Paper No. 1

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

APPLE INC.
Petitioner,

v.

CONTENTGUARD HOLDINGS, LLC
Patent Owner.

Patent No. 6,963,859
Issued: November 8, 2005
Filed: January 16, 2003
Inventors: Mark J. Stefik, Peter L. Pirolli
Title: Content Rendering Repository

Inter Partes Review No. IPR2015-00440

Petition for *Inter Partes* Review of
U.S. Patent No. 6,963,859

Table of Contents

I. Introduction

A. Certification the '859 Patent May Be Contested by Petitioner

Petitioner certifies U.S. Patent No. 6,963,859 (Ex. 1001) (the '859 patent) is available for *inter partes* review. Petitioner also certifies it is not barred or estopped from requesting *inter partes* review of the claims of the '859 patent. Neither Petitioner, nor any party in privity with Petitioner, has filed a civil action challenging the validity of any claim of the '859 patent. The '859 patent has not been the subject of a prior *inter partes* review by Petitioner or a privy of Petitioner. Petitioner also certifies this petition for *inter partes* review is timely filed as it is filed within one year of December 23, 2013, the date the Court in the Eastern District of Texas issued a summons on the basis of Patent Owner's complaint for patent infringement of the '859 patent against Apple in civil action No. 2:2013-cv-01112. See Ex. 1068. Because the date of this petition is less than one year from December 23, 2013, this petition complies with 35 U.S.C. § 315(b).

B. Fee for Inter Partes Review (§ 42.15(a))

The Director is authorized to charge the fee specified by 37 CFR § 42.15(a) to Deposit Account No. 50-1597.

C. Mandatory Notices (37 CFR § 42.8(b))

1. Real Party in Interest (§ 42.8(b)(1))

The real party in interest of this petition pursuant to § 42.8(b)(1) is Apple

Inc. ("Apple") located at One Infinite Loop, Cupertino, CA 95014.

2. Other Proceedings (§ 42.8(b)(2))

The '859 patent is at issue in Civ. No. 2:2013-cv-01112 (E.D. Tx.), 2:14-cv-00061 (E.D. Tx.) and 3:14-cv-00498 (N.D. Cal.). IPR2015-00441 and -00442 filed concurrently also involve the '859 patent. Each petition advances unique grounds based different primary references (*i.e.*, Kahn (Ex. 1018) and Rosen (Ex. 1020)). The Kahn petition presents grounds based on the obvious combination of its server-based secure content distribution scheme with software-based client systems taught by Linn (Ex. 1058). The Rosen petition presents grounds based on the obvious adaptation of its hardware-based client and server DRM systems to use the usage rights language technique of Hartrick (Ex. 1021). The present petition advances grounds based on the obvious adaptation of the flexible ABYSS copy protection system to use public-key encryption, alone or as implemented using Hartrick's usage rights language technique. Each petition presents a unique correlation of the claims to the prior art, and warrants independent institution of trial. In addition, Petitioner is challenging four other patents related to the '859 patent, and has streamlined its challenges by asserting the same three prior art combinations against all of the patents. Petitioner respectfully requests the Board institute each petition, as each presents distinct and non-redundant grounds.

3. Lead and Backup Lead Counsel (§ 42.8(b)(3))

Lead Counsel	Backup Lead Counsel
Jeffrey P. Kushan (Reg. No. 43,401)	Michael R. Franzinger (46,335)
jkushan@sidley.com	mfranzinger@sidley.com
(202) 736-8914	(202) 736-8583

4. Service Information (§ 42.8(b)(4))

Service on Petitioner may be made by e-mail (iprnotices@sidley.com), or by mail or hand delivery to: Sidley Austin LLP, 1501 K Street, N.W., Washington, D.C. 20005. The fax number for lead and backup lead counsel is (202) 736-8711.

5. Proof of Service (§§ 42.6(e) and 42.105(a))

Proof of service of this petition is provided in **Attachment A**.

II. Identification of Claims Being Challenged (§ 42.104(b))

Claims **1, 3, 5, 8, 13, 15, 16, 19-21, 24, 29, 31, 33, 36, 40, 42, 43, 46-48, 51, 58, 60, 62, 65, 69, 71, 72, 75-77**, and **81** of the '859 patent ("the challenged claims") are unpatentable as being obvious under 35 U.S.C. § 103. Specifically:

(i) The challenged claims are obvious based on ABYSS: A Trusted Architecture for Software Protection by Liam Comerford and Steve White ("ABYSS") (Ex. 1016) in view of *Cryptography and Data Security* by Dorothy Denning ("Denning") (Ex. 1017); and

(ii) The challenged claims are obvious based on ABYSS (Ex. 1016) and Denning (Ex. 1017) further in view of EP0567800 to Hartrick et al. ("Hartrick") (Ex. 1021).

The evidence relied upon in support of this petition is listed in **Attachment B**.

III. Relevant Information Concerning the '859 Patent

A. Effective Filing Date

The '859 patent issued from Appl. No. 10/345,390, filed on January 16, 2003. Through continuation and divisional applications, the '390 application claims the benefit back to U.S. Application No. 08/344,760, filed on November 23, 1994. Solely for the purpose of this proceeding, Petitioner is treating the effective filing date of the '859 patent as not earlier than **November 23, 1994**.

B. Overview of the '859 Patent

1. The '859 Specification

The '859 patent is a member of a family of patents issued to Stefik et al., including, *inter alia*, U.S. Patent Nos. 7,523,072; 7,269,576; 7,225,160; 8,370,956; and 8,393,007 (also subject to IPRs filed by Petitioner). Although designated "continuation" applications, the Stefik patents have non-identical disclosures with each other, and with prior filed applications. Ex. 1013 at ¶ B2-3.

The '859 patent describes a scheme that allows authors to control the use and distribution of digital "content" (*e.g.*, software, books, video). Ex. 1001 at 6:33-65, Fig. 1. In this scheme, each digital copy of the content is encapsulated into a "digital work"—*i.e.*, a digital file to which "usage rights" are permanently attached and which is stored in and distributed via "repositories." *Id.* at 6:11-12

("A key feature of the present invention is that usage rights are permanently 'attached' to the digital work."), 6:33-65, 49:52-54, 50:8-12. The "usage rights" data permanently attached to each copy of a digital work is used by repositories to control the manner in which a digital work can be used or distributed, as well as to specify any conditions on exercising those manners of use. *Id.* at 6:3-9, 6:11-12, 51:7-10. "Repositories" are "trusted devices" that store individual copies of digital works and regulate any copying or distribution of those copies by enforcing the usage rights attached to each work. *Id.* at 3:58-62, 6:17-21.

The '859 patent explains that digital works with "attached usage rights" and "repositories" are the two central features that provides its systems with "distinct advantages over prior systems." *Id.* at 6:22-32, 1:11-12. Indeed, the '859 patent shows that its scheme requires these two features to ***always*** be present – if either is removed, the author loses "control" of the work and the "content genie" can get "out of the bottle":

> [I]n the prior art, payment of fees are primarily for the initial access. In such approaches, ***once a work has been read, computational control over that copy is gone***. Metaphorically, "***the content genie is out of the bottle*** and no more fees can be billed." In contrast, ***the present invention never separates the fee descriptions from the work. Thus, the digital work genie only moves from one trusted bottle (repository) to another***, and all uses of copies are potentially controlled and billable.

Id. at 6:22-32, 6:13-16 ("the usage rights . . . assigned by a creator and subsequent distributor ***will always remain with a digital work***"), 6:17-18 ("The enforcement elements of the present invention are embodied in repositories."). Thus, if content is separated from its usage rights or if a digital work is transferred to an untrusted system (*i.e.*, a system that is not a repository) nothing prevents a user from gaining unregulated access to the content, making unlimited copies of it or distributing unprotected content to others. *Id.*; Ex. 1013 at ¶ B11. In short, the scheme fails.

The '859 patent indicates that authors can use these two "key elements" to create different types of distribution schemes (*e.g.*, consumer redistribution, "commercial libraries," or paid distribution chains) by specifying particular manners of using and distributing each work, and setting conditions on such manners of use, in the usage rights attached to the digital work. Ex. 1001 at 42:50; 43:1, 25; 44:1, 24, 53; 45:37. The '859 patent asserts that being able to attach usage rights to digital works and to permit redistribution of content but only via repositories distinguishes its scheme over the prior art, particularly simple schemes which provide encrypted content then regulate the delivery and use of decryption keys to access that content. *Id.* at 1:55-57, 2:54-60, 3:30-34, 3:45-52. Contrary to the '859 patent's contentions, use of trusted systems and usage rights to regulate the distribution and use of digital content, including after delivery to a user, was well known before 1994. Ex. 1013 at ¶ A62-90, A91-97, B13-15; *see* Ex. 1016 at

38; Ex. 1027 at 1137; Ex. 1026 at 4; Ex. 1029 at 3:22-25; Ex. 1019 at Fig. 2.

2. Representative Claim

Independent claims 1, 29 and 58 define a system, method and computer readable medium, but recite the same operative steps. *See* Ex. 1013 at ¶ C143-47. Claim 1 is representative, defining a "*rendering system adapted for use in a distributed system for managing use of content*, *[and] operative to rendering content in accordance with usage rights associated with the content*." The rendering system (*e.g.*, a computer) is part of a "*distributed system*" for distributing and rendering content (*e.g.*, network of computers that each can render and distribute content), and it is configured to render content (*e.g.*, execute software) and enforce "*usage rights associated with the content*." *See* Ex. 1001 at Figs. 2 & 4b, 8:17-20 ("A computer system may [be] a 'multi-function' device . . .").

Claim 1 specifies the rendering system has two functional components: "[i] ***a rendering device*** configured to render the content; and [ii] a ***distributed repository*** *coupled to said rendering device* and ***including a requester mode of operation and server mode of operation*** …" *Id.* at 51:21-25 (emphasis added). The '859 patent explains these two functional components can reside in a single device, such as a computer that renders protected software. *Id.* at 29:44-47 ("the requester and server may be the same device and the transactions . . . would be entirely internal."), 8:17-20.

Claim 1 then specifies three requirements for the "*distributed repository*":

> [i] *the server mode of operation [] enforce[s] usage rights associated with the content and permit the rendering device to render the content in accordance with a manner of use specified by the usage rights,*
> [ii] *the requester mode of operation [] request[s] access to content from another distributed repository, and*
> [iii] *said distributed repository is operative to receive a request to render the content and permit the content to be rendered only if a manner of use specified in the request corresponds to a manner of use specified in the usage rights.*

Id. at 51:25-38. A parallel set of claims depends from each of claims 1, 29, and 58, and specify limitations on the content, the usage rights, and the rendering system.

C. The Person of Ordinary Skill in the Art

A person of ordinary skill in the relevant art in November 1994 would have been a person with a Bachelor of Science degree in Electrical Engineering, Computer Engineering, or Computer Science and either a Master of Science in one of those fields or two or more years of industry experience with designing and/or building cryptographic hardware or software. Ex. 1013 at ¶ A61.

D. Prosecution History

The '859 patent issued from Appl. No. 10/345,390, filed January 16, 2003. During examination, all pending claims were rejected over the prior art. Ex. 1002 at 185 (Sept. 8, 2004). In response, Patent Owner amended then pending claim 1,

replacing "repository" with "distributed repository" that had both a "requester mode of operation and server mode of operation." *Id.* at 217 (Mar. 8, 2005). The Examiner allowed the amended claims, concluding the prior art did not show a repository interacting with a distribution network. Ex. 1002 at 266-67 (May 6, 2005). As explained below, the Examiner was incorrect. Indeed, the premise that repositories used within DRM schemes would not be distributed ignores the central purpose of DRM schemes –the controlled distribution of ***digital*** content.

E. Claim Construction

The '859 patent claims an effective filing date of November 23, 1994, and will expire on February 9, 2015. Consequently, if trial is instituted, the claims must be construed using their ordinary and customary meaning, as would be understood by a person of ordinary skill in the art, at the time of the invention, in light of the language of the claims, the specification, and the prosecution history of record. IPR2014-00247, Order (Paper 20) at 2-3 (citing *Phillips v. AWH Corp.*, 415 F.3d 1303, 1313-17 (Fed. Cir. 2005) (*en banc*)); *see In re Rambus*, 694 F.3d 42, 46 (Fed. Cir. 2012) ("the Board's review of the claims of an expired patent is similar to that of a district court's review").

The '859 patent provides a glossary defining many terms used in the claims. Ex. 1001 at 49:30-51:14 (hereinafter "glossary"). These explicit definitions control the construction of the defined terms. In this respect, the Board construed

the meaning of certain terms of the '859 patent in IPR2013-00137 based on the definitions in the glossary. Those constructions were based on substantial evidence and should be used here.

1. "Content" / "Digital Works"

Each independent claim recites the term "***content.***" "Content" is "*data (i.e., 'digital information') representing a 'digital work' whose use or distribution is regulated by repositories using usage rights permanently attached to that instance of the digital work.*" Ex. 1013 at ¶ B114. This conclusion is compelled by the '859 glossary. First, the glossary defines "content" as "the digital information (*i.e.*, raw bits) representing a ***digital work***." Ex. 1001 at 49:51-53, 5:64-66 ("Herein the terms 'digital work', 'work' and 'content' refer to any work that has been reduced to a digital representation."). The glossary then defines a "***digital work***" as being "any encapsulated digital information" to which "usage rights and fees" are attached.[1] *Id.* at 50:8-12, 10:45-46 ("It is ***fundamental*** to the present invention that the ***usage rights are treated as part of the digital work***."); 6:22-32, 8:46-54. The

[1] Fees are a component of a usage right. *See* § III.E.3; Ex. 1001 at 18:19-25 ("Each usage right must specify a right code [and may] specify conditions which must be satisfied before the right can be exercised. These conditions are copy count, control, time, access and ***fee conditions***." (emphasis added)).

'859 patent makes clear this "attachment" of the usage rights to the digital work cannot be altered – it is "permanent." *Id*. at 6:11-16; *see* Ex. 1013 at ¶ B108-09.

As explained in § III.B, the '859 patent specifies that usage rights are permanently attached to each copy of the "content" when a creator (*e.g.*, an author) reduces a work to digital form and places it into a repository. Ex. 1001 at 6:35-38 ("[A] creator creates a digital work . . . [and] then determine[s] the appropriate usage rights and fees, attach[es] them to the digital work, and store[s] them in Repository 1…"), Fig. 1. The '859 patent also emphasizes that it is fundamental to its scheme that content is only used by or distributed between repositories in the form of a "digital work" to which usage rights are permanently attached – if content is disseminated without usage rights, the "genie is out of the bottle" and the scheme fails. *Cf. id.* at 6:14-16 ("the ***usage rights*** … ***will always remain with a digital work***") (emphasis added), 6:25-32; Ex. 1013 at ¶ B10, B111.

The '859 patent thus only describes "content" being stored, copied or distributed in the form of a "digital work," and describes no processes where content is transmitted by or via devices that are not repositories. Ex. 1013 at ¶¶ B112-13; Ex. 1001 at 6:40-61 ("[t]he digital work remains securely in Repository 1 until a request for access is received … If access is granted, repository 1 transmits the digital work to repository 2"). The '859 patent repeatedly emphasizes each instance of "digital information" representing a work

only exists within a "digital work" to which usage rights are permanently attached, and each copy of a "digital work" can only reside in and be transferred between repositories. Ex. 1001 at 6:29-32, 6:35-38, Fig. 1; Ex. 1013 at ¶ B113-15.

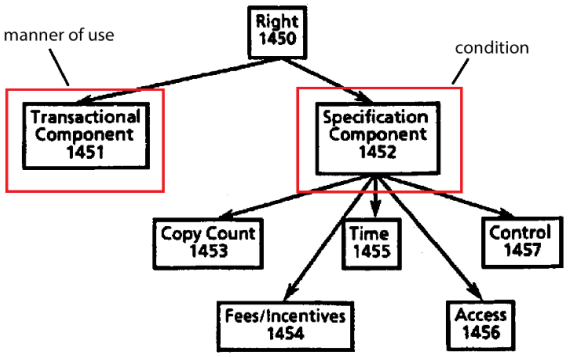
2. "Usage Rights" and "Manner of Use"

Each independent claim recites the terms "***usage rights***" and "***manner of use***." The glossary defines "***usage rights***" as "[a] language for defining the ***manner*** in which a digital work may be used or distributed, ***as well as any conditions*** on which use or distribution is premised." Ex. 1001 at 51:7-10 (emphasis added). A "***usage right***" thus is a data construct that consists of "*statements in a language defining one manner in which the digital work may be used or distributed, and, optionally, one or more conditions on which use or distribution is premised, and is permanently attached to one digital work.*" Ex. 1013 at ¶ B60. Multiple usage rights may be attached to one digital work (each specifying a particular manner of use and associated conditions), but the converse is not true – one "usage right" cannot be attached to multiple digital works. Ex. 1013 at ¶ B54-55. Also, "[u]sage rights . . . are attached to [a] digital work." Ex. 1001 at 50:62-63, 6:16-17 ("[a] key feature of the present invention is that usage rights are permanently 'attached' to the digital work").

The '859 patent shows each usage right will include a "***manner of use***" (*i.e.*, one of a number of actions defined in the usage rights language that can be taken

with the digital work) and may include, as a distinct element within the usage right, one or more optional "***conditions***" governing that manner of use. *Id.* at 18:19-25, 51:7-10, 17:27-36. The '859 patent shows each "statement" in the usage rights language specifies one "right code" and optionally may specify one or more "conditions" on exercising that right code. *Id.* at 18:19-25. Consistent with this, Figure 14 depicts each "usage right" (1450) having one "transactional component" (1451) and a "specifications component" (1452) with multiple "conditions" (1453-1457) within it. *Id.* at 17:24-25. The "transactional ***component*** 1451 corresponds to a particular way in which a digital work may be used or distributed," (*id.* at 17:27-30), while the "specifications ***components*** 1452 are used to specify conditions which must be satisfied prior to the right being exercised or to designate various transaction related parameters," (*id.* at 17:33-36). *See id.* at 9:47-52, 17:30-38, 10:1-29.

Figure 14



The '859 patent shows a particular "manner of use" within a "usage right" is implemented as a value (the "right code") representing the action that can be taken with the digital work. As it explains, "[e]ach usage right ***must specify a right code***," (*id.* at 18:21-23), which corresponds to the particular "use or distribution

privilege[] embodied by the right," such as "COPY or PRINT," (*id.* at 17:25-28). *See id.* at 9:47-54 ("each right will have a right code field 1001 and status information field 1002"), 18:21-25, Fig. 10; 16:63-17:3, 17:6-8, 18:35-41 ("the grammar provides a catalog of possible rights that can be associated with parts of digital works"), 18:13-14 ("The ***set of rights*** attached to a digital work ***define how that digital work may be transferred, used, performed or played***.") (emphases added); Ex. 1013 at ¶ B38. Examples of "manners of use" (actions) include "play," "copy," "transfer," "backup," and "install." Ex. 1001 at 18:35-19:52. The "manner of use" specified by the value of the right code in any particular instance of a digital work is enforced by the repository when it reads that right code value and interprets it using the usage rights language. *Id.* at 16:66-17:8.

"Conditions" are implemented as distinct elements relative to the "manner of use" within the usage right. *Id.* at 6:5-9. Thus, the same element within the "usage right" cannot be both a "manner of use" and a "condition" (*e.g.*, the "right code" does not specify a condition). Ex. 1013 at ¶ B43, B49. The '859 patent also explains each condition is designed to limit a particular manner of use, rather than applying more generally to a digital work as a whole. *See* Ex. 1001 at 20:18-20 ("The copy-count is associated ***with each right***, as opposed to there being just a single copy-count for the digital work."). Also, while a "manner of use" is a mandatory element of a "usage right," a "condition" is not. *Id.* at 18:21-25 ("Each

right may also ***optionally*** specify conditions which must be satisfied before the right can be exercised."); *see* fn. 1.

"Usage rights" also are distinct from data within a digital work or provided independently that is used by repositories to enforce usage rights. As the '859 patent explains, "[t]he enforcement elements of the present invention are embodied in repositories." *Id.* at 6:17-18. Thus, data constructs such as encryption keys, digital certificates, and digital signatures that are used by the repository to control access to a digital work or data within are not "usage rights." IPR2013-00138, Paper 54 at 38:18-39:1. Rather, per the '859 patent, a repository interprets information within "usage rights" to determine ***which*** actions can be taken with the work, while enforcement elements (*e.g.*, the decryption keys, digital certificates, digital signatures) are used to provide access to information in the digital work by controlling its decryption and authenticating access requests and instances of digital works. Ex. 1001 at 6:3-7 (usage rights "***define how a digital work can be used*** and ***if it can be further distributed***." (emphases added)), 16:66-17:3.

Thus, each "***usage right***" will take the form of data permanently attached to a particular copy of a digital work that specifies (i) one "manner of use" out of a defined set of manners of use defined in the usage rights language that may be taken with that copy of the digital work and (ii) optionally includes one or more conditions associated with exercising that manner of use. These data contain the

"statements" of the language or data encoding such statements, and they are distinct from data that is used by a repository to enforce the "usage right" (*e.g.*, encryption keys, digital certificates, digital signatures). Ex. 1013 at ¶ B61.

Finally, a "***manner of use***" is "*a defined way of using or distributing a digital work (for example, PLAY, COPY, or PRINT), as distinct from conditions which must be satisfied before that way of using or distributing the digital work is allowed.*" Ex. 1013 at ¶ B59.

3. "Repository"

Each challenged claim recites a "***repository.***" A "repository" is "*a trusted system in that it maintains physical, communications and behavioral integrity, and it supports usage rights by controlling which actions can be taken with a digital work by using information in that work's usage rights*." Ex. 1001 at 6:17-18, 50:48-52; *see* IPR2013-00137, Paper 58 at 8 (July 1, 2014) ("ZTE Final Dec."). In earlier proceedings involving the '859 patent, the Board adopted the glossary definition for the term "repository" – "a trusted system in that it maintains physical, communications and behavioral integrity" and it supports usage rights. *Id.* at 8; Ex. 1001 at 50:48-52. The Board explained a repository must possess three types of integrity: (i) "***physical integrity***" (*i.e.*, a repository "'prevent[s] access to information by a non-trusted system.'") (ZTE Final Dec. at 9-10); (ii) "***communications integrity***" (*i.e.*, a repository "'only communicates with other

devices that are able to present proof that they are trusted systems, for example, by using security measures such as encryption, exchange of digital certificates, and nonces.'") (*id.* at 10-11); and (iii) "***behavioral integrity***" (*i.e.*, a repository "'requir[es] software to include a digital certificate in order to be installed in the repository.'") (*id.* at 11; Ex. 1001 at 11:62-12:5, 12:24-33, 12:40-50).

Certain additional clarifications are warranted regarding "digital certificates" and "supporting usage rights." For example, both "communications" and "behavioral" integrity require use of "digital certificates." Ex. 1001 at 7:33-35 ("Identification certificates are the means by which a repository is identified as 'trustworthy'."), 12:24-29; ZTE Final Dec. at 10-11 (a repository will only transfer content to devices that are "able to present proof that they are trusted systems"). The glossary defines a digital certificate (*i.e.*, "Identification (Digital) Certificate") as "[a] ***signed*** digital message that attests to the identity of the possessor." Ex. 1001 at 50:16-19 (emphasis added); *see id.* at 26:54-57. This is consistent with how a person of ordinary skill in the art would understand that term: a digital certificate is a digitally signed message that binds an identity to a public key. Ex. 1017 at 170; Ex. 1013 at ¶ B69-71. Digital certificates are issued by a trusted certificate authority, whose digital signature on the certificate attests that the entity identified in the certificate registered the identified public key. *Id.* at ¶ B71.

The '859 patent explains, to ensure behavioral integrity, software

applications distributed as "content" must be "certified" by distributing a digital certificate with the software. Ex. 1001 at 12:40-43 (the digital certificates provide "proof of [the software's] certification"), 12:47-50 ("If the ***digital certificate*** cannot be found ***in the digital work . . .*** then the software cannot be installed."); 41:28-42:6 (showing a "digital certificate" used to certify "a digital work as ***runnable software*** on a repository"); *see* ZTE Final Dec. at 16-18. A person of ordinary skill would understand that "certifying" a digital work necessarily involves using a digital signature created by the entity identified in the certificate (*e.g.*, by encrypting a digest of the software file with its private key). Ex. 1013 at ¶ B73; Ex. 1001 at 41:45-57 (explaining a repository validates an encrypted "tamper-checking code," *i.e.*, a digital signature, associated with the software). This conventional digital signature technique is what enables a repository receiving the software to ***verify*** the ***source*** (the entity identified in the certificate) and ***integrity*** (that the software had not been modified since being signed). Ex. 1013 at ¶ B74-76. Otherwise, any entity could simply copy and append a digital certificate to the software, making it appear as if the software was certified. *Id.*

Each repository must be able to support usage rights – "[t]he enforcement elements of the present invention are embodied in repositories." Ex. 1001 at 6:17-18. To do this, repositories interpret the usage right to determine what actions can be taken with the digital work. *Id.* at 16:66-17:3 ("Usage rights statements are

interpreted by repositories and are used to determine what transactions can be [] carried out for a digital work and [the] parameters for those transactions."); Ex. 1013 at ¶ B62, B88. Without repositories, usage rights are incapable of controlling the use or distribution of the digital work; this is why every illustration in the '859 patent of digital works being transferred or rendered shows a repository storing, transferring, or rendering the digital work. *Id.* at ¶ B53, B64; Ex. 1001 at 6:35-65.

4. "Distributed repository"

Each independent claim recites the term "***distributed repository***," and provides that the distributed repository is part of a "rendering system." As explained below, the proper construction of the term "distributed repository" is "*a 'repository' capable of operating in server and requester mode, which communicates with other repositories over a network using digital certificates*." Ex. 1013 at ¶ B96. A "distributed" repository also must possess all of the attributes of repositories, as explained in § III.E.3.

The preamble of each claim states the "rendering system" is for use in "*a distributed system for managing use of content*." Ex. 1001 at 51:15-20. Each claim also specifies that "distributed repositories" must be capable of operating in both "requester" and "server" modes of operation, and each must be able to communicate with "another distributed repository." *See id.* at 51:31-32. Based on this description, a person of ordinary skill in the art would understand that a

"distributed repository" is a type of a repository that must be able to interact with other repositories over a network, *i.e.*, in a distributed system. Ex. 1013 at ¶ B91.

The term "distributed repository" appears nowhere in the '859 patent disclosure; it was added by an amendment to the claims during prosecution. Ex. 1002 at 217 (Mar. 8, 2005); *see* § III.D. However, the specification describes an embodiment ("repository 201") as having a requester and server mode of operation and as part of a distributed system of repositories. Ex. 1001 at 7:4-17, Fig. 2; Ex. 1013 at ¶ B92. The '859 patent explains that interactions in the distributed system occur using a "repository transaction protocol," which are communications involving exchanges of digital certificates. *See* Ex. 1001 at 7:12-17 ("[T]he repository 201 may communicate with a plurality of other repositories . . . . Communication between repositories occurs utilizing a repository transaction protocol 205."), 25:38-47, 26:30-67.

5. "Rendering system" and "video system"

Claim 1 specifies that a "**rendering system**" comprises "a rendering device" coupled to "a distributed repository." The glossary defines "rendering system" as "[t]he combination of a rendering repository and a rendering device." *Id.* at 50:43-45, 50:37-42 (describing "typical" features of a "rendering repository"). The proper construction of a "rendering system" is "*a device that can render content and is coupled to a repository*." Ex. 1013 at ¶ B130-35.

"Rendering" refers to the process of displaying a digital work in a human-perceptible or usable form (*e.g.*, printing text onto paper, displaying an image on a video display, or executing a software program). Ex. 1013 at ¶ B140. The glossary lists several examples of rendering systems, including "printing systems, display systems, general purpose computer systems, video systems or audio systems." Ex. 1001 at 50:45-47. The '859 patent differentiates rendering from enforcement actions performed on digital works, such as decryption of data or checking of digital certificates or signatures. *Id.* at 3:58-62 (explaining "rendering device" is "render[s] the content" while a "repository" is "coupled to the rendering device and operative to enforce usage rights associated with the content"), 6:17-18.

Claims 13, 40, and 69 recite the term "**video system**." A "video system" is a type of rendering system that is able to display video content, such as a computer system that can play videos. *Id.* at 50:45-47; *see id.* at Figs. 4a-4b, 8:16-31.

6. "Rendering device" and "execution device"

Each independent claim recites the term "**rendering device**," and claims 16 and 72 recite the term "**execution device**." Properly construed, a "**rendering device**" is "*a device that can render at least one type of content (e.g., software or a video)*," (*id.* at 51:21); an "**execution device**" is "*a device that can render software*," (*id.* at 52:5-9). The '859 patent explains a computer can act as "a 'multifunction' device" that contains both an execution device and a display

device. *Id.* at 8:16-31, Fig. 4b. Thus, both terms encompass a computer system that can render content in a digital work. Ex. 1013 at ¶¶ B136-B141.

7. Means for storing the content

Claims 3, 31 and 60 specify the claimed repository comprises "**means for storing the content**." Claim 62 specifies the means for storing content further comprises "means for storing content after rendering." The Board construed the first means element "to be content storage 1204, such as an optical disk, a magnetooptical storage system, a magnetic tape, and a solid state storage." IPR2013-00137, Paper 17 at 15-16. The disclosure also shows a repository with a "storage system 1207," which comprises different storage media such as solid state storage or an optical disk. Ex. 1001 at 13:20-22, 13:37-47, Fig. 12. The proper construction is as the Board found, but it may also include "storage system 1207."

8. Means for communication with a master repository

Claim 24 specifies the rendering system of claim 1 comprises "**means for communicating with a master repository for obtaining an identification certificate for the repository**." In IPR2013-00137, the Board construed this means "to be the external interface 1206, which provides a signal connection with another device." *See* Paper 17 at 16. This conclusion is supported by the '859 disclosure. *See* Ex. 1001 at 13:20-22, 13:52-59.

IV. Analysis of the Patentability of the Claims of the '859 Patent

The '859 patent has three independent claims, each of which specifies the same operative steps. *See* § III.B.2. Each claim specifies a "*rendering system*" (*e.g.*, a computer) that is part of a "*distributed system*" and that is configured to render digital content (*e.g.*, execute software or play a video) and to enforce the usage rights attached to that content. The rendering system comprises a "rendering device" coupled to a "distributed repository," which is configured to retrieve works from other repositories and to enforce the usage rights attached to each work.

ABYSS describes techniques for attaching a range of usage rights to software applications, and for using trusted systems to distribute and render the software and to enforce the usage rights. ABYSS teaches its systems can use public key encryption, but leaves many implementation details to the knowledge of a person of ordinary skill. To the extent those techniques are not described in ABYSS, they were well-known to those of skill in the art, as described in Denning. Each reference presented in this petition is prior art under § 102(b), and Petitioner requests the Board to institute on these grounds to avoid any dispute over priority.

A. Overview of ABYSS (Ex. 1016)

ABYSS (Ex. 1016) was published in 1987, and is prior art under 35 U.S.C. § 102(b). Ex. 1016 at 38. The operation and features of the systems described in ABYSS are explained in greater detail in Ex. 1013 at ¶ C1-170.

1. The ABYSS Scheme

ABYSS describes a scheme for "solving the software protection problem" that allows software vendors "to obtain technical enforcement of virtually any terms and conditions of the sale of their software, including . . . rental software." Ex. 1016 at 38, 39. ABYSS explains its processes enable "the trusted execution of application software" by leveraging the ABYSS computer architecture, in which a "protected processor" is integrated into every computer. *Id.* at 39. The "protected processor" is a tamperproof module that contains a processor that decrypts and executes software, and a secure memory for storing encryption keys and other data. *Id.* The protected processor's operation is managed by a "supervisor process," which implements secure protocols for installing and executing software and for communicating with other protected processors. *Id.* at 40, 43; *see id.* at 44-45.

A software vendor uses the protected processor's built-in functionality to control the distribution and use of software by including a separate file called a "Right-To-Execute" that defines the usage conditions "under which an application may execute." *Id.* at 39, 40. A vendor distributes its software in encrypted form along with a Right-To-Execute containing a decryption key, data specifying how the software can be used, and any conditions on its use. *Id.* at 39, 40, 48-49.

2. The Right-To-Execute

ABYSS shows its techniques allow a software vendor to obtain technical enforcement of a "very wide range" of terms and conditions of use. Ex. 1016 at

49. It explains "***any terms and conditions that can be embodied in data in the protected processor, as a part of the Right-To-Execute, can be enforced***." *Id.* (emphasis added), 40 (a Right-To-Execute contains "[i]nformation about how [it] may be used by supervisor software. ***This allows the software vendor to control what the supervisor may do*** with the Right-To-Execute." (emphasis added)).

The Right-To-Execute allows the vendor to "***control the entire range of actions that can be taken with respect to the application***," such as granting the right to execute, transfer, or loan software. *Id.* at 40 (emphasis added), 44-45 ("[T]he protected processor checks that this Right-To-Execute may be used to ***execute*** application programs."), 40 ("the software vendor may choose ***not to allow the Right-To-Execute to be transferred***"), 49 (describing a "***lending*** library") (emphases added); *see id.* at 38, 39, 43, 50. A vendor optionally may specify ***conditions*** governing use of the software. A Right-To-Execute can include limitations such as an "expiration time and date" and "a maximum number of allowed uses," which will be enforced by the protected processor. *Id.* at 49. These features "open[] up a wide variety of new capabilities in computing systems." *Id.*; *see* Ex. 1013 at ¶ C6, C41, C74-75, C147-48, *see also id.* at C34-45.

The ABYSS scheme enables software vendors to regulate the use and distribution of individual copies of software, including by "specify[ing] conditions . . . ***for a particular copy of an application***." *Id.* at 49 (emphasis added); Ex. 1013

at ¶ C37-41. One way a Right-To-Execute can be bound to a copy of an application is by encrypting each copy of that application with a different key.

> Once the part *P* of the application to be protected is complete, it is encrypted under an application key *A*, chosen by the software vendor, to form $E_A(P)$. ***The application key may be unique*** to each application, or even ***to each copy of each application.***

Ex. 1016 at 43 (emphases added). Because the decryption key is embedded in the Right-To-Execute, each copy of the application needs its own unique Right-To-Execute. *Id.* at 40. Each Right-To-Execute includes "identifying information" or "reference information" – an identifier that uniquely associates it with a particular copy of the application. Ex. 1016 at 44; Ex. 1013 at ¶ C41, C150.

Each Right-To-Execute is encrypted with a "supervisor key," which is a secret key associated with a particular protected processor or group of processors. Ex. 1016 at 44, 48 ("Supervisor keys may be unique to a given protected processor, common to all processors made by a given manufacturer, or anything in between."), 49; Ex. 1013 at ¶ C44-45, C23-29. By using different supervisor keys to encrypt each Right-To-Execute, a software vendor can limit transfer and redistribution of its software by creating Rights-To-Execute that can be "loaded only onto [particular] processors," such as those that have "the trusted manufacturer's supervisor key." Ex. 1016 at 48. This feature allow a software vendor to create unique types of Rights-To-Execute, such as one that is specific to

a particular processor or to a particular group of processors. *Id.* at 49 ("Some software packages . . . are distributed to only a select group of users. . . . Targeted distribution can be achieved with unique supervisor keys."); Ex. 1013 at ¶ C25-27.

3. The Protected Processor

The protected processor is a trusted component of the system and enforces the terms and conditions of use in the Right-To-Execute. Ex. 1016 at 39 ("The conditions under which an application may execute are . . . [in] a Right-To-Execute. ***These conditions are enforced by the protected processor.***"), 40, 43. The protected processor checks the terms and conditions of use before permitting a user to execute, copy, or transfer the software. *Id.* at 40, 38, 39, 43, 50. For example, when a user requests to execute or transfer an installed software application, the protected processor uses the reference information to find the Right-To-Execute, checks the terms and conditions of use in it, and then permits or denies the operation. *Id.* at 43-45; Ex. 1013 at ¶ C55-58. In addition to using the protected processor's built-in functionality, a vendor also can allow the software application to help enforce or even modify the terms and conditions of use. Ex. 1016 at 40.

The features of protected processors "enable[] the ***trusted*** execution of applications." *Id.* at 50; *id.* ("[T]he protected processor serves as a ***trustworthy*** agent . . .") (emphases added). ABYSS explains:

> ***The protected processor is a logically, physically, and procedurally secure unit.*** It is ***logically secure***, in that an application cannot

> directly access the supervisor process . . .[or other applications] to violate their protection. It is ***physically secure . . .*** in that it is contained in a tamper-resistant package . . . . It is ***procedurally secure*** in that the services which move information . . . into and out of the protected processor cannot be used to subvert the protection.

Id. at 39 (emphases added). The secure operations of the protected processor are managed by a "supervisor process," which "is responsible for ensuring the logical and procedural security of the protected processor" and can be customized by the manufacturer to provide different features and services. *See id.* at 40, 43 ("Each [supervisor] service could use [different techniques] . . . as authorization mechanisms."), 49; Ex. 1013 at ¶ C18-22.

The protected processor's design prevents "untrusted" devices from accessing unprotected data – software is distributed in encrypted form and the decryption key is securely stored in the Right-to-Execute. *See* Ex. 1016 at 39 (processor contains "secure memory for storage of Rights-To-Execute"), 40 (the software "cannot be modified directly . . ."), 50 ("strong cryptographic methods are used to manage both [software and Rights-To-Execute]"). Because the protected processor can be trusted, software executed on it can be trusted as well – users cannot modify the software to subvert the security protections. *Id.* at 39.

4. Installing and Executing Software

To install software, the Right-To-Execute must be loaded into the protected

processor. *Id.* at 44. The protected processor will not store the Right-To-Execute unless it receives "authorization from externally-supplied ***authorization processes***." *Id.* at 39, 44, 40. Only a computer system with an authorized protected processor can access or execute the protected software or software components. *Id.* at 44; Ex. 1013 at ¶ C11-17, C48-54, C121.

ABYSS explains "[t]here are many possible authorization processes." Ex. 1016 at 44. For example, where it is possible to establish "a direct communication channel between two protected processors" (*id.* at 42), authorization can be obtained from a remote protected processor through a "cryptographic transaction" with a remote processor (*id.* at 44). Examples of common "cryptographic transactions" include receiving a digitally signed receipt from the vendor, registering a public key, or receiving a digital certificate. Ex. 1013 at ¶ C48-52. If direct communications are not possible, authorization can be provided using "tokens" distributed on a smart card or cartridge. Ex. 1016 at 42.

After the Right-To-Execute is installed into the protected processor, the software can be executed. *Id.* at 44, 43 ("Executing Protected Software[:] . . . The user simply types: wondercalc and the application executes as usual."). When the user requests to execute the application, the protected processor uses "reference information" (*e.g.*, an identifier) to find the Right-To-Execute and validates the terms and conditions of use in it. *Id.* at 44-45. If execution is permitted, "the

protected processor use[s] [message authentication or manipulation codes ("MACs")] to authenticate the application before it is executed" and confirm it has not been modified. *Id.* at 43, 44-45 ("If the decryption . . . yields a valid ***message authentication or manipulation detection code***, execution . . is begun. If not, *P* is purged . . ."). A "MAC" is formed by making a digest or hash of the software and encrypting the hash with a supervisor key. Ex. 1013 at ¶ A134-37. The public-key version of this technique is a digital signature. *Id.* at ¶ C163-67, C97-101.

An application's integrity is checked each time a user requests to execute the application. Ex. 1016 at 39 ("Software . . . cannot be modified arbitrarily by the user"), 40, 44-45. A person of skill would have understood the integrity of the application should be validated both before installing or executing the software – it would be illogical to install software that could never be executed (*e.g.*, because it was corrupted or had been modified). Ex. 1013 at ¶ C59-60. In 1994, the integrity of software was commonly validated before installation. *Id.* at ¶ A98-100.

5. Communications Between Protected Processors

Protected processors communicate with each other using a cryptographically secure protocol. Ex. 1016 at 42, 43, 50. Two protected processors will establish a secure channel, mediated by a session key, which can be used "to move both software and Rights-To-Execute between protected processors." *Id.* at 50. "Common supervisor keys [] allow secure two-way communication between any

two protected processors." *Id.* at 49. Protected processors use supervisor keys to authenticate (*i.e.*, identify) each other because only a protected processor can decrypt messages that have been encrypted with a supervisor key. *Id.* at 42 (session will be initiated only if they "possess an encryption/decryption key in common"), 47, 49; Ex. 1013 at ¶ C44-45, C121.

These secure communication capabilities allow a protected processor to acquire "both software and Rights-To-Execute . . . on demand" from a software vendor (*i.e.*, in response to a user's request), by "download from host systems," or by download from a "software lending library." Ex. 1016 at 50, 49; *see* § IV.D.2; Ex. 1013 at ¶ C70-74. They also allow for network-based authorization processes. Ex. 1013 at ¶ C48-53; *see* § IV.A.4. In each case, the protected processor communicates with other protected processors over a network to "create[] a trusted path for distributing Rights-To-Execute" and software. Ex. 1016 at 44.

ABYSS presents a simplified description of secure communication processes and acknowledges that "[a]n actual implementation would include communication handshaking, verification of the completion of each step of the service, and error recovery." Ex. 1016 at 43. Thus, a person of ordinary skill would understand that standard cryptographic security features should be used to implement systems based on ABYSS, such as handshaking and authentication techniques, as well as use of timestamps (nonces) to prevent replay attacks. Ex.

1013 at ¶ C70, C159-65; Ex. 1017 at 170-75, 178-79. ABYSS makes clear that while it describes examples that use symmetric "supervisor" keys, its systems "can employ either a symmetric cryptosystem ***or a public key cryptosystem***." Ex. 1016 at 43. While the implementation details are not explicitly discussed in ABYSS, such details, were well-known in the art. Ex. 1013 at ¶ C159-70, C81-94.

B. Overview of Denning (Ex. 1017)

Denning is a textbook published in 1983 and is cited in ABYSS. Ex. 1016 at 40, 51. It describes basic, well-known cryptographic techniques including, *inter alia*, symmetric and public key encryption (*id.* at 7-14, 145-155); digital signatures (*id.* at 14, 109); techniques for creating, distributing, and using digital certificates (*e.g.*, to distribute keys, authenticate identity, and initiate secure communications) (*id.* at 15-16, 170); and authenticating and securing communications (*id.* at 170, 178-179). Ex. 1013 at ¶ A137-48, C81-94, C159-70.

1. Use of Digital Certificates in Communications

Denning illustrates use of digital certificates. In one example, a certificate for an entity "*A*" is a message that contains *A*'s identity, *A*'s public key, and a timestamp, and that is digitally signed by a trusted authority. Ex. 1017 at 170 ("Upon registering a public [key] E_A with the system, a user *A* receives a signed certificate from *S* containing E_A."). The trusted authority's signature authenticates that *A* has registered that public key; thus, *A* can prove its identity by providing the

digital certificate and by demonstrating it can decrypt messages that are encrypted with the public key in the certificate. *Id*.; Ex. 1013 at ¶ A144-49, C76-89.

Digital certificates were (and are) routinely used to initiate secure and authenticated communications sessions between two entities or devices. Ex. 1017 at 178-79 (describing a certificate exchanging protocol – "[L]et C_A and C_B be *A*'s and *B*'s certificates [], signed by *S*. *A* and *B* can exchange a key *K* with the following protocol . . ."); Ex. 1013 at ¶ A144-53, C81-89. One entity initiates a session by sending its digital certificate, a timestamp, and some additional data to the other entity. Ex. 1017 at 178-79. Certificates are exchanged, and each entity uses the public key in the other entity's certificate to encrypt a message with parameters for the session, including a symmetric session key for subsequent communications. *Id.* ("A public-key cryptosystem can also be used to exchange session keys []."); Ex. 1013 at ¶ A146-52. This process can include additional security features, such as a handshake. Ex. 1017 at 175, 179.

2. Use of Digital Signatures

Denning also describes well-known applications of digital signatures. *Id.* at 14; Ex. 1013 at ¶ A98-100, C96-103. To digitally sign a message, a compressed digest (a cryptographic hash) of the message is encrypted with the signing entity's private key and appended to the message. *Id.* at 109; Ex. 1013 at ¶ A142. The digital signature can be verified by decrypting the signature with the entity's

authenticated public key (*i.e.*, the public key in a digital certificate), re-computing the digest, and then comparing the two. Ex. 1017 at 14, 109. If the values match, that verifies the message's authenticity (*i.e.*, it was signed by entity identified in the certificate) and integrity (*i.e.*, it has not been modified since being signed). *Id.* Digital signatures have many uses (*e.g.*, signing a software product allows a user to validate the software's source and integrity). *Id.* at 15; Ex. 1013 at ¶ C96-103; C165-67. Denning describes digital signatures being used in software protection:

> The first involves distributing network software to the individual nodes of a network. If the software is signed, ***the nodes can check the validity of the software before execution.*** The second involves running privileged programs in operating systems. ***The system (preferably hardware) could refuse to execute any program in privileged mode that is not properly signed by a program verifier, making it impossible for someone to substitute a program that could run in privileged mode and wreak havoc in the system.*** This idea could be extended to all programs, with ***the system refusing to execute any code that has not been signed by some authority***.

Ex. 1017 at 15-16. Based on Denning, a person of ordinary skill would have understood that, to enable users to validate the digital signature, the software would need to include the digital certificate (including the public key) of the entity certifying the software (*e.g.*, the "program verifier"). Ex. 1013 at ¶ C164-68.

C. Overview of Hartrick (Ex. 1021)

European Patent No. EP 0 567 800 to Hartrick ("Hartrick") (Ex. 1021) was

published on November 3, 1993, and is prior art to the '859 claims under 35 U.S.C. § 102(b). Ex. 1021; Ex. 1013 at ¶ B206. The operations and techniques described in Hartrick are explained below and in Ex. 1013 at ¶ E70-103.

Hartrick describes a technique for controlling the use and distribution of "softcopy" (*i.e.*, digital) books. Ex. 1021 at 1:3-8, 5:19-38. Each softcopy book is a "structured document" that has been formatted using SGML, a markup language that uses "tags" to describe each "element" (*e.g.*, component) of a document. *Id.* at 4:25-36, 5:3-8, 5:42-47; *see id.* at 1:28-39, 4:38-42, 9:41-49, 11:37, 12:45. A softcopy book reading program uses these tags to, *inter alia*, format the output displayed to a user. *Id.* at 1:12-16, 2:21-28. Hartrick shows books can be distributed individually, or in a group. *See id.* at 13:50-54.

Hartrick shows use of specialized SGML tags (called security tags and royalty tags) to enable authors to specify whether the book can be reproduced and any associated fees for doing so. *Id.* at 3:12-22, 3:28-32, 5:48-52. These tags allow authors to control end-users' ability to, *e.g.*, "print[] pages of a structured document," "writ[e] [] a structured document into bulk storage," and transmit via "telecommunication [] soft copies of a structured document." *Id.* at 3:53-58; 5:24-26, 30-32, 35-37; 10:1-14. The tags can be placed "within the structured document text . . . or in a royalty payment information file which accompanies the book." *Id.* at 5:48-52. A royalty payment program on the user's computer enforces

compliance with the tags. *Id.* at 6:9-22, Fig. 1.

Hartrick describes "security labels" as "special structured document tag[s]" that specify how the book can be used (*id.* at 3:28-32, 3:48-50), and the royalty payment program interprets these security tags to prohibit certain operations on the book. *Id.* at 3:53-4:9. For example, if a book contains the "Do Not Copy" tag, the royalty payment program will permit a user to view the book, but inhibit printing, copying, or transmitting it. *Id.*; Ex. 1013 at ¶ E80-82. If a user requests to print a book that contains this tag, the requested "printing operation[s] will be aborted in response to the [tag's] presence." Ex. 1021 at 3:53-57. A book can also contain other security tags, such as a "Do Not Distribute" tag, which prevents the document from being transmitted. *Id.* at 3:28-32, 4:4-9; Ex. 1013 at ¶ E83-85.

Each book can also include royalty tags that allow an author or publisher to specify different fees that must be paid to reproduce the book or any part of the book (*e.g.*, a different fee for each chapter). Ex. 1021 at 5:42-47, 10:1-14, 10:41-54, 11:46-12:36. In certain embodiments, Hartrick depicts a general "royalty" tag as regulating any type of "reproduction" of a book (*e.g.*, printing, copying, ***and*** transmitting). *Id.* at 11:46-12:36. Hartrick also shows separately regulating individual actions a user can take with a softcopy book (*e.g.*, print, copy, transmit) using the royalty tags. *Id.* at 5:24-26, 30-32, 35-37; 6:9-22; 10:7-18. Hartrick also separately defines in the claims methods for controlling different rights, specifying

a publisher can separately authorize each of printing, copying, or transmitting a book. *Id.* at 21:56-22:3, 22:7-10 (print); 22:34-38, 22:42-45 (copy); 23:13-14, 23:21-24 (transmit). A person of ordinary skill would have understood the Hartrick description as teaching that different royalty tags can be used to specify fees for the different operations being taken by a user with a softcopy book, as it uses a distinct tag for each action (*e.g.*, printing, copying, and transmitting) regarding the book (*e.g.*, "[print-royalty]" and "[transmit-royalty]"). Ex. 1013 at ¶¶ E85-87, E90-98; *see* Ex. 1021 at 5:19-38, 6:9-22, 10:7-18.

D. A Person of Ordinary Skill Would Implement Systems in ABYSS Based on Denning Using Public-Key Techniques

1. Implementing Public-Key Techniques

ABYSS teaches its systems can be implemented using either of the two standard cryptographic systems (*i.e.*, "either a symmetric [] or a public key cryptosystem"). Ex. 1016 at 43. A person of ordinary skill would be familiar with public-key encryption ("PKI") and would consider Denning (cited in ABYSS) for guidance. Ex. 1013 at ¶ C80-82. Doing so would lead that person to incorporate public-key encryption into at least four different elements of the ABYSS system. *Id.* at ¶ C80; *see id.* at C76-104, A143-53.

First, based on ABYSS's own teachings, a person of ordinary skill would recognize PKI techniques could be used both to identify other protected processors and to encrypt communications between the processors. *See* § IV.B.1; Ex. 1016 at

42; Ex. 1013 at ¶ C83-89, C105, C111-22, C158-70. A skilled artisan implementing PKI in ABYSS would have considered it routine to configure the protected processors to initiate communications by exchanging digital certificates. Ex. 1013 at ¶ C83-85, C87-89, C118-22, A148-52; Ex. 1017 at 170, 175, 178-179; *see* § IV.B.1. When so configured, protected processors would identify other trusted devices (*i.e.*, other protected processors) by validating the other device's digital certificate, and ensuring it was issued by a particular trusted authority (*e.g.*, the processor's manufacturer or a software vendor). Ex. 1013 at ¶ C84, C119-22. Such a configuration would be a standard use of PKI technology taught in the contemporaneous literature. Ex. 1013 at ¶ C83, C92-94, C119, A132, A153-59.

Second, a person of ordinary skill would recognize public-key encryption could be used to protect the integrity of software. Ex. 1013 at ¶ C95-103. Instead of using a symmetric MAC to protect software integrity, it would have been routine to distribute the software with a digital signature and digital certificate, as described in Denning. *See* § IV.B.2; Ex. 1017 at 15-16; Ex. 1013 at ¶ C63-64, C95-103, A137-41. A person of skill would have understood doing so would offer benefits, such as simplifying the authentication and validation of the software. Ex. 1013 at ¶ C101. That person also would have understood the same technique should be applied to system software, such as updates to the supervisor software that operated the protected processor, as this would improve system integrity. *Id.*

at ¶ C100-03, A98-100; *see* Ex. 1016 at 45 (each protected processor contains a "supervisor key[] . . . used only for communicating with the [] manufacturer"). Thus, if a manufacturer distributed an update to the supervisor software, it would have been obvious to digitally sign the update and have each protected processor validate the signature before installing it. Ex. 1013 at ¶ 102-03.

Third, when implementing the supervisor keys using PKI techniques, a person of ordinary skill could have used public-key encryption to encrypt the Rights-To-Execute. *See* § IV.A.2; Ex. 1013 at ¶ C85-89. Before transmitting a Right-To-Execute to another protected processor, the Right-To-Execute would be encrypted with the public key in the recipient processor's digital certificate. *Id.* at ¶ C85-89, C92. This would allow the symmetric supervisor keys to be replaced by public-private key pairs and digital certificates. Ex. 1013 at ¶ C81-89.

Fourth, a person of skill could have incorporated PKI techniques into the authorization processes. Ex. 1013 at ¶ C104-05. ABYSS shows authorization can be obtained from a remote protected processor through a "cryptographic transaction," (Ex. 1016 at 44), and one example of such a transaction that a person of skill would be familiar with is receiving a message identifying the processor that was digitally signed by the vendor (*e.g.*, a receipt). Ex. 1013 at ¶ C49-52, C114-17. In that configuration, the protected processor would verify it had received a valid receipt before installing or executing an application. *Id.* at ¶ C104-05.

Each of these would have been an obvious implementation choice for a person of ordinary skill based on the guidance within ABYSS and Denning. *Id.* at ¶ C76-80. Although ABYSS suggests that symmetric encryption may be a more cost-effective solution for the "low-end of the market," this statement was made in 1987 when ABYSS was published; developments after 1987 but before 1994 would have alleviated those early concerns. *Id.* at ¶ C90-94; Ex. 1016 at 38, 39, 43. For example, in 1988, the ITU released the X.509 certificate authority standard and in 1993 the IETF released RFC 1422, which defined a certificate authority scheme as part of the Privacy Enhanced Mail suite. Ex. 1013 at ¶ A154-59; *see* Ex. 1047; Ex. 1048. Both standards were widely adopted before 1994, lowering the cost and simplifying the process of using PKI and certificate authorities. Ex. 1013 at ¶ C90-94. And, at least one pre-1994 publication suggested use of public key encryption to implement ABYSS schemes. *Id.*; *see* Ex. 1057 at 132-32, 151.

2. The Operation of the Implemented System

ABYSS describes the following "software lending library" example:

> A ***single copy of an application can be kept on a file server***. The protected processor associated with this server ***can store a Right-To-Execute*** which enables up to some fixed number of copies to execute, for instance. ***When a user requests the application, a single Right-To-Execute is also transferred to the workstation, and the server's count is decremented.*** The transferred Right-To-Execute may require renewal every minute or so to continue to be valid. . . . ***If the***

> ***workstation does not check back with the server, the server knows that the workstation's Right-To-Execute has expired***, so the server can increment its count of Rights-To-Execute for that application.

Ex. 1016 at 49 (emphases added). It would have been obvious for a person of skill to implement this example using PKI techniques given the guidance in ABYSS and Denning. Ex. 1013 at ¶ C73, C106-26; *see* § IV.D.1.

To create the lending library, the entity operating the file server must first obtain the application and a Right-To-Execute that permits lending. Ex. 1013 at ¶ C110. ABYSS shows this being done by having the file server's protected processor initiate a secure channel with a software vendor's protected processor, and downloading the software and Right-to-Execute. *Id.*; *see* § IV.A.5; Ex. 1016 at 47, 49, 50 (using "secure two-way communication between the software vendor and the user . . . [b]oth software, and Rights-To-Execute can then be distributed . . . by download"). In a PKI implementation, the secure, authenticated channel would be initiated by exchanging digital certificates. *See* § IV.B.1; Ex. 1013 at ¶ C81-87.

ABYSS explains that its scheme can retain control over distribution of individual copies of software by setting parameters within the Right-To-Execute. Ex. 1016 at 39, 40, 48-50. To implement the lending library example, a vendor could create a Right-To-Execute using the "targeted distribution" technique described in ABYSS, which limits distribution to a defined group of users. *See id.* at 49; § IV.A.2; Ex. 1013 at ¶ C25-26, C111-17. When implemented with PKI, the

vendor could restrict distribution to processors that had a digital certificate issued by a particular authority for the group (*e.g.*, the software vendor). *See* § IV.D.1; Ex. 1013 at ¶ C83-89, C113. The vendor also could specify as a condition that the Right-To-Execute can only be installed by workstations that have obtained a digitally signed receipt from the vendor. *Id.* at ¶ C104-05, C115-17.

To be used as ABYSS describes, the file server's protected processor will install the Right-To-Execute after obtaining authorization to do so via a "cryptographic transaction" with the software vendor's protected processor (*e.g.*, to obtain a digitally signed receipt). *Id.* at ¶ 104-05; *see* § IV.A.4; Ex. 1016 at 44. The protected processor uses the vendor's digital certificate to validate the digital signature on the application, and if valid, installs the software. *See* § IV.A.4; Ex. 1016 at 45 (validating a MAC); Ex. 1017 at 15-16 ("the system [can] refus[e] to execute any code that has not been signed by some authority"); Ex. 1013 at ¶ C95-103. After the software is installed, the file server can begin lending it out. *Id.*

To join the library in a PKI implementation, a user's protected processor would register its public key with the group's certificate authority and obtain a digital certificate. Ex. 1013 at ¶ C113-17; Ex. 1016 at 49; Ex. 1017 at 170; *see* § IV.A.4. The user also needs to be authorized by obtaining a digitally signed receipt from the software vendor's protected processor. Ex. 1013 at ¶ C114-17; Ex. 1016 at 44. The user then can request from the file server a copy of the

software and a Right-To-Execute. *Id.* at 49, 50. To initiate the request, the workstation's protected processor sends its digital certificate to the file server's protected processor. Ex. 1017 at 178-79; Ex. 1013 at ¶ C118-21. The file server then validates the certificate was issued by the group's certificate authority, and if valid, establishes the connection. Ex. 1013 at ¶ C121-23; Ex. 1017 at 178-79.

ABYSS explains the lending library file server will check whether copies of the application are available for lending in response to a request. *Id.* at 49. If so, it derives a new Right-To-Execute that permits executing the application (but not transfer or loan) and has an expiration time, and it updates its copy counter. *Id.* Then, in a PKI implementation, it would encrypt the Right-To-Execute with the public key in the user's authorized certificate, and transmit the application and the Right-To-Execute to the user's protected processor. *See* § IV.A.2, D.1; Ex. 1013 at ¶ C85-87. The Right-To-Execute is installed on the workstation if the user has obtained authorization (*e.g.*, a digitally signed receipt), and the workstation's protected processor will enforce any terms and conditions of use for the application, such as an expiration time. Ex. 1013 at ¶ C86, C124-25; *see* § IV.A.2-4. When the loaned Right-To-Execute expires, the file server's protected processor automatically will increment its count of available copies because it "knows" the workstation will enforce the condition. Ex. 1016 at 49, 39 (any "conditions are enforced by the protected processor"); Ex. 1013 at ¶ C6-10, C74-75, C124-27.

E. ABYSS and Denning Teach "Usage Rights" and "Repositories"

1. ABYSS Describes "Usage Rights" Attached to "Content"

"Usage rights" are a "key feature" of the Stefik scheme and are a required element of each challenged claim. *See* § III.E.2. In the ABYSS scheme, information in a Right-To-Execute is a "*usage right.*"

First, a Right-To-Execute includes data specifying one or more actions that may be taken with a particular copy of the installed software, such as whether it can be run, transferred to a different computer, or loaned to other computers. *See* § IV.A.2; Ex. 1016 at 40 ("transfer[]"), 44-45 ("execute"), 49 ("lend[]"); *see also* Ex. 1001 at 18:35-19:52 (possible rights include "play," "copy," "transfer," "loan," and "install"); Ex. 1013 at ¶ C147-48. It can include multiple rights (*e.g.*, right to both execute and transfer software). Ex. 1016 at 40, 45. ABYSS thus shows a "usage right" specifying one or more "manners of use." Ex. 1013 at ¶ C148.

Second, a Right-To-Execute may contain data specifying "conditions" governing whether the installed software can be used or distributed such as "an expiration time and date, or a maximum number of allowed uses of the application." Ex. 1016 at 49, 40; Ex. 1013 at ¶ C74-75, C149. ABYSS thus shows a "usage right" optionally specifying conditions. Ex. 1013 at ¶ 149; *see* § IV.A.2.

Third, ABYSS explains the protected processor uses this information in the Right-To-Execute to determine the actions that can be taken with respect to the

software, and any conditions on taking those actions. *See* § IV.A.1, A.3-4; Ex. 1016 at 39, 40; Ex. 1013 at ¶ C146-47, C3-12, C40-43, C56-57, C155.

Fourth, a "Right-To-Execute" is attached to a particular copy of software in the ABYSS scheme. *See* § IV.A.2; Ex. 1016 at 43 ("The application key may be unique . . . ***to each copy*** of each application."), 49 ("A software vendor can specify the conditions . . . ***for a particular copy of an application***."); *see also id.* at 40, 42, 44-45; Ex. 1013 at ¶ C37-45, C150. Each copy of an application can be encrypted with its own key, and the Right-To-Execute will be uniquely attached to it using "identifying information." Ex. 1016 at 40, 44; Ex. 1013 at ¶ C35-40. While the Right-To-Execute is stored in a separate file from the application, ABYSS shows each time an application is transmitted over a network, its Right-To-Execute is transmitted with the application. Ex. 1016 at 49, 50; Ex. 1013 at ¶ C46-47, C68, C73. A person of ordinary skill would understand the attachment is permanent because if the Right-To-Execute containing the unique decryption key is removed, the encrypted application is useless. Ex. 1013 at ¶ C35-40, C146-50.

Finally, the data representing the "manners of use" and the "conditions" are data contained within each "Right-To-Execute," and these data represent the statements in a language for defining how the application may be used and distributed. Ex. 1013 at ¶ C146-52; *see* Ex. 1052 (CG CC Brief) at 8. Thus, ABYSS teaches "usage rights." *See* § III.E.2; Ex. 1013 at ¶ C146-52.

2. ABYSS in View of Denning Uses "Repositories"

ABYSS in view of Denning teaches use of protected processors that are "repositories" within the meaning of the '859 patent (*i.e.*, they "maintain[] physical, communications and behavioral integrity" and enforce usage rights). *See* § III.E.3; Ex. 1013 at ¶¶ C153-70

First, the ABYSS systems ***maintain physical integrity***. *See* ZTE Final Dec. at 13-14 ("prevent[s] access to information by a non-trusted system"). The ABYSS system has both physical security (*e.g.*, tamper-proof design) and prevents access to unprotected works by "untrusted" devices using a variety of encryption-based techniques. *See* § IV.A.1, A.3; Ex. 1013 at ¶ C1-7, C11-22, C156-57.

Second, a protected processor will only transfer an application and its Right-To-Execute to another trusted protected processor using encrypted channels. Ex. 1016 at 43, 47, 49. Before transmitting an application and its Right-To-Execute, a protected processor will verify that the other device is a trusted protected processor. *See* § IV.A.5; Ex. 1016 at 42, 43, 47, 49, 50. When implemented using a public key infrastructure, digital certificates are used to establish the secure communication channels. *See* § IV.A.5, B.1, D.1; Ex. 1013 at ¶ C82-89, 158. The protected processors thus ***maintain communications integrity*** as they will "only communicate with other devices that are able to present proof that they are trusted systems, e.g., by using securing measures such as encryption, exchange of digital

certificates, and nonces." ZTE Final Dec. at 14-15; Ex. 1013 at ¶ C158-62.

Third, a person of ordinary skill implementing ABYSS in view of Denning would have configured the scheme to use digital signatures and digital certificates to validate the authenticity and integrity of an application both before installing and executing it. *See* § IV.A.4, B.2, D.1; Ex. 1013 at ¶ C95-103. ABYSS in view of Denning, thus, shows use of protected processors that will ***maintain behavioral integrity*** because they "require software to include a digital certificate in order to be installed in the repository." ZTE Final Dec. at 15-16; Ex. 1013 at ¶ C163-69.

Finally, the protected processors enforce usage rights, and provide a trusted path for software distribution. *See* § IV.A.1-3; Ex. 1016 at 39 ("The conditions . . . ***are enforced by the protected processor.***"); Ex. 1013 at ¶ C2-17, C146-52, C155. Thus, particularly when implemented using PKI, systems having protected processors are "repositories." Ex. 1013 at ¶ C153-70.

F. ABYSS and Denning Render the Challenged Claims Obvious

1. Claims 1, 29 and 58

a) "rendering system adapted for use in a distributed system for managing use of content, . . . operative to rendering content in accordance with usage rights associated with the content"

ABYSS describes a method for distributing software ("*content*") with a Right-to-Execute, which is uniquely associated with the software and contains terms and conditions of use ("*usage rights*"). *See* Ex. 1016 at 39-40, 44-45, 49;

§§ IV.A.2, E.1; Ex. 1013 at ¶ C6, C34-43, C123-24. The protected processor is an apparatus that can execute software and contains a secure memory for storing key information. *See* §§ IV.A.1, 3; Ex. 1016 at 38-40, 43-45, 48-50. The protected processor also contains a supervisor process for enforcing usage rights and maintaining the logical and procedural security of the system. Ex. 1016 at 39, 43. Thus, ABYSS describes systems, methods, and software for controlling the distribution and execution of software. *Id*. Ex. 1013 at ¶ C174.

A protected processor is part of a computer system, and when executing or transferring applications, each protected processor will enforce any terms and conditions of use in the Right-To-Execute (a "*rendering system*" that is "*operative to render content in accordance with usage rights*"). *See* § IV.E.1-2; Ex. 1016 at 39, 40, 42, 44-45; Ex. 1013 at ¶ C9-13, C34-45. ABYSS shows protected processors as part of "*a distributed system for controlling use of software*." *See* § IV.E.1-2; Ex. 1016 at 42, 43, 47, 49, 50; Ex. 1017 at 178-79; Ex. 1013 at ¶ C173, C68-73, C80-85. ABYSS describes, for example, communications between protected processors to transfer software and Rights-To-Execute, and to obtain authorizations during installation processes. *See* § IV.A.3-4; Ex. 1016 at 38-40, 42-45, 50; Ex. 1013 at ¶ C47-52. It also describes a software lending library in which a file server loans copies of software and Rights-To-Execute to user workstations on the network. *See* § IV.D.2; Ex. 1016 at 49; Ex. 1013 at ¶C106-26.

b) "a rendering device configured to render content"

The ABYSS computer system contains a protected processor ("*rendering device*") that can execute software ("*render content*"). Ex. 1013 at ¶ C3-22, C55-67. Thus, ABYSS teaches a system, method and medium which constitutes a "*rendering device configured to render the content*" pursuant to the '859 patent. *Id.* at ¶ C11-22, C174-75; *see* § IV.A.1, A.3-4; Ex. 1016 at 38-40, 42-45, 48-50.

c) "distributed repository coupled to said rendering device"

In a PKI implementation, a workstation with a protected processor is a "*distributed repository*" because each is a "repository" (*see* § IV.E.2), each can request access to content from another "repository," and each enforces compliance with "usage rights" in response to requests. *See* § IV.D.2; Ex. 1016 at 44, 49, 50; Ex. 1017 at 170, 175, 178-79; Ex. 1013 at ¶ C11-22, C55-67, C153-70.

Each protected processor can communicate with a network of protected processors, including ones operated by software vendors and entities operating lending libraries. Ex. 1013 at ¶ 173; Ex. 1016 at 44, 49, 50. When implemented using PKI, protected processors use digital certificates to initiate secure communications when downloading content and obtaining authorizations. *See* § IV.D.1, E.2; Ex. 1017 at 178-79; Ex. 1013 at ¶ C81-89. Each also possesses a "*requester*" and "*server*" mode, as explained below. Thus each protected processor is a "*distributed repository*." Ex. 1013 at ¶ C173.

d) having a "server mode . . . operative to enforce usage rights"

ABYSS shows each computer with a protected processor enforces the terms and conditions of use of protected applications ("*server mode of operation*"). Ex. 1013 at ¶C9-10, C174. When a user attempts to execute or transfer an application, the protected processor receives a request to access the Right-To-Execute, uses "reference information" to locate it, and then determines if the software can be executed or transferred based on data in the Right-To-Execute. Ex. 1016 at 39, 44-45; Ex. 1013 at ¶ C5-13, C34-43. If the requested use is permitted, the protected processor will permit the action; otherwise it will deny it. *See* § IV.A.2-4; Ex. 1016 at 38-40, 42-45, 48-50; Ex. 1013 at ¶ C42-43. The same process is used by a file server receiving a request to borrow an application from remote workstation. *See* § IV.E.2; Ex. 1016 at 44, 49, 50; Ex. 1013 at ¶ C106-09, C123-26, C174.

e) having a "requestor mode . . . operative to request access to content from another distributed repository"

ABYSS shows a protected processor can request a copy of an application and a Right-To-Execute from other protected processors ("*request access to content from another distributed repository*"). For example, a workstation can request an application and Right-To-Execute from a protected processor hosting a software lending library. *See* § IV.E.2; Ex. 1016 at 49, 50; Ex. 1013 at ¶ C73, C118. And, ABYSS shows that both workstations and file servers can request an

application and Right-To-Execute from a protected processor associated with the software vendor and downloaded them "on demand." Ex. 1016 at 50, 47, 49; *see* § IV.A.5; Ex. 1013 at ¶ C46-47, C68. ABYSS also shows protected processors need to request access to content by requesting authorization from a software vendor's protected processor during the installation process. *See* § IV.A.4; Ex. 1016 at 40, 42-45; Ex. 1013 at ¶ C48-53, C104-05.

f) "receiving a request to render the content and permit . . . render[ing] only if [the requested] manner of use . . . corresponds to a manner of use specified in the usage right"

After protected software has been installed, a protected processor will respond to requests to access the software, such as requests to execute or transfer it ("*receive a request to render the content*"). *See* § IV.A.3-4; Ex. 1016 at 38-40, 42-45, 50; Ex. 1013 at ¶ C6-10, C55-58. If the user requests to execute the software, execution will be permitted only if the Right-To-Execute permits execution ("*permit the content to be rendered only if a manner of use specified in the request corresponds to a manner of use specified in the usage rights*"). *Id.* Similarly, if the user requests to transfer the software, the protected processor will allow the request only if transfer is permitted (same). Ex. 1013 at ¶ C69-70; C147, C155.

Thus, implementing ABYSS using PKI as suggested by ABYSS in view of Denning would have rendered claims 1, 29, and 58 obvious to a person of ordinary skill. Ex. 1013 at ¶ 174-75.

2. Claims 3, 31, and 60 and Claims 5, 33, and 62

Claims 3, 31, and 60 depend from claims 1, 29, and 58, and specify the repository comprises "*means for storing the content*." The protected processor is a tamper-resistant module that includes a "secure memory for storage of Rights-To-Execute." Ex. 1016 at 39, 44, 48, 50; Ex. 1013 at ¶ C3-C6. ABYSS shows the encrypted application can be stored in the secure memory (*id.* at 39), or in any storage device connected to the system because the decryption key (in the Right-to-Execute) is stored in the protected processor. *Id.* at 39, 50, 40. When software is installed, it is "copied, put onto a hard disk, etc." (*i.e.*, put into memory) on the user's computer. *Id.* at 42-43, 39, 44-45; Ex. 1013 at ¶ C177-83.

Claims 5, 33, and 62 depend from claims 3, 31, and 60, and specify the "*means for storing the content*" stores "*content after rendering*." ABYSS shows that software is kept stored in memory after execution. Ex. 1013 at ¶ C11-15, C67; Ex. 1016 at 39, 40, 44-45, 50. Thus, ABYSS discloses memory that is used as a "*means for storing content*," including after rendering. Ex. 1013 at ¶ C177-83.

3. Claims 8, 36, and 65 and Claims 13, 40, and 69

Claims 8, 36, and 65 depend from claims 5, 33, and 62, and specify the content comprises "*video*." ABYSS explains it provides a "general security base" that be leveraged for other applications. Ex. 1016 at 38, 50. In 1994, it was common for skilled artisans to apply copy protection techniques to all types of

content (Ex. 1013 at ¶ A101-09, A115-19), and a person of ordinary skill would have recognized the ABYSS architecture could be used to protect content other than software, (*id.* at ¶ C134-39, C184-88). To do so, content would be distributed in encrypted form, with a Right-To-Execute containing a decryption key and terms and conditions of use. *Id.* at ¶ C139. Substituting other types of content, such as video content, would be handled in the same manner and each part of the ABYSS system would operate with no change in function. Ex. 1013 at ¶ C138-39.

Claims 13, 40, and 69 depend from claims 1, 29, and 58, and specify the rendering device comprises a "*video system*." As just explained, a person of ordinary skill would have understood that the ABYSS could be used to protect content such as video. Ex. 1013 at ¶ C134-39. A computer that can play video content is a "video system" within the meaning of the claims. *See* § III.E.5. A person of ordinary skill would have understood that the protected processor could be included in computer systems that could play videos. Ex. 1016 at 38 (protected processors can be included in a "range of computing systems"). Thus, ABYSS in view of Denning would have rendered claims 8, 36, and 65, and claims 13, 40, and 69, obvious to a person of ordinary skill. Ex 1013 at ¶ C184-89.

4. Claims 15, 42, and 71

Claims 15, 42, and 71 depend from claims 1, 29, and 58, and specify the rendering device comprises a "*computer system*" and the repository comprises

"*software executed on the computer system*." Each protected processor is installed into a computer system, a supervisor process controls its operation, and it can execute protected software. Ex. 1016 at 39; Ex. 1013 at ¶ C11-17, C18-22, C55-67, C190. Thus, ABYSS renders claims 15, 42, and 71 obvious. *Id.*

5. Claims 16, 43, and 72

Claims 16, 43, and 72 depend from claims 1, 29, and 58, and specify that an "*execution engine*" is coupled to the repository and the repository permits the engine to "*execute a computer program*" in accordance with the usage rights. As explained with respect to claims 1, 29, and 58, protected processors are installed in computer systems, execute software, and enforce terms and conditions of use. *See* § IV.A.1, F.1; Ex. 1016 at 39; Ex. 1013 at ¶ C2-17, C55-57. ABYSS thus teaches the limitations of claims 16, 43, and 72. Ex. 1013 at ¶ C193, C174-75. .

6. Claims 19, 46, and 75

Claims 19, 46, and 75 depend from claims 1, 29, and 58, and specify the "*manner of use is a manner of displaying*." ABYSS explains "any terms and conditions that can be embodied in data in the protected processor, as a part of the Right-To-Execute, can be enforced." Ex. 1016 at 49. A person of ordinary skill in the art would have found it obvious to include a manner of displaying in the Right-To-Execute. Ex. 1013 at ¶ C197. For example, for "demonstration" software intended to be distributed as a "marketing aid" (Ex. 1016 at 49), it would have

been obvious to specify in the Right-To-Execute that program output be marked with a label, such as "Demo Version" or "Call to Order," to encourage sales of the full version. *Id.*; Ex. 1013 at ¶ C75, C197. Thus, a person of skill would have found it obvious to specify a "manner of displaying" in the Right-To-Execute. *Id.*

7. Claims 20, 47, and 76

Claims 20, 47, and 76 depend from claims 1, 29, and 58, and specify that the "*manner of use is a manner of playing*." "Playing" a work includes processes such as "playing a video game, running a computer program, or displaying a document on a display." Ex. 1001 at 18:49-53. ABYSS shows the Right-To-Execute can specify whether a user can "execute" the application. *See* § IV.A.2; Ex. 1016 at 45; Ex. 1013 at ¶ C148, C197. Thus, ABYSS discloses a "manner of playing." *Id.*

8. Claims 21, 48, and 77

Claims 21, 48, and 77 depend from claims 1, 29, and 58, respectively, and specify "*the rendering device and the repository are integrated into a secure system having a secure boundary*." ABYSS explains that "[t]he protected processor is a ***logically, physically, and procedurally secure unit***. . . It is physically secure (which is indicated by the heavy box in Figure 1), in that it is contained in a tamper-resistant package. . ." Ex. 1016 at 39. The protected processor executes ("*renders*") software and it provides integrity and enforces any Rights-To-Execute ("*repository*"). *Id.* at 39, 40, 43; Ex. 1013 ¶ C11-22, C55-67,

C153-70, C199. Thus, the rendering device and the repository are integrated into a secure boundary, which teaches the limitations of claims 21, 48, and 77. *Id.*

9. Claims 24, 51, and 81

Claims 24, 51, and 81 depend from claims 1, 29, and 58, respectively, and specify the rendering system can "*communicate with a master repository for obtaining an identification certificate for the repository*." *See* § III.E.8. In the PKI configuration, a person of skill would have used a public-key cryptosystem for the supervisor keys. *See* § IV.D.1; Ex. 1013 at ¶ C77-89. In that configuration, a protected processor would obtain digital certificates by registering a public key with a certificate authority. *Id.*; Ex. 1017 at 170; *See* § IV.D.1. For example, to join a lending library, a protected processor would register a public key with the vendor, and the vendor's protected processor ("*master repository*") issues them a unique digital certificate that would be required to receive or install the Right-To-Execute. Ex. 1013 at ¶ C110-17, C204; Ex. 1016 at 43, 47-48, 49. Registering a public key with a certificate authority and obtaining a digital certificate was a well-known, conventional technique. Ex. 1013 at ¶ C203-06, A144-60; Ex. 1017 at 170. ABYSS shows protected processors are part of computer systems and can communicate over a network (Ex. 1016 at 50), which means they necessarily have access to a network interface for communicating with remote devices. Ex. 1013 at ¶ C206-07. Thus, when ABYSS is implemented using PKI, a protected processor

will use a network interface to communicate with a software vendor ("*means for . . . communicating with a master repository*"), and request and obtain a digital certificate ("*obtain[] an identification certificate*"). Ex. 1013 at ¶ C202-08. Thus, ABYSS and Denning render claims 24, 51, and 81 obvious. *Id.*

G. ABYSS (Ex. 1016) and Denning (Ex. 1017) Further in View of Hartrick (Ex. 1021) Renders the Challenged Claims Obvious

ABYSS in view of Denning would have rendered obvious each of the claimed systems and processes for the reasons set forth in § IV.E-F. Patent Owner may argue that ABYSS does not disclose use of a "usage rights language." A person of ordinary skill in the art would have considered implementation of the ABYSS systems using a "usage rights language" to have been obvious based on ABYSS and Denning in view of Hartrick.

1. A Person of Ordinary Skill Implementing ABYSS and Denning Would Have Incorporated Hartrick's Technique

ABYSS explains that each Right-To-Execute contains data specifying how the software can be used and distributed. *See* § IV.A.2; Ex. 1016 at 49 ("any terms and conditions that can be embodied in data in the protected processor . . . can be enforced"). A person of skill would have found it logical to use specialized security and royalty SGML tags, as taught in Hartrick, to specify the terms and conditions of use because it would increase the granularity of control that vendors could exercise over software and a skilled artisan would have viewed using the

tags as an obvious way to specify terms and conditions of use. Ex. 1013 at ¶ C127-33, C152. That person would have modified the protected processor's supervisor process to use the specialized SGML tags to specify control over different actions a user can take with software and to specify any conditions. *Id.* at ¶ C20-22, C152; E86, E93-98. For example, the "Do Not Copy" tag could specify that a user could only execute an application, or a "Do Not Distribute" tag could specify that a user could not transfer the application. Ex. 1013 at ¶ E80-86; *see* § IV.A.2. Hartrick's SGML tags also could be modified to reflect the various actions described in ABYSS, such as by having an "execute" tag, a "transfer" tag, and a "lend" tag. Ex. 1013 at ¶ C152, E97-98. To specify an "expiration date" or a "maximum number of uses" as shown in ABYSS, a person of skill could add an "expire" tag or a "max use" tag, analogous the "amount" tag described Hartrick. *Id.* A person of skill would have known that SGML was a convenient mechanism for specifying terms and conditions because SGML is highly configurable and it is simple to change the set of tags that are used in a particular system. Ex. 1013 at ¶ C152. Specifying a set of tags to use in a particular system is a simple and routine design choice. *Id.*

2. Claims 1, 29, and 58

ABYSS shows a Right-To-Execute contains data specifying the manner in which software can be used (*e.g.*, transfer or execute) and restrictions on that use (*e.g.*, an expiration date). *See* § IV.A.2; Ex. 1016 at 40, 45, 50 A person of skill

would have found it obvious to express those rights and restrictions using SGML statements as taught and suggested in Hartrick. Ex. 1013 at ¶ C152. ABYSS in view of Denning teaches all the other features of "usage rights." *See* § IV.E.1 (Ex. 1016 at 39, 40, 43-45, 49). Thus, claims 1, 29, and 58 would have been obvious based on ABYSS and Denning in further view of Hartrick. Ex. 1013 at ¶ C152.

3. Claims 3, 5, 8, 13, 15, 16, 19-21, 24, 31, 33, 36, 40, 42, 43, 46-48, 51, 60, 62, 65, 69, 71, 72, 75-77, and 81

Claims 3, 5, 8, 13, 15, 16, 19-20, 24, 31, 33, 36, 40, 42, 43, 46-47, 51, 60, 62, 65, 69, 71, 72, 75-76, and 81 depend from claims 1, 29, and 58. As explained in § IV.F, ABYSS implemented using PKI as taught by Denning suggest systems and processes having all of the elements of these claims, and thus, each of which would have been obvious based on ABYSS and Denning in view of Hartrick.

H. No Secondary Considerations Exist

As described above, ABYSS in view of Denning and in further view of Hartrick teaches systems and processes that render *prima facie* obvious each of the challenged claims of the '859 patent. No secondary indicia of non-obviousness exist having a nexus to the putative "invention" of the '859 patent contrary to that conclusion. Petitioner reserves its right to respond to any assertion of secondary indicia of non-obviousness advanced by Patent Owner.

I. The Relationship of ABYSS and Denning to EP 139

Various implementations of the ABYSS architecture have been described in

the prior art with varying degrees of specificity, including one shown in European Patent 0 268 139 ("EP 139") (Ex. 1028). IPR2013-00137, the Board declined to institute *inter partes* review on the basis that EP 139 anticipated certain of the '859 claims, finding the petition failed to establish that EP 139 disclosed a system having a "requester mode" of operation. Paper 17 at 22. ABYSS, in contrast, has never been considered by the Board or the Office with respect to the '859 patent, and contains a substantially different disclosure than EP 139. Ex. 1013 at ¶ A71, A73. For example, ABYSS explicitly describes the feature the Board found to be missing from EP 139: that protected processors can request software and a Right-To-Execute from another processor. Ex. 1016 at 49 ("When a user requests the application, . . .")). ABYSS describes other features, *e.g.,*: (i) protected processors enforce usage rights, (*id.* at 39, 40, 43), (ii) using public key encryption (*e.g.*, Denning), (*id.* at 43), and (iii) purchasing software over a network, (*id.* at 49).

V. Conclusion

Petitioner respectfully submits that the evidence presented in this Petition establishes a reasonable likelihood that Petitioner will prevail in establishing the challenged claims are unpatentable, and requests that Trial be instituted.

Dated: December 22, 2014

Respectfully Submitted,

/Jeffrey P. Kushan/
Jeffrey P. Kushan
Registration No. 43,401

Sidley Austin LLP
1501 K Street NW
Washington, DC 20005
jkushan@sidley.com
(202) 736-8914
Attorney for Petitioner

PETITION FOR *INTER PARTES* REVIEW

OF U.S. PATENT NO. 6,963,859

Attachment A:

Proof of Service of Petition

CERTIFICATE OF SERVICE

I hereby certify that on this 22nd day of December, 2014, a copy of this Petition for *Inter Partes* Review, Attachments and Exhibits, has been served in its entirety by Federal Express on the following address for Patent Owner:

ContentGuard Holdings, Inc.
6900 N. Dallas Parkway, Suite 850
Plano, Texas, 75024

Prosecution Counsel:

Marc S. Kaufman
Reed Smith LLP
1301 K Street, N.W.
Suite 1100 – East Tower
Washington, DC 20005

Litigation Counsel:

Samuel F. Baxter (sbaxter@mckoolsmith.com)
MCKOOL SMITH, P.C.
104 East Houston, Suite 300
Marshall, Texas 75670

Respectfully submitted,

/Jeffrey P. Kushan/
Jeffrey P. Kushan
Reg. No. 43,401
Attorney for Petitioner

PETITION FOR *INTER PARTES* REVIEW

OF U.S. PATENT NO. 6,963,859

Attachment B:

Exhibit List

Exhibit #	Reference Name
1001	U.S. Patent No. 6,963,859
1002	File History of U.S. Patent No. 6,963,859
1003	U.S. Patent No. 7,523,072
1004	File History of U.S. Patent No. 7,523,072
1005	U.S. Patent No. 8,370,956
1006	File History of U.S. Patent No. 8,370,956
1007	U.S. Patent No. 8,393,007
1008	File History of U.S. Patent No. 8,393,007
1009	U.S. Patent No. 7,269,576
1010	File History of U.S. Patent No. 7,269,576
1011	U.S. Patent No. 7,225,160
1012	File History of U.S. Patent No. 7,225,160
1013	Declaration of Alan Sherman, Ph. D. regarding patents 859, 072, 956, 007, 576
1014	Curriculum Vitae for Alan Sherman
1015	[RESERVED]
1016	S. White & L. Comerford, ABYSS: A Trusted Architecture for Software Protection (1987)
1017	D. Denning, Cryptography and Data Security (1982)
1018	U.S. Patent No. 6,135,646 to Kahn
1019	Proceedings of the Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment, Journal of the Interactive Multimedia Association Intellectual Property Project (Jan. 1, 1994)
1020	U.S. Patent No. 5,557,518 to Rosen
1021	EP 0567800 A1 to Hartrick
1022	File History for U.S. Patent No. 6, 135,646 to Kahn

Exhibit #	Reference Name
1023	U.S. Patent No. 5,629,980 to Stefik
1024	5,477,263 (O'Calleghan)
1025	U.S. Patent No. 5,260,999, issued November 9, 1993 to Wyman
1026	J. Moffett et al., SPECIFYING DISCRETIONARY ACCESS CONTROL POLICY FOR DISTRIBUTED SYSTEMS, Computer Communications, vol 13 no 9, pp 571-580 (Nov. 1990)
1027	R. Mori & M. Kawahara, Superdistribution: The Concept and Architecture, The Transactions of The IEICE, Vol. E73, No. 7, pp. 1133-1146 (July 1990)
1028	European Patent Publication 0 268 139
1029	U.S. Patent No. 5,412,717 to Fischer
1030	The Copy-Protection Wars, PC Magazine
1031	US 4,658,093 to Hellman
1032	US 5,940,504 to Griswold
1033	G. Davida et al., Defending Systems Against Viruses through Cryptographic Authentication, IEEE 1989
1034	V. Cina et al., ABYSS: A Basic Yorktown Security System: PC Asset Protection Concepts, IBM Research Report Number RC 12401 (Dec. 1986)
1035	U.S. 5,109,413 to Comerford
1036	S. Weingart, Physical Security for the μABYSS System, Proceedings of the IEEE Symposium on Security and Privacy (Apr. 27-29, 1987)
1037	FIPS 140-1
1038	4,278,837 to Best
1039	William Aspray, The Stored Program Concept, IEEE Spectrum Sept. 1990
1040	Glenn C. Reid, Thinking in Postscript, Addison-Wesley (1990)
1041	Bruce Schneier, Applied Cryptography (Chapters 1-3, 7, 12, 13, 17) (1994)
1042	Samuelson, Intellectual Property Rights for Digital Library and Hypertext Publishing Systems: An Analysis of Xanadu (Dec. 1991)
1043	Kahn & Cerf, "The Digital Library Project Volume 1: The World of Knowbots (Draft)" (1988)

Exhibit #	Reference Name
1044	Internet Dreams - Stefik foreword
1045	CNRI Digital Library Workshop
1046	Diffie-Hellman, New Directions in Crypto 1976
1047	X.509 Standard
1048	RFC 1422, Privacy Enhanced Mail (PEM)
1049	Steiner et al., Kerberos: An Authentication Service for Open Network Systems (MIT 1988)
1050	Needham & Schroeder, Using Encryption for Authentication in Large Networks of Computers (Dec. 1978)
1051	J. Saltzer & M. Schroeder, The Protection of Information in Computer Systems, Proceedings of the IEEE, Vol. 63, No. 9 (Sept. 1975)
1052	ContentGuard's Opening Claim Construction Brief, 2:13-cv-01112 (EDTX) Dkt No. 304
1053	U.S. Patent Application File History No. 09/778,001
1054	U.S. Patent Application File History Nos. 08/967,084 and 08/344,760
1055	U.S. Patent No. 4,977,594 to Shear
1056	Henry H. Perritt, Jr., "*Knowbots, Permissions Headers and Contract,"* Paper for the Conference on Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment (April 30, 1993)
1057	*Dyad: A System for Using Physically Secure Coprocessors*, J.D. Tygar and Bennett Lee, Proceedings of the Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment, Journal of the Interactive Multimedia Association Intellectual Property Project (Jan. 1, 1994)
1058	"*Copyright and Information Services in the Context of the National Research and Education Network*," R.J. Linn, Proceedings of the Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment, Journal of the Interactive Multimedia Association Intellectual Property Project (Jan. 1, 1994)
1059	"*The Strategic Environment for Protecting Multimedia*," Brian Kahin, Proceedings of the Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment, Journal of the Interactive Multimedia Association Intellectual Property Project (Jan. 1, 1994)

Exhibit #	Reference Name
1060	Amir Herzberg, Shlomit S. Pinter, "Public Protection of Software," in Advances in Cryptology: Proc. Crypto 85, Hugh C. Williams (cd.), pp. 158 (1986)
1061	S. White & L. Comerford, ABYSS: A Trusted Architecture for Software Protection (1990)
1062	T. Berners-Lee & D. Connolly, RFC 1866, Hypertext Markup Language v2.0 (Nov. 1995)
1063	[RESERVED]
1064	[RESERVED]
1065	[RESERVED]
1066	[RESERVED]
1067	[RESERVED]
1068	Summons in Case No. 2:13-CV-01112
1069	Declaration of Dr. Goodrich in support of ContentGuard's Opening Claim Construction Brief – Exhibit K of Doc. No. 304