

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC and
LENOVO (UNITED STATES) INC.,
Petitioners,

v.

ERICSSON AB,
Patent Owner.

Case No. IPR2025-00003
Patent No. 9,313,178

**PETITION FOR INTER PARTES REVIEW
UNDER 35 U.S.C. §§ 311-319 AND 37 C.F.R. § 42.1 et seq.**

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	MANDATORY NOTICES	4
	A. Real Party-In-Interest	4
	B. Related Matters.....	4
	1. United States Patent & Trademark Office	4
	2. United States Patent Trial and Appeal Board	5
	3. U.S. District Court for the Eastern District of Texas	6
	4. U.S. District Court for the Eastern District of North Carolina	6
	5. International Trade Commission (“ITC”)	6
	C. Counsel and Service Information - § 42.8(b)(3) and (4).....	6
III.	FOUNDATIONS FOR STANDING.....	7
IV.	FOUNDATIONS FOR UNPATENTABILITY	7
V.	BACKGROUND	8
	A. ’178 Patent.....	8
	B. Prosecution History	8
VI.	PERSON OF ORDINARY SKILL IN THE ART (“POSITA”).....	9
VII.	CLAIM INTERPRETATION	10
VIII.	OVERVIEW OF PRIOR ART	10
	A. Peterka	10
	B. Suzuki	13
	C. Bocharov.....	14
	D. Peterka308	17
	E. Chen.....	18
IX.	<u>GROUND 1</u> : PETERKA RENDERS CLAIMS 1-6, 10, 12, 16-18, AND 20 OBVIOUS	18
	A. Claim-by-Claim Analysis.....	19
	1. Claim 1	19

a.	[1PRE] “A method for handling secure distribution of content comprising:”	19
b.	[1A] “initiating a media playback request and receiving a playback request response;”	19
c.	[1B] “parsing content information from the playback request response, the content information including content encryption keys, content encryption key identifiers, and content encryption key expiration times;”	23
i.	Content Information.....	23
(1)	Content Encryption Keys	23
(2)	Content Encryption Key Identifiers	25
(3)	Content Encryption Key Expiration Times	26
ii.	Peterka Teaches a Playback Request Response Comprising “Content Information” for Multiple Segments	27
d.	[1C] “retrieving content and manifest files from a content delivery server;”	28
e.	[1D] “detecting content encryption key rotation boundaries between periods of use of different content encryption keys in decrypting retrieved content;”	31
f.	[1E] “issuing requests to a license server ahead of a key rotation boundary to retrieve a second content encryption key to be used after a content encryption key rotation boundary is reached; and”	34
g.	[1F] “applying the second key for content decryption after the key rotation boundary is reached.”	36
2.	Claim 2: “The method of claim 1, wherein the content encryption keys and content encryption key identifiers returned in the playback request response include the content encryption key and associated identifier currently being applied as well as a future content encryption key and associated identifier yet to be applied.”	37
3.	Claim 3: “The method of claim 1, wherein the content encryption key expiration returned in the playback request	

response is expressed as an expected minimum interval between periodic key rotation.”	38
4. Claim 4: “The method of claim 3, further comprising: timing the prefetching of a next un-retrieved content encryption key based on an expected expiration of the content encryption key currently being used.”	41
5. Claim 5: “The method of claim 4, further comprising: prefetching a next un-retrieved key a fixed duration before the expected expiration of the content encryption key currently being used.”	41
6. Claim 6: “The method of claim 4, further comprising: prefetching a next un-retrieved key within a period of time after the expected expiration of the content encryption key currently being applied, the period of time beginning at the expected expiration of the content encryption key currently being used, and ending at a duration calculated as a fixed percentage of a fixed periodic content encryption key expiration interval.”	43
7. Claim 10: “The method of claim 1, wherein license server communications are secured and authenticated using a selected one of Secure Sockets Layer and a token-based technique using a token encrypted with a symmetric key.”	45
8. Claim 12: “The method of claim 1, wherein actual content encryption key rotation boundaries are detected based on real-time in-band notifications.”	48
9. Claim 16	48
a. [16PRE] “A computerized device operable as a client for handling secure distribution of content, comprising:”	48
b. [16A] “memory operative to store computer program instructions; one or more processors; input/output interface circuitry; and interconnect circuitry coupling the memory, processors and input/output interface circuitry together,”	49
c. [16B]-[16G].....	49
10. Claims 17-18, and 20	50
X. <u>GROUND 2: PETERKA IN VIEW OF SUZUKI RENDERS CLAIMS 1-6, 10, 12, 16-18, AND 20 OBVIOUS</u>	50

A. Peterka-Suzuki.....	51
B. Claim-by-Claim Analysis.....	54
1. Claims 1-2, 10, 12, 16-18, and 20	54
2. Claims 3-6	55
XI. <u>GROUND 3: PETERKA IN VIEW OF BOCHAROV RENDERS</u> CLAIMS 1-6, 10, 12, 14, 16-18, AND 20 OBVIOUS.....	57
A. Peterka-Bocharov	57
B. Claim-by-Claim Analysis.....	60
1. Claims 1-6, 10, 12, 16-18, and 20	60
2. Claims 12 and 20.....	62
3. Claim 14: “The method of claim 12, wherein content encryption key rotation boundaries are detected based on notifications present in the content manifest file, the content manifest file being selected from an m3u8 file, an IIS Smooth Streaming manifest fle, a DASH MPD file and a proprietary manifest file.”	64
XII. <u>GROUND 4: PETERKA IN VIEW OF SUZUKI AND BOCHAROV</u> RENDERS CLAIMS 1-6, 10, 12, 14, 16-18, AND 20 OBVIOUS	66
A. Peterka-Suzuki-Bocharov.....	66
B. Claim-by-Claim Analysis.....	67
1. Claims 1-6, 10, 12, 14, 16-18, and 20	67
XIII. <u>GROUND 5: PETERKA IN VIEW OF BOCHAROV, PETERKA308,</u> AND CHEN RENDERS CLAIMS 7-8 AND 19 OBVIOUS	69
A. Peterka-Bocharov-Peterka308-Chen	69
B. Claim-by-Claim Analysis.....	72
1. Claims 7 and 19: “The [method of claim 1 / computerized device of claim 16] wherein subsequent content encryption key identifiers are predictable based on a predetermined known progression.”	72
2. Claim 8: “The method of claim 7, further comprising: the identifier being calculated as monotonically increasing sequential integer values based on the number of segments or	

video frames generated during a fixed periodic content encryption key expiration interval.”	73
XIV.NO BASIS EXISTS FOR DISCRETIONARY DENIAL	73
XV. CONCLUSION.....	75

TABLE OF AUTHORITIES

CASES

<i>10X Genomics, Inc. v. Bio-Rad Labs., Inc.</i> , IPR2020-00086, Paper 8 (PTAB Apr. 27, 2020)	24
<i>Advanced Bionics, LLC v. MED-EL Elektromedizinische Geräte GmbH</i> , IPR2019-01469, Paper 6 (PTAB Feb. 13, 2020)	73
<i>Apple Inc. v. Fintiv, Inc.</i> , IPR2020-00019, Paper 11 (PTAB Mar. 20, 2020).....	74
<i>Ex Parte Spangler</i> , Appeal No. 2018-003800 (PTAB Feb. 20, 2019)	39
<i>General Plastic Indus. Co. v. Canon Kabushiki Kaisha</i> , IPR2016-01357, Paper 19 (PTAB Sept. 6, 2017)	74
<i>KSR Int’l Co. v. Teleflex Inc.</i> , 550 U.S. 398 (2007)	41
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.</i> , 868 F.3d 1013 (Fed. Cir. 2017)	10
<i>Valve Corp. v. Elec. Scripting Prods., Inc.</i> , IPR2019-00062, Paper 11 (PTAB Apr. 2, 2019)	74
<i>Videndum Production Solutions, Inc., v. Rotolight Ltd.</i> , IPR2023-01218, Paper 12 (PTAB April 19, 2024)	74

REGULATIONS

37 C.F.R. § 42.100(b)	10
37 C.F.R. § 42.104(a).....	7

STATUTES

35 U.S.C. § 102	8
35 U.S.C. § 103	7
35 U.S.C. § 282(b)	10

35 U.S.C. § 314.....	74
35 U.S.C. § 325.....	73

OTHER AUTHORITIES

83 Fed. Reg. 51,340 (Oct. 11, 2018).....	10
Manual of Patent Examining Procedure (MPEP) § 2117.....	46, 65
USPTO Binding Interim Guidance Addressing PTAB’s Approach to Discretionary Denials, June 21, 2022, available at https://www.uspto.gov/sites/default/files/documents/interim_proc_discretionary_denials_aia_parallel_district_court_litigation_memo_20220621.pdf	74

APPENDIX LISTING OF EXHIBITS

Exhibit	Description
1001	U.S. Patent No. 9,313,178 (“178 Patent”)
1002	Prosecution History of U.S. Patent No. 9,313,178 (“178 FH”)
1003	Declaration of Anthony J. Wechselberger (“Wechselberger”)
1004	Curriculum Vitae of Anthony J. Wechselberger (“Wechselberger-CV”)
1005	U.S. Patent Publication No. 2002/0172368 (“Peterka”)
1006	Certified Translation of Japanese Published Patent Application No. 2004-166153 (“Suzuki”)
1007	U.S. Patent Publication No. 2010/0235528 (“Bocharov”)
1008	U.S. Patent Publication No. 2008/0270308 (“Peterka308”)
1009	European Patent Publication No. 1,418,756 A2 (“Chen”)
1010	U.S. Patent Publication No. 2005/0138362 (“Kelly”)
1011	U.S. Patent No. 8,677,428 (“Lewis”)
1012	U.S. Patent No. 8,789,196 (“Pestoni”)
1013	International Publication No. WO 2010/107625 A2 (“Sood”)
1014	U.S. Patent No. 8,806,050 (“Chen050”)
1015	T. Stockhammer, “Dynamic adaptive streaming over HTTP: Standards and design principles,” in Proc. ACM MMSys’11, San Jose, CA, USA, Feb. 23–25, 2011, pp. 133–144 (“Stockhammer”)
1016	W. Van Lancker, D. Van Deursen, E. Mannens and R. Van de Walle, “HTTP adaptive streaming with Media Fragment URIs,” 2011 IEEE International Conference on Multimedia and Expo, Barcelona, Spain, 2011, pp. 1-6, doi: 10.1109/ICME.2011.6012149 (“Van Lancker”)
1017	3GPP TS 26.234 version 9.3.0 Release 9 (2010-06), Technical Specification, “Universal Mobile Telecommunications Systems (UMTS); LTE; Transparent end-to-end Packet Streaming Services (PSS); Protocols and codecs,” Reference RTS/TSGS-0426234v930 (June 2010) (“3GPP-AHS”)
1018	R. Pantos (Ed.), HTTP Live Streaming, draft-pantos-http-live-streaming-06 (Mar. 31, 2011) (“Pantos”)
1019	RFC793: Transmission Control Protocol (September 1981) (“RFC793”)
1020	RFC2616: Hypertext Transfer Protocol – HTTP/1.1 (June 1999) (“RFC2616”)
1021	The Secure Sockets Layer Protocol, V. 3.0, <i>available at</i> https://web.archive.org/web/19970614041044/http://home.netscape.com/eng/ssl3/ssl-toc.html and

	https://web.archive.org/web/19970617034012/http://home.netscape.com/eng/ssl3/3-SPEC.HTM#1 (“SSL-Standard”)
1022	U.S. Patent No. 8,756,673 (“Barrus”)
1023	Yuan et al., “Layered access control for MPEG-4 FGS video,” Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429), Barcelona, Spain, doi: 10.1109/ICIP.2003.1247012 (“Yuan”)
1024	U.S. Patent App. Pub. No. 2005/0060316 (“Kamath”)
1025	RESERVED
1026	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 2 (Feb. 25, 2022)
1027	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 9 (July 8, 2022)
1028	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 11 (Sept. 1, 2022)
1029	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 13 (Nov. 28, 2022)
1030	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 14 (Nov. 28, 2022)
1031	<i>Apple Inc. v. Ericsson AB</i> , IPR2022-00618, Paper 17 (Dec. 30, 2022)
1032	<i>In the Matter of Certain Electronic Devices, Including Mobile Phones, Tablets, Laptops, Components Thereof, and Products Containing the Same</i> , Inv. No. 337-TA-1376, Order No. 25 (Mar. 13, 2024)
1033	<i>In the Matter of Certain Electronic Devices, Including Mobile Phones, Tablets, Laptops, Components Thereof, and Products Containing the Same</i> , Inv. No. 337-TA-1376, Complaint, Exhibit 24 (October 12, 2023)
1034	<i>Telefonaktiebolaget LM Ericsson et al. v. Lenovo</i> , 5:23-cv-00570, Order Granting Motion to Stay (E.D.N.C. Feb. 9, 2024)
1035	USPTO Binding Interim Guidance Addressing PTAB’s Approach to Discretionary Denials, June 21, 2022, <i>available at</i> https://www.uspto.gov/sites/default/files/documents/interim_proc_discretionary_denials_aia_parallel_district_court_litigation_memo_20220621.pdf .

CLAIM LISTING

Element	Claim Language
[1PRE]	A method for handling secure distribution of content comprising:
[1A]	initiating a media playback request and receiving a playback request response;
[1B]	parsing content information from the playback request response, the content information including content encryption keys, content encryption key identifiers, and content encryption key expiration times;
[1C]	retrieving content and manifest files from a content delivery server;
[1D]	detecting content encryption key rotation boundaries between periods of use of different content encryption keys in decrypting retrieved content;
[1E]	issuing requests to a license server ahead of a key rotation boundary to retrieve a second content encryption key to be used after a content encryption key rotation boundary is reached; and
[1F]	applying the second key for content decryption after the key rotation boundary is reached.
[2]	The method of claim 1, wherein the content encryption keys and content encryption key identifiers returned in the playback request response include the content encryption key and associated identifier currently being applied as well as a future content encryption key and associated identifier yet to be applied.
[3]	The method of claim 1, wherein the content encryption key expiration returned in the playback request response is expressed as an expected minimum interval between periodic key rotation.
[4]	The method of claim 3, further comprising: timing the prefetching of a next un-retrieved content encryption key based on an expected expiration of the content encryption key currently being used.

Element	Claim Language
[5]	The method of claim 4, further comprising: prefetching a next un-retrieved key a fixed duration before the expected expiration of the content encryption key currently being used.
[6]	The method of claim 4, further comprising: prefetching a next un-retrieved key within a period of time after the expected expiration of the content encryption key currently being applied, the period of time beginning at the expected expiration of the content encryption key currently being used, and ending at a duration calculated as a fixed percentage of a fixed periodic content encryption key expiration interval.
[7]	The method of claim 1, wherein subsequent content encryption key identifiers are predictable based on a predetermined known progression.
[8]	The method of claim 7, further comprising: the identifier being calculated as monotonically increasing sequential integer values based on the number of segments or video frames generated during a fixed periodic content encryption key expiration interval.
[10]	The method of claim 1, wherein license server communications are secured and authenticated using a selected one of Secure Sockets Layer and a token-based technique using a token encrypted with a symmetric key.
[12]	The method of claim 1, wherein actual content encryption key rotation boundaries are detected based on real-time in-band notifications.
[14]	The method of claim 12, wherein content encryption key rotation boundaries are detected based on notifications present in the content manifest file, the content manifest file being selected from an m3u8 file, an IIS Smooth Streaming manifest file, a DASH MPD file and a proprietary manifest file.
[16PRE]	A computerized device operable as a client for handling secure distribution of content, comprising:
[16A]	memory operative to store computer program instructions; one or more processors; input/output interface circuitry; and interconnect

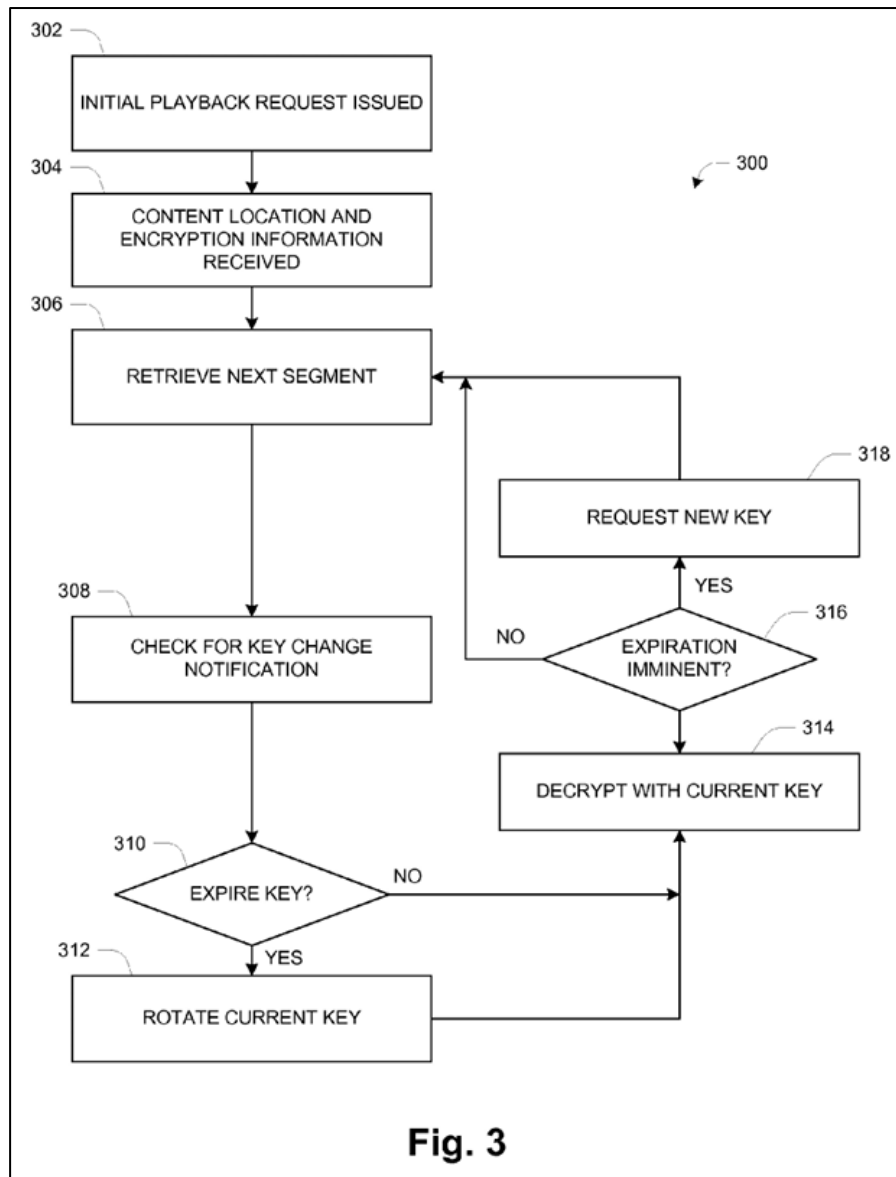
Element	Claim Language
	circuitry coupling the memory, processors and input/output interface circuitry together,
[16B]	wherein the processors are operative to execute the computer program instructions from the memory to cause the computerized device to: initiate a media playback request and receive a playback request response;
[16C]	parse content information from the playback request response, the content information including content encryption keys, content encryption key identifiers, and content encryption key expiration times;
[16D]	retrieve content and manifest files from a content delivery server;
[16E]	detect content encryption key rotation boundaries between periods of use of different content encryption keys in decrypting retrieved content;
[16F]	issue requests to a license server ahead of a key rotation boundary to retrieve a second content encryption key to be used after a content encryption key rotation boundary is reached; and
[16G]	apply the second key for content decryption after the key rotation boundary is reached.
[17]	The computerized device of claim 16, wherein the content encryption keys and content encryption key identifiers returned in the playback request response include the content encryption key and associated identifier currently being applied as well as a future content encryption key and associated identifier yet to be applied.
[18]	The computerized device of claim 17, wherein the computer program instructions further cause the computerized device to time the prefetching of a next un-retrieved content encryption key based on an expected expiration of the content encryption key currently being used.
[19]	The computerized device of claim 16, wherein subsequent content encryption key identifiers are predictable based on a predetermined known progression.

Element	Claim Language
[20]	The computerized device of claim 16, wherein actual content encryption key rotation boundaries are detected based on real-time in-band notifications.

I. INTRODUCTION

Petitioners request *inter partes* review (“IPR”) and cancellation of claims 1-8, 10, 12, 14, and 16-20 (“Challenged Claims”) of U.S. Patent No. 9,313,178 (EX-1001, “’178 Patent”).

The ’178 Patent states that delivery of streaming multimedia content has moved away from using private networks and towards Web-based delivery of files over the Internet, known as over-the-top (“OTT”) delivery. EX-1001, 1:15-42. Because OTT delivery utilizes public network infrastructure, the ’178 Patent states that traditional content protection paradigms must be modified. *Id.* To this end, the ’178 Patent provides for use of segmented content in which segments are protected by a rotation of serial, time-limited encryption keys. *Id.*, 1:43-65. FIG. 3 below illustrates the ’178 Patent’s content downloading/decryption with key rotation.



As shown above, a new key required to decrypt an upcoming media segment is requested by the client *before* the key currently in use expires (steps 316, 318)—when expiration is “imminent.” EX-1001, 11:38-62. The ’178 Patent’s system thereby attempts to avoid service interruptions caused by a client not having the key needed to decrypt a received media segment. *Id.*

The '178 Patent's claims are directed to this basic concept. However, the concept of rotating keys in OTT streaming was well-known before the '178 Patent, as was obtaining such keys on an as-needed basis, including in time periods *before they were needed*. For example, Peterka (EX-1005) alone (Ground 1) and Peterka combined with Suzuki (EX-1006), which discloses specific detail regarding key identifiers and when to seek new keys (Ground 2), each render obvious both independent claims (claims 1, 16) and claims 2-6, 10, 12, 17-18, and 20.

Peterka was previously considered by the Board in an IPR petition filed by an unrelated petitioner (Apple Inc., in IPR2022-00618), which combined Peterka with Bocharov (EX-1007), Peterka308 (EX-1008), and Chen (EX-1009). Combinations with Peterka using those references are presented herein (Grounds 3, 5). The Board instituted review in IPR2022-00618 (EX-1028, 32), but terminated the proceeding due to settlement before Final Written Decision. EX-1031, 2-3. This unfinished record—institution on previous IPR grounds using Peterka combined with the same references presented herein—urges final determination of patentability for these claims. Petitioners respectfully request institution of IPR and that the Challenged Claims be cancelled as unpatentable.

II. MANDATORY NOTICES

A. Real Party-In-Interest

Petitioners are Real Party-in-Interests to this Petition. The following entities have also been accused of infringement in the ongoing Related Matters listed under U.S. District Court for the Eastern District of North Carolina and the International Trade Commission: Lenovo (Beijing) Co., Limited; Lenovo Group Ltd.; Lenovo Information Products Shenzhen Co. Ltd.; Lenovo PC HK Limited; Lenovo (Shanghai) Electronics Technology Co. Ltd.; Motorola Mobility LLC; Motorola (Wuhan) Mobility Technologies Communication Co., Ltd. These entities do not have the ability to control, fund, or be involved in any aspect of this proceeding. As such, Petitioners do not believe these defendants are Real Parties-in-Interest.

B. Related Matters

A decision in this proceeding could affect or be affected by the following:

1. United States Patent & Trademark Office

The application from which U.S. Patent No. 9,313,178 issued is a Continuation of U.S. Patent Application No. 13/530,997 (U.S. Patent No. 8,751,807), filed June 22, 2012. U.S. Patent No. 9,313,178 also claims priority to U.S. Provisional Application No. 61/500,316, filed June 23, 2011.

The following U.S. patent applications claim the benefit of priority to U.S. Patent 9,313,178 and U.S. Provisional Application No. 61/500,316:

(i) U.S. Patent Application No. 15/067,219 (U.S. Patent No. 9,912,975), filed March 11, 2016; (ii) U.S. Patent Application No. 15/910,911 (U.S. Patent No. 10,477,257), filed March 2, 2018; (iii) U.S. Patent Application No. 16/672,463 (U.S. Patent No. 11,122,313), filed November 3, 2019; (iv) U.S. Patent Application No. 17/388,389 (U.S. Patent No. 11,483,604), filed July 29, 2021; (v) U.S. Patent Application No. 17/973,138, filed October 25, 2022 (Notice of Allowance issued July 31, 2024).

2. United States Patent Trial and Appeal Board

The '178 Patent was at issue in *Apple Inc. v. Telefonaktiebolaget LM Ericsson et al.*, Case No. IPR2022-00618. The Board instituted review on grounds using the same references relied upon in Grounds 3 and 5 herein. EX-1026, 1-2; EX-1028, 32. The parties in IPR2022-00618 settled, and the proceeding was terminated, prior to any finding on the patentability of any Challenged Claim. EX-1031, 2-3. As discussed *infra* § XIV, Petitioners were not involved in IPR2022-00618.

Concurrently with the present Petition, Petitioners are filing a petition for *inter partes* review of related U.S. Patent No. 11,122,313, under case no. IPR2025-00004. Petitioners request that both petitions be reviewed by the same panel, as the patents are commonly assigned and share the same specification.

3. U.S. District Court for the Eastern District of Texas

Ericsson Inc. et al. v. Samsung Electronics Co., Ltd. et al., Case No. 2:21-cv-

4. U.S. District Court for the Eastern District of North Carolina

Telefonaktiebolaget LM Ericsson et al. v. Lenovo (United States), Inc. et al.,
No. 5:23-cv-00570 (“the EDNC litigation”).

5. International Trade Commission (“ITC”)

(i) *Certain Electronic Devices with Wireless Connectivity, Components Thereof, and Products Containing Same*; Inv. No. 337-TA-1245; (ii) *Certain Electronic Devices, Including Mobile Phones, Tablets, Laptops, Components Thereof, and Products Containing the Same*; Inv. No. 337-TA-1376 (“the 1376 Investigation”).

C. Counsel and Service Information - § 42.8(b)(3) and (4)

Lead Counsel	Scott A. McKeown, Reg. No. 42,866
Backup Counsel	Anant K. Saraswat, Reg. No. 76,050 Turhan F. Sarwar, pending admission <i>pro hac vice</i> Elizabeth A. DiMarco, pending admission <i>pro hac vice</i>
Service Information	<u>E-mail:</u> SMcKeown-PTAB@wolfgreenfield.com ASaraswat-PTAB@wolfgreenfield.com TSarwar-PTAB@wolfgreenfield.com Elizabeth.DiMarco@wolfgreenfield.com <u>Post and hand delivery:</u> Wolf, Greenfield & Sacks, P.C. 600 Atlantic Avenue Boston, MA 02210-2206 <u>Telephone:</u> 617-646-8000 <u>Facsimile:</u> 617-646-8646

A power of attorney is submitted with the Petition. Counsel for Petitioners consent to service of all documents via electronic mail.

III. GROUNDS FOR STANDING

Petitioners certify that the '178 Patent is available for IPR and that Petitioners are not barred or estopped from requesting IPR of the Challenged Claims. 37 C.F.R. § 42.104(a).

IV. GROUNDS FOR UNPATENTABILITY

The table below identifies each ground of unpatentability presented herein. *See* Wechselberger, ¶¶66-70, 21-34, 222-223.

Ground and Reference(s)		Claims	Basis
1	Peterka (EX-1005)	1-6, 10, 12, 16-18, 20	§ 103
2	Peterka, Suzuki (EX-1006)	1-6, 10, 12, 16-18, 20	§ 103
3	Peterka, Bocharov (EX-1007)	1-6, 10, 12, 14, 16-18, 20	§ 103
4	Peterka, Suzuki, Bocharov	1-6, 10, 12, 14, 16-18, 20	§ 103
5	Peterka, Bocharov, Peterka308 (EX-1008), Chen (EX-1009)	7-8, 19	§ 103

The '178 Patent claims priority to a provisional application filed June 23, 2011. Each reference applied herein qualifies as prior art to each Challenged Claim even if Patent Owner ("PO") could establish entitlement to an effective filing date of June 23, 2011, on at least the bases shown below:

Reference	Filed	Published	Pre-AIA § 102 Prior Art Basis
Peterka	10/26/2001	11/21/2002	§ 102(a)-(b), (e)
Suzuki		06/10/2004	§ 102(b)
Bocharov	03/16/2009	09/16/2010	§ 102(a), (e)
Peterka308	08/22/2007	10/30/2008	§ 102(a)-(b), (e)
Chen		5/12/2004	§ 102(b)

V. BACKGROUND

A. '178 Patent

The '178 Patent relates to “encryption key rotation for live streaming media.” EX-1001, 1:15-23; Wechselberger (EX-1003), ¶35. The '178 Patent describes a server that streams segments of encrypted media content to a client. EX-1001, 1:38-42, 1:63-65. Different segments in a media stream may be encrypted with different content encryption keys. *Id.*, 1:45-65, 2:62-3:8. The client communicates with a license server to pre-fetch content encryption keys ahead of key rotation boundaries in the media stream, and the client applies a retrieved key to decrypt a corresponding segment of the media stream. *Id.*, 11:3-25, 11:38-62. The '178 Patent refers to “transitioning between use of different keys” as “key ‘rotation’.” *Id.*, 1:63-65.

B. Prosecution History

The Examiner identified claims 1-20 as allowable in a first office action. EX-1002, 67-71 (citing only Kamath (EX-1024)); Wechselberger, ¶36. In October 2015, Examiner allowed the application, noting that “the prior art of record [Kamath] does

not teach detecting key rotation boundaries and issuing requests to a license server ahead of the key rotation boundaries.” EX-1002, 29-31.

In February 2022, Apple filed an IPR petition challenging claims 1-20 (IPR2022-00618). The Board instituted on grounds that used the same references used in Grounds 3 and 5, herein. EX-1026, 1-2; EX-1028, 32. The parties in IPR2022-00618 settled after institution, and the proceeding was terminated before Final Written Decision. EX-1031, 2-3. As discussed *infra* § XIV, Petitioners were not involved in IPR2022-00618. Wechselberger, ¶37.

VI. PERSON OF ORDINARY SKILL IN THE ART (“POSITA”)

A POSITA would have had a bachelor’s degree in computer science, computer engineering, electrical engineering, or a related field, and 2-3 years of practical engineering experience relating to information security systems that employ cryptographic keys to encrypt/decrypt digital data. Wechselberger, ¶¶38-42; *see id.*, ¶¶1-20. Additional education could substitute for professional experience, or vice versa. *Id.*

VII. CLAIM INTERPRETATION

Claim terms are construed herein using the standard used in civil actions under 35 U.S.C. § 282(b),¹ in accordance with “the ordinary and customary meaning” as understood by a POSITA and the patent’s prosecution history. 37 C.F.R. § 42.100(b); *see also* Wechselberger, ¶43. The Board need only construe terms to the extent necessary to resolve disputes between the parties. *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017); 83 Fed. Reg. 51,340, at 51,353 (Oct. 11, 2018). No claim term requires that the Board adopt an exact outer boundary construction because the prior art relied upon herein meets the claims under any plausible interpretation as detailed below.

VIII. OVERVIEW OF PRIOR ART

A. Peterka

Peterka (EX-1005) relates to secure multimedia distribution over a network. Peterka, Abstract, FIG. 1, [0038], [0006]-[0012]. Peterka’s “client” devices download streaming media from a content distribution system comprising servers. *Id.*, [0038]. Content is divided into segments that are each encrypted with a different key prior to streaming. *Id.*, [0080]-[0082], [0091]-[0101], [0122]-[0125], FIGs. 7A-

¹ In the 1376 Investigation, the ALJ issued a claim construction order construing certain claim terms (EX-1032). Those constructions do not impact the grounds herein.

7B. Peterka discloses servers distributing content to clients using purchase mechanisms, including “pay-by-time (PBT),” in which users pay for the duration of content watched. Peterka, [0053], [0080], [0099]; Wechselberger, ¶¶45-47.

Encrypted media content segments are streamed to clients in packets, e.g., Real-Time Transport Protocol (“RTP”) packets. Peterka, [0043]. Media content may be in, e.g., MPEG-4 format. *Id.*, [0119]. Peterka teaches using these content packets to signal key changes. *Id.*, [0116]-[0120] (teaching “key change” notifications in, e.g., headers). Peterka’s “signaling method” informs a client when to start using a new key for decryption, regardless of how the key was obtained by the client. Wechselberger, ¶¶48-49. Peterka separately teaches several ways for a client to obtain the next key before it is needed for decryption. *Id.*

For each encrypted content segment, Peterka’s client obtains the key used to encrypt the segment—e.g., a content key (“CK”)—and applies that key to decrypt the segment before playing it. *Id.*, [0011], [0091]-[0093], [0127], FIG. 7A; Wechselberger, ¶50. For obtaining keys, Peterka describes a push model (where a server keeps track of keys at each client and pushes keys as needed), a pull model (where a client keeps track of its keys and actively requests keys as needed), and a push-pull model (e.g., where a server only pushes keys to clients which confirmed receipt of previous key). Peterka, [0105]-[0115]. All these methods for **obtaining**

keys are compatible with Peterka's signaling method informing clients when to *change* keys used for decryption. The Grounds herein rely on Peterka's *pull model*.

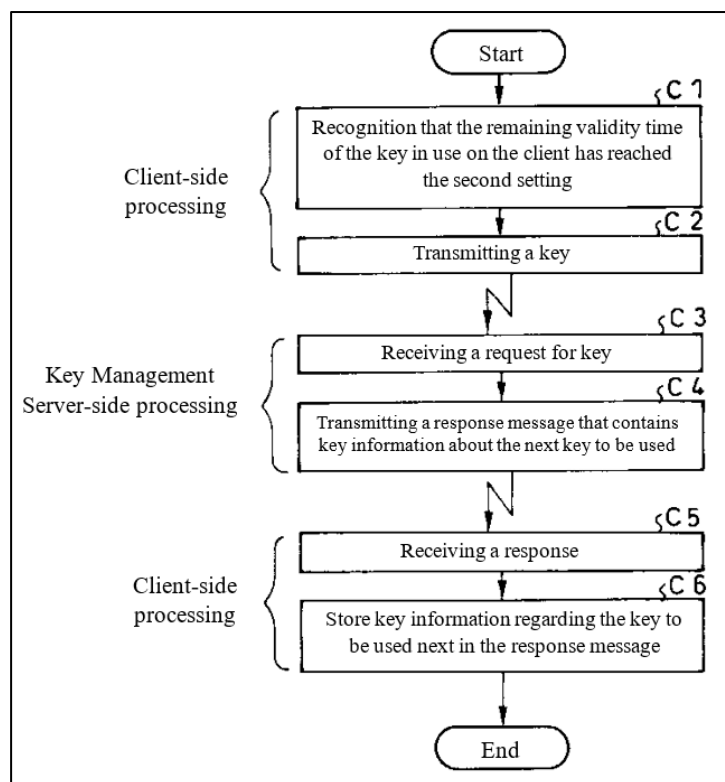
Each key, along with information about the key, is distributed to a client in an Entitlement Control Message ("ECM") or an Entitlement Management Message ("EMM"). *Id.*, [0104], [0121]-[0125], FIG. 7B. In Peterka's "pull model" for obtaining keys, "each client keeps track of the keys and their expiration times and actively requests new keys before the current keys expire so as to avoid service interruptions." *Id.*, [0105]-[0106], FIG. 8. To mitigate excessive server loads at media segment boundaries when clients actively request and negotiate keys for segments, Peterka teaches distributing *multiple* keys (e.g., multiple ECMs) to a client—e.g., for the current *and* next media segment—in response to such a request. *Id.*, [0080]; *see also id.*, [0130], FIG. 7B; Wechselberger, ¶¶51-52.

Peterka also discloses adjusting the quality (e.g., bit rate) of viewed media (e.g., for an MPEG-4 format stream), but does not specify the mechanism by which clients receive information that facilitates such adjustments. Peterka, [0054], [0119]; Wechselberger, ¶53. POSITAs would have understood Peterka's teachings about content being available at different quality levels (Peterka, [0054])—and users having the option to choose media quality—to be implemented with, e.g., Peterka's PBT mechanism, where users are charged based on the time duration of content viewed, at a chosen quality/bit rate. Wechselberger, ¶54.

B. Suzuki

Suzuki (EX-1006) relates to securely providing media content encrypted with keys that are “periodically” changed at “regular intervals.” Suzuki, Abstract, [0001]-[0005]. Suzuki teaches “enabl[ing] real-time data processing” of encrypted media content packets “by eliminating the confirmation delay” during key changes. *Id.*, [0011]-[0012]. POSITAs would have understood Suzuki’s system to be well-suited for streaming video. Wechselberger, ¶¶55-56; Suzuki, [0144].

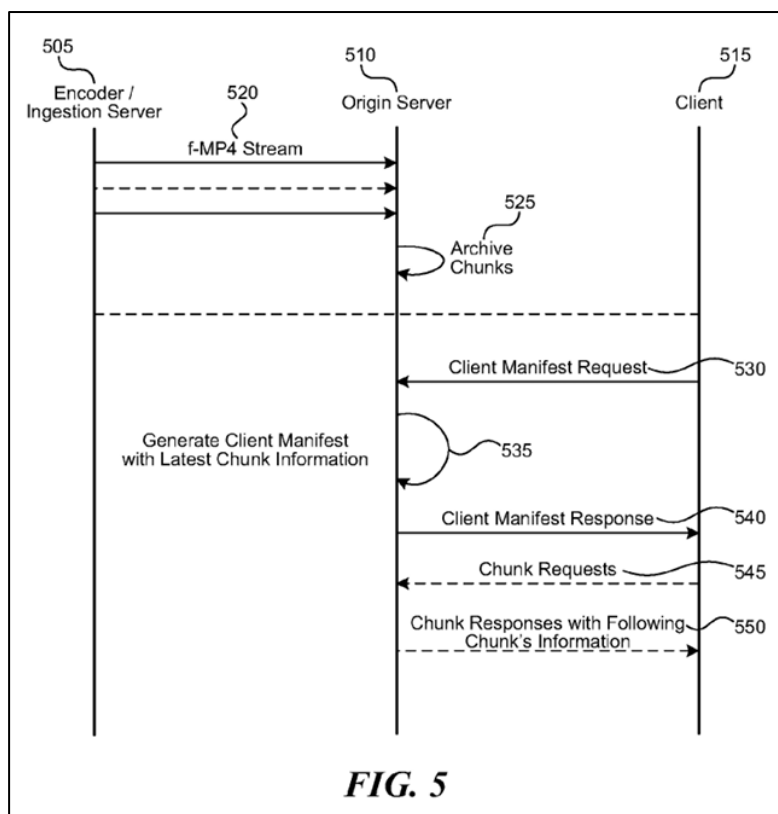
Suzuki addresses confirmation delay using a client “key management means... that transmits a key information request to the key management server when the remaining validity time of the key in use reaches” a particular value, and a key management server “key management means” that “returns a response message” with “the decryption key to be used next, its key number, and its remaining validity time to the client.” Suzuki, [0016]. Suzuki’s client performs “processing” that includes “recogni[zing]” that the value of “the remaining validity time has decreased and reached a [particular] setting value.” *Id.*, [0036], [0059], FIG. 4 (below).



C. Bocharov

Bocharov (EX-1007) is directed to streaming media fragments to a client, in which “manifest” files containing metadata about each fragment (e.g., encoding information about the fragment) are embedded in the fragments along with “media content”—thus “eliminat[ing] the usage of a typical control channel.” Bocharov, Abstract, [0001], [0007], [0014], [0030]-[0031], [0053], FIGs. 3-5; Wechselberger, ¶57.

Figure 5 depicts network interactions in Bocharov’s system:



Above, “encoder [server] 505 continuously provides media data 520 to... origin server 510”—e.g. “fragments of an MP4 stream based on a live event.” Bocharov, [0053]. “The origin server 510 receives a manifest request 530 from a client 515,” and “generates 535 a client manifest based on the latest media fragment information.” *Id.* “The origin server 510 provides a client manifest response 540 to the client 515,” “[t]he client 515 then sends one or more media fragment requests 545, and the origin server 510 responds 550 with the requested media fragment and potentially information about subsequent media fragments.” *Id.*; *see also id.*, [0006], [0014]-[0016], [0021]-[0031], [0049]-[0057], [0060]. After sending a client the initial client manifest response, Bocharov teaches servers “includ[ing]” additional

manifests to accompany the media content in fragments the client receives. *Id.*, [0023]-[0024], [0007], [0014]-[0016]; Wechselberger, ¶¶58-59.

Bocharov's "client manifest" includes information useful for a client requesting media fragments from a server. Wechselberger, ¶60. For example, Bocharov teaches providing timestamps for each fragment archived up to the point of the request, which the client uses to compose a URL for requesting fragments from the server. Bocharov, [0030]-[0031].

Bocharov's client manifest further identifies available encodings of the media content. *Id.*, [0049]. For programs encoded in multiple bit rates, the client references Bocharov's manifests to "begin requesting fragments in whichever encoding the client chooses." *Id.* Bocharov's manifests enable switching between different encodings mid-stream. Wechselberger, ¶61. "For example, the client may initially select a low bit rate encoding and select higher bit rate encodings for subsequent fragments until network bandwidth limits the client's ability to receive the fragments at a bit rate." Bocharov, [0049]; *see also id.*, [0023]-[0025], [0030]-[0031], [0052], [0057]. Additionally, Bocharov's manifests facilitate a "payment model" where clients pay by quality and "switching to particular bit rates" is "enable[d] or disable[d]" "based on whether the conditions of the payment model have been met." *Id.*, [0060]; Wechselberger, ¶62.

D. Peterka308

Peterka308 (EX-1008) describes techniques for decrypting secured video segments for playback from a DVR. Peterka308, Abstract, FIGs. 2-6; Wechselberger, ¶¶63-64. In Peterka308, a DVR “receives encrypted content from [a] content source 102.” Peterka308, [0021]. “[C]ontent protection module 106” decrypts the content so that the DVR “may re-encrypt the content and record the re-encrypted content by storing it on a hard drive 202.” *Id.* “DRM component 204” of the DVR can be configured to “generate[] a unique CEK [(content encryption key)] for each segment of copy never content at a predefined time interval during recording.” *Id.*, [0024].

Peterka308’s CEK need not be “stored in the hard drive 202.” *Id.*, [0025]. “Rather, a time stamp, which indicates the relative time value from the beginning of the recording to the start of the segment, is generated and stored at the time that each unique CEK is determined.” *Id.* “Each time stamp is dynamically added to [a] content license as the recording progresses.” *Id.* “During playback, the master key and the time stamp for a particular segment may be utilized, at least in part, to derive the CEK for that segment so that the encrypted content for that segment... may be decrypted.” *Id.*; *see also id.*, [0026]-[0028].

E. Chen

Chen (EX-1009) describes techniques for encrypting video on-demand (“VOD”) “material with a changing cryptographic key.” Chen, [0008]; *see id.*, [0008]-[0011], [0027]-[0028], [0034]-[0038], FIGS. 2A-2D; Wechselberger, ¶65. Chen’s “technique for implementing rapid cryptographic key changes” uses any of four schemes “for breaking up (subdividing, segmenting, sectioning) the content to be encoded/encrypted into a plurality of ‘crypto periods’, covered by different cryptographic keys”: “fixed crypto period,” “fixed number of packets,” “fixed ‘marker’ count,” and “random crypto period.” Chen, [0034]; *see id.*, [0037]-[0047].

IX. GROUND 1: PETERKA RENDERS CLAIMS 1-6, 10, 12, 16-18, AND 20 OBVIOUS

Peterka discloses a multimedia distribution system that uses a pay-by-time purchase mechanism and a “pull” key distribution method where clients monitor key expiration times to request new keys before they expire. Wechselberger, ¶71; Peterka, [0053], [0078]-[0082], [0105]-[0106], [0110]; *supra* § VIII.A. Every element of claims 1 and 16 is satisfied by or obvious over Peterka’s pull method. As discussed *infra* § IX.A.1.d, Peterka—which discloses servers that “adjust the rate in real time based on the immediate state of the network” and clients that “adjust the bit rate” of media streamed to the user (Peterka, [0054]), e.g., in the MPEG-4 format (*id.*, [0119])—also teaches retrieval of conventional manifest files from content delivery servers to enable such adjustments. Wechselberger, ¶¶72-73.

A. Claim-by-Claim Analysis

1. Claim 1

a. [1PRE] “A method for handling secure distribution of content comprising:”

PO has conceded that Claim 1’s preamble is not limiting. EX-1032, 15-17. Nevertheless, Peterka discloses it. Peterka, [0012] (“[M]ethod of providing program material to... clients”), [0083] (“[E]ncrypted material is provided for distribution to a client.”), [0094] (“[E]ncryption keys can be distributed to clients to facilitate reception of... program content.”), FIGs. 1, 3; Wechselberger, ¶74.

b. [1A] “initiating a media playback request and receiving a playback request response;”

In Peterka, clients request access to media content and receive keys to gain access to at least a portion of a media stream. Peterka, [0080]-[0082] (“When the client negotiates encryption keys with the caching server, it will start receiving” segments of selected “content” from an ongoing show/event), [0091]-[0092] (“FIG. 7A illustrates a plot showing the number of *requests*^{2/} for a program” and “the distribution of content keys”), [0141]-[0143] (disclosing embodiments where “*request* [for program content is] sent by the client to the caching server” and the client receives encryption keys and encrypted content), FIGs. 7A-7B, 20A-20B. Peterka’s disclosed client-initiated requests to access media, e.g., program content,

² Emphasis added unless otherwise indicated.

are each *initiating a media playback request* as recited. Wechselberger, ¶¶75-76. This interpretation is consistent with how “media playback requests” are described in the ’178 specification. EX-1001, 2:15-21, 9:49-52; Wechselberger, ¶76.

As discussed *supra* § VIII.A, Peterka teaches distributing multiple keys (e.g., current and next) to a client when it negotiates access to media content. Peterka, [0080], [0130], FIG. 7B. Specifically, Peterka describes “a key management approach where each [client] is given keys to the current as well as the next segment.” *Id.*, [0080]. Peterka teaches that such an approach can “mitigate[]” “large load[s]” on a server that occur at “the segment boundaries” of a media stream—e.g., in a pull model where users pay for time increments corresponding to media segments and a server is likely to receive key requests from many clients at each media segment boundary. *Id.*; *see also id.*, [0053]; Wechselberger, ¶77.

Peterka FIG. 7B (annotated below)—“showing exemplary distribution of cryptographic keys during portions of a program” (Peterka, [0021])—confirms that this key management approach was applicable/desirable in any distribution scheme in which key requests from many clients are expected at segment boundaries. Wechselberger, ¶¶78-79.

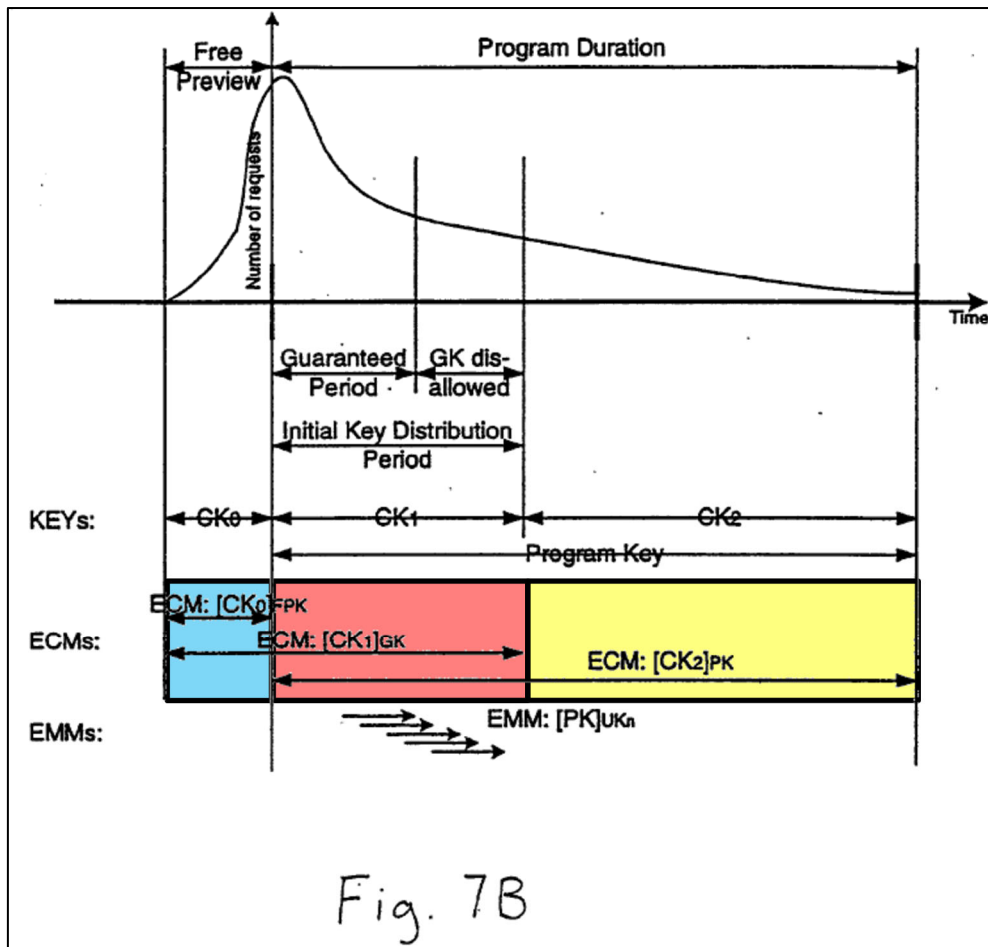


Figure 7B shows two content keys CK_0 and CK_1 distributed in separate ECMs for each request received during a free preview period (blue), and two content keys CK_1 and CK_2 distributed in separate ECMs for each request received during an initial key distribution period (red). Wechselberger, ¶79. ECMs with CK_1 are distributed to clients *before* the period in which CK_1 is used (red). *Id.*; Peterka, [0130] (user “receive[s] the next necessary decryption key within the initial key distribution”). ECMs with CK_2 are distributed to clients *before* the period in which CK_2 is used (yellow). Peterka, [0130]. Peterka’s illustration of one CK per ECM is consistent with Peterka’s teachings regarding the fields in each ECM/EMM, which are

characterized as containing information about one key. Peterka, [0121] (e.g., “*the* key in the message”); Wechselberger, ¶79 n.3.

Peterka discloses that each key (along with information about the key, *see infra* § IX.A.1.c) is “distributed to the client *in response* to the request by the client” in an EMM or ECM. Peterka, [0104], FIG. 8. The multiple EMMs/ECMs sent *in response* to a particular *media playback request* collectively constitute a *playback request response* received by a client, as claimed. Peterka, [0104]-[0106], [0121], [0122]-[0125], FIGS. 7B, 8; Wechselberger, ¶¶80-81.

To the extent “playback request response” is interpreted to require data transmitted in one message—a narrow interpretation that improperly uses the claim term “response” interchangeably with the word “message”—it would have been obvious to package multiple keys/entitlement messages (ECMs or EMMs) into a “single” “combined” “multicast message,” in view of Peterka’s express teachings that this was a viable option for “improved efficiency,” and a POSITA’s knowledge that multiple keys were conventionally transmitted in one message in response to a client request. Wechselberger, ¶81 (citing Peterka, [0104], [0121]; Suzuki, [0076]-[0077], FIGs. 5-6).

- c. [1B] “parsing content information from the playback request response, the content information including content encryption keys, content encryption key identifiers, and content encryption key expiration times;”

The '178 Patent uses “parsing” when describing retrieval of information from a playback request response. *See* EX-1001, 2:15-21, 9:52-56; Wechselberger, ¶82. The specific manner of parsing is left to POSITAs. As discussed *supra* § IX.A.1.b, Peterka discloses a server that returns multiple ECMs/EMMs (*playback request response*) in response to a request to access/play at least a portion of a stream (*media playback request*). The ECMs/EMMs contain information responsive to the request, and POSITAs understood that the client parses the information from the ECMs/EMMs to distinguish among data included in the response. Wechselberger, ¶¶83-84 (citing Peterka, [0104]-[0106], [0121]-[0125], FIG. 7B). Peterka discloses, e.g., each ECM/EMM including separate fields for “a sequence number or a time stamp,” and “the time remaining in the lifetime of the key.” Peterka, [0121].

i. Content Information

Peterka’s ECMs/EMMs provide the “*content information*” in Element [1B] (Wechselberger, ¶85):

(1) Content Encryption Keys

The '178 Patent describes “*content encryption keys*” as keys used for encrypting or decrypting “the content item.” EX-1001, 1:45-58, 3:51-52;

Wechselberger, ¶86. Claims 1 and 16 make clear that “content encryption keys” must be used *at least* for decrypting content. Peterka’s ECMs/EMMs carry keys to clients, including “content key[s]” (“CKs”) that were “used to encrypt the content itself” and are used by the client to decrypt content (*content encryption keys*). Peterka, [0101], [0106], [0116], [0121]. Multiple keys are conveyed in response to a playback request, as discussed *supra* § IX.A.1.b, and parsed.

Elements [1D]-[1F] require content encryption keys to be used/applied for decrypting content. Peterka teaches embodiments where CKs are themselves encrypted and must be decrypted before use. Peterka, [0094]-[0104]; Wechselberger, ¶¶87-88. Encrypted CKs meet the broad reading of “content encryption keys” that PO applied in the 1376 Investigation for infringement—i.e., an encrypted key is a “content encryption key” even though it must first be decrypted before it can be used/applied for decrypting content. *See 10X Genomics, Inc. v. Bio-Rad Labs., Inc.*, IPR2020-00086, Paper 8, at 17-22 (PTAB Apr. 27, 2020).

POSITAs would also have understood that another well-known, conventional option was to transmit Peterka’s CKs in clear (or unencrypted) form, which would satisfy the term “content encryption keys” even if interpreted to exclude encrypted

content keys.³ Wechselberger, ¶89 (citing, for corroboration, Suzuki, [0016], [0072], FIG. 5).

(2) Content Encryption Key Identifiers

The '178 specification says each content encryption key may be assigned a “unique” “*content encryption key identifier*.”⁴ EX-1001, 3:29-32; Wechselberger, ¶90. Peterka discloses that each ECM/EMM includes “a *sequence number* or a *time stamp* to protect against replay attacks.” Peterka, [0121]. Because the sequence number/timestamp is unique to each ECM/EMM (to prevent replay attacks), and each ECM/EMM carries a different content encryption key (*supra* § IX.A.1.b), the sequence number/timestamp each constitutes a unique *content encryption key identifier*. Wechselberger, ¶91. This is consistent with the '178 specification's teaching of “content encryption key identifiers” that are “selected based on a segment number of the prepared content” and thus apply to an entire media segment. EX-1001, 3:34-35, 4:8-10; *see also id.*, 6:1-3, 6:17-19; Wechselberger, ¶92 (POSITAs would have recognized that one obvious way to “select” an identifier based on a segment number was by using the segment number *itself* as the identifier).

³ PO disputes in the 1376 Investigation that this narrower interpretation is correct.

⁴ As shown, Peterka meets this limitation even if “content encryption key identifier” is construed to mean an identifier that *uniquely* identifies a key.

As discussed *infra* § IX.A.1.c.ii, the received keys' respective identifiers are conveyed in response to a playback request and parsed. Wechselberger, ¶93.

Alternatively, POSITAs would have been motivated (in view of, e.g., FIG. 7B, which *shows* delivery of content keys with unique designations—CK₀, CK₁, CK₂) to implement Peterka's client to receive and use a key identifier to identify the content key parsed from an ECM/EMM. Wechselberger, ¶94. In systems like Peterka's ([0080, [0122], [0130], FIG. 7B), where multiple keys are utilized, POSITAs would have recognized the need to differentiate between keys, would have been motivated to do so with key identifiers, and would have reasonably expected success. Wechselberger, ¶¶93-96 (citing, for corroboration, Suzuki, [0013] (client "table" with "key number" for each key), [0016] (client receiving "key number" for each key), [0030]-[0031], [0043]-[0048], [0080]-[0084] (client using "key number" to apply correct key), FIG. 5; Pestoni, 12:16-35 ("a key identifier for each key"))).

(3) Content Encryption Key Expiration Times

Peterka discloses that an ECM/EMM includes a field indicating "the time remaining in the lifetime of the key" contained in the ECM/EMM, which constitutes a *content encryption key expiration time*. Peterka, [0121]; *see also id.*, [0106] ("[T]he key and *its expiration time* are distributed to the client..."), [0105] ("Under the pull model, each client keeps track of the keys and *their expiration times* and actively requests new keys before the current keys expire[.]"), FIG. 8;

Wechselberger, ¶97. This interpretation is consistent with the '178 specification's discussion of requesting new content encryption keys before the end of the current key's lifespan. EX-1001, 6:66-7:2, 9:47-62, 11:7-25; *see id.*, 6:25-32, 8:58-61, 9:20-25; Wechselberger, ¶98. As discussed *infra* § IX.A.1.c.ii, the received keys' respective expiration times are conveyed in response to a playback request and parsed. Wechselberger, ¶99.

**ii. Peterka Teaches a Playback Request Response
Comprising “Content Information” for Multiple
Segments**

As discussed, each ECM/EMM in Peterka is parsed for the required content information in Element [1B]. Wechselberger, ¶100. As explained *supra* § IX.A.1.b, Peterka discloses sending multiple ECMs/EMMs in response (*playback request response*) to a given client-initiated request to access media (*media playback request*). POSITAs understood that in the *playback request response* each ECM/EMM contains a CK and the corresponding *identifiers* and *expiration time* described *supra* §§ IX.A.1.c.i(2)-IX.A.1.c.i(3). Peterka, [0121]; Wechselberger, ¶100. Thus, Peterka satisfies the requirement that each “*playback request response*” includes multiple “*content encryption keys*,” “*content encryption key identifiers*,” and “*content encryption key expiration times*,” as recited. Wechselberger, ¶100.

To the extent that PO asserts no single ECM/EMM meets the requirement for multiple “*content encryption keys*,” “*content encryption key identifiers*,” and

“*content encryption key expiration times*,” such an argument places undue significance on Peterka’s individual message and ignores that Peterka teaches sending *more than one* key (and thus more than one ECM/EMM) *in response* to a given client-initiated request to access media and that these two or more ECMs/EMMs are the “*playback request response*.” Wechselberger, ¶101. Further, as discussed *supra* § IX.A.1.b, it would have been obvious to package multiple ECMs/EMMs in one multicast message that included multiple “*content encryption keys*,” “*content encryption key identifiers*,” and “*content encryption key expiration times*,” as claimed. *Id.*

d. [1C] “retrieving content and manifest files from a content delivery server;”

Peterka discloses a device (e.g., client computer 112) *retrieving content from a content delivery server* (e.g., origin 108, a content exchange 116, or caching server). Peterka, [0038] (“[C]lient computer 112 is shown interacting with the active directory 104 to select a content object for downloading.”), [0039]-[0040] (“[O]rigin servers 108 determine the preferred source to direct the client computers in order to download content objects.... That source can be origin processor 108 itself or one of the content exchanges 116.”), [0060] (“Origin Content (OCS)—Server at a content provider’s computer that provides content.”), [0071] (OCS “becomes part of the distribution network”), [0083], [0135], FIGs. 1, 19; Wechselberger, ¶102.

Peterka does not explicitly describe “manifest files.” However, manifest files were well-known as “conventional” features of media “streaming platforms” like those taught by Peterka which stream, e.g., segments of MPEG-4 format media at various bit rates. Peterka, [0053]-[0054], [0119]; Wechselberger, ¶¶103-104 (citing Lewis (EX-1011), 1:28-29, 2:64-3:4, 7:6-9, 7:38-45). Manifests provided, e.g., “information about the encodings available from” a media streaming system “and fragments stored by the system up to the current time,” and other useful information that clients could access and use to help facilitate streaming. Wechselberger, ¶¶105-106 (citing Bocharov, [0023]-[0024], [0030]-[0031]; Pestoni (EX-1012), 8:39-9:19; Sood (EX-1013), Abstract, [0019]-[0020], [0046]-[0047]; Chen050 (EX-1014), 15:31-36, 15:66-16:8; Stockhammer (EX-1015), 134-135; Van Lancker (EX-1016), Abstract, §§ 1, 3; 3GPP-AHS (EX-1017), § 12.2.5; Pantos (EX-1018), 4-5). The ’178 specification lists manifest file formats that were well-known or in development in 2011—“m3u8,” “Smooth Streaming,” “DASH MPD.” EX-1001, 4:16-20, 7:56-59, 10:57-60; Wechselberger, ¶¶107-108 (citing Stockhammer, 133-135; Van Lancker, Abstract, §§ 1, 3; 3GPP-AHS, § 12.2.5; Pantos, 4-5). Moreover, in the 1376 Investigation, PO concedes Element [1C] encompasses conventional “manifest files”—e.g., a “MPD” file describing “different resolutions and bandwidth for each media file.” EX-1033, 13-14.

Peterka renders obvious clients retrieving conventional manifest files from content delivery servers. Wechselberger, ¶109. Peterka discloses that media (e.g., in MPEG-4 format) may be available at different “quality (i.e., bit rates)” and describes the “server... adjust[ing] the rate in real time based on the immediate state of the network,” and the “client... adjust[ing] the bit rate... in accordance with user selection.” Peterka, [0054], [0119]; *see also id.*, [0059]. Based on Peterka’s disclosures regarding bit-rate adjustments, it would have been obvious to implement Peterka’s client to retrieve from the content delivery server both the content itself and manifest files describing the content, which at minimum included information regarding available “quality (i.e., bit rates)” that would allow the disclosed bit-rate adjustments. Wechselberger, ¶109.

Moreover, this would have been obvious because it was conventional for content servers to (1) create manifest files with information regarding how a particular media segment is encoded (e.g., bit rates in multi-bit-rate streaming services) and (2) send such information to clients. Wechselberger, ¶¶104-110 (citing Bocharov, [0024], [0030]-[0031]; Lewis, 1:28-29, 2:64-3:4, 7:6-9; Pestoni, 8:39-9:19; Sood, Abstract, [0019]-[0020], [0046]-[0047]; Chen050, 15:31-36, 15:66-16:8; Stockhammer, 134-135; Van Lancker, Abstract, §§ 1, 3; 3GPP-AHS, § 12.2.5; Pantos, 4-5). POSITAs would have reasonably expected success implementing Peterka to include such functionality. *Id.*

It also would have been obvious to implement Peterka's client, which retrieves media streams divided into segments (Peterka, [0080]-[0082], [0091]-[0093], [0099]-[0101], [0122]-[0125], FIGs. 7A-7B), to retrieve *multiple* manifest files from the content server, with each subsequent manifest file describing subsequent corresponding media segments. Wechselberger, ¶110. It was conventional for servers to send clients updated manifests with updated information that may not have been available when the client initially accessed a media stream. *Id.* (citing Bocharov, [0023]-[0025], [0032]). Sending a new manifest file describing information regarding a segment (along with the segment itself) would have beneficially reduced the complexity of parsing manifest files at the client and associating various data with multiple segments. *Id.* POSITAs would have reasonably expected success with such an implementation. *Id.*

e. [1D] “detecting content encryption key rotation boundaries between periods of use of different content encryption keys in decrypting retrieved content;”

The '178 specification indicates that, for media segments encrypted by successive keys, “key rotation boundaries” occur whenever a new key is required to decrypt a segment—regardless of whether the key change is due to normal period-based key expiration or some other reason. *See* EX-1001, 7:29-33; Wechselberger, ¶111. The patent describes detecting such boundaries by: (1) checking the expiration time of the current key (normal period-based expiration) or (2) checking for an

explicit key change notification/signal. Wechselberger, ¶111; EX-1001, 9:47-49, 10:39-42, 11:2-7, 11:38-45. Element [1D] covers either type of “detecting.” Wechselberger, ¶111.

As discussed *supra* § VIII.A, Peterka discloses a pull model for key distribution, where “each client ***keeps track of the keys and their expiration times*** and actively requests new keys before the current keys expire so as to avoid service interruptions.” Peterka, [0105]. “[T]he server need not monitor the status of the client’s keys; rather, the responsibility of ***determining when a new key is required***” is “passed to the client.” *Id.*, [0106]. Peterka’s clients “***monitor***,” e.g., “the expiration time of the program segment key and send an individual key update request before the key expires.” *Id.*, [0114]; *see also id.*, [0080]-[0082] (“The client will ***monitor*** the expiration time for each pay period.”). Peterka thus discloses clients that ***detect*** upcoming ***content encryption key rotation boundaries*** based on monitored key expiration times that allow the client to request new keys ahead of the upcoming boundary. Wechselberger, ¶112. These detected boundaries occur ***between periods of use of different content encryption keys*** because each content encryption key is used to encrypt a different media segment (Peterka, [0080]-[0082], [0091]-[0093], [0099]-[0101], [0122]-[0125], FIGs. 7A-7B), in the same way the ’178 specification describes. Wechselberger, ¶113; EX-1001, 2:22-30, 1:45-56.

PO previously argued that Peterka does not meet [1D] because [1D] excludes detecting “normal period-based expiration” of keys. EX-1027, 12-13. The IPR2022-00618 panel correctly rejected this interpretation, finding that Peterka’s pull-model-clients—which “keep[] track of... keys and their expiration times”—do indeed meet [1D]. EX-1028, 20-23; Wechselberger, ¶114. The Board should again reject any narrow interpretation that excludes time-based boundary detection.

Even if PO’s interpretation is accepted, Peterka meets [1D] because Peterka teaches boundary detection through use of “signaling” that allows a client to determine when to “begin *utilizing* [a] new content key.” Peterka, [0116]-[0117]; Wechselberger, ¶115. Peterka teaches using “a predetermined bit” in the content stream “to indicate if an old or current content key should be used as opposed to a new content key which has recently been distributed to the client.” Peterka, [0117]. Peterka’s client “*check[ing]*” this “predetermined bit in a packet” to “determine the proper content key to use” *additionally meets* [1D]’s “detect[ion],” because the bit signals a key change. Wechselberger, ¶115; Peterka, [0116]-[0121], FIG. 14.

PO’s previous attorney-argument suggesting signaling is exclusive to Peterka’s push model (EX-1027, 20-21) is incorrect. Peterka’s teachings regarding “signaling” *when to use* keys (Peterka, [0116]-[0121]) are *complementary* to Peterka’s pull model for acquiring keys (*id.*, [0105]-[0106]). When to use keys and when to acquire keys are orthogonal concepts; signaling when to use a key in the

client's possession *does not require* Peterka's server to *push* updated keys. Wechselberger, ¶¶115-116 (citing, for corroboration, Suzuki, [0013], [0016], [0043]-[0048], [0059]-[0060], [0076]-[0077], FIGs. 2, 5, describing system with pull key distribution *and* key change signaling in content stream). At minimum, using Peterka's key change signaling with Peterka's pull model would have been obvious. Wechselberger, ¶117; Peterka, [0121] (ECM/EMM includes *both* key lifetime *and* parity bit); *see also, e.g.*, Suzuki, [0016].

- f. [1E] “issuing requests to a license server ahead of a key rotation boundary to retrieve a second content encryption key to be used after a content encryption key rotation boundary is reached; and”

Peterka discloses a client issuing a request ahead of the expiration of a current key (*ahead of a key rotation boundary*) to retrieve a new key (*retrieve a second content encryption key*)⁵ that is, as discussed *infra* § IX.A.1.g, *to be used after a content encryption key rotation boundary is reached*. Peterka, [0105]-[0106] (“Under the pull model, each client keeps track of the keys and their expiration times

⁵ The ITC agreed with PO that [1D]/[16E] and [1E]/[16F] “do not need to be performed in the recited order.” EX-1032, 21-25. Regardless, Peterka's timer-based detection (*supra* § IX.A.1.e) indisputably occurs *before* (and facilitates) its requests “to retrieve a second content encryption key”—meeting the claims even if [1D]/[16E] *must* occur before [1E]/[16F]. Wechselberger, ¶118 n.7.

and actively *requests* new keys *before the current keys expire so as to avoid service interruptions.*”); Wechselberger, ¶¶118-119; *supra* §§ IX.A.1.b-IX.A.1.c. Peterka discloses issuing such requests to a “key distribution server” (*license server*). Peterka, [0110] (“[E]ach client requests a new key and is individually responded to by *the key distribution server.*”); *cf.* EX-1001, 3:44-45 (“[A] license server is responsible for *distributing* content encryption *keys to clients.*”).

The claim appears to recite *multiple* “request[s] to a license server... to retrieve a second content encryption key,” though the ’178 specification only discloses a client issuing one request to a license server for each new key that it needs. EX-1001, 11:21-25 (“client 110 must issue a new content encryption key request to the license server 106 to retrieve the new content encryption key”), 11:3-62, FIG. 3; *see also id.*, 1:45-50 (“[E]ach content encryption key is provided upon expiration of a period of use of a serially preceding content encryption key.”), 9:63-65 (describing two requests to obtain a new key, but only one to a license server); Wechselberger, ¶120. In the 1376 Investigation, PO interprets Element [1E] to encompass issuing one license request for each key that is needed, consistent with the specification. *See* EX-1033, 14-15; Wechselberger, ¶120. POSITAs would have understood Peterka—in which a client issues a series of (multiple) requests for keys to decrypt upcoming media segments during a stream—meets this limitation at least to the same extent it

is described in the '178 Patent. Peterka, [0080]-[0082], [0105]-[0106]; Wechselberger, ¶121.

Additionally, clients participating in media distribution systems commonly re-issued requests to a server if a first request failed due to a timeout, a server error, or transmission failure. Wechselberger, ¶122 (citing EX-1019, EX-1020). To ensure that a client would have a next key available in the event of a failed request, it would have been obvious to implement Peterka to re-issue a request for a second content encryption key if a first request failed for any reason, and POSITAs would have reasonably expected success with such an implementation. *Id.*

g. [1F] “applying the second key for content decryption after the key rotation boundary is reached.”

As discussed *supra* § IX.A.1.f, Peterka discloses its client actively requesting a new key (the *second content encryption key* in [1E]) from a key distribution server *before* the current key expires (the *key rotation boundary*), to avoid service interruptions. Wechselberger, ¶123; Peterka, [0101], [0105]-[0106], [0110]. POSITAs understood the *second content encryption key* recited in [1E] is *the second key for content decryption* recited in [1F]. Wechselberger, ¶123. Peterka discloses *applying* the new key it requested (*the second key for content decryption after the key rotation boundary is reached*—e.g., after the previous key expires. Peterka, [0106] (“decryption by the client” occurs “by utilizing the cryptographic key that was requested” prior to previous key’s expiration), [0116] (“[T]he content

key is utilized for decrypting the program content throughout the course of the distribution of the program content.... [W]hen a new content key is implemented, the implementation will be signaled to the clients so that the clients can ***begin utilizing the new content key with which they have been provided***"); see also *id.*, [0090] ("receive keys for decrypting the encrypted portion of the program"), [0092] ("keys are required to decrypt the remainder of the program content"), [0108] ("the program content is distributed for decryption by the client utilizing the transmitted key"); Wechselberger, ¶124; *supra* § VIII.A.

2. **Claim 2: "The method of claim 1, wherein the content encryption keys and content encryption key identifiers returned in the playback request response include the content encryption key and associated identifier currently being applied as well as a future content encryption key and associated identifier yet to be applied."**

As described *supra* §§ IX.A.1.b-IX.A.1.c, Peterka discloses or renders obvious a client receiving multiple ***content encryption keys***—for a ***current*** and next (***future***) segment, each in one ECM/EMM—and identifiers for those keys (***associated identifiers***). Peterka, [0080] ("keys to the current as well as the next segment"), [0130], FIG. 7B; Wechselberger, ¶125. Peterka thus meets claim 2.

3. Claim 3: “The method of claim 1, wherein the content encryption key expiration returned in the playback request response is expressed as an expected minimum interval between periodic key rotation.”

Peterka discloses media streams that are divided into “purchasable segments” to allow users to pay according to the amount of time—e.g., number of segments—that the user consumes. Peterka, [0078]-[0082], [0099]-[0101]. As discussed *supra* §§ VIII.A, IX.A.1.c, each segment is encrypted by a different content key. Peterka also indicates that segments have a defined *period* that POSITAs would have understood corresponds to *periodic key rotation*. *Id.*, [0081]-[0082] (“client monitor[s] the expiration time for each pay *period*” and correspondingly “request[s] a new set of keys from the” server), [0110] (client is “*periodically* given new keys” and “server... issue[s] new keys at regular intervals”); Wechselberger, ¶¶126-127.

The ’178 specification leaves to the POSITA how to “*express[]*” an “expiration” “as an *expected minimum interval* between periodic key rotation.” Wechselberger, ¶128. As discussed *supra* § IX.A.1.c.i(3), Peterka discloses distributing a “key and *its expiration time*” to a client and that each key distribution message (ECM/EMM) includes a field for the “*time remaining* in the lifetime of the key” provided in the message. Peterka, [0082], [0105]-[0106], [0121], FIG. 8; *supra* § IX.A.1.c; Wechselberger, ¶129. Peterka thus teaches key expiration times expressed as a remaining *amount* of time in the key’s lifetime—e.g., number of minutes/seconds (an *interval*)—instead of a specific time-of-day at which the key

expires (e.g., 12:00am). Peterka, [0121]; Wechselberger, ¶129. Given Peterka’s disclosure of periodically-distributed keys that are issued “at regular intervals” (Peterka, [0082], [0110]), POSITAs would have understood that the disclosed expiration time in each ECM/EMM (Peterka, [0121]) constitutes a “minimum interval” that the key in the ECM/EMM is “expected” to be used “between” periodic key changes. Wechselberger, ¶130 (explaining that Peterka’s key would not have been “expected” to be used for *less* than the time remaining in the key’s lifetime before the next key change, because that would cause service interruptions). Peterka’s expirations are thus *expressed as an expected minimum interval between periodic key rotation*, as claimed. *Id.*

Alternatively, this would have been obvious. Wechselberger, ¶131. The question of exactly how to “express[]” key “expiration”—e.g., as an expected *minimum* interval (between periodic key rotation), an expected *maximum* interval, or both—is an obvious matter of design choice that was (1) within the skill of POSITAs to implement, and (2) not discussed in the specification, let alone assigned any particular importance or criticality. *Id.* (noting that the specification does not once use “expected,” “minimum,” or “interval”); *Ex Parte Spangler*, Appeal No. 2018-003800, Decision, at 4-8 (PTAB Feb. 20, 2019) (informative).

For example, POSITAs had reasons to use (*express*) Peterka’s “expiration time[]”/“time remaining in the lifetime of the key” (Peterka, [0082], [0105]-[0106],

[0121]) as both an *expected minimum interval between periodic key rotation* and an expected maximum, such that keys were expected to expire and rotate on a fixed, periodic basis that defines a minimum (and maximum) interval between periodic key rotation—and would have reasonably expected success doing so. Wechselberger, ¶¶132-133. This, for example, would have predictably helped ensure that keys were used by the client in sync with the key rotation occurring at the encrypting server, and helped increase security by ensuring clients could not use keys after their expected expirations. Wechselberger, ¶133.

Likewise, POSITAs had reasons to express Peterka’s “expiration time[]”/“time remaining in the lifetime of the key” (Peterka, [0082], [0105]-[0106], [0121]) as only an *expected minimum interval between periodic key rotation*—as opposed to an expected *maximum* interval, where a client would be prohibited from applying a key in its possession after the time remaining in the key’s lifetime goes to zero—and would have reasonably expected success doing so. Wechselberger, ¶133. This, for example, would have predictably helped ensure that keys were used by the client in sync with the key rotation occurring at the encrypting server while also providing flexibility, allowing Peterka’s clients to continue using an already-acquired key in scenarios where it may be deemed unnecessary to rotate the key (e.g., when a final key rotation boundary occurs near the end of a program). *Id.*

At the very least, either option—two of three finite options (expirations expressed as minimum, maximum, or both)—would have been obvious to try. Wechselberger, ¶134; *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 421 (2007).

4. Claim 4: “The method of claim 3, further comprising: timing the prefetching of a next un-retrieved content encryption key based on an expected expiration of the content encryption key currently being used.”

The ’178 specification uses “prefetching” to encompass acquiring a key before it is needed. EX-1001, 11:12-18, 11:48-53; Wechselberger, ¶135. As discussed *supra*, Peterka discloses its client “keep[ing] track of”/“monitor[ing]” key expiration times and requesting a new key (*a next un-retrieved content encryption key*) before the new key is needed (*prefetching*), based on when (and before) “current keys expire” (*based on an expected expiration of the content encryption key currently being used*). Peterka, [0105]-[0106], [0082]; *supra* §§ IX.A.1.e-IX.A.1.f, IX.A.3; Wechselberger, ¶¶135-136.

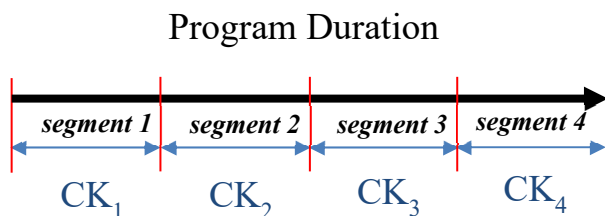
5. Claim 5: “The method of claim 4, further comprising: prefetching a next un-retrieved key a fixed duration before the expected expiration of the content encryption key currently being used.”

Claim 5 depends from claim 4, requiring that a “prefetching” of “a next un-retrieved key” occurs at “a fixed duration before the expected expiration” of the currently-used key. Wechselberger, ¶137. There were a limited number of ways to implement precisely *when* Peterka’s client acquires new keys. Wechselberger,

¶138. One conventional, straightforward way to implement Peterka’s client—which, as discussed *supra*, already receives expiration times expressed as the “time remaining” in a key’s lifetime (Peterka, [0121]), “keeps track of”/“monitor[s]” key “expiration times[,] and actively requests new keys before the current keys expire so as to avoid service interruptions” (*id.*, [0105])—was to have the client request new keys a specific amount of time (a ***fixed duration***) before the time remaining in the key’s lifetime reached zero (“before the current keys expire” (*id.*)). Wechselberger, ¶¶138-139 (citing Suzuki, [0036], [0059], [0076]-[0077], FIGs. 4-6). Moreover, POSITAs would have appreciated that consistently acquiring new keys a specific amount of time (a ***fixed duration***) before the current keys expire was a way “to avoid service interruptions” (Peterka, [0105]) because this would have helped ensure new keys were requested with sufficient time for servers to respond to large numbers of client key requests prior to key rotation boundaries—e.g., by helping predictably account for server delay (delays that occur while processing client requests) and network latency (delays that occur when data travels across a network) with a reasonable expectation of success. Peterka, [0105]; Wechselberger, ¶140.

6. **Claim 6:** “The method of claim 4, further comprising: prefetching a next un-retrieved key within a period of time after the expected expiration of the content encryption key currently being applied, the period of time beginning at the expected expiration of the content encryption key currently being used, and ending at a duration calculated as a fixed percentage of a fixed periodic content encryption key expiration interval.”

Claim 6 depends from claim 4, requiring that a prefetching of “a next un-retrieved key” occurs *after* the current key expires (“within a period of time after the expected expiration of the content encryption key currently being applied”). Wechselberger, ¶141. Peterka’s system, which meets claim 5 as discussed *supra* § IX.A.5, also meets claim 6. Wechselberger, ¶142. The diagram below—depicting a content stream divided into segments of equal duration and encrypted by a series of content keys (CK₁, CK₂, etc.)—helps illustrate why.



In Peterka, a client acquires CK₂ during segment 1, *before* CK₁ expires. Peterka, [0105]-[0106] (a client “actively requests new keys before the current keys expire”). A client later acquires CK₃ during segment 2, *after* CK₁ but *before* CK₂ expires. *Id.* Both CK₂ and CK₃ are *prefetched* because they are acquired before they are needed. Wechselberger, ¶143; EX-1001, 11:12-18, 11:48-53.

Claims 4-6 each recite “a next un-retrieved key.” Claims 5-6 *do not* recite the *same* “next un-retrieved key” or “the next un-retrieved key” that was recited in claim 4. Wechselberger, ¶144. Thus, claim 5 encompasses a client prefetching CK₂ (“a next un-retrieved key”) during segment 1 (*before* the current key, CK₁, expires), and claim 6 encompasses prefetching CK₃ (a different “next un-retrieved key”) during segment 2 (*after* the current key, CK₁, expires).⁶ Wechselberger, ¶144.

As discussed for claim 5 (*supra* § IX.A.5), there were a limited number of ways to implement when Peterka’s client acquires new keys, of which one was a “fixed duration” before the current key’s expiration. Wechselberger, ¶145. In such an implementation (with reference to the diagram and example described above), that would have entailed acquiring CK₂ a fixed duration before the current key CK₁’s expiration, satisfying claim 5. Wechselberger, ¶¶142, 145.

In such an implementation, POSITAs would *also* have understood that CK₃ was acquired “*within*” “*the period of time beginning at the expected expiration of the content encryption key currently being used,*” (*after* CK₁’s expiration) “*and ending at*” a fixed duration *before* the next key rotation boundary (*before* CK₂’s

⁶ This is a different interpretation of claim 6 than the one presented by Apple in IPR2022-00618, which did not reconcile the “prefetching” and “expiration” limitations. EX-1026, 40-41.

expiration). Wechselberger, ¶146. Acquiring CK₃ at a fixed duration *before* CK₂'s expiration—which, for Peterka's periodic key rotation (Peterka, [0082], [0110], would be a fixed percentage of the key rotation period—constituted *a duration calculated as a fixed percentage of a fixed periodic content encryption key expiration interval*, as claimed. Wechselberger, ¶146.

Alternatively, POSITAs would have had reasons to implement Peterka to time the “prefetching” of a new key (discussed *supra* § IX.A.4) based on a fixed percentage of a fixed periodic key expiration interval, which would have met this claim. Wechselberger, ¶147. Such an implementation would have enabled Peterka's clients to seek new keys for every key rotation period, no matter how short the rotation period was, and would have ensured that clients could prefetch keys even for high-security media streams in which keys were rotated very quickly. *Id.* A POSITA would have reasonably expected success with such an implementation. *Id.*

7. Claim 10: “The method of claim 1, wherein license server communications are secured and authenticated using a selected one of Secure Sockets Layer and a token-based technique using a token encrypted with a symmetric key.”

Claim 10 requires “license server communications” to be “secured and authenticated using a *selected one*” of two methods for securing communications with a license server: (1) “Secure Sockets Layer” and (2) “a token-based technique using a token encrypted with a symmetric key.” Wechselberger, ¶148. The '178 Patent specification does not describe any embodiment that uses both methods, and

the claim lists no other alternatives. *Id.* Thus, only one listed alternative must be satisfied to meet the claim. MPEP § 2117 (“Claims that set forth a list of alternatives from which a selection is to be made are typically referred to as Markush claims [and] any claim that recites alternatively usable members, regardless of format, should be treated as a Markush claim.”). Both alternatives were obvious. Wechselberger, ¶¶148-153.

Secure Sockets Layer (“SSL”) was a well-known standard for securing communications channels using encryption (*see* EX-1021), used by numerous websites in (and before) 2011. Wechselberger, ¶149. Peterka discloses its client-server communications occurring over the Internet. Peterka, [0004], [0015], [0038], [0135]. POSITAs would have had reasons to implement Peterka’s client-server communications over the Internet using SSL, as this was a prevailing standard for Internet-based communications, and a conventional, established, predictable option for preventing eavesdropping, tampering, or message forgery. Wechselberger, ¶¶149-150. A POSITA would have reasonably expected success with such an implementation. *Id.*

Peterka also discloses the claimed “token-based technique.” Wechselberger, ¶151. A POSITA understood the claimed ***token-based technique*** to encompass exchanging encryption key information in an encrypted form (***token***). *Id.* (citing EX-1001, 6:57-58; Barrus (EX-1022), 1:26-39, 5:18-25). Peterka discloses a

framework for securing communications between clients and license servers by requiring clients to present a security “ticket” to the server when requesting access to content or fetching keys from the server. Peterka, [0066] (“The client delivers the secure object and a ticket with its own entitlement data... to the caching server.... If all [content access] rules are satisfied, the caching server will grant access to the requested content...”), [0075], [0136], [0142]-[0144], FIGs. 19-21. Peterka explains that the ticket or similar data is “authenticated and encrypted so that the client cannot revise his own entitlements.” *Id.*, [0146], FIG. 23. The client ticket thus provides an encrypted security “token” allowing the server to authenticate a client and determine whether to grant a request for keys to access streaming media content. Wechselberger, ¶152.

Peterka does not expressly detail whether its client ticket (or other token) is encrypted with a *symmetric key*, though this would have been an obvious option because “symmetric” encryption was one of two common encryption options at the time of the alleged invention, and one that did not require the creation of public-private key pairs or the maintenance of more than one key. Wechselberger, ¶153 (citing Kelly (EX-1010), [0004]). A POSITA would have reasonably expected success implementing Peterka to use symmetric keys to encrypt its client ticket (or other token). Wechselberger, ¶153.

8. Claim 12: “The method of claim 1, wherein actual content encryption key rotation boundaries are detected based on real-time in-band notifications.”

As discussed *supra* § IX.A.1.e, Peterka discloses a client checking content packets for a signal to start applying a new key, which meets PO’s narrow interpretation of the “detect” limitation. Peterka, [0116]-[0121], FIG. 14; Wechselberger, ¶154. Such detection was “based on real-time in-band notifications,” because the signals are embedded in Peterka’s content packets (*in-band*) and occur when the client needs to switch keys for decryption (in *real-time*). Wechselberger, ¶155; EX-1001, 4:13-15. This also constitutes detection of “*actual* content encryption key rotation boundaries” because a different key is needed for decryption before and after the signal. Wechselberger, ¶155.

9. Claim 16

a. [16PRE] “A computerized device operable as a client for handling secure distribution of content, comprising:”

Peterka discloses a device (*computerized device*) operable as a client for handling secure distribution of content. Peterka, [0038]-[0042], FIG. 1 (“client computer 112”), FIG. 2 (describing “individual system elements” of devices including client computer 112); *supra* § IX.A.1.a; Wechselberger, ¶156.

- b. [16A] “memory operative to store computer program instructions; one or more processors; input/output interface circuitry; and interconnect circuitry coupling the memory, processors and input/output interface circuitry together,”**

Peterka discloses how “individual system elements from FIG. 1,” which includes “client computer 112,” may be implemented with system components like those depicted in FIG. 2. Peterka, [0041], FIG. 2; *see also id.*, [0038]-[0043], FIG. 1; Wechselberger, ¶157. Peterka discloses that such a system (e.g., client computer 112) includes memory operative to store computer program instructions (e.g., working memories 209/291/292/293), one or more processors (e.g., CPU(s) 201, processing acceleration 207), input/output interface circuitry (e.g., I/O devices 202/203), and interconnect circuitry coupling the memory, processors, and input/output interface circuitry together (e.g., bus 208). Wechselberger, ¶157.

c. [16B]-[16G]

Peterka further provides that the processors are operative to execute computer program instructions from the memory to cause the client device to perform operations described in Elements [16B]-[16G], for the same reasons described *supra* for Elements [1A]-[1F], as summarized below. Wechselberger, ¶158.

Claim 16 Element	Corresponding Claim 1 Element	Section
[16B]	[1A]	§ IX.A.1.b
[16C]	[1B]	§ IX.A.1.c
[16D]	[1C]	§ IX.A.1.d
[16E]	[1D]	§ IX.A.1.e
[16F]	[1E]	§ IX.A.1.f
[16G]	[1F]	§ IX.A.1.g

10. Claims 17-18, and 20

Peterka’s computerized device respectively satisfies Claims 17-18 and 20 for the same reasons described in the corresponding sections listed below.

Wechselberger, ¶159.

System Claim	Corresponding Method Claim	Section
17	2	§ IX.A.2
18	4	§ IX.A.4
20	12	§ IX.A.8

X. GROUND 2: PETERKA IN VIEW OF SUZUKI RENDERS CLAIMS 1-6, 10, 12, 16-18, AND 20 OBVIOUS

To the extent the “identifiers” recited in Element [1B] or the timing requirements for the “prefetching” recited in claims 4-6 are not considered disclosed or rendered obvious by Peterka alone, Suzuki expressly teaches those limitations. Wechselberger, ¶160. Modifying Peterka in view of Suzuki separately renders obvious claims 1-6, 10, 12, 16-18, and 20. *Id.*

A. Peterka-Suzuki

As discussed *supra* § IX.A.1.e, Peterka discloses clients acquiring “keys” for multiple media segments, “keep[ing] track of”/“monitor[ing]” keys “and their expiration times,” and “determining when a new key is required.” Peterka, [0080]-[0082], [0105]-[0106], [0114]. Peterka leaves to POSITAs how to keep track of keys and their expiration times, and how to determine *when* a new key is required. Suzuki teaches such known details, describing an implementation of such functionality using “key identifiers” for keeping track of keys, and using a client’s clock signal to request a new key a fixed amount of time before the current key expires. Suzuki, [0071]-[0084].

Like Peterka, Suzuki teaches providing a client with “key information” regarding a “key currently in use (key A) and the next key to be used (key B).” *Id.*, [0072]-[0077], FIGs. 5-6; Wechselberger, ¶¶161-162. Suzuki teaches acquiring this information by submitting a “key information request” to a server, which responds with a “key response message” that includes (1) the current and next keys (“key A,” “key B”), (2) each key’s “key number” (“key number A,” “key number B”), and (3) each key’s “remaining validity time.” Suzuki, [0072]-[0077], FIGs. 5-6. Suzuki includes teachings regarding how to use its “remaining validity time” and “key number.”

“Remaining validity time” is used to determine when to request new keys. Suzuki, [0076]-[0077]. To “eliminat[e]... confirmation delay” during key changes—e.g., due to client-server “time lag” (*id.*, [0008])—Suzuki teaches implementing a client to submit a “request” for a new key when “the remaining validity time” of the current key “reaches” a specific, greater-than-zero “setting value.” *Id.*, [0076]. The client calculates this “setting value” as a “certain percentage of the time from the initial value” of the “remaining validity time of the key.” Suzuki, [0070]. The client knows when this “setting value” is reached by “updat[ing] the remaining validity time” for each key in its possession using the client’s “clock signal[]” and performing “processing” that includes “recogni[zing]” that “the value of the remaining validity time has decreased and reached a [particular] setting value.” Suzuki, [0033], [0036], [0059], FIG. 4; Wechselberger, ¶163. Thereby, a client can request and “obtain key information regarding the next key to be used *before the key is changed.*” Suzuki, [0080]-[0081].

“Key number” is used to “identify[] the key,” allowing a client to associate keys with their respective “remaining validity time[s]” (e.g., in a table) and to help the client keep track of keys, for example when determining which key to apply when decrypting a segment of content. *Id.*, [0030], [0043]-[0048], [0080], FIG. 2; Wechselberger, ¶164. A POSITA would have recognized the importance of such

identifiers to differentiate amongst keys, and that providing unique values (such as key numbers) to keys would ensure that the system does not confuse keys. *Id.*

Peterka and Suzuki are in the same field of art. Wechselberger, ¶165. Like Peterka (as discussed *supra* § IX.A.3), Suzuki discloses “periodically” changing keys that protect distributed content. Suzuki, [0003], [0012]; Peterka, [0082], [0110]. POSITAs would have had reasons to implement Peterka’s client in view of Suzuki to acquire and use “key information” about a next key—its “remaining validity time” and “key number”—at a set time *before* the expiration of the current key, and thereby facilitate timely key acquisition and use. Suzuki, [0076]-[0077], [0080]-[0081], [0043]-[0048]; Wechselberger, ¶166. This would have allowed a POSITA to “eliminat[e] the confirmation delay” during key changes (Suzuki, [0011]), and realize Peterka’s embodiment where “each client keeps track of the keys and their expiration times and actively requests new keys before the current keys expire so as to avoid service interruptions.” Peterka, [0105]. Because Suzuki expressly teaches a specific way to achieve what Peterka expressly discloses (Peterka, [0082], [0105]), a POSITA would have reasonably expected success applying Suzuki’s teachings to Peterka. Wechselberger, ¶¶166-167.

B. Claim-by-Claim Analysis

1. Claims 1-2, 10, 12, 16-18, and 20

As discussed, Peterka-Suzuki applies Suzuki's teachings regarding "key numbers" and "remaining validity times" to Peterka. Peterka-Suzuki's client receives a "key number" and "remaining validity time" corresponding to each received "content encryption key" described *supra* §§ IX.A.1.b, IX.A.1.c.i(1). Wechselberger, ¶168. Peterka-Suzuki uses the "key number" to keep track of keys and the "remaining validity time" to determine when to request a next key. Suzuki, [0043]-[0048], [0072]-[0077], [0080]-[0081], FIGS. 2, 5-6; Wechselberger, ¶169.

Peterka-Suzuki's "key numbers" uniquely identify keys and are *content encryption key identifiers*, as claimed. Wechselberger, ¶170.

Peterka-Suzuki's "remaining validity times"—which correspond to Peterka's "time remaining in the lifetime of the key" in an ECM/EMM (Peterka, [0121]) and are used to request new keys before the end of a current key's lifespan—are *content encryption key expiration times*, as claimed. Wechselberger, ¶170.

By sending requests for additional keys whenever the remaining lifetime/validity time of a current key reaches a particular "setting value," Peterka-Suzuki "detect[s]" key rotation boundaries as claimed in Elements [1D]/[16E] (*in addition to* the signaling-based detection described *supra* § IX.A.1.e, citing Suzuki) and "issu[es]" requests to a license server *ahead of a key rotation boundary* as

claimed in [1E]/[16F]. Suzuki, [0033], [0036], [0059], [0072]-[0077], FIGs. 4-6; Wechselberger, ¶171.

For the remaining elements in claims 1 and 16, and the dependent claims listed below, Peterka-Suzuki meets each element/claim for the same reasons Peterka alone meets them, as summarized below. Wechselberger, ¶172.

Claim/Element	Section
[1PRE]	§ IX.A.1.a
[1A]	§ IX.A.1.b
[1B]	<i>Supra</i> § X.B.1
[1C]	§ IX.A.1.d
[1D]	§ IX.A.1.e; <i>supra</i> § X.B.1
[1E]	<i>Supra</i> § X.B.1
[1F]	§ IX.A.1.g
[2]	§ IX.A.2
[10]	§ IX.A.7
[12]	§ IX.A.8
[16PRE]	§ IX.A.9.a
[16A]	§ IX.A.9.b
[16B]	§ IX.A.1.b
[16C]	<i>Supra</i> § X.B.1
[16D]	§ IX.A.1.d
[16E]	§ IX.A.1.e; <i>supra</i> § X.B.1
[16F]	<i>Supra</i> § X.B.1
[16G]	§ IX.A.1.g
[17]	§ IX.A.2
[18]	§ IX.A.4
[20]	§ IX.A.8

2. Claims 3-6

As discussed *supra* §§ X.A, X.B.1, Peterka-Suzuki's client uses Suzuki's "remaining validity time" as Peterka's "time remaining in the lifetime of the key"

(Peterka, [0121]) for the received content encryption key in an ECM/EMM, and uses Suzuki's method of "recogni[zing]" that "the value of the remaining validity time has decreased and reached a" particular "setting value" to determine when to acquire new keys. Suzuki, [0033], [0036], [0059], [0072]-[0077], FIGs. 4-6; Wechselberger, ¶173. In Peterka-Suzuki, consistent with both references, keys are changed periodically (Suzuki, [0003], [0012]; Peterka, [0082], [0110]) and Peterka-Suzuki's use of Suzuki's "remaining validity time" would have met claim 3 (or would have been obvious to implement in ways that met claim 3) for the same reasons described *supra* § IX.A.3. Wechselberger, ¶174.

Further, Peterka-Suzuki's client acquires a new key before it is needed (***times the prefetching of a next un-retrieved content encryption key***) when the "remaining validity time" of the current key reaches a fixed "setting value." *E.g.*, Suzuki, [0076]-[0077]. A POSITA understood that this disclosed prefetching is both ***based on*** (claim 4) and occurring ***a fixed duration before*** (claim 5) ***the expected expiration of the content encryption key currently being used***, as recited. Wechselberger, ¶175.

Moreover, in view of Suzuki's teachings, POSITAs would have had reasons to implement Peterka-Suzuki's periodic "prefetching"—which occurs a set amount of time before each expiration of a series of periodically-changing keys—to occur no later than "at a duration calculated as a fixed percentage of a fixed periodic

content encryption key expiration interval” (claim 6). Wechselberger, ¶176. As noted above, Suzuki expressly teaches a client calculating the “setting value” at which it requests a new key as a “*certain percentage* of the time from the initial value” of the “remaining validity time of the key.” Suzuki, [0070]; Wechselberger, ¶176.

XI. GROUND 3: PETERKA IN VIEW OF BOCHAROV RENDERS CLAIMS 1-6, 10, 12, 14, 16-18, AND 20 OBVIOUS

To the extent retrieving manifest files from a content delivery server is not considered disclosed or rendered obvious by Peterka alone, modifying Peterka in view of Bocharov would have rendered obvious claims 1-6, 10, 12, 14, 16-18, and 20. Wechselberger, ¶177.

A. Peterka-Bocharov

As discussed *supra* § VIII.A, Peterka describes techniques for distributing streaming media content to clients. Wechselberger, ¶178. Servers are configured to stream encrypted media content to clients, which apply decryption keys to playback content. Peterka, [0002], [0004], [0011], [0038], [0078]-[0082], [0091]-[0093], [0127], FIG. 7A. Peterka also teaches streaming content at different “quality (i.e., bit rates),” which the client can “purchase” and “adjust... in accordance with user selection.” *Id.*, [0054]; Wechselberger, ¶179.

As discussed *supra* § VIII.C, Bocharov discloses techniques that allow a client to obtain fragments of media “in whichever encoding the client chooses,”

using “manifest files” that are provided with the media fragments. Bocharov, [0049]; *see also id.*, [0007], [0014], [0016], [0020]-[0023], [0049]-[0062], FIG. 5.

Both references stream similar types of media (e.g., MP4 files) and apply segmentation to divide media into fragments for distribution. Peterka, [0080], [0119]; Bocharov, [0020], [0034], [0038]-[0039]; Wechselberger, ¶¶180-181.

In view of Bocharov, a POSITA would have been motivated to implement Peterka’s client to issue a client manifest request to a server to obtain manifests (Bocharov, [0053], FIG. 5)—transmitted e.g., in the media fragments that also contain the media content (*id.*, [0023]-[0024])—which (1) included information allowing a client to request delivery of streaming media content, using, e.g., a URL (*id.*, [0030]-[0031]); (2) identified different “quality” or “bit rate” encodings that the client could request or switch to (*id.*, [0030]-[0031], [0048]-[0050], [0060]); and (3) provided a list of fragments in the media stream with information indicative of a URL for each fragment, so that individual clients could reference manifests in the media stream to request timely delivery of fragments at a desired encoding and maintain a continuous stream (*id.*). Wechselberger, ¶182.

Implementing Peterka’s system to use Bocharov’s manifests would have provided an effective solution for enabling clients to timely request delivery of streaming media content—e.g., at a particular encoding/quality/bit rate—and maintain a continuous stream. Wechselberger, ¶183. Such an implementation using

manifests would have met claims 1-6, 10, 12, 14, 16-18, and 20 for the reasons discussed *infra* §§ XI.B.1, XI.B.3.

Implementing Peterka’s system to use Bocharov’s manifests would also have provided an effective solution for allowing clients to switch between multiple versions of a media program having different encodings/qualities/bit rates. Wechselberger, ¶184. Peterka acknowledged the benefit of providing different qualities of streams that could accommodate changing user demands and network conditions, and Bocharov’s manifests would have allowed Peterka’s system to embed, within media segments/fragments provided to clients, both media content and information needed to switch between different encodings/qualities/bit rates of the media stream. Peterka, [0054]; Bocharov, [0049], [0053]. Indeed, Bocharov itself expressly describes the use of manifests to improve tiered-quality purchasing models similar to Peterka’s (Peterka, [0054])—e.g., when explaining that a “system may enable or disable switching to particular bit rates based on whether the conditions of the payment model have been met” by including needed “information” “in the manifest provided to the client.” Bocharov, [0060]; Wechselberger, ¶184. Such an implementation—in which Bocharov’s manifests are used to enable switching between encodings/qualities/bit rates—would additionally have met claims 12 and 20 for the reasons discussed *infra* § XI.B.2.

B. Claim-by-Claim Analysis

1. Claims 1-6, 10, 12, 16-18, and 20

Elements [1C]/[16D] recite “[retrieve/retrieving] content and manifest files from a content delivery server.” For the reasons discussed *supra* § XI.A, Peterka-Bocharov applies Bocharov’s manifest file teachings to Peterka’s system, such that Peterka-Bocharov’s clients retrieve the manifests taught in Bocharov as part of the media stream, and the manifests include encoding information that the client may request or switch to. Wechselberger, ¶185.

As discussed *supra* § IX.A.1.d, Peterka discloses a client (e.g. client computer 112) that *retrieves content from a content delivery server* (e.g., origin 108, a context exchange 116, or caching server). Peterka, [0038]-[0039], [0040], [0060], [0071], [0083], [0135], FIG. 1. This is consistent with Bocharov’s teachings of clients like Peterka’s which retrieve content from a content delivery server by requesting media “fragments.” Bocharov, [0006]-[0007], [0015], [0021]-[0022]; *supra*, § VIII.C; Wechselberger, ¶¶186-187.

Bocharov expressly discloses *retrieving manifest files from a content delivery server*. Bocharov, [0053] (“The origin server 510 receives a manifest request 530 from a client 515” which “provides a client manifest response 540 to the client 515.”), FIG. 5; *see also id.*, [0023]-[0025], [0030]-[0035], [0049]-[0052]; *supra*, § VIII.C; Wechselberger, ¶188. Bocharov also expressly teaches retrieving

multiple manifest files, e.g., updated manifests with information not initially available that are transmitted with subsequent media fragments. Bocharov, [0024] (“subsequent manifests provided with each media fragment”), [0032] (“the server may send the fragment along with manifest information about the next few fragments”), [0023]-[0025]. Thus, Peterka-Bocharov’s client *retrieves content and manifest files from a content delivery server*, as claimed in Elements [1C]/[16D]. Wechselberger, ¶189.

For the remaining elements in claims 1 and 16, and the dependent claims listed below, Peterka-Bocharov meets each element/claim for the same reasons Peterka alone meets them, as summarized below. Wechselberger, ¶190.

Claim/Element	Section
[1PRE]	§ IX.A.1.a
[1A]	§ IX.A.1.b
[1B]	§ IX.A.1.c
[1C]	<i>Supra</i> § XI.B.1
[1D]	§ IX.A.1.e
[1E]	§ IX.A.1.f
[1F]	§ IX.A.1.g
[2]	§ IX.A.2
[3]	§ IX.A.3
[4]	§ IX.A.4
[5]	§ IX.A.5
[6]	§ IX.A.6
[10]	§ IX.A.7
[12]	§ IX.A.8
[16PRE]	§ IX.A.9.a
[16A]	§ IX.A.9.b
[16B]	§ IX.A.1.b
[16C]	§ IX.A.1.c

[16D]	<i>Supra</i> § XI.B.1
[16E]	§ IX.A.1.e
[16F]	§ IX.A.1.f
[16G]	§ IX.A.1.g
[17]	§ IX.A.2
[18]	§ IX.A.4
[20]	§ IX.A.8

2. Claims 12 and 20

Claims 1 and 16 recite detecting “content encryption boundaries between periods of use of different content encryption keys in decrypting retrieved content”—while claims 12 and 20 recite detecting “*actual* content encryption key rotation boundaries.” A POSITA understood claims 12 and 20 at minimum encompass separate detection of “actual boundaries” that occurs *in addition to* the detection recited in claims 1 and 16. Wechselberger, ¶191.

In addition to the boundary detection discussed *supra* § IX.A.1.e, Peterka-Bocharov alternatively meets claims 12 and 20 because Peterka-Bocharov’s client separately detects content encryption key boundaries based on information in Bocharov’s manifest files that tell the client to switch to a different encoding of a media stream. Wechselberger, ¶192. As discussed *supra* § XI.A, Peterka-Bocharov’s manifest cause a client to switch to a different encoding (and different quality) of the media stream due to, e.g., failure to meet payment requirements. Bocharov, [0060], [0048]-[0050]; Wechselberger, ¶193. Such a use of Peterka-Bocharov’s manifests constituted *real-time in-band notifications*, as claimed,

because Bocharov’s manifests are transmitted as part of the media content stream along with corresponding media content (*in-band*)—Bocharov, [0007] (“server embeds incremental information in media fragments that eliminates the usage of a typical control channel”); EX-1001, 4:13-15—and the client would have to switch to a different quality encoding of the stream upon receiving such a notification (in *real-time*). Wechselberger, ¶194; *see also* Bocharov, [0016], [0025], [0032].

It was conventional for different encodings of a media stream (Peterka, [0054]; Bocharov, [0049], [0060]) to be encrypted using different keys. Wechselberger, ¶195; Yuan (EX-1023), 517-519 (“To support access to a video sequence at different quality levels in non-scalable coding, the same video would have to be compressed into separate files of different qualities, with each file encrypted with a different encryption key.”). Thus, a notification in a manifest instructing a client to switch to a different encoding for a stream would have satisfied claim 12’s requirement of *actual key rotation boundaries*, because switching to a different encoding for a stream entailed needing a new key to decrypt content, and therefore corresponded to an actual key rotation boundary. Wechselberger, ¶196; Pestoni, 8:39-63 (known to obtain a manifest in advance to secure decryption information); Lewis, 7:38-45 (known to use a manifest file to manage keys dealing with time windows and rental periods for “enhanced security.”). The ’178 specification itself acknowledges “protocols” in 2011 that “employ[ed] segment-

based rate adaptation to deal with fluctuations in bandwidth, *whereby segment boundaries provide natural demarcation points for switching bitrates*” (EX-1001, 2:31-37; Wechselberger, ¶196), and the proposition that actual key rotation boundaries occur when a client switches between encodings/qualities is consistent with PO’s broad infringement arguments in the 1376 Investigation.

To the extent such notification is considered insufficient to meet claims 12 and 20, it would have been obvious to implement Peterka-Bocharov’s manifest to provide information needed for clients to detect the need for and acquire the keys necessary to switch to a different encoding. Wechselberger, ¶197. This would have been consistent with the “pull method” for key distribution used in Peterka-Bocharov, in which clients actively request new keys, and a POSITA would have reasonably expected success with such an implementation. *Id.* (citing Peterka, [0053], [0078]-[0082], [0105]-[0106], [0110]).

3. Claim 14: “The method of claim 12, wherein content encryption key rotation boundaries are detected based on notifications present in the content manifest file, the content manifest file being selected from an m3u8 file, an IIS Smooth Streaming manifest file, a DASH MPD file and a proprietary manifest file.”

A POSITA understood that claim 14 encompasses an interpretation where the term “content manifest file” corresponds to one of the “manifest files” recited in independent claim 1, from which claim 14 depends. Wechselberger, ¶198. Claim 14 requires the recited content manifest file to be “*selected from*” a closed list of

alternative manifest file types. *Id.* Only one listed alternative must be satisfied to meet the claim. MPEP § 2117 (“Claims that set forth a list of alternatives from which a selection is to be made are typically referred to as Markush claims [and] any claim that recites alternatively usable members, regardless of format, should be treated as a Markush claim.”).

Bocharov teaches that its manifest file can be an Internet Information Server (“IIS”) smooth streaming manifest file. Bocharov, [0036], [0038], [0010], FIG. 2; *see also id.*, Abstract (“smooth streaming system”), [0014] (same), [0030] (describing a “typical client manifest” that includes “SmoothStreamingMedia” fields). POSITAs would have had reasons to apply Bocharov’s use of an IIS Smooth Streaming manifest file in Peterka-Bocharov since Bocharov confirms that such a manifest file was suitable to carry the fragment information of the types (e.g., quality information) disclosed in Bocharov. Wechselberger, ¶¶199-200; Bocharov, [0030]. Further, a POSITA would have recognized benefits of using a known manifest file format that could be parsed by clients according to the dictates of the IIS format and the teachings of Peterka and Bocharov. *Id.* This would have been a simpler and more convenient alternative to using, e.g., a new manifest file format that was not defined in a standard. *Id.*

XII. GROUND 4: PETERKA IN VIEW OF SUZUKI AND BOCHAROV RENDERS CLAIMS 1-6, 10, 12, 14, 16-18, AND 20 OBVIOUS

To the extent retrieving manifest files from a content delivery server is not considered obvious over Peterka-Suzuki (discussed *supra* § X), modifying Peterka-Suzuki in view of Bocharov—for the same reasons a POSITA would have modified Peterka alone in view of Bocharov, discussed *supra* § XI.A—would separately have rendered obvious claims 1-6, 10, 12, 14, 16-18, and 20. Wechselberger, ¶201.

A. Peterka-Suzuki-Bocharov

As discussed *supra* § X.A, a POSITA had reasons (and would have been motivated) to implement Peterka’s client in view of Suzuki to acquire and use “key information” about a next key—its “remaining validity time” and “key number”—at a set time *before* the expiration of the current key, thereby facilitating timely key acquisition and use. Suzuki, [0076]-[0077], [0080]-[0081], [0043]-[0048]; Wechselberger, ¶¶202-204.

For the same reasons discussed *supra* § XI.A, in view of Bocharov’s teachings, a POSITA would have been motivated to further modify Peterka-Suzuki’s client to issue a client manifest request to a server to obtain manifests that include information that enable the client to request media and switch between different quality encodings. Bocharov, [0049]; *see also id.*, [0007], [0014], [0016], [0020]-[0023], [0049]-[0062], FIG. 5; *supra* §§ VIII.C. The same reasoning described for Peterka alone applies to Peterka-Suzuki because the content manifest teachings of

Bocharov are complementary to and do not conflict with the key identifier and key acquisition teachings of Suzuki. Wechselberger, ¶204. A POSITA would also have reasonably expected success with an implementation combining Peterka-Suzuki and Bocharov for the same reasons described *supra* §§ X.A and XI.A, for each individual combination. *Id.*

B. Claim-by-Claim Analysis

1. Claims 1-6, 10, 12, 14, 16-18, and 20

Peterka-Suzuki-Bocharov applies (to Peterka's system) Suzuki's teachings regarding key numbers and remaining validity times and Bocharov's teachings regarding manifest files. Wechselberger, ¶205. Peterka-Suzuki-Bocharov's client receives a "key number" and "remaining validity time" that correspond to each received "content encryption key" described *supra* §§ IX.A.1.b, IX.A.1.c.i(1), and uses the "key number" taught by Suzuki to keep track of keys and the "remaining validity time" to determine when to request a new key. Wechselberger, ¶205; Suzuki, [0043]-[0048], [0072]-[0077], [0080]-[0081], FIGs. 2, 5-6; *supra* § X.B.1. The client separately retrieves the manifests taught in Bocharov as part of the media stream, and the manifests include information regarding available stream encodings that the client may request or switch to. Wechselberger, ¶206; Bocharov, [0023]-[0025], [0030]-[0035], [0049]-[0053], FIG. 5; *supra* § XI.B.1.

Peterka-Suzuki-Bocharov meets each element/claim for the same reasons described in the sections summarized below. Wechselberger, ¶207.

Claim/Element	Section
[1PRE]	§ IX.A.1.a
[1A]	§ IX.A.1.b
[1B]	§ X.B.1
[1C]	§ XI.B.1
[1D]	§§ IX.A.1.e, X.B.1
[1E]	§ X.B.1
[1F]	§ IX.A.1.g
[2]	§ IX.A.2
[3]	§§ IX.A.3, X.B.2
[4]	§§ IX.A.4, X.B.2
[5]	§§ IX.A.5, X.B.2
[6]	§§ IX.A.6, X.B.2
[10]	§ IX.A.7
[12]	§§ IX.A.8, XI.B.2
[14]	§ XI.B.3
[16PRE]	§ IX.A.9.a
[16A]	§ IX.A.9.b
[16B]	§ IX.A.1.b
[16C]	§ X.B.1
[16D]	§ XI.B.1
[16E]	§§ IX.A.1.e, X.B.1
[16F]	§ X.B.1
[16G]	§ IX.A.1.g
[17]	§ IX.A.2
[18]	§§ IX.A.4, X.B.2
[20]	§§ IX.A.8, XI.B.2

XIII. GROUND 5: PETERKA IN VIEW OF BOCHAROV, PETERKA308, AND CHEN RENDERS CLAIMS 7-8 AND 19 OBVIOUS

Implementing Peterka-Bocharov in view of Peterka308 and Chen would have rendered obvious claims 7-8 and 19, which recite requirements related to content encryption key identifiers. Wechselberger, ¶208.

A. Peterka-Bocharov-Peterka308-Chen

Peterka contemplates a client acquiring keys for multiple future media segments and using *multi-bit* signals embedded in streaming media packets to select the correct key from among those keys. Peterka, [0117] (“*if* a single bit is used”), [0118]-[0121]; Wechselberger, ¶209. At minimum, acquiring keys for multiple future segments would have been obvious to reduce the “load” of client requests at segment boundaries. Peterka, [0080], [0105]; Wechselberger, ¶209 (citing, for corroboration, Pestoni, 12:1-46).

As discussed *supra* § VIII.D, Peterka308 discloses techniques for correlating content encryption keys and key identifiers with information indicating a location in a media stream where the content encryption key should be applied. Wechselberger, ¶210. Peterka308 proposes using a “time stamp” to derive the appropriate key for a “segment” of encrypted media content, where the “time stamp” indicates the “relative time value” from the beginning of the media content to the start of the segment. Peterka308, [0025]. In Peterka’s system, which distributes keys to clients before streaming segments that require those keys, a POSITA would have

sought to apply Peterka308's suggested use of segment locators (e.g., timestamps) as content encryption key identifiers to provide a more robust method of signaling to clients which key (e.g., which CK) to apply at the start of each segment, and would have reasonably expected success doing so. Wechselberger, ¶211. This would be beneficial, e.g., for a client that held keys for multiple future segments, because it could readily compare the current playback location of a media stream to the respective location information of the future segment keys to determine which key to apply at the next key rotation boundary. *See* Peterka, [0117]-[0121]; Peterka308, [0025]. Implementing Peterka308's teaching would have beneficially allowed clients to more effectively manage keys obtained for segments multiple periods in advance. Wechselberger, ¶211.

To the extent Peterka does not expressly disclose what criteria content servers apply to subdivide media content into "segments" (where each segment is encrypted by a different key), Chen discloses criteria that were known for this task. *Supra*, § VIII.E. Chen teaches a first option ("fixed crypto period") defining "a crypto time interval and chang[ing] the key each time the crypto time-interval passes." Chen, [0034], [0037]-[0038], FIG. 2A. A second option ("fixed number of packets") involves determining "a number of content packets 'n' corresponding to a suitable time interval and chang[ing] the cryptographic key every 'n' packets." *Id.*, [0034], [0039]-[0041], FIG. 2B. A third ("fixed 'marker' count") involves using "a suitable

MPEG-II field type as a ‘marker,’ such as an I-frame header,” and “chang[ing] the cryptographic key every time ‘n’ markers have passed in the stream.” *Id.*, [0034], [0042]-[0044], FIG. 2C.

Each option was a viable design alternative that was obvious to try in view of the limited number of methods available to specify locations in a media stream. Wechselberger, ¶¶212-215. Each option also would have beneficially allowed the encoding and decoding devices to reference a common metric that could be obtained/used to select an appropriate key at each boundary at the appropriate time, corresponding to the same location in the media content. *Id.* Clients would thus have an additional basis for determining and predicting upcoming key rotations, improving the reliability of the system in the event of errors such as corrupted bits in RTP packets. *Id.* Moreover, when distributing content encryption key identifiers that reference locations of segment boundaries according to the teachings of Peterka and Peterka³⁰⁸, it would have been obvious for the identifiers to relate to the corresponding segmentation method applied based on the teachings of Chen—e.g., identifiers that indicate the elapsed time of the media stream according to a crypto time interval, a number of content packets, or a number of markers (e.g., I-frames) passed in the stream. *Id.*

B. Claim-by-Claim Analysis

1. **Claims 7 and 19: “The [method of claim 1 / computerized device of claim 16] wherein subsequent content encryption key identifiers are predictable based on a predetermined known progression.”**

Peterka308 discloses “time stamp”-based content encryption key identifiers which a POSITA understood were *predictable based on a predetermined known progression* (e.g., every “five minutes has elapsed”). Peterka308, [0024], [0029], [0032]. Chen further discloses techniques for segmenting media content based on predetermined known progressions such as fixed time intervals, fixed number of packets in a content stream, and fixed “marker” count (e.g., fixed number of I-frames passed in an MPEG-II stream). Chen, [0034]-[0044], FIGs. 2A-2C; *supra* § VIII.E.

For the reasons described *supra* § XIII.A, it would have been obvious to combine Peterka, Bocharov, Peterka308, and Chen such that subsequent content encryption key identifiers would indicate locations in a media stream that the corresponding content encryption key should apply based on a timestamp, time interval, packet count, or marker count—each of which is *predictable based on a predetermined known progression* as claimed, at least to the same extent this limitation is described in the ’178 Patent, which describes assigning identifiers “based off” of known progressions such as “sequential integer value[s]” and “wall clock time offset[s].” Wechselberger, ¶¶216-218; *cf.* EX-1001, 5:63-6:1.

2. **Claim 8: “The method of claim 7, further comprising: the identifier being calculated as monotonically increasing sequential integer values based on the number of segments or video frames generated during a fixed periodic content encryption key expiration interval.”**

Chen discloses techniques for segmenting media content based on criteria such as a count of frame “markers” (e.g., I-frames) that pass in a video stream. Chen, [0034]-[0044], FIGs. 2A-2C; *supra* § VIII.E. As described above (§ XIII.A), a POSITA had reason to implement Peterka-Bocharov-Peterka308-Chen such that content encryption key identifiers indicate locations in a media stream that a content encryption key should be applied to ***based on*** a count of frame “markers” passed in the media stream. Wechselberger, ¶¶219-220. A POSITA understood the frame marker count provides an identifier that is ***calculated as monotonically increasing sequential integer values*** (because frame number is an integer that increases over time) and ***based on a number of video frames generated during a fixed periodic content encryption key expiration interval*** (e.g., corresponding to the “fixed” marker count of Chen). Wechselberger, ¶221.

XIV. NO BASIS EXISTS FOR DISCRETIONARY DENIAL

There is no basis for denial under 35 U.S.C. § 325. The references applied herein were not cited during prosecution of the ’178 Patent. *Advanced Bionics, LLC v. MED-EL Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6 (PTAB Feb. 13, 2020) (precedential).

There is no basis for denial under 35 U.S.C. § 314. The EDNC litigation is stayed pending resolution of the ITC’s 1376 Investigation. EX-1034. Additionally, “the PTAB no longer discretionarily denies petitions based on applying *Fintiv* to a parallel ITC proceeding.” EX-1035, 1-2, 5-7. Moreover, *Fintiv* Factors 1-5—concerning effects on (and of) parallel district court litigation—**all** favor institution because ***there is no parallel litigation*** between Petitioner Google and PO. Google has neither challenged the validity of the ’178 Patent nor been sued. Further, the Petition’s merits are compelling (Factor 6), which “alone demonstrates that the PTAB should not discretionarily deny institution under *Fintiv*.” EX-1035, 4-5.

Denial under § 314 in view of IPR2022-00618—a proceeding in which the Board instituted on similar grounds presented herein, using similar combinations (Grounds 3-4), but terminated prior to Final Written Decision due to settlement—is also unwarranted. IPR2022-00618 did not involve Petitioners or any party with a “significant relationship under *Valve, General Plastic* factor one.” *Videndum Production Solutions, Inc., v. Rotolight Ltd.*, IPR2023-01218, Paper 12, at 5-6 (PTAB April 19, 2024) (vacating denial of institution on *General Plastic* grounds where petitioner was not part of prior IPR). Indeed, because any substantive overlap with IPR2022-00618 involves grounds previously instituted upon but left unresolved, the prior proceeding, if anything, urges final determination of patentability for these claims.

XV. CONCLUSION

The Board should institute review and cancel the Challenged Claims.

Dated: October 1, 2024

Respectfully submitted,
Petitioners

/Scott A. McKeown/
Scott A. McKeown, Reg. No. 42,866
WOLF, GREENFIELD & SACKS, P.C.

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24, the undersigned certifies that the foregoing Petition for *Inter Partes* Review contains 13,983 words excluding a table of contents, a table of authorities, Mandatory Notices under § 42.8, a certificate of service or word count, or appendix of exhibits or claim listing. Petitioners have relied on the word count feature of the word processing system used to create this paper in making this certification.

Date: October 1, 2024

/Dara Del Rosario/

Dara Del Rosario

Paralegal

WOLF, GREENFIELD & SACKS, P.C.

CERTIFICATE OF SERVICE UNDER 37 C.F.R. § 42.6 (E)(4)

I certify that on October 1, 2024, I will cause a copy of the foregoing document, including any exhibits or appendices filed therewith, to be served via Overnight FedEx at the following correspondence address of record for the patent:

Ericsson Inc.
6300 Legacy Drive
M/S EVR 1-C-11
Plano, TX 75024

Date: October 1, 2024

/Dara Del Rosario/
Dara Del Rosario
Paralegal
WOLF, GREENFIELD & SACKS, P.C.