2015-1686, -1749

UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

TECSEC, INC.

Plaintiff-Appellant,

v.

ADOBE SYSTEMS, 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-Appellees.

Appeal from the United States District Court for the Eastern District of Virginia in 1:10-CV-115, Judge Leonie M. Brinkema

---

CORRECTED OPENING BRIEF FOR PLAINTIFF-APPELLANT

Michael A. Oakes
Michael A. O'Shea
HUNTON & WILLIAMS LLP
2200 Pennsylvania Avenue, N.W.
Washington, D.C. 20037
Telephone: (202) 955-1500
Facsimile: (202) 778-2201

Gregory N. Stillman
HUNTON & WILLIAMS LLP
500 East Main Street
Suite 1000
Norfolk, Virginia 23510
Telephone: (757) 640-5300
Facsimile: (757) 625-7720

August 3, 2015

U.S. Patent No. 5,369,702

Claim 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.

Claim 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 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.

UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

Nos. 2015-1686, -1749

Certificate of Interest

Counsel for Plaintiff-Appellant TecSec, Inc. certify the following:

1. The full name of every party or amicus represented by me is:

TecSec, Incorporated

2. The name of the real party in interest (if the party named in the caption is not the real party in interest) represented by me is:

TecSec, Incorporated

3. All parent corporations and any publicly held companies that own 10 percent or more of the stock of the party or amicus curiae represented by me are:

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 are:

Attorneys who appeared in the trial court: HUNTON & WILLIAMS LLP, Thomas J. Cawley, Brian M. Buroker, Michael A. Oakes, Michael A. O'Shea, Ryan Phair, Stephen Sayers, Torsten Kracht, Jeffrey S. Leaning, Gregory N. Stillman, Richard L. Wyatt, Jr., Rachael R. Yocum, DINOVO PRICE ELLWANGER & HARDY LLP, Andrew G. DiNovo, Jay Ellwanger, Raymond W. Mort III, Adam G. Price

Attorneys expected to appear in this Court: HUNTON & WILLIAMS LLP, Michael A. Oakes, Michael A. O'Shea, Gregory N. Stillman

/s/ Michael A. Oakes

TABLE OF CONTENTS

TABLE OF ABBREVIATIONS viii

INTRODUCTION 1

STATEMENT OF RELATED CASES 5

JURISDICTIONAL STATEMENT 6

STATEMENT OF THE ISSUES 7

STATEMENT OF THE CASE 8

I. FACTUAL BACKGROUND 8

A. TecSec And The DCOM Patents 8

B. The Accused Adobe Acrobat Products 9

II. PRIOR PROCEEDINGS 13

SUMMARY OF ARGUMENT 16

ARGUMENT 18

I. Standard Of Review 18

II. The District Court Erred In Granting Summary Judgment Of No Infringement Sua Sponte Without Proper Notice 18

III. The District Court Erred in Construing "Selecting a Label" 21

A. "Selecting A Label" Was Not In Dispute And Should Not Have Been Construed 21

B. The District Court Improperly Imported A "Pre-Existing Label" Limitation Into The Construction For The Term "Selecting A Label For An Object" 23

IV. The District Court Erred In Finding That The Limitation "Selecting A Label" Is Not Met By The Accused Acrobat Products 27

A. The Asserted System Claims Do Not Contain The "Selecting A Label" Limitation 27

B. The Accused Products Meet The "Selecting a Label" Limitation Of The Method Claims 32

V. The District Court Erred In Construing The Term "Label".....36

A. "Label" Is Correctly Construed As "An Identifier Associated With An Object".....36

B. The District Court's Construction Improperly Reads In A Limitation From A Description Of The Prior Art In The Background Section Of The Specification 38

VI. The District Court Erred In Holding That The Encryption Dictionary With Password Security Is Not A Label 41

VII. The District Court Erred In Construing "Object-Oriented Key Manager" 44

VIII. The District Court Erred In Construing "Display Header".....47

IX. The District Court Erroneously Decided Sua Sponte That The "Labelling" Limitation Is Not Met.....49

X. This Case Should Be Reassigned On Remand.....50

CONCLUSION.....55

TABLE OF AUTHORITIES

Cases

Amdocs (Israel) Ltd. v. Openet Telecom, Inc.,
56 F. Supp. 813 (E.D. Va. 2014) 52

Anderson v. Liberty Lobby, *Inc.*,
477 U.S. 242 (1986) 18

Charles E. Hill & Assocs., Inc. v. Compuserve Interactive Servs., Inc.,
33 Fed. Appx. 527 (Fed. Cir. 2002) 31

Chicago Bd. Options Exch., Inc. v. Int'l Sec. Exch., LLC,
677 F.3d 1361 (Fed. Cir. 2012) 31

Coward v. Jabe,
532 Fed. Appx. 328 (4th Cir. 2013) 53

Cybor Corp. v. FAS Techs.,
138 F.3d 1448 (Fed. Cir. 1998) (en banc) 18

DSW, Inc. v. Shoe Pavilion, Inc.,
537 F.3d 1342 (Fed. Cir. 2008) 31

Emblaze Ltd. v. Apple, Inc.,
No. 5:11-cv-01079-PSG, 2014 U.S. Dist. LEXIS 144197 (N.D. Cal. Oct. 9, 2014) 39

Fin Control Sys. Pty, Ltd. v. OAM, Inc.,
265 F.3d 1311 (Fed. Cir. 2001) 20

Greenlaw v. United States,
554 U.S. 237 (2008) 19

Henderson ex rel. Henderson v. Shinseki,
562 U.S. 428 (2011) 19

In re Am. Acad. Of Science Tech Ctr.,
367 F.3d 1359 (Fed. Cir. 2004) 42

Laitram Corp. v. NEC Corp.,
62 F.3d 1388 (Fed. Cir. 1995) 25

McNeil v. Wisconsin,
501 U.S. 171 (1991) 19

Mikkelsen Graphic Eng'g, Inc. v. Zund Am., Inc.,
541 Fed. Appx. 964 (Fed. Cir. 2013) 19, 20, 49

NL Indus., Inc. v. GHR Energy Corp.,
940 F.2d 957 (5th Cir. 1991).....20

Phillips v. AWH Corp.,
415 F.3d 1303 (Fed. Cir. 2005) 26, 37, 40, 42, 46, 48

Renishaw PLC v. Marposs Societa' per Azioni,
158 F.3d 1243 (Fed. Cir. 1998).....44

Research Corp. Techs., Inc. v. Microsoft Corp.,
536 F.3d 1247 (Fed. Cir. 2008).....50

SciMed Life Sys., Inc. v. Advanced Cardiovascular Sys., Inc.,
242 F.3d 1337 (Fed. Cir. 2001).....40

Source Vagabond Sys. Ltd. v. Hydrapak, Ltd.,
753 F.3d 1291 (Fed. Cir. 2014).....24

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

TecSec, Inc. v. Int'l Bus. Mach. Corp.,
731 F.3d 1336 (Fed. Cir. 2013)..... 5, 51

TriMed, Inc. v. Stryker Corp.,
608 F.3d 1333 (Fed. Cir. 2010)..... 50, 53

U.S. Surgical Corp. v. Ethicon, Inc.,
103 F.3d 1554 (Fed. Cir. 1997).....23

United States v. Guglielmi,
929 F.2d 1001 (4th Cir. 1991)..... 51, 53

Vitronics Corp. v. Conceptronic, Inc.,
90 F.3d 1576 (Fed. Cir. 1996).....26

Statutes

28 U.S.C. § 1295(a)(1) 6

28 U.S.C. § 1331 6

28 U.S.C. § 1338(a) 6

28 U.S.C. § 2106 50

Other Authorities

10A Charles Alan Wright & Arthur R. Miller, Federal Practice & Procedure § 2720 (3d ed. 1998) 20

Rules

Fed. R. Civ. P. 56(f) 31

TABLE OF ABBREVIATIONS

Parties

TecSec	TecSec, Incorporated
Adobe	Adobe Systems, Incorporated

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	'702 Patent, '452 Patent, '755 Patent and '781 Patent collectively
Asserted System Claims	'702 Patent claims 8 and 9 and '781 Patent claims 14 and 15
Asserted Method Claims	'702 Patent claims 1 and 4, '452 Patent claim 1, '755 Patent claim 1, and '781 Patent claims 1 and 3.
Asserted DCOM Claims	Asserted DCOM System Claims and Asserted DCOM Method Claims

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 Leonie Brinkema presiding

INTRODUCTION

In a prior appeal in this same action, this Court reversed the district court on seventeen of eighteen claim construction rulings and its *sua sponte* ruling that every asserted means-plus-function claim element lacked supporting structure, and vacated the district court's judgment of no infringement in favor of the defendants, including Adobe. On remand, the district court enacted an extraordinary case management plan whereby Adobe was permitted to move for summary judgment of no infringement before discovery even commenced. Although Adobe represented to the district court that its motion would be targeted, Adobe instead took a shotgun approach and advanced at least nine disparate theories of non-infringement, many of which were discussed in no more than a paragraph. Under the court's plan, TecSec was afforded a mere twenty-one days for written discovery to obtain evidence to oppose Adobe's motion.[1] Immediately after receiving Adobe's motion for summary judgment, TecSec served a focused set of Requests for Production and Interrogatories, but Adobe refused to respond to the Requests at all, arguing that they were untimely because the discovery period was shorter than the 30-day period for responding to discovery requests afforded by the Federal Rules of Civil Procedure. When TecSec moved to compel, the district

[1] The period was later extended to thirty-five days. TecSec was also permitted two weeks to take three party depositions, including one deposition pursuant to Federal Rule of Civil Procedure 30(b)(6).

court struck eighteen of TecSec's twenty-eight Requests for Production and all but two of TecSec's Interrogatories, finding that TecSec's Requests sought "information not necessary to respond to defendant Adobe's Motion for Summary Judgment." A139-A140 (Dkt. 724). Despite the extremely abbreviated discovery period and narrow scope of permitted discovery, TecSec was able to obtain and provide sufficient evidence to rebut every issue raised in Adobe's motion, including an unrebutted expert declaration and express admissions from Adobe's 30(b)(6) witness on infringement.

While the case management process leading up to the summary judgment hearing was unorthodox, this appeal turns on the procedural and substantive errors made by the district court beginning at the summary judgment hearing itself. At the hearing, rather than addressing the issues raised by Adobe's motion, the district court focused almost exclusively on a new non-infringement defense that Adobe admittedly never raised. Specifically, without prior notice, the district court suggested another claim limitation ("selecting a label") that it believed was potentially not present in the Accused Product. While the district court stated that it may order the parties to brief the issue further, it never actually did so. A3932, 2:14-19 ("[T]he issues have been extensively briefed, but I actually have a question, and I don't think that either side has really addressed or briefed this issue, so I want to raise it with you orally here in court, and most likely I'm going to send

you back to the offices to brief it in detail."). The district court first addressed Adobe's counsel, who acknowledged that Adobe did not make the argument raised by the court, believing that the nine issues that Adobe did brief presented better non-infringement arguments, and declined the district court's invitation at the hearing to brief this new issue. A3934, A3950. The district court then asked for a "quick response" from TecSec, and TecSec's counsel explained that the Accused Products clearly met this limitation and that Adobe likely had not moved on this newly-raised ground because it wholly lacked merit. When the district court asked TecSec if it wished to brief the issue, TecSec declined because Adobe had not taken the district court's invitation to move on the issue, because TecSec had not been permitted to take discovery on the issue, and because the court did not indicate that it was considering granting summary judgment absent a motion from Adobe. The district court concluded the hearing and did not order briefing from the parties on "selecting a label." A3950.

Two months after the hearing, the district court entered an order granting summary judgment of no infringement for Adobe based primarily on the "selecting a label" argument that the district court raised for the first time at the hearing. The district court noted that "the arguments that Adobe made were not in the Court's view in most cases overwhelming or persuasive" and that "[t]he Court primarily found non-infringement on an issue that was not actually extensively briefed by

either side." A4365-A4366, 5:15-6:1. The district court's *sua sponte* grant of summary judgment was not only procedurally defective and premature, but demonstrably wrong in substance as well. Despite no discovery having been permitted on the issue, the evidence in the record and concessions from Adobe's counsel demonstrate that the "selecting a label" limitation is clearly met by the accused Acrobat product. The district court's ruling to the contrary, and the district court's other summary judgment rulings, violate the Federal Rules of Civil Procedure, misapplies settled principles of patent law, and must be reversed.

STATEMENT OF RELATED CASES

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

1) *TecSec, Inc. v. Int'l Bus. Mach. Corp.*, 466 Fed. Appx. 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).

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 Memorandum Opinion (Dkt. No. 772) (A1-A42) and Judgment (Dkt. No. 774) (A44) that Adobe does not infringe the asserted patents under the district court's claim constructions. 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 granting summary judgment of no infringement *sua sponte* on an issue that the court raised for the first time at the summary judgment hearing, on which Adobe declined to move, and on which TecSec was precluded from taking discovery.

2. Whether the district court erred in construing *sua sponte* the term "selecting a label for an object" by reading in a limitation that the label must be "pre-existing," when that limitation appears nowhere in the claims, specification, or prosecution history, and then finding no infringement based on that construction.

3. Whether the district court erred in holding *sua sponte* that the system claims necessarily include the method claim limitation of "selecting a label" and therefore were not infringed, even though that limitation does not appear in the system claims.

4. Whether the district court erred in construing "label" by reading in limitations from the specification and in finding no infringement based on that construction.

5. Whether the district court erred in construing other terms that were not essential to the district court's judgment, namely "object-oriented key manager" and "display header," by reading in limitations from the specification.

STATEMENT OF THE CASE

I. FACTUAL BACKGROUND

A. 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-five 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 multi-level 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 three additional DCOM Patents – the '452, '755 and '781 Patents – contain additional claims directed at security systems and methods that employ a combination of both access control and encryption to protect digital data.

The DCOM Patents disclose systems and methods for protecting sensitive information so that any digital object, for example, a particular section, sentence, word, or even a single character, could be labeled and then protected by a combination of access control and encryption. When an object is labeled, 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). To obtain access to an encrypted object, a user must enter the appropriate identifier, such as a password or security certificate, which the system uses to verify if access to the object should be allowed. If the user has permission, access to the object is obtained; if the user does not have rights to access the object, the DCOM system precludes access. Once access has been granted, the user must then have the appropriate key to enable decryption to actually see the object. The DCOM Patents provide for multi-level security by applying two or more layers of access control and encryption to objects.

B. The Accused Adobe Acrobat Products

The Accused Products relevant to the present appeal are the well-known Adobe Acrobat Standard and Pro products, Versions 5 and higher, which can be used to create, protect and read PDF files. PDF files or documents consist of multiple objects and can be viewed and manipulated using software, such as Adobe Acrobat, installed on hardware. A2482, Fig. 1 (depicting PDF components);

A2483, ¶ 4, A2485. ¶ 6; A3506-A3507, 22:19-23:24. Certain objects of a PDF document, specifically strings and streams, can be encrypted. *See* A2491, ¶ 21; A3510, 26:1-21. Using Acrobat, a user can choose to encrypt the entire PDF document, the document but not the metadata, or only the document's attachments. A3542-A3543, 70:2-71:15; A3772, ¶ 40. Acrobat also applies access controls, such as passwords and digital certificates, to restrict access to these encrypted objects. A3772, ¶ 40; A3777, ¶ 45.

Acrobat provides three different options by which PDF documents can be secured and encrypted: 1) password security; 2) certificate security; and 3) LiveCycle Rights Management security. *See* A3747; A3756; A3537, 63:13-19. 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. A3748; A3538, 66:2-6; A3544-A3547, 73:20-76:25; A3552, 81:7-11.

For example, using Acrobat, a user can first select a particular PDF document and then select one of the document security options listed above, such as "Encrypt with Password." A3581-A3582, 10:20-11:4. By selecting this option, the user causes Acrobat to select and access a security handler. A3515, 34:1-9; A3518-A3519, 37:13-38:20; A3584-A3585, 13:20-14:12; A3772, ¶ 40. Selecting this option also selects which type of encryption dictionary will be attached to the subsequently encrypted object. A3514-A3515, 33:5-34:4; A3532-A3533, 51:17-

52:4; A3585-A3586, 14:20-15:1; A3587-A3588, 17:7-18:5; A3660; A3664; A3791, ¶ 69; A3792, ¶ 72; A3772, ¶ 40. The contents of the encryption dictionary are determined by the security mechanism, algorithm and permissions selected by the user. A3522, 41:2-21; A3527-A3529, 46:18-48-5; A3541-A3543, 69:6-71:15; A3582-A3584, 11:10-13:10; A3772, ¶ 40. Specifically, for password security, the user must select: a compatibility level, which causes Acrobat to select a particular encryption algorithm; which objects are desired for encryption; a password for access control to the document; and finally, various permission levels for the document (e.g., open-only, restrict printing, etc.). *Id.*

Upon saving the results of the selections described above, Acrobat encrypts the requested object(s) using the selected encryption algorithm. A3585, 14:3-12. Acrobat's security handler labels the encrypted object(s) by adding an encryption dictionary to the trailer of the PDF document. A3585, 14:3-12; A3790-A3791, ¶¶ 67-68. Among other things, the encryption dictionary contains the information necessary to access and decrypt the encrypted object, including information regarding the encryption algorithm and strings for password validation. A3656; A3660; A3587, 17:9-22. The encryption dictionary is not encrypted so that the PDF document can be recovered. A3656; A3510, 26:17-19.

To access the PDF document protected with password security as described above, the document is opened and Acrobat first attempts to open/access the

document using a null user password. *See* A2493-A2494, ¶ 27; A3592, 22:2-9. If that fails (and it will if the document is protected with a password), the user is prompted to supply a password. A3592-A3593, 22:10-23:3. Upon entry of the password, the security handler attempts to verify that password against the information stored in the encryption dictionary. A3587, 17:9-15; A3593, 23:4-14; A3595, 25:4-18; A3790, ¶ 67. Upon successful verification of the password, the security handler begins the decryption process by generating a key based on information contained in the encryption dictionary and elsewhere in Acrobat. *See* A3593, 23:15-17; A3594, 24:22-24; A3597, 27:17-23; A3599, 29:9-14; A3790, ¶ 67. The document can be decrypted using this key.

Acrobat provides functionality for the embedding of PDF documents or other objects into other PDF documents that serve as container objects —knows as PDF portfolios and PDF envelopes. A2487-A2490, ¶¶ 15-17; A3754-A3755; A3512-A3513, 30:15-31:2; A3546, 75:7-11; A3552, 81:4-24. Any number of encrypted PDF documents can be embedded within other PDF documents. A3546, 75:21-23. Embedded files can be individually encrypted prior to embedding and have access control in the form of passwords or certificates applied to them. A2490-A2491, ¶ 20; A3546-A3547, 75:21-76:25. Likewise, the container file itself can be encrypted and have access control applied to it. *Id.* Even when a secured container file is decrypted and accessed, the individual embedded PDF

files or objects remain protected until accessed. The passwords between the attached encrypted PDF document and the encrypted container PDF document can be different, and the encryption dictionaries ***must*** be different. A3628-A3630, 70:18-72:20. Thus, Acrobat can create a web of encrypted and password protected PDF documents within other encrypted and password protected PDF documents. *See* A36-A37.

II. 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").[2] TecSec's claims against IBM then proceeded to dispositive motion practice while its claims against the Defendants were stayed. Following the district court's grant of summary judgment of non-infringement in IBM's favor and construction of certain claim terms. *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.

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

Upon remand, 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 claim constructions.[3] 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 allowed TecSec to proceed against Adobe only, and over TecSec's objection 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.

Two months later, the district court issued an opinion granting summary judgment of no infringement to Adobe. The district court either rejected or

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

deferred ruling on nearly every ground raised by Adobe in its motion, in several instances finding that TecSec had demonstrated that the Accused Product met the limitations in the claims or that at least a genuine issue of material fact existed. However, the district court granted summary judgment of no infringement based on its own claim construction and non-infringement position raised *sua sponte* 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).

SUMMARY OF ARGUMENT

I. The Federal Rules of Civil Procedure guarantee a litigant sufficient notice and discovery to assure that the relevant issues have been fully and fairly resolved. In this case, the district court adopted an unconventional summary judgment briefing schedule and discovery plan that allowed Adobe to file a pre-discovery summary judgment motion with only a limited and expedited post-motion discovery period. The district court then raised a new claim construction issue and non-infringement argument *sua sponte* at the summary judgment hearing that was not briefed by either party and on which TecSec was precluded from taking discovery, and granted summary judgment to Adobe on that ground. The district court's failure to provide TecSec with adequate notice and an opportunity to respond constitutes reversible error and requires that the district court's judgment be vacated.

II. The district court's summary judgment decision was also improper because the record, especially when viewed in the light most favorable to TecSec, clearly shows that genuine issues of material fact exist regarding TecSec's patent infringement claims. With respect to the Asserted System Claims, the "selecting a label" limitation that the district court found lacking is not even present in the claims. Nevertheless, the district court ignored the actual claim language and held, without analysis or reference to the record, that the limitation is inherent in the

claims and that TecSec waived any argument to the contrary. This conclusion is baseless. With respect to the Asserted Method Claims, the evidence shows that a user selects a label when securing a PDF document with Acrobat by selecting whether to utilize password-based encryption or certificate-based encryption.

III. The district court erred in construing "label" and in granting summary judgment of no infringement based on its construction. The district court's construction not only imports a limitation from the specification into the claims, but the portion of the specification from which the limitation came referred to the prior art and not the claimed invention. Applying the proper construction, there is no dispute that the accused Acrobat products include a "label," but even under the district court's flawed construction, the evidence in the record demonstrates that a genuine issue of material fact exists as to this limitation.

IV. The district court erred in construing the terms "object-oriented key manager" and "display header" by reading in limitations from the specification. While these constructions did not form the basis of any finding of no infringement, the district court's erroneous constructions could negatively impact TecSec's case against the remaining defendant, the claims against which have been stayed for more than five years, and should be reversed.

ARGUMENT

I. Standard Of Review

A district court's claim construction presents issues of law that are reviewed on appeal *de novo*. *Cybor Corp. v. FAS Techs.*, 138 F.3d 1448, 1456 (Fed. Cir. 1998) (en banc). This Court also reviews a district court's grant of summary judgment *de novo*. *Hologic, Inc. v. SenoRx, Inc.*, 639 F.3d 1329, 1334 (Fed. Cir. 2011). Applying the proper claim construction, this Court must reverse if there was a genuine issue of material fact, affording all reasonable inferences and construing the evidence in the light most favorable to TecSec. *Anderson v. Liberty Lobby*, *Inc.*, 477 U.S. 242, 255 (1986).

II. The District Court Erred In Granting Summary Judgment Of No Infringement *Sua Sponte* Without Proper Notice

The district court committed serious procedural error when it granted summary judgment against TecSec on an issue not raised by Adobe and without providing proper notice. The Supreme Court has repeatedly warned district courts that their focus should be on addressing the issues raised by the parties and not on advancing their own legal theories and arguments:

> In our adversary system, in both civil and criminal cases, in the first instance and on appeal, we follow the principle of party presentation. That is, we rely on the parties to frame the issues for decision and assign to courts the role of neutral arbiter of matters the parties present.

Greenlaw v. United States, 554 U.S. 237, 243 (2008); *accord Henderson ex rel. Henderson v. Shinseki*, 562 U.S. 428, 434 (2011) ("Under [our adversary] system, courts are generally limited to addressing the claims and arguments advanced by the parties."); *McNeil v. Wisconsin*, 501 U.S. 171, 181 n.2 (1991) ("What makes a system adversarial rather than inquisitorial is . . . the presence of a judge who does not (as an inquisitor does) conduct the factual and legal investigation himself, but instead decides on the basis of facts and arguments pro and con adduced by the parties.").

Pursuant to Rule 56(f) of the Federal Rules of Civil Procedure, district courts may only grant summary judgment on a ground not raised by a party under limited circumstances, and only after giving notice and a reasonable time to respond. And while the Rules permit courts to utilize this procedure, grants of summary judgment on a ground raised by the court are "generally disfavored, because they risk depriving a losing party of adequate notice and opportunity to oppose summary judgment." *Mikkelsen Graphic Eng'g, Inc. v. Zund Am., Inc.*, 541 Fed. Appx. 964, 972 (Fed. Cir. 2013) (vacating summary judgment of invalidity when proper notice was not provided). "If the court believes a non-moving party that has not filed a cross-motion for summary judgment on a particular issue is nonetheless entitled to judgment on that issue, 'great care must be exercised to assure that the original movant has had an adequate opportunity to show that there is a genuine

issue and that the opponent is not entitled to judgment as a matter of law.'" *Id.* (quoting 10A Charles Alan Wright & Arthur R. Miller, Federal Practice & Procedure § 2720 (3d ed. 1998)). If a court fails to provide the parties with adequate notice or an opportunity for the losing party to present evidence in opposition to the motion, granting summary judgment *sua sponte* is inappropriate. *Mikkelsen Graphic*, 541 Fed. Appx. at 972-73; *Fin Control Sys. Pty, Ltd. v. OAM, Inc.*, 265 F.3d 1311, 1321 (Fed. Cir. 2001) (summary judgment procedurally improper where the issue was not the subject of any motion and the record did not indicate that the issue had been fully litigated). Any reasonable doubt about whether a plaintiff received notice that its entire case was at risk because the court intended to grant summary judgment in favor of the defendant must be resolved in favor of the plaintiff. *See NL Indus., Inc. v. GHR Energy Corp.*, 940 F.2d 957, 965 (5th Cir. 1991).

Here, Adobe admitted that it did not argue the "selecting a label" limitation in its summary judgment motion. A3934-A3935, 4:11-5:6. In addition, Adobe declined the district court's invitation to brief the issue and the district court never indicated that it was planning to grant summary judgment absent a motion from Adobe. Nevertheless, the district court acknowledged that it "primarily found non-infringement on an issue that was not actually extensively briefed by either side" and that "the issue that [the court] ultimately rested our opinion on was not one that

Adobe had certainly highlighted." A4365, 5:17-18. Because the court limited discovery to issues raised by Adobe in its motion, it is beyond dispute that TecSec was prohibited from taking discovery on this issue during the abbreviated discovery period. As a result, proper notice was not provided to TecSec, the safeguards of Rule 56(f) were not followed, and summary judgment must be reversed.[4]

III. The District Court Erred in Construing "Selecting a Label"

A. "Selecting A Label" Was Not In Dispute And Should Not Have Been Construed

District Court's Construction	Proper Construction
Choosing a pre-existing label.	No construction necessary; plain and ordinary meaning.

As discussed above, during oral argument the district court inquired about the claim term "selecting a label for an object" and asked why Adobe had not moved for summary judgment based on that limitation. A3932-A3933, 2:12-3:1. Importantly, this term appears only in the Asserted Method Claims, and does not

[4] For the same reasons, the district court's finding that TecSec failed to show infringement under the doctrine of equivalents is flawed. A40. TecSec did not provided a detailed doctrine of equivalents analysis with respect to the "selecting a label" term because Adobe did not raise that as a non-infringement argument in its motion and TecSec was precluded from taking discovery on the issue or addressing it in its opposition. Indeed, TecSec had no notice prior to the hearing that "selecting a label" was in dispute at all.

appear in any of the Asserted System Claims.[5] The district court conceded that neither party proposed a construction for the term, and the district court itself failed to propose a construction for the term at the hearing. A19.

The district court's holding that the term was in dispute appears to be based on the flawed premise that TecSec proposed during oral argument that "selecting a label" and "creating a label" were synonymous. A22. This is not true. While the parties were asked at the hearing to discuss briefly whether this newly raised limitation was present in the Accused Product, the district court did not suggest that the term needed to be construed, nor did the court ask the parties for argument as to the proper construction for the term. However, during the hearing it became clear that no construction for the term was necessary because the parties essentially agreed on the meaning of the term. Specifically, Adobe argued that "***[s]electing a label*** is an action that is taken by the user of the system and ***is the action of saying I want to apply a particular kind of label*** or a particular label."). A3933, 3:12-14. Counsel for TecSec offered a nearly identical understanding of the term:

> [S]electing a label is selecting what that label will have on it. I mean, I have to tell the computer do I want the password security-type label that is going to invoke the password security handler, do I want a

[5] As discussed infra at section IV.A, the district court mistakenly concludes that each of the system claims must incorporate this step from the method claims inherently, but this conclusion is indisputably incorrect, as the step of "selecting a label" appears nowhere in the system claims. As a result, the grant of summary judgment of no infringement of the system claims based on a limitation that does not exist in the system claims is clearly erroneous.

> certificate-based label that is going to invoke the certificate security. Those are totally different labels. So I'm selecting which label I want the document to apply. If I want password security, I choose one type of label. If I want certificate security, I choose a separate type of label.

A3938, 8:5-14.

Accordingly, the meaning of the term was not in dispute, and no construction should have been provided. *U.S. Surgical Corp. v. Ethicon, Inc.*, 103 F.3d 1554, 1568 (Fed. Cir. 1997) (claim construction is for "resolution of disputed meanings.").

B. The District Court Improperly Imported A "Pre-Existing Label" Limitation Into The Construction For The Term "Selecting A Label For An Object"

The district court's opinion fails to acknowledge the parties' agreement on the meaning of the term, and instead adopts a construction with a limitation not advocated for by either party. Specifically, the district court unnecessarily substituted the word "choosing" for the equally clear word "selecting" and read in a "pre-existing" limitation, without citing any support from the specification that would demonstrate unambiguously that the plain claim language should be so limited. There is no just reason for this limitation. Even Adobe admitted that a fully created label need not be "pre-existing" — a user can select a label by instructing the system "***to apply a particular type of label***," even if the final label is not created until a later step in the process. A3933, 3:12-14. With no dispute

between the parties as to the unambiguous meaning of the term, no further construction was needed or warranted.

The district court's importation of a limitation from thin air distorts the plain meaning of the words of the claim and is not supported by the intrinsic evidence. *See Source Vagabond Sys. Ltd. v. Hydrapak, Ltd.*, 753 F.3d 1291, 1299-1300 (Fed. Cir. 2014) ("[A]n analysis that adds words to the claim language without support in the intrinsic evidence . . . does not follow standard canons of claim construction." (quotation marks and bracketing omitted)). There is nothing inherent in the word selecting that requires selection from a pre-existing, static group of options.

The district court acknowledges that the language of Claim 1 of the '702 Patent alone does not mandate that the label be "pre-existing," but justifies its construction through a flawed claim differentiation argument. A9, A20, A22. The district court notes that because dependent Claim 2 adds the limitation "creating an ***object*** in an application . . .," the limitation in independent Claim 1 of "selecting an ***object***" cannot include "creating an ***object***." (emphasis added). The district court then analogizes that "selecting a ***label***" in Claim 1 cannot similarly encompass "creating a ***label***," even though these are distinct and separate limitations and even though no dependent claim adds a limitation of "creating a label." The district court cites no case law for this novel theory of claim differentiation – likely

because the district court's understanding of the doctrine of claim differentiation is directly inapposite to this Court's precedent. An independent claim is necessarily broader than a dependent claim, and as a result the independent claim must be interpreted broadly enough so that it encompasses all of the dependent claims. *See Laitram Corp. v. NEC Corp.*, 62 F.3d 1388, 1392 (Fed. Cir. 1995) (holding that the printing by "strobing" limitation expressly claimed in the dependent claim necessarily must be covered by the more general language in the independent claim). Stated another way, Claim 1 must be construed in such a way such that it encompasses everything set forth in its dependent claims. If dependent Claim 2 includes the step of "creating an object," then independent Claim 1 must be construed broadly enough to cover, but not require, the step of "creating an object." It is error to read the scope of the independent claims more narrowly than the dependent claims, as the district court has done here. *Id.*

The district court also acknowledged that "the specification does not further refine 'selecting a label.'" A20. Nevertheless, as with its novel claim differentiation position, the district court looked to the specification's discussion of unrelated claim terms that have nothing to do with "selecting a label" in order to support the importation of this limitation. Relying on examples from the specification where a user selects ***objects*** to encrypt, the district court decided that ***objects*** must first be created in order to be selected. *Id.* The district court then

concluded by analogy that for ***labels*** to be selected, they must first be created. This logic fails for several reasons. First, it is black letter law that examples in the specification may not be used to import limitations into the claims. *See Phillips v. AWH Corp.*, 415 F.3d 1303, 1323 (Fed. Cir. 2005) ("[A]lthough the specification often describes very specific embodiments of the invention, we have repeatedly warned against confining the claims to those embodiments."). Second, the examples referenced by the district court relate to the selection of an ***object***, not the "selection of a ***label***" term construed by the court, and thus provide no support whatsoever for the importation of a limitation for this unrelated claim term. Third, the specification confirms that because of the dynamic nature of objects, the particular steps of the DCOM invention can be accomplished "on the fly," negating the implication of a pre-existing, static group of labels. A158, 6:16-17.

Finally, the district court resorts to extrinsic evidence that was not provided by any party or supported by an expert in the form of a dictionary definition. Again, the district court's reasoning is flawed. First, it is improper to rely on extrinsic evidence when the words in the claim are clear and unambiguous. *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1583 (Fed. Cir. 1996) ("where the public record unambiguously describes the scope of the patented invention, reliance on any extrinsic evidence is improper."). Second, there is no evidence that

a person of skill in the art would read the patent's disclosure and conclude that the same definition from the dictionary should apply.

In short, there is no justification in claims, specification or prosecution history to support the importation of the "pre-existing label" limitation into the claims, and the district court's construction must be reversed.

IV. The District Court Erred In Finding That The Limitation "Selecting A Label" Is Not Met By The Accused Acrobat Products

In addition to the procedural error caused by the district court granting summary judgment based on an argument not raised by Adobe or briefed by the parties, the district court also erred substantively in determining that a label is not selected with the accused Acrobat product. First, although the district court granted judgment of no infringement based on this limitation for the Asserted Method Claims ***and*** the Asserted System Claims, this limitation is not present in the system claims. Second, although TecSec was precluded from taking discovery on this limitation, the evidence in the record still demonstrates that a label is selected using Acrobat under both the plain and ordinary meaning of the term "selecting a label" as well as under the district court's erroneous construction.

A. The Asserted System Claims Do Not Contain The "Selecting A Label" Limitation

The district court made an egregious error in granting summary judgment of no infringement of the Asserted System Claims based on a limitation that is not

even present in the claims. The district court stated that "[e]very claim at issue contains either 'selecting a label' or 'selecting a first label' as an element. A39. This is demonstrably false. A plain reading of the claims shows that the system claims lack the "selecting a label" (or "selecting a first label") language of the method claims. Moreover, as can be seen below, representative method Claim 1 and representative system Claim 8 of the '702 Patent are unique and substantially different in claim scope.

Claim 1	**Claim 8**
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	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

Claim 1	**Claim 8**
authorization is granted.	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 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.

As can be seen, the system claims are much more than the method claims simply rewritten in a system format. While the method claims claim the use of an object-oriented key manager, the system claims require that access is restricted and encryption is applied by specific hardware and software modules based on "label conditions," a term that is not present in the method claims.

The district court acknowledges the differences in claim language only in a footnote, making the extraordinary and incorrect ruling that the claims use only "slightly different language to describe substantially the same invention." A39, n.22. In doing so, the district court read a limitation from the method claims into the system claims, holding that "all system claims include selection through the 'object labelling subsystem' and the 'object label identification subsystem,' which restrict access based on the 'label conditions' present in a selected label." *Id.* This is plain error – nothing in the system claims indicates that a label must be selected and pre-existing – only that the system has modules capable of labeling an object and identifying a label. Ironically, in supporting its construction for the term "label," the district court cited to the term "label conditions" in Claim 8 to support its finding that "selecting a label" and "selecting label components" were two separate concepts, implicitly acknowledging that Claim 1 and Claim 8 have different scope. A22. ("[d]ifferent terms generally have different

meanings")(citing *Chicago Bd. Options Exch., Inc. v. Int'l Sec. Exch., LLC*, 677 F.3d 1361, 1369 (Fed. Cir. 2012)).

Finally, the district court found that because the parties grouped the asserted claims together for the purposes of the motion for summary judgment, "any argument that they should be treated differently is deemed waived." A39, n.22. This ignores the fact that the "selecting a label" issue was raised *sua sponte* by the district court at the summary judgment hearing without any indication from the district court that it could potentially impact the system claims. Had Adobe raised this in its motion or had the district court provided advance notice of this issue, the parties certainly would have briefed and presented argument on whether the system claims inherently contain the limitations of the method claims. Without any prior notice, there can be no waiver. *See* Fed. R. Civ. P. 56(f).

This Court has confirmed in other cases that method claims and system claims describe unique inventions and it is error to read the steps of a method claim presumptively into a system claim. *See, e.g.*, *DSW, Inc. v. Shoe Pavilion, Inc.*, 537 F.3d 1342, 1348 (Fed. Cir. 2008) (finding it is error to import a limitation from one type of claim into another type of claim where the claim language does not indicate that the limitation is required); *Charles E. Hill & Assocs., Inc. v. Compuserve Interactive Servs., Inc.*, 33 Fed. Appx. 527, 534 (Fed. Cir. 2002) (finding system claims do not contain the steps of the method claims). The district court's

conclusion that the system claims implicitly contain the method claim steps, without notice to the parties and based on a cursory and incorrect analysis in a footnote, constitutes procedural and substantive error and must be reversed.

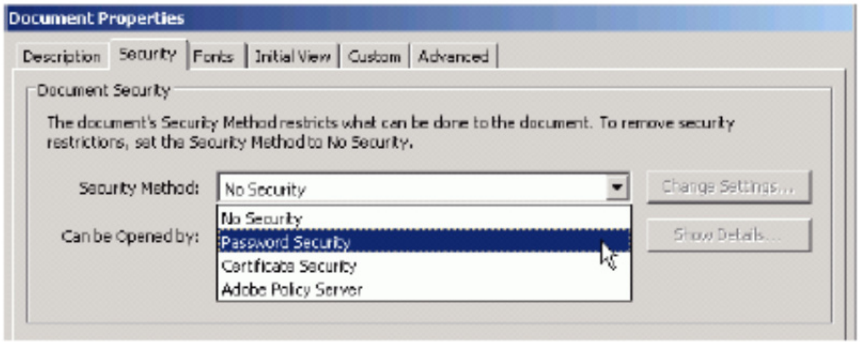
B. The Accused Products Meet The "Selecting a Label" Limitation Of The Method Claims

Each of the Asserted Method Claims requires the step of "selecting a label." Under either the plain language of the term "selecting a label" or the district court's flawed construction, a label must be selected in order to use document security with Acrobat. The district court found that the encryption dictionary for an encrypted PDF file created with Acrobat constitutes a "label" as claimed in the DCOM Patents, and the undisputed evidence showed that Adobe itself used Acrobat to create encrypted PDFs with these labels and instructed its customers to do the same. A37-A39. Thus, there can be no dispute that the applied label was "selected" or "chosen" at some point in the process. That alone creates a genuine issue of fact that precludes summary judgment.

As discussed supra at III.A, Adobe conceded at the summary judgment hearing that "***[s]electing a label*** is an action that is taken by the user of the system and ***is the action of saying I want to apply a particular kind of label*** or a particular label."). A3933, 3:12-14. This is precisely how the accused Acrobat products work. The very first selection that a user must make when applying document security to a PDF file is to choose the security method. A3581-A3582,

10:20-11:4; A3611, 45:12-15; A3747. As shown below, Acrobat presents the user with four choices: No Security, Password Security, Certificate Security, and Adobe Policy Server. A3747.

Figure 2 Security method selection



By choosing a particular security method, the user also selects a particular security handler and the type of encryption dictionary that will be applied to the subsequently encrypted PDF file or object. A3515-A3516, 34:6-35:14. Because the format for the particular types of encryption dictionaries are part of the PDF standard, they are necessarily pre-existing. *See* A3660; A3664; A3791, ¶ 69; A3792-A3793, ¶ 72.

The specific components that will be included with this label are chosen by the user following the initial security method election. A3581-A3583, 10:20-12:23; A3748-A3749; A3756. As Adobe's declarant, Mr. Kaufman, confirmed, the encryption dictionary is then built once the components are entered and the PDF file is saved.

> When you've selected a security handler and you ask to encrypt the document, the encryption happens when the document is saved. At the time the document is saved, the security handler is called to generate the contents of the encrypt[ion] dict[ionary], since it knows what it wants.

A3532-A3533, 51:22-52:4.

Thus, the encryption dictionary is a particular type of pre-existing label – it only has to be populated with the desired choices for encryption and access control (these choices being input by a user) based on the user-selected security method (the label components). Therefore, even under the district court's construction, a user enabling document security with Acrobat selects (i.e., chooses) a (pre-existing) label.

The district court misconstrued TecSec's argument at the hearing and failed to consider the evidence in the record on this issue. While the district court cited to other parts of the argument discussing how the label is created, the district court ignored TecSec's argument that was entirely consistent with the testimony of Adobe's 30(b)(6) witness and the argument of Adobe's counsel:

> [S]electing a label is selecting what that label will have on it. I mean, I have to tell the computer do I want the password security-type label that is going to invoke the password security handler, do I want a certificate-based label that is going to invoke the certificate security. Those are totally different labels. So I'm selecting which label I want the document to apply. If I want password security, I choose one type of label. If I want certificate security, I choose a separate type of label.

A3938, 8:5-14.

The district court also failed to acknowledge the testimony of TecSec's expert, who opined:

> By selecting the "encrypt with password" option, Acrobat selects an object oriented key manager, namely the password security handler. This selection also selects contents for the label, e.g., information to be written into the encryption dictionary.

A3772, ¶ 40.

Thus, the user must select the type of label to be used, and Acrobat must select and add the final label to the trailer of an encrypted PDF file. Dr. Jones testified that this process meets the limitation of "selecting a label." A3790, ¶ 67 ("Under the proper construction, it is my opinion that Adobe Acrobat selects a label for a PDF object, labels the encrypted PDF object, reads the label and determines access authorization based on the label."). Adobe failed to retain an expert and this testimony stands unrebutted. At a minimum, a genuine issue of material fact exists as to whether Acrobat meets this limitation, and the district court's judgment must be reversed.

V. The District Court Erred In Construing The Term "Label"

A. "Label" Is Correctly Construed As "An Identifier Associated With An Object"

District Court's Construction	Proper Construction
A series of letters or numbers, separate from but associated with the sending of an object, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated object.	An identifier associated with an object.

The term "label" appears in each of the asserted claims and is properly construed as "an identifier associated with an object." This construction comports with the plain and ordinary meaning of the term. The district court erred in importing a limitation from the background section of the specification that pertains to the prior art and not the claimed invention. This construction is incorrect because the specification clearly distinguishes the DCOM invention from the prior art and uses a more flexible and expansive definition of label. Indeed, both Adobe and the district court had to alter the language used to describe the prior art in an effort to make it fit the DCOM invention. Even with that alteration, the district court's construction is nonsensical when viewed in the context of the DCOM Patents' claims.

The specification does not give the term a special meaning separate from the plain and ordinary meaning of label. The background section of the DCOM

Patents, when discussing the prior art, describes a narrower type of label than that used by the prior art for the sending of messages. *See, e.g.*, A156, 2:58-61. However, the remainder of the specification when describing the DCOM invention confirms that the claimed label in the DCOM Patents is broader and more flexible than that disclosed in the prior art. For example, the specification in the '452 Patent states:

> 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 fie. The label may define the device at which the fie 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.

A174, 5:16-27. TecSec's proposed construction – "an identifier associated with an object" – comports with the plain and ordinary meaning of the word "label" and is in line with the usage of the term as used in the DCOM Patents.

Other portions of the specification support this construction as well. Examples 1, 2, and 3 in the '702 Patent describe the simple act of selecting a label so that the system can identify the object. A159, 8:5; A160, 9:14; A160, 10:20, A2789, ¶ 64. Indeed, when describing the claimed invention (referred to in the specification as the "present invention"), the patentee never used language to

indicate that he was acting as his own lexicographer such that a special meaning should be given for the term "label."

Extrinsic evidence also supports this definition. *See Phillips*, 415 F.3d at 1314 (holding courts can also consider "extrinsic evidence concerning relevant scientific principles, the meaning of technical terms, and the state of the art."). For example, the Microsoft Press Computer Dictionary, 1991, defines label as "an identifier." A3736-A3738. In addition, TecSec's expert, whose testimony is unrebutted, confirmed that the term "label" is used in the DCOM Patents in accord with the plain and ordinary meaning of the word, and that TecSec's proposed construction is how a person of skill in the art would understand the word "label." *See* A3789, ¶¶ 64-65.

B. The District Court's Construction Improperly Reads In A Limitation From A Description Of The Prior Art In The Background Section Of The Specification

The district court's construction is improper because it reads in a limitation for a portion of the background section of the '702 Patent that is referring to a prior art patent. *See* A156, 2:34-40; *see also* A26. There can be no doubt that the language in the background section (which the district court then modified for its construction) refers to the prior art U.S. Patent No. 5,396,707, not the claimed invention. *See* A156, 2:58-61 ("A system such as that described above is disclosed in U.S. patent application Ser. No. 08/009,741 . . ."). The inventor of the '702

Patent used this portion of the background to explain how the DCOM invention was an improvement over the prior art '707 patent. *See* A156, 2:30-66; *see also* A3789-A3790, ¶ 66.[6] It is improper to assume that the background discussion of prior art defines the claimed invention. *See, e.g., Emblaze Ltd. v. Apple, Inc.*, No. 5:11-cv-01079-PSG, 2014 U.S. Dist. LEXIS 144197, at *15-*16 (N.D. Cal. Oct. 9, 2014) (holding the background section in a patent describes prior art and should not be assumed to describe claimed invention).

The district court places great stock in the fact that the patentee used the words "this invention" in the background section as conclusive evidence that the patentee was describing the claimed invention. The court's conclusion is both factually and legally wrong. In each case where the patentee was describing the claimed invention, the patentee deliberately used the words "the present invention." *See, e.g.*, A156, 1:6; A157, 3:12, 18, 21, 25, 29 and 42; 4:14, 31, 38 and 40; A158, 5:18. However, in the background section, when describing the prior art '707 patent, the inventor used the term "this invention" to make clear that he was not referring to the claimed DCOM invention. A156, 2:34-35.

[6] Importantly, the prior art '707 patent has a different inventorship than the patents-in-suit and as such does not contribute to the claimed embodiments. Roy Follendore is the inventor on the prior art '707 patent and Greg Shanton is the inventor of the '702 Patent (as well as the other patents-in-suit). The prior art '707 patent refers to Net Shield while the patents-in-suit describe DCOM. *See* A158, 5:18-19. These are clearly two different inventions.

As further proof that the district court's construction is incorrect, the construction alters the language from the specification in a significant way. First, the '702 Patent specification states that a label as used with the prior art '707 patent is ". . . separate from but associated with the sending of a message." A156, 2:36-37. In the district court's construction, "message" has been altered to "object," because the district court acknowledges that the claims are drawn to labeling objects, rather than messages. This reinforces the conclusion that the portion of the specification from which the district court's limitation was drawn was referring to the prior art '707 patent, which pertained to the sending of messages, rather than the DCOM invention, which pertains to the securing of objects.

To further reinforce this point, the district court's construction is nonsensical in view of the '702 Patent's disclosure because there is no requirement in the claims or the specification of the DCOM Patents that objects must be "sent." Therefore, "label" as used in the DCOM Patents cannot be limited to the "sending of an object." The district court asserts that its definition does not require the sending of an object, but only that the label be "associated with the sending of an object." A31. This is a distinction without a difference. There is no support for reading a "sending" limitation into the claims based solely on a description of the prior art. *See Phillips*, 415 F.3d at 1320 (finding a "cardinal sin" of patent law is

reading limitations into the claims from the specification) (relying on *SciMed Life Sys., Inc. v. Advanced Cardiovascular Sys., Inc.*, 242 F.3d 1337, 1340 (Fed. Cir. 2001)). The district court's construction should be reversed, and TecSec's proposed construction should be adopted.

VI. The District Court Erred In Holding That The Encryption Dictionary With Password Security Is Not A Label

In its infringement contentions, TecSec identified the encryption dictionary of Acrobat, as implemented with either password security or certificate security, as the claimed "label." In its summary judgment opinion, the district court held that the encryption dictionary with certificate security met the construction of "label," but found that Acrobat's encryption dictionary when used with password security did not meet the court's construction for "label." A38-A39. Specifically, the district court held that a password does not identify a specific user. A38. This decision was in error because under either the district court's improper construction of "label" or TecSec's proper construction, the encryption dictionary with password security constitutes the claimed "label."

There is no genuine dispute that the encryption dictionary with password security is an identifier associated with an object, and therefore meets TecSec's proposed construction of label. But the encryption dictionary also meets the district court's construction, because it contains the information needed to verify whether a user or group is authorized to view a particular PDF file. A3790-A3791,

¶ 68. In fact, the plain and ordinary meaning of "password" is "security method that identifies a specific authorized user . . . by a unique string of characters that a user types as an identification code." A3740-A3741; A3743.

If password security with Acrobat is used, an object is selected, a password is selected, a key is generated, the object is encrypted, and an encryption dictionary is created for that object and appended to the object. A3583, 12:13-23; A3587-A3588, 17:1-18:23. Acrobat with password security uses two different passwords — "O" and "U" — which stand for "owner" and "user," respectively. A2492, ¶ 24. Thus, these passwords are different and identify different people. These passwords by their nature grant the holder different levels of access to the Acrobat document. A2492, ¶ 24; A3522, 41:2-18. Accordingly, Acrobat treats an entered "U" password as belonging to a user and an entered "O" password as belonging to an owner. Upon an access attempt, Acrobat's security handler will check the encryption dictionary to attempt to verify the password, which identifies a particular authorized user. A3538, 66:2-6; A3592-A3593, 22:16-23:17; A3594, 24:22-24; A3638; A3660. There is no requirement in the DCOM Patents that the "label" identify a specific person by name, and it was error to import this limitation. *See, e.g., Phillips*, 415 F.3d at 1316 (holding claims should be given their "broadest reasonable construction 'in light of the specification as it would be interpreted by one of ordinary skill in the art'" (quoting *In re Am. Acad. Of Science*

Tech Ctr., 367 F.3d 1359, 1364 (Fed. Cir. 2004))). In fact, even the background section upon which the district court relies makes that clear. A156, 2:30-57 (discussing labels that can be to restrict based on roles, groups, etc.).

Acrobat's encryption dictionary used with password security meets the district court's definition of label because the "U" and "O" passwords identify a person (i.e., user or owner). TecSec's expert confirmed this in his unrebutted declaration:

> Accordingly, the encryption dictionary contains the password that identifies a person who can access a document because only a person with the correct password can access and decrypt a PDF document. Thus, the possession of the password is a form of identifying an authorized user.

A3791-A3792, ¶70 (citing A3651-A3680; A3748; A3524-A3525, 43:4-44:5; A3538, 66:2-6; A3554, 83:14-25; A3561, 99:7-11; A3561-A3562, 99:24-100:2; A3575-3576, 121:2-122:2; A3594, 24:2-4; A3596, 26:4-6). In conclusion, TecSec's expert testified:

> It is my opinion that Adobe Acrobat's encryption dictionary which contains the password (i.e., the owner and/or user password), permission settings and encryption algorithm provides access control and meets the "label" limitation.

A3790-A3791, ¶ 68. Adobe offered no expert testimony to rebut TecSec's expert opinion.

The undisputed evidence in the record shows that a genuine issue of fact exists as to whether Acrobat's encryption dictionary with password security meets

the "label" limitation of the DCOM Patents, and the district court's grant of summary judgment on this ground must be reversed.

VII. The District Court Erred In Construing "Object-Oriented Key Manager"

District Court's Construction	Proper Construction
A software component that manages the encryption of an object, on an object-by-object basis, to achieve multi-level security, including the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys.	Software that controls access to the algorithm used to encrypt and decrypt objects.

Although it did not form the basis for any of the district court's summary judgment decisions, the district court construed the term "object-oriented key manager," which appears only in the Asserted Method Claims. "Object oriented key manager" should be construed as "software that controls access to the algorithm used to encrypt and decrypt objects." The district court's construction imports an unwarranted limitation from the specification pertaining to the concept of "key management," rather than the claimed structural element of a "key manager."

"[T]he claim construction inquiry . . . begins and ends in all cases with the actual words of the claim." *Renishaw PLC v. Marposs Societa' per Azioni*, 158 F.3d 1243, 1248 (Fed. Cir. 1998). Here, Claim 1 of the '702 Patent is illustrative.

The steps of the claim require "accessing an object-oriented key manager," as well as the related steps of "selecting an encryption algorithm" and "encrypting the object according to the encryption algorithm." The only function that the object-oriented key manager performs with respect to Claim 1 is controlling access to the algorithm used for encryption.

The specification further supports TecSec's construction as informed by the claims. Here, during prosecution the inventor explicitly defined key manager and key management as separate things in response to a § 112 rejection, as follows:

> 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 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.

A3719-A3720; A3729. The patentee further amended the specification to include this language at col. 1, line 61 – col. 2, line 4. *See* A156, 1:61-2:4. Thus, the express language added by the patentee makes a distinction between "key management" and a "key manager," and provides the controlling definition for "key manager."

Importantly, TecSec's proposed construction is exactly the express definition provided by the patentee, with one modification (omitting "plain text")

to account for the scope of the claims. TecSec's construction omits the words "***plain text*** objects" from its definition in reference to the types of objects that can be secured using DCOM. TecSec omitted this language given the broader construction of object already provided by the district court (construed as "any distinct, separate entity"). Thus, while the specification describes "key manager" in terms of plain text objects, the claims recite "objects" more generally (as evidenced by use of the term "object-oriented key manager").

TecSec's proposed construction is further supported by its unrebutted expert declaration, which states: a person of skill in the art reading the specification and prosecution history would understand that the key manager is "software that controls access to the algorithm used to encrypt and decrypt objects." *See* A3787, ¶ 58. The district court did not acknowledge this evidence in the record, however.

The district court's construction confuses the claimed "key ***manager***" with "key ***management***." This does not comport with the language of the claims and written description, however, because the claims only recite a "key manager." Nothing in the specification indicates that the "key manager" is limited to software that must perform every one of the enumerated list of functions related to the general concept of key management. *See* A159, 8:11-12; A160, 9:20-21 and 10:25-26; A3787-A3788, ¶ 60. Indeed, the specification of the '702 Patent provides examples of the object oriented key manager performing various

functions in exemplary embodiments, which differ from those of the district court's construction. *Id.* The district court's construction, therefore, reads in limitations from the claims improperly. *See Phillips*, 415 F.3d at 1320.

In an attempt to support its flawed construction, the district court explains that the inventor must have, "for whatever reason," limited the embodiment claimed in Claim 1 from the broader list of functions that the district court imported into its construction (in other words to a subset of the various aspects of key management). A25. The district cites no support for this conclusion. The limitations imported by the district court's construction cannot be correct because there is no mention of the key manager performing functions such as "distributing" and "destroying" keys within the claims or the specification.

Thus, the district court's construction for "object-oriented key manager" should be reversed and TecSec's construction adopted.

VIII. The District Court Erred In Construing "Display Header"

District Court's Construction	Proper Construction
A header for making visually perceptible to a user.	No construction necessary; plain and ordinary meaning.

The district court merged its construction of "displaying" into its construction of "display header" rather than separately construing the two terms. A35. The district court construed "display" to mean "making visually perceptible to a user." This was based on the plain and ordinary meaning of the term — which

TecSec does not dispute is the proper construction. However, TecSec also argued for the same principle to be applied to the term "display header." A3485-3486. The district court did not follow this proposal and instead the district court construed "display header" to mean "a header for making visually perceptible to a user." A35.

"Displaying" is a verb while "display," in the context of "display header," is an adjective. Adobe proposed that the two terms have the same meaning. The district court followed Adobe's proposal and, while acknowledging that the terms are different parts of speech, still proceeded to apply the same construction for both. As a result, the district court's construction of "display header" results in a nonsensical construction.

As noted by TecSec's expert declarant, Dr. Jones, the claim in which the term appears (i.e., claim 1 of the '755 Patent) does not require any display or other conveyance to a user. A3797, ¶ 84. The claim only requires a creation of the display header. *Id*. The district court's construction lacks any support in the specification and fails to render the correct meaning of the term. By reading a limitation into the claim that is not present in the original language, the district court erred. *See, e.g.*, *Phillips*, 415 F.3d at 1323.

The district court's construction of "display header" should be reversed and the plain and ordinary meaning of the term applied.

IX. The District Court Erroneously Decided *Sua Sponte* That The "Labelling" Limitation Is Not Met

Footnote 23 presents another issue raised *sua sponte* by the district court. In this footnote, the district court addressed the claim term of "labelling." A40, n.23. While the term "labelling" was raised for construction by Adobe, this term did not form the basis of any non-infringement argument briefed by Adobe and the district court did not raise this issue at the hearing.

The district court construed "labelling" as urged by TecSec to mean "attaching a label." But the court proceeded to create its own non-infringement position without any notice to the parties. In a footnote, the district court held that "Acrobat does not attach an encryption dictionary to an encrypted PDF document; instead, it inserts the encryption dictionary in the (pre-existing) trailer for the file." *Id.* The district court then stated that the encryption dictionary is not attached to the object but rather is a part of the object. *Id.* To the extent that this footnote is intended to set forth a finding of no infringement based on the "labelling" limitation, it is once again both procedurally and factually flawed.

As to the procedural deficiencies, Adobe did not raise this issue in its motion, TecSec was permitted no discovery on the issue, the district court never mentioned the issue at the hearing, and no argument was heard on the issue. There can be no doubt that the safeguards of Rule 56(f) were not followed, notice was not

provided, and summary judgment is inappropriate. *See Mikkelsen Graphic*, 541 Fed. Appx. at 972.

Substantively, the district court first made the incorrect statement that "each asserted claim requires some form of 'labelling.'" A40, n.23. The system claims do not require the step of "labelling." Instead, the system claims have an "object labelling subsystem" that "compris[es] logic means for limiting object access, subject to label conditions." *See, e.g.*, '702 Patent, Claim 8 (inside cover). In addition, it is undisputed that in the Accused Products, when a user applies the document security feature of Acrobat, the encryption dictionary (which the district court found to be the claimed "label") is added to the PDF file's trailer, which itself is attached to the encrypted PDF content. A2483-2484, ¶¶ 7-8; A3514, 33:5-17; A3790-A3791, ¶ 68. Accordingly, the "labelling" limitation is met when the encryption dictionary (i.e., the label) is created and added to the trailer of the encrypted PDF file. Because of the lack of notice and the substantive errors, and to the extent that the district court's commentary in footnote 23 was intended as a finding of no infringement, it must be reversed.

X. This Case Should Be Reassigned On Remand

TecSec respectfully requests that if the district court's judgment of no infringement is reversed, this case be remanded with instructions to reassign the case to a different judge pursuant to the power vested under 28 U.S.C. § 2106.

This Court evaluates a request to transfer to a different judge under the law of the regional circuit, in this case the Fourth Circuit. *TriMed, Inc. v. Stryker Corp.*, 608 F.3d 1333, 1343-44 (Fed. Cir. 2010); *Research Corp. Techs., Inc. v. Microsoft Corp.*, 536 F.3d 1247, 1255 (Fed. Cir. 2008). The Fourth Circuit considers the following three factors: (1) whether the original judge would reasonably be expected upon remand to have substantial difficulty in putting out of his or her mind previously-expressed views or findings determined to be erroneous or based on evidence that must be rejected, (2) whether reassignment is advisable to preserve the appearance of justice, and (3) whether reassignment would entail waste and duplication out of proportion to any gain in preserving the appearance of fairness. *United States v. Guglielmi*, 929 F.2d 1001, 1007 (4th Cir. 1991).

Regarding the first factor, if this case is remanded, it appears that the district court judge will have substantial difficulty in putting out of her mind previously-expressed views against TecSec. In her first summary judgment opinion, the district court judge found, *sua sponte*, that 16 claim terms were in means-plus-function format, and without any analysis found each of the means-plus-function claims to be lacking corresponding structure. This Court reversed, finding that two of the claim terms were not in means-plus-function format, and that for the remaining 14 claim terms, the algorithms disclosed in the specification were so detailed and complete that "[s]hort of providing source code, it is difficult to

envision a more detailed disclosure." *TecSec, Inc. v. Int'l Bus. Mach. Corp.*, 731 F.3d 1336, 1349 (Fed. Cir. 2013).

In her second summary judgment opinion, the district court judge found, *sua sponte*, that TecSec would be unable to prove infringement based on a claim construction and non-infringement argument that Adobe declined to brief, and based on an issue on which the district court prohibited TecSec from taking discovery. As set forth above, this second grant of summary judgment is erroneous, and should be reversed.

In that same second summary judgment opinion, and in a subsequent hearing, the district court indicated, *sua sponte*, that it believed that the Accused Product also does not infringe for other reasons not raised by Adobe or briefed by the parties, and that the asserted DCOM Patents may be invalid under 35 U.S.C. § 101, even though no party had ever raised either defense. A12, n.10; A4363, 3:16-22 ("I do recognize that we dropped in a footnote, and I'm sure TecSec has read that footnote carefully, that we think under the *Alice* decision, which is new since the case was first filed, that the patents at issue might, in fact, have some significant vulnerabilities at this point, but that's not the case that we addressed, and I think that issue has to be left for another day possibly between other parties and TecSec."). In support of her invalidity predisposition, the district court judge cited to one of her own cases (currently on appeal), where she found the asserted

patents invalid on remand after a prior grant of summary judgment of no infringement was reversed by this Court. A12, n.10 (citing *Amdocs (Israel) Ltd. v. Openet Telecom, Inc.*, 56 F. Supp. 813, 820 (E.D. Va. 2014) (J. Brinkema)). The district court here appears to have predetermined that a third and fourth grant of summary judgment may be appropriate if this Court reverses, without the benefit of any discovery, briefing or expert testimony. As a result, it is highly unlikely that the district court judge will be able to be impartial for the remainder of the case since a pattern of error exists based on the district court judge's propensity to grant summary judgment *sua sponte*. *See, e.g.*, *Coward v. Jabe*, 532 Fed. Appx. 328, 331 (4th Cir. 2013) (reversing this same district court judge for second grant of summary judgment without providing proper notice under Fed. R. Civ. P. 56(f)).

Regarding the second factor, reassigning the case would preserve the appearance of justice. If reversed here, the district court judge will have been reversed twice on summary judgment, after deciding multiple issues *sua sponte* without proper notice, and has already signaled that she has additional concerns about the validity of the patents and infringement, even before any party raised these issues. Thus, it appears unlikely that TecSec would receive a fair opportunity to have its claims decided on remand without reassignment. *See* TriMed, 608 F.3d at 1344 (reassigning to different judge on remand after reversing on summary judgment twice); *Guglielmi*, 929 F.2d at 1007 ("[t]he procedural history of the

instant case has left us convinced that, absent some affirmative act on our part, we shall be locked in an endless cycle of remands and renewed appeals that will move us no closer to discerning a meaningful exercise of the sentencing judge's discretion").

Regarding the third factor, reassignment to a new judge would not entail waste or duplication since the case is still in the early stages, with the summary judgment motion against Adobe having been filed before discovery commenced, and with eight additional defendants having been stayed for more than five years without discovery underway.

The district court has now twice entered summary judgment against TecSec, deciding issues *sua sponte* without briefing on the arguments that formed the basis of the court's decisions, and has indicated that a third and fourth grant of summary judgment may be forthcoming if the case is remanded. Under these unusual circumstances, reassignment is warranted.

CONCLUSION

For the above-mentioned reasons, TecSec respectfully requests from this Court the following relief:

(1) an order setting forth the proper constructions for the DCOM Patents;

(2) an order vacating the district court's Order granting judgment of no infringement; and

(3) on remand, instructions to reassign this case to a different judge in the Eastern District of Virginia.

Respectfully Submitted

/s/ Michael A. Oakes

Michael A. Oakes
Michael A. O'Shea
HUNTON & WILLIAMS LLP
2200 Pennsylvania Avenue, N.W.
Washington, D.C. 20037
Telephone: (202) 955-1894
Facsimile: (202) 857-3891

Gregory N. Stillman
HUNTON & WILLIAMS LLP
500 East Main Street
Suite 1000
Norfolk, Virginia 23510
Telephone: (757) 640-5300
Facsimile: (757) 625-7720

Counsel for TecSec, Inc.

August 3, 2015

ADDENDUM

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

TECSEC, INC., Plaintiff, v. INTERNATIONAL BUSINESS MACHINES CORPORATION, et al., Defendants.	))))))))	1:10cv115 (LMB/TCB)

MEMORANDUM OPINION

Before the Court is defendant Adobe Systems Incorporated's Motion for Entry of its Proposed Claim Constructions and for Summary Judgment of Non-Infringement. [Dkt. No. 710] ("Motion for Summary Judgment"). For the reasons that follow, the motion for entry of Adobe's proposed claim constructions will be granted in part and the motion for summary judgment of non-infringement will be granted.

I. BACKGROUND

TecSec, Inc. ("TecSec" or "plaintiff") is the owner of United States Patents Nos. 5,369,702 (the "'702 Patent"), 5,680,452 (the "'452 Patent"), 5,717,755 (the "'755 Patent"), and 5,898,781 (the "'781 Patent").[1] Because all four patents address features of what the written description describes as a "Distributed Cryptographic Object Method" ("DCOM"), for the purposes of this litigation the four patents are collectively referred to as the "DCOM Patents."[2]

[1] The '452 Patent and '755 Patent are direct continuations of the '702 Patent, and the '781 Patent is a direct continuation of the '755 Patent.

[2] TecSec also owns, and alleged that Adobe infringed, United States Patent No. 6,694,433, directed to an "XML Encryption Scheme" (the "XML Patent"). While this litigation has been pending, TecSec filed for supplemental examination of the XML Patent and the Patent Office issued a "Final Rejection" of all asserted claims of that patent. Although that decision was termed a "final" rejection, TecSec is currently in the process of appealing the rejection of the

TecSec alleges that Adobe Acrobat ("Acrobat"), a piece of software made by defendant Adobe Systems, Inc. ("Adobe" or "defendant") infringes the DCOM Patents.

A. Encryption Basics

Because the DCOM Patents describe a method for securing digital objects using encryption, some background on encryption is helpful to understanding the claimed invention. Encryption is the process whereby original data, referred to as "plaintext," can be transformed into encoded data, referred to as "ciphertext." Charles P. Pfleeger & Shari Lawrence Pfleeger, Security in Computing 35-37 (Pearson Education, Inc. 3d ed. 2003). Encrypting data helps to prevent unauthorized users from accessing that data.

Digital files can be encrypted using encryption "keys." Id. Like physical keys for physical locks, encryption keys can be used to "lock" data so that unauthorized users cannot access it, and also allow authorized users to "unlock" the data. Id. at 37. The encryption key is applied to the plaintext message pursuant to an encryption algorithm, resulting in ciphertext. Id. An authorized user can then decrypt the ciphertext to gain access to the plaintext. Id. In the simplest form of key-based encryption, referred to as a "symmetric key" system, the key used to encrypt the plaintext information is also used to decrypt the ciphertext. Id. The standard housekey system is a physical symmetric key system, in that the same key is used to both lock and unlock the door. Systems which use passwords, secret phrases which an authorized user must enter to gain access to the information, also tend to be symmetric key systems. In more complicated encryption systems, a different key is used to encrypt the plaintext than is used to decrypt the ciphertext. Id. Those systems are referred to as "asymmetric key" systems because

---

XML Patent to the Patent Trial and Appeal Board, Status Report Regarding U.S. Patent No. 6,694,433 [Dkt. No. 758], and has agreed to drop its claims against Adobe involving the XML Patent and to proceed against Adobe on an accelerated schedule regarding the DCOM Patents.

different keys are used to encrypt and decrypt. Id. The equivalent physical system would have one key used to lock a door, but a different key used to unlock that same door. See id.

Digital certificates, which are particularly relevant to the present action, use an asymmetric key system to protect information. A digital certificate is "similar to a passport or driver's license" in that it can be used with confidence to identify an individual or organization. Umesh H. Rao & Umesha Nayak, The InfoSec Handbook 172 (Apress Media 2014). The certificate is issued by a trusted third-party "certificate authority" which verifies the identity of the individual before the certificate is issued. Id. at 171-72. Digital certificates can be used to secure data through public key cryptography, which is an asymmetric key system. Id. at 170-71. The certificate authority issues an individual a public key and a private key. The public key can be accessed by anyone, but the private key is only available to the individual. Id. Data encrypted with an individual's public key may only be decrypted with that individual's private key, and data encrypted with an individual's private key may only be decrypted with that individual's public key. Id. Accordingly, the public key from an individual's digital certificate can be used to encrypt a document. Because only the individual has the private key, which is the only way to decrypt the document, the sender can be sure that the content of the document will remain secure and will only be seen by the intended recipient. See id. "An individual or organization may have any number of certificates issued by different [certificate authorities]." Id. at 172.

B. The DCOM Patents

The DCOM Patents describe a method and system which allows a user to flexibly secure computer objects by encrypting a first object, nesting the encrypted first object into a second object, and then encrypting the second object. '702 Patent col. 4 lines 14-34. The encryption and nesting process may be repeated any number of times, and each encryption iteration can use a

different encryption key. Id. col. 11 lines 40-55. As a result, users can securely transmit a single object containing multiple other objects, allowing recipients to decrypt only those objects to which each particular recipient is entitled access. Id. In the context of a company, for example, "[e]very 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." Id. col. 11 lines 44-48. Figure 3, which illustrates the process at a high level, shows "an encrypted object that contains a web of embedded encrypted objects nested within the other encrypted objects[:]"

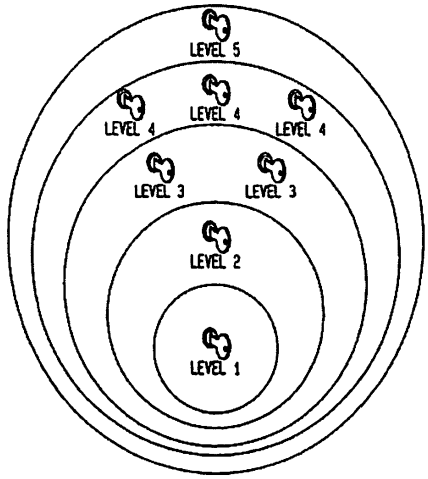


FIG. 3

Id. col. 11 lines 18-30. The object in Figure 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." Id.

In addition to multi-level encryption, the DCOM Patents use labels to provide protection:

> [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.

Id. col. 2 lines 44-56. Accordingly, after encryption a user may select a label for an object, and the label is attached to the object. Id. Claim 1 of the '702 Patent illustrates the labelling and encryption systems working together to ensure security:

> 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.

Id. col. 12 lines 1-15. TecSec admits that "[c]laim 1 of the '702 [P]atent is exemplary." TecSec's Brief in Opposition to Adobe Systems Inc.'s Proposed Claim Constructions and Motion for Summary Judgment of Non-Infringement [Dkt. No. 741] ("MSJ Opp'n") at 8.

C. Adobe Acrobat

"Adobe offers a family of software applications under the name Acrobat that are used to view, create, manipulate, print and manage PDF documents." Declaration of Marc Kaufman [Dkt. No. 712] Att. 1 ("Kaufman Dec.") ¶ 4.[3] PDF, which stands for "portable document format," is an open source standard file format which allows a document to be electronically

[3] TecSec's expert, Dr. Mark Jones, does not offer any opinions regarding the general PDF standard or describing its general operation; however, Dr. Marc Kaufman, Adobe's Fed. R. Civ. P. 30(b)(6) witness, does provide such background and TecSec does not dispute Dr. Kaufman's description. Accordingly, the basic structure of a PDF document is not in dispute.

displayed "with the same form, font and layout as it would if it were printed, regardless of the particular software, hardware or operating system used to display the document." Id. ¶ 2. To view a document stored in the PDF format, a user must have a program that can interpret and display a PDF document (a "PDF reader") installed on her computer. Id. ¶ 4. Because PDF is an open file format, any company or programmer could create their own PDF reader. Id. Acrobat is Adobe's implementation of a PDF reader. Id. Successive versions of Acrobat have been introduced as the PDF standard evolved. See id. ¶ 5.

Understanding the PDF standard is necessary to understand how Acrobat functions. A PDF document consists of four components: objects, file structure, document structure, and content streams. Id. ¶ 6. The components are shown below:

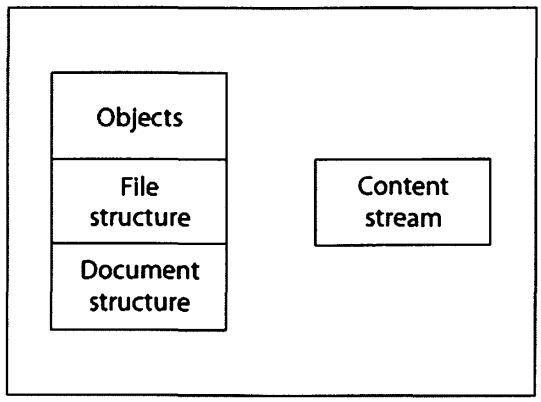
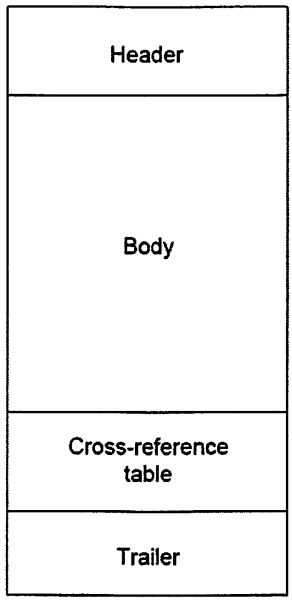


Figure 1 – PDF Components

Id. Objects "build" most of the elements that a user sees when a PDF document is opened. There are seven different types of objects. Id. ¶¶ 6-7. Relevant to this civil action is the "dictionary" type of object, which is a table containing pairs of objects.[4] Id. ¶ 7f. The document structure

[4] "A dictionary object is an associative table containing pairs of objects, known as the dictionary's entries. The first element of each entry is the key and the second element is the

"consists of a number of data structures that define the structure of the document itself, such as which objects belong on a given page." Id. ¶ 9. Content streams contain instructions describing the appearance of a page in a PDF document. Id. ¶ 9. File structures "determine[] how objects are stored in a PDF document and how they are accessed." Id. ¶ 8. The file structure for a PDF document is shown below:



Id. The "body" contains all of the objects that make up the document's content, the "cross-reference table" identifies the locations of objects within the document, and the "trailer" specifies the location of the cross-reference table and includes a "trailer dictionary" which organizes references to important parts of the file. Id.

---

value." Id. ¶ 7f. In that manner, the structure of a dictionary object is the same as the structure of an English language dictionary, where a word is the "key" and the word's definition is the "value." A dictionary object uses that flexible structure "to collect and tie together the attributes of a more complex data structure, such as a font or a page of a document[.]" Id. "[E]ach entry in the dictionary specif[ies] the name and value of an attribute of the more complex structure." Id.

A PDF document may have other files, including other PDF documents, attached to it. Id. ¶ 11. A file which is attached to a PDF document is held within the body section of the PDF document, and is stored like any other content. Id. ¶¶ 12-13. A user can see and select any attached files when the PDF document is opened in Acrobat. Id. If the user selects and tries to open an attached file, the file "is opened like any other file on a computer." Id. ¶ 13. For example, if a Microsoft Word file is attached to a PDF document and the user attempts to open the Microsoft Word file, the file will be opened by Microsoft Word. Id. Similarly, when a user attempts to open a PDF document attached to another PDF document, the attached PDF document will be opened in another session of Acrobat. Id. ¶ 13. If the attached PDF document also has attachments, those attachments are treated in the same way. Id.

In addition, a PDF document may also contain metadata. Metadata is "global information about the document (as opposed to its content or structure)," and may include "the document's title, author, and creation and modification dates." PDF Reference, Adobe Portable Document Format Version 1.7 at 843 (Adobe Systems Incorporated 6th ed. 2006).[5] Metadata is used to "assist in cataloguing and searching for documents in external databases." Id. Metadata is stored either within a dictionary in the trailer of the document, id., or within the body of the document. Id. at 845-47.

D. Encrypting a PDF Document Using Adobe Acrobat

The parties agree that Acrobat has the capability to encrypt a PDF document. Kaufman Dec. ¶ 18; Declaration of Mark T. Jones, Ph.D [Dkt. No. 738] ("Jones Dec.") ¶¶ 22-23. When using Acrobat there are three options for encryption: (1) encrypt everything in the file (including

[5] Available at http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/pdf_reference_1-7.pdf.

metadata and any attachments); (2) encrypt everything except the metadata; and (3) encrypt only the attachments. Jones Dec. ¶ 36. One method for encrypting a PDF document is to use a password. Id. ¶¶ 40, 45. If the document is encrypted using a password, the document can be protected by both an "owner" password, which gives a person unrestricted access to the document and any function allowed by Acrobat, and a "user" password, with which the owner can limit what a user may do once he has access to the file. Declaration of Michael A. Oakes [Dkt. No. 737] Ex. 2 ("Kaufman 30(b)(6) Dep.") at 22-25. For example, the owner of a document can use Acrobat to allow those who enter the user password only to read but not to print the document. If a person enters the owner password, however, she can use any function such as saving, printing, editing, etc. In essence, creating an owner and user password creates two "tiers" of use for a document. When the owner of a PDF document selects the option to encrypt that PDF document, the owner is prompted to create both passwords.

After a PDF document has been encrypted, information necessary to decrypt the document is stored in the trailer section of the file in a special object called the "encryption dictionary." See, e.g., id. at 17; Kaufman Dec. ¶ 22; Jones Dec. ¶ 68. "[T]he encryption dictionary is not visible to users, but is part of the internal structure of the document." Kaufman Dec. ¶ 22. If the PDF document is password-secured, the encryption dictionary includes values which are used to test the owner and user password (the "O key" and "U key," respectively) to determine if the recipient entering the password is authorized to access the document.[6] Kaufman 30(b)(6) Dep. at 17. The encryption dictionary is not generated immediately after the owner of the PDF document creates the passwords. See id. at 14. Instead, Acrobat automatically creates

[6] The encryption dictionary includes other values as well, including the permission settings and the algorithm used to encrypt the PDF document. Jones Dec. ¶ 68.

the O key, U key, and encryption dictionary only after the user chooses to "save" the PDF document. See e.g. id. at 14-17. If the user does not save the document, the encryption dictionary does not exist at all. See id. In other words, the encryption dictionary does not exist until the user selects "save." Moreover, neither the owner nor the user passwords themselves are directly included in the encryption dictionary, only the values assigned to those passwords are included in the encryption dictionary. Id.

A PDF document may also be encrypted using digital certificates. Jones Dec. ¶¶ 40, 45. The process for using digital certificates to encrypt a PDF document is slightly different than the process for using a password. Once the user decides which parts of the PDF document are to be encrypted,[7] the user selects the digital certificates of the recipients to whom the PDF document will be sent.[8] Kaufman 30(b)(6) Dep. at 45. The user can also sort the recipients into groups, and can restrict the functions available to each group. Id. at 49-50. Similar to password security, however, encryption of the PDF document and creation of the encryption dictionary do not occur until after the user chooses to save the document. Id. at 54, 63. When the user saves the document, Acrobat generates a random number, referred to as a "file key," which is used to encrypt the PDF document.[9] Id. at 51-52, 61. The file key is encrypted for each recipient with that recipient's public key. Id. at 52. Acrobat then automatically creates the encryption dictionary, and the encrypted file keys are inserted into the encryption dictionary along with

[7] As with password security, Acrobat can encrypt either everything in the file (including metadata and any attachments), everything except the metadata, or only the attachments using certificate security.

[8] The digital certificates may be stored with Adobe, or on the user's hard drive, or on a storage device attached to the computer. Id. at 46.

[9] Once the file key is generated, Acrobat follows the same process to encrypt the PDF document with the certificate as it does when encrypting a PDF document using password security. Id. at 61.

identifiers for the related digital certificates (although not the certificates themselves). Id. at 52-54. By grouping recipient certificates together and assigning those groups functions that they can perform, the owner of a PDF document may also use certificate security to create tiers of use for a document.

II. PROCEDURAL HISTORY

TecSec originally filed this action against thirteen defendants asserting infringement of the DCOM Patents as well as other patents not relevant to the current motion. To make the litigation manageable, IBM was selected as the first defendant against which the litigation would proceed. In that litigation, several claim terms, including "multimedia" and "multi-level . . . security," were construed, IBM was granted summary judgment on a finding that it did not infringe any asserted patent, and the Federal Circuit affirmed. TecSec, Inc. v. IBM et al., 466 Fed. App'x 882 (Fed. Cir. Jan. 18, 2012). TecSec and IBM subsequently settled.

Rather than continue to prosecute its claims against the remaining defendants, TecSec stipulated that it could not prove infringement of the DCOM Patents against any defendant under the construction of "multimedia," and could not prove infringement of the DCOM Patents against Paypal, Inc. ("PayPal") under the construction of "multi-level . . . security." Accordingly, judgment was entered against TecSec, and TecSec appealed. At TecSec's request, the litigation was stayed pending the outcome of the appeal. The Federal Circuit affirmed the construction of "multi-level . . . security," but reversed the construction of "multimedia." TecSec, Inc. v. IBM et al., 731 F.3d 1336, 1345-46 (Fed. Cir. 2013). Defendants filed a petition for a writ of certiorari, and the stay of this litigation was continued while the petition was pending. After the petition was denied on June 2, 2014, and the mandate was returned, TecSec agreed to proceed against Adobe under only the DCOM Patents on an accelerated schedule.

III. DISCUSSION

TecSec alleges that Acrobat infringes claims 1, 4, 8, and 9 of the '702 Patent, claim 1 of the '452 Patent, claim 1 of the '755 Patent, and claims 1, 3, 14, and 15 of the '781 Patent (collectively, the "asserted claims"). Adobe moves for summary judgment of non-infringement, centering its arguments on the "multi-level . . . security" and "labelling" limitations in the claims, as well as presenting arguments regarding the object-oriented key manager. Motion for Summary Judgment Att. 1 ("MSJ Br."). Although TecSec asserts in its pleading that "Acrobat [i]nfringes [t]he DCOM Patents," MSJ Opp'n at 8, TecSec has not moved for summary judgment of infringement, despite receiving the opportunity to do so. See November 10, 2014 Order [Dkt. No. 724].

The asserted claims are set forth below.[10] Claim 4 of the '702 Patent and claim 3 of the '781 Patent depend on unasserted claims; because dependent claims include the limitations of the claims on which they depend, the limitations of the unasserted claims must also be addressed.

'702 Patent	
Claim 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;

[10] The asserted claims are exceptionally broad, and provide little detail regarding how a person of ordinary skill in the art would implement the claimed invention. The lack of detail renders the claims remarkably abstract – "[i]ndeed, it is difficult to conceive of broader terms with which the idea of" multi-level encryption "could be described." Amdocs (Israel) Ltd. v. Openet Telecom, Inc., No. 1:10cv910, 2014 WL 5430956, at *6 (E.D. Va. Oct. 24, 2014) (invalidating claims to correlating two network records under 35 U.S.C. 101 for being drawn to an unpatentable abstract idea). As validity is not presently before the Court, however, there is no cause to consider validity issues at this time.

	G) reading the object label; H) determining access authorization based on the object label; and I) decrypting the object if access authorization is granted."
Claim 4	"[The method of claim 1, further comprising the step of embedding the encrypted object in a second object after labelling the encrypted object, and], 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."
Claim 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 module; 4) 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 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."
Claim 9	"The system of claim 8, wherein the digital logic means further comprises means for embedding a first object within a second object."

'452 Patent	
Claim 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."

'755 Patent	
Claim 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 according to the encryption algorithm; 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."

'781 Patent	
Claim 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."
Claim 3	"[The method of claim 1, further comprising embedding the encrypted first object in a second object after labeling the encrypted first object, and] further comprising: A) selecting a second label for the second object; B) selecting an encryption algorithm; C) encrypting the second object; D) labelling the encrypted second object with a second object label."

Claim 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 of the 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."
Claim 15	"The system of claim 14, wherein the data processor further comprises embedding logic for embedding a first object within a second object."

A. Standard of Review

Summary judgment is appropriate where the record demonstrates "that there is no genuine issue as to any material fact and that the moving party is entitled to judgment as a matter of law." Fed. R. Civ. P. 56(c). A genuine issue of material fact exists "if the evidence is such that a reasonable jury could return a verdict for the nonmoving party." Anderson v. Liberty Lobby, Inc., 477 U.S. 242, 247-48 (1986).

The Court must view the record in the light most favorable to the nonmoving party, see Bryant v. Bell Atl. Md., Inc., 288 F.3d 124, 132 (4th Cir. 2002); however, the "mere existence of a scintilla of evidence in support of the [nonmovant's] position will be insufficient; there must be evidence on which the jury could reasonably find for the [nonmovant]." Anderson, 477 U.S. at

252; see also Othentec Ltd. v. Phelan, 526 F.3d 135, 140 (4th Cir. 2008); TecSec, Inc. v. Int'l Bus. Machines Corp., 763 F. Supp. 2d 800, 804-05 (E.D. Va. 2011) aff'd, 466 F. App'x 882 (Fed. Cir. 2012). When relying on expert testimony, "[a] party does not manufacture more than a merely colorable dispute by submitting an expert declaration that something is black when the moving party's expert says it is white; there must be some foundation or basis for the opinion." Invitrogen Corp. v. Clontech Labs., 429 F.3d 1052, 1080 (Fed. Cir. 2005).

B. Claim Construction

"The first step of the infringement analysis is claim construction," Nazomi Comm'ns, Inc. v. Nokia Corp., 739 F.3d 1339, 1343 (Fed. Cir. 2014), an issue of law for determination by the Court, Markman v. Westview Instruments, Inc., 517 U.S. 370, 387 (1996), which "begins with the language of the claim." Power Integrations, Inc. v. Fairchild Semiconductor Intern., Inc., 711 F.3d 1348, 1360 (Fed. Cir. 2013). The general rule is that the words of the claim are "given their ordinary and customary meaning as understood by a person of ordinary skill in the art when read in the context of the specification and prosecution history." Thorner v. Sony Computer Entm't, 669 F.3d 1362, 1365 (Fed. Cir. 2012). The "primary focus in determining the ordinary and customary meaning of a claim limitation is to consider the intrinsic evidence of record, viz., the patent itself, including the claims, the specification and, if in evidence, the prosecution history, from the perspective of one of ordinary skill in the art." Atofina v. Great Lakes Chem. Corp., 441 F.3d 991, 996 (Fed. Cir. 2006) (citing Phillips v. AWH Corp., 415 F.3d 1303, 1312-17 (Fed. Cir. 2005) (en banc)).

One of "only two exceptions" to the general rule that words in a claim should be given their plain meaning to one of ordinary skill is "when a patentee sets out a definition and acts as his own lexicographer." Thorner, 669 F.3d at 1365. "To act as its own lexicographer, a patentee

must 'clearly set forth a definition of the disputed claim term' other than its plain and ordinary meaning." Id. (quoting CCS Fitness, Inc. v. Brunswick Corp., 288 F.3d 1359, 1366 (Fed. Cir. 2002)). For example, the Federal Circuit has found that a patentee acted as his own lexicographer to define the term "multiple embossed" when "the specification stated: 'Multiple embossed' means two or more embossing patterns are superimposed on the web to create a complex pattern of differing depths after embossing." Id. (quoting 3M Innovative Props. Co. v. Avery Dennison Corp., 350 F.3d 1365, 1369 (Fed. Cir. 2003)). Although "claims must be read in view of the specification, of which they are a part," Philips, 415 F.3d at 1315 (internal quotation marks omitted), limitations from the specification may not be read into the claims. Id. at 1320.

The second exception to the general rule is "when a patentee disavows the full scope of a claim term either in the specification or during prosecution," id. at 1365; however, "[a]bsent a clear disavowal in the specification or the prosecution history, the patentee is entitled to the full scope of its claim language." Id. at 1366 (quoting Home Diagnostics, Inc. v. LifeScan, Inc., 381 F.3d 1352, 1358 (Fed. Cir. 2004)). "Mere criticism of a particular embodiment encompassed in the plain meaning of a claim term is not sufficient to rise to the level of a clear disavowal." Id.

In construing a claim term, the Court begins with the language of the claims themselves. Philips, 415 F.3d at 1314. "[T]he context in which a term is used in the asserted claim can be highly instructive," and "claim terms are normally used consistently throughout the patent." Id. "Other claims of the patent in question, both asserted and unasserted, can . . . be valuable sources of enlightenment as to the meaning of a claim term." Id. Accordingly, "[d]ifferences among claims can . . . be a useful guide in understanding the meaning of particular claim terms. For example, the presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent claim." Id. at 1314-

15 (citations omitted). In particular, "'claim differentiation' refers to the presumption that an independent claim should not be construed as requiring a limitation added by a dependent claim," Curtiss-Wright Flow Control Corp. v. Velan, Inc., 438 F.3d 1374, 1380 (Fed. Cir. 2006), though the doctrine can be overridden if the inventor provides an express definition. TecSec, Inc. v. Intern. Bus. Mach. Corp., 731 F.3d 1336, 1345 (Fed. Cir. 2013).

When performing claim construction, the Federal Circuit has expressed a preference for intrinsic evidence over dictionaries or other extrinsic evidence. "Intrinsic evidence . . . is a more reliable guide to the meaning of a claim term than are extrinsic sources like technical dictionaries, treatises, and expert testimony." Chamberlain Grp., Inc. v. Lear Corp., 516 F.3d 1331, 1335 (Fed. Cir. 2008). "[T]he specification is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term." Power Integrations, 711 F.3d at 1361 (quoting Vitronics Corp. v. Conceptronic, Inc., 90 F.3d 1576, 1582 (Fed. Cir. 1996)). "Although definitions based on dictionaries, treatises, industry practice, and the like are often important aids in interpreting claims, they may not be 'used to contradict claim meaning that is unambiguous in light of the intrinsic evidence.'" ArcelorMittal France v. AK Steel Corp., 700 F.3d 1314, 1320 (Fed. Cir. 2012) (quoting Phillips, 415 F.3d at 1312-17).

"The district court . . . has an obligation . . . to carefully consider, and independently decide, the issues in the case." LSI Indus., Inc. v. ImagePoint, Inc., 279 F. App'x 964, 969 (Fed. Cir. 2008). This includes an "independent obligation to determine the meaning of the claims, notwithstanding the views asserted by the adversary parties." Bancorp Servs., LLC v. Sun Life Assur. Co., 687 F.3d 1266, 1274 (Fed. Cir. 2012) (quoting Exxon Chem. Patents, Inc. v. Lubrizol Corp., 64 F.3d 1553, 1555 (Fed. Cir. 1995)). "When the parties present a fundamental dispute

regarding the scope of a claim term, it is the court's duty to resolve it." O2 Micro Intern. Ltd. v. Beyond Innovation Tech. Co., Ltd., 521 F.3d 1351, 1362 (Fed. Cir. 2008).

The Court previously construed "object" to mean "any distinct, separate entity" and "multi-level . . . security" to mean "security achieved when encrypted objects are nested within other objects which are also encrypted, possibly within other objects, resulting in multiple layers of encryption." Accordingly, those constructions are not in dispute. The parties dispute five terms. By proposing single constructions across all DCOM Patents, TecSec and Adobe agree that each claim term should have a single definition.

1. "selecting a label for the object"

The parties did not propose a construction for the clause "selecting a label for the object." At oral argument, the Court advised the parties that it viewed the claim term as potentially dispositive, depending on whether "selecting a label" includes "creating a label" or "selecting the components that go into a label." Transcript of Jan. 23, 2015 Hearing [Dkt. No. 769] ("MSJ Oral Argument Transcript") at 3-4, 7. Accordingly, the parties were given the opportunity to brief the issue. Id. at 3, 5, 7, 20. The parties advised that they did not believe that briefing was necessary, id. at 7, 20, but clearly dispute whether "selecting a label" includes "creating a label" or "selecting the components that go into a label." See id. at 3-8. As "the parties present a fundamental dispute regarding the scope of a claim term, it is the court's duty to resolve it." O2 Micro Intern. Ltd., 521 F.3d at 1362.

Claim 1, standing alone, does not provide much insight into a more precise definition of "selecting a label." The claim uses the term "selecting" three times ("selecting an object to encrypt," "selecting a label for the object," and "selecting an encryption algorithm"), but the claim does not provide any detail regarding what it means to "select." Of significance to the

infringement analysis, nothing in the claim clarifies whether the object, label, or algorithm must exist before they are selected or whether selecting includes actually creating the object, label, or algorithm.

Claim 1's dependent claims, however, provide some insight. Claim 2 adds to claim 1 "creating an object in an application prior to accessing the object-oriented key manager." That language clarifies that before an object can be selected, it must first be created.[11] Under the doctrine of claim differentiation, the addition of the "creating" step in claim 2 provides strong evidence that the "selecting" step in claim 1 does not include "creating." To read "selecting an object" to include "creating an object" would be to render the added step in claim 2 superfluous. Moreover, "claim terms are normally used consistently throughout the patent," Philips, 415 F.3d at 1314, therefore, the term "selecting" in "selecting an object" should have the same meaning when used in the claim term "selecting a label." Because there is no evidence that the inventor intended "selecting" to mean something different when dealing with an object rather than when dealing with a label or encryption algorithm, the language of the claims provides strong evidence that "selecting a label" does not include "creating a label."

The specification also forms part of the intrinsic evidence used in claim construction, Philips, 415 F.3d at 1315, and supports this analysis. Although the specification does not further refine "selecting a label," in each example where a user "[s]elects object(s) to [e]ncrypt," the user first "creates an object." '702 Patent col. 8 lines 1-4, col. 9 lines 11-14, and col. 10 lines 17-19. In addition, before a user "[s]elects object(s) to [p]review," the user "creates an encrypted

[11] Claim 13 similarly restricts claim 12.

object." Col. 10 line 18 – col. 11 line 2. The written descriptions of the other patents in the DCOM family express the same difference between "selecting" and "creating."[12]

Although limitations from the specification may not be read into the claims, the description of "labelling" in the specification, consistent with its use in the claims, indicates that the label is "selected" from, for example, a list of users or terminals which are allowed to see the message. Id. col. 2 lines 31-57. As described in the specification, "selecting a label" requires a user to actively choose a pre-existing label. In order for "the sender [to] 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," id., the user must have pre-existing terminals from which to choose; and have the capability to actively choose which terminals can access the encrypted object.

Extrinsic evidence may also be relevant to claim construction. Philips, 415 F.3d at 1317-18. To the extent that it is necessary, extrinsic evidence provides further support for this construction of the meaning of "selecting." A dictionary published during the same time as the initial application for the DCOM Patents defines "select" as "chosen from a number or group by fitness or preference." Webster's Third New International Dictionary 2058 (1993). From that standard definition, it is obvious that in order to be selected the selected item must first exist. Accordingly, the intrinsic and extrinsic evidence supports the conclusion that to "select[] a label for the object," the label must have been created before the selection is made.

---

[12] '755 Patent col. 7 line 66 – col 8 line 2, col 9 lines 5-9, 39-42, and 56-59, and col. 10 lines 6-9 and 41-43; '452 Patent col. 10 lines 59-62, col. 11 lines 26-29 and 63-66, col. 12 lines 46-49 and 65-68, and col. 13 lines 45-47; '781 Patent col. 7 lines 18-21 and 55-58; col. 8 lines 5-8, 27-30, and 64-67, and col. 9 lines 33-36.

During oral argument on the motion for summary judgment, counsel for TecSec argued that "selecting a label is selecting what that label will have on it," explaining that "[a] user . . . selects all the components that go into a label," and then the "software creates [the label] based on the user inputs." MSJ Oral Argument Transcript at 8. Under this argument, "selecting a label" means "selecting the components of the label" because "[t]hat is how software has to work. Software doesn't work in a vacuum. A user has to tell it what to do." Id. at 8. Adobe responded that merely selecting components of the label is not what the patents describe because the language used is "selecting the label." Id. at 13-14.

Adobe has the better of the argument. The language of claim 1 is clear that the "label" is selected; the claim does not state that the "components of a label" are selected. That distinction is supported by other claims which draw a distinction between the label and the components within the label. For example, claim 8 states that "label conditions" within an "object labelling subsystem" limit object access. If "label" meant the same thing as "label conditions," one would expect the same term to be used for each. Different terms generally have different meanings. Chicago Bd. Options Exch., Inc. v. Int'l. Secs. Exch., LLC, 677 F.3d 1361, 1369 (Fed. Cir. 2012).

The conclusion that "selecting a label" is different from "selecting the components of a label" is also supported by the written description. Immediately after stating that "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," the '702 Patent states that "[t]hus, a sending user can specify label conditions." Col. 2 lines 40-46. This language indicates that the label contains conditions on which access is regulated. Additionally, the '702 Patent lists components of the object when describing what is

encapsulated, rather than merely stating that the object is encapsulated, col. 8 lines 18-21, thereby supporting the conclusion that if "selecting a label" meant "selecting label components," the claim would say so.

TecSec's argument that a piece of software must create the label does not change this result. That a computer must create a label at some point in order for the user to select it does not mean that selecting a label and selecting the components used in its creation are the same thing. Neither does the Court's construction require a user to somehow reach into a computer and manipulate the electrical data to select a label; it merely requires that the label be chosen from labels which had previously been created. The selection can only occur after the creation, and selection must be of the label itself, rather than the components or conditions which are used to create the label. Accordingly, the phrase "selecting a label" is construed to mean "choosing a pre-existing label," and does not include selecting the components for the label.[13]

2. "object-oriented key manager"

The term "object-oriented key manager" only appears in the asserted method claims.[14] TecSec argues that the term means "software that controls access to the algorithm used to encrypt and decrypt objects." Adobe argues that it means "a software component that manages the encryption of an object, on an object-by-object basis, to achieve multi-level security, including the process of generating, distributing, changing, replacing, storing, checking on, and

[13] As will be explained later, the construction of "selecting" is dispositive on the issue of infringement. Because TecSec and Adobe have vigorously litigated the other claim terms, those terms are also ripe for construction. O2 Micro Intern. Ltd., 521 F.3d at 1362. Resolution of the meaning of those terms will also assist in early resolution of the claims that TecSec asserts against the remaining defendants.

[14] Specifically, those are claims 1 and 4 of the '702 Patent, claim 1 of the '755 Patent, claim 1 of the '452 Patent, and claims 1 and 3 of the '781 Patent.

destroying cryptographic keys." Both Adobe and TecSec draw their definitions from a section of the written description introduced during the prosecution of the '702 Patent after a rejection by the examiner. Declaration of Michael A. Oakes [Dkt. No. 737] Ex. 9 ("O.A. Response") at 11. The examiner had rejected claim 1 (then the only pending claim) under 35 U.S.C. 112 ¶ 2 (now 35 U.S.C. 112(b)) as being indefinite. Id. at 10. One of the terms that the examiner identified as being indefinite was "key manager." Id. In response, the applicant added the following language to the written description:

> 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.

Id. at 1-2. The applicant stated that the language was added "to define key manager and to explain accessing the key manager." Id. at 11.

Adobe draws its construction from that description by focusing on "the function of key management," though it adds that the object-oriented key manager "manages the encryption of an object, on an object-by-object basis, to achieve multi-level security." TecSec draws its construction from how the key manager functions "[u]nder normal operational circumstances."

TecSec's proposed construction will be rejected because it does not comport with the language of the claims and the written description. TecSec purports to be defining the term "object-oriented key manager," but its proposed construction indicates that the "key manager" controls access to an algorithm, not a key. From its plain language, a "key manager" must, at some level, manage a key or keys, and TecSec's definition does not provide for that. TecSec's proposed definition also omits a portion of the written description that it purports to quote.

Specifically, TecSec seeks to construe the term as "software that controls access to the algorithm used to encrypt and decrypt objects." The section from which TecSec draws its definition, however, states that the key manager "control[s] access to the algorithm used to encrypt and decrypt plain text objects." Col. 2 lines 1-2 (emphasis added).

The intrinsic evidence shows that Adobe's construction is more accurate. That construction rests on the unstated but reasonable assumption that a "key manager" is a "thing that performs key management." In defining "key manager," then, it is appropriate to draw from the definition of "key management" provided by the '702 Patent: that "[t]he function of key management is to perform the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys." Col. 1 lines 63-66. The definition in the written description does not provide support for Adobe's inclusion of the requirement that the key manager "manages the encryption of an object, on an object-by-object basis, to achieve multi-level security." Accordingly, that clause will not be included.

TecSec argues that the claims of the '702 Patent contradict Adobe's construction because the claims do not reflect all of the functions in Adobe's definition. That argument fails because an inventor may disclose many different embodiments of an invention in the written description, and may describe many different capabilities of the same embodiment. Although claim 1 of the '702 Patent reflects only "the cryptographic functions of encrypting an object and decrypting an object, not distributing or destroying keys," MSJ Opp'n at 19, that statement does not mean that including the capability to distribute or destroy keys in the claim construction is incorrect; it only means that the inventor of the '702 Patent chose, for whatever reason, to focus the claims of that particular patent on a subset of functions of which the object-oriented key manager was capable.

For these reasons, Adobe's proposed definition is modified and the term "object oriented key manager" is construed to mean "a software component that is capable of performing the process of generating, distributing, changing, replacing, storing, checking on, and destroying cryptographic keys." Because the intrinsic evidence in the patent is sufficiently clear and unambiguous, reference to extrinsic evidence is unnecessary.

3. "label"

The term "label" appears in all of the asserted claims. TecSec argues that the term means "an identifier associated with an object," while Adobe argues that it means "a series of letters or numbers, separate from but associated with the sending of an object, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated object and which provides a level of protection in addition to cryptographic protection." Adobe adopts its proposed construction from the '702 Patent, which states that:

> 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.

Col. 2, lines 34-40.[15] Because the claims themselves do not include a "message," Adobe replaces "message" with "object," which does appear in the claims, and adds the phrase "and which provides a level of protection in addition to cryptographic protection" because, Adobe argues, "[t]he specifications of the DCOM [P]atents explain that the labels are attached by software to an encrypted object to add a layer of security." MSJ Br. at 21. To support this construction, Adobe again cites the written description of the '702 Patent which provides that "[b]y being able to

[15] This definition also appears in the '755 Patent and the '781 Patent. '755 Patent col. 2 lines 32-38; '781 Patent col. 2 lines 22-28.

compartment every object by label attributes and algorithm attributes, multi-level multimedia security is achieved." Col. 4 lines 22-25.

In contrast, TecSec argues that its construction of "label" as "an identifier associated with an object" is appropriate because its construction comports with the "ordinary and customary meaning" of the term "label." TecSec also argues that the "specification does not give ["label"] a special meaning," and that extrinsic evidence, for example a technical dictionary, supports this "plain meaning." MSJ Opp'n at 21. TecSec attempts to explain the language quoted by Adobe as simply part of the disclosure of another patent, U.S. Patent No. 5,369,707 (the "'707 Patent"), which was included in the '702 Patent as a description of the prior art which the '702 Patent sought to replace. The paragraph immediately following the section that Adobe cites for its definition states that "[a] system such as that described above is disclosed in [the patent application which became the '707 Patent], filed Jan. 27, 1993, the specification of which is incorporated by reference herein." '702 Patent Col. 2 lines 58-61.

The plain text of the '702 Patent shows that TecSec's argument is incorrect and that Adobe has correctly relied on that section. The written description never uses the word "invention" to apply to anything other than the system described by the '702 Patent. At no point does the written description refer to the system described by the application which would become the '707 Patent as an "invention;" by contrast, it refers to the system of the '702 Patent as "the invention" at least eighteen times.[16] Because the only thing that the written description

[16] See, e.g., col. 1 line 5 ("field of the invention"); col. 1 line 13 ("background of the invention"); col. 3 line 11 ("summary of the invention"); col. 3 lines 12, 18, 21, 25, 29, and 42, col. 4 lines 14, 31, 38, 39-40, 47, 52, col. 5 line 18, col. 11 line 61 ("the present invention"); col. 4 lines 56-57 ("detailed description of the invention") (emphasis added).

refers to as an "invention" is the system described by the '702 Patent, "this invention" must mean "the invention described by the '702 Patent."

The language quoted by TecSec, referring to the '707 Patent, does not contradict this result. It is possible, and even common, for inventors to seek patents for different aspects of a single product or technology (for example, TecSec received the four patents at issue by claiming variations of the DCOM system). That a specific definition for "label" was used in the '707 Patent does not preclude an inventor from using the same definition in the '702 Patent, or a multitude of other patents, as well. Inventors often use terms of art, which have the same definition across an entire field. It is similarly unsurprising that different inventors employed by the same company would use the same definition when applying for a patent. Although TecSec attempts to make much from the fact that the '702 and '707 Patents have different inventors, the cover pages of the two patents reveal that both patent applications were prepared and prosecuted by the same two individuals.

During oral argument, counsel for TecSec disparaged Adobe's proposed construction, describing it as drawn from the background of the invention which only functioned to describe the prior art. MSJ Oral Argument Transcript at 16-17. Nothing prevents an inventor from defining a relevant claim term in the background section of the written description. TecSec appears to recognize this fact, as it draws its proposed construction of "object-oriented key manager" from the background section of the '702 Patent. See supra.

TecSec's expert Dr. Jones opined that "the inventor of the '702 Patent purposefully distinguished his invention from the ['707 Patent] . . . and the section relied upon by Adobe is relevant only to the '707 [P]atent." Jones Dec. ¶ 66. That opinion does not change the Court's analysis because Dr. Jones does not explain the basis for his opinion or why the section relied

upon by Adobe is only relevant to the '707 Patent, and fails to provide any evidence to support his opinion. "[T]here must be some foundation or basis" for an expert opinion. Invitrogen Corp., 429 F.3d at 1080. As Dr. Jones' opinion is not supported by any evidence, other than his bald assertion, the Court accords that opinion little weight in its analysis.

Finally, TecSec argues that the written description of the '452 Patent, which describes "label" in more detail[17] (while still bearing a definition similar to that provided in the '702 Patent) shows that the DCOM Patents use a different, more flexible, definition of "label" than the definition in the '707 Patent. MSJ Opp'n at 22. TecSec's argument is flawed for several reasons. First, the section of the '452 Patent that TecSec cites is entirely consistent with the definition of "label" provided in the '702 Patent, '755 Patent, and '781 Patent – which makes sense, considering that the '452 Patent adopts the definition from the '702 Patent almost verbatim.[18] '452 Patent col. 3 lines 1-7. Moreover, the '452 Patent does not have the "system such as that described above" language (referring to the '707 Patent) on which TecSec hangs its argument. Lastly, any arguable modification to the definition caused by the '452 Patent could not affect the definition in the '702 Patent because the application which became the '452 Patent was filed after the '702 Patent already issued. The evidence, across all of the DCOM Patents, indicates that the inventor intended "label" to have a particular definition as expressed in almost identical terms in each patent. Moreover during the more than five years between the filing of the application that became the first DCOM Patent to the issuance of the last DCOM Patent, TecSec

---

[17] The '452 Patent states that "[a] file 'label,' relative to the present invention mean 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." Col. 3 lines 1-7.

[18] See supra n.17.

made no effort to modify the quoted language in any meaningful way. This course of conduct is powerful evidence that the inventor intended a special definition for the term "label," and not the ordinary meaning that TecSec now seeks.

For the foregoing reasons, by using the language "[a] file label for the purposes of this invention means" the patentee indicated the intent to provide a special definition for the word "label," rather than the ordinary definition. See Thorner, 669 F.3d at 1366 (quoting with approval a pre-Philips decision finding that the inventor acted as his own lexicographer).

TecSec argues that Adobe's definition is inappropriate because Adobe replaces the word "message," which appears in the written description, with "object," which appears in the claims. MSJ Opp'n at 22. Contrary to TecSec's arguments, the change of "message" to "object" does not defeat Adobe's proposed construction. For the reasons explained above, it is clear that the inventor intended a special definition for the term "label" when drafting the '702 Patent. "The terms used in patent claims are not construed in the abstract, but in the context in which the term was presented and used by the patentee, as it would have been understood by a person of ordinary skill in the field of the invention on reading the patent documents." Fenner Investments, Ltd. v. Cellco P'ship, 778 F.3d 1320, 1322-23 (Fed. Cir. 2015). The relevant section of the written description defines a "label for purposes of this invention" by reference to a "message." '702 Patent col. 2 lines 31-40. The claims define the invention, however, by reference to an "object." In light of the inventor's intent to provide a particular definition for "label" while claiming an invention involving "objects," "the context in which the term was presented and used by the patentee," Fenner, 778 F.3d at 1322, indicates that substitution of "object" for "message" is appropriate in the construction of "label."

TecSec also argues that the phrase "a series of letters or numbers, which may or may not be encrypted, separate from but associated with the sending of an object," from Adobe's definition and adapted from the definition in the '702 Patent, is "nonsensical" in the context of the claims because "there is no requirement in the claims or the specification of the DCOM Patents that objects must be 'sent.'" MSJ Opp'n at 22. Importantly, however, the definition does not require actual sending of the object, but only that the label is "associated" with the sending of the object. "Determining access authorization based on the object label" and that the method is "for providing multi-level multimedia security in a data network," both limitations present in the claims, indicate that the label is "associated" with the sending of the object, as provided in the definition.

TecSec also attacks the last clause of Adobe's construction of "label," which states that the label "provides a level of protection in addition to cryptographic protection." That clause does not appear in the definition in column 2 of the '702 Patent, and was added by Adobe because "[t]he specifications of the DCOM patents explain that the labels are attached by software to an encrypted object to add a layer of security." MSJ Br. at 22. TecSec argues that the added clause appears nowhere in the claims or specification of the '702 Patent, and there is nothing inherent in the word "label" that requires it to provide additional protection. MSJ Opp'n at 22. Instead, the label is merely the method through which a subsystem determines whether to allow access. Id. at 23.

TecSec has the better of the argument. Although the specification describes the label as being used to provide security in addition to encryption, that feature already appears in the claims. In claim 1, for example, access authorization is determined based on the object label, and the object is only decrypted if access authorization is granted. As the feature already appears in

the claims, there is no basis on which to add it to the construction for "label." Further, the added clause runs afoul of the same problem as TecSec's proposed construction – namely, that the inventor provided an explicit definition for the term "label" which does not include the added clause. The inventor may be his own lexicographer, and when he elects to do so his usage will be given effect.

Accordingly, the Court construes the term "label" to mean "a series of letters or numbers, separate from but associated with the sending of an object, which identifies the person, location, equipment, and/or organization which is permitted to receive the associated object."

4. "labelling"

The term "labelling" appears in all of the asserted claims. TecSec argues that the term means "attaching a label," and Adobe argues it means "attachment of a label to an object by software after encryption of that object." Adobe argues that its construction is supported by the written descriptions of the DCOM Patents because the patents provide that labelling produces protection "in addition to cryptographic protection." Adobe Systems Incorporated's Reply in Support of its Proposed Claim Constructions and Motion for Summary Judgment of Non-Infringement [Dkt. No. 753] ("Reply Br.") at 12. TecSec responds that the temporal limitation proposed by Adobe introduces duplicative limitations into the claim terms. MSJ Opp'n at 23-24. For example, claim 1 requires "labelling the encrypted object" which means that labelling necessarily occurs after encryption of the object. Although the particular implementation of claim 1 requires labelling after encryption, there is nothing in the specification which requires all labelling to occur after encryption. See col. 2 lines 31-57. Accordingly, TecSec's construction of the term "labelling" to mean "attaching a label" is the proper construction.

5. "access authorization"

The term "access authorization" appears in all of the asserted method claims.[19] TecSec argues the term means "authorization to access an object," while Adobe argues that it means "authorization to access an object with particular permissions based on the identity of the accessing entity rather than a password." Adobe argues that its construction is supported by other terms in the asserted claims because the "plain claim language dictates that 'access authorization' is based on the object label." MSJ Br. at 24 (emphasis removed). In essence, Adobe reads its construction of "label" into "access authorization." Adobe does not cite any section of the DCOM Patents to support its proposed construction. See id. at 24-25. TecSec responds that the claim term "is written in plain English that would be well-understood by a person of ordinary skill in the art when read in light of the" DCOM Patents and that the language of the claim requires access authorization based on a label, "any construction of the term 'label' is present in the claim language." MSJ Opp'n at 27.

Adobe's construction of "access authorization" is too restrictive. The plain language of the claims requires access authorization "based on the object label," and the Court has construed a "label" to identify the "person, location, equipment and/or organization which is permitted to receive the associated object." Adobe seeks to introduce into the construction of "access authorization" the requirement that the authorization is "based on the identity of the accessing entity." To the extent that "identity" is simply a reflection of the "person, location, equipment, and/or organization" language, imposing that requirement is duplicative of the "label" construction because the plain language of the claims requires that access authorization be

[19] Specifically, in claims 1 and 4 of the '702 Patent, claim 1 of the '755 Patent, claim 1 of the '452 Patent, and claims 1 and 3 of the '781 Patent.

granted "based on the label" – i.e., based on the person, location, equipment, and/or organization which is permitted to receive the associated object. Any definition of "access authorization" narrower than the definition of "label" would contradict the language of the claims, which specifically states that access authorization is based on the label. In either case, Adobe's construction of "access authorization" is not appropriate.

The '702 Patent does not give any special meaning to the term "access authorization." Accordingly, the term is construed in accordance with its ordinary meaning to be "authorization to access an object."

6. "display" / "displaying"

The term "displaying" appears in claim 1 of the '452 Patent, and the term "display" appears in claim 1 if the '755 Patent (through reference to a "display header"). Adobe argues that both terms should be construed to mean "making visually perceptible to a user." MSJ Br. at 25. Adobe argues that its construction is correct in light of the Summary of the Invention of the '452 Patent, which states that "[t]he label may appear as a header to authorized users. . . . 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." Id. (quoting '452 Patent col. 6 lines 7-13). TecSec responds that Adobe's construction is unnecessary because the terms are easily understandable and so should be given their plain and ordinary meaning, and that Adobe's "one size fits all" definition does not match the use of "display" (as part of the term "display header") in the '755 Patent. MSJ Opp'n at 28-29.

"Display" and "displaying" cannot be given the exact same construction because the two terms are different parts of speech. In the context of the claims at issue, "display" is an adjective modifying "header," and "displaying" is a verb. There is no evidence that the inventor intended

to act as his own lexicographer and provide a special definition for those terms. Accordingly, those terms will be given their plain and ordinary meaning. The plain and ordinary meaning is "the meaning [that the term] would have to persons in the field of the invention, when read and understood in light of the entire specification and prosecution history." Fenner, 778 F.3d at 1323 (citing Phillips, 415 F.3d at 1312-17). Adobe argues that its own definition is the "ordinary meaning of display." MSJ Br. at 25. TecSec does not propose its own definition, and does not dispute that Adobe's proposed construction is consistent with the ordinary meaning. See MSJ Opp'n at 25-26. Accordingly, "displaying" is construed to mean "making visually perceptible to a user." As "display" is used as an adjective in the context of the '755 Patent's "display header" term, "display" is construed together with "header" to mean "a header for making visually perceptible to a user."

C. Infringement

Remaining before the Court is Adobe's motion that it should be granted summary judgment on TecSec's claim that Acrobat directly infringes the DCOM Patents.[20] "To prove direct infringement, the plaintiff must establish by a preponderance of the evidence that one or more claims of the patent read on the accused [product] literally or under the doctrine of equivalents." Cross Med. Prods., Inc. v. Medtronic Sofamor Danek, Inc., 424 F.3d 1293, 1310 (Fed. Cir. 2005). To prove direct infringement of a method claim, "a patentee must prove that

[20] Adobe originally presented argument regarding divided and indirect infringement, see MSJ Br. at 26-29, but those issues were struck from consideration on TecSec's motion. November 14, 2014 Order [Dkt. No. 728]. At oral argument on the motion to strike, counsel for TecSec admitted that if Acrobat does not directly infringe the DCOM Patents, then Adobe does not infringe under any other theory of infringement. Transcript of November 14, 2014 Hearing [Dkt. No. 745] at 6-7. Adobe also argues that it does not infringe the asserted system claims because Adobe does not sell computer hardware, MSJ Br. at 27-28; that argument was previously dismissed on November 14, 2014. [Dkt. No. 728].

each and every step of the method or process was performed." Aristocrat Techs. Australia Pty. Ltd. v. Int'l Game Tech., 709 F.3d 1348, 1362 (Fed. Cir. 2013).

TecSec and Adobe present argument regarding whether Acrobat performs multiple claim terms; however, because failure to generate a genuine issue of material fact on a single claim term precludes a finding of infringement as a matter of law, the Court will only address a subset of those arguments.

1. Multi-level Encryption

Adobe claims that it should be granted summary judgment on TecSec's claim that Acrobat performs multi-level encryption because TecSec has not created a genuine issue of material fact regarding whether Acrobat performs that function. In response, TecSec has provided evidence that Acrobat can perform multi-level encryption by the following steps: (1) the user opens a first PDF document using a first session of Acrobat; (2) the user selects the encryption conditions for the first PDF document; (3) the user saves the first PDF document to create a first encrypted PDF document; (4) the user creates a second PDF document using a second session of Acrobat; (5) the user attaches the first encrypted PDF document to the second PDF document; (6) the user selects the encryption conditions for the second PDF document (which has the encrypted first PDF document attached); and (7) the user saves the second PDF document, thereby nesting the first encrypted PDF document within the second encrypted PDF document. MSJ Opp'n at 12-16.

Adobe claims that TecSec's argument is defeated by the fact that multiple sessions of Acrobat must be used for Acrobat to operate in the manner described by TecSec. Reply Br. at 7-8. Adobe does not cite to any precedent for its seemingly novel argument that a computer program only infringes a patent if it performs all of the elements of a claim in a single session of

that program. See id. TecSec has not accused a single session of Acrobat of infringing the DCOM Patents, but instead has accused the Acrobat program of infringement. Not only has TecSec presented evidence that Acrobat could be used to perform multi-level encryption, Kaufman 30(b)(6) Dep. at 70-72; Oakes Dec. Ex. 6-7, TecSec has also presented evidence that Adobe has instructed its customers that Acrobat could be used in that manner. Oakes Dec. Ex. 5. Specifically, Adobe posted to its website, and therefore instructed its users, that

> If you want to send multiple documents together in a PDF envelope, and you want those PDF attachments to retain their protections after the outer PDF envelope is opened – then you should protect the PDF files individually before they are attached. You then have an option of encrypting the outer PDF envelope hosting the attachments.

Id. This provides evidence that use of multiple sessions to provide multiple layers of encryption was an intended, rather than unforeseen, use of Acrobat. In sum, TecSec has provided sufficient evidence that Adobe has informed its customers that Acrobat could be used in a manner which provides multi-level encryption, and has instructed those customers how to do so. This evidence defeats Adobe's motion for summary judgment on this issue.

2. Whether the Encryption Dictionary is a Label

Adobe argues that Acrobat's encryption dictionary is not a "label" as defined by the DCOM Patents because the encryption dictionary "does not identify a person, a location, equipment, or an organization" as required by the DCOM Patents, and therefore summary judgment of non-infringement is appropriate. MSJ Br. at 23.[21] TecSec responds that the encryption dictionary is a label both when the PDF document is secured using a password and when the PDF document is secured using certificates. MSJ Opp'n at 24-25. Specifically, when

[21] Adobe also argues that TecSec disclaimed use of passwords during prosecution of the '702 Patent. MSJ Br. at 19. Even if TecSec did not disclaim the use of passwords, Acrobat still does not infringe as a matter of law and therefore Adobe's disclaimer argument need not, and will not, be addressed.

using password security, the password identifies the user and when using digital certificates, the encryption dictionary contains information which identifies the owner of the relevant digital certificates. Id. at 25.

Adobe has the better of the argument regarding password security. As explained previously, the encryption dictionary does not contain either the user or owner passwords. Kaufman 30(b)(6) Dep. at 17. Instead, the encryption dictionary contains an "O key" and "U key" used to test the information entered by the user to see if the user entered the correct password. Id. Even if the encryption dictionary included the actual user and owner passwords Adobe would still have the better of the argument because under Acrobat's system, the user and owner passwords are not linked to the identity of a particular user. Like a housekey, anyone's possession of the password grants access. The password is simply a talisman which grants access irrespective of the identity of the person using it. When using password security, the encryption dictionary does not identify any person, location, equipment, or organization and therefore the encryption dictionary does not constitute a label when password security is used. Accordingly, Adobe's prevails on the argument that the encryption dictionary is not a label when password security is used.

In contrast, TecSec has presented evidence that the encryption dictionary is a label when digital certificates are used because the encryption dictionary contains identifiers for the digital certificates related to the public keys used to encrypt the file key. Id. at 52-54. This finding is supported by Mr. Kaufman who, when asked if "the certificate IDs identify each of the individual recipients," responded "[y]es." Id. at 54. Therefore, TecSec has presented sufficient evidence that, when using certificate security, the encryption dictionary identifies the person who

is permitted to receive the associated object, thereby defeating Adobe's argument that the encryption dictionary is not a label when certificate security is used.

3. Selecting a Label

Every claim at issue contains either "selecting a label" or "selecting a first label" as an element.[22] Under the Court's construction of "selecting a label," the label must exist before it can be selected. Therefore, the element of "selecting a label" means that the label itself, not the conditions of the label, must be selected.

TecSec argues that the encryption dictionary generated by Acrobat and inserted into the trailer section of a PDF document as part of the encryption process is the "label" required by the claims. MSJ Opp'n at 24. As previously explained, the encryption dictionary qualifies as a label under the Court's construction only when certificate security is used.

Even if TecSec's view of the encryption dictionary generated by Acrobat as the label was accepted, TecSec has not raised a genuine dispute of material fact establishing that Acrobat performs the required "selecting" step because it is undisputed that a user of Acrobat does not select the encryption dictionary (TecSec's "label"). MSJ Opp'n at 24; Kaufman 30(b)(6) Dep. at 35, 54, 63; MSJ Oral Argument Transcript at 6 ("The user doesn't create the label."). Under the Court's construction of "selecting a label," the label itself must be selected, not created, which under TecSec's argument means that the encryption dictionary (TecSec's label) must be selected;

[22] All asserted method claims include the "selecting" language, and all system claims include selection through the "object labelling subsystem" and "object label identification subsystem," which restrict access based on "label conditions" present in a selected label. Because the claims use only "slightly different language to describe substantially the same invention," they will be treated the same. Ohio Willow Wood Co. v. Alps South, LLC, 735 F.3d 1333, 1342 (Fed Cir. 2013). Moreover, the parties have grouped all asserted claims together for the purposes of the motion for summary judgment, any argument that they should be treated differently is deemed waived. Voter Verified, Inc. v. Permier Election Solutions, Inc., 698 F.3d 1374, 1382 (Fed. Cir. 2012).

however, as explained above in Section I.D, the encryption dictionary is automatically generated by Acrobat when the user elects to "save" the document. Given the unrebutted evidence of how Acrobat generates the encryption dictionary, Acrobat does not meet the "selecting a label" element. Therefore, Adobe is entitled to summary judgment of non-infringement.

Even if TecSec were correct that selecting the conditions for a label is the same as selecting a label, it remains undisputed that the encryption dictionary does not exist when the conditions of the label are selected; instead, the encryption dictionary is not created until the user saves the file. Kaufman 30(b)(6) Dep. at 35, 54, 63. Under the Court's construction, the label must be created before it can be selected. In other words, in Acrobat the encryption dictionary (TecSec's label) does not exist when the user selects the various conditions for access to the object. It is not until the user saves the document that the encryption dictionary (TecSec's label) comes into existence. Accordingly, under this Court's construction of "selecting," Acrobat does not infringe, and Adobe's motion for summary judgment of non-infringement will be granted.[23]

4. Doctrine of Equivalents

TecSec argues that even if Acrobat does not literally infringe the claims of the DCOM Patents, Acrobat infringes under the doctrine of equivalents. MSJ Opp'n at 30. "Even without literal infringement of a certain claim limitation, a patentee may establish infringement under the doctrine of equivalents if an element of the accused device 'performs substantially the same

[23] Although not briefed by the parties, there are other aspects of the DCOM Patents that Acrobat does not perform. For example, each asserted claim requires some form of "labelling." At TecSec's urging, "labelling" has been construed to mean "attaching a label." Acrobat does not attach an encryption dictionary to an encrypted PDF document; instead, it inserts the encryption dictionary into the (pre-existing) trailer for that file. Jones Dec. ¶ 68. Indeed, far from being attached to the encrypted object, "[t]he encryption dictionary is part of the trailer portion of a PDF document." Id. Thus, what TecSec alleges is a label is not "attached to" the object, it is part of the object.

function in substantially the same way to obtain the same result as the claim limitation.'" EMD Millipore Corp. v. AllPure Techs., Inc., 768 F.3d 1196, 1202 (Fed. Cir. 2014) (quoting AquaTex Indus., Inc. v. Techniche Solutions, 419 F.3d 1374, 1382 (Fed. Cir. 2005)). To support a finding under the doctrine of equivalents, a patentee must "provide particularized testimony and linking argument as to the insubstantiality of the differences between the claimed invention and the accused device or process, or with respect to the function, way, result test . . . Such evidence must be presented on a limitation-by-limitation basis." W.L. Gore & Assocs., Inc. v. Medtronic, Inc., 874 F. Supp. 2d 526, 542 (E.D. Va. 2012). TecSec argues that it has provided the required analysis in its infringement contentions; however, TecSec's contentions are not supported by any evidence, such as an expert declaration, establishing that the differences between Acrobat and the DCOM Patents are insubstantial, or that Acrobat performs the same function as the DCOM Patents, in the same way, to reach the same result. Accordingly, TecSec's doctrine of equivalents argument fails as a matter of law.

5. Liability After October 20, 2013

Adobe argues that because the DCOM Patents expired on October 20, 2013, Adobe does not have any potential liability after that date. MSJ Br. at 28; Reply Br. at 20. TecSec does not contest that argument, see MSJ Opp'n, which is correct as a matter of law. Kearns v. Chrysler Corp., 32 F.3d 1541, 1550 (Fed. Cir. 1994) ("[T]here can be no infringement once the patent expires."). Accordingly, Adobe's Motion for Summary Judgment that it is not liable for any allegedly infringing acts occurring after October 20, 2013 will also be granted.

IV. CONCLUSION

For the reasons stated above, defendant Adobe Systems Incorporated's Motion for Entry of its Proposed Claim Constructions and for Summary Judgment of Non-Infringement. [Dkt. No.

710] will be GRANTED IN PART as to its claim constructions and GRANTED as to summary judgment of non-infringement by an appropriate Order to be issued with this Memorandum Opinion.

Entered this 7th day of May, 2015.

Alexandria, Virginia

/s/
Leonie M. Brinkema
United States District Judge

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

Alexandria Division

MAY - 7 2015
CLERK, U.S. DISTRICT COURT
ALEXANDRIA, VIRGINIA

TecSec, Incorporated, Plaintiff, v. International Business Machines Corporation, et al., Defendants.	)	Civil Action No. 1:10cv115-LMB-TCB

JUDGMENT

Pursuant to the order of this Court entered on May 7th, 2015 and in accordance with Federal Rules of Civil Procedure 58, JUDGMENT is hereby entered in favor of the defendant, Adobe Systems Incorporated and against the plaintiff, TecSec, Incorporated.

FERNANDO GALINDO, CLERK OF COURT

By: _____
Deputy Clerk

Dated: May 7th, 2015
Alexandria, Virginia

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.

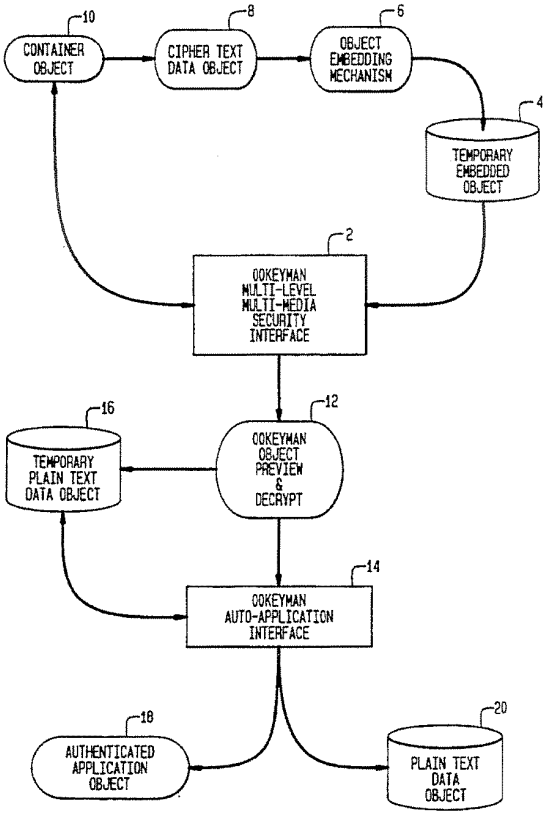
Primary Examiner—Bernarr E. Gregory

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



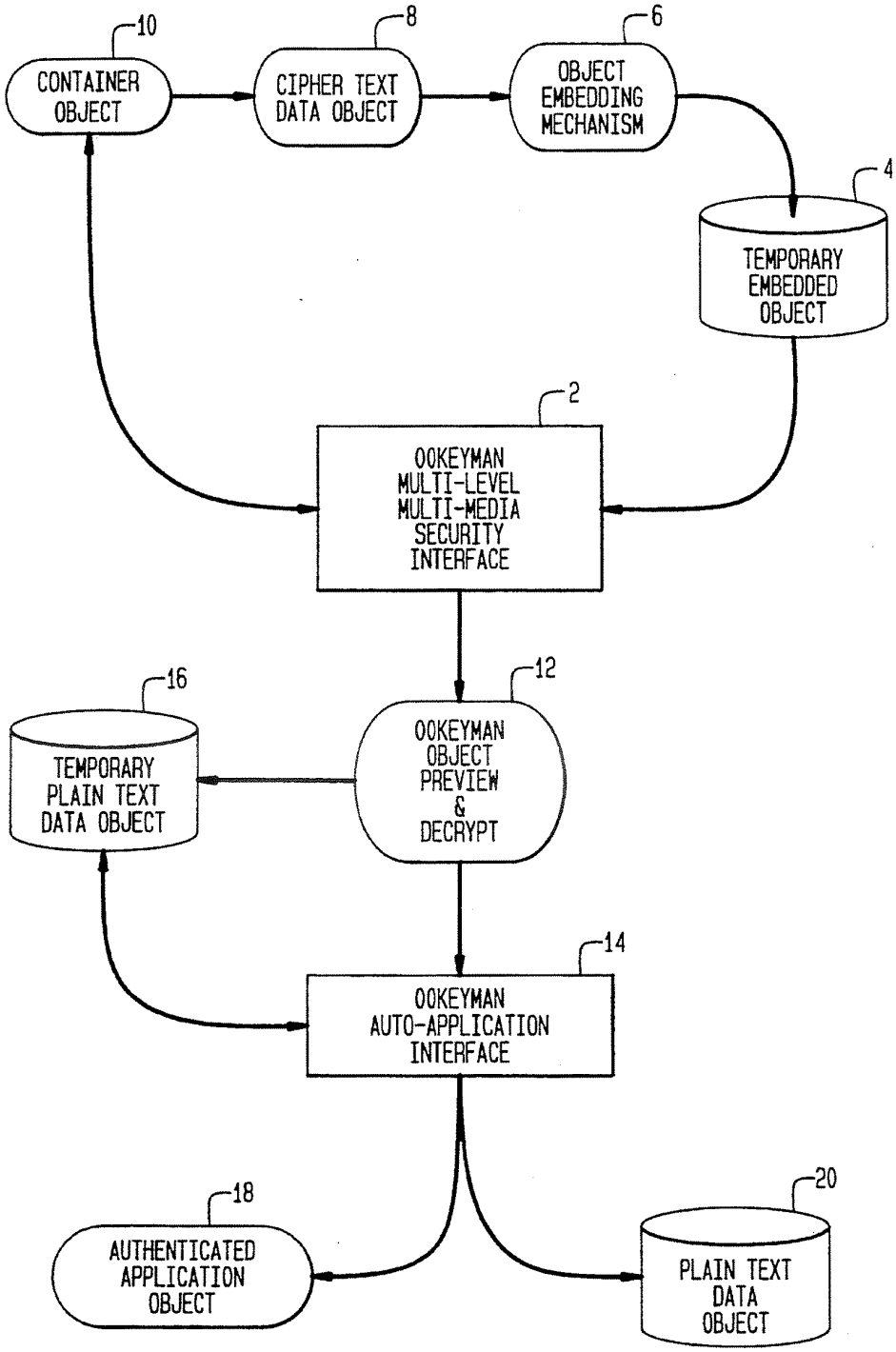

10
CONTAINER OBJECT
8
CIPHER TEXT DATA OBJECT
6
OBJECT EMBEDDING MECHANISM
4
TEMPORARY EMBEDDED OBJECT
2
OOKEYMAN MULTI-LEVEL MULTI-MEDIA SECURITY INTERFACE
16
TEMPORARY PLAIN TEXT DATA OBJECT
12
OOKEYMAN OBJECT PREVIEW & DECRYPT
14
OOKEYMAN AUTO-APPLICATION INTERFACE
18
AUTHENTICATED APPLICATION OBJECT
20
PLAIN TEXT DATA OBJECT

FIG. 1

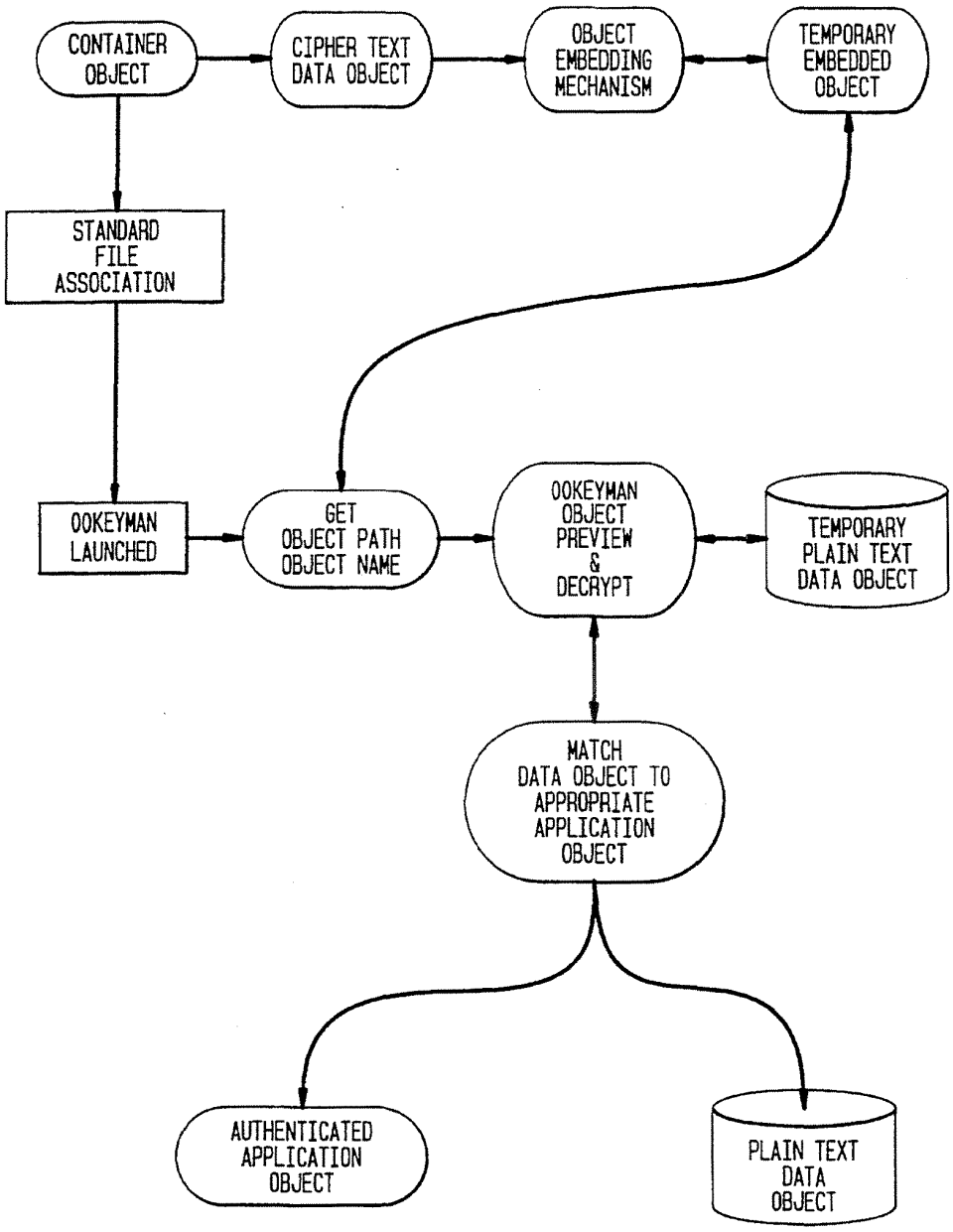

CONTAINER OBJECT
CIPHER TEXT DATA OBJECT
OBJECT EMBEDDING MECHANISM
TEMPORARY EMBEDDED OBJECT
STANDARD FILE ASSOCIATION
OOKEYMAN LAUNCHED
GET OBJECT PATH OBJECT NAME
OOKEYMAN OBJECT PREVIEW & DECRYPT
TEMPORARY PLAIN TEXT DATA OBJECT
MATCH DATA OBJECT TO APPROPRIATE APPLICATION OBJECT
AUTHENTICATED APPLICATION OBJECT
PLAIN TEXT DATA OBJECT

FIG. 2

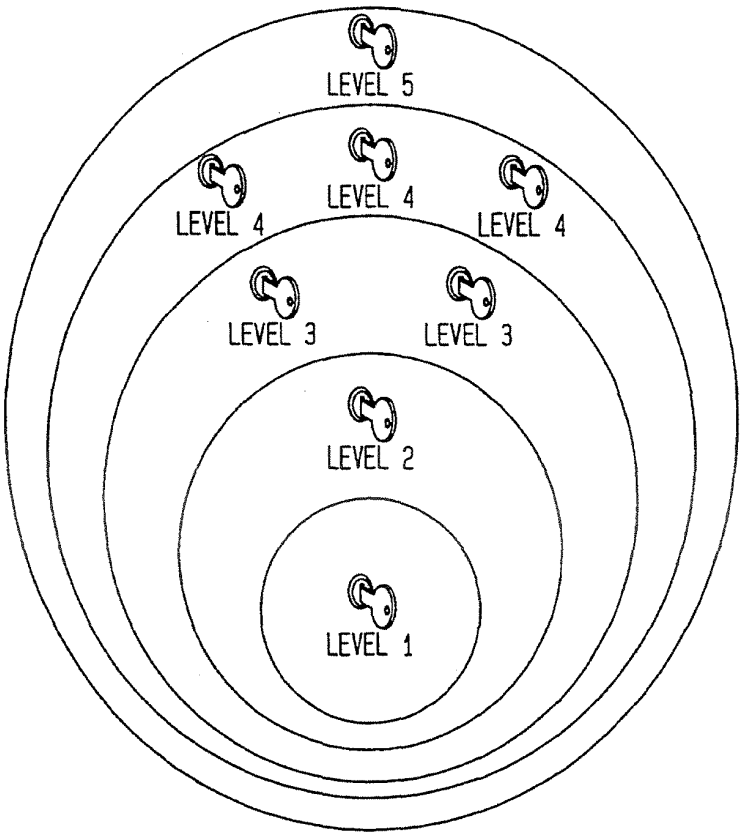


FIG. 3

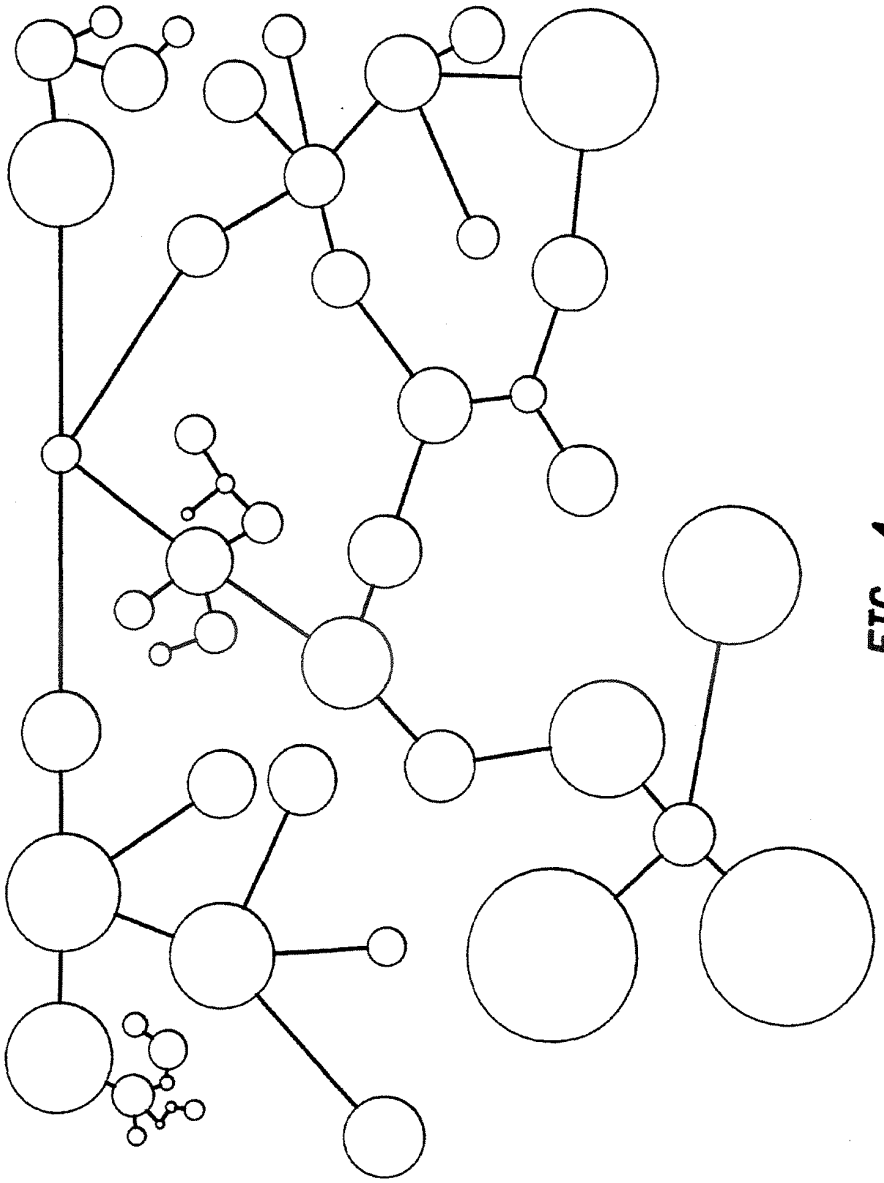


FIG. 4

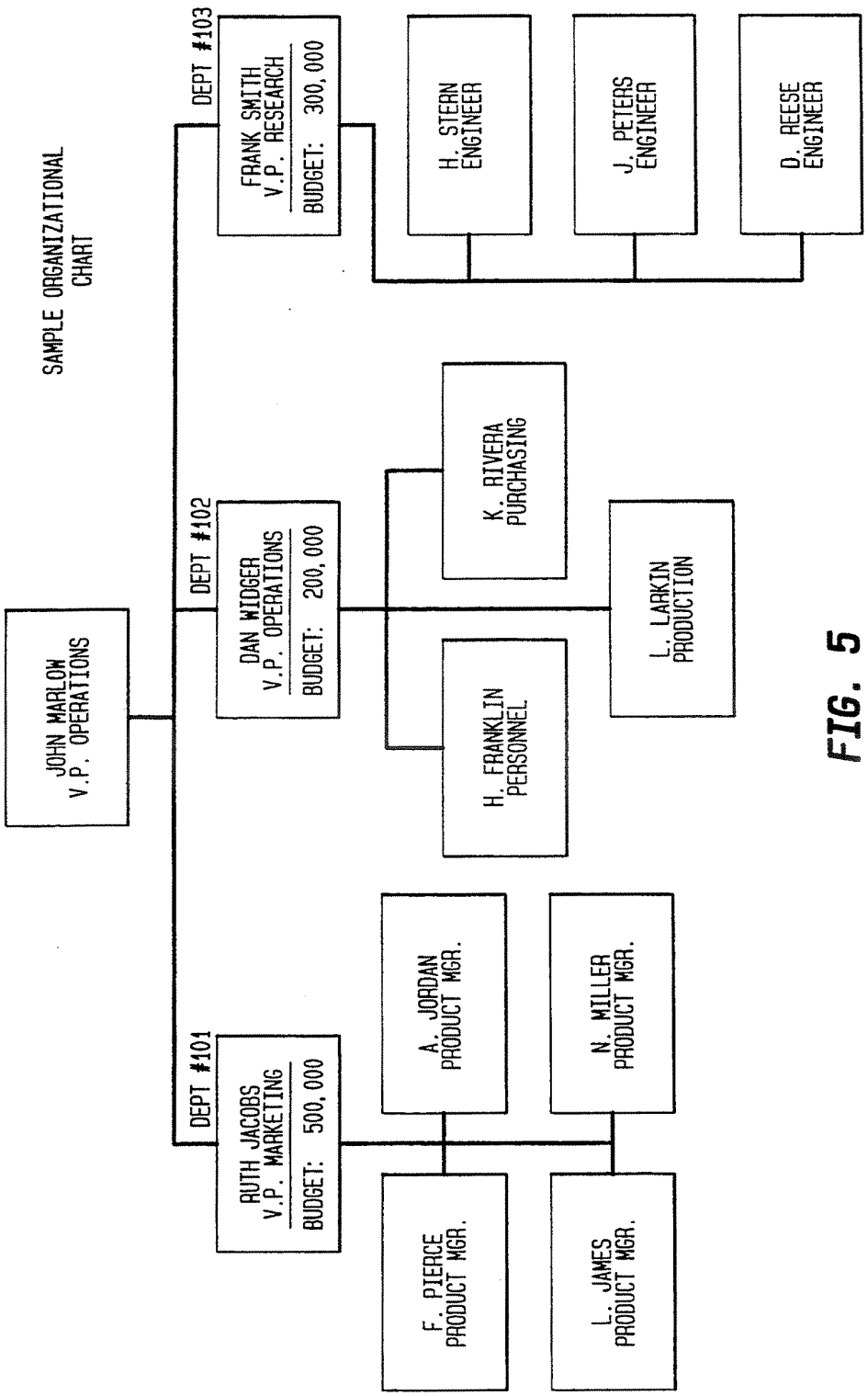

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW
V.P. OPERATIONS
DEPT #101
RUTH JACOBS
V.P. MARKETING
BUDGET: 500,000
DEPT #102
DAN WIDGER
V.P. OPERATIONS
BUDGET: 200,000
DEPT #103
FRANK SMITH
V.P. RESEARCH
BUDGET: 300,000
A. JORDAN
PRODUCT MGR.
F. PIERCE
PRODUCT MGR.
N. MILLER
PRODUCT MGR.
L. JAMES
PRODUCT MGR.
H. FRANKLIN
PERSONNEL
K. RIVERA
PURCHASING
L. LARKIN
PRODUCTION
H. STERN
ENGINEER
J. PETERS
ENGINEER
D. REESE
ENGINEER

FIG. 5

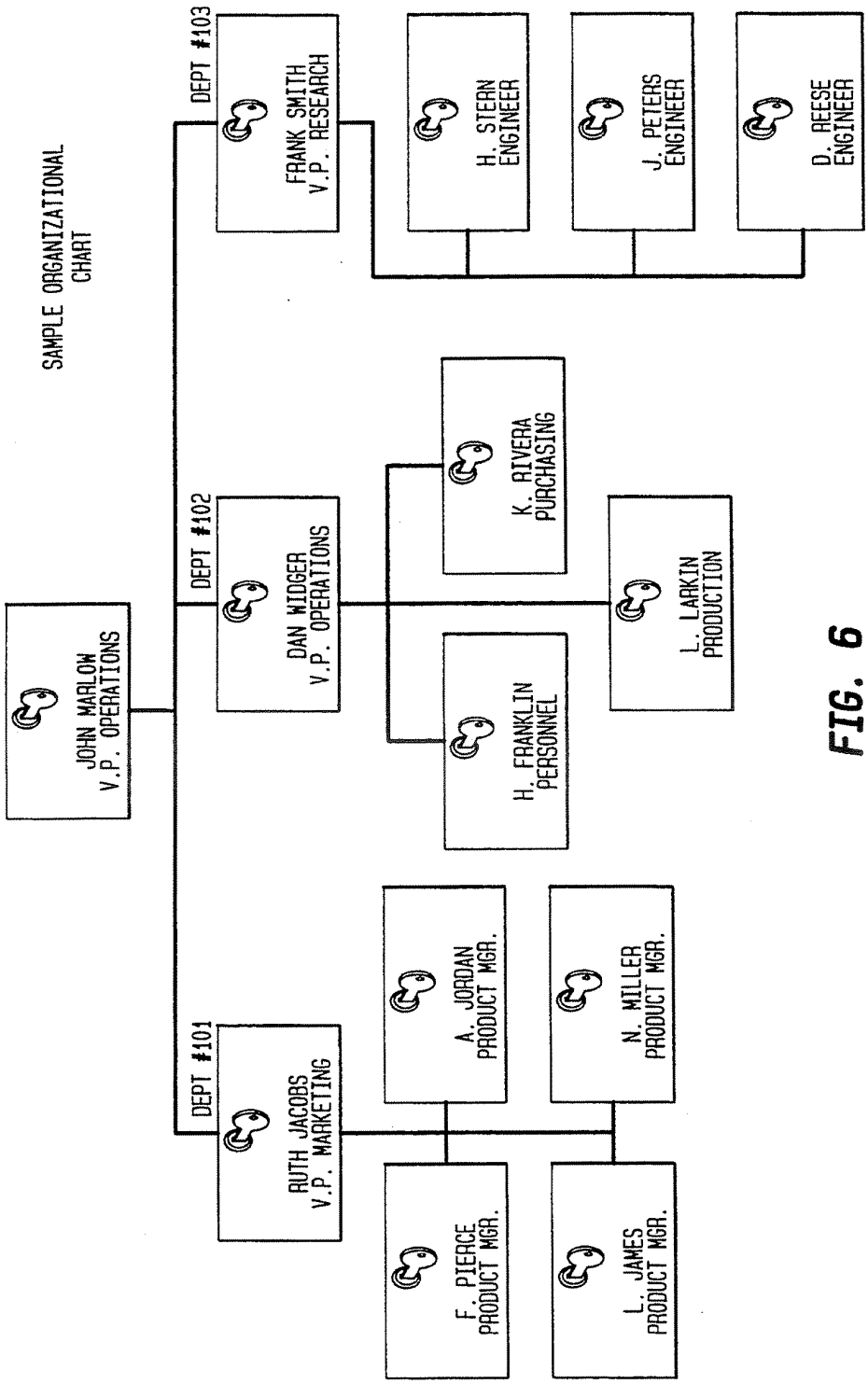

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW
V.P. OPERATIONS
DEPT #103
FRANK SMITH
V.P. RESEARCH
H. STERN
ENGINEER
J. PETERS
ENGINEER
D. REESE
ENGINEER
DEPT #102
DAN WIDGER
V.P. OPERATIONS
K. RIVERA
PURCHASING
L. LARKIN
PRODUCTION
H. FRANKLIN
PERSONNEL
DEPT #101
RUTH JACOBS
V.P. MARKETING
A. JORDAN
PRODUCT MGR.
N. MILLER
PRODUCT MGR.
F. PIERCE
PRODUCT MGR.
L. JAMES
PRODUCT MGR.

FIG. 6

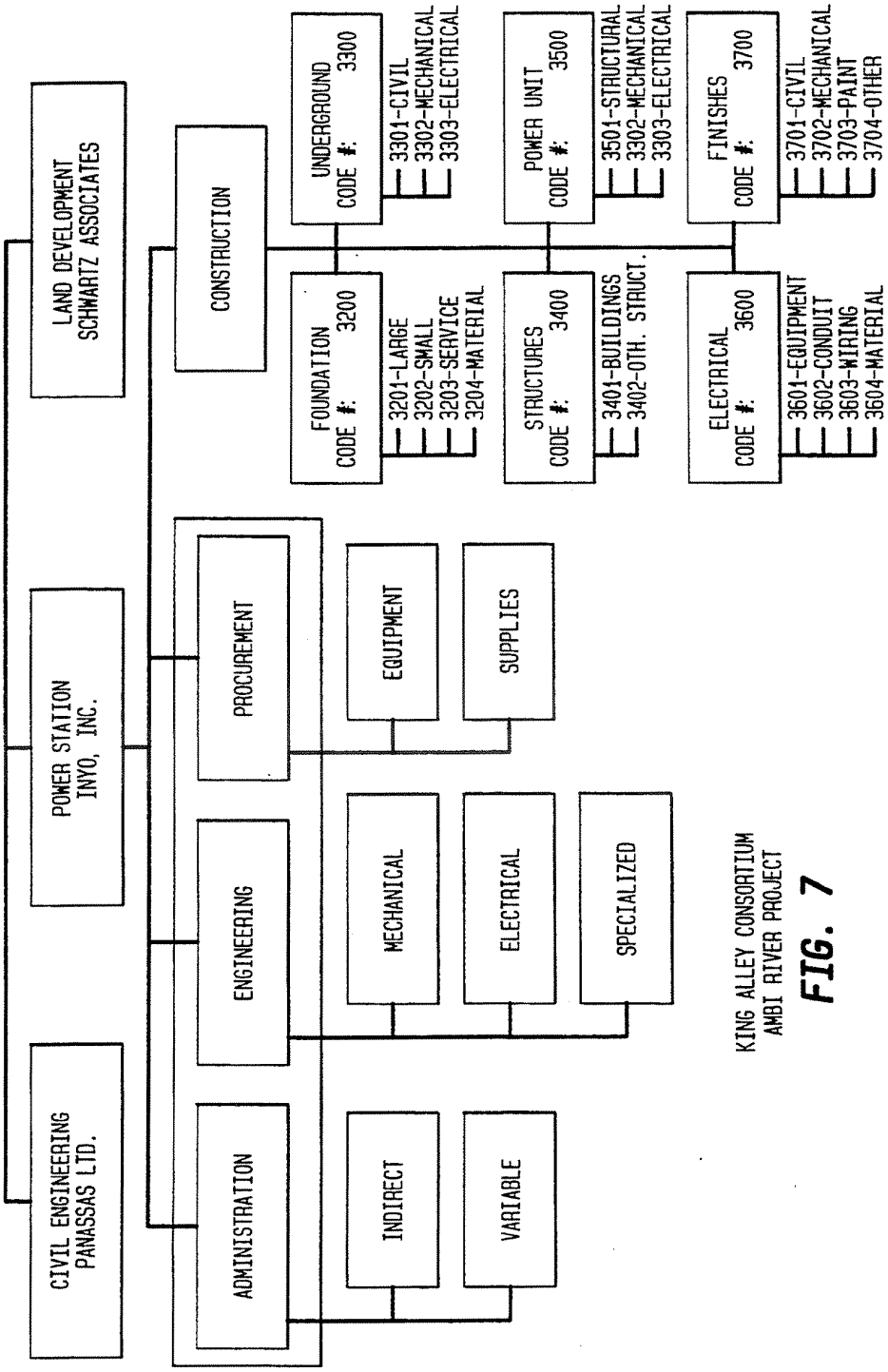
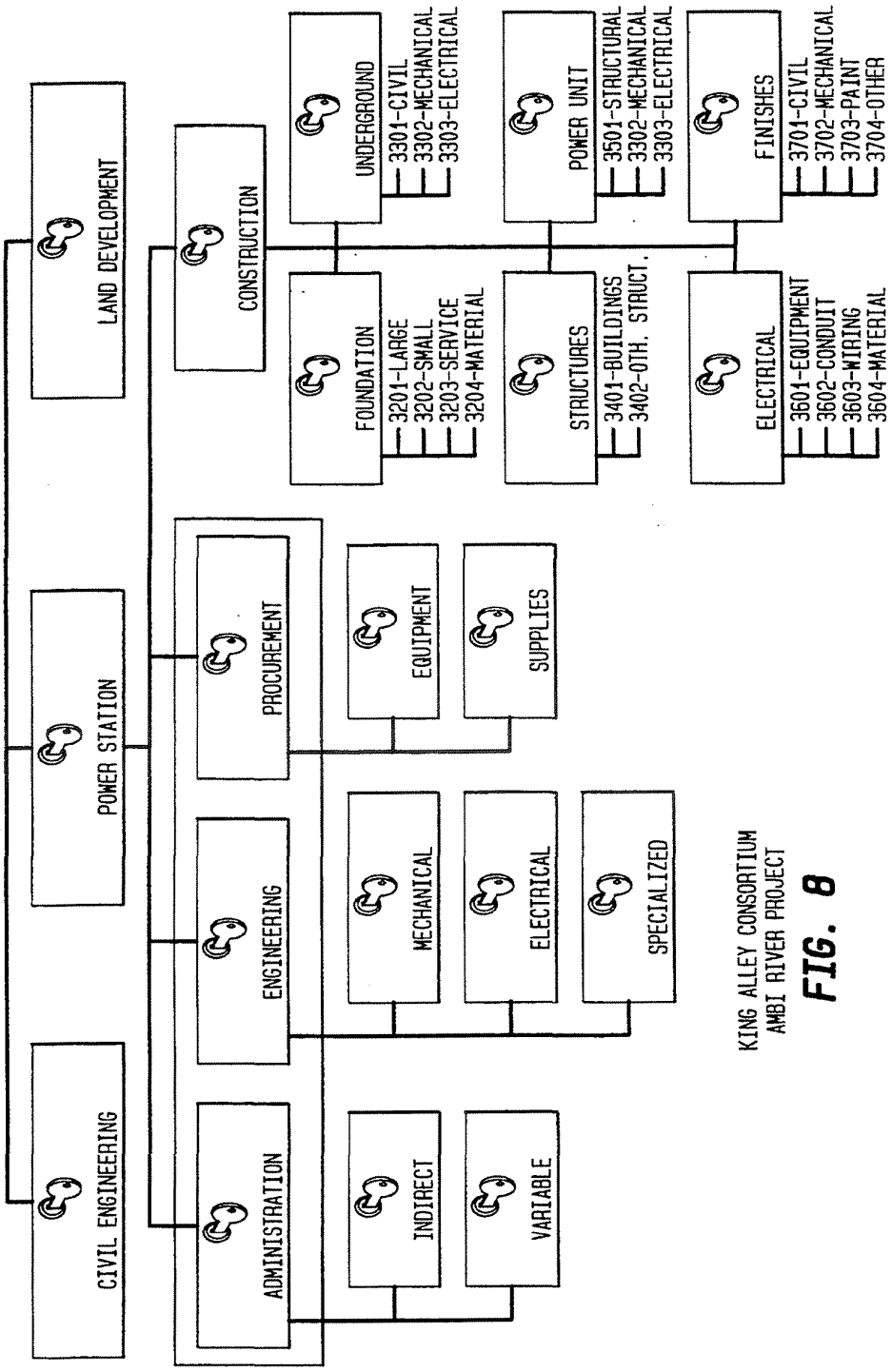

LAND DEVELOPMENT
SCHWARTZ ASSOCIATES
POWER STATION
INYO, INC.
CIVIL ENGINEERING
PANASSAS LTD.
CONSTRUCTION
PROCUREMENT
EQUIPMENT
SUPPLIES
ENGINEERING
MECHANICAL
ELECTRICAL
SPECIALIZED
ADMINISTRATION
INDIRECT
VARIABLE
UNDERGROUND
CODE #: 3300
3301-CIVIL
3302-MECHANICAL
3303-ELECTRICAL
FOUNDATION
CODE #: 3200
3201-LARGE
3202-SMALL
3203-SERVICE
3204-MATERIAL
POWER UNIT
CODE #: 3500
3501-STRUCTURAL
3302-MECHANICAL
3303-ELECTRICAL
STRUCTURES
CODE #: 3400
3401-BUILDINGS
3402-OTH. STRUCT.
FINISHES
CODE #: 3700
3701-CIVIL
3702-MECHANICAL
3703-PAINT
3704-OTHER
ELECTRICAL
CODE #: 3600
3601-EQUIPMENT
3602-CONDUIT
3603-WIRING
3604-MATERIAL
KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 7



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

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-

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

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

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;

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);

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

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.

* * * * *

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

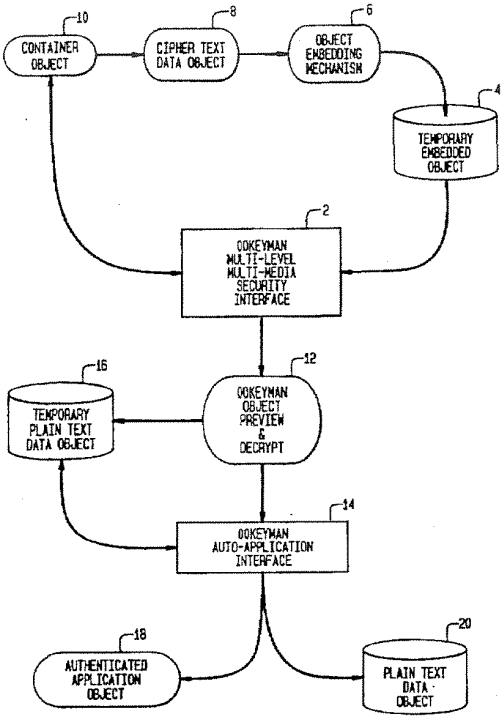
Primary Examiner—Bernarr E. Gregory

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



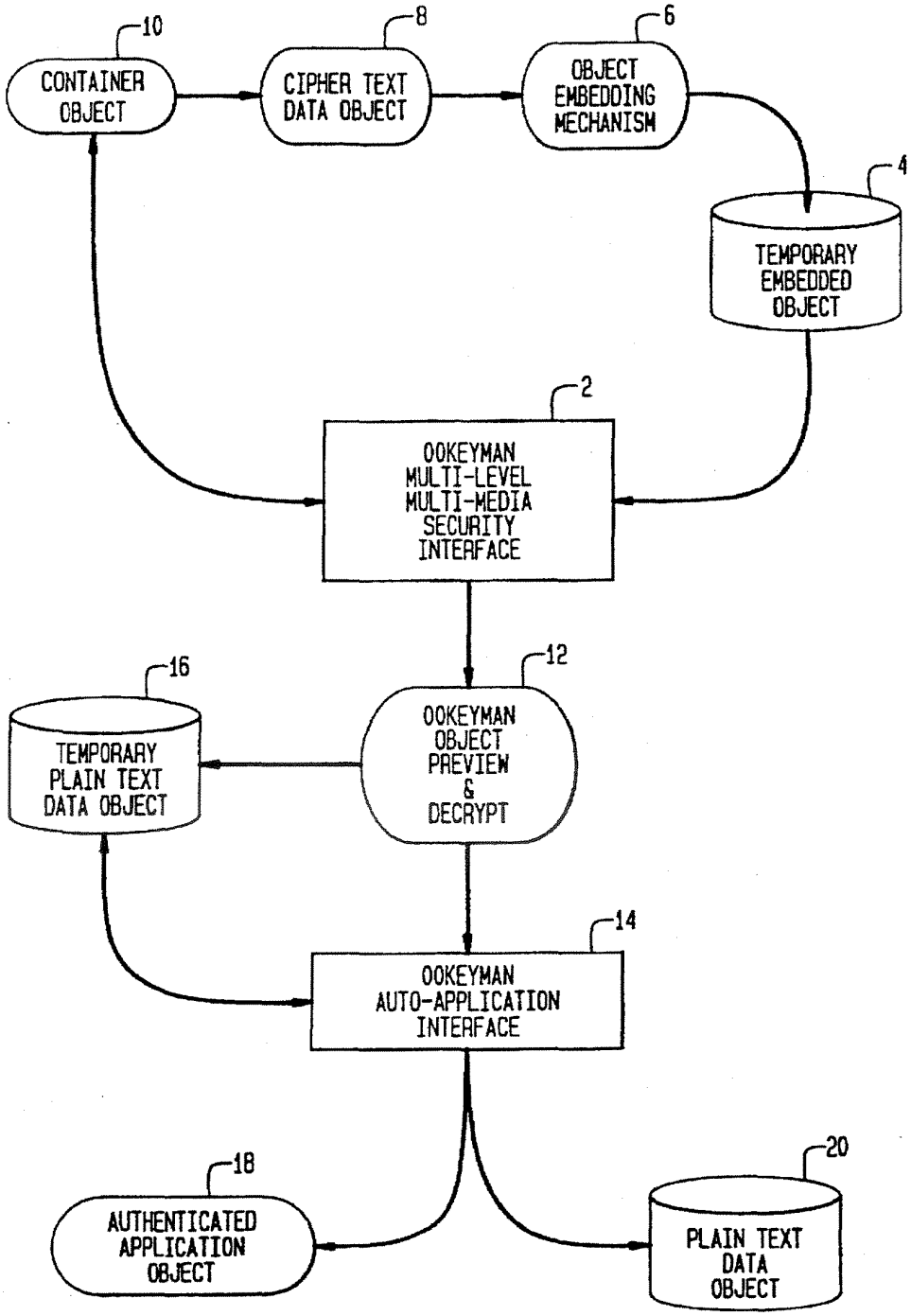

10
CONTAINER OBJECT
8
CIPHER TEXT DATA OBJECT
6
OBJECT EMBEDDING MECHANISM
4
TEMPORARY EMBEDDED OBJECT
2
OOKEYMAN MULTI-LEVEL MULTI-MEDIA SECURITY INTERFACE
16
TEMPORARY PLAIN TEXT DATA OBJECT
12
OOKEYMAN OBJECT PREVIEW & DECRYPT
14
OOKEYMAN AUTO-APPLICATION INTERFACE
18
AUTHENTICATED APPLICATION OBJECT
20
PLAIN TEXT DATA OBJECT

FIG. 1

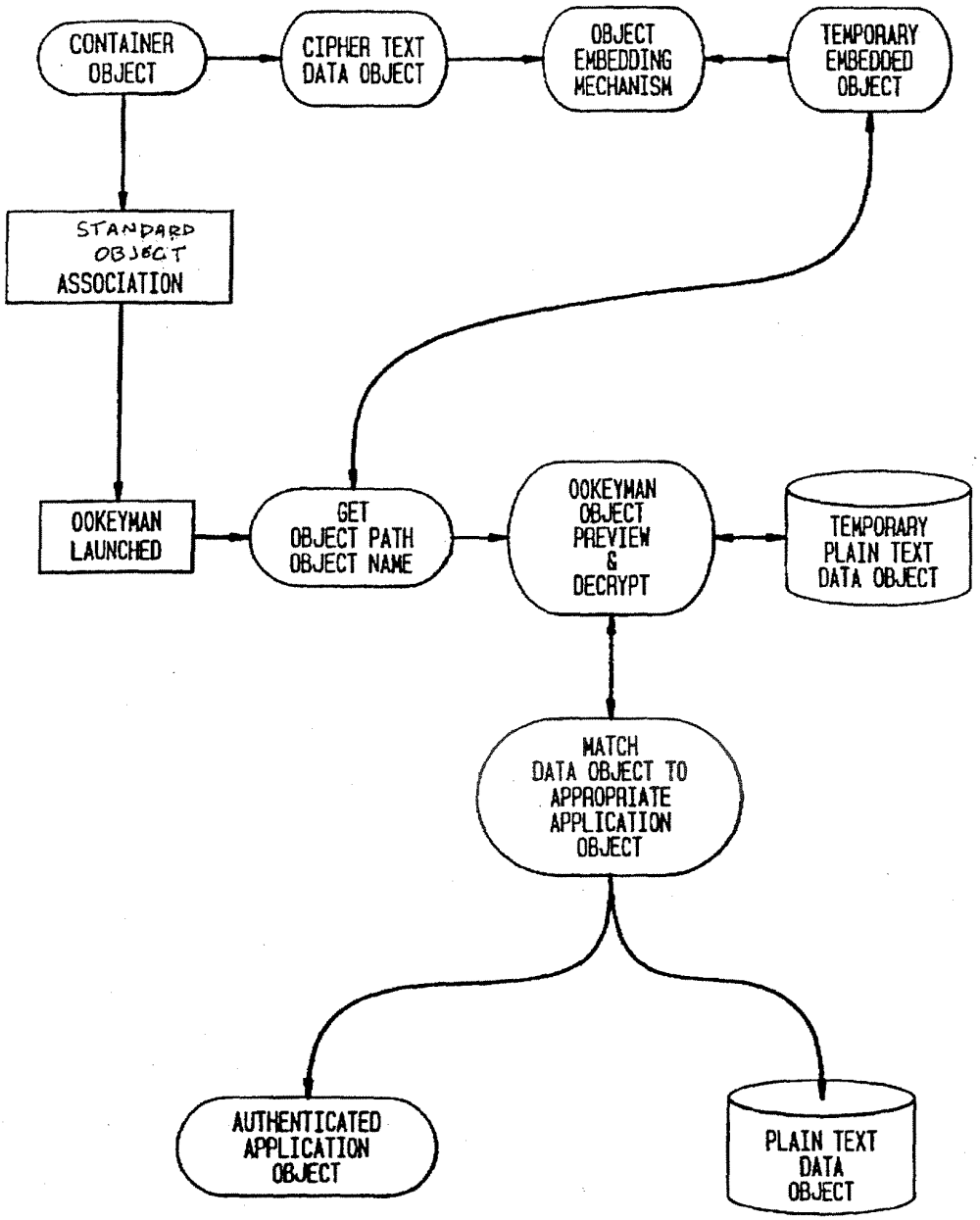

CONTAINER OBJECT
CIPHER TEXT DATA OBJECT
OBJECT EMBEDDING MECHANISM
TEMPORARY EMBEDDED OBJECT
STANDARD OBJECT ASSOCIATION
OOKEYMAN LAUNCHED
GET OBJECT PATH OBJECT NAME
OOKEYMAN OBJECT PREVIEW & DECRYPT
TEMPORARY PLAIN TEXT DATA OBJECT
MATCH DATA OBJECT TO APPROPRIATE APPLICATION OBJECT
AUTHENTICATED APPLICATION OBJECT
PLAIN TEXT DATA OBJECT

FIG. 2

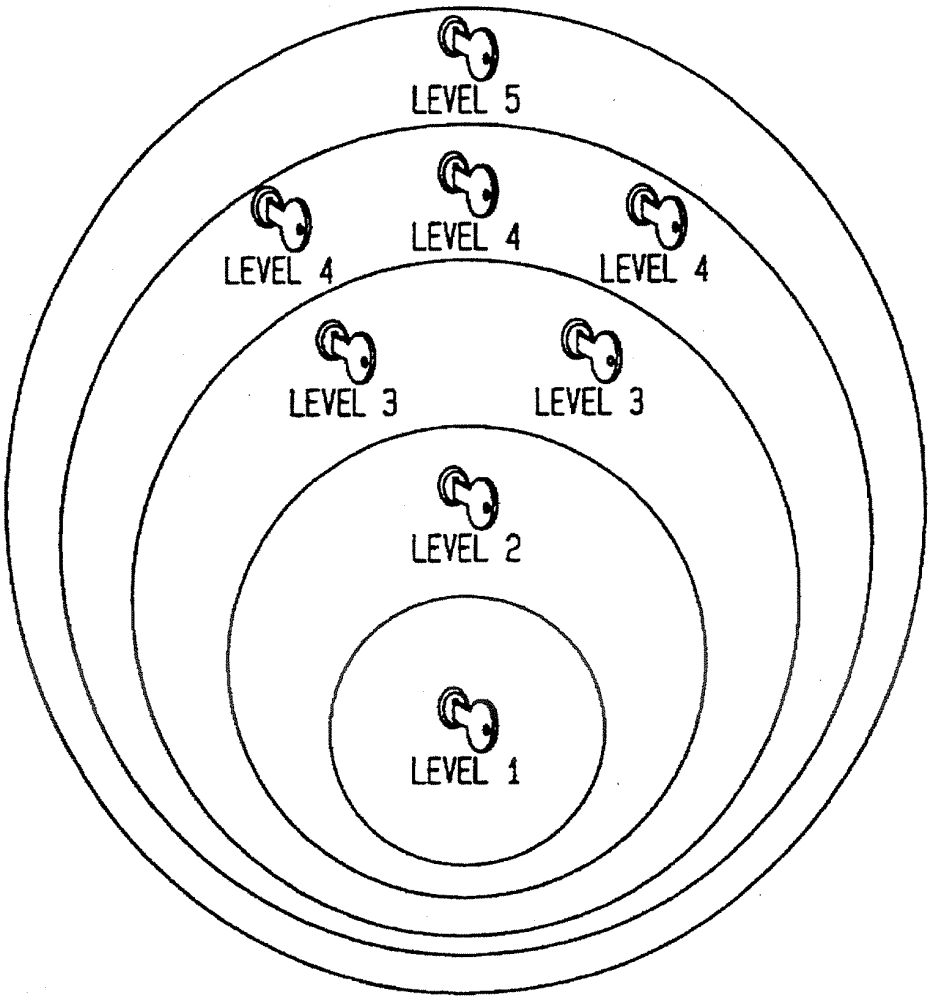


FIG. 3

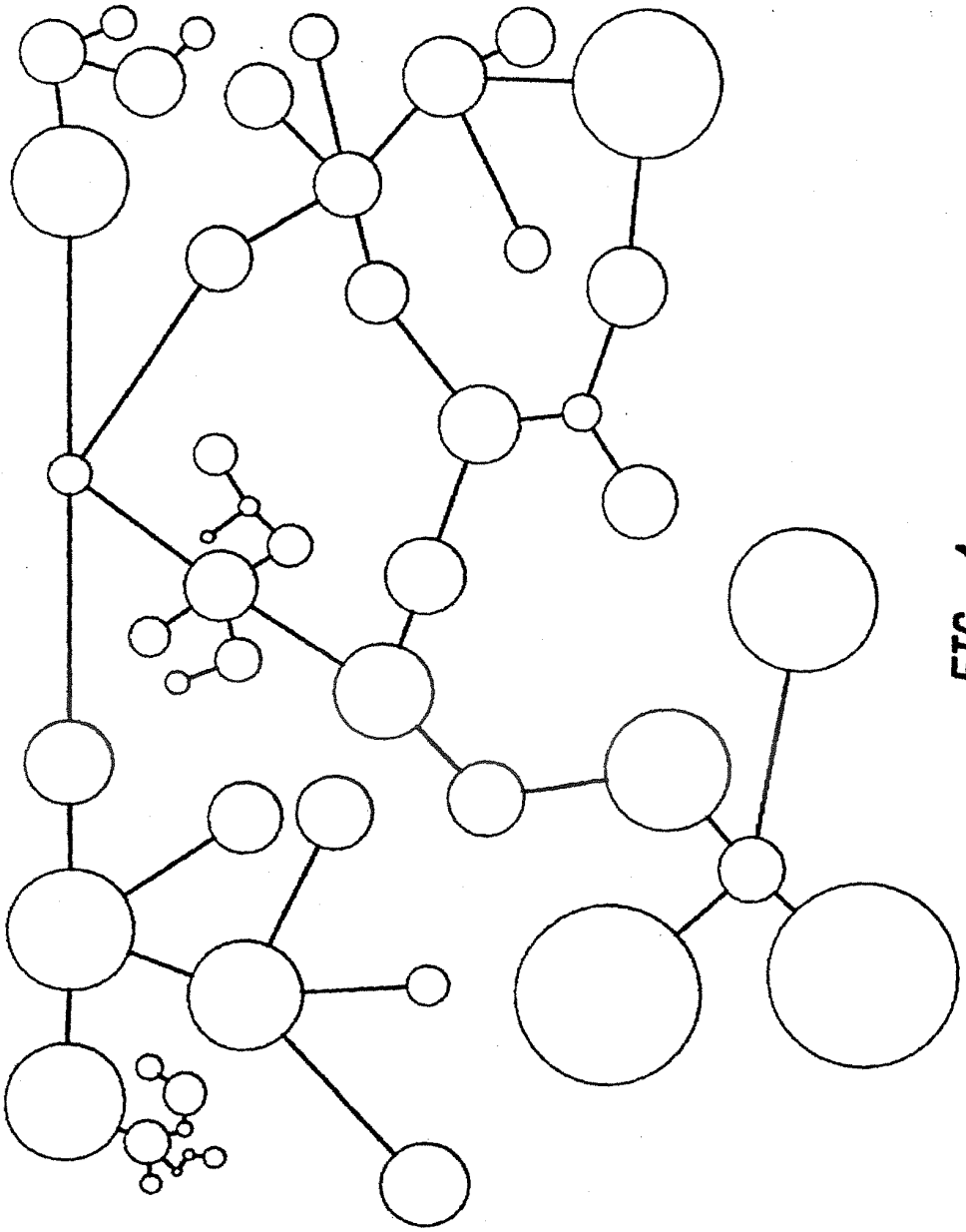


FIG. 4

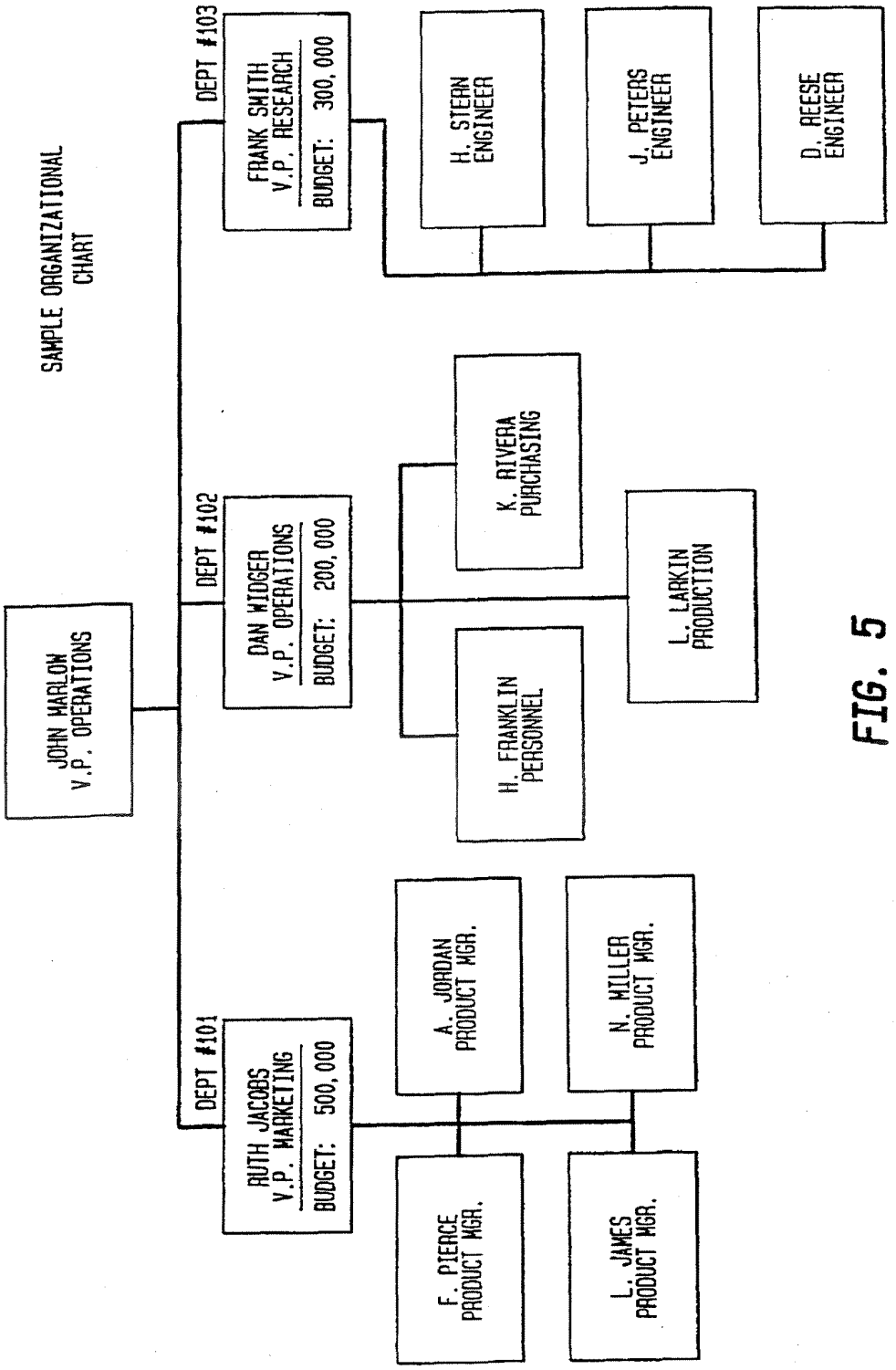

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW
V.P. OPERATIONS
DEPT #103
FRANK SMITH
V.P. RESEARCH
BUDGET: 300,000
H. STERN
ENGINEER
J. PETERS
ENGINEER
D. REESE
ENGINEER
DEPT #102
DAN WIDGER
V.P. OPERATIONS
BUDGET: 200,000
K. RIVERA
PURCHASING
L. LARKIN
PRODUCTION
H. FRANKLIN
PERSONNEL
DEPT #101
RUTH JACOBS
V.P. MARKETING
BUDGET: 500,000
A. JORDAN
PRODUCT MGR.
N. MILLER
PRODUCT MGR.
F. PIERCE
PRODUCT MGR.
L. JAMES
PRODUCT MGR.

FIG. 5

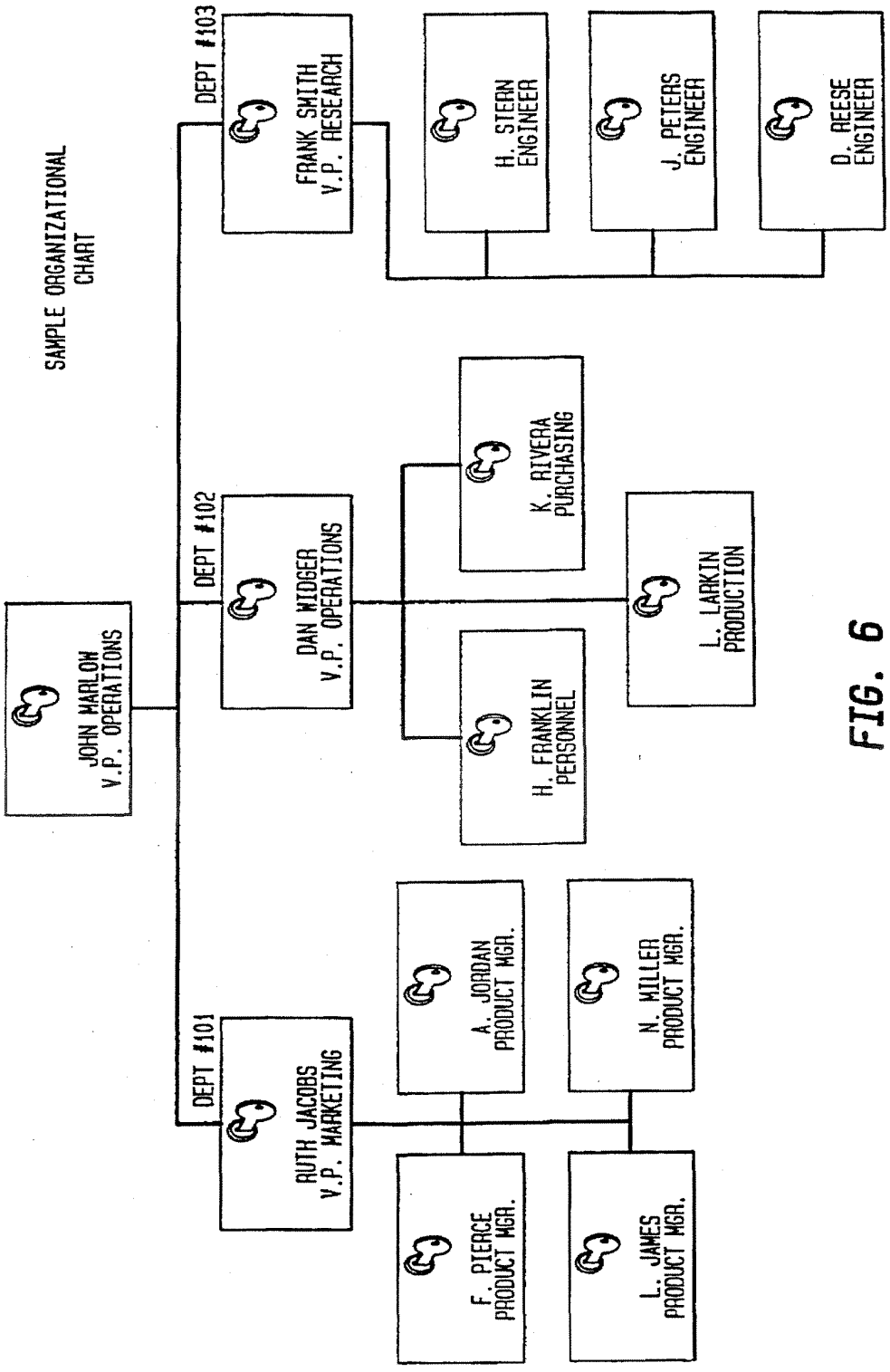

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW V.P. OPERATIONS
DEPT #103
FRANK SMITH V.P. RESEARCH
H. STERN ENGINEER
J. PETERS ENGINEER
D. REESE ENGINEER
DEPT #102
DAN WIDGER V.P. OPERATIONS
K. RIVERA PURCHASING
L. LARKIN PRODUCTION
H. FRANKLIN PERSONNEL
DEPT #101
RUTH JACOBS V.P. MARKETING
A. JORDAN PRODUCT MGR.
N. MILLER PRODUCT MGR.
F. PIERCE PRODUCT MGR.
L. JAMES PRODUCT MGR.

FIG. 6

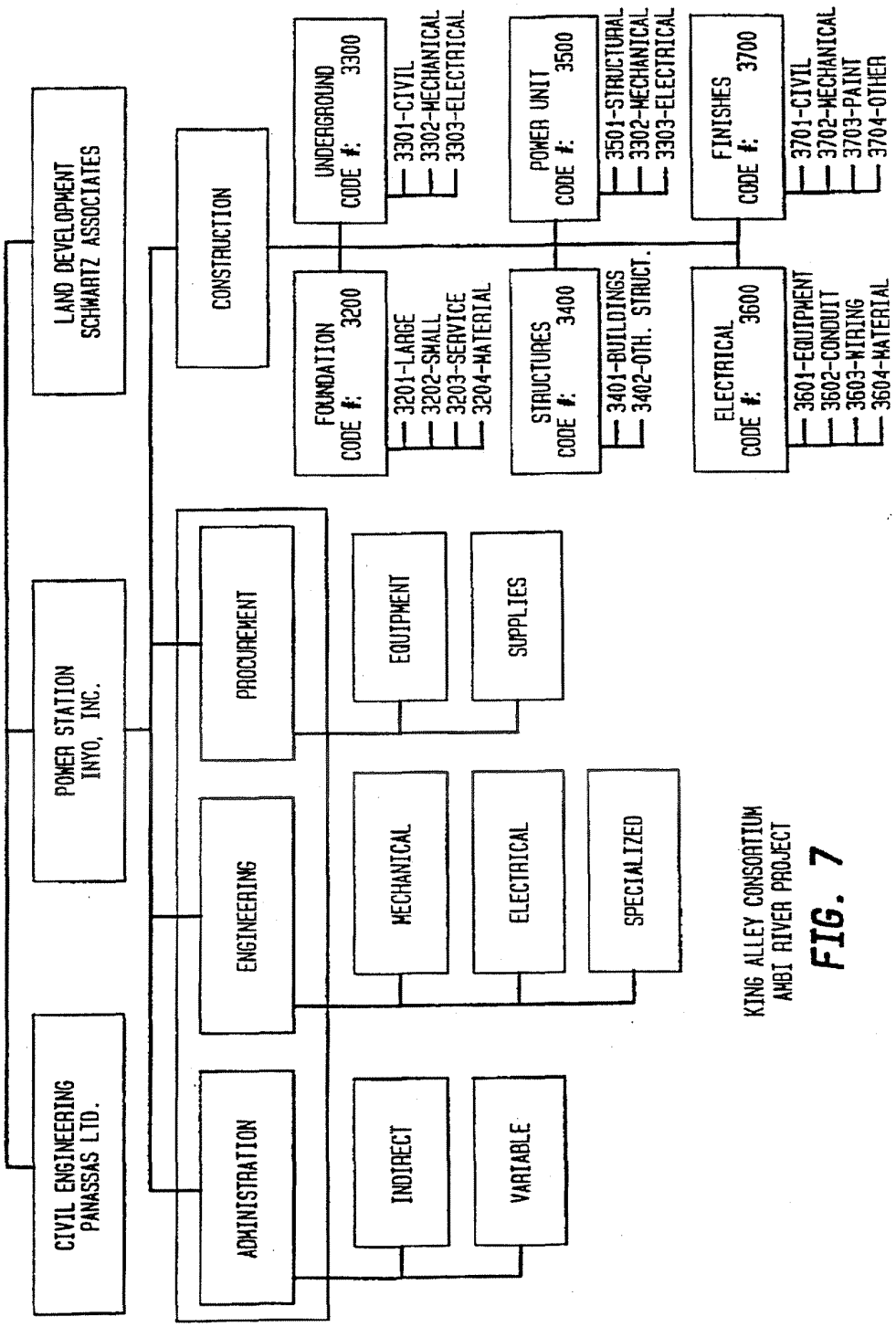

LAND DEVELOPMENT
SCHWARTZ ASSOCIATES
POWER STATION
INYO, INC.
CIVIL ENGINEERING
PANASSAS LTD.
CONSTRUCTION
UNDERGROUND
CODE #: 3300
3301-CIVIL
3302-MECHANICAL
3303-ELECTRICAL
FOUNDATION
CODE #: 3200
3201-LARGE
3202-SMALL
3203-SERVICE
3204-MATERIAL
POWER UNIT
CODE #: 3500
3501-STRUCTURAL
3302-MECHANICAL
3303-ELECTRICAL
FINISHES
CODE #: 3700
3701-CIVIL
3702-MECHANICAL
3703-PAINT
3704-OTHER
STRUCTURES
CODE #: 3400
3401-BUILDINGS
3402-OTH. STRUCT.
ELECTRICAL
CODE #: 3600
3601-EQUIPMENT
3602-CONDUIT
3603-WIRING
3604-MATERIAL
PROCUREMENT
EQUIPMENT
SUPPLIES
ENGINEERING
MECHANICAL
ELECTRICAL
SPECIALIZED
ADMINISTRATION
INDIRECT
VARIABLE
KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 7

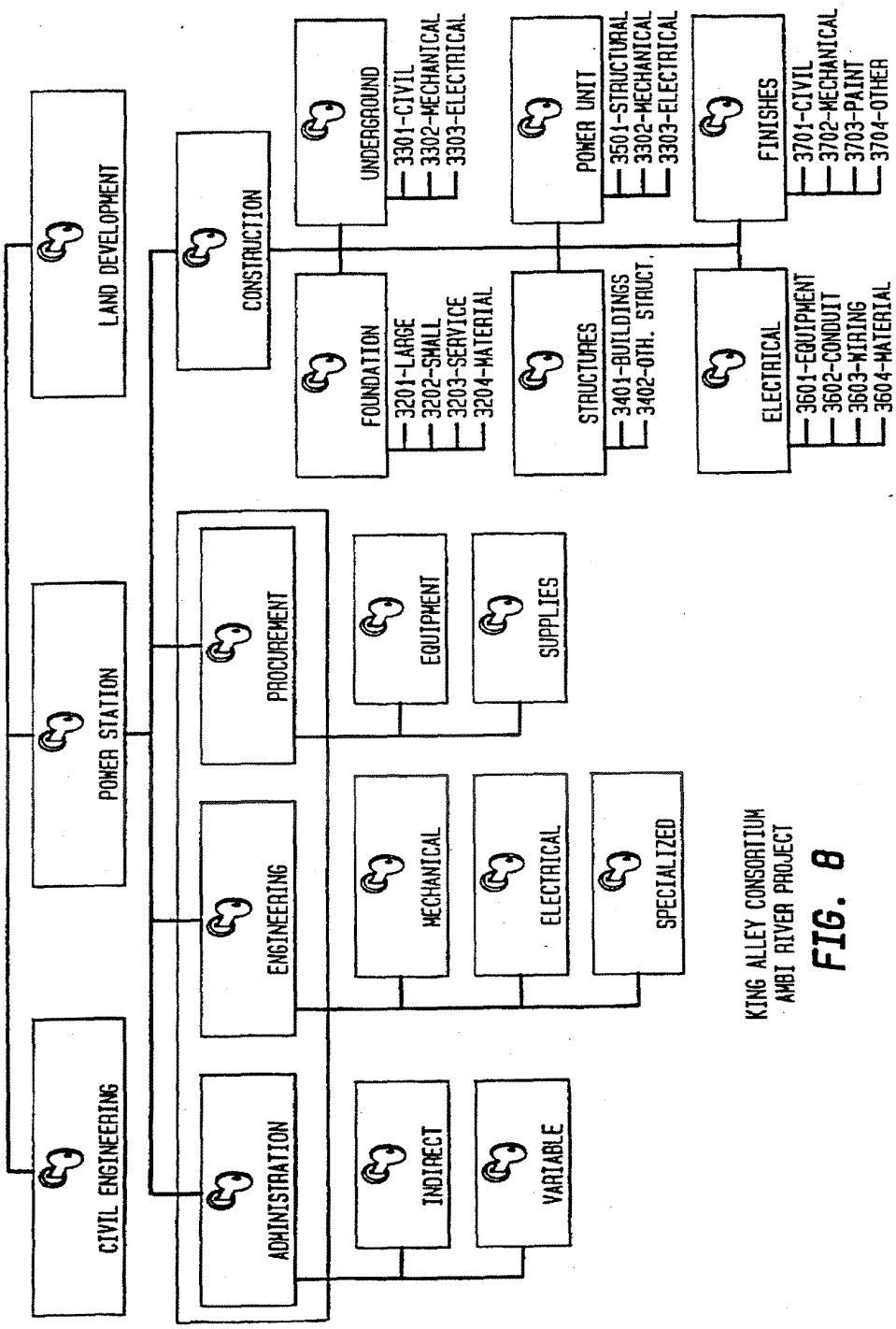

LAND DEVELOPMENT
POWER STATION
CIVIL ENGINEERING
CONSTRUCTION
PROCUREMENT
EQUIPMENT
SUPPLIES
ENGINEERING
MECHANICAL
ELECTRICAL
SPECIALIZED
ADMINISTRATION
INDIRECT
VARIABLE
UNDERGROUND
3301-CIVIL
3302-MECHANICAL
3303-ELECTRICAL
FOUNDATION
3201-LARGE
3202-SMALL
3203-SERVICE
3204-MATERIAL
POWER UNIT
3501-STRUCTURAL
3302-MECHANICAL
3303-ELECTRICAL
STRUCTURES
3401-BUILDINGS
3402-OTH. STRUCT.
FINISHES
3701-CIVIL
3702-MECHANICAL
3703-PAINT
3704-OTHER
ELECTRICAL
3601-EQUIPMENT
3602-CONDUIT
3603-WIRING
3604-MATERIAL
KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

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

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,

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-

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

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;

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;

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;

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.

* * * * *

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

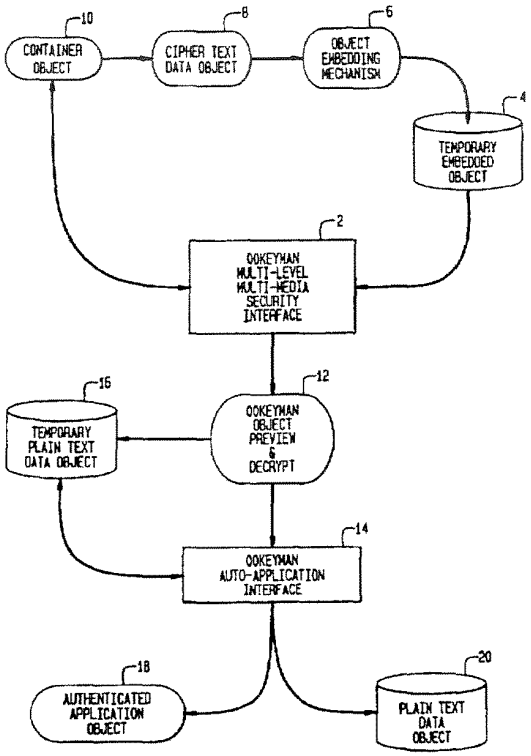
Primary Examiner—Bernarr E. Gregory

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



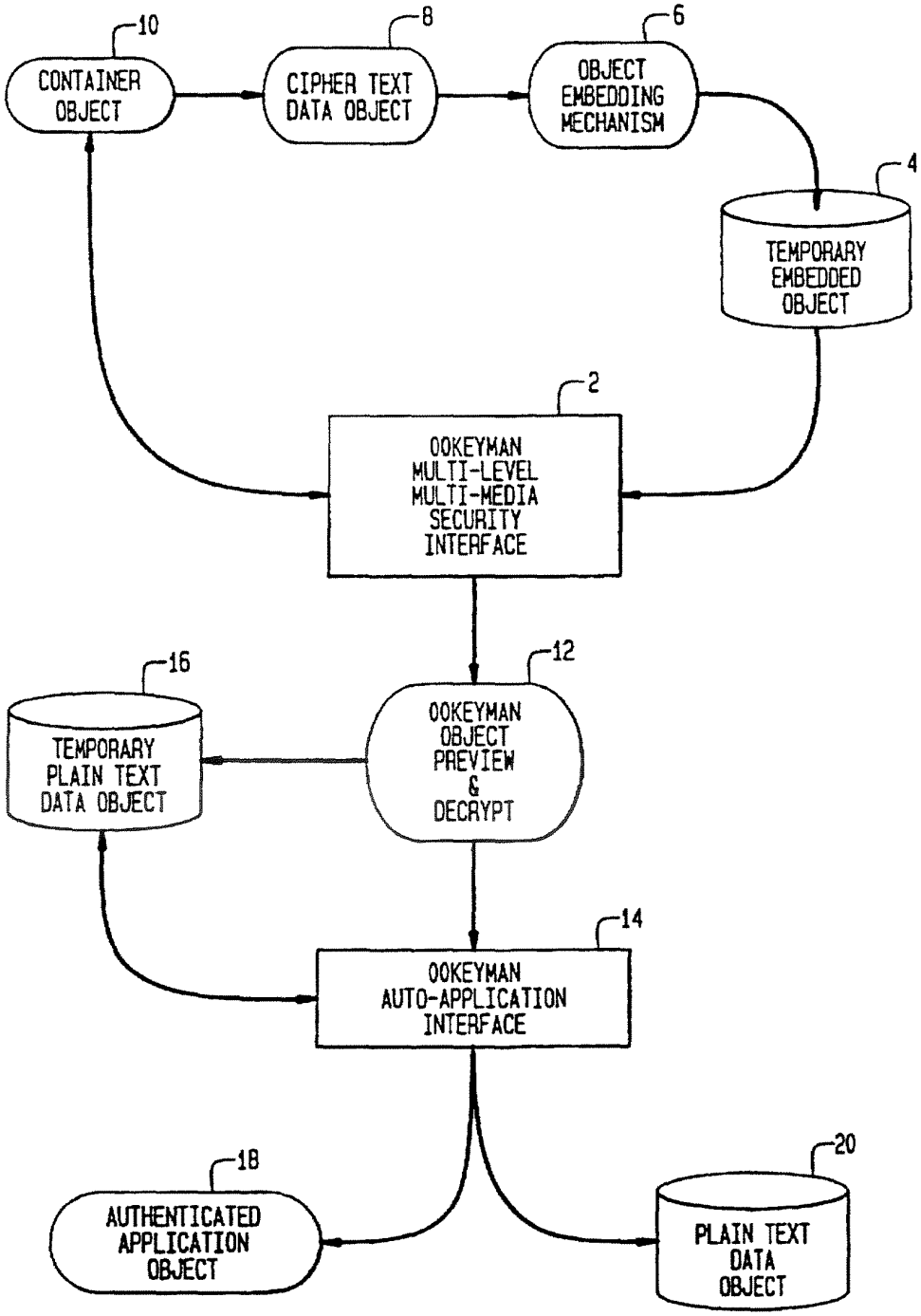

10
CONTAINER OBJECT
8
CIPHER TEXT DATA OBJECT
6
OBJECT EMBEDDING MECHANISM
4
TEMPORARY EMBEDDED OBJECT
2
OOKEYMAN MULTI-LEVEL MULTI-MEDIA SECURITY INTERFACE
16
TEMPORARY PLAIN TEXT DATA OBJECT
12
OOKEYMAN OBJECT PREVIEW & DECRYPT
14
OOKEYMAN AUTO-APPLICATION INTERFACE
18
AUTHENTICATED APPLICATION OBJECT
20
PLAIN TEXT DATA OBJECT

FIG. 1

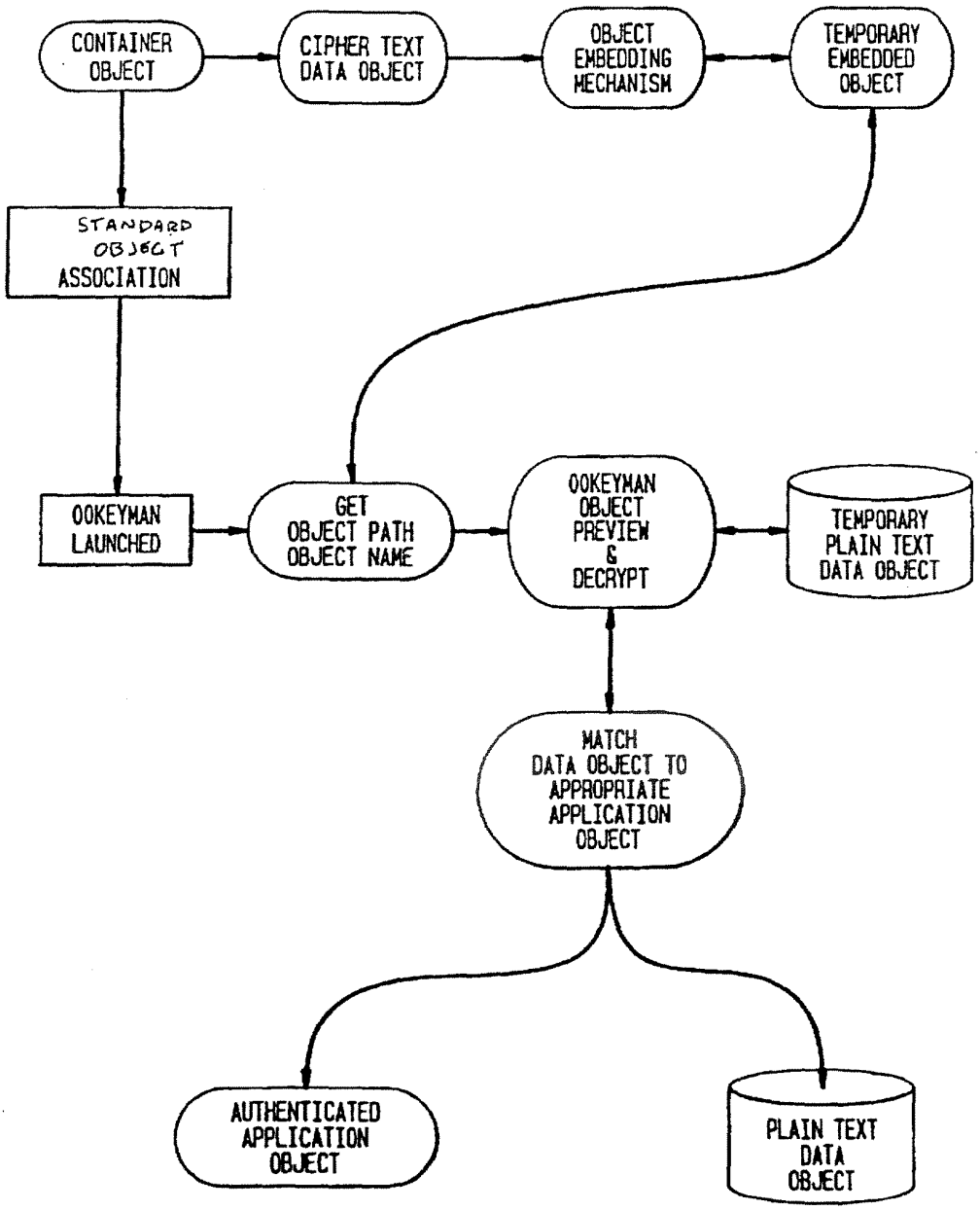

CONTAINER OBJECT
CIPHER TEXT DATA OBJECT
OBJECT EMBEDDING MECHANISM
TEMPORARY EMBEDDED OBJECT
STANDARD OBJECT ASSOCIATION
OOKEYMAN LAUNCHED
GET OBJECT PATH OBJECT NAME
OOKEYMAN OBJECT PREVIEW & DECRYPT
TEMPORARY PLAIN TEXT DATA OBJECT
MATCH DATA OBJECT TO APPROPRIATE APPLICATION OBJECT
AUTHENTICATED APPLICATION OBJECT
PLAIN TEXT DATA OBJECT

FIG. 2

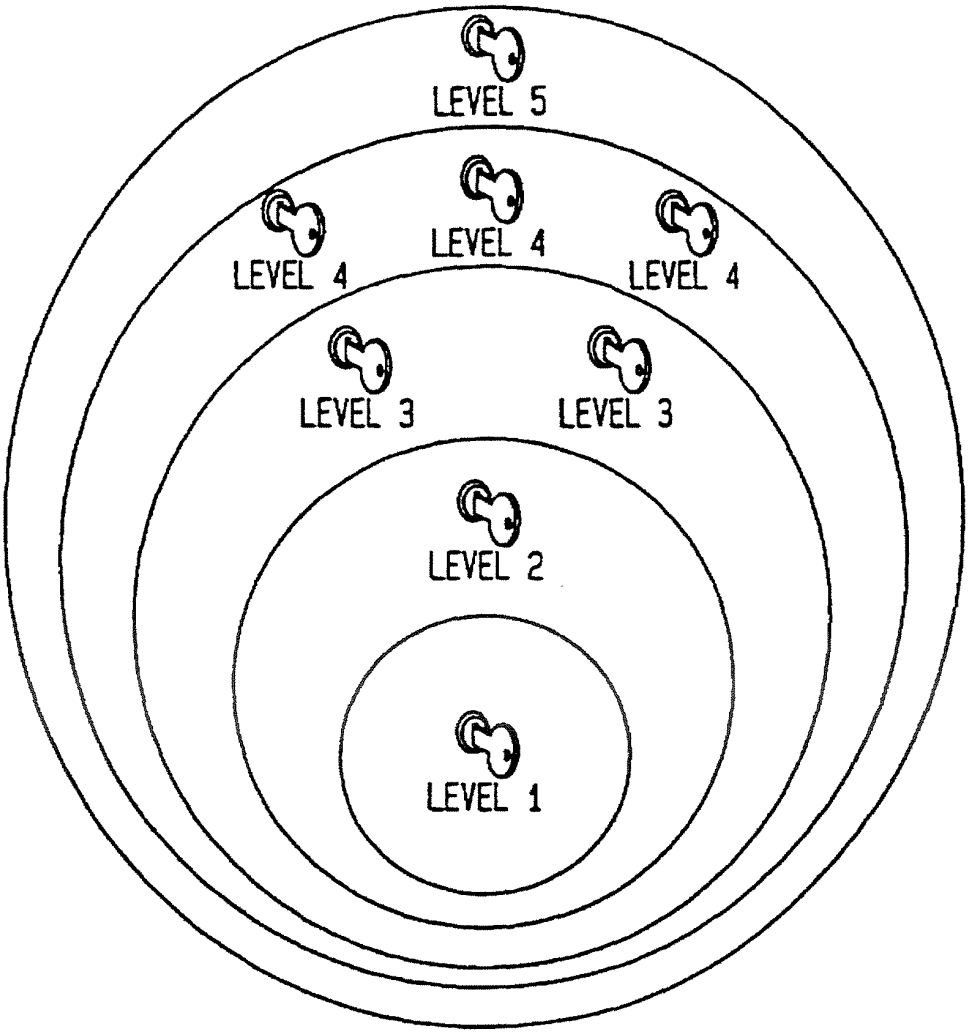


FIG. 3

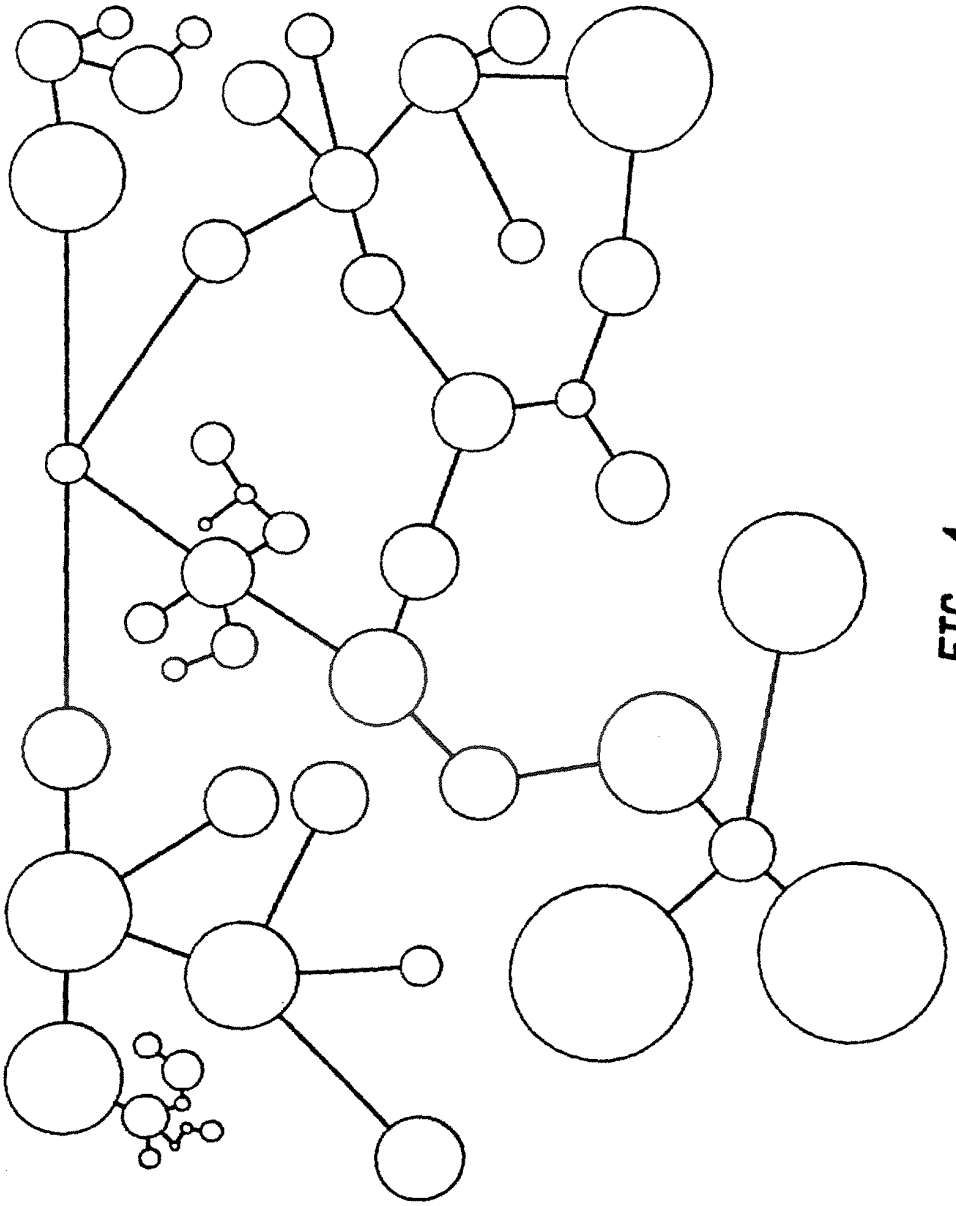


FIG. 4

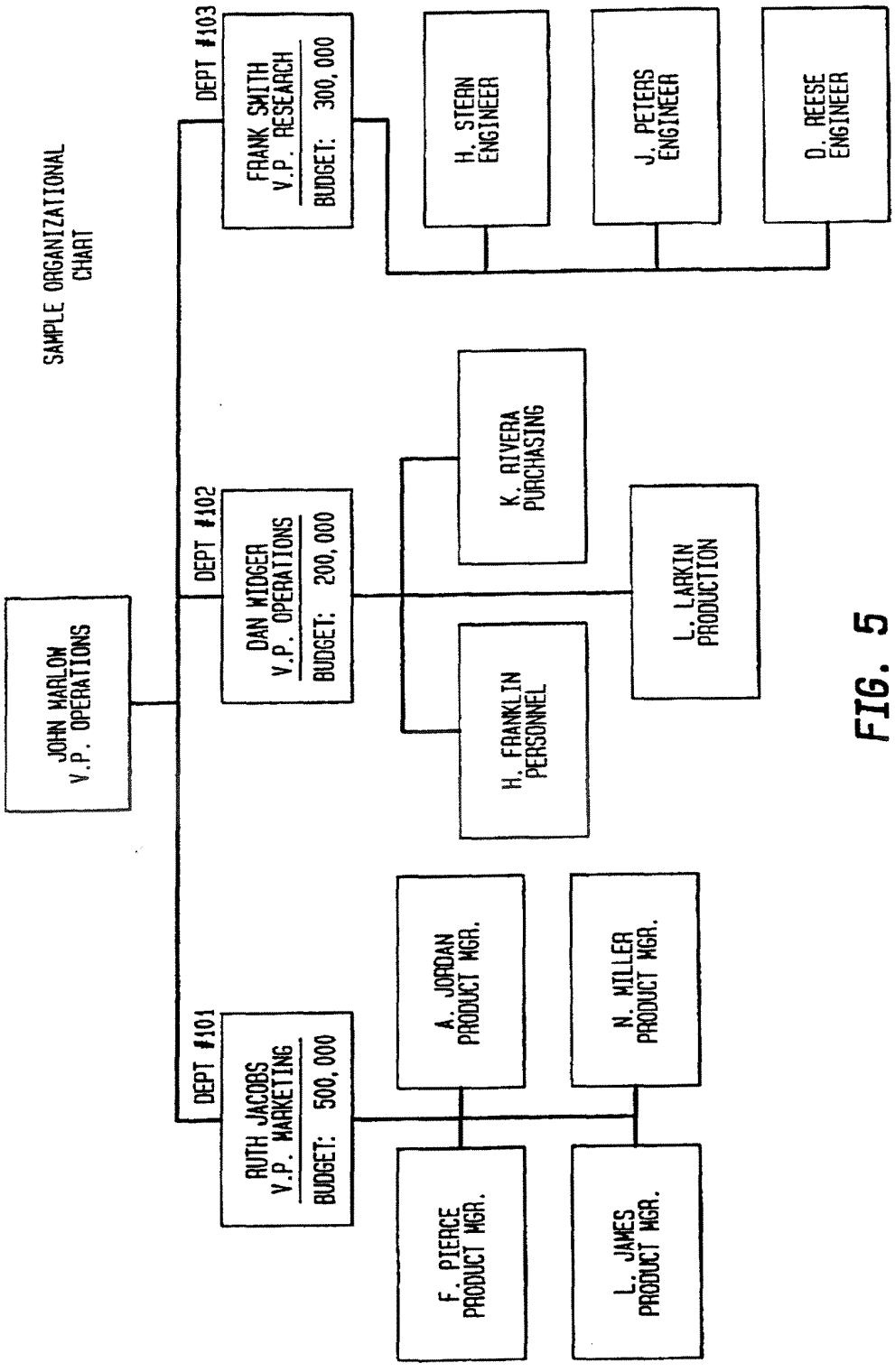

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW
V.P. OPERATIONS
DEPT #101
RUTH JACOBS
V.P. MARKETING
BUDGET: 500,000
DEPT #102
DAN MIDGER
V.P. OPERATIONS
BUDGET: 200,000
DEPT #103
FRANK SMITH
V.P. RESEARCH
BUDGET: 300,000
A. JORDAN
PRODUCT MGR.
F. PIERCE
PRODUCT MGR.
N. MILLER
PRODUCT MGR.
L. JAMES
PRODUCT MGR.
H. FRANKLIN
PERSONNEL
K. RIVERA
PURCHASING
L. LARKIN
PRODUCTION
H. STERN
ENGINEER
J. PETERS
ENGINEER
D. REESE
ENGINEER

FIG. 5

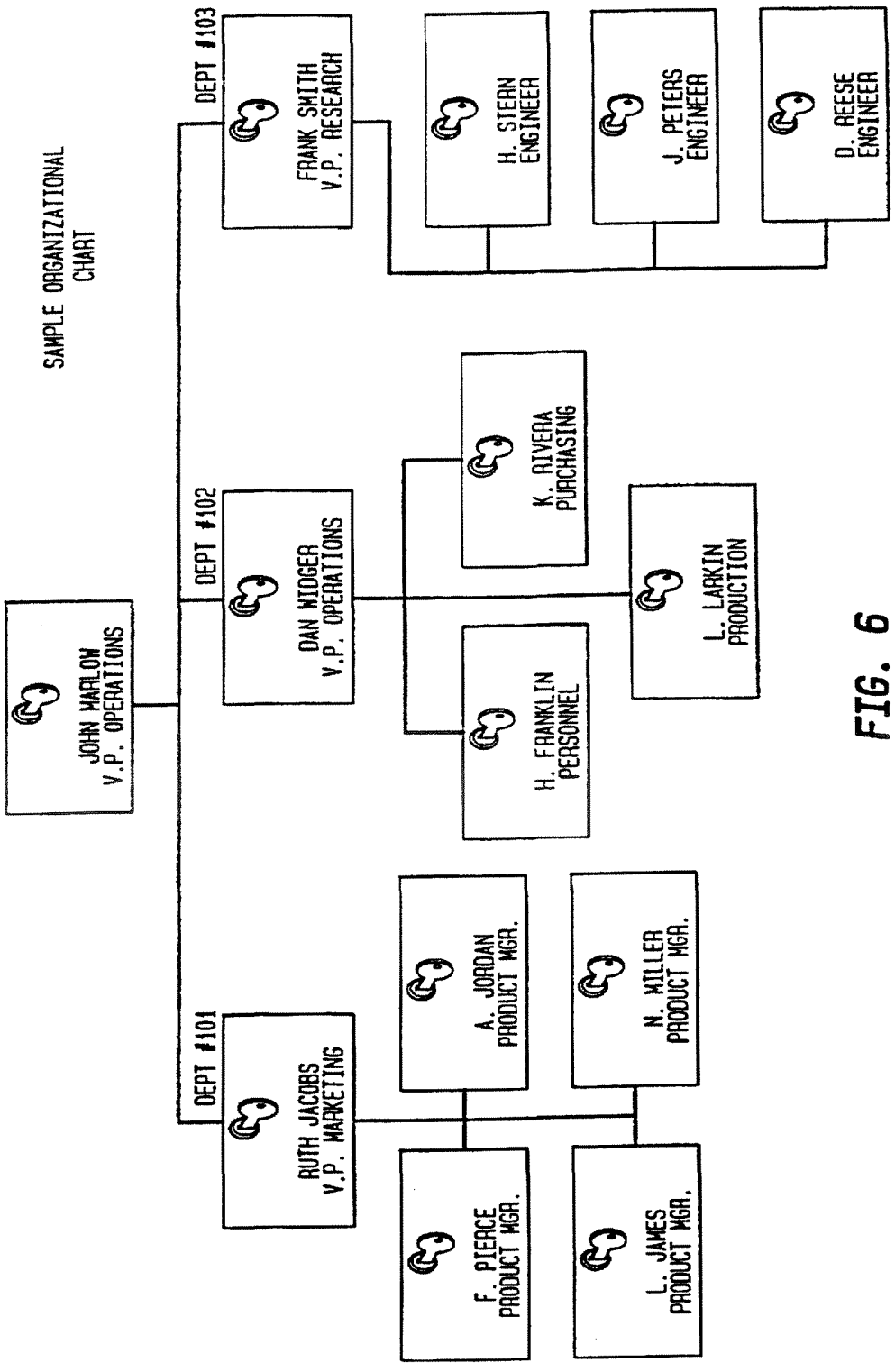

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW V.P. OPERATIONS
DEPT #101
RUTH JACOBS V.P. MARKETING
DEPT #102
DAN MIDGER V.P. OPERATIONS
DEPT #103
FRANK SMITH V.P. RESEARCH
A. JORDAN PRODUCT MGR.
F. PIERCE PRODUCT MGR.
N. MILLER PRODUCT MGR.
L. JAMES PRODUCT MGR.
K. RIVERA PURCHASING
H. FRANKLIN PERSONNEL
L. LARKIN PRODUCTION
H. STERN ENGINEER
J. PETERS ENGINEER
D. REESE ENGINEER

FIG. 6

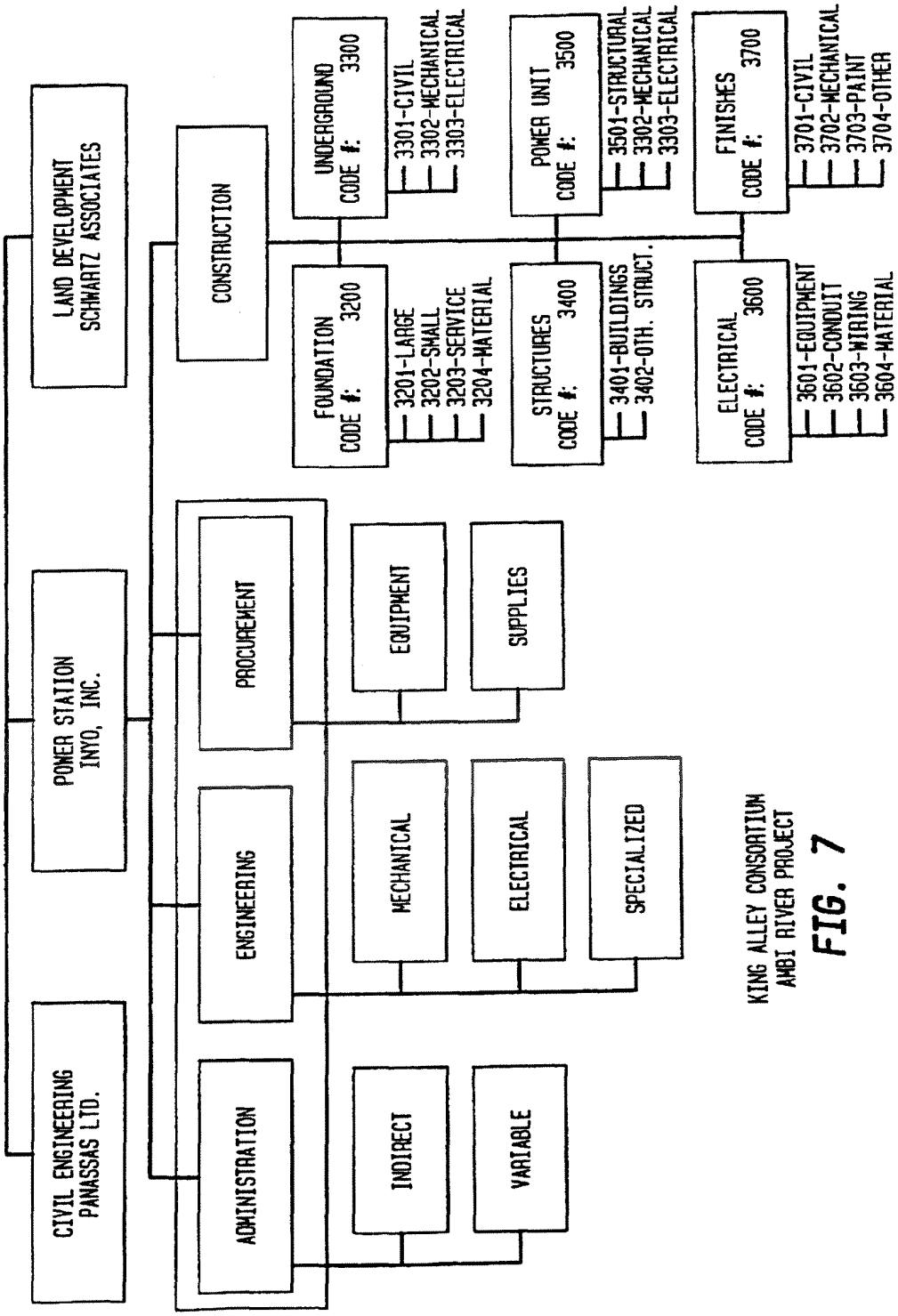
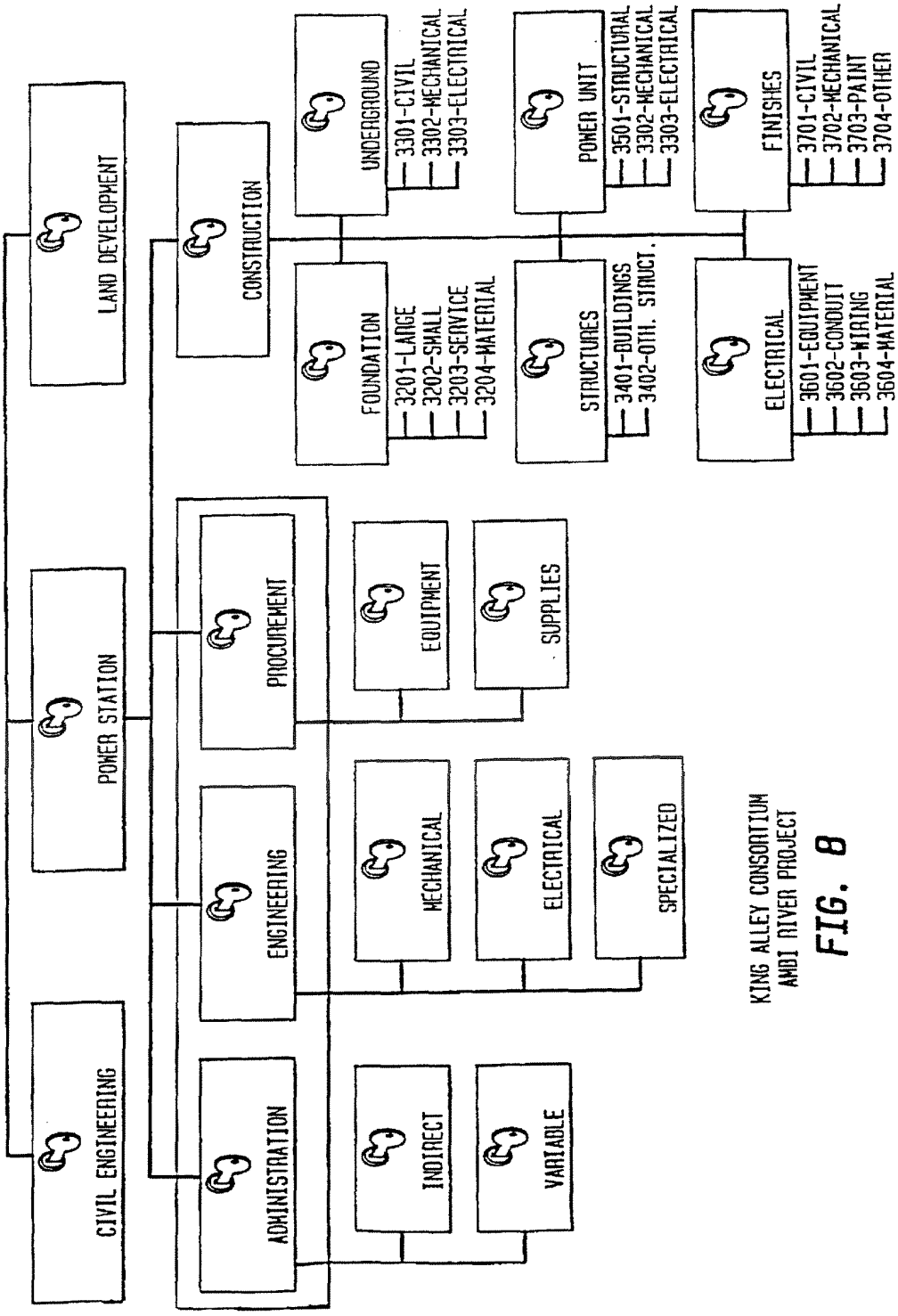

KING ALLEY CONSORTIUM
AMBI RIVER PROJECT
CIVIL ENGINEERING
PANASSAS LTD.
POWER STATION
INYO, INC.
LAND DEVELOPMENT
SCHWARTZ ASSOCIATES
ADMINISTRATION
ENGINEERING
PROCUREMENT
CONSTRUCTION
INDIRECT
VARIABLE
MECHANICAL
ELECTRICAL
SPECIALIZED
EQUIPMENT
SUPPLIES
FOUNDATION
CODE #: 3200
3201-LARGE
3202-SMALL
3203-SERVICE
3204-MATERIAL
UNDERGROUND
CODE #: 3300
3301-CIVIL
3302-MECHANICAL
3303-ELECTRICAL
STRUCTURES
CODE #: 3400
3401-BUILDINGS
3402-OTH. STRUCT.
POWER UNIT
CODE #: 3500
3501-STRUCTURAL
3302-MECHANICAL
3303-ELECTRICAL
ELECTRICAL
CODE #: 3600
3601-EQUIPMENT
3602-CONDUIT
3603-WIRING
3604-MATERIAL
FINISHES
CODE #: 3700
3701-CIVIL
3702-MECHANICAL
3703-PAINT
3704-OTHER

FIG. 7



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

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

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

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

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

(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

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

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.

* * * * *

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

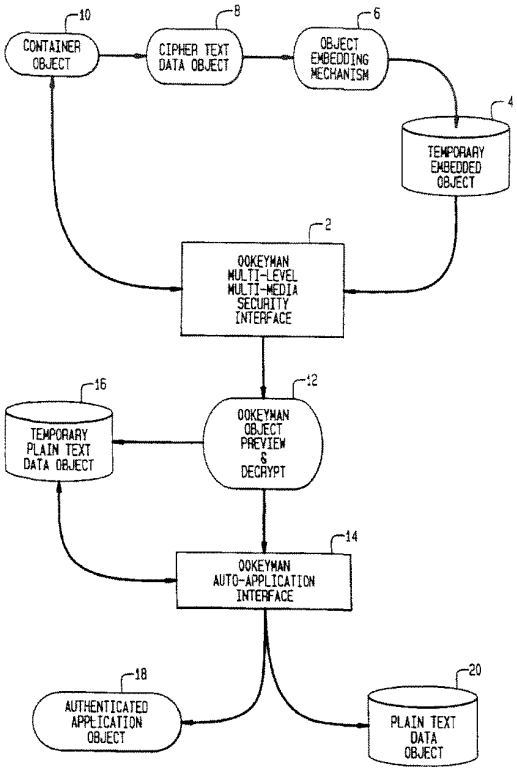
4,218,582	8/1980	Hellman et al.	380/30
4,405,829	9/1983	Rivest et al.	380/30
4,424,414	1/1984	Hellman et al.	380/30
4,713,753	12/1987	Boebert et al.	380/4 X
4,864,616	9/1989	Pond et al.	380/25
4,955,082	9/1990	Hattori et al.	455/440
4,962,533	10/1990	Krueger et al.	380/25
4,984,272	1/1991	McIlroy et 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



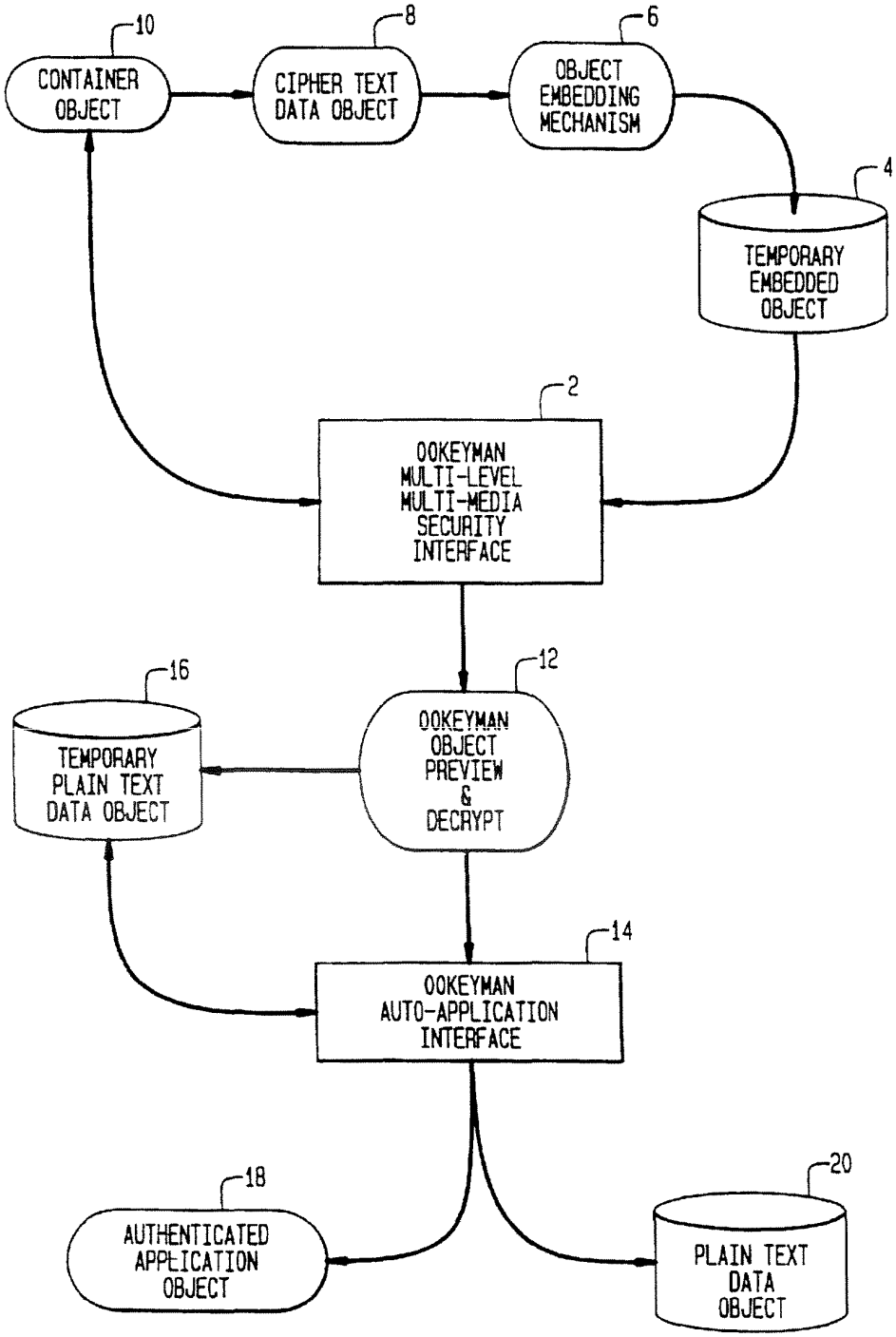

10
CONTAINER OBJECT
8
CIPHER TEXT DATA OBJECT
6
OBJECT EMBEDDING MECHANISM
4
TEMPORARY EMBEDDED OBJECT
2
OOKEYMAN MULTI-LEVEL MULTI-MEDIA SECURITY INTERFACE
16
TEMPORARY PLAIN TEXT DATA OBJECT
12
OOKEYMAN OBJECT PREVIEW & DECRYPT
14
OOKEYMAN AUTO-APPLICATION INTERFACE
18
AUTHENTICATED APPLICATION OBJECT
20
PLAIN TEXT DATA OBJECT

FIG. 1

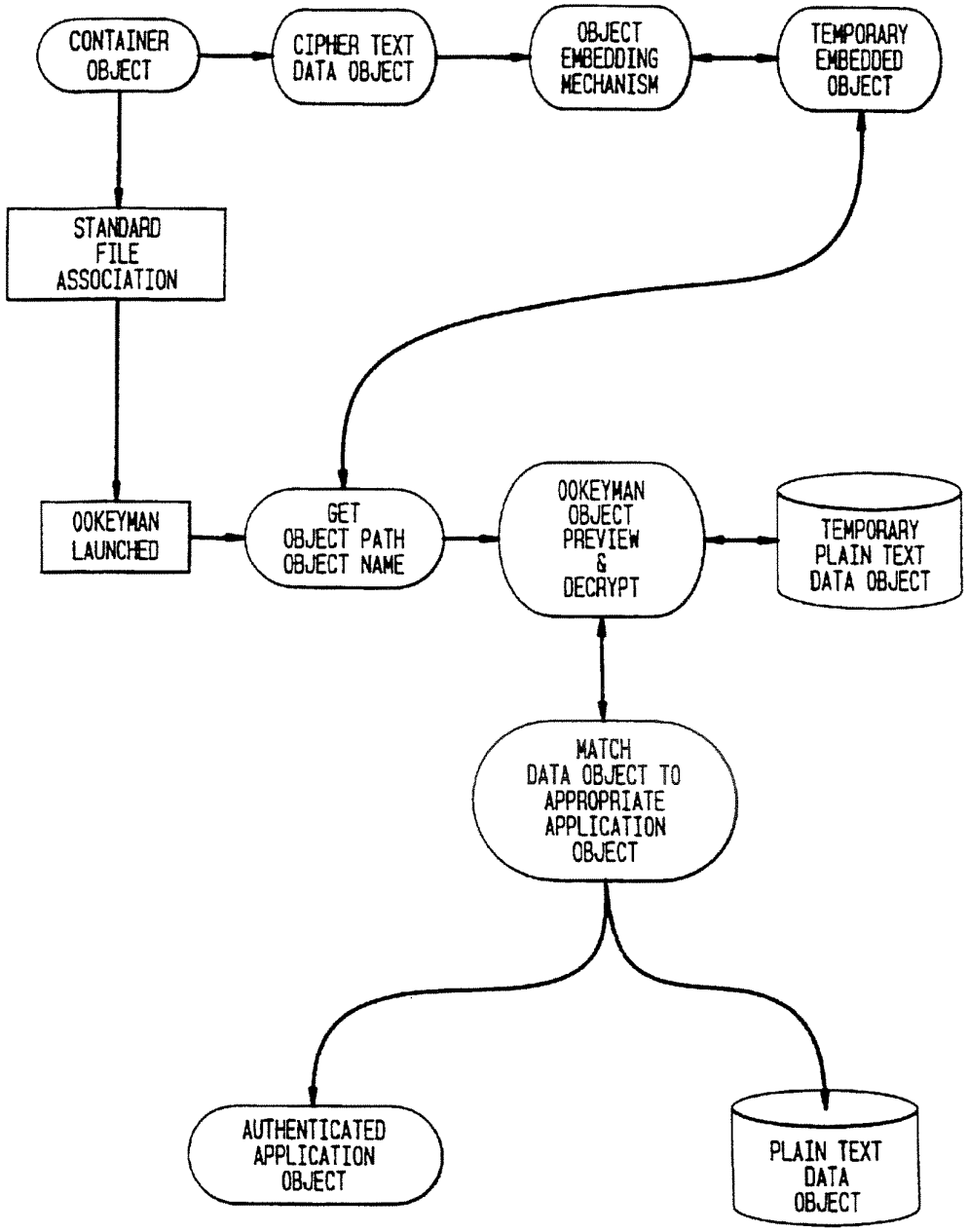

CONTAINER OBJECT
CIPHER TEXT DATA OBJECT
OBJECT EMBEDDING MECHANISM
TEMPORARY EMBEDDED OBJECT
STANDARD FILE ASSOCIATION
OOKEYMAN LAUNCHED
GET OBJECT PATH OBJECT NAME
OOKEYMAN OBJECT PREVIEW & DECRYPT
TEMPORARY PLAIN TEXT DATA OBJECT
MATCH DATA OBJECT TO APPROPRIATE APPLICATION OBJECT
AUTHENTICATED APPLICATION OBJECT
PLAIN TEXT DATA OBJECT

FIG. 2

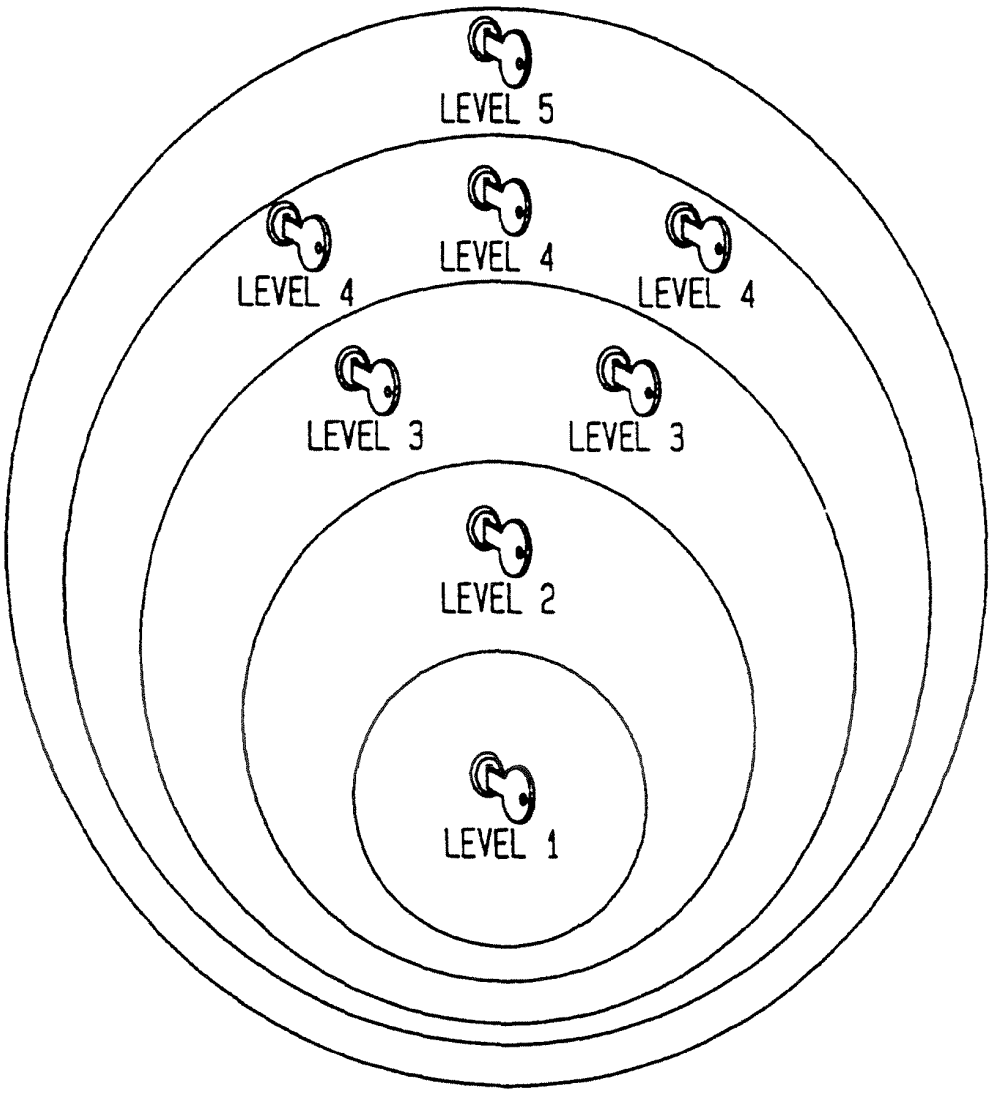


FIG. 3

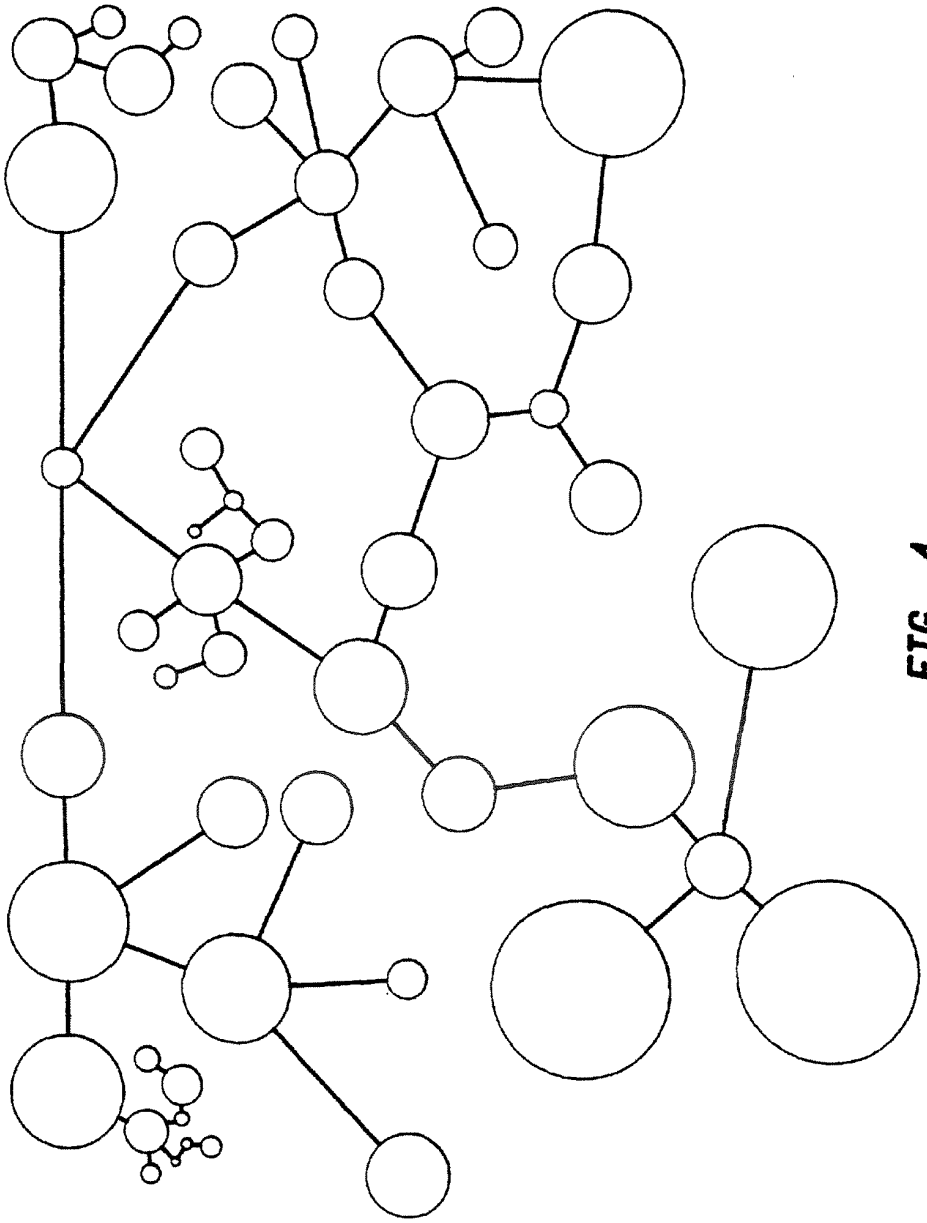


FIG. 4

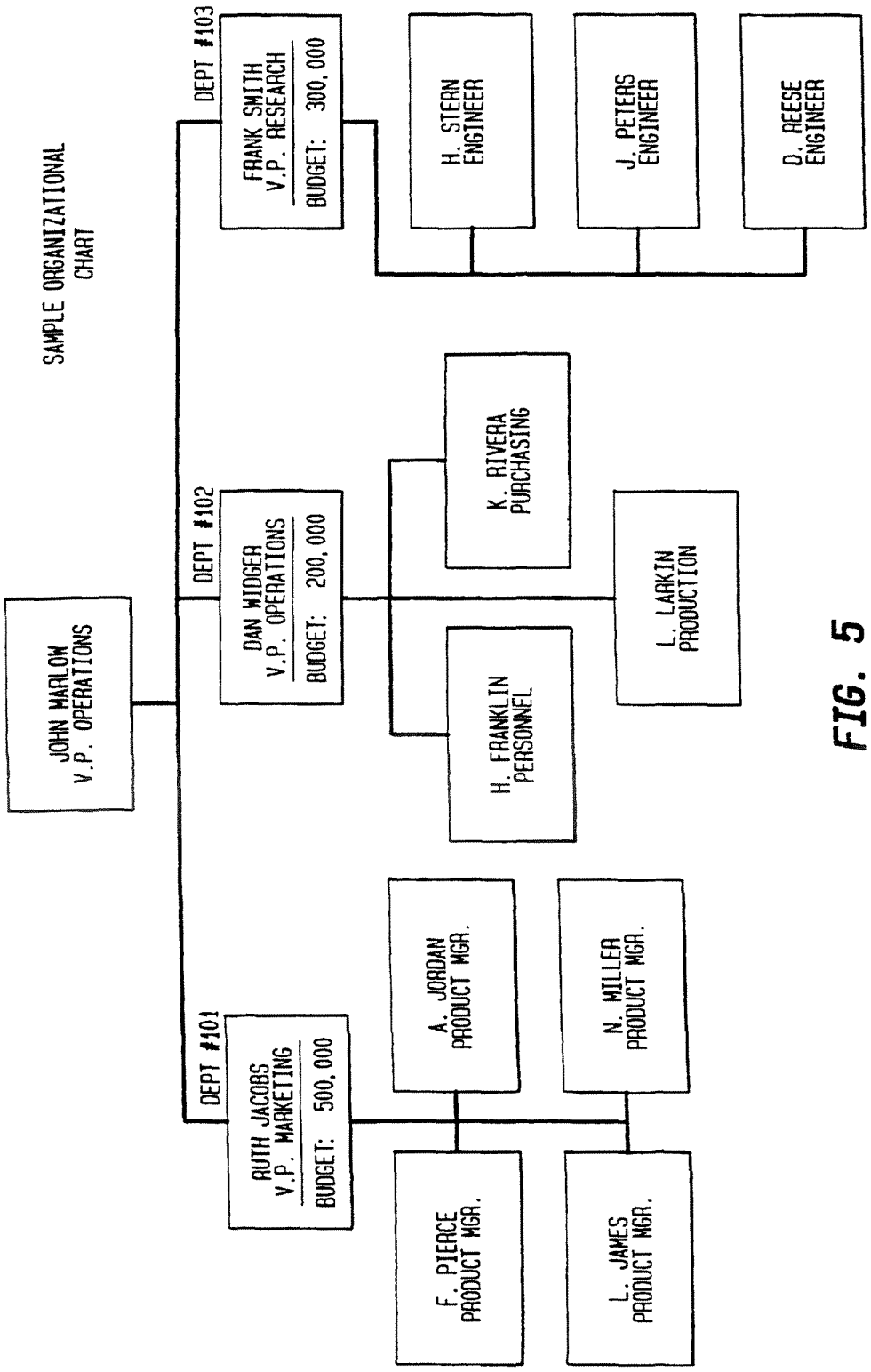

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW
V.P. OPERATIONS
DEPT #103
FRANK SMITH
V.P. RESEARCH
BUDGET: 300,000
H. STERN
ENGINEER
J. PETERS
ENGINEER
D. REESE
ENGINEER
DEPT #102
DAN WIDGER
V.P. OPERATIONS
BUDGET: 200,000
K. RIVERA
PURCHASING
L. LARKIN
PRODUCTION
H. FRANKLIN
PERSONNEL
DEPT #101
RUTH JACOBS
V.P. MARKETING
BUDGET: 500,000
A. JORDAN
PRODUCT MGR.
N. MILLER
PRODUCT MGR.
F. PIERCE
PRODUCT MGR.
L. JAMES
PRODUCT MGR.

FIG. 5

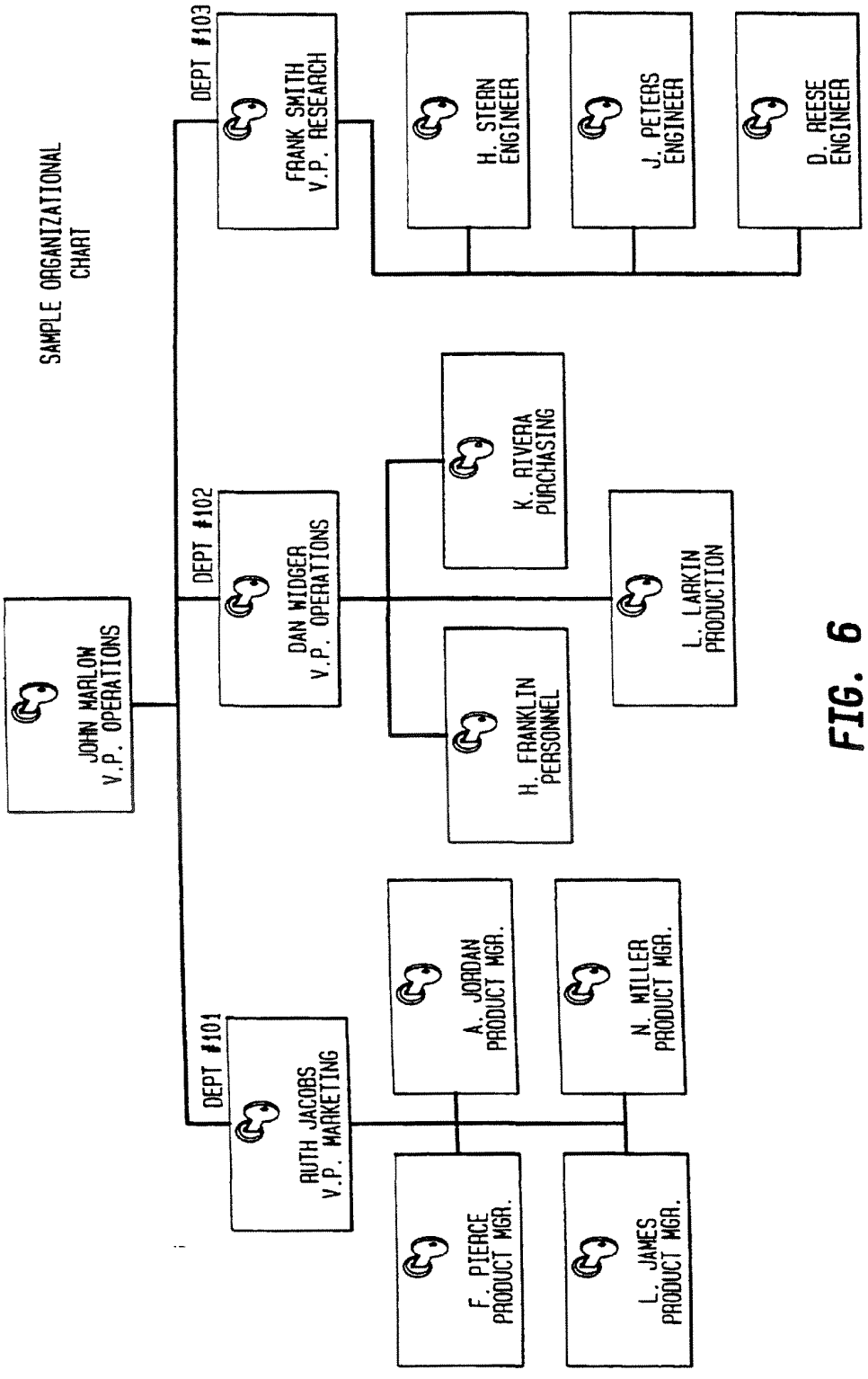

SAMPLE ORGANIZATIONAL CHART
JOHN MARLOW V.P. OPERATIONS
DEPT #103
FRANK SMITH V.P. RESEARCH
H. STERN ENGINEER
J. PETERS ENGINEER
D. REESE ENGINEER
DEPT #102
DAN WIDGER V.P. OPERATIONS
K. RIVERA PURCHASING
L. LARKIN PRODUCTION
H. FRANKLIN PERSONNEL
DEPT #101
RUTH JACOBS V.P. MARKETING
A. JORDAN PRODUCT MGR.
N. MILLER PRODUCT MGR.
F. PIERCE PRODUCT MGR.
L. JAMES PRODUCT MGR.

FIG. 6

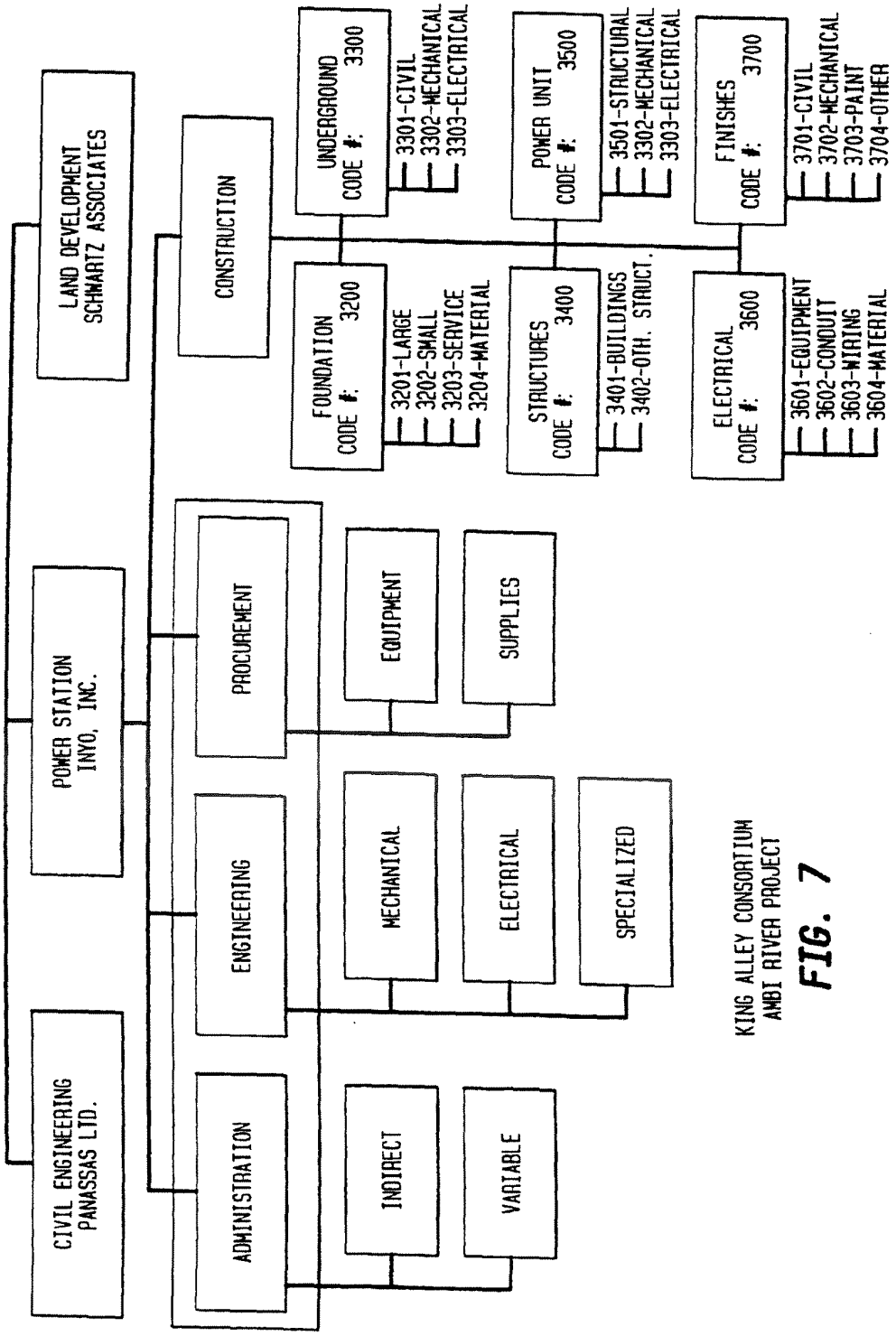
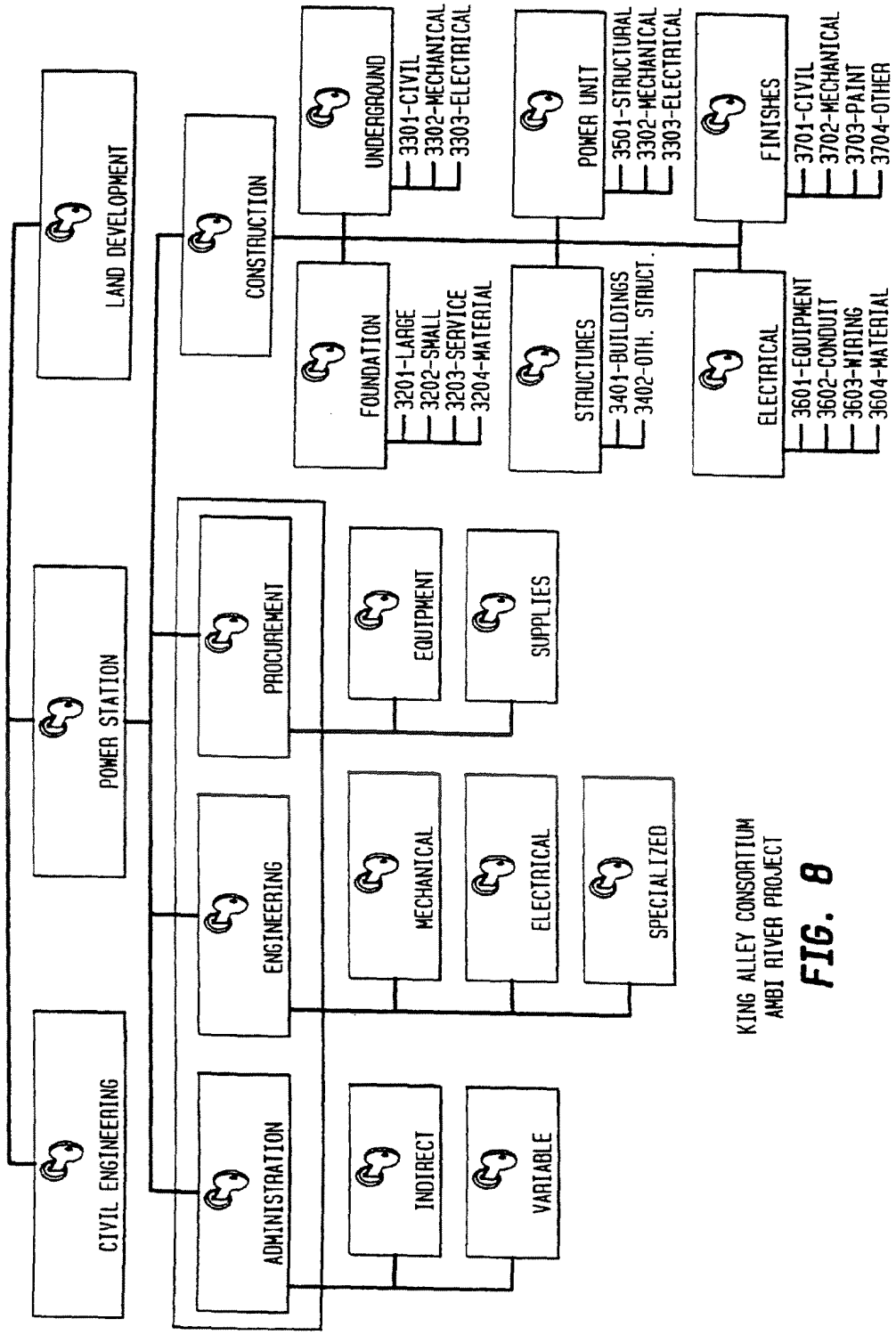

KING ALLEY CONSORTIUM
AMBI RIVER PROJECT
CIVIL ENGINEERING
PANASSAS LTD.
POWER STATION
INYO, INC.
LAND DEVELOPMENT
SCHWARTZ ASSOCIATES
ADMINISTRATION
ENGINEERING
PROCUREMENT
CONSTRUCTION
INDIRECT
VARIABLE
MECHANICAL
ELECTRICAL
SPECIALIZED
EQUIPMENT
SUPPLIES
FOUNDATION
CODE #: 3200
3201-LARGE
3202-SMALL
3203-SERVICE
3204-MATERIAL
UNDERGROUND
CODE #: 3300
3301-CIVIL
3302-MECHANICAL
3303-ELECTRICAL
STRUCTURES
CODE #: 3400
3401-BUILDINGS
3402-OTH. STRUCT.
POWER UNIT
CODE #: 3500
3501-STRUCTURAL
3302-MECHANICAL
3303-ELECTRICAL
ELECTRICAL
CODE #: 3600
3601-EQUIPMENT
3602-CONDUIT
3603-WIRING
3604-MATERIAL
FINISHES
CODE #: 3700
3701-CIVIL
3702-MECHANICAL
3703-PAINT
3704-OTHER

FIG. 7



KING ALLEY CONSORTIUM
AMBI RIVER PROJECT

FIG. 8

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

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

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

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;

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-

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;

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.

* * * * *

CERTIFICATE OF SERVICE

I hereby certify that on the 3rd day of August, 2015, 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 12,240 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.