cipher text or plain text. This is the transport vehicle for a standard object mechanism that embeds objects. A non-container object is an object that does not contain other objects.

Multi-Level Multimedia Security is defined as the ability to have simultaneous control over the knowledge and information flow of numerous media formats while allowing for clear data separation. At some levels the multi-level multimedia security becomes transparent. Examples of multimedia objects would include a file that contained two or more of the following: sound objects, video objects, graphic V, text objects, chart objects, table objects, and form objects.

Disclosure

The present invention, which shall be referred to as the Distributed Cryptographic Object Method ("DCOM"), is able to control which objects are visible to a specific user, which object attributes are inherited by other objects, which objects are available for use, and which level of system implementation can become transparent.

A function of the present invention is to securely manage and track encrypted objects. The present invention can securely manage and track a single encrypted object, or it can securely manage and track encrypted objects embedded within other encrypted objects. The capability to securely manage and track encrypted objects within other encrypted objects is only limited by storage space.

Referring to FIG. **1**, the system of the present invention is described. The system has a standard Multi-Level Security object interface **2** that interfaces with a plain text container object's encrypted embedded object(s) **4**. It does this through a standard application **6** that has the capability to embed an object in a container object, such as Microsoft® Object Package for Windows™. After the encrypted object (s) is/are embedded in a standard container object(s) **10** and the container object(s) **10** is/are encrypted, the original encrypted object(s) and the new encrypted container object (s) is/are ready for transport.

The new encrypted object(s) can be easily transported/routed without modification over any network that supports binary travel. The original encrypted objects can be deleted because all information from the original encrypted objects is encapsulated in the embedded encrypted object. All of the nested embedded encrypted objects will appear to a user as a single encrypted object until extracted with a standard object embedding/extracting mechanism through the process of the present invention. To activate an embedded encrypted object, the user simply selects the encrypted object to initiate the process, launching the OOKeyMan® application, as shown in FIG. **2**. The user/encrypted object authentication process is started and if the user/encrypted object is/are approved, encrypted object information can be returned and used by the user. Examples of such information include:

A. Plain Text Object Name
B. Plain Text Object Location
C. Plain Text Object Application
D. Plain Text Object Environment
E. Plain Text Object Date
F. Plain Text Object Time
G. Plain Text Object Digital Signature
H. Code word Object Tracking Label
I. Cluster Object Tracking Label
J. Device Object Tracking Label
K. Use Object Label
L. Algorithm Object Type

At this point, the authenticated user is given the option to decrypt the requested embedded encrypted object **12**. After decryption, a check is done to match the encrypted object's plain text object application to the correct Intelligent Auto Application Interface **14**. If the correct Auto Application Interface is not found, a notice is returned and the object is copied to a temporary location **16**; otherwise, the Auto Application Interface process is started. During this process, the encrypted object is matched to the appropriate authenticated application object **18** according to the returned encrypted object information. The correct authenticated

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

application object **18** is then activated with the plain text object **20**. Due to the relative dynamic nature of objects, the system of the present invention is able to accomplish all tasks "on the fly".

The scope of the DCOM directly correlates to the level at which the DCOM was embedded into the system. The scope of the DCOM covers the implemented embedded system level and all system levels above that, appearing transparent to all levels beneath the implemented embedded system level. For instance, if the present invention were embedded at the Open System Interconnection ("OSI") 7 Application layer, then the scope of the present invention would cover objects on that level and above. In this scenario, the present invention could run transparent to OSI levels 1 through 6. This implementation produces the most flexible DCOM. At this level and above, the system of the present invention is able to provide multi-level multimedia security while staying at the document level. This cross-application compatibility or document-level security is critical to the evolving component-based document centered computer system desktop. The present invention achieves cross-application multi-level multimedia security at the document level through its use of Object-Based Security.

The preferred implementation of the present invention at the application layer is OOKeyMan®. OOKeyMan® is a Microsoft® Windows™ 3.1 stand-alone application, but the present invention can be applied to other environments using other key managers. OOKeyMan® provides Document-Level Security through its use of Object Based Security.

Some examples of computer systems to which the method of the present invention can be applied to ensure the authentication and data integrity of objects include:

IBM® OS/2

IBM® System Object Method (SOM)

Microsoft® Object Package

Microsoft® Object Linking and Embedding (OLE)

Microsoft® Windows™ NT

Microsoft® Cairo Operating System

Microsoft® Chicago Operating System

Taligent (joint venture Operating System of Macintosh and IBM)

Macintosh® Compound Document Standard

Macintosh® Operating System

Novell®

Novell® Netware Directory Services (NDS)

Novell® APPWARE

Object Management CORBA (Common Object Request Broker Architecture)

Apple OPENDOC

Next PDO (Portable Distributed Objects System and Distributed Objects)

Unix Object-Oriented Systems

Virtual/Alternate Reality Systems

Other Object-Oriented Operating Systems

By applying the method of the present invention to the above example computer systems, the security of a system can be moved to a more abstract object level. By securing objects with cryptography, a level of security is achieved much higher than that of common access control mechanisms such as password or passphrase protection alone.

The steps for embedding an Encrypted Embedded OOKeyMan® object are as follows:

1. User Creates a plain text Object by using a standard application;

2. User Encrypts Object with OOKeyMan®;

3. User uses a standard Container Object;

4. Using Standard object to embed Encrypted Embedded OOKeyMan® object into Container Object;

5. Encrypt Container Object;

6. Repeat Steps 1 through 5 until all Objects are encrypted; 7. Multi-Level Multimedia Security achieved at the document level.

The same method may be followed using multiple encrypted embedded OOKeyMan® objects.

Examples of The Distributed Cryptographic Object Method

The next three examples demonstrate some of the capabilities of the present invention through the OOKeyMan® implementation. The following resources were used in these examples:

Software:

MS-DOS 5.0

Microsoft® Windows™ 3.1

Microsoft® Word for Windows™ 2.0c

Standard Microsoft Object Package

WordPerfect 5.2 for Windows™

OOKeyMan 1.Ob

Auto Application Interface for Word 1.0

Auto Application for WordPerfect 5.2 for Windows 1.0

Hardware:

486 50 MHz DX with 16 megabytes of RAM

The examples involve two of the most popular standard Microsoft® Windows™ applications on the market today: Word™ for Windows™ and WordPerfect™ for Windows™. The examples also use a standard object, Standard Microsoft Object Package, to embed the encrypted files in a standard container object. The interfaces used for these examples were the Auto Application Interface for Word for Windows 1.0 and the Auto Application for WordPerfect 5.2 for Windows 1.0.

The first example shows the ability of OOKeyMan® to securely manage and track singly or multiply embedded encrypted objects within other encrypted objects. This is done with a single application.

The second example shows the ability of OOKeyMan® to securely manage and track singly or multiply embedded encrypted objects within other encrypted objects. The embedded encrypted objects can even be part of encrypted objects from other applications. This example is performed in a cross-application manner between Word for Windows™ and WordPerfect for Windows™.

The third example shows the standard DCOM process.

EXAMPLE 1

Document Level Multi-Level Multimedia Security (using Microsoft Word for Windows™ and WordPerfect 5.2 for Windows™)

OOKeyMan® Process:

1. Lock Object

A. User creates an object(s) in Word for Windows™ or WordPerfect for Windows™;

B. User Initiates OOKeyMan® sequence;

C. User selects object(s) to encrypt;

D. User selects labels for object;

E. User selects an algorithm for encryption;

F. User selects lock object;

G. OOKeyMan® Object Manager performs Setup and Internal Checks;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

H. OOKeyMan® Object Manager Calls Key Management System object;
I. OOKeyMan® Object Manager Calls Cryptographic Algorithm object;
J. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
K. Encrypted OOKeyMan® Object Created;
i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type
ii. Results in New Encrypted Object being created
iii. Results in Plain Text Object Being Delete if Requested
L. OOKeyMan® Object Manager Returns To Word for Windows™ or WordPerfect for Windows™.
2. Unlock Object
A. User creates an encrypted object(s) using Word for Windows™ or WordPerfect for Windows™;
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan® Decrypt Object;
F. OOKeyMan® Object Manager performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager Calls Algorithm object;
I. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the decrypted object;
J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows™ or WordPerfect for Windows™.
3. Preview Object
A. User creates an encrypted object(s) in Word for Windows™ or WordPerfect for Windows™;
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan® Display Header Object;
F. OOKeyMan® Object Manager performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the Header object.

EXAMPLE 2

Cross-Application Multi-Level Multimedia Security at The Document Level

(Between Microsoft® Word for Windows™ and WordPerfect 5.2 for Windows™)
OOKeyMan® Process:
1. Lock Object
A. User creates an object(s) in Word for Windows™ or WordPerfect for Windows™;
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Encrypt;
D. User Selects Labels for object;

E. User selects an algorithm for encryption;
F. User Selects Lock Object;
G. OOKeyMan® Object Manager performs Setup and Internal Checks;
H. OOKeyMan® Object Manager Calls Key Management System object;
I. OOKeyMan® Object Manager Calls Cryptographic Algorithm object;
J. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
K. Encrypted OOKeyMan® Object Created;
i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type
ii. Results in New Encrypted Object being created
iii. Results in Plain Text Object Being Delete if Requested
L. OOKeyMan® Object Manager Returns To Word for Windows™ or WordPerfect for Windows™.
2. Unlock Object
A. User creates an encrypted object(s) in Word for Windows™ or WordPerfect for Windows™;
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan® Decrypt Object;
F. OOKeyMan® Object Manager performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager Calls Algorithm object;
I. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the decrypted object;
J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows™ or WordPerfect for Windows™.
3. Preview Object
A. User creates an encrypted object(s) in Word for Windows™ or WordPerfect for Windows™;
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan® Display Header Object;
F. OOKeyMan® Object Manager Performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the Header object.

EXAMPLE 3

Standard Distributive Cryptographic Object Method Process (DCOMP)

OOKeyMan® Process:
1. Lock Object
A. User creates an object(s);
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Encrypt;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

D. User Selects Labels for object;
E. User Selects an algorithm for encryption;
F. User Selects Lock Object;
G. OOKeyMan® Object Manager Performs Setup and Internal Checks;
H. OOKeyMan® Object Manager Calls Key Management System object;
I. OOKeyMan® Object Manager Calls Cryptographic Algorithm object;
J. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
K. Encrypted OOKeyMan® Object Created;
i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type
ii. Results in New Encrypted Object being created
iii. Results in Plain Text Object Being Delete if Requested
L. OOKeyMan® Object Manager Returns To Application Object.
2. Unlock Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan® Decrypt Object;
F. OOKeyMan® Object Manager Performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager Calls Algorithm object;
I. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
3. Preview Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan® sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan® Display Header Object;
F. OOKeyMan® Object Manager Performs Setup and Internal Checks;
G. OOKeyMan® Object Manager Calls Key Management System object;
H. OOKeyMan® Object Manager waits for Selected Algorithm object to finish and create the Header object.

The process of the present invention can be applied to a vast number of areas in the real world. Whether it be the physical topology of the local area network/wide area network environment or the dynamic structure of an organization, the DCOM process will change dynamically to reflect the current state of the object in question.

FIG. 3 and FIG. 4 show an encrypted object that contains a web of embedded encrypted objects nested within the other encrypted objects. The object shown in FIG. 3 contains ten embedded encrypted objects at five various levels. The encrypted object embedded in level 5 was embedded in an object in level four, level four objects in level 3 and so on. The plain text object containing the level 5 encrypted object can then be encrypted for further security. This single encrypted object encapsulates all of the data associated with the encrypted objects within it and therefore the entire encrypted object can then be sent out via any transport mechanism supporting binary file transfer.

FIG. 4 shows an encrypted object that contains a web of embedded encrypted objects nested within it. All of the attached embedded encrypted objects are fused together resulting in a single encapsulated encrypted object. The system of the present invention is powerful enough to dynamically adapt to accommodate N dimensional objects.

The single encrypted objects shown in both FIGS. 3 and 4 can act as a secure package and can be sent out for distribution to an entire organization (e.g., E-mail). This single encrypted object can represent a branch(s), department(s), or even an entire company. Every employee would receive the single encrypted file, but they would only be able to unravel the portions that corresponded to them and acquire no knowledge of other existing embedded encrypted objects. For example, FIG. 5 displays a sample organization chart. When applied, the system of the present invention will control the knowledge/information flow of the organization and will allow for clear data separation, further compartmentalization through multiple algorithm use, and document-level security. With the improved communication paths, an organization would become more efficient. FIG. 6 demonstrates the use of the DCOM in conjunction with the dynamic structure of a sample organization. Since the system of the present invention is dynamic in nature, it can adapt to any organizational size or type (For example, see FIGS. 7 and 8).

Preferred and alternate embodiments of the present invention have now been described in detail. It is to be noted, however, that this description of these specific embodiments is merely illustrative of the principles underlying the inventive concept. It is therefore contemplated that various modifications of the disclosed embodiments will, without departing from the spirit and scope of the invention, be apparent to persons of ordinary skill in the art.

What is claimed is:

1. A method for providing multi-level multimedia security in a data network, comprising:
a) accessing an object-oriented key manager;
b) selecting a first object to encrypt;
c) selecting a first label for the first object;
d) encrypting the first object;
e) labelling the encrypted first object;
f) displaying the first label as a header array;
g) reading the first object label;
h) determining access authorization based on the first object label; and
i) decrypting the first object if access authorization is granted.

2. The method of claim 1, further comprising embedding the encrypted first object in a second object after labelling the encrypted first object.

3. The method of claim 2, further comprising:
a) selecting a second label for the second object;
b) encrypting the second object;
c) labelling the encrypted second object with a second object label; and
d) incorporating the second label into the header array.

4. The method of claim 3, further comprising:
a) reading the header array;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

b) determining access authorization based on the second object label; and

c) decrypting the second object if access authorization is granted.

5. The method of claim **4**, further comprising presenting a representation of the first object only if access authorization is granted according to the second object label.

6. The method of claim **2**, wherein the encrypted first object is a printer object.

7. The method of claim **1**, wherein the first object is an application document, and further comprising:

a) creating the first object in an application prior to accessing the object-oriented key manager; and

b) returning the encrypted first object to the application prior to reading the first object label.

8. The method of claim **7**, wherein access authorization is print-only authorization.

9. The method of claim **1**, further comprising:

a) compressing the first object prior to encrypting the first object; and

b) decompressing the first object after decrypting the first object.

10. The method of claim **1**, further comprising:

a) compressing the first object immediately prior to encrypting the first object; and

b) decompressing the first object immediately after decrypting the first object.

11. The method of claim **1**, further comprising:

a) arranging a plurality of encrypted objects such that at least some of the plurality of encrypted objects are embedded within others of the encrypted objects;

b) selecting a label for each of at least some of the plurality of encrypted objects;

c) labelling each of the plurality of encrypted objects for which a label was selected;

d) incorporating each of the labels into the header array.

12. The method of claim **11**, further comprising:

a) reading a selected label;

b) determining access authorization to the object associated with the selected label based on the selected label and further based on all labels associated with objects in which the associated object is embedded; and

c) decrypting the associated object if access authorization is granted.

13. The method of claim **1**, wherein the header array is an access control list separate from the encrypted first object.

14. A system for providing multi-level multimedia security in a data network, comprising:

A) digital logic means, the digital logic means comprising:

1) a system memory means for storing data;

2) an encryption algorithm module, comprising logic for converting unencrypted objects into encrypted objects, the encryption algorithm module being electronically connected to the system memory means for accessing data stored in the first system memory;

3) means for arranging a plurality of encrypted objects such that at least some of the plurality of encrypted objects are embedded within others of the encrypted objects;

4) an object labelling subsystem, comprising logic means for limiting object access, subject to label conditions of a selected object and further subject to conditions of all labels associated with objects in which the selected object is embedded, the object labelling subsystem being electronically connected to the system memory means for accessing data stored in the system memory means and the object labelling subsystem being further electronically connected to the encryption algorithm module to accept inputs from the encryption algorithm module;

5) a decryption algorithm module, comprising logic for converting encrypted objects into unencrypted objects, the decryption algorithm module being electronically connected to the system memory means for accessing data stored in the system memory means; and

6) an object label identification subsystem, comprising logic for limiting object access, subject to label conditions, the object label identification subsystem being electronically connected to the system memory means for accessing data stored in the system memory means and the object label identification subsystem being further electronically connected to the decryption algorithm module to accept inputs from the deception algorithm module.

B) the encryption algorithm module working in conjunction with the object labelling subsystem to create an encrypted object such that the object label identification subsystem limits access to an encrypted object.

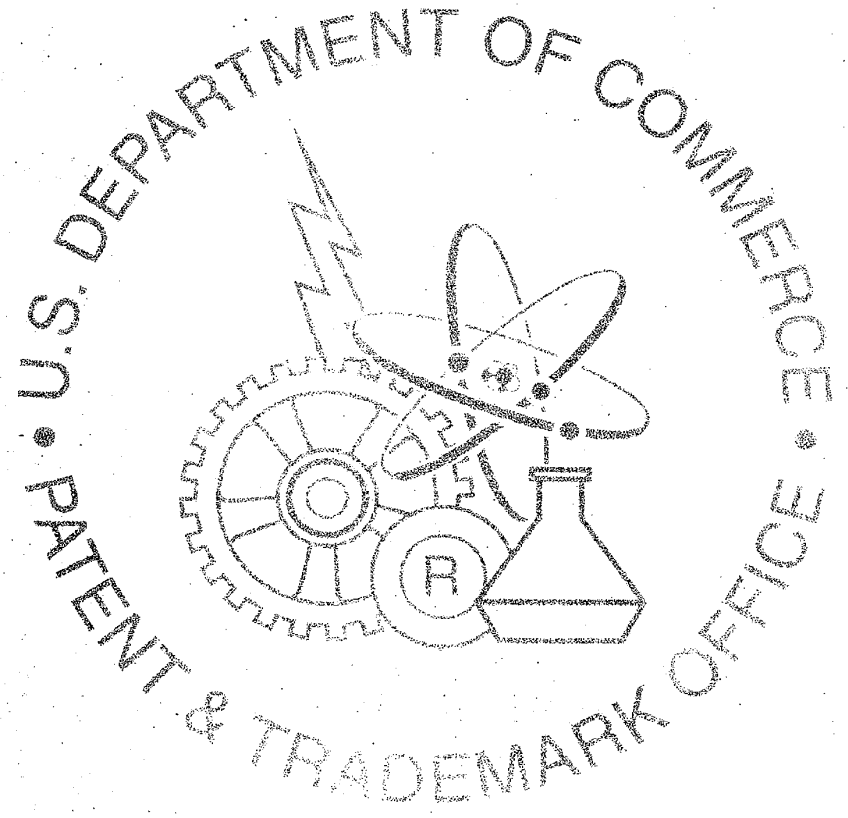
15. The system of claim **14**, wherein the digital logic means further comprises means for accessing computer program applications stored in the system memory means.

16. The system of claim **14**, wherein one of the encrypted objects is a printer object.

17. The system of claim **14**, further comprising means for compressing one of the encrypted objects and means for decompressing one of the encrypted objects.

* * * * *

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



TEC00140794

PTO-1683
(Rev. 7-96)

U 7214101

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:
UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

November 25, 2009

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,717,755*
ISSUE DATE: *February 10, 1998*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

P. SWAIN
Certifying Officer

UNITED STATES PATENT AND TRADEMARK OFFICE
DEPARTMENT OF COMMERCE

Plaintiff's Trial Exhibit
3
TecSec. v. Adobe Case no. 1:10cv115

TEC00140796

US005717755A

United States Patent [19]

Shanton

[11] Patent Number: 5,717,755

[45] Date of Patent: *Feb. 10, 1998

[54] **DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD**

[75] Inventor: **M. Greg Shanton**, Fairfax, Va.

[73] Assignee: **TECSEC,Inc.**, Vienna, Va.

[*] Notice: The term of this patent shall not extend beyond the expiration date of Pat. No. 5,369,702.

[21] Appl. No.: **304,867**

[22] Filed: **Sep. 13, 1994**

Related U.S. Application Data

[63] Continuation of Ser. No. 138,857, Oct. 18, 1993, Pat. No. 5,369,702.

[51] **Int. Cl.**[6] **H04L 9/32**; H04L 9/00

[52] **U.S. Cl.** **380/25**; 380/4; 380/9; 380/21; 380/23; 380/28; 380/49; 380/50; 340/825.31; 340/825.34

[58] **Field of Search** 380/3, 4, 5, 9, 380/10, 21, 23, 24, 25, 28, 30, 43, 49, 50; 340/825.31, 825.34; 455/33.3

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,218,582	8/1980	Hellman et al.	380/30
4,405,829	9/1983	Rivest et al.	380/30
4,424,414	1/1984	Hellman et al.	380/30
4,713,753	12/1987	Boebert et al.	380/4 X
4,864,616	9/1989	Pond et al.	380/25
4,955,082	9/1990	Hattori et al.	455/33.3
4,962,533	10/1990	Krueger et al.	380/25
4,984,272	1/1991	McIlroy et al.	380/25
5,052,040	9/1991	Preston et al.	380/4
5,065,429	11/1991	Lang	380/25
5,191,611	3/1993	Lang	380/25
5,204,961	4/1993	Barlow	380/4 X
5,369,702	11/1994	Shanton	380/4
5,369,707	11/1994	Follendore, III	380/25

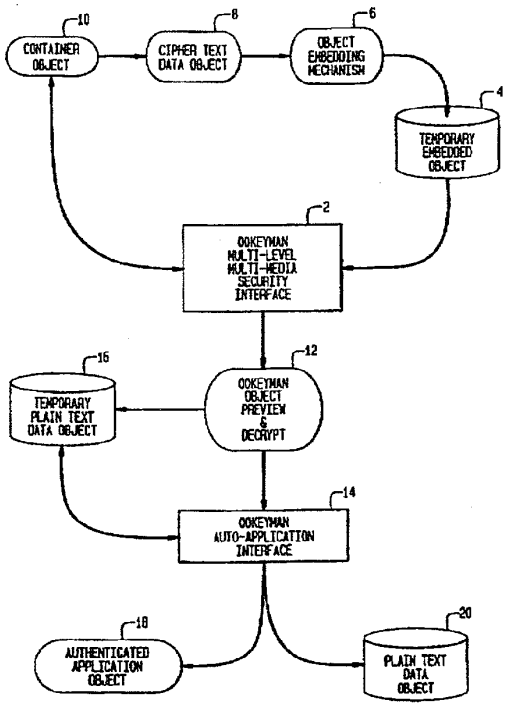
Primary Examiner—Bernarr E. Gregory

Attorney, Agent, or Firm—Rabin, Champagne & Lynt, P.C.

[57] **ABSTRACT**

A system for increasing the security of a computer system, while giving an individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By effecting compartmentalization of every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

20 Claims, 8 Drawing Sheets



Copy provided by USPTO from the PIRS Image Database on 11/24/2009

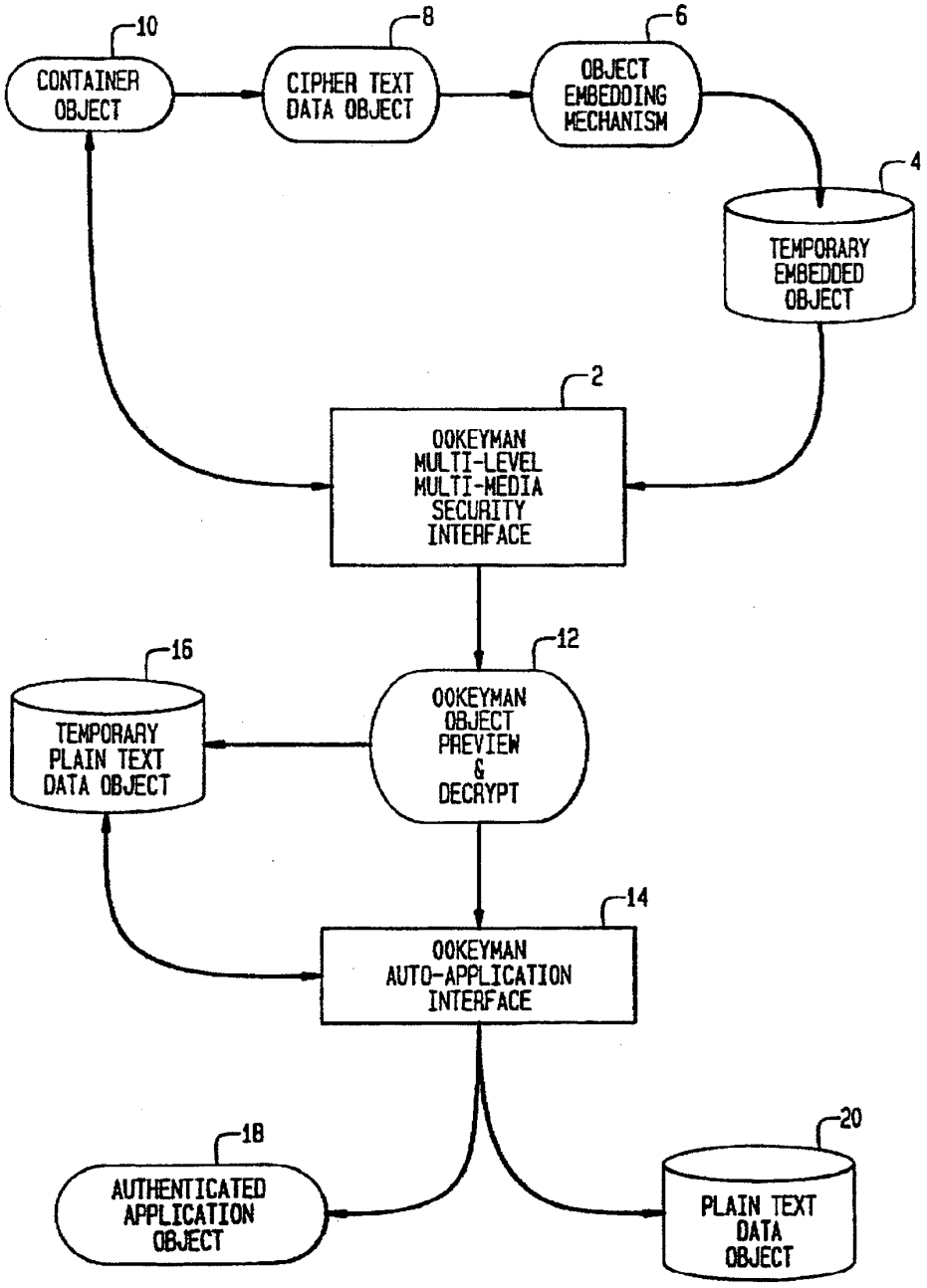


FIG. 1

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

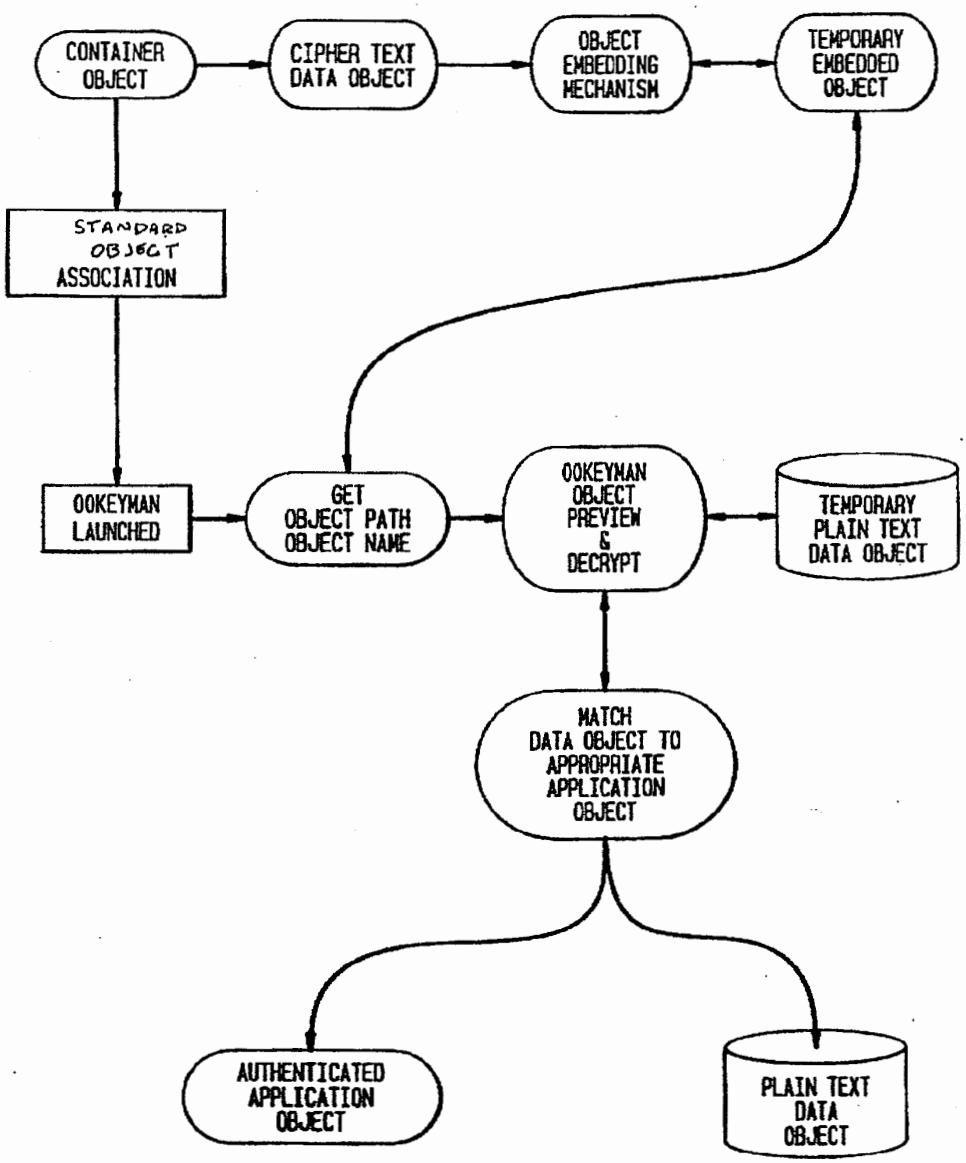


FIG. 2

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

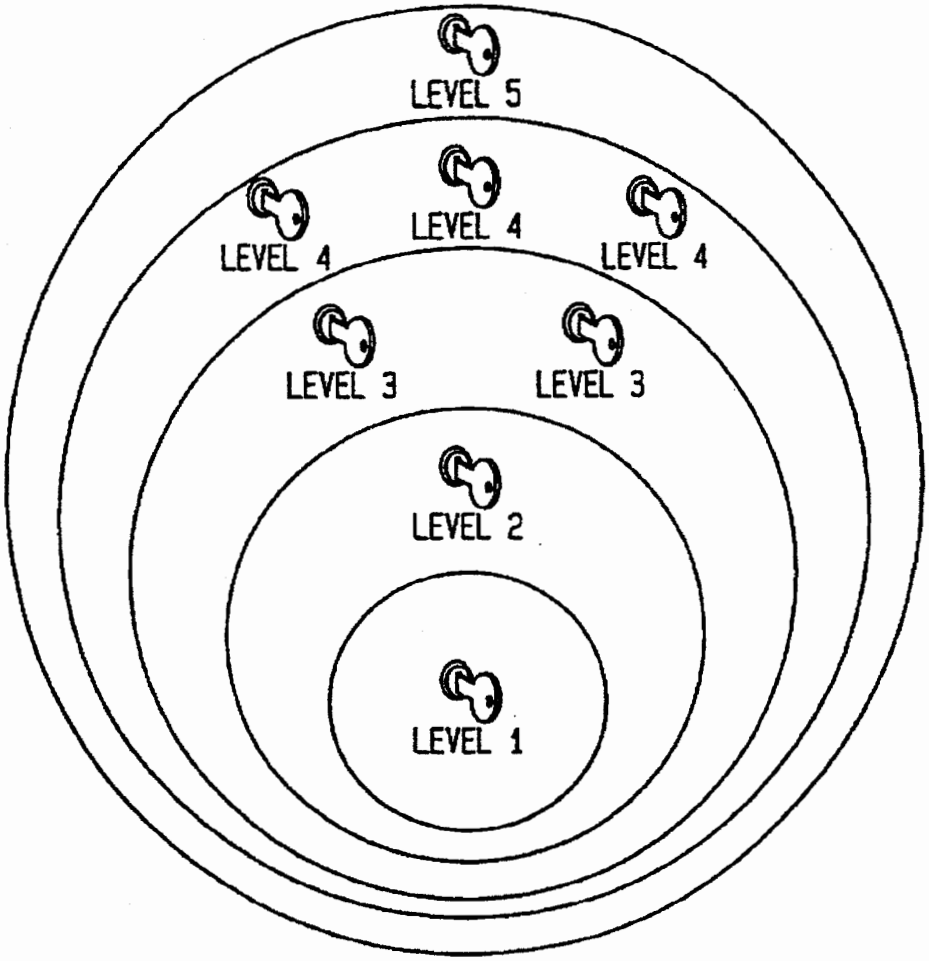


FIG. 3

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

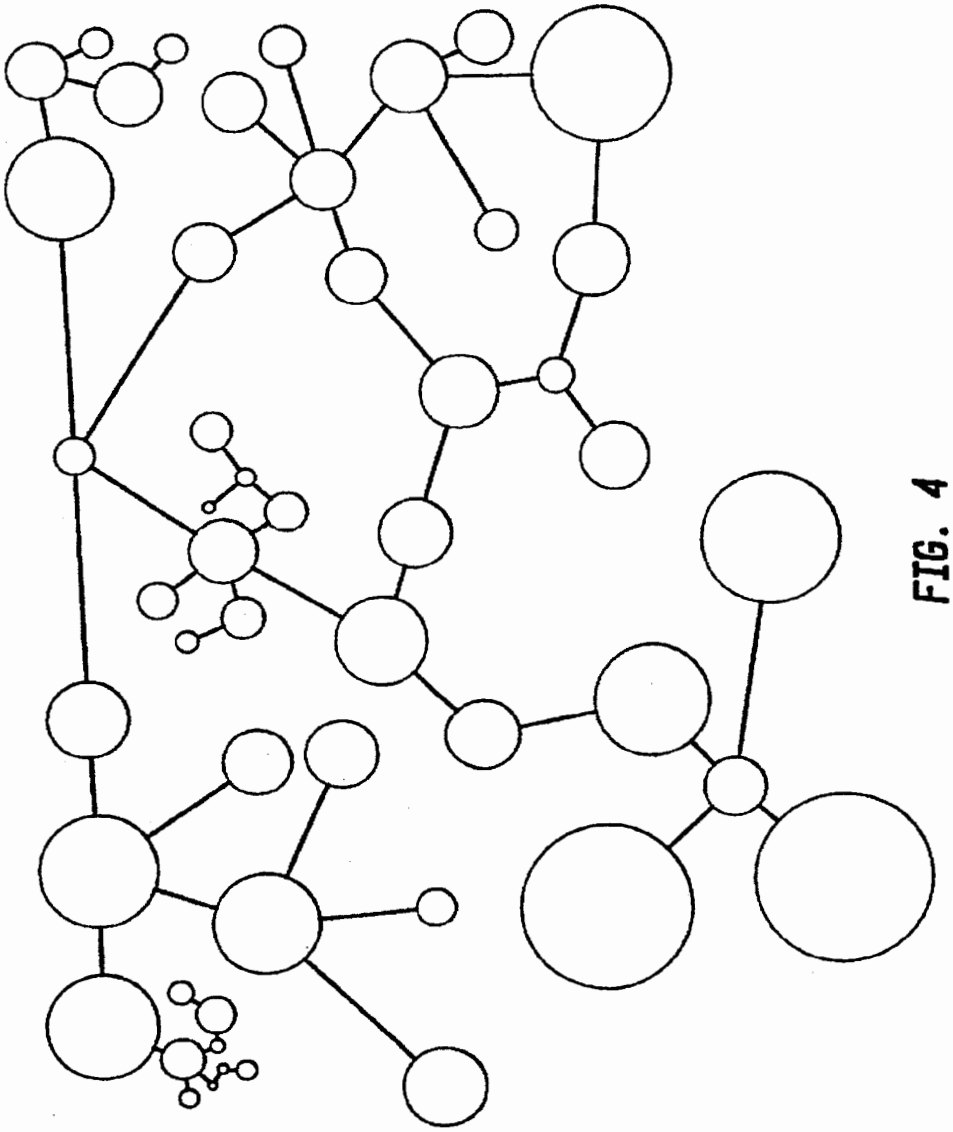


FIG. 4

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

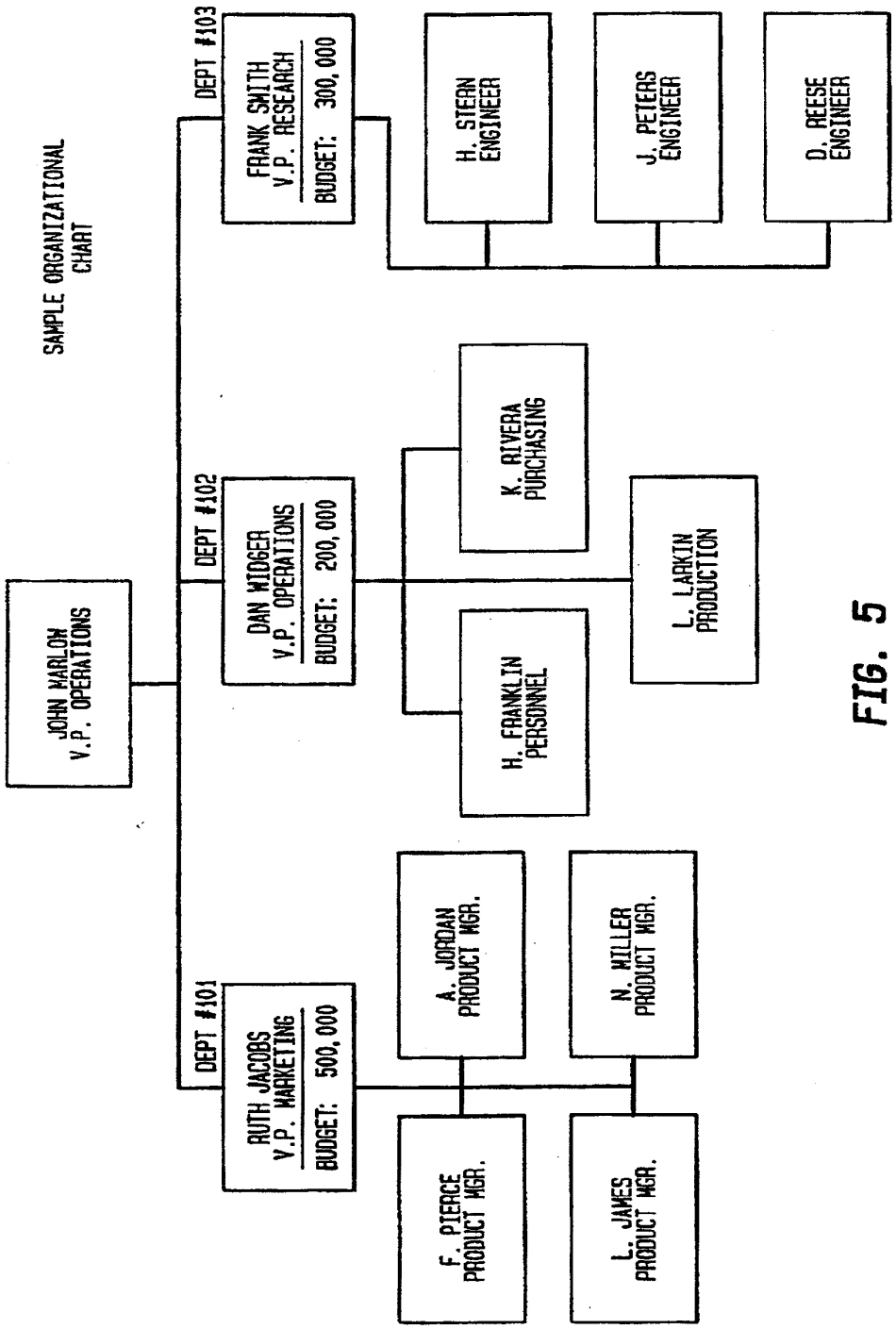


FIG. 5

Case: 19-2192 Document: 20 Page: 158 Filed: 10/07/2019

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

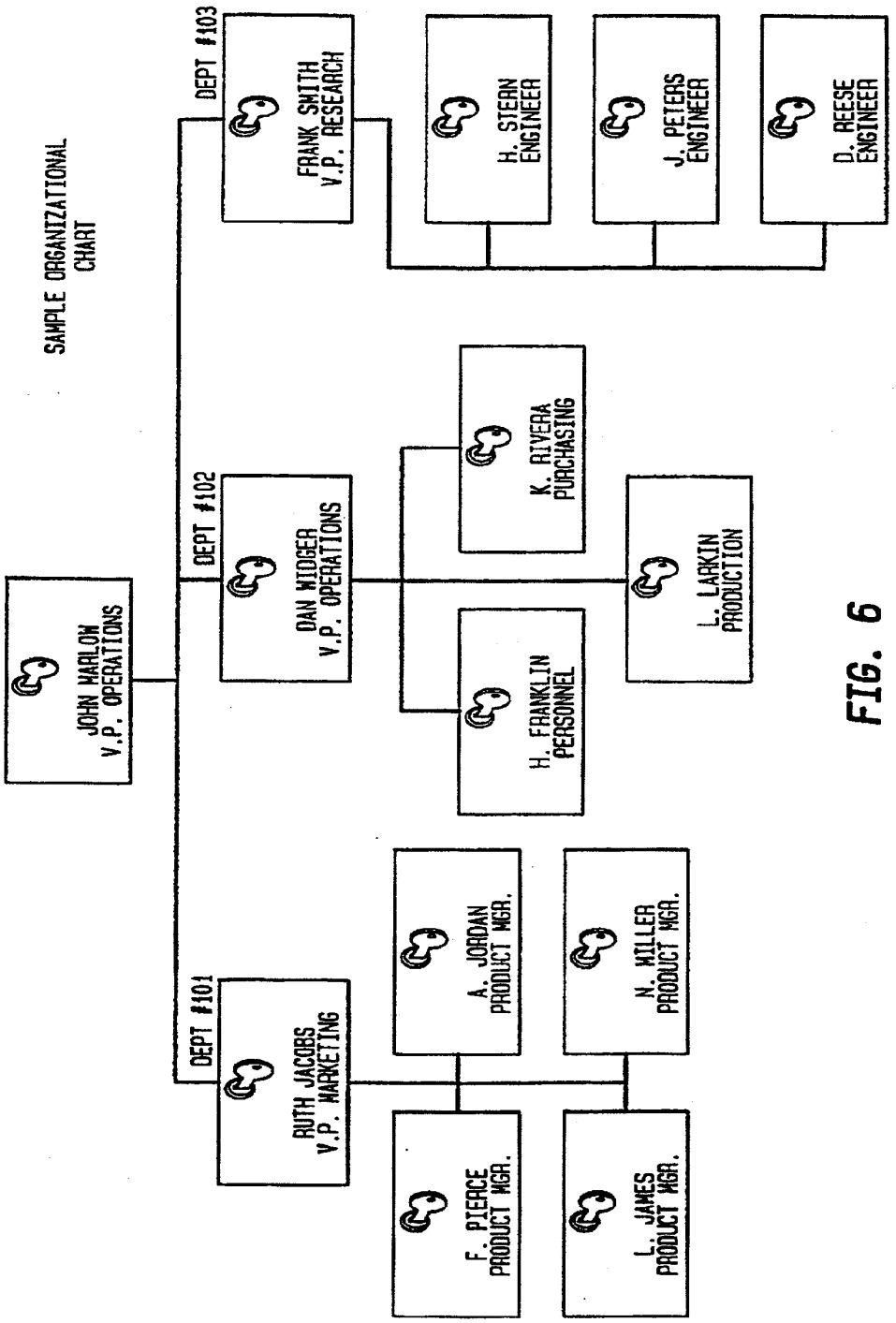
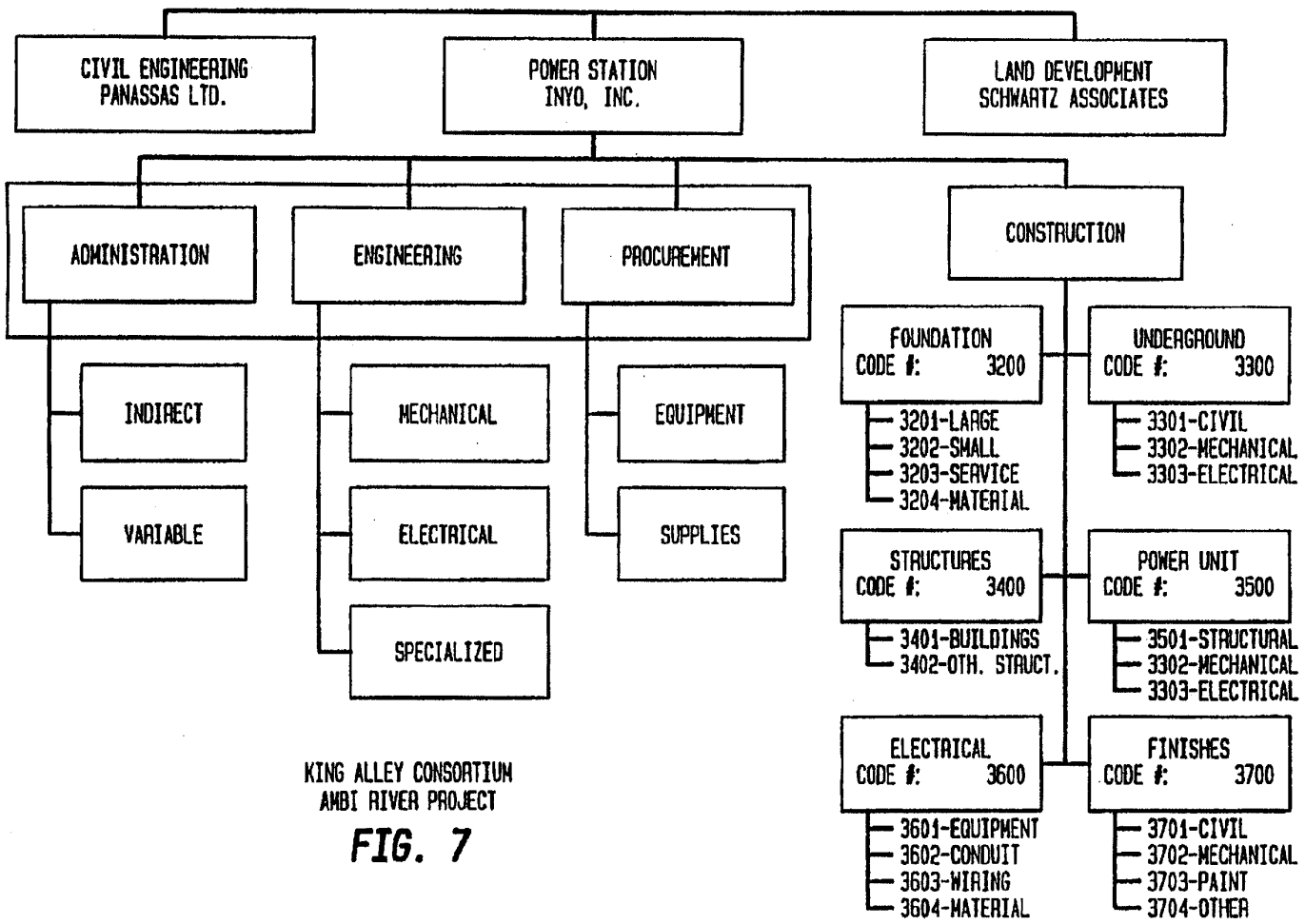


FIG. 6

Case: 19-2192 Document: 20 Page: 159 Filed: 10/07/2019

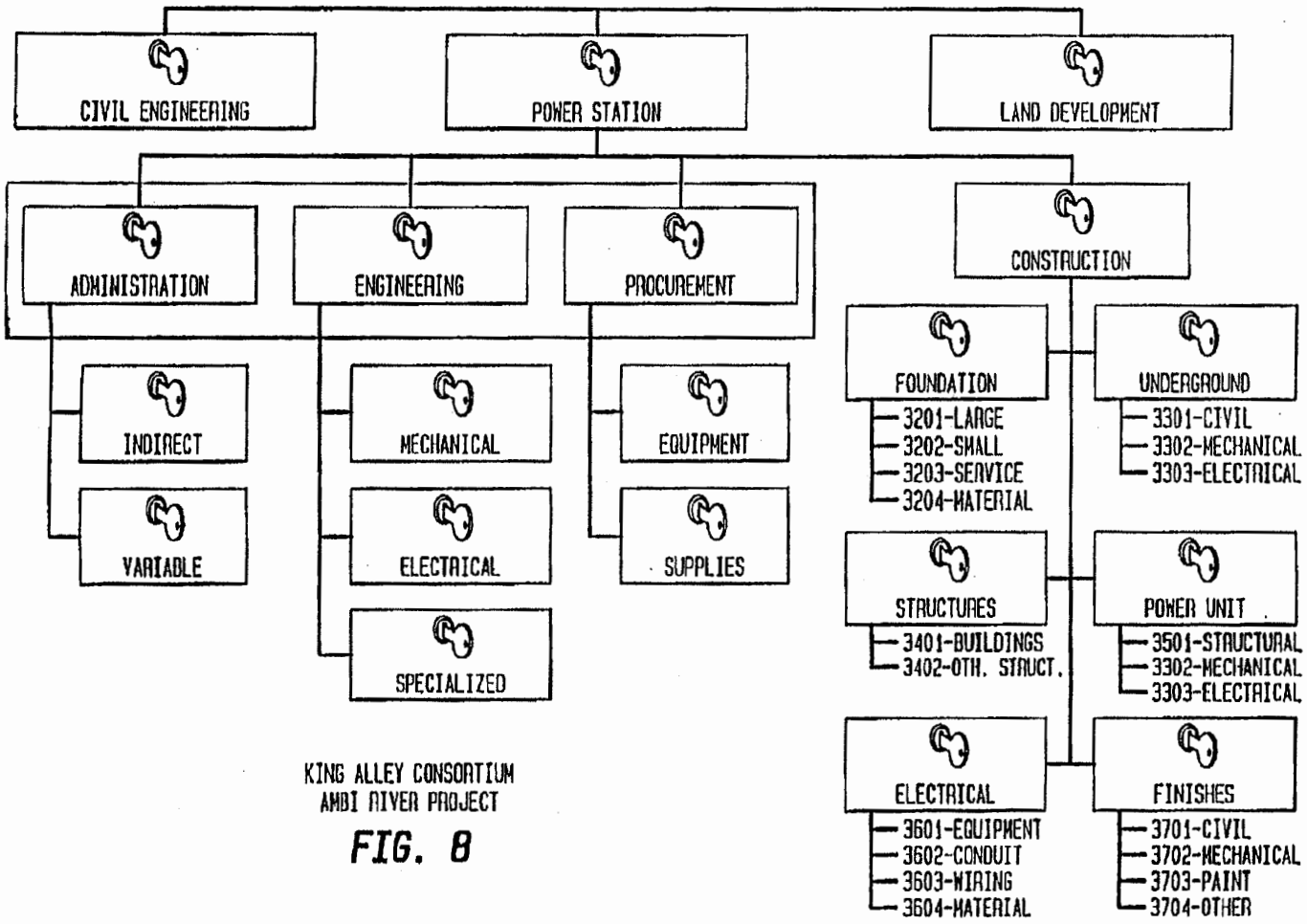
Copy provided by USPTO from the PIRS Image Database on 11/24/2009



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 7

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD

This is a continuation of U.S. patent application Ser. No. 08/138,857 filed Oct. 18, 1993; now U.S. Pat. No. 5,363,702.

FIELD OF THE INVENTION

The present invention relates generally to a system that can be used to restrict access to computer data. In particular, the system of the present invention restricts access in a flexible way, identifying objects for restriction and nesting restriction requirements through the use of embedded objects.

BACKGROUND OF THE INVENTION

While the specter of "spies" eagerly trying to obtain the defense information of various countries is very much still present in the defense and intelligence community, an equally massive threat now exists from technological or commercial "spies" who desire to obtain commercial and technical information from competing companies. These agents use sophisticated means similar to those used by the defense and intelligence community in order to obtain commercially valuable information that reveals the plans and commercial activities of competitors thereby allowing the aggressor company to obtain a competitive advantage in the marketplace. Theft of commercially valuable information is a very real and ever present threat.

To combat this type of commercial spying, various complex systems have evolved to protect company proprietary information. These systems involve physical controls over personnel as well as over the data flowing in and out of a company. For example, most computer systems used within companies require a password to be entered before the system can be accessed. It is frequently the case that confidential or company proprietary information must be passed electronically from one location to another in order to convey that information within the company in a timely fashion. Such electronic communication is easily susceptible to interception if not protected in some other form.

Cryptographic systems have evolved to fill the needs of companies and individuals wanting to protect the proprietary commercial information of a company from competitors and those who generally should not have that information. Encryption of data is therefore a critical requirement in denying access to confidential information from those who are not so authorized. Cryptographic "keys" are an essential part of the information encryption process. The cryptographic key, or "key" for short, is a sequence of letters, numbers, or bytes of information which are manipulated by a cryptographic algorithm to transform data from plain (readable) text to a series of unintelligible text or signals known as encrypted or cipher text. The key is then used by the receiver of the cipher text to decrypt the message back to plain text. However, for two people to communicate successfully using keys, each must use the same key, assuming that the same encryption/decryption algorithm is used on both ends of the communication.

Various methods have evolved to manage the distribution of keys. Such methods of distribution are collectively referred to as "key management" The function of key management is to perform the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys. Under normal operational circumstances, the key manager begins and ends a cryptographic session by controlling access to the algorithm used to encrypt and decrypt plain text objects. Thus, a user who wants to encrypt an object or decrypt an object must first access the key manager so that an encryption algorithm may be chosen.

Simple encryption of data being communicated between two points only provides one level of security, however. Encryption limits data communication to those who have the key. Anyone who has the key is privy to any communication at any location. That is, if a group of people are working on a particular project, they will all presumably share a key for decrypting information relating to the project. Some of the project group may be working in one location, while the rest of the group may be located in a distant city. If one member of the group wants to send a communication to a particular member in the other city, the key will afford him no protection because everyone in the project shares the same key. Likewise, if someone wants to communicate a message to a subset of the group, for example, only to management personnel, this key would again provide her with no extra security. In another case, someone may want to send a message that is capable of being read only at a particular computer terminal, or of being printed only at a particular printer. In these and other cases, multilevel multimedia key access, or individual keys issued to each person, would provide a solution, albeit one that is quite unwieldy, inflexible, and difficult to manage by a security officer or key administrator.

A secure method of labelling files or messages that are sent from a sending user to a receiving user over a network can provide a level of protection in addition to cryptographic protection. A file "label" for purposes of this invention means a series of letters or numbers, which may or may not be encrypted, separate from but associated with the sending of a message, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated message. Using a secure labelling regimen, a network manager or user can be assured that only those messages meant for a certain person, group of persons, and/or location(s) are in fact received, decrypted, and read by the intended receiver. Thus, a sending user can specify label conditions that limit access to the transmitted message. For example, many people within a company may have the key necessary to read a data file that a sender may transmit from his computer terminal to other terminals at another site within his company. The sender may, however, wish to restrict reception to those persons present at a particular terminal. By employing a secure labelling technique in addition to encryption, the sender can be assured that people having the correct key to decrypt the message but working at different terminals will not receive or be allowed to access the communication. Access may be limited to particular people as well.

A system such as that described above is disclosed in U.S. patent application Ser. No. 08/009,741, Filed Jan. 27, 1993, the specification of which is incorporated by reference herein, now U.S. Pat. No. 5,389,707.

A system that can limit access on an object level would be more flexible and would offer still more protection. Access could be specified on an object-by-object basis, and objects could be embedded within other objects, providing an access hierarchy for users.

The ability to cryptographically secure objects ensures the authentication and data integrity of the particular object or objects in question. If a device were able to cryptographically control an object(s) or nested object(s), then that device

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

would have total control over the entire object and all other objects within it. This type of control over the knowledge/information flow would allow for clear data separation, and at some levels could become a transparent method. A system that is able to do this would be able to achieve multi-level multimedia security.

SUMMARY OF THE INVENTION

It is therefore an objective of the present invention to provide a system to insure that properly specified kinds of information in a network system flows only to designated locations and to further insure that such information is only read by those individuals who are designated to review that information.

It is a further objective of the present invention to provide a system that recognizes objects and permits or denies access on the object level.

It is an additional objective of the present invention to provide a system in which objects may be embedded within other objects, resulting in an access hierarchy for users of the system.

It is another objective of the present invention to provide a system in which access control is transparent to the user.

These and other objectives and advantages of the present invention will be apparent to those of ordinary skill in the art upon inspection of the detailed description, drawings, and appended claims.

The definition and concept of objects varies greatly depending on with whom you consult. Everything around you in your daily life is an object. Your car, your car keys, books, people, desks, etc. objects are entities by themselves, but they may contain other objects, in either single or multiple configurations. Objects can change their make up dynamically by inheritance. Objects can inherit the attributes of other objects and the inheritance features can change dynamically "on the fly" during the operation of the objects.

In the context of the present invention, an object can come in a vast number of forms, shapes or sizes and can be either passive or active, dynamic or static. An object may stay dormant until it is acted upon, or it may be an active participant, dynamically auditing and verifying every transaction that occurs in a system. Examples of what an object can be include a bit of information, a byte of information, Sound Clips, Video Clips, Graphic Images, text, charts, tables, forms, controls, MDIForms, variables, executable files, video files, binary files, text files, data files, container files, graphic files, application file(s), Library files, a directory, a collection of directories, a hard disk, multiple hard disks, any hardware component, any software component, a complete computer system, a single network, multiple networks.

Thus, an object is any distinct, separate entity. In a computer or data communication context, entities that may be treated as objects include:

1) Program objects, representing applications such as word processors, spreadsheets, games, etc., as well as utilities and operating systems;
2) Folder objects, representing collections of other objects;
3) Data file objects, including information such as text, memos, letters, spreadsheets, video, and sound; and
4) Device objects, such as printers, fax modems, plotters, and CD-ROM drives.

In object linking and embedding, an object can be any userselected group of data, such as a block of text, a set of spreadsheet cells, a chart, sounds, or a graphical image. This data can be embedded in or linked to another document created by a different application. For example, a folder may represent a directory and may contain a group of files, or it may represent a group of programs. Folders can also contain other folders.

In object-oriented programming, a program consists of a set of related but self-contained objects that can contain both code and data.

The present invention is able to increase the security of the system, while at the same time giving the individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By being able to compartment every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

Multi-level security is achieved because encrypted objects may be nested within other objects which are also encrypted, possibly within other objects, resulting in multiple layers of encryption. Multimedia security is achieved because objects are encrypted. Where other encryption systems encrypt only files or other data, the system of the present invention encrypts any object, encompassing all forms of media. Thus, the nesting of individually encrypted objects provides security that is multi-level and multimedia.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a block diagram of the system of the present invention.

FIG. **2** shows a block diagram of the system of the present invention when an embedded encrypted object is activated.

FIG. **3** shows an object containing ten embedded encrypted objects at five various levels.

FIG. **4** shows an encrypted object that contains a web of embedded encrypted objects nested within it.

FIG. **5** shows a sample organizational chart.

FIG. **6** shows the present invention used in conjunction with the dynamic structure of a sample organizational chart.

FIG. **7** shows a larger, more complicated sample organizational chart.

FIG. **8** shows the present invention used in conjunction with the dynamic structure of the larger, more complicated sample organizational chart.

DETAILED DESCRIPTION OF THE INVENTION

Definitions

OOKeyMan stands for the Object-Oriented Key Manager. OOKeyMan is a Microsoft Windows stand alone application.

The Auto Application Interface is an intelligent front end and back end interface between a standard Microsoft Windows 3.1 application and OOKeyMan.

An Encrypted Embedded object is an encrypted OOKeyMan object which can contain a single plain text object that has been encapsulated within the encrypted object, or it can contain an infinite web of encrypted objects matched with plain text objects or other cipher text objects.

A container object is an object that contains other objects. These objects can be either cipher text or plain text. This is

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

the transport vehicle for a standard object mechanism that embeds objects. A non-container object is an object that does not contain other objects.

Multi-Level Multimedia Security is defined as the ability to have simultaneous control over the knowledge/information flow of numerous media formats while allowing for clear data separation. At some levels the multi-level multimedia security becomes transparent. Examples of multi-media objects would include a file that contained two or more of the following: sound objects, video objects, graphic V. text objects, chart objects, table objects, and form objects.

Disclosure

The present invention, known as the Distributed Cryptographic Object Method ("DCOM"), is able to control which objects are visible to a specific user, which object attributes are inherited by other objects, which objects are available for use, and which level of system implementation can become transparent.

The main function of the DCOM is to securely manage and track encrypted objects. The DCOM can securely manage and track a single encrypted object, or it can securely manage and track encrypted objects embedded within other encrypted objects. The capability to securely manage and track encrypted objects within other encrypted objects is only limited by storage space.

Referring to FIG. 1, the DCOM system is described. The DCOM has a standard Multi-Level Security object interface **2** that interfaces with the plain text container object's encrypted embedded object(s) **4**. It does this through a standard application **6** that has the capability to embed an object in a container object, such as Microsoft's Object Package for Windows. After the Encrypted object(s) is/are embedded in a standard container object(s) **10** and the container object(s) **10** is/are encrypted, the original encrypted object(s) and the new encrypted container object (s) is/are ready for transport.

The new encrypted object(s) can be easily transported/routed over any network that supports binary travel without modification. The original encrypted objects can be deleted because all information from the original encrypted objects is encapsulated in the embedded encrypted object. All of the nested embedded encrypted objects will appear to a user as a single encrypted object until extracted with a standard object embedding/extracting mechanism through the DCOM process. To activate an embedded encrypted object, the user simply selects the encrypted object to initiate the DCOM process, launching the OOKeyMan application, as shown in FIG. **2**. The user/encrypted object authentication process is started and if the user/encrypted object is/are approved, the following encrypted object information can be returned and used by the user:

A. Plain Text Object Name
B. Plain Text Object Location
C. Plain Text Object Application
D. Plain Text Object Environment
E. Plain Text Object Date
F. Plain Text Object Time
G. Plain Text Object Digital Signature
H. Code word Object Tracking Label
I. Cluster Object Tracking Label
J. Device Object Tracking Label
K. Use Object Label
L. Algorithm Object Type

At this point, the authenticated user is given the option to decrypt the requested embedded encrypted object **12**. After decryption, a check is done to match the encrypted object's plain text object application to the correct Intelligent Auto Application Interface **14**. If the correct Auto Application Interface is not found, a notice is returned and the object is copied to a temporary location **16**, otherwise the Auto Application Interface process is started. During this process the encrypted object is matched to the appropriate authenticated application object **18** according to the returned encrypted object information. The correct authenticated application object **18** is then activated with the plain text object **20**. Due to the relative dynamic nature of objects, the DCOM is able to accomplish all tasks "on the fly".

The scope of the DCOM directly correlates to the level at which the DCOM was embedded into the system. The scope of the DCOM would cover the implemented embedded system level and all system levels above that, appearing transparent to all levels beneath the implemented embedded system level. For instance, if the DCOM were was embedded at the Open System Interconnection ("OSI") **7** Application layer, then the scope of the DCOM would cover objects on that level and above. In this scenario, the DCOM could run transparent to OSI levels **1** through **6**. On the basis of current technology, this implementation would produce the most flexible DCOM. At this level and above, the DCOM is able to provide multi-level multi-media security while staying at the document level. This cross-application compatibility or document-level security is critical to the evolving component based document centered computer system desktop. The DCOM achieves cross-application multi-level multi-media security at the document level through its use of Object-Based Security.

The current implementation of the DCOM at the application layer is called the Object-Oriented Key Manager (OOKeyMan). Currently, OOKeyMan is a Microsoft Windows **3.1** stand alone application, but the DCOM can be applied to other environments. OOKeyMan provides Document-Level Security through its use of Object Based Security.

Some examples of where the DCOM can be applied to ensure the authentication and data integrity of objects include:

IBMt's OS/2 2.X and above
IBMt's System Object Method(SOM)
Microsoft's Object Package
Microsoft's Object Linking and Embedding 1.0(OLE 1.0)
Microsoft's Object Linking and Embedding 2.0(OLE 2.0)
Microsoft's Windows NT 3.1
Microsoft's Cairo (Future Operating System)
Microsoft's Chicago (Future Operating System)
Taligent (joint venture future Operating System of Macintosh and IBM)
Macintosh's Compound Document Standard
Macintosh Operating System
Novell
Novell Netware Directory Services(NDS)
Unix Object-Oriented Systems
Virtual/Alternate Reality Systems
Future Object-Oriented Operating Systems

By applying the DCOM to the above examples, the security of a system can be moved to a more abstract object level. By securing objects with cryptography, a level of

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

security is achieved much higher than that of common access control mechanisms such as password or pass phrase protection.

The steps for embedding an Encrypted Embedded OOKeyMan Object(s) are as follows:

1. User Creates a plain text Object by using a standard application;
2. User Encrypts Object(s) with OOKeyMan;
3. User uses a standard Container Object;
4. Using Standard object to embed Encrypted Embedded OOKeyMan object(s) into Container Object;
5. Encrypt Container Object;
6. Repeat Steps 1 through 5 until all Objects are encrypted;
7. Multi-Level Multimedia Security achieved at the document level.

EXAMPLES OF THE DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD

The following resources were used in the following examples:

Software:

MS-DOS 5.0

Microsoft Windows 3.1

Microsoft Word for Windows 2.0c

Standard Microsoft Object Package

WordPerfect 5.2 for Windows

OOKeyMan 1.0b

Auto Application Interface for Word 1.0

Auto Application for WordPerfect 5.2 for Windows 1.0

Hardware:

486 50 MHz DX with 16 megabytes of RAM

The next two examples demonstrate some of the capabilities of the DCOM through the OOKeyMan implementation. The examples involve two of the most popular standard Microsoft Windows applications on the market today; Word™ for Windows® and WordPerfect™ for Windows®. The examples also use a standard object, Standard Microsoft Object Package, to embed the encrypted files in a standard container object. The interfaces used for these examples were the Auto Application Interface for Word for Windows 1.0 and the Auto Application for WordPerfect 5.2 for Windows 1.0.

The first example shows the ability for OOKeyMan to securely manage and track single or multiple embedded encrypted objects within other encrypted objects. This is done with a single application.

The second example shows the ability for OOKeyMan to securely manage and track single or multiple embedded encrypted objects within other encrypted objects. The embedded encrypted objects can even be part of encrypted objects from other applications. This example is performed in a cross-application manner between Word for Windows and WordPerfect for Windows.

Example 1

Document Level Multi-Level Multimedia Security (using Microsoft Word for Windows and WordPerfect 5.2 for Windows)

OOKeyMan Process

1. Lock object

A. User creates an object(s) in Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Encrypt;

D. User Selects Labels for object;

E. User Selects an algorithm for encryption;

F. User Selects Lock Object;

G. OOKeyMan Object Manager perform Setup and Internal Checks;

H. OOKeyMan Object Manager Calls Key Management System object;

I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;

J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;

K. Encrypted OOKeyMan Object Created;

i. Results in Encapsulation of

a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type

ii. Results in New Encrypted Object being created

iii. Results in Plain Text Object Being Delete if Requested

L. OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.

2. Unlock Object

A. User creates an encrypted object(s) using Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Decrypt;

D. User Selects Unlock object;

E. OOKeyMan Decrypt Object;

F. OOKeyMan Object Manager performs Setup and Internal Checks;

G. OOKeyMan Object Manager Calls Key Management System object;

M. OOKeyMan Object Manager Calls Algorithm object;

I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object;

J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.

3. Preview Object

A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Preview;

D. User selects Preview Object;

E. OOKeyMan Display Header Object;

F. OOKeyMan Object Manager performs Setup and Internal Checks;

G. OOKeyMan Object Manager Calls Key Management System object;

H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

Example 2

Cross-Application Multi-Level Multimedia Security at The Document Level

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

(Between Microsoft Word for Windows and WordPerfect 5.2 for Windows)
OOKeyMan Process:
1. Lock Object
A. User creates an object(s)in Word for Windows or WordPerfect for Windows;
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Encrypt;
D. User Selects Labels for object;
E. User Selects an algorithm for encryption;
F. User Selects Lock Object;
G. OOKeyMan Object Manager Performs Setup and Internal Checks;
H. OOKeyMan Object Manager Calls Key Management System object;
I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;
J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
K. Encrypted OOKeyMan Object Created;
i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type
ii. Results in New Encrypted Object being created
iii. Results in Plain Text Object Being Delete if Requested
OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.
2. Unlock Object
A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan Decrypt Object;
F. OOKeyMan Object Manager Performs Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager Calls Algorithm object;
I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.
3. Preview Object
A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan Display Header Object;
F. OOKeyMan Object Manager Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

Example 3

Standard Distributive Cryptographic Object Method Process (DCOMP)
OOKeyMan Process
1. Lock Object
A. User creates an object(s);
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Encrypt;
D. User Selects Labels for object;
E. User Selects an algorithm for encryption;
F. User Selects Lock Object;
G. OOKeyMan Object Manager Performs Setup and Internal Checks;
H. OOKeyMan Object Manager Calls Key Management System object;
I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;
J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;
K. Encrypted OOKeyMan Object Created;
i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm ObJect Type
ii. Results in New Encrypted Object being created
iii. Results in Plain Text Object Being Delete if Requested
L. OOKeyMan Object Manager Returns To Application Object.
2. Unlock Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan Decrypt Object;
F. OOKeyMan Object Manager Performs Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager Calls Algorithm object
I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
3. Preview Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan Display Header Object;
F. OOKeyMan Object Manager Performs Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System objects
H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.
The DCOM process can be applied to a vast number of areas in the real world. Whether it be the physical topology of the local area network/wide area network environment or

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

the dynamic structure of an organization, the DCOM process will change dynamically to reflect the current state of the object in question.

FIG. 3 and FIG. 4 show an encrypted object that contains a web of embedded encrypted objects nested within the other encrypted objects. The object shown in FIG. 3 contains ten embedded encrypted objects at five various levels. The encrypted object embedded in level 5 was embedded in an object in level four, level four objects in level 3 and so on. The plain text object containing the level 5 encrypted object can then be encrypted for further security. This single encrypted object encapsulates all of the data associated with the encrypted objects within it and therefore the entire encrypted object can then be sent out via any transport mechanism supporting binary file transfer.

FIG. 4 shows an encrypted object that contains a web of embedded encrypted objects nested within it. All of the attached embedded encrypted objects are fused together resulting in a single encapsulated encrypted object. The DCOM is powerful enough to dynamically adapt to accommodate N dimensional objects. In the very near future computing systems incorporating technology such as Virtual/Alternate Reality and Cyberspace, will need systems that can secure N dimensions.

The single encrypted objects shown in both FIGS. 3 and 4 can act as a secure package and can be sent out for distribution to an entire organization (e.g. E-mail). This single encrypted object can represent a branch(s), department(s), or even an entire company. Every employee would receive the single encrypted file, but they would only be able to unravel the portions that corresponded to them and acquire no knowledge of other existing embedded encrypted objects. For example, FIG. 5 displays a sample organization chart. When applied, the DCOM would control the knowledge/information flow of the organization and would allow for clear data separation, further compartmentalization through multiple algorithm use, and document-level security. With the improved communication paths, an organization would become more efficient. FIG. 6 demonstrates the use of the DCOM in conjunction with the dynamic structure of a sample organization. Since the DCOM is dynamic in nature, it can adapt to any organizational size or type (For example, see FIGS. 7 and 8).

Preferred and alternate embodiments of the present invention have now been described in detail. It is to be noted, however, that this description of these specific embodiments is merely illustrative of the principles underlying the inventive concept. It is therefore contemplated that various modifications of the disclosed embodiments will, without departing from the spirit and scope of the invention, be apparent to persons skilled in the art.

What is claimed is:

1. A method for providing multi-level multimedia security in a data network, comprising:

A) accessing an object-oriented key manager;

B) selecting a first object to encrypt;

C) selecting a first label for the first object;

D) selecting an encryption algorithm;

E) encrypting the first object according to the encryption algorithm;

F) labelling the encrypted first object wherein the labelling comprises creating a display header;

G) reading the first object label;

H) determining access authorization based on the first object label; and

I) allowing access to the first object only if access authorization is granted.

2. The method of claim 1, further comprising embedding the encrypted first object in a second object after labelling the encrypted first object.

3. The method of claim 2, further comprising:

A) selecting a second label for the second object;

B) selecting an encryption algorithm;

C) encrypting the second object; and

D) labelling the encrypted second object with a second object label.

4. The method of claim 3, further comprising

A) reading the second object label;

B) determining access authorization based on the second object label; and

C) allowing access to the second object only if access authorization is granted.

5. The method of claim 4, further comprising decrypting the second object.

6. The method of claim 4, wherein allowing access to the second object includes deleting the second object.

7. The method of claim 3, wherein the second label is a second plurality of labels.

8. The method of claim 2, wherein embedding the first encrypted object in a second object includes covering the first encrypted object.

9. The method of claim 8, further comprising:

A) selecting a second label for the second object;

B) selecting an encryption algorithm;

C) encrypting the second object;

D) labelling the second encrypted object with a second object label;

E) reading the second object label;

F) determining access authorization based on the second object label; and

G) uncovering the second object only if access authorization is granted.

10. The method of claim 1, wherein the first label is a first plurality of labels.

11. The method of claim 1, further comprising decrypting the first object.

12. The method of claim 1, wherein allowing access to the first object includes deleting the first object.

13. The system of claim 1, wherein allowing access to the first object includes access to the header file.

14. A system for providing multi-level multimedia security in a data network, comprising:

A) a data processor, the data processor comprising:

1)a system memory, comprising stored data;

2)an encryption algorithm module, comprising logic for converting unencrypted objects into encrypted objects, the encryption algorithm module being disposed to access data stored in the system memory;

3)an object labelling subsystem, comprising logic means for limiting object access, subject to label conditions, the object labelling subsystem being disposed to access data stored in the system memory and the object labelling subsystem being further disposed to accept inputs from the encryption algorithm module;

4)a decryption algorithm module, comprising logic for converting encrypted objects into unencrypted objects, the decryption algorithm module being disposed to access data stored in the system memory means; and

5)an object label identification subsystem, comprising logic for limiting object access, subject to label

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

conditions, the object label identification subsystem being disposed to access data stored in the system memory and the object label identification subsystem being further disposed to accept inputs from the decryption algorithm module;

B) the encryption algorithm module working in conjunction with the object labelling subsystem to create an encrypted object such that the object label identification subsystem limits access to an encrypted object.

15. The system of claim **14**, wherein the data processor further comprises embedded logic that embeds a first object within a second object.

16. The system of claim **15**, wherein the data processor further comprises covering logic that covers the first object with the second object.

17. The system of claim **16**, wherein the logic for limiting object access includes logic to uncover the first object.

18. A system for providing multi-level multimedia security in a data network, comprising:

A) means for accessing an object-oriented key manager;

B) means for selecting a first object to encrypt;

C) means for selecting a label for the first object;

D) means for selecting an encryption algorithm;

E) means for encrypting the first object;

F) means for embedding the first object within a second object;

G) means for labelling the encrypted first object;

H) means for reading the label;

I) means for determining access authorization based on the label; and

J) means for accessing the first object only if access authorization is granted;

K) the means for embedding the first object within a second object including means for covering the first object with a second object.

19. The system of claim **18**, further comprising:

A) means for selecting the second object to encrypt;

B) means for selecting a second label for the second object;

C) means for encrypting the second object;

D) means for labelling the encrypted second object;

E) means for reading the second label;

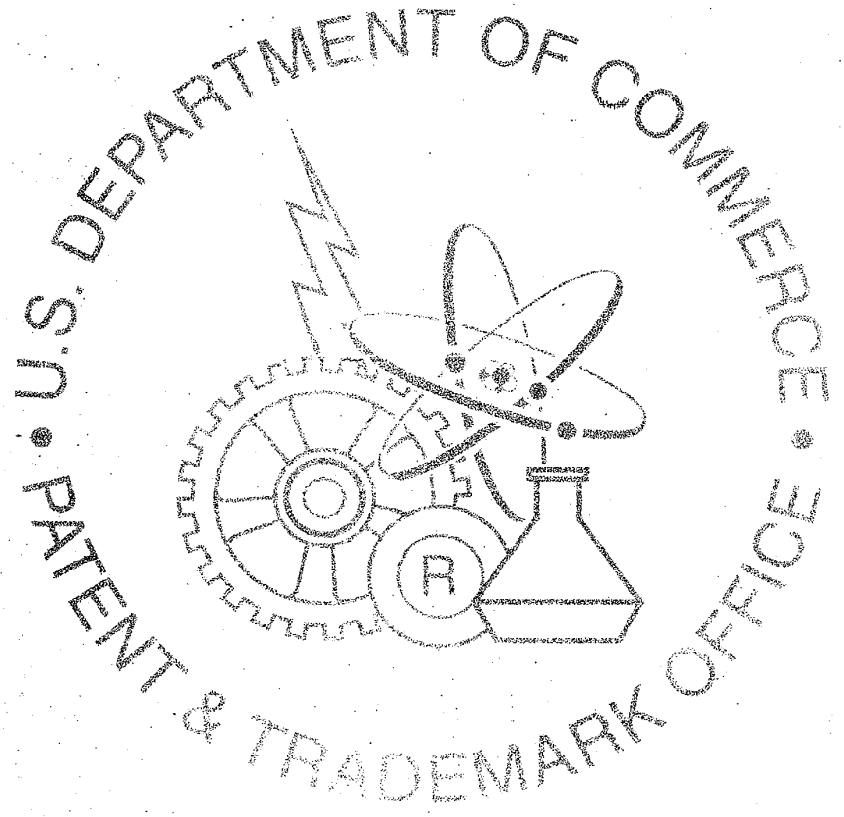
F) means for determining access authorization based on the second label; and

G) means for accessing the second object only if access authorization is granted.

20. The system of claim **19**, wherein the means for accessing the second object includes means for uncovering the first object.

* * * * *

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



TEC00140813

PTO-1683
(Rev. 7-96)

TEC00140814

U 7214101

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:
UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

November 25, 2009

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,898,781*
ISSUE DATE: *April 27, 1999*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

P. SWAIN
Certifying Officer

UNITED STATES PATENT AND TRADEMARK OFFICE · DEPARTMENT OF COMMERCE

Plaintiff's Trial Exhibit
4
TecSec. v. Adobe Case no. 1:10cv115

TEC00140815

US005898781A

United States Patent [19]

Shanton

[11] Patent Number: 5,898,781

[45] Date of Patent: * Apr. 27, 1999

[54] **DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD**

[75] Inventor: **M. Greg Shanton**, Fairfax, Va.

[73] Assignee: **TecSec Incorporated**, Vienna, Va.

[*] Notice: This patent is subject to a terminal disclaimer.

[21] Appl. No.: **08/927,043**

[22] Filed: **Sep. 10, 1997**

Related U.S. Application Data

[63] Continuation of application No. 08/304,867, Sep. 13, 1994, Pat. No. 5,717,755, which is a continuation of application No. 08/138,857, Oct. 18, 1993, Pat. No. 5,369,702.

[51] **Int. Cl.**[6] **H04L 9/32**; H04L 9/00

[52] **U.S. Cl.** **380/25**; 380/4; 380/9; 380/21; 380/23; 380/28; 380/49; 380/50; 340/825.31; 340/825.34

[58] **Field of Search** 380/4, 9, 21, 23, 380/25, 28, 49, 50, 59, 30, 3, 5, 10, 24, 43; 455/440; 340/825.31, 825.34

[56] **References Cited**

U.S. PATENT DOCUMENTS

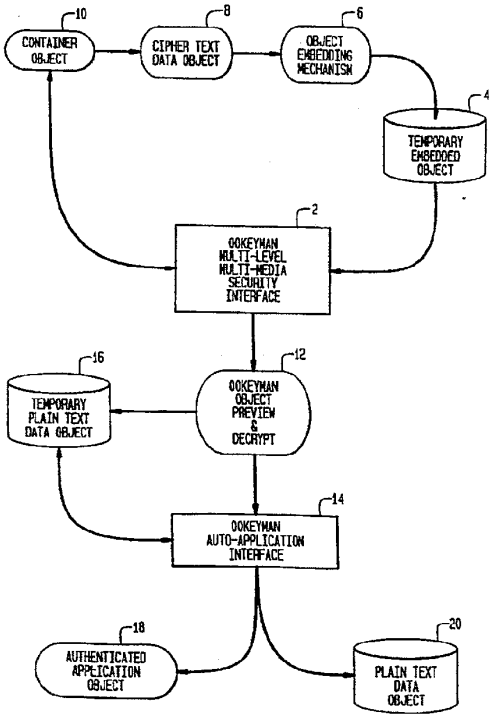
4,218,582	8/1980	Hellman et al.	380/30
4,405,829	9/1983	Rivest et al.	380/30
4,424,414	1/1984	Hellman et al.	380/30
4,713,753	12/1987	Boebert et al.	380/4 X
4,864,616	9/1989	Pond et al.	380/25
4,955,082	9/1990	Hattori et al.	455/440
4,962,533	10/1990	Krueger et al.	380/25
4,984,272	1/1991	McIlroy et al.	380/25
5,052,040	9/1991	Preston et al.	380/4
5,065,429	11/1991	Lang	380/25
5,191,611	3/1993	Lang	380/25
5,204,961	4/1993	Barlow	380/4 X
5,369,702	11/1994	Shanton	380/4
5,369,707	11/1994	Follendore, III	380/25
5,680,452	10/1997	Shanton	380/4
5,717,755	2/1998	Shanton	380/25

Primary Examiner—Bernarr E. Gregory
Attorney, Agent, or Firm—Rabin & Champagne, P.C.

[57] **ABSTRACT**

A system for increasing the security of a computer system, while giving an individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allow users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By effecting compartmentalization of every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

20 Claims, 8 Drawing Sheets



Copy provided by USPTO from the PIRS Image Database on 11/24/2009

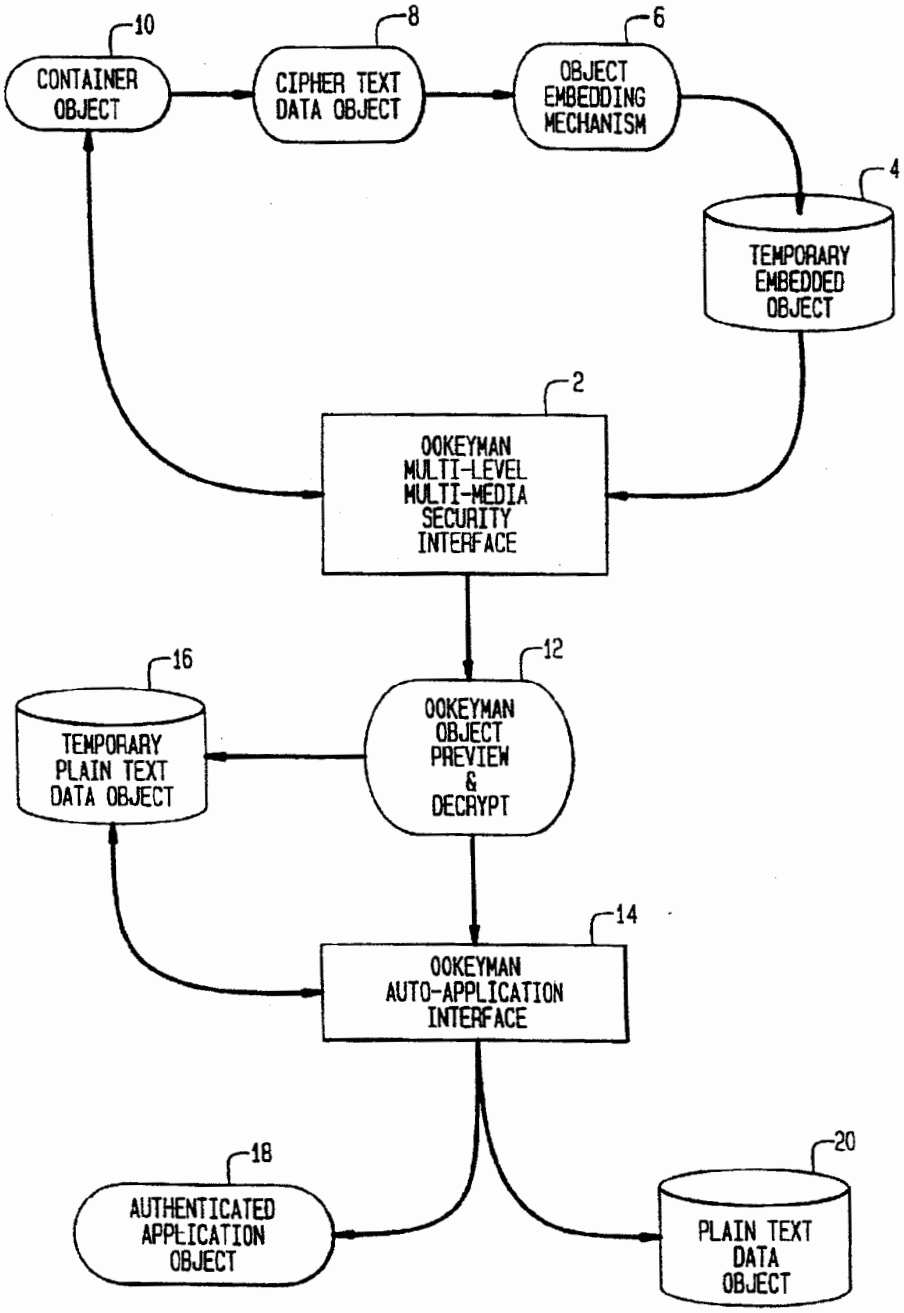


FIG. 1

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

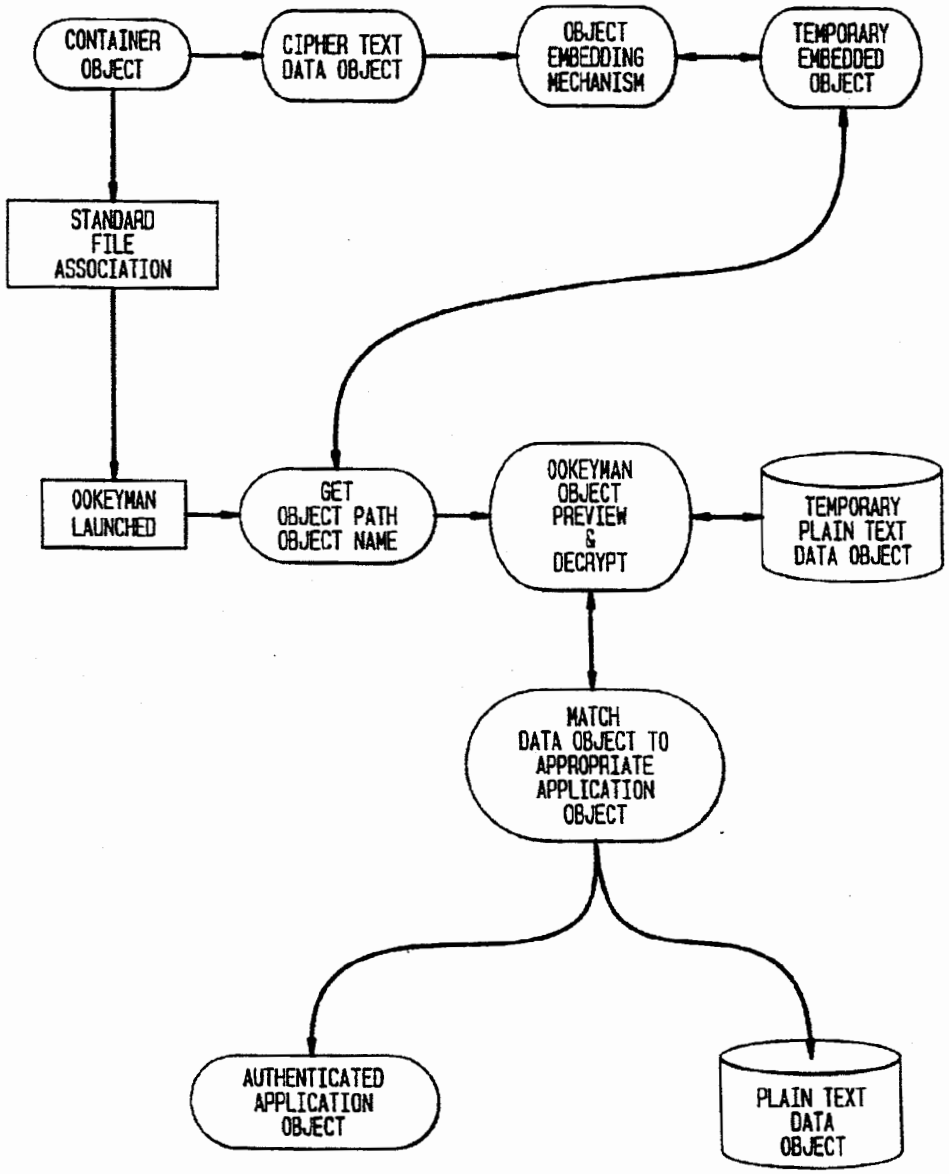


FIG. 2

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

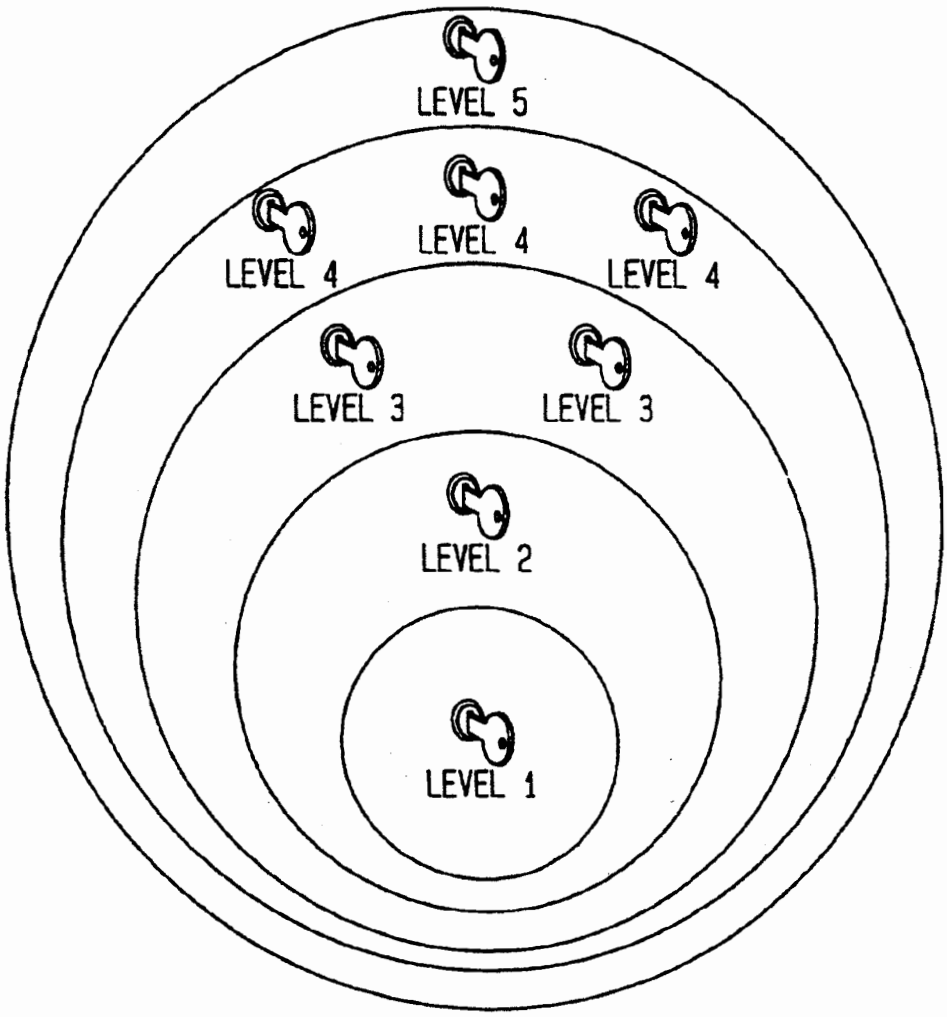


FIG. 3

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

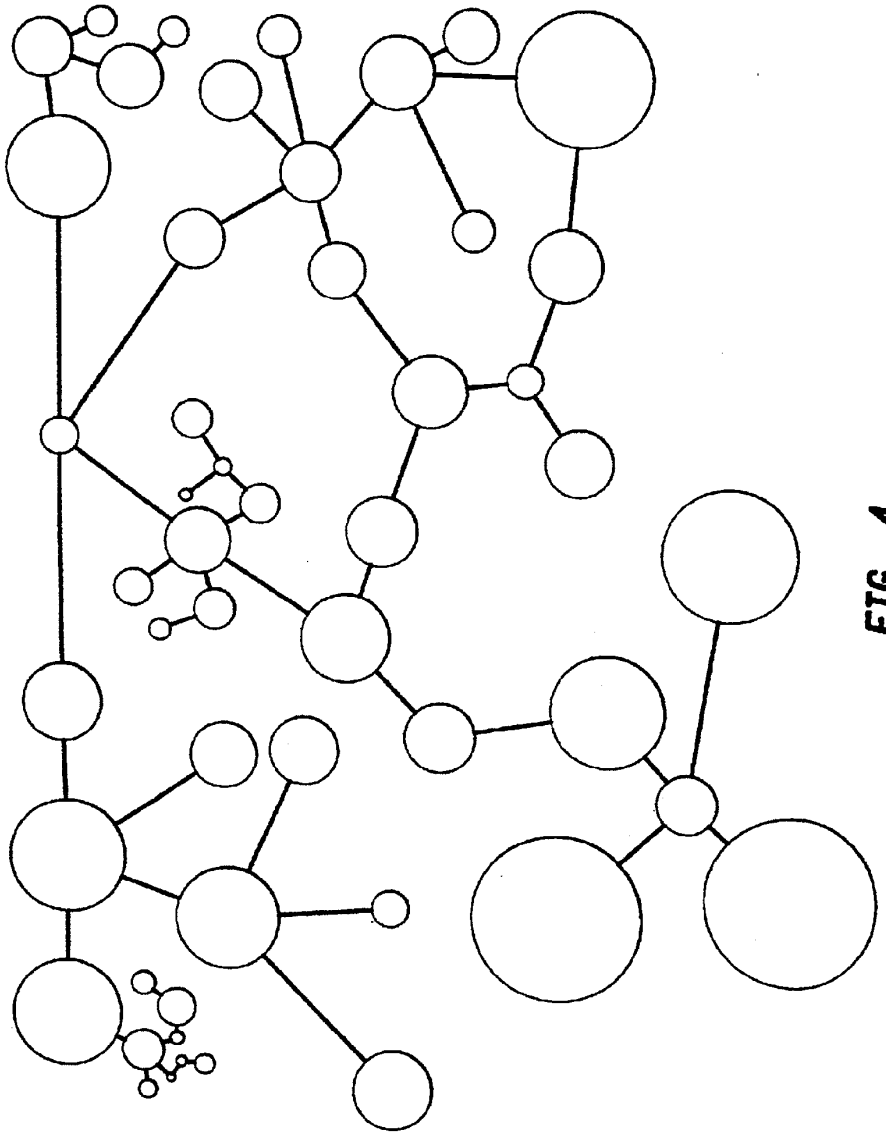


FIG. 4

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

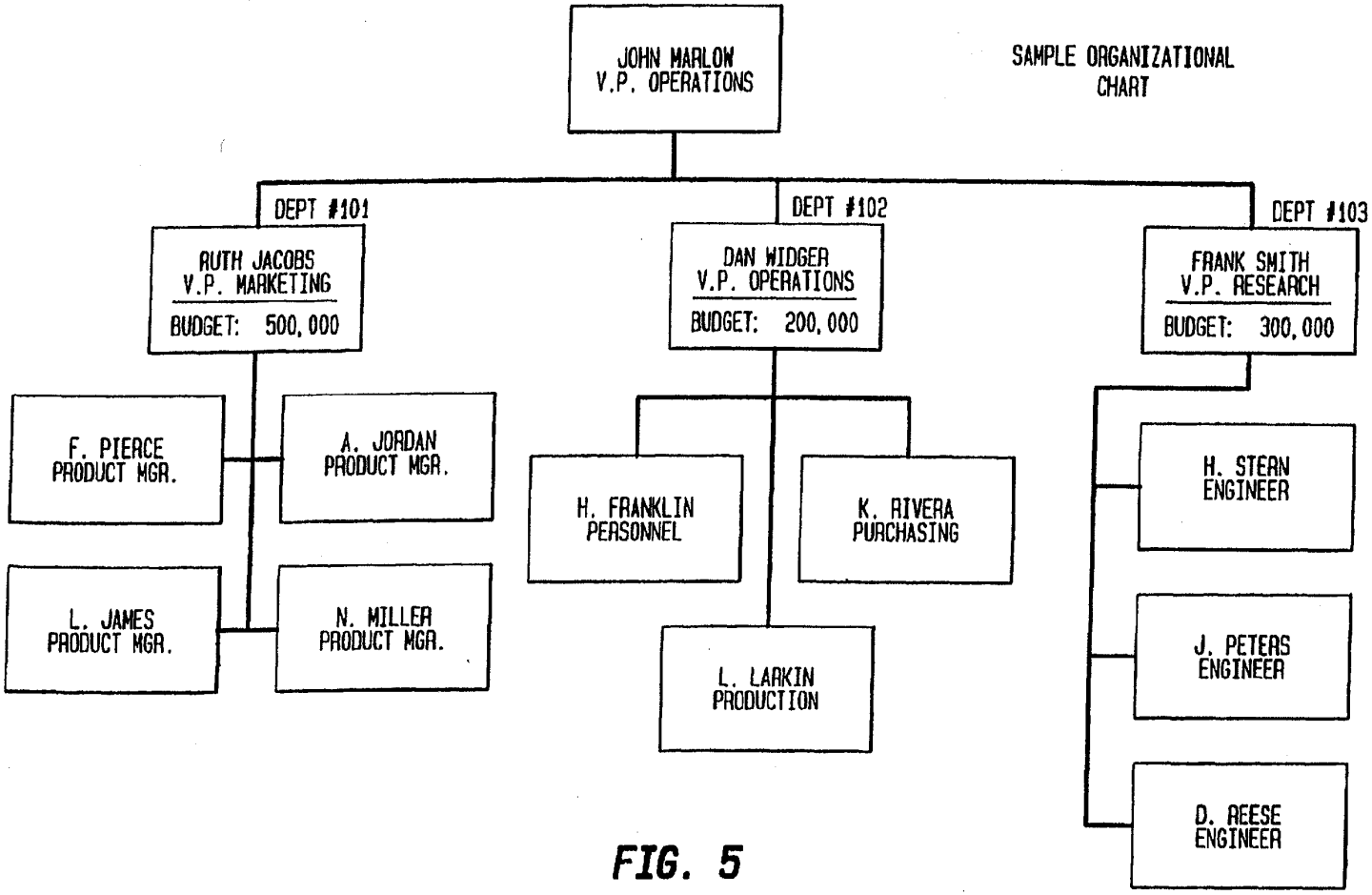


FIG. 5

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

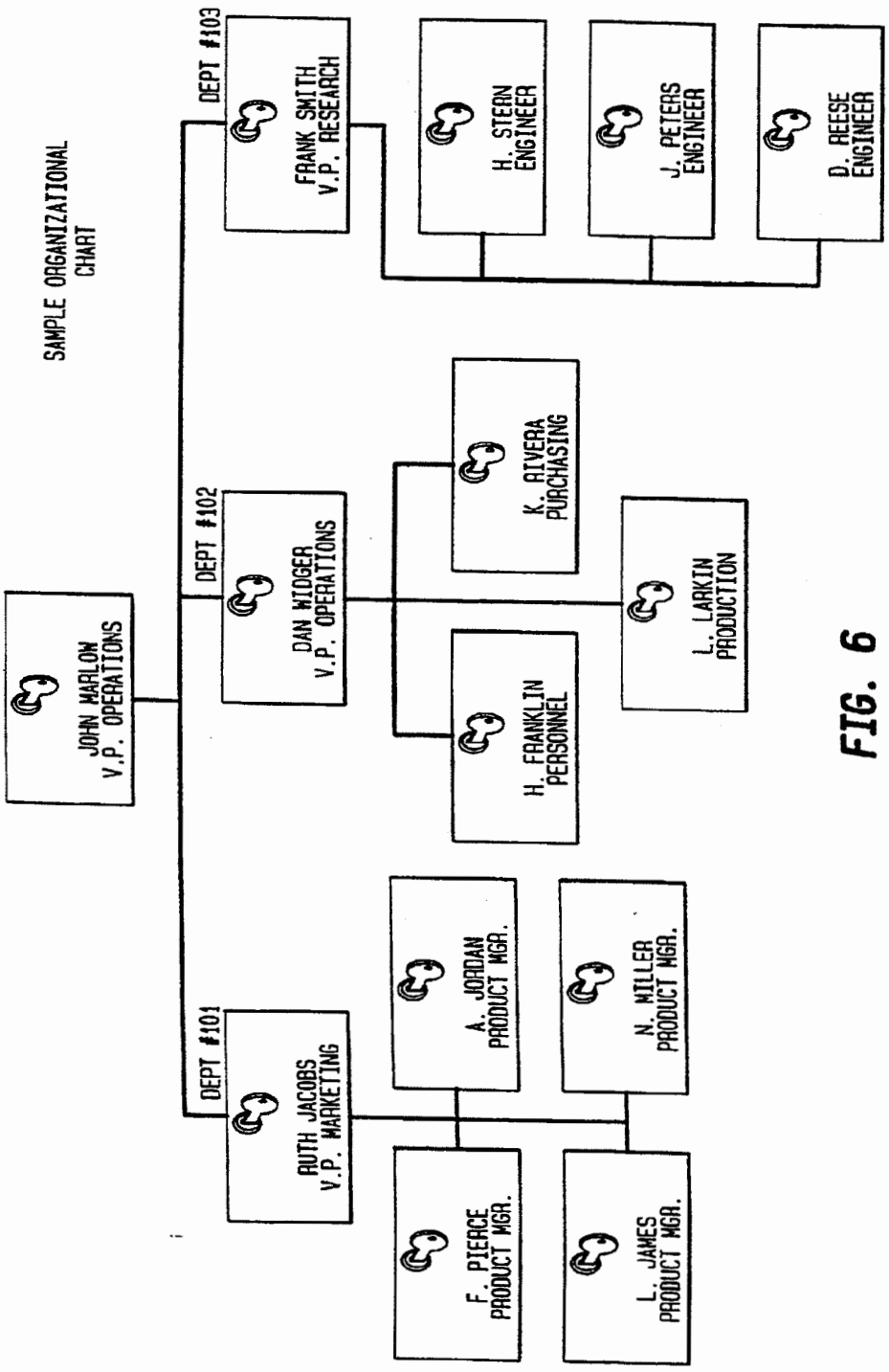


FIG. 6

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 178 Filed: 10/07/2019

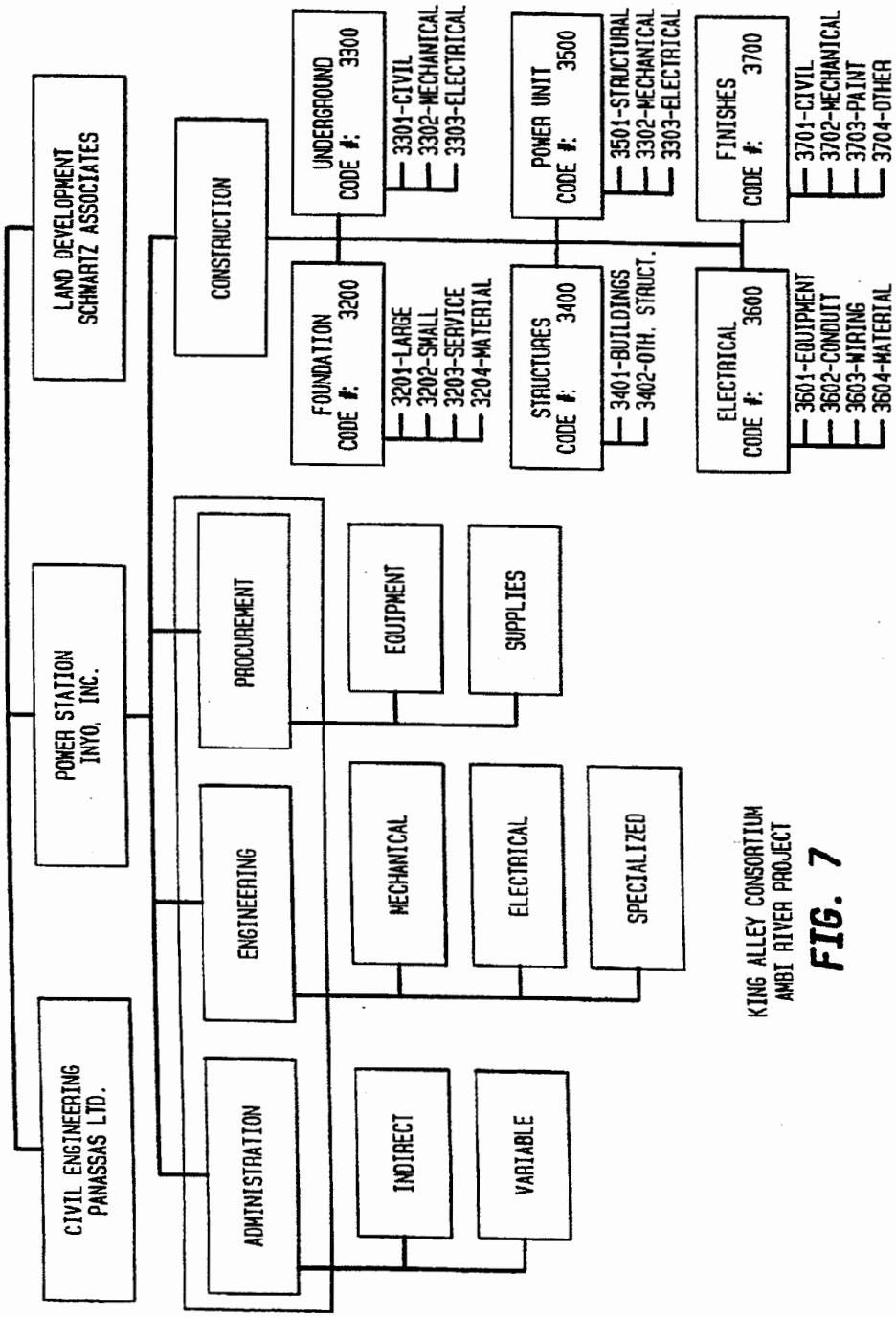
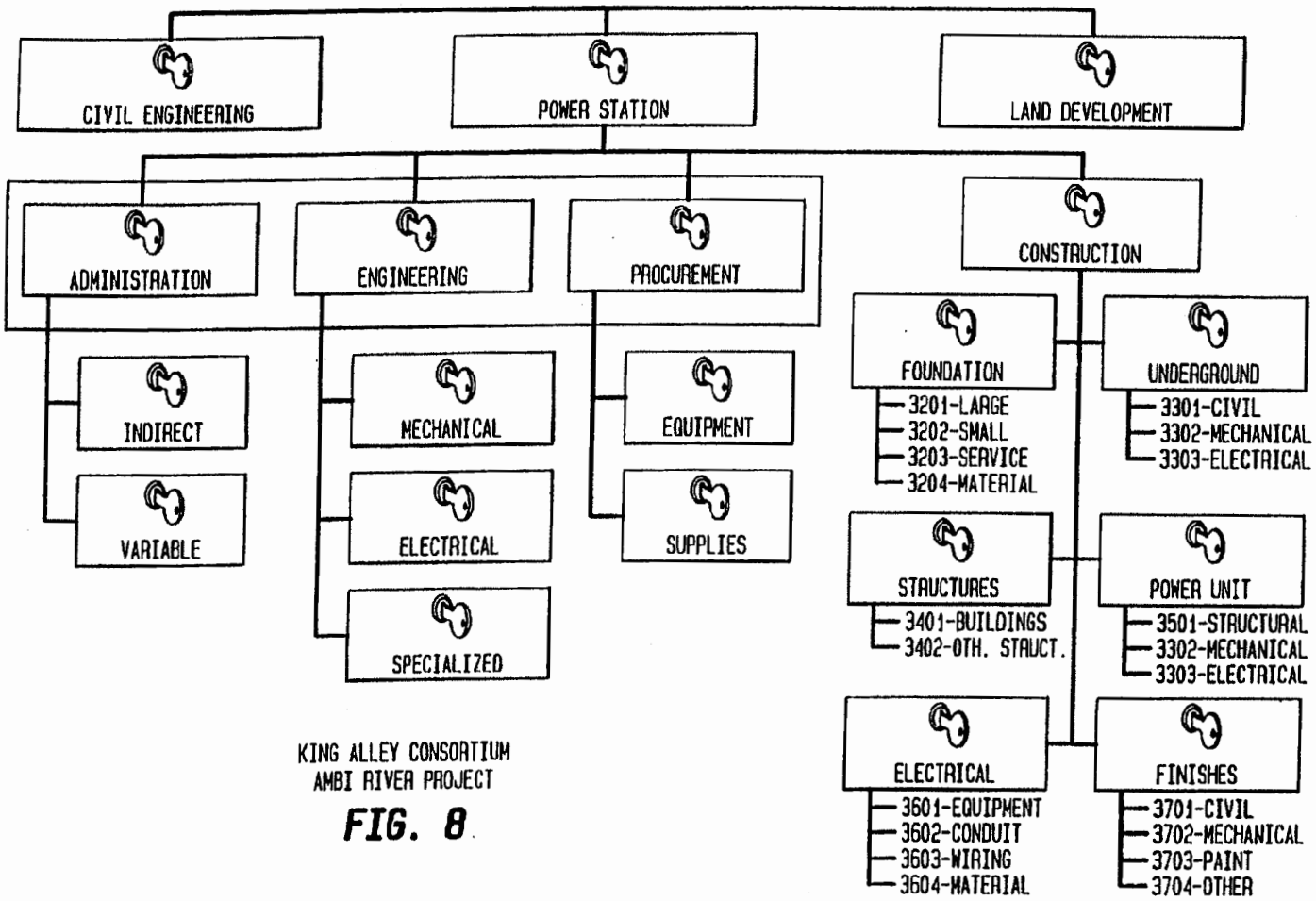


FIG. 7

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

Case: 19-2192 Document: 20 Page: 179 Filed: 10/07/2019



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

DISTRIBUTED CRYPTOGRAPHIC OBJECT METHOD

This is a Continuation of application Ser. No. 08/304,867, filed Sep. 13, 1994, now U.S. Pat. No. 5,717,755, which is a continuation of prior application Ser. No. 08/138,857, filed Oct. 18, 1993, now U.S. Pat. No. 5,369,702, issued Nov. 29, 1994.

FIELD OF THE INVENTION

The present invention relates generally to a system that can be used to restrict access to computer data. In particular, the system of the present invention restricts access in a flexible way, identifying objects for restriction and nesting restriction requirements through the use of embedded objects.

BACKGROUND OF THE INVENTION

While the specter of "spies" eagerly trying to obtain the defense information of various countries is very much still present in the defense and intelligence community, an equally massive threat now exists from technological or commercial "spies" who desire to obtain commercial and technical information from competing companies. These agents use sophisticated means similar to those used by the defense and intelligence community in order to obtain commercially valuable information that reveals the plans and commercial activities of competitors thereby allowing the aggressor company to obtain a competitive advantage in the marketplace. Theft of commercially valuable information is a very real and ever present threat.

To combat this type of commercial spying, various complex systems have evolved to protect company proprietary information. These systems involve physical controls over personnel as well as over the data flowing in and out of a company. For example, most computer systems used within companies require a password to be entered before the system can be accessed. It is frequently the case that confidential or company proprietary information must be passed electronically from one location to another in order to convey that information within the company in a timely fashion. Such electronic communication is easily susceptible to interception if not protected in some other form.

Cryptographic systems have evolved to fill the needs of companies and individuals wanting to protect the proprietary commercial information of a company from competitors and those who generally should not have that information.

Encryption of data is therefore a critical requirement in denying access to confidential information from those who are not so authorized. Cryptographic "keys" are an essential part of the information encryption process. The cryptographic key, or "key" for short, is a sequence of letters, numbers, or bytes of information which are manipulated by a cryptographic algorithm to transform data from plain (readable) text to a series of unintelligible text or signals known as encrypted or cipher text. The key is then used by the receiver of the cipher text to decrypt the message back to plain text. However, for two people to communicate successfully using keys, each must use the same key, assuming that the same encryption/decryption algorithm is used on both ends of the communication.

Simple encryption of data being communicated between two points only provides one level of security, however. Encryption limits data communication to those who have the key. Anyone who has the key is privy to any communication at any location. That is, if a group of people are working on a particular project, they will all presumably share a key for decrypting information relating to the project. Some of the project group may be working in one location, while the rest of the group may be located in a distant city. If one member of the group wants to send a communication to a particular member in the other city, the key will afford him no protection because everyone in the project shares the same key. Likewise, if someone wants to communicate a message to a subset of the group, for example, only to management personnel, this key would again provide her with no extra security. In another case, someone may want to send a message that is capable of being read only at a particular computer terminal, or of being printed only at a particular printer. In these and other cases, multilevel multimedia key access, or individual keys issued to each person, would provide a solution, albeit one that is quite unwieldy, inflexible, and difficult to manage by a security officer or key administrator.

A secure method of labelling files or messages that are sent from a sending user to a receiving user over a network can provide a level of protection in addition to cryptographic protection. A file "label" for purposes of this invention means a series of letters or numbers, which may or may not be encrypted, separate from but associated with the sending of a message, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated message. Using a secure labelling regimen, a network manager or user can be assured that only those messages meant for a certain person, group of persons, and/or location(s) are in fact received, decrypted, and ready by the intended receiver. Thus, a sending user can specify label conditions that limit access to the transmitted message. For example, many people within a company may have the key necessary to read a data file that a sender may transmit from his computer terminal to other terminals at another sits within his company. The sender may, however, wish to restrict reception to those persons present at a particular terminal. By employing a secure labelling technique in addition to encryption, the sender can be assured that people having the correct key to decrypt the message but working at different terminals will not receive or be allowed to access the communication. Access may be limited to particular people as well.

A system such as that described above is disclosed in U.S. patent application Ser. No. 08/009,741, the specification of which is incorporated by reference herein.

A system that can limit access on an object level would be more flexible and would offer still more protection. Access could be specified on an object-by-object basis, and objects could be embedded within other objects, providing an access hierarchy for users.

The ability to cryptographically secure objects ensures the authentication and data integrity of the particular object or objects in question. If a device were able to cryptographically control an object(s) or nested object(s), then that device would have total control over the entire object and all other objects within it. This type of control over the knowledge/information flow would allow for clear data separation, and at some levels could become a transparent method. A system that is able to do this would be able to achieve multi-level multimedia security.

SUMMARY OF THE INVENTION

It is therefore an objective of the present invention to provide a system to insure that properly specified kinds of information in a network system flows only to designated

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

locations and to further insure that such information is only read by those individuals who are designated to review that information.

It is a further objective of the present invention to provide a system that recognizes objects and permits or denies access on the object level.

It is an additional objective of the present invention to provide a system in which objects may be embedded within other objects, resulting in an access hierarchy for users of the system.

It is another objective of the present invention to provide a system in which access control is transparent to the user.

These and other objectives and advantages of the present invention will be apparent to those of ordinary skill in the art upon inspection of the detailed description, drawings, and appended claims.

The definition and concept of objects varies greatly depending on with whom you consult. Everything around you in your daily life is an object. Your car, your car keys, books, people, desks, etc. Objects are entities by themselves, but they may contain other objects, in either single or multiple configurations. Objects can change their make up dynamically by inheritance. Objects can inherit the attributes of other objects and the inheritance features can change dynamically "on the fly" during the operation of the objects.

In the context of the present invention, an object can come in a vast number of forms, shapes or sizes and can be either passive or active, dynamic or static. An object may stay dormant until it is acted upon, or it may be an active participant, dynamically auditing and verifying every transaction that occurs in a system. Examples of what an object can be include a bit of information, a byte of information, Sound Clips, Video Clips, Graphic Images, text, charts, tables, forms, controls, MDIForms, variables, executable files, video files, binary files, text files, data files, container files, graphic files, application file(s), Library files, a directory, a collection of directories, a hard disk, multiple hard disks, any hardware component, any software component, a complete computer system, a single network, multiple networks.

The present invention is able to increase the security of the system, while at the same time giving the individual user a large amount of flexibility and power. To give users the most power and flexibility, a standard object that has the capability to embed objects is used. To allow users even more flexibility, a standard object tracking mechanism is used that allows users to distribute multiple encrypted embedded objects to other individuals in a single encrypted object. By being able to compartment every object by label attributes and algorithm attributes, multi-level multimedia security is achieved.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a block diagram of the system of the present invention.

FIG. **2** shows a block diagram of the system of the present invention when an embedded encrypted object is activated.

FIG. **3** shows an object containing ten embedded encrypted objects at five various levels.

FIG. **4** shows an encrypted object that contains a web of embedded encrypted objects nested within it.

FIG. **5** shows a sample organizational chart.

FIG. **6** shows the present invention used in conjunction with the dynamic structure of a sample organizational chart.

FIG. **7** shows a larger, more complicated sample organizational chart.

FIG. **8** shows the present invention used in conjunction with the dynamic structure of the larger, more complicated sample organizational chart.

DETAILED DESCRIPTION

Definitions

OOKeyMan stands for the Object-Oriented Key Manager. OOKeyMan is a Windows stand alone application.

The Auto Application Interface is an intelligent front end and back end interface between a standard Microsoft Windows 3.1 application and OOKeyMan.

An encrypted Embedded object is an encrypted OOKeyMan object which can contain a single plain text object that has been encapsulated within the encrypted object, or it can contain an infinite web of encrypted objects matched with plain text objects or other cipher text objects.

A container object is an object that contains other objects. These objects can be either cipher text or plain text. This is the transport vehicle for a standard object mechanism that embeds objects. A non-container object is an object that does not contain other objects.

Multi-Level Multimedia Security is defined as the ability to have simultaneous control over the knowledge/information flow of numerous media formats while allowing for clear data separation. At some levels the multi-level multimedia security becomes transparent. Examples of a multi-media objects would include a file that contained two or more of the following: sound objects, video objects, graphic V, text objects, chart objects, table objects, and form objects.

Disclosure

The present invention, known as the Distributed Cryptographic Object Method ("DCOM"), is able to control which objects are visible to a specific user, which object attributes are inherited by other objects, which objects are available for use, and which level of system implementation could become transparent.

The main function of the DCOM is to securely manage and track encrypted objects. The DCOM can securely manage and track a single encrypted object, or it can securely manage and tack encrypted objects embedded within other encrypted objects. The capability to securely manage and track encrypted objects within other encrypted objects is only limited by storage space.

Referring to FIG. **1**, the DCOM system is described. The DCOM has a standard Multi-Level Security object interface **2** that interfaces with the plain text container object's encrypted embedded object(s) **4**. It does this through a standard application **6** that has the capability to embed an object in a container object, such as Microsoft's Object Package for Windows. After the Encrypted object(s) is/are embedded in a standard container object(s) **10** and the container object(s) **10** is/are encrypted, the original encrypted object(s) and the new encrypted container object (s) is/are ready for transport.

The new encrypted object(s) can be easily transported/routed over any network that supports binary travel without modification. The original encrypted objects can be deleted because all information from the original encrypted objects is encapsulated in the embedded encrypted object. All of the nested embedded encrypted objects will appear as a single

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

encrypted object until extracted with a standard object embedding/extracting mechanism through the DCOM process. To activate an embedded encrypted object, the user simply selects the encrypted object to initiate the DCOM process, launching the OOKeyMan application, as shown in FIG. **2**. The user/encrypted object authentication process is started and if the user/encrypted object is/are approved, the following encrypted object information can be returned and used by the user:

A. Plain Text Object Name
B. Plain Text Object Location
C. Plain Text Object Application
D. Plain Text Object Environment
E. Plain Text Object Date
F. Plain Text Object Time
G. Plain Text Object Digital Signature
H. Code word Object Tracking Label
I. Cluster Object Tracking Label
J. Device Object Tracking Label
K. Use Object Label
L. Algorithm Object Type

At this point, the authenticated user is given the option to decrypt the requested embedded encrypted object **12**. After decryption, a check is done to match the encrypted object's plain text object application to the correct Intelligent Auto Application Interface **14**. If the correct Auto Application Interface is not found, a notice is returned and the object is copied to a temporary location **16**, otherwise the Auto Application Interface process is started. During this process the encrypted object is matched to the appropriate authenticated application object **18** according to the returned encrypted object information. The correct authenticated application object **18** is then activated with the plain text object **20**. Due to the relative dynamic nature of objects, the DCOM is able to accomplish all tasks "on the fly".

The scope of the DCOM directly correlates to the level at which the DCOM was embedded into the system. The scope of the DCOM would cover the implemented embedded system level and all system levels above that, appearing transparent to all levels beneath the implemented embedded system level. For instance, if the DCOM were was embedded at the Open System Interconnection ("OSI") **7** Application layer, then the scope of the DCOM would cover objects on that level and above. In this scenario, the DCOM could run transparent to OSI levels 1 through 6. On the basis of current technology, this implementation would produce the most flexible DCOM. At this level and above, the DCOM is able to provide multi-level multi-media security while staying at the document level. This cross-application compatibility or document-level security is critical to the evolving component based document centered computer system desktop. The DCOM achieves cross-application multi-level multi-media security at the document level through its use of Object-Based Security.

The current implementation of the DCOM at the application layer is called the Object-Oriented Key Manager (OOKeyMan). Currently, OOKeyMan is a Microsoft Windows 3.1 stand alone application, but the DCOM can be applied to other environments. OOKeyMan provides Document-Level Security through its use of Object Based Security.

Some examples of where the DCOM can be applied to ensure the authentication and data integrity of objects include:

IBM's OS/2 2.X and above
IBM's System Object Method(soM)
Microsoft's Object Package
Microsoft's Object Linking and Embedding 1.0(OLE 1.0)
Microsoft's Object Linking and Embedding 2.0(OLE 2.0)
Microsoft's Windows NT 3.1
Microsoft's Cairo (Future Operating System)
Microsoft's Chicago (Future Operating System)
Taligent (joint venture future Operating System of Macintosh and IBM)
Macintosh's Compound Document Standard
Macintosh Operating System
Novell
Novell Netware Directory Services (NDS)
Unix Object-Oriented Systems
Virtual/Alternate Reality Systems
Future Object-Oriented Operating Systems

By applying the DCOM to the above examples, the security of a system can be moved to a more abstract object level. By securing objects with cryptography, a level of security is achieved much higher than that of common access control mechanisms such as password or pass phrase protection.

The steps for embedding an Encrypted Embedded OOKeyMan Object(s) are as follows:

1. User Creates a plain text object by using a standard application;
2. User Encrypts Object(s) with OOKeyMan;
3. User uses a standard Container Object;
4. Using Standard object to embed Encrypted Embedded OOKeyMan object(s) into Container Object;
5. Encrypt Container Object;
6. Repeat Steps 1 through 5 until all Objects are encrypted;
7. Multi-Level Multimedia Security achieved at the document level.

Examples of the Distributed Cryptographic Object Method

The following resources were used in the following examples:

Software:
MS-DOS 5.0
Microsoft Windows 3.1
Microsoft Word for Windows 2.0c
Standard Microsoft Object Package
WordPerfect 5.2 for Windows
OOKeyMan 1.Ob
Auto Application Interface for Word 1.0
Auto Application for WordPerfect 5.2 for Windows 1.0

Hardware:
486 50 MHz DX with 16 megabytes of RAM

The next two examples demonstrate some of the capabilities of the DCOM through the OOKeyMan implementation. The examples involve two of the most popular standard Microsoft Windows applications on the market today; Word™ for Windows® and WordPerfect™ for Windows®. The examples also use a standard object, Standard Microsoft Object Package, to embed the encrypted files in a standard container object. The interfaces used for these examples were the Auto Application Interface for Word for Windows 1.0 and the Auto Application for WordPerfect 5.2 for Windows 1.0.

The first example shows the ability for OOKeyMan to securely manage and track single or multiply embedded

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

encrypted objects within other encrypted objects. This is done with a single application.

The second example shows the ability for OOKeyMan to securely manage and track single or multiply embedded encrypted objects within other encrypted objects. The embedded encrypted objects can even be part of encrypted objects from other applications. This example is done cross-application between Word for Windows and WordPerfect for Windows.

EXAMPLE 1

Document Level Multi-Level Multimedia Security (with Microsoft Word for Windows and Wordperfect 5.2 for Windows)

OOKeyMan Process

1. Lock Object

A. User creates an object(s) in Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Encrypt;

D. User Selects Labels for objects;

E. User Selects an algorithm for encryption;

F. User Selects Lock Object;

G. OOKeyMan Object Manager Setup and Internal Checks;

H. OOKeyMan Object Manager Calls Key Management system object;

I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;

J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;

K. Encrypted OOKeyMan Object Created;

i. Results in Encapsulation of

a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type

ii. Results in New Encrypted Object being created

iii. Results in Plain Text Object Being Delete if Requested

L. OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.

2. Unlock Object

A. User creates an encrypted object(s) Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Decrypt;

D. User Selects Unlock object;

E. OOKeyMan Decrypt Object;

F. OOKeyMan Object Manager Setup and Internal Checks;

G. OOKeyMan Object Manager Calls Key Management System object;

H. OOKeyMan Object Manager Calls Algorithm object;

I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object;

J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.

3. Preview Object

A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Preview;

D. User selects Preview Object;

E. OOKeyMan Display Header Object;

F. OOKeyMan Object Manager Setup and Internal Checks;

G. OOKeyMan Object Manager Calls Key Management System object;

H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

EXAMPLE 2

Cross-Application Multi-Level Multimedia Security at the Document Level

(Between Microsoft Word for Windows and WordPerfect 5.2 for Windows)

OOKeyMan Process:

1. Lock Object

A. User creates an object(s) in Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Encrypt;

D. User Selects Labels for object;

E. User Selects an algorithm for encryption;

F. User Selects Lock Object;

G. OOKeyMan Object Manager Setup and Internal Checks;

H. OOKeyMan Object Manager Calls Key Management System object;

I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;

J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;

K. Encrypted OOKeyMan Object Created;

i. Results in Encapsulation of

a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Date
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code word object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type

ii. Results in New encrypted Object being created

iii. Results in Plain Text Object Being Delete if Requested

L. OOKeyMan Object Manager Returns To Word for Windows or WordPerfect for Windows.

2. Unlock Object

A. User creates an encrypted object(s) Word for Windows or WordPerfect for Windows;

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Decrypt;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

D. User Selects Unlock object;
E. OOKeyMan Decrypt Object;
F. OOKeyMan Object Manager Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager Calls Algorithm object;
I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
J. If the User/encrypted Object are authenticated the plain text object is activated along with Word for Windows or WordPerfect for Windows.
3. Preview Object
A. User creates an encrypted object(s) in Word for Windows or WordPerfect for Windows;
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan Display Header Object;
F. OOKeyMan Object Manager Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

EXAMPLE 3

Standard Distributive Cryptographic Object Method Process(DCOMP)

OOKeyMan Process:

1. Lock Object

A. User creates an object(s);

B. User Initiates OOKeyMan sequence;

C. User Selects object(s) to Encrypt;

D. User Selects Labels for object;

E. User Selects an algorithm for encryption;

F. User Selects Lock Object;

G. OOKeyMan Object Manager Setup and Internal Checks;

H. OOKeyMan Object Manager Calls Key Management System object;

I. OOKeyMan Object Manager Calls Cryptographic Algorithm object;

J. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the encrypted object;

K. Encrypted OOKeyMan Object Created;

i. Results in Encapsulation of
a. Plain Text Object
b. Plain Text Object Name
c. Plain Text Object Location
d. Plain Text Object Application
e. Plain Text Object Environment
f. Plain Text Object Data
g. Plain Text Object Time
h. Plain Text Object Digital Signature
i. Code Word Object Tracking Label
j. Cluster Object Tracking Label
k. Device Object Tracking Label
l. Use Object Label
m. Algorithm Object Type

ii. Results in New Encrypted Object being created

iii. Results in Plain Text Object Being Delete if Requested

L. OOKeyMan Object Manager Returns To Application Object.
2. Unlock Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Decrypt;
D. User Selects Unlock object;
E. OOKeyMan Decrypt Object;
F. OOKeyMan Object Manager Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager Calls Algorithm object
I. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the decrypted object.
3. Preview Object
A. User creates an encrypted object(s);
B. User Initiates OOKeyMan sequence;
C. User Selects object(s) to Preview;
D. User selects Preview Object;
E. OOKeyMan Display Header Object;
F. OOKeyMan Object Manager Setup and Internal Checks;
G. OOKeyMan Object Manager Calls Key Management System object;
H. OOKeyMan Object Manager waits for Selected Algorithm object to finish and create the Header object.

The DCOM process can be applied to a vast number of areas in the real world. Whether it be the physical topology of the local area network/wide area network environment or the dynamic structure of an organization, the DCOM process will change dynamically to reflect the current state of the object in question.

FIG. 3 and FIG. 4 show an encrypted object that contains a web of embedded encrypted objects nested within the other encrypted objects. The object shown in FIG. 3 contains ten embedded encrypted objects at five various levels. The encrypted object embedded in level 5 was embedded in an object in level four, level four objects in level 3 and so on. The plain text object containing the level 5 encrypted object can then be encrypted for further security. This single encrypted object encapsulates all of the data associated with the encrypted objects within it and therefore the entire encrypted object can then be sent out via any transport mechanism supporting binary file transfer.

FIG. 4 shows an encrypted object that contains a web of embedded encrypted objects nested within it. All of the attached embedded encrypted objects are fused together resulting in a single encapsulated encrypted object. The DCOM is powerful enough to dynamically adapt to accommodate N dimensional objects. In the very near future computing systems incorporating technology such as Virtual/Alternate Reality and Cyberspace, will need systems that can secure N dimensions.

The single encrypted objects shown in both FIGS. 3 and 4 can act as a secure package and can be sent out for distribution to an entire organization (e.g. E-mail). This single encrypted object can represent a branch(s), department(s), or even an entire company. Every employee would receive the single encrypted file, but they would only be able to unravel the portions that corresponded to them and acquire no knowledge of other existing embedded encrypted objects. For example, FIG. 5 displays a sample organization chart. When applied, the DCOM would control the knowledge/information flow of the organization and would allow for clear data separation, further compartmentalization through multiple algorithm use, and document-level secu-

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

rity. With the improved communication paths, an organization would become more efficient. FIG. **6** demonstrates the use of the DCOM in conjunction with the dynamic structure of a sample organization. Since the DCOM is dynamic in nature, it can adapt to any organization size or type (For example, see FIGS. **7** and **8**).

Preferred and alternate embodiments of the present invention have now been described in detail. It is to be noted, however, that this description of these specific embodiments is merely illustrative of the principles underlying the inventive concept. It is therefore contemplated that various modifications of the disclosed embodiments will, without departing from the spirit and scope of the invention, be apparent to persons skilled in the art.

What is claimed is:

1. A method for providing multi-level multimedia security in a data network, comprising:

A) accessing an object-oriented key manager;

B) selecting a first object to encrypt;

C) selecting a first label for the first object;

D) selecting an encryption algorithm;

E) encrypting the first object according to the encryption algorithm;

F) labelling the encrypted first object;

G) reading the first object label;

H) determining access authorization based on the first object label; and

I) allowing access to the first object only if access authorization is granted.

2. The method of claim **1**, further comprising embedding the encrypted first object in a second object after labelling the encrypted first object.

3. The method of claim **2**, further comprising:

A) selecting a second label for the second object;

B) selecting an encryption algorithm;

C) encrypting the second object; and

D) labelling the encrypted second object with a second object label.

4. The method of claim **3**, further comprising:

A) reading the second object label;

B) determining access authorization based on the second object label; and

C) allowing access to the second object only if access authorization is granted.

5. The method of claim **4**, wherein allowing access to the second object includes decrypting the second object.

6. The method of claim **4**, wherein allowing access to the second object includes deleting the second object.

7. The method of claim **3**, wherein the second label is a second plurality of labels.

8. The method of claim **2**, wherein embedding the first encrypted object in a second object includes covering the first encrypted object.

9. The method of claim **8**, further comprising:

A) selecting a second label for the second object;

B) selecting an encryption algorithm;

C) encrypting the second object;

D) labelling the second encrypted object with a second object label;

E) reading the second object label;

F) determining access authorization based on the second object label; and

G) uncovering the second object only if access authorization is granted.

10. The method of claim **1**, wherein the first label is a first plurality of labels.

11. The method of claim **1**, wherein allowing access to the first object includes decrypting the first object.

12. The method of claim **1**, wherein allowing access to the first object includes deleting the first object.

13. The method of claim **1**, wherein labelling the encrypted first object includes attaching a file header to the first object, and allowing access to the first object includes allowing access to the file header.

14. A system for providing multi-level multimedia security in a data network, comprising:

1) a system memory for storing data;

2) an encryption algorithm module, comprising logic for converting unencrypted objects into encrypted objects, the encryption algorithm module being disposed to access data stored in the system memory;

3) an object labelling subsystem, comprising logic for applying label conditions to an object, the object labelling subsystem being disposed to access data stored in the system memory and the object labelling subsystem being further disposed to accept inputs from the encryption algorithm module;

4) a decryption algorithm module, comprising logic for converting encrypted objects into unencrypted objects, the decryption algorithm module being disposed to access data stored in the system memory means; and

5) an object label identification subsystem, comprising logic for limiting object access, according to the label conditions, the object label identification subsystem being disposed to access data stored in the system memory and the object label identification subsystem being further disposed to accept inputs from the decryption algorithm module;

6) wherein the encryption algorithm module and the object labelling subsystem together create an encrypted object such that the object label identification subsystem limits access to the encrypted object.

15. The system of claim **14**, wherein the data processor further comprises embedding logic for embedding a first object within a second object.

16. The system of claim **15**, wherein the data processor further comprises covering logic for covering the first object with the second object.

17. The system of claim **16**, wherein the logic for limiting object access includes logic to uncover the first object.

18. A system for providing multi-level multimedia security in a data network, comprising:

A) means for accessing an object-oriented key manager;

B) means for selecting a first object to encrypt;

C) means for selecting a label for the first object;

D) means for selecting an encryption algorithm;

E) means for encrypting the first object;

F) means for embedding the first object within a second object;

G) means for labelling the encrypted first object;

H) means for reading the label;

I) means for determining access authorization based on the label; and

J) means for accessing the first object only if access authorization is granted;

Copy provided by USPTO from the PIRS Image Database on 11/24/2009

K) the means for embedding the first object within a second object including means for covering the first object with a second object.

19. The system of claim **18**, further comprising:

A) means for selecting the second object to encrypt;

B) means for selecting a second label for the second object;

C) means for encrypting the second object;

D) means for labelling the encrypted second object;

E) means for reading the second label;

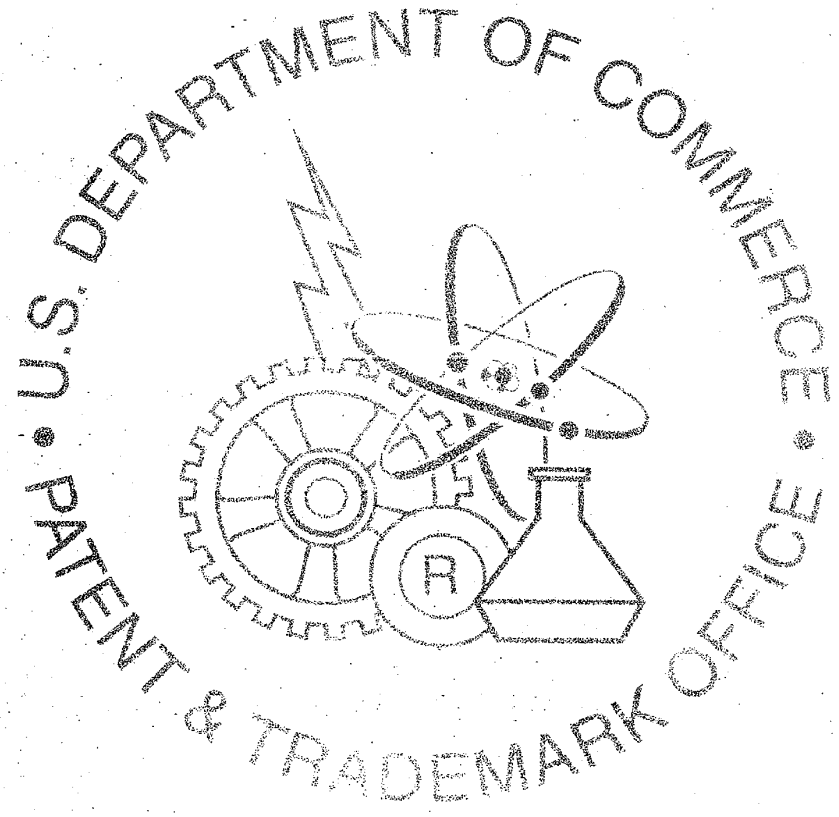
F) means for determining access authorization based on the second label; and

G) means for accessing the second object only if access authorization is granted.

20. The system of claim **19**, wherein the means for accessing the second object includes means for uncovering the first object.

* * * * *

Copy provided by USPTO from the PIRS Image Database on 11/24/2009



TEC00140832

PTO-1683
(Rev. 7-96)

TEC00140833

CERTIFICATE OF SERVICE

I hereby certify that on the 7th day of October, 2019, the foregoing CORRECTED OPENING BRIEF FOR PLAINTIFF-APPELLANT was electronically filed with the Clerk of the Court using the CM/ECF system which will issue an electronic notification to counsel of record.

/s/ Michael A. Oakes
Michael A. Oakes

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE REQUIREMENTS

The undersigned, an attorney, hereby certifies that:

1. This brief complies with the type-volume limitation of Federal Rule of Appellate Procedure 32(a)(7)(B) and Federal Circuit Rule 32(b). This brief contains 13,855 words, excluding the parts of the brief exempted by Federal Rule of Appellate Procedure 32(a)(7)(B)(iii) and Federal Circuit Rule 32(b).

2. This brief complies with the typeface requirements of Federal Rule of Appellate Procedure 32(a)(5) and the type style requirements of Federal Rule of Appellate Procedure 32(a)(6).

This brief has been prepared in a proportionally spaced typeface using Microsoft Word in fourteen-point Times New Roman type style.

/s/ Michael A. Oakes
Michael A. Oakes
Counsel for Appellant TecSec, Inc.