

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TREND MICRO, INC.
Petitioner

v.

TAASERA LICENSING LLC

Case No. IPR-2023-00801
Patent No. 8,327,441

**PETITION FOR INTER PARTES REVIEW
OF U.S. PATENT NO. 8,327,441**

Filed on behalf of Trend Micro, Inc.
By: Robert M. Hansen
Scott B. Amankwatia
Matthew J. Anderson
The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703) 391-2900
Fax: (703) 391-2901

TABLE OF CONTENTS

I.	Introduction.....	1
II.	Grounds for Standing (37 C.F.R. § 42.104(a)).....	2
III.	Identification of Challenge (37 C.F.R. § 42.104(b)).....	2
A.	Citation of Prior Art	2
B.	Statutory Grounds for Challenge.....	3
IV.	The '441 patent.....	3
A.	Claim of Priority of the '441 patent	3
B.	Overview of the '441 patent.....	3
C.	Summary of the Prosecution History	9
D.	Level of Ordinary Skill in the Art.....	10
E.	Claim Construction.....	10
V.	HOW THE CLAIMS ARE UNPATENTABLE.....	11
A.	Overview of Asserted Prior Art	11
i.	Munetoh.....	11
ii.	Rajan.....	13
B.	Ground 1: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious in view of Munetoh and Rajan.....	16
C.	Motivation to Combine Munetoh and Rajan.....	16
1.	Claim 1	18
	[1a.] A method of providing an attestation service for an application at runtime executing on a computing platform using an attestation server, comprising:	18
	[1b.] receiving, by the attestation server remote from the computing platform:	21

[1c.] a runtime execution context indicating attributes of the application at runtime, wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application;.....	22
[1d.] a security context providing security information about the application, wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components;	24
[1e.] generating, by the attestation server, a report indicating security risks associated with the application based on the received runtime execution context and the received security context, as an attestation result; and.....	27
[1f.] sending, by the attestation server, the attestation result associated with the application.....	31
2. Claim 2	32
[2a.] generating, by the attestation server, an application artifact as a reference for changes in a subsequent execution context; and	32
[2b.] sending the generated application artifact such that subsequent changes to the runtime execution context are tracked based on the generated application artifact.	33
3. Claim 3	35
[3a.] The method according to claim 2, further comprising digitally signing the attestation results:.....	35
[3b.] wherein the attributes further comprise parent-child process associations of the application.	36
4. Claim 4	37

	The method of claim 1, wherein the received security context is an introspective security context, and wherein the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components.....	37
5.	Claim 5	39
	The method of claim 4, wherein the generating of the report indicating security risks associated with the application includes generating, by the attestation server, one or more security assertions that pertain to the received runtime execution context and the received introspective security context.	39
6.	Claim 6	40
	The method according to claim 1, further comprising authenticating the application using a plurality of collaboration services.	40
7.	Claim 7	41
	The method according to claim 1, further comprising controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results.....	41
8.	Claim 9	42
	The method according to claim 1, further comprising providing confidence metrics in the attestation results indicating a level of security risk by different classifications such that a restriction on a user's transaction with the application are applied based on the level of security risk indicated by the confidence metrics in the attestation results.....	42
D.	Ground 2: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious over Rajan in view of Munetoh.....	43
E.	Motivation to Combine Rajan and Munetoh.....	44

1. Claim 1	45
[1a.] A method of providing an attestation service for an application at runtime executing on a computing platform using an attestation server, comprising:.....	45
[1b.] receiving, by the attestation server remote from the computing platform:.....	47
[1c.] a runtime execution context indicating attributes of the application at runtime, wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application;.....	48
[1d.] a security context providing security information about the application, wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components;	49
[1e.] generating, by the attestation server, a report indicating security risks associated with the application based on the received runtime execution context and the received security context, as an attestation result; and.....	51
[1f.] sending, by the attestation server, the attestation result associated with the application.....	53
2. Claim 2	55
[2a.] generating, by the attestation server, an application artifact as a reference for changes in a subsequent execution context; and	55
[2b.] sending the generated application artifact such that subsequent changes to the runtime execution context are tracked based on the generated application artifact.	56
3. Claim 3	58

[3a.] The method according to claim 2, further comprising digitally signing the attestation results:.....	58
[3b.] wherein the attributes further comprise parent-child process associations of the application.	58
4. Claim 4	59
The method of claim 1, wherein the received security context is an introspective security context, and wherein the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components.....	59
5. Claim 5	59
The method of claim 4, wherein the generating of the report indicating security risks associated with the application includes generating, by the attestation server, one or more security assertions that pertain to the received runtime execution context and the received introspective security context.	59
6. Claim 6	60
The method according to claim 1, further comprising authenticating the application using a plurality of collaboration services.	60
7. Claim 7	60
The method according to claim 1, further comprising controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results.....	60
9. Claim 9	61
The method according to claim 1, further comprising providing confidence metrics in the attestation results indicating a level of security risk by different classifications such that a restriction on a user's transaction with the application are applied based on the level of	

**security risk indicated by the confidence metrics in the attestation
results.....61**

VI. NO BASIS FOR DISCRETIONARY DENIAL.....61

VII. CONCLUSION 63

VIII. Mandatory Notices (37 C.F.R. § 42.8(a)(1)).....63

TABLE OF AUTHORITIES

Cases

<i>Advanced Bionics, LLC v. MED-EL Elektromedizinische Gerate GmbH</i> , IPR2019-01469, Paper 6 (Feb. 13, 2020).....	63
<i>Cellco Partnership v. Huawei Device Co.</i> , IPR2020-01117, Paper 10 (Feb. 3, 2021).....	64
<i>Graham v. John Deere Co.</i> , 383 U.S. 1 (1966)	16, 44
<i>KSR Int’l Co. v. Teleflex, Inc.</i> , 550 U.S. 398 (2007)	16, 45
<i>Phillips v. AWH Corp.</i> , 415 F.3d 1303 (Fed. Cir. 2005) (en banc)	10
<i>Wellman, Inc. v. Eastman Chem. Co.</i> , 642 F.3d 1355 (Fed. Cir. 2011)	11

Statutes

35 U.S.C. § 102(b)(pre-AIA)	2
35 U.S.C. § 103(a)	passim
35 U.S.C. § 325(d)	61

Rules

37 C.F.R. § 42.10(a).....	64
37 C.F.R. § 42.104(a).....	2
37 C.F.R. § 42.104(b)	2
37 C.F.R. § 42.8(a)(1)	63
37 C.F.R. § 42.8(b)(2).....	64
37 C.F.R. § 42.8(b)(3).....	64
37 C.F.R. §42.100(b)	10

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 8,327,441 to Kumar <i>et al.</i>
1002	<i>Curriculum Vitae</i> of A.L. Seth Nielson, Ph.D.
1003	Expert Declaration of Dr. Seth Nielson under 37 C.F.R. § 1.68.
1004	Prosecution History of '441 to Kumar <i>et al.</i>
1005	Seiji Munetoh, <i>Integrity management Infrastructure for Trusted Computing</i>, IEICE TRANS. INF. & SYST., Vol. E91-D, No. 5, 1242-1251, (May 2008).
1006	Hridesh Rajan, <i>Tisa: Toward Trustworthy Services in a Service-Oriented Architecture</i>, IEEE Transactions on Services Computing, 201-213, (October-December 2008).
1007	Declaration of Ingrid Hsieh-Yee, Ph.D.
1008	Steve Anderson, <i>Web Services Trust Language (WS-Trust)</i>, (February 2005).
1009	George Moncrief, <i>ezHPC Security Architecture</i>, IEEE Computer Society, (2006).
1010	<i>Taasera Licensing LLC, v. Trend Micro Incorporated</i>, Plaintiff's Disclosure of Asserted Claims and Infringement Contentions with '441 Claim Chart, 2:21-cv-00441-JRG-RSP, (Jul 26, 2022).
1011	Ajay Surie, <i>Rapid Trust Establishment for Pervasive Computing</i>, IEEE Computer Society, 24-30, (2007).
1012	Minjin Kwon, <i>PROBE: A Process Behavior-based Host Intrusion Prevention System</i>, Department of Computer Science and Engineering, Korea University, Seoul, (2008).

I. INTRODUCTION

Trend Micro, Inc. (“Petitioner”) petitions for *inter partes* review of claims 1-12 of U.S. Patent No. 8,327,441 to Kumar *et al.*, entitled “System and method for application attestation.” (“the ’441 patent,” EX1001), which according to PTO records, is assigned to Taasera Licensing, Inc. (“Taasera”).

As will be shown, computer security concerns were rapidly increasing and systems were in development to assure the integrity of computing networks years before the ’441 patent application was filed. A consortium of key industry players assembled as the Trusted Computing Group (TCG) were developing specifications and requirements for hardware-based architectures, software systems and modules with the goal of guaranteeing security, integrity and confidentiality of data.

EX1006, p.2. More than two years before the ’441 patent application was filed, the TCG had developed and publicized industry specifications defining an Integrity Management Infrastructure in which Platform Trust Services measure the integrity of a computing platform at runtime and produce verifiable Integrity Reports.

EX1005, p.1. More than two years after such publications, Patent Owner Taasera sought to patent essentially the same technology.

II. Grounds for Standing (37 C.F.R. § 42.104(a))

Petitioner certifies that the '441 patent is available for *inter partes* review and that Petitioner is not barred or estopped from requesting an *inter partes* review challenging the patent claims on the grounds identified in this Petition.

III. Identification of Challenge (37 C.F.R. § 42.104(b))

A. Citation of Prior Art

Petitioner cites the following prior art references.

Munetoh, et al. “Integrity Management Infrastructure for Trusted Computing”

IEICE TRANS. INF. & SYST., Vol. E91-D, No. 5, 1242-1251, (May 2008)

(“Munetoh”, EX1005) is prior art under pre-AIA § 102(b) because it was published and publicly accessible by June 16, 2008, more than one year before the earliest possible priority date of the '441 patent. EX1007. Munetoh was not identified or considered during prosecution of the '441 patent.

Rajan, et al. “Tisa: Toward Trustworthy Services in a Service-Oriented

Architecture,” IEEE Transactions on Services Computing, 201-213, (October-

December 2008) (“Rajan”, EX1006) is prior art under pre-AIA § 102(b) because it was published and publicly available no later than March 24, 2009, more than one year before the earliest possible priority date of the '441 patent. EX1007. Rajan was not identified or considered during prosecution of the '441 patent.

B. Statutory Grounds for Challenge

Petitioner requests review of '441 patent claims 1-7 and 9 (the “Challenged Claims”) on the following grounds:

GROUND 1: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious over Munetoh (EX1005) in view of Rajan (EX1006).

GROUND 2: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious over Rajan (EX1006) in view of Munetoh (EX1005).

IV. The '441 patent

A. Claim of Priority of the '441 patent

The '441 patent issued December 4, 2012, from U.S. Patent Application No. 13/399,065, filed February 17, 2012, which claimed priority to Provisional Application No. 61/443,854, filed February 17, 2011. EX1001, p.1. Solely for the purposes of this Petition, the earliest priority date for the '441 patent is taken to be February 17, 2011 (the “priority date”), so the Challenged Claims should be examined under pre-AIA statutes.

B. Overview of the '441 patent

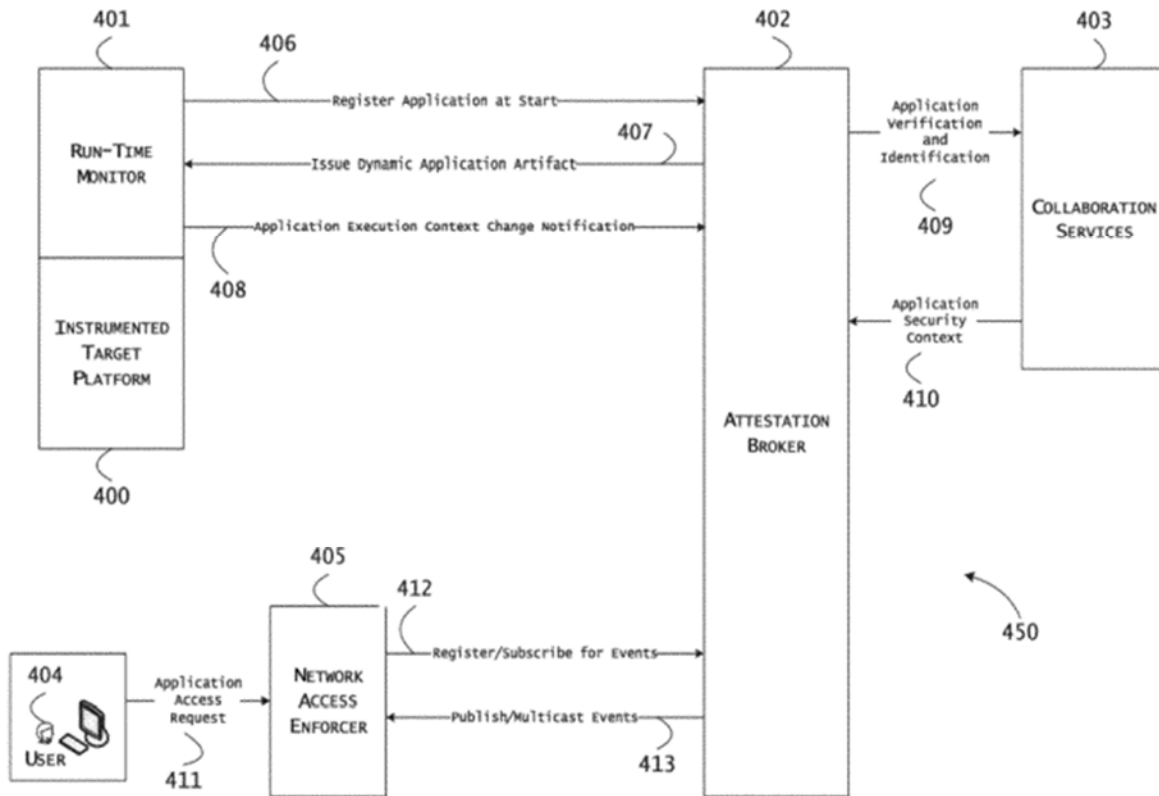
The '441 patent relates to computer security. EX1001, 1:15-18, 5:45-67; EX1003, ¶71. As background, the patent recognizes that in recent trends in

computing “enterprise software is no longer owned by the customer, but instead the Information Technology infrastructure can be provided by a third party and the software applications may be sold as service offerings.” *Id.*, 1:20-25. “Security in such environments may be an emerging challenge.” *Id.*, 5:66-67.

To address such security concerns, the ’441 patent discloses systems and methods “of providing an attestation service for an application at runtime executing on a computing platform using an attestation server.” *Id.*, 26:27-29. The application may be an application targeted for use by a computing platform operated by a user that may desire an assurance or an attestation that the application is appropriate to be used. *Id.*, 7:7-12. In an attestation service, “applications may be secured by signing-on or authenticating applications onto the network using an attestation process (e.g., via a single factor or multi-factor attestation).” *Id.*, 6:1-4. The patent discloses a “method, apparatus and/or system to establish user-to-application connections based on the dynamic attestation of applications” EX1001, 1:41-44.

FIG. 4 reproduced below illustrates the system and methods of the ’441 patent.

FIG. 4



Referring to FIG. 4, a network 450 may include an instrumented target platform 400, a runtime monitor 401, an attestation broker or attestation server 402, collaboration services 403, an interactive user 404, and/or a network access enforcer 405. *Id.*, 14:9-12,12:64-67.

The specification explains that “[t]he user 404 may establish a physical connection over network 450 and may initiate an application access request 411 to commence or initiate a transaction with a target application hosted on the instrumented target platform 401.” *Id.*, 14:12-16. “Applications running on the

instrumented target platform 400 may be inspected by the runtime monitor 401 for application execution context and state changes.” *Id.*, 14:17-19.

The “runtime monitor may generate a runtime application execution context (e.g., metadata), which may include, for example, the file hash digests and/or file attributes for loaded components of the running target application. The running target application may be registered by the runtime monitor 401 with the attestation broker 402 via a start notification 406 and the runtime application execution context (e.g., that may include the metadata) may be sent by the runtime monitor 401 to the attestation broker 402 via an application execution context change notification 408.” *Id.*, 14:27-37.

In response, the attestation broker or attestation server 402 “may verify the authenticity of the running application on the instrumented target platform 400 with a real time or near real time exchange 409 of the received metadata with one or more collaboration services 403 and the return of the application security context 410 based on the exchanged metadata.” *Id.*, 14:39-44, 12:64-67.

“A network access enforcer 405 may register or subscribe with the attestation broker 402 as a web service over a web services protocol interface 412:413 for notifications (e.g., publications) of application statements or reports (e.g., claims) for running applications (e.g., all running applications) on a plurality of instrumented target platforms 400.” *Id.* 15:5-10. “The network access enforcer

405 may query the attestation broker 402 for user specific application bindings configured for the attestation broker 402 to determine authorization controls based on dynamic multi-factor application attestation, and real time (or near real time) confidence metrics based on the local execution context 408 and assessed security context 410.” *Id.*, 1528-34. “[T]he network access enforcer 405 ... may deny access on an authenticated user based on the level of concern ... as expressed in the user-to-application policy bindings provisioned for the attestation broker 402.” *Id.*, 15:34-41.

The Challenged Claims recite a “method of providing an attestation service for an application at runtime executing on a computing platform using an attestation server.” *Id.*, 26:27-29.

In the method recited in claim 1, the attestation server receives a runtime execution context indicating attributes of the application at runtime, and a security context providing security information about the application. *Id.*, 26:30-39. The attributes comprise one or more executable file binaries of the application and loaded components of the application, and the security information comprises an execution analysis of the one or more executable file binaries and the loaded components. *Id.*

Based on based on the received runtime execution context and the received security context, the attestation server generates as an attestation result. *Id.*, 26:40-

44. The attestation server then sends the attestation result associated with the application. *Id.*, 26:44-45. The claim does not identify where the attestation result is sent.

Claim 2 expands the method to include the attestation result generating an application artifact as a reference for changes in a subsequent execution context, and sending the generated application artifact such that subsequent changes to the runtime execution context are tracked based on the generated application artifact. *Id.*, 26:46-52. Again, the claim does not identify where the generated application artifact is sent. The specification explains that “the attestation service may be configured to generate an artifact based on a runtime local execution context of the running application instance on the instrumented target platform.” *Id.*, 2:40-43. Claim 3 adds digitally signing the attestation results and limits the attributes to further including parent-child process associations of the application. *Id.*, 26:53-56.

Claim 4 clarifies that the received security context is an introspective security context, and the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components. *Id.*, 26:53-56. Claim 5 further clarifies that the attestation server generates one or more security assertions that pertain to the received runtime execution context and the received introspective security context. *Id.*, 26:63-67.

Claim 6 adds to the method authenticating the application using a plurality of collaboration services. *Id.*, 27:1-3. Claim 7 adds to the method controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results. *Id.*, 27:4-7.

C. Summary of the Prosecution History

Prosecution of the 13/399,065 application that issued as the '441 patent began with a non-final Office Action on April 27, 2012, in which claims 1-26 were rejected under 35 U.S.C. § 102(e) as unpatentable under U.S. Patent Publication 2011/0179477 to Starnes. EX1004, pp.111-119. Applicant initiated an Examiner Interview on June 7, 2012. EX1004, pp.107.

After an Examiner Interview, Taasera responded on June 29, 2012, by amending claims 1, 3-5, 13-19, and 21-26. Of relevance to the Challenged Claims, independent claim 1 was amended to recite “wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application... wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components”.

A Notice of Allowance then issued September 17, 2012, without an Examiner's statement of reasons for allowance. *Id.*, pp.48-49.

D. Level of Ordinary Skill in the Art

As established by the Declaration of Dr. Seth Nielson dated April 6, 2023 (EX1003), a person having ordinary skill in the art (PHOSITA) at the time of the earliest claimed priority date for the '948 patent would have had a bachelor's degree in computer science, or a related field, or an equivalent technical degree or equivalent work experience, and an additional one to two years of education or experience in the field of computer security topics. More education can supplement practical experience and vice versa. EX1003, ¶¶81.

E. Claim Construction

In this proceeding, claims are interpreted under the same standard applied by Article III courts (i.e., the *Phillips* standard). 37 C.F.R. §42.100(b); *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc).

So far in parallel litigation neither Taasera nor any defendants have proposed a claim construction. For purposes of this proceeding only, Petitioner proposes that all claim terms should be given their plain and ordinary meanings and no specific claim constructions are necessary to apply the asserted prior art to the Challenged Claims.¹ See *Wellman, Inc. v. Eastman Chem. Co.*, 642 F.3d 1355, 1361 (Fed. Cir. 2011) (“[C]laim terms need only be construed to the extent

¹ Petitioner reserves the right in other proceedings to construe the Challenged Claims and pursue indefiniteness theories.

necessary to resolve the controversy.”).

V. HOW THE CLAIMS ARE UNPATENTABLE

The following discussion and the accompanying evidence demonstrate that there is a reasonable likelihood that Petitioner will prevail with respect to the Challenged Claims.

A. Overview of Asserted Prior Art

Petitioner submits that claims 1-7 and 9 of the '441 patent are invalid as unpatentable under 35 U.S.C. § 103(a) in view of the combination of Munetoh and Rajan.

i. Munetoh

Munetoh is directed towards an integrity management infrastructure for trusted computing based on a standard published by the Trusted Computing Group (TCG). EX1003, ¶¶83-84; EX1005, pp.1. In Munetoh, a chain of trust is used to check the security and authenticity of various hardware and software components. EX1005, pp.1; EX1003, ¶¶84,89-91. In addition to managing the integrity of components on a device, Munetoh describes infrastructure for confirming and attesting to parties requesting the service can confirm trust in the service. EX1005, pp.4-5; EX1003, ¶¶86-87,91. Specifically, Munetoh provides an integrity management model that includes a remote verifier that provides platform trust

services (PTS). EX1005, pp.4, Fig.-1; EX1003, ¶¶87,93. Information about the trustworthiness of a platform is collected and monitored by a client (agent) installed on the platform. EX1005, pp.5-6. The system is illustrated in Figure 1 of Munetoh which is reproduced below.

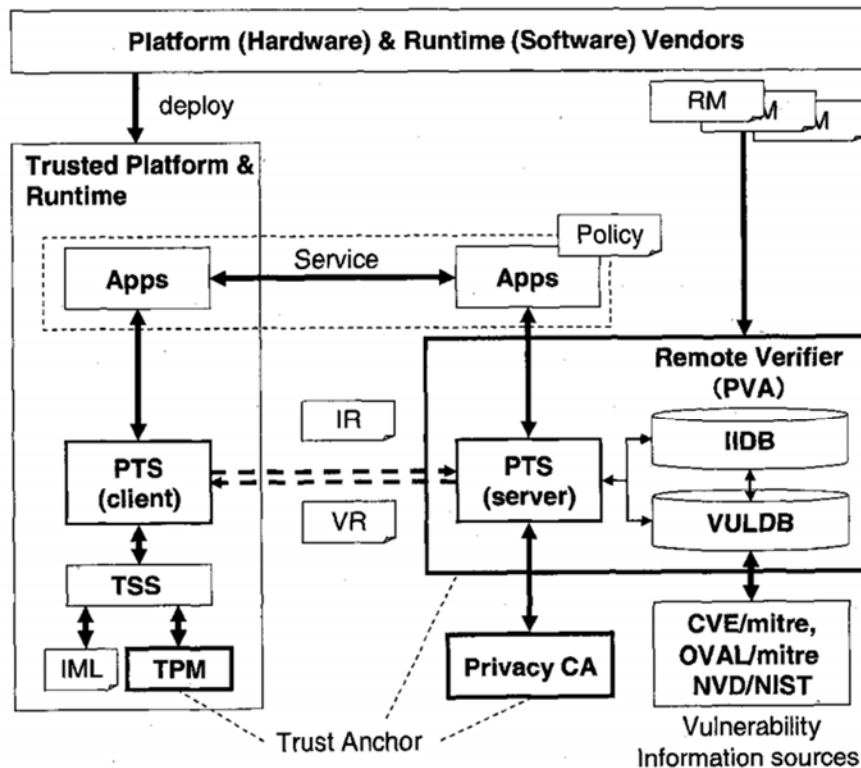


Fig.1 PTS Client/Server architecture.

The platform trust services at the remote verifier includes a finite state machine that records the change of states on a platform received in an integrity measurement log (IML). *Id.* The applications running on the server being analyzed for trust may also be measured and attested by the remote verifier by analyzing the behavior and integrity of software components that are loaded and executed. EX1005, pp.3-6. Specifically, hash values are measured for binary images of

software components and these components are verified to ensure they are known and intact. EX1005, pp.2-3. Therefore, a user (relying party) requesting application services may be provided by the platform with an attestation (validation result) generated by the remote verifier. EX1005, pp.4; EX1003, ¶¶86-87.

Informationally, the remote verifier serves as an information clearinghouse and analyzer, including receiving reference manifest from third parties that developed the hardware and software to be attested, receiving a policy that needs to be met for the relying party, receiving a platform configuration register (PCR) that is an immutable record of integrity for an application and its components, and receiving vulnerability data for applications to be analyzed on the platform. EX1005, pp.3-4,7-8. The PCR data and the IML may be combined in an integrity report which can also include security policy information. The IR is then compared to the relying party's requested security policy to see if the security of the platform meets the policy, and the relying party is informed of the result in a validation result. EX1005, pp.4-5; EX1003, ¶¶88,92-93.

ii. Rajan

Rajan is directed toward providing trustworthy services in a service-oriented architecture based on a standard published by the Trusted Computing Group (TCG). EX1006, p.2; EX1003, ¶¶95,101. This architecture is shown in Fig. 1:

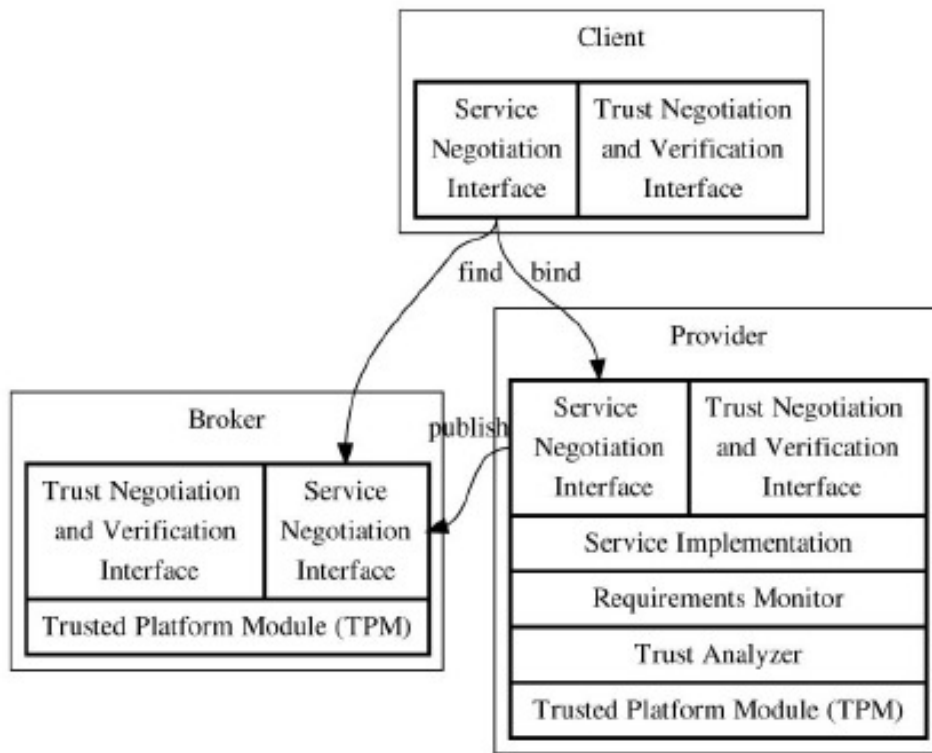


Fig. 1. Proposed architectural extensions.

EX1006, Fig. 1.

This architecture of Rajan includes components for client, provider, and broker. *Id.* Among other things, Rajan discloses “an extension of the traditional notion of a service-oriented architecture that allows clients, brokers, and providers to negotiate and validate the integrity of a requirements monitor.” *Id.*, p.1.

Rajan describes several different hashes, keys and certificates that are used to sign various hardware and software components of server providing services. EX1006, pp.3,5. Rajan proposes adding a trust negotiation and verification interface to facilitate negotiation of the desired level of trust between the

components. EX1006, p.4. Further, a requirements monitor on the service provider's side verifies conformance of the service implementation to certain desired nonfunctional requirements of the client. *Id.* The trust analyzer and the trust platform module (TPM) on the service provider together attest to the validity of the requirements monitor. EX1006, pp.4,6.

Once the monitor is attested and begins running, then the actions of programs on the service provider can be monitored and traced. EX1006, pp.3-4. This provides ongoing trust and security verification to a client device using the service or application on the attested server. EX1006, pp.6,8. Rajan implements this trust infrastructure with a separate trust broker that facilitates attestation of the service provider and enables negotiation of a security policy between the service provider and the client. EX1006, pp.4,7; EX1003, ¶¶96-99. After the attestation broker has received information from the service provider and the client and any negotiation is complete, the client requests an attestation for a service request and the broker verifies that the requirements are met for the service request and binds the client and service provider. EX1006, p.4. In this manner, the broker not only serves to validate the service provider and negotiate any policy differences, but also provides an attested monitor to secure ongoing operations of applications on the service provider. EX1006, pp.8-10; EX1003, ¶¶98-101.

B. Ground 1: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious in view of Munetoh and Rajan

Petitioner evaluates the scope and content of the prior art, any differences between the art and the claim, and the knowledge of person of ordinary skill in the art in accordance with *Graham v. John Deere Co.*, 383 U.S. 1 (1966) and *KSR Int'l Co. v. Teleflex, Inc.*, 550 U.S. 398 (2007). Petitioner is unaware of any secondary considerations relevant to the '441 patent and a more than adequate written rationale is provided to support the combination of Munetoh and Rajan under *KSR*. 550 U.S. at 418.

C. Motivation to Combine Munetoh and Rajan

A PHOSITA would have been motivated to combine the teachings of Munetoh and Rajan because both Munetoh and Rajan disclose systems using the technical standards of the Trusted Computing Group (TCG) to perform trust attestation for computers and software components. EX1005, pp.1; EX1006, pp.2; EX1003, ¶¶101,107-108. Likewise, both Munetoh and Rajan use a trusted platform module (TPM) as the core trust analyzer and root of trust. EX1003, ¶107; EX1006, p.2. Therefore, a PHOSITA would have understood that the core technology of each of these references was compatible with each other and interoperable under the standard. EX1003, ¶110.

Munetoh discloses an attestation client (agent) operating on a trusted platform and discloses an attestation server (remote verifier). EX1003, ¶111;

EX1005, pp.2-3. The broker element of the system of Rajan operates much like the PVA or remote verifier of Munetoh and works using similar information (integrity measurements) in accordance with the TCG standard. EX1005, p.2; EX1006, p.4; EX1003, ¶¶112-113. A PHOSITA would have been motivated to combine the teachings of Rajan into this framework to enable the system of Munetoh to cover more asset configurations including a service-oriented computing architecture. EX1003, ¶112; EX1006, pp.1-2. Likewise, a PHOSITA would have found the functionality of the broker of Rajan, with all of its advantages of applying to a service architecture, to be a natural fit for the remote verifier or PVA of Munetoh. EX1003, ¶¶114-117; EX1005, p.4(disclosing a dedicated verifier as a possible implementation).

Further, a PHOSITA also would have been motivated to incorporate Rajan's detailed measuring of the runtime integrity of applications. EX1003, ¶118. Munetoh already discloses measuring the integrity of running binaries starting with the lowest level BIOS data and all subsequently loaded components (i.e., software). EX1003, ¶118; EX1005, p.8. Because these components are measured when loaded (i.e., running), the measurement is of one or more applications at runtime. *Id.* Rajan discloses a more fine-grained analysis of runtime behavior of application by tracing their behavior with a requirements monitor. EX1006, pp.5-6; EX1003, ¶119. A PHOSITA would have been motivated to use Rajan's

approach for “trace-based” attestation to attest to the run-time behavior of the software. EX1006, pp.5-8; EX1003, ¶119. A PHOSITA would have understood that combining Rajan’s trace-based analysis with the system of Munetoh would enable attestation of the composition of the software at runtime as well as monitoring of precise behavior (such as writing files) and, therefore, better security – application behavior is something both references describe as useful. EX1005, p.9; EX1006, p.7; EX1003, ¶120. Thus, a PHOSITA would have been motivated to combine Munetoh and Rajan for these advantages and would have required little experimentation and had a high chance of success in doing so. EX1003, ¶121.

1. Claim 1

[1a.] A method of providing an attestation service for an application at runtime executing on a computing platform using an attestation server, comprising:

As noted above, Munetoh teaches an implementation of remote attestation of a platform according to the Trusted Computing Group (TCG) specifications. EX1003, ¶125; EX1005, p.4. Many of the principles and concepts in Munetoh were commonly known at the time of Munetoh’s publication due to the publication and implementation of this standard. *Id.* The remote attestation disclosed in Munetoh enables service providers or platforms with a Trusted Platform Module (TPM) or Platform Trust Services (PTS) module to attest security and integrity to

relying parties (i.e., those relying on the service and attestation). EX1003, ¶¶125-126; EX1005, p.4, Fig.-1.

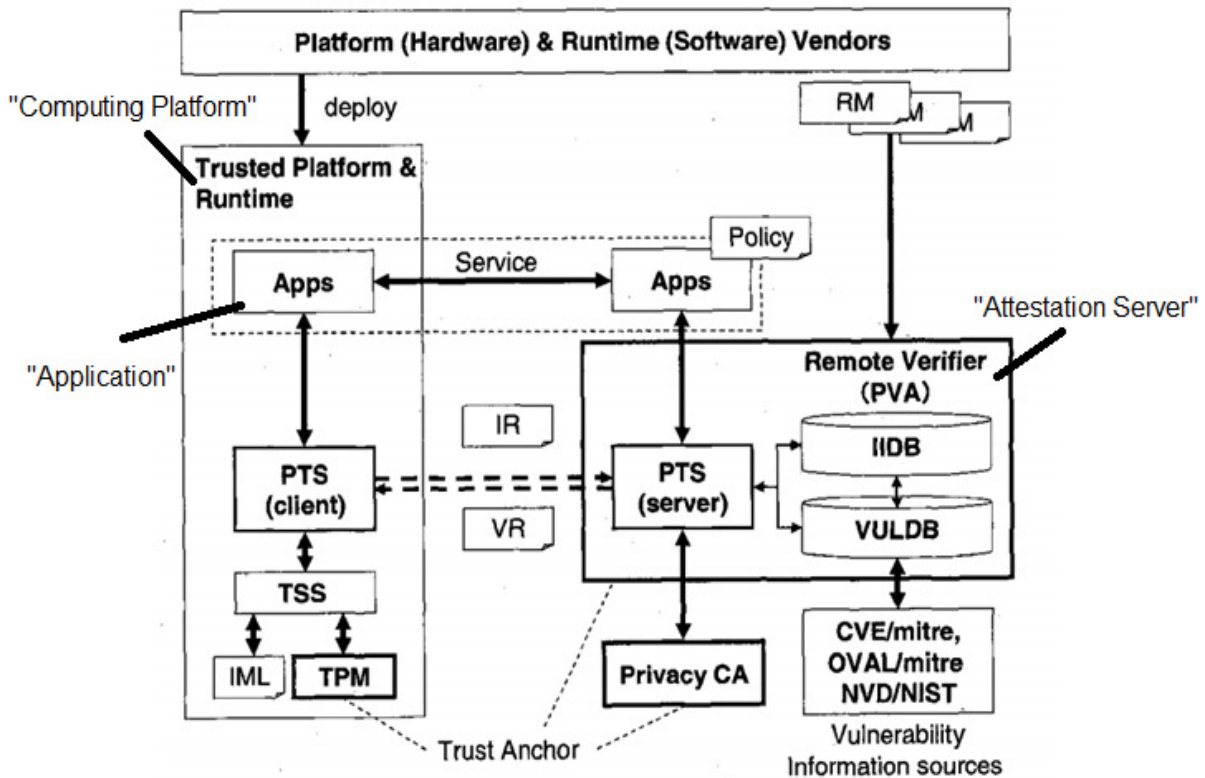


Fig. 1 PTS Client/Server architecture.

EX1005, Fig. 1 (annotated).

Munetoh discloses “a method of providing an attestation service for an application at runtime executing on a computing platform” because it describes a method for the Platform Trust Services (PTS) that “allows applications running on the platform to be measured and attested before it starts using the service, and thus allowing to establish trust between the client and the server.” EX1005, p.5; EX1003 ¶127. Munetoh’s Platform Trust Services (PTS) is an “attestation service” as it measures and attests to applications on a platform, and at least one of the

applications running on the platform is “application at runtime executing on a computing platform.” *Id.*

Munetoh further discloses that the provision of the Platform Trust Services (PTS) (“providing an attestation service”) is made using a server separate from the computing platform. EX1005, pp.4-5, Fig.-1; EX1003, ¶¶126, 131. Munetoh discloses a client-server architecture where the client is the platform (with applications) to be verified/attested and the server is the remote verifier or Platform Validation Authority (PVA). EX1005, pp.4-5, Fig.-1; EX1003, ¶132. Both the client and the server of Munetoh include PTS elements (client/server) which include a number of other modules and models that between them determine whether or not to validate or attest the client platform. EX1003, ¶¶126, 128, 132; EX1005, pp.4-5, Fig.-1. Munetoh discloses that the remote verifier (PVA) server trusts an application on the client (computer platform) based on the attestation data sent by the client. EX1003, ¶131; EX1005, pp.6, 8, Fig.-3. Further, Munetoh discloses that part of the measurement of the integrity of the computer platform to be attested by the TPM includes measuring and attesting applications running on the platform. EX1005, p.5; EX1003, ¶¶131-132. After the server-side PTS has confirmed the integrity of all components on the platform based on the information from the client-PTS and reference manifests for the applications, the server-side PTS or remote verifier may send a validation result to the client-PTS attesting to

the integrity of the platform. EX1003, ¶¶125-133; EX1005, pp.4-6,8. The remote verifier server with server-side PTS is an “attestation server.” *Id.*

For these reasons, a PHOSITA would understand that Munetoh discloses the claim preamble 1[a]. *Id.*

[1b.] receiving, by the attestation server remote from the computing platform:

Munetoh discloses this limitation. EX1003, ¶146.

Munetoh discloses “the attestation server remote from the computing platform” because it describes the platform verification authority (PVA) server with server-side PTS (corresponding to “the attestation server”) as a remote verifier remote from the platform with applications (“computing platform”) to be attested. EX1005, pp.3-5, Fig.-1; EX1003, ¶147.

Munetoh further discloses that the remote verifier or PVA (“attestation server”) “receiving” information from the computing platform. EX1003, ¶¶148-149; EX1005, pp.3-5. The PVA contains a server-side platform trust services (PTS) that includes an integrity management model (IMM) which complies with the TCG standard. *Id.* The server-side PTS is an entity which verifies the integrity reports (IRs) received from the client based on reference manifests (RMs) received at the server from software vendors and returns a Verification Result (VR) or Validation Result (used interchangeably in Munetoh) (corresponding to “attestation”). EX1003, ¶¶146-148; EX1005, pp.5-6, Fig.-1, Fig.-3. The IRs

received from the client-side PTS on a trusted platform (corresponding to “the computer platform”) and the VR sent from the server-side are protected and signed by HTTPS. EX1003, ¶¶148-152; EX1005, pp.4-6. Therefore, the remote verifier (PVA) system and server-side PTS of Munetoh teach this limitation. EX1003, ¶¶145-148.

For these reasons, a PHOSITA would understand that Munetoh discloses the claim limitation 1[b].

[1c.] a runtime execution context indicating attributes of the application at runtime, wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application;

Munetoh discloses “a runtime execution context indicating attributes of the application at runtime” because it describes an integrity report (IR) that “represents a snapshot of the platform integrity measurement” and may include “assertions made by the components running on the client platform” (corresponding to “runtime execution context indicating attributes of the application at runtime”). EX1005, p.2, Fig.-1; EX1003, ¶¶160-161. Specifically, in Munetoh, the hashes of binary components are stored as Platform Configuration Registers (PCRs) that record the integrity measurements for the applications on the platform as a fingerprint in the chain of trust. EX1005, pp.2-3; EX1003, ¶¶160-161, 165. In this manner, a running application can be determined to be trusted based on its binary code being unmodified. *Id.*

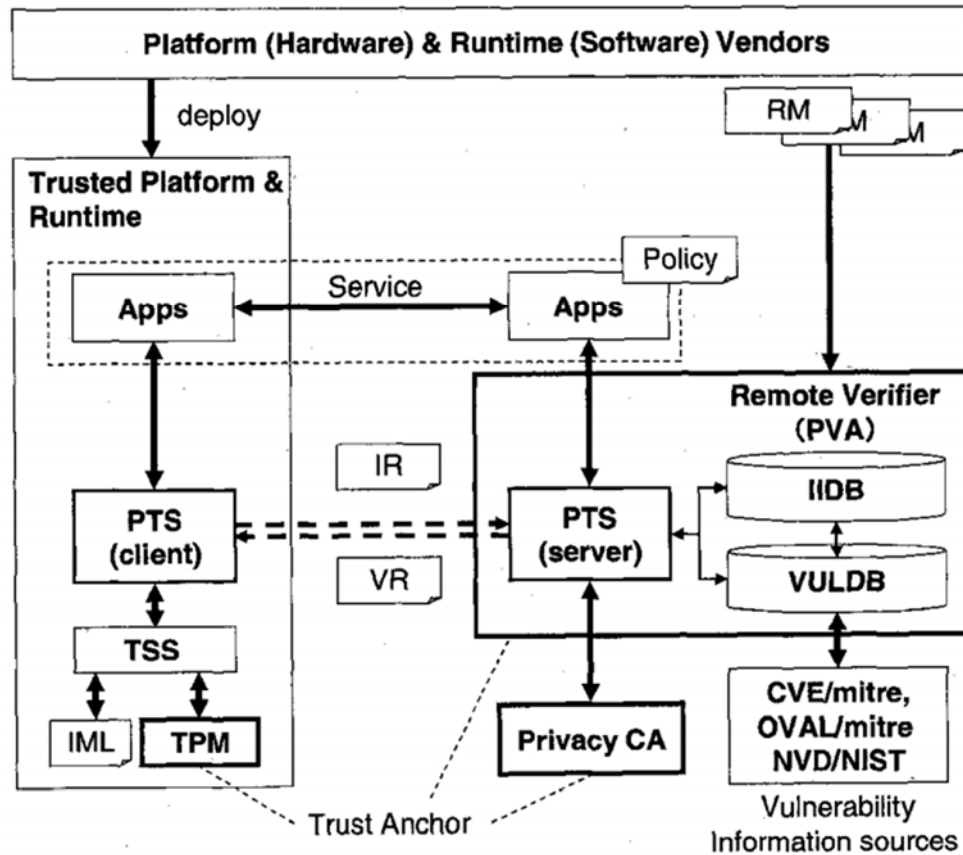


Fig. 1 PTS Client/Server architecture.

EX1005, Fig. 1.

Munetoh further teaches “wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application.” EX1003, ¶¶161-164; EX1005, p.2. Munetoh describes a trusted bootstrap process that “comprises an iterative process of ‘measurement’, loading, and execution of software components” (corresponding to “executable file binaries of the application and loaded components of the application”). *Id.* “[T]he PTS needs to have knowledge about the platform architecture, including the composition of various components running on the platform.” EX1005, p.6. Thus,

the modules and applications are run and measured for integrity, and these measurements are included in the Integrity Report (IR). EX1005, p.3-4; EX1003, ¶¶163-164. The IR is generated by collecting information from the Integrity Measurement Log (IML) and the Platform Configuration Register (PCR) values signed by the TPM, EX1005, p.4. The IR including the software boot measurements and the application measurements is sent to the remote verifier for analysis. *Id.* Munetoh describes that the platform may “send additional information in an attestation, such as the integrity measurement log, which includes a list of components loaded on the platform and measured by the TPM, and how these components are composed.” EX1005, p.3; EX1003, ¶163. Therefore, Munetoh teaches this limitation.

For these reasons, a PHOSITA would understand that Munetoh discloses the claim limitation 1[c]. EX1003, ¶169.

[1d.] a security context providing security information about the application, wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components;

Munetoh teaches “a security context providing security information about the application.” EX1005, pp.1,4-8; EX1003, ¶172. Munetoh describes collecting client-side integrity data into an integrity measurement log (IML) that includes a real time log of state changes in applications of the platform (corresponding to “a security context providing security information about the application”). EX1003,

¶172; EX1005, pp.1,4-5. Using the TPM module on the client, “the integrity data for each component running on the platform is recorded in the security chip and securely checked by remote attestation.” *Id.* In Munetoh, “[t]he term ‘integrity’ is used to denote the state of the components within a computing platform, and the ultimate objective of Trusted Computing is to ascertain that the components in a computing platform are in known and trustworthy states, and have not been tampered with.” EX1005, p.4; EX1003, ¶¶173-175.

As discussed above with respect to limitation 1[c], Munetoh discloses applications or components loaded and executed on the computing platform (“one or more executable file binaries and the loaded components”).

Munetoh teaches “wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components.” EX1005, pp.3-6; EX1003, ¶¶171,174. Munetoh states that the Integrity Measurement Log (IML) “includes a list of components loaded on the platform and measured by the TPM and how they are composed.” EX1005, p.3; EX1003, ¶¶171,174. Munetoh discloses a finite state machine (FSM) that analyzes the IML and IR based on a behavioral model that represents platform states and “the integrity of the component that is loaded and executed”, which corresponds to “the security information comprises an execution analysis of the one or more executable file binaries and the loaded components”. EX1005, pp.5-6; EX1003, ¶¶172-174.

The remote verifier (PVA) server uses this information to “verif[y] the security state from the latest components.” EX1005, p.7.

To the extent that Patent Owner argues that Munetoh does not disclose or teach “an execution analysis” of the applications running on a computer platform, Rajan discloses. EX1006, pp.6,8, Fig.4-5; EX1003, ¶¶176-178. Like Munetoh, Rajan is directed toward providing trustworthy services in a service-oriented architecture based on the standard published by the Trusted Computing Group (TCG). EX1006, p.2.

Petitioner notes that the terms client and server in Rajan are used in relation to the service being attested rather than the attestation service itself. EX1003, ¶¶134-135. Nevertheless, like in Munetoh, two of the systems disclosed in Rajan contain trusted platform modules and one of these systems attests the other just as in Munetoh, so the functional mapping of Rajan to Munetoh is clear even if the client and server terms are different. EX1003, ¶¶134-135. Specifically, the provider of Rajan corresponds to the platform/client-side of Munetoh, the broker of Rajan corresponds to the remote verifier or server-side of Munetoh, and the client of Rajan corresponds to the “relying party” of Munetoh. EX1003, ¶¶134-135.

Rajan explains that its requirements monitor traces the operations of running programs and traces variables, methods, and arguments of the programs (which is similar to the IML information of Munetoh) as they execute to ensure information

security during runtime (e.g., credit card numbers) (corresponding to “an execution analysis”). EX1006, pp.6,8, Fig.4-5; EX1003, ¶¶176-178. Finally, Rajan discloses sending the trace data (i.e., the execution analysis) along with the static attestation of the software stack (i.e., the runtime execution context). EX1003, ¶¶178-179; EX1006, p.7.

A PHOSITA would have recognized that Rajan’s trace-based execution analysis corresponds with “assertions made by the components running on the client platform” as disclosed by Munetoh. EX1003, ¶180. A PHOSITA would have been motivated to combine Rajan’s execution analysis with Munetoh’s IML security context to take advantage of Rajan’s fine-grained, operation-level attestation for services. EX1003, ¶179.

For these reasons, a PHOSITA would understand that Munetoh in combination with Rajan discloses the limitation 1[d]. EX1003, ¶181.

[1e.] generating, by the attestation server, a report indicating security risks associated with the application based on the received runtime execution context and the received security context, as an attestation result; and

As discussed above with respect to limitation 1[a], Munetoh discloses a remote verifier (PVA) server (corresponding to “attestation server”).

Munetoh teaches “generating, by the attestation server, a report indicating security risks associated with the application...as an attestation result” because it describes verifying at the remote verifier (PVA) (corresponding to “attestation

server”) the composition of the received PCR data in the integrity reports (IRs) from the platforms based on the reference manifests from the third-party vendors to ensure the provenance of the application (corresponding to “security risks associated with the application based on the received runtime execution context”). EX1005, p.2-3,5, Fig.-4; EX1003, ¶185.

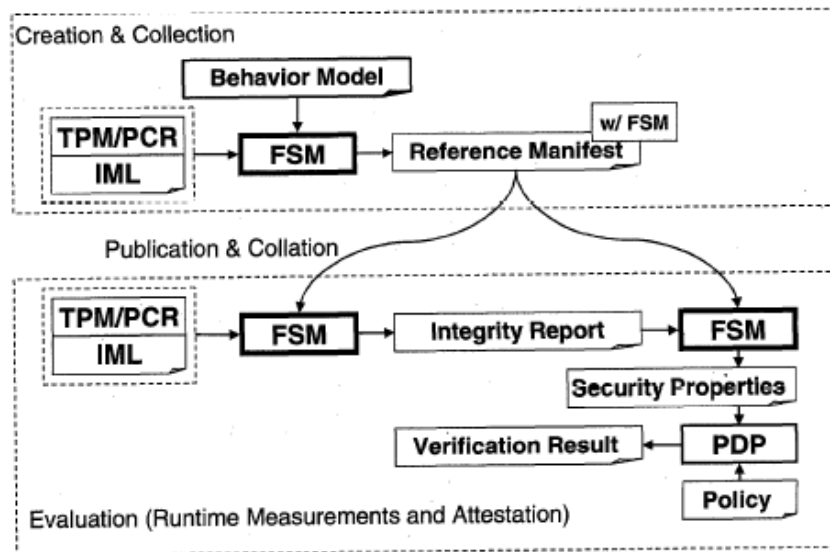


Fig. 4 IMM steps and data flow with FSM.

EX1005, Fig. 4.

Munetoh discloses that the remote verifier (PVA) considers the IML data that includes state changes in the running application (as taught by Munetoh-Rajan above) on the server and compares these logs with vulnerability information to assess threats (corresponding to “security risks associated with the application based on the received security context”). EX1005, pp.6-7; EX1003, ¶¶186-189. Based on the analysis of the IML data and the PCR data in the IR, the server-side PTS determines whether to trust and validate the computer platform that sent the

IR (corresponding to “generating, by the attestation server, a report indicating security risks associated with the application”). EX1005, 6-7; EX1003, ¶190. The remote verifier (PVA) issues a validation result (VR) of Valid, Invalid, or Unverified, and includes in the result details to explain the decision (corresponding to “generating ...a report indicating security risks associated with the application...as an attestation result”). EX1005, pp.5-7; EX1003, 185-192.

Alternatively, the report that is generated from the evaluating or testing of the IRs to determine vulnerabilities using information (e.g., CVE identifiers) in a vulnerabilities database (VULDB) corresponds to “a report indicating security risks associated with the application...as an attestation result”). EX1003, ¶193; EX1005, pp.2,5-7, Fig.-1. Munetoh describes using at least an Open Vulnerability and Assessment Language (OVAL) tool, implemented at the PTS server-side, “which is a generic open framework for representing the vulnerability test processes. OVAL can describe the test cases to detect the presence of vulnerabilities. EX1005, pp.6-7. “Vulnerability information...includes the CVE information stored in the database, and each CVE entry is associated with a security score for the CVSS” (Common Vulnerability Scoring System). EX1005, p.7; EX1003, ¶193. Further, Munetoh discloses that the test results output by the remote verifier (PVA) are “based on the received runtime execution context and the received security context” because Munetoh describes that they are based on at

least the Integrity Measurement Log (IML) and Integrity Report (IR)

(corresponding to “the received runtime execution context and the received security context”). EX1005, pp.1,4-7, FIGS.1,3-4; EX1003, ¶¶186-190.

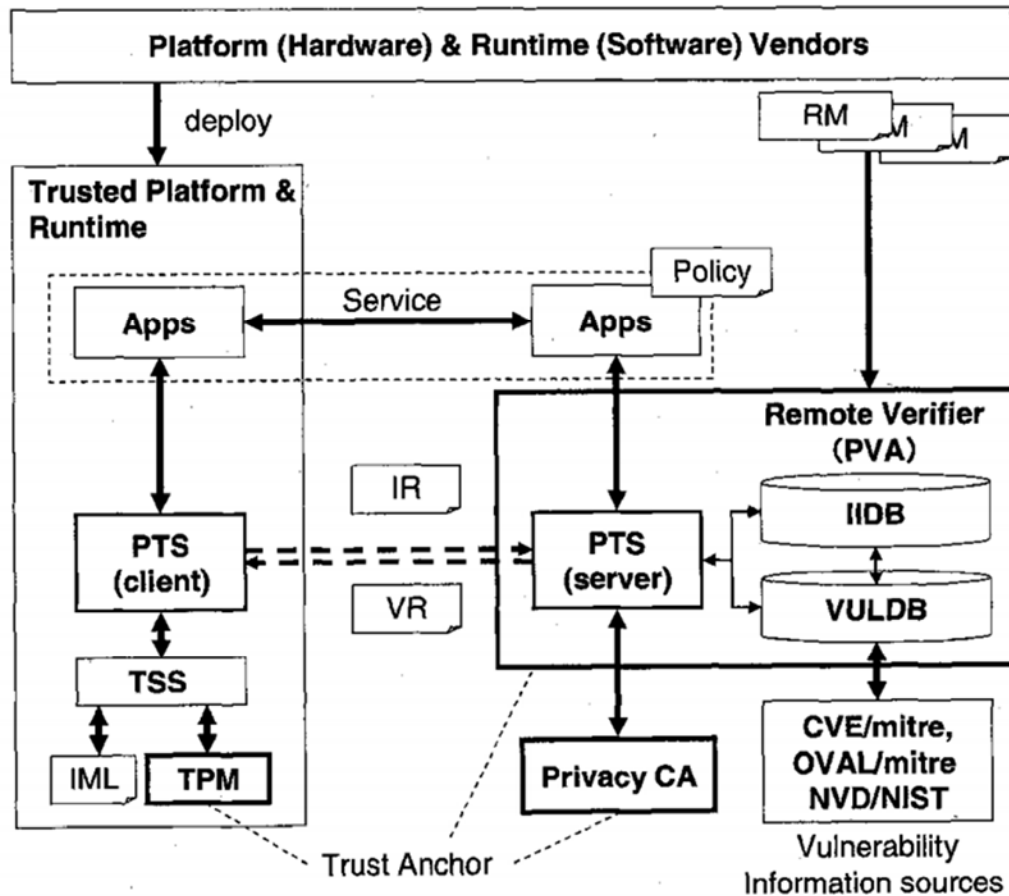


Fig. 1 PTS Client/Server architecture.

EX1005, Fig. 1.

Rajan discloses consistent and compatible features corresponding to those in preamble 1[e] thereby supporting a motivation to combine.

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses the limitations recited in claim limitation 1[e]. EX1003, ¶197.

[1f.] sending, by the attestation server, the attestation result associated with the application.

Munetoh teaches “sending, by the attestation server, the attestation result associated with the application” because it describes that the validation result (VR) issued by the PVA or remote verifier (corresponding to “the attestation server”) is sent to the client or platform to be attested via HTTPS, where the validation result includes the detailed conditions at the computer platform that led to the result (corresponding to “sending...the attestation result”). EX1003, ¶¶200-203; EX1005, pp.6-7.

Alternatively, Munetoh teaches or suggests “sending, by the attestation server, the attestation result associated with the application” because it describes that the PVA or remote verifier (corresponding to “the attestation server”) reports the test results for vulnerabilities, which a PHOSITA would understand to mean that the results are sent out. EX1003, ¶204; EX1005, pp.6-7.

For these reasons, a PHOSITA would understand that Munetoh discloses claim limitation 1[f]. EX1003, ¶207.

Therefore, a PHOSITA would have found the limitations recited in claim 1 to be obvious based on Munetoh in view of Rajan. EX1003, ¶122. Accordingly,

claim 1 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

2. Claim 2

The method of claim 1, further comprising:

[2a.] generating, by the attestation server, an application artifact as a reference for changes in a subsequent execution context; and

Rajan teaches “generating...an application artifact as a reference for changes in a subsequent execution context” because it describes generating a reference hash of the software stack of the service provider. EX1006, pp.6-7; EX1003, ¶212. Rajan discloses that this reference hash is used for comparison throughout the execution of the service to ensure that the software stack of the service provider has not changed. EX1006, pp.6-7; EX1003, ¶¶212-213. Rajan teaches that the reference hash (corresponding to “application artifact” is generated at the broker server (corresponding to “attestation server”). EX1003, ¶212; EX1006, p.7.

Rajan discloses:

During service execution, the broker and the provider must communicate to verify that: the provider and the broker are indeed the same principals [sic] that agreed to the contract during the publish step, the requirements monitor as well as the software stack on the provider’s end is not compromised, and the trace provided by this monitor does not violate the properties desired by the client.

The trust analyzer component, which derives its root of trust from the TPM on the provider’s side, serves to verify the integrity of the

requirements monitor. It provides a report that includes an immutable integrity measurement of the local architecture along with a time stamp to indicate the freshness of the measurement.

EX1006, p.6.

This disclosure in Rajan makes clear that the reference hash is used for comparison throughout the execution of the service to ensure that the software stack of the service provider has not changed and is created to enable measurement of the integrity of the application's environment in the future (corresponding to "an application artifact as a reference for changes in a subsequent execution context"). *Id.*; EX1003, ¶¶214-215.

Thus, Rajan teaches an application artifact as a reference for changes in a subsequent execution context as claimed. EX1003, ¶¶214-215. Further, based on this use case, a PHOSITA would have understood "reference measurements" to be used for comparison against subsequent measurements sent during service execution. EX1003, ¶215(citing EX1011).

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses claim limitation 2[a]. EX1003, ¶217.

[2b.] sending the generated application artifact such that subsequent changes to the runtime execution context are tracked based on the generated application artifact.

Rajan discloses generating a reference hash (i.e., computeSHA1Hash) for the software stack up to the requirements monitor in order to identify unauthorized

changes in the stack in the future. EX1003, ¶¶214-215; EX1006, p.7. Based on this reference hash, subsequent changes to the runtime execution context of the software stack are tracked. *Id.* Further, as discussed with respect to limitation 1[e], Munetoh discloses sending Validation Results (VR) to the platform seeking attestation. EX1005, p.5, Fig.-1; EX1003, ¶221. A PHOSITA would have been motivated to combine the current static hash disclosed by Rajan with the sending of the Validation Result (VR) in Munetoh to reduce errors and enable policy negotiation. EX1003, ¶222; EX1005, p.6; EX1006, p.4.

Specifically, Munetoh discloses that re-initiated authentication is expected between client and server as indicated by having the client do its own initial checks to reduce networking load caused by obvious errors that would cause attestation to fail. EX1003, ¶¶221-222; EX1005, p.6. Likewise, Rajan anticipates a back-and-forth negotiation over security/requirements policy before a valid attestation is issued. EX1006, p.4. Thus, it would have been obvious to a PHOSITA to include Rajan's computed hash of the attestation checksum in the VR sent back to the attestation client so that the client could know what its current state is recorded as at the attestation server and potentially fix errors if there are any. EX1003, ¶222.

To the extent that the patent owner asserts that logging data corresponds to the "sending" recited in this claim, Rajan also teaches storing the application artifact on the attestation server. EX1003, ¶223(citing 1010, p.20).

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses the limitations recited in claim limitation 2[b]. EX1003, ¶¶224-225.

Therefore, a PHOSITA would have found the limitations recited in claim 2 to be obvious based on Munetoh in view of Rajan. EX1003, ¶226. As shown, the Munetoh-Rajan combination discloses all limitations of claim 1 from which claim 2 depends. Accordingly, claim 2 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

3. Claim 3

[3a.] The method according to claim 2, further comprising digitally signing the attestation results:

Munetoh discloses transmitting the validation results (VR) within an HTTPS message body to the client. EX1003, ¶229; EX1005, p.5. HTTPS involves signing of packets with a Message Authentication Code (MAC) and transmitting the packets over an encrypted channel (i.e., secure socket layer – SSL). EX1003, ¶230(citing EX1009).

Alternatively, Munetoh teaches reporting the results of the vulnerability tests of the IR and IML which a PHOSITA would understand would be sent to the computing platform over the HTTPS channel disclosed as connecting them. EX1003, ¶229; EX1005, p.7.

For these reasons, a PHOSITA would understand that Munetoh discloses the limitations recited in claim limitation 3[a]. EX1003, ¶¶229-230.

[3b.] wherein the attributes further comprise parent-child process associations of the application.

As described regarding limitations 1[c] and 1[d], Munetoh discloses that the remote verifier receives an integrity measurement log (IML) including PCR values that measure the components that are loaded on the platform and how these components are composed (corresponding to “parent-child”). EX1003, ¶¶231-232; EX1005, p.3. The IML is used by the remote verifier as an input to the FSM for behavior analysis of software running on the platform – software launching software (corresponding to “parent-child process associations of the application”). EX1003, ¶233; EX1005, p.6, Fig.-4.

The FSM evaluates the software on the platform based on the running states of this software in the IML. EX1003, ¶233; EX1005, pp.5-6. Munetoh states: “[e]ach state in the behavior model represents a platform state, and each transition represents the integrity of the component that is loaded and executed.” *Id.* A PHOSITA would have understood that analyzing how “components are composed” when components are running executables would include analyzing which process launched another process (i.e., parent-child process relationships). EX1003, ¶¶233-235(citing EX1012). To the extent that this disclosure is not explicit, a PHOSITA would have known that process relationships are relevant for trusted computing

and that the relationships between processes provide significant information with respect to determining the “health” of a platform. *Id.*

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses claim limitation 3[b]. EX1003, ¶236.

Therefore, a PHOSITA would have found the limitations recited in claim 3 to be obvious based on Munetoh in view of Rajan. EX1003, ¶236. As shown, the Munetoh-Rajan combination discloses all limitations of claims 1 and 2 from which claim 3 depends. Accordingly, claim 3 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

4. Claim 4

The method of claim 1, wherein the received security context is an introspective security context, and wherein the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components.

Munetoh discloses that in addition to the PCR values that track runtime execution states and are recorded in the IML, the IML includes additional data including “a list of components loaded on the platform and measured by the TPM, and how these components are composed.” EX1003, ¶239; EX1005, p.5. A PHOSITA would have understood the “composition” of the components is introspective of the operation of the system because it relates to the internals of the system’s construction. EX1003, ¶¶239-240.

Munetoh describes using the IML to assess the integrity of a running application and its states or state changes. EX1003, ¶¶242-243; EX1005, p.6, Fig.-5. Indeed, the IML also provides an introspective perspective to the software by tracking/logging the changing states. *Id.* Similarly, the '441 patent describes introspection as behavioral or predictive analytics based on historic state information. EX1001, 9:37-46. Munetoh states that “[t]he Integrity Measurement Architecture supports such measurements for the Linux kernel, and enhances the role of the TPM for measuring not only the static state of a system but also the dynamic state.” EX1005, p.5. Therefore, a PHOSITA would understand this disclosure of Munetoh to teach “the received security context is an introspective security context, and wherein the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components.” EX1003, ¶240-243.

Further, Rajan teaches that its trace analysis of running binaries includes internal operations of the programs / binaries (e.g., file writes) (“dynamic analysis”). EX1006, pp.5-6; EX1003, ¶¶244-245. A PHOSITA would have been motivated to combine this dynamic analysis with the introspective analysis described in Munetoh for at least the reasons and advantages provide by Rajan as noted in the motivation to combine above. *Id.*

Therefore, a PHOSITA would have found the limitations recited in claim 4 to be obvious based on Munetoh in view of Rajan. EX1003, ¶246. As shown, the Munetoh-Rajan combination discloses all limitations of claim 1 from which claim 4 depends. Accordingly, claim 4 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

5. Claim 5

The method of claim 4, wherein the generating of the report indicating security risks associated with the application includes generating, by the attestation server, one or more security assertions that pertain to the received runtime execution context and the received introspective security context.

Under either of the alternative interpretations of Munetoh for the “report” in claim limitation 1[e], Munetoh discloses using the data provided in the PCR values and the IML to identify vulnerabilities (corresponding to “security risks”) in software indicated as loaded on the computing platform in the IML. EX1003, ¶¶249-250; EX1005, pp.4-7. Munetoh discloses generating and reporting the results of test of the platform for vulnerabilities by the remote verifier (“attestation server”). *Id.* “The report, which describes the presence of vulnerabilities and the degrees of security risks, is generated by verifying the integrity of the files and evaluating them against the vulnerability information.” *Id.* The detection of vulnerabilities are security assertions that a specific file is vulnerable to a specific risk and as modified by Rajan’s dynamic risk analysis (“one or more security

assertions that pertain to the received runtime execution context and the received introspective security context”). EX1003, ¶¶250-251.

Therefore, a PHOSITA would have found the limitations recited in claim 5 to be obvious based on Munetoh in view of Rajan. EX1003, ¶¶252. As shown, the Munetoh-Rajan combination discloses all limitations of claims 1 and 4 from which claim 5 depends. Accordingly, claim 5 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

6. Claim 6

The method according to claim 1, further comprising authenticating the application using a plurality of collaboration services.

Munetoh discloses that various software vendors (e.g., for the operating system as well as the applications of the platform) produce reference manifests (RMs) for the software they produce. EX1005, p.4; EX1003, ¶¶255. Munetoh states: “The vendors release software with RMs and the software is deployed into the target platforms.” *Id.* Further, the vulnerability data used by the remote verifier of Munetoh is sourced from open sources, vendors, and security advisories and used to analyze the validity of packages on the platform. EX1005, p.8; EX1003, ¶¶255-256. For example, Munetoh discloses various vendors or entities that provide or maintain CVE (Common Vulnerabilities and Exposures) Identifiers that can be used to identify information security vulnerabilities. *Id.*, p.7. The services provided by these software vendors and entities providing CVE information correspond to

“a plurality of collaboration services.” EX1005, p.4; EX1003, ¶257. A reference manifest is an assertion made by a manufacturer on the composition of components and their integrity metrics (i.e., hash values) is called the Reference Manifest (RM).” The expected behavior or static hashes of the RM is used to verify the integrity of the application (corresponding to “authenticating the application using a plurality of collaboration services”). EX1003, ¶¶255-257.

Therefore, a PHOSITA would have found the limitations recited in claim 6 to be obvious based on Munetoh in view of Rajan. EX1003, ¶258. As shown, the Munetoh-Rajan combination discloses all limitations of claim 1 from which claim 6 depends. Accordingly, claim 6 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

7. Claim 7

The method according to claim 1, further comprising controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results.

Rajan teaches determining an attestation result specifically for a service or application based on operational traces transmitted from the service to the broker (corresponding to “attestation server”). EX1003, ¶194; EX1006, p.7. Rajan teaches the attestation result of the broker is calculated during runtime based on both the static hash of the data and the trace data of the application. EX1006, p.8; EX1003, ¶261. Rajan further discloses that the client limits the user’s transaction (i.e., with

the service of the service provider) by aborting the transaction if a violation or timeout is reported, which corresponds to “controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results”. *Id.*

Therefore, a PHOSITA would have found the limitations recited in claim 7 to be obvious based on Munetoh in view of Rajan. EX1003, ¶262. As shown, the Munetoh-Rajan combination discloses all limitations of claim 1 from which claim 7 depends. Accordingly, claim 7 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

8. Claim 9

The method according to claim 1, further comprising providing confidence metrics in the attestation results indicating a level of security risk by different classifications such that a restriction on a user's transaction with the application are applied based on the level of security risk indicated by the confidence metrics in the attestation results.

Munetoh discloses that the CVEs received from third party sources describing vulnerabilities in software include scores relating to the severity of risk that each CVE poses to the system (corresponding to “providing confidence metrics in the attestation results indicating a level of security risk by different classifications”). EX1003, ¶¶265-267; EX1005, p.8. Munetoh further discloses that the vulnerability analysis report includes application-level vulnerabilities, and that the CVSS score is used as a basis for denying access to vulnerable components of

the platform (corresponding to “a restriction on a user’s transaction with the application are applied based on the level of security risk indicated by the confidence metrics in the attestation results”). EX1003, ¶¶265-267; EX1005, Table-1, pp.7-8. Munetoh discloses that the verification/validation result format includes a verification status as well as explanations for the status, which would include the report of the security risks derived from the vulnerability assessment described above. EX1003, ¶¶187-192.

Therefore, a PHOSITA would have found the limitations recited in claim 9 to be obvious based on Munetoh in view of Rajan. EX1003, ¶268. As shown, the Munetoh-Rajan combination discloses all limitations of claim 1 from which claim 9 depends. Accordingly, claim 9 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Munetoh in view of Rajan.

D. Ground 2: Claims 1-7 and 9 are unpatentable under 35 U.S.C. § 103(a) as obvious over Rajan in view of Munetoh

Petitioner evaluates the scope and content of the prior art, any differences between the art and the claim, and the knowledge of person of ordinary skill in the art in accordance with *Graham v. John Deere Co.*, 383 U.S. 1 (1966) and *KSR Int’l Co. v. Teleflex, Inc.*, 550 U.S. 398 (2007). Petitioner is unaware of any secondary considerations relevant to the ’441 patent and a more than adequate written rationale is provided to support the combination of Rajan and Munetoh under *KSR*

550 U.S. at 418.

E. Motivation to Combine Rajan and Munetoh

A PHOSITA would have been motivated to combine the teachings of Rajan and Munetoh because both Rajan and Munetoh disclose systems using the technical standards of the Trusted Computing Group (TCG) to perform trust attestation for computers and software components. EX1005, pp.1; EX1006, pp.2; EX1003, ¶¶101,107-108. Likewise, both Rajan and Munetoh use a trusted platform module (TPM) as the core trust analyzer and root of trust. EX1003, ¶107; EX1006, p.2. Therefore, a PHOSITA would have understood that the core technology of each of these references was compatible with each other and interoperable under the standard. EX1003, ¶110,270-271.

The core attestation technology, such as the use of TPMs and PCRs, is the same. A PHOSITA would have been motivated to combine Munetoh's static vulnerability analysis with Ranjan's dynamic trace-based analysis because Munetoh's static analysis enables policy decisions based on known security-risks associated with an application. EX1003, ¶270.

1. Claim 1

[1a.] A method of providing an attestation service for an application at runtime executing on a computing platform using an attestation server, comprising:

Rajan teaches the implementation of “an attestation service for an application at runtime executing on a computing platform using an attestation server” in order to attest the integrity of service-oriented computing architectures. EX1003, ¶¶140,274; EX1006, p.1. Rajan discloses that the provider (“computing platform”) provides many services (including a software stack or an “application”) to clients and needs to be attested in order for those clients to safely use the provider. EX1003, ¶¶139-140; EX1006, p.3. Rajan’s implementation includes a broker (corresponding to “using an attestation server”) that receives a client’s service requirements and checks them against the measurements from the TPM on the provider. EX1006, pp.4-6; EX1003, ¶¶140-141. “The TPM on the service provider’s site is used to attest to the integrity of the provider’s software stack as well its identity.” EX1006, p.3. The provider’s TPM performs integrity measurements of applications and hardware which are stored as platform configuration registers (PCRs) (corresponding to “providing an attestation service for an application executing on a computing platform”). *Id.* Further, Rajan discloses detailed monitoring performed by the requirements monitor that includes traces of program operations during execution. *Id.* Therefore, Rajan teaches “an attestation service for an application at runtime”. Accordingly, the combination of

Rajan and Munetoh teaches “providing an attestation service for an application at runtime executing on a computing platform using an attestation server” as recited in this claim element. EX1003, ¶142,275.

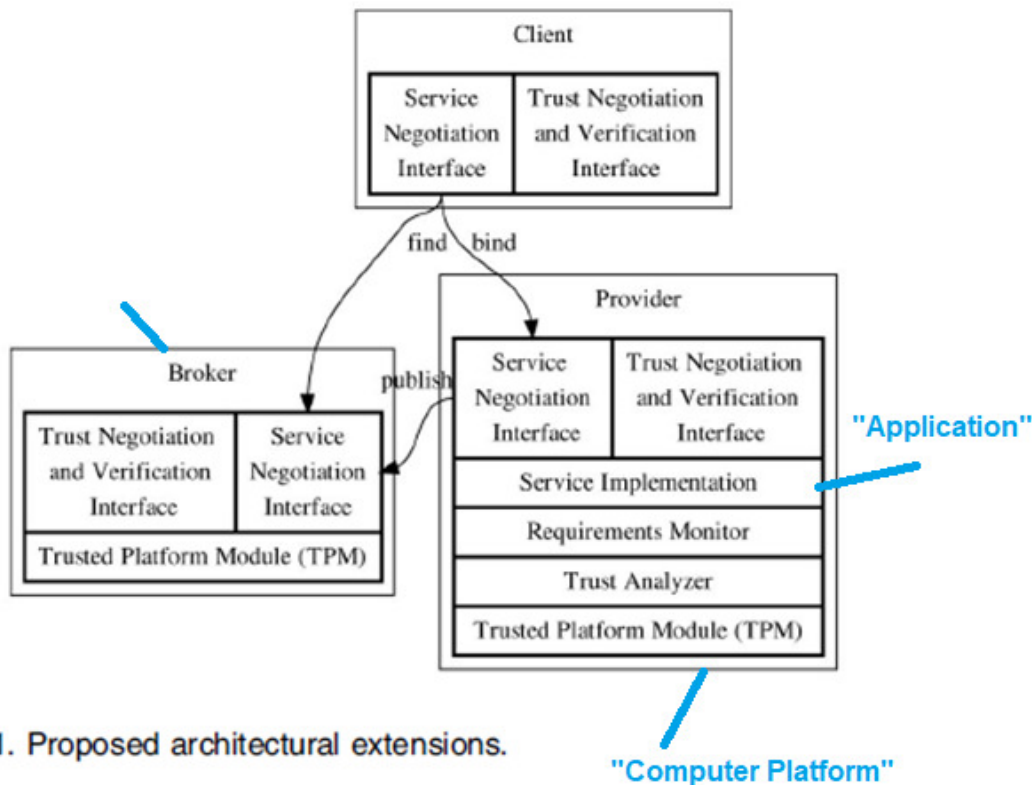


Fig. 1. Proposed architectural extensions.

EX1006, Fig.1 (annotated)

Petitioner notes that the terms client and server in Rajan are used in relation to the service being attested rather than the attestation service itself. EX1003, ¶¶134-135. Nevertheless, like in Munetoh, two of the systems disclosed in Rajan contain trusted platform modules and one of these systems attests the other just as in Munetoh, so the functional mapping of Rajan to Munetoh is clear even if the client and server terms are different. EX1003, ¶¶134-135. Specifically, the service

provider of Rajan corresponds to the platform/client-side of Munetoh, the broker of Rajan corresponds to the remote verifier or server-side of Munetoh, and the client of Rajan corresponds to the “relying party” of Munetoh. EX1003, ¶¶134-138.

For these reasons, a PHOSITA would understand that Rajan discloses claim limitation 1[a]. EX1003, ¶275.

[1b.] receiving, by the attestation server remote from the computing platform:

Rajan teaches a service provider (corresponding to “the computer platform”) that connects to a broker (corresponding to “the attestation server”) for attestation in a publish step that includes data describing the services at the service provider.

EX1003, ¶¶153-154; EX1006, p.3, Fig.-2. The broker receives integrity measurements from the service provider describing the security state of a monitor and analyzer on the service provider via a secure channel. EX1003, ¶¶153-155;

EX1006, p.5, Fig.-2. Therefore, the broker as disclosed in Rajan teaches this

limitation. EX1003, ¶278.

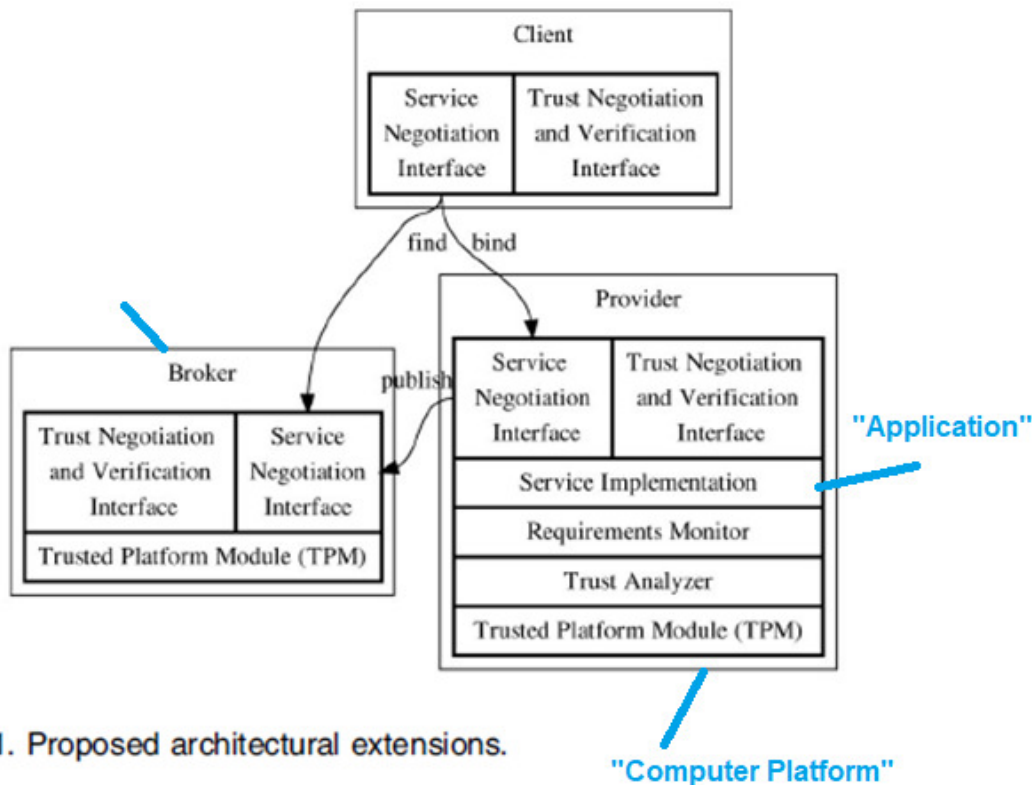


Fig. 1. Proposed architectural extensions.

EX1006, Fig.1 (annotated)

For these reasons, a PHOSITA would understand that Rajan discloses claim limitation 1[b].

[1c.] a runtime execution context indicating attributes of the application at runtime, wherein the attributes comprise one or more executable file binaries of the application and loaded components of the application;

Rajan discloses tracing software execution in the software stack

(corresponding to “a runtime execution context indicating attributes of the application”) including while the software is operating (via a requirements

monitor). EX1003, ¶¶166-167; EX1006, pp.6-7. Rajan discloses that the traces are a “static checksum of the software stack” (corresponding to “the attributes comprise one or more executable file binaries of the application”) and logs of running services (corresponding to “the attributes comprise one or more ...loaded components of the application”). EX1006, pp.6,8, Fig.4-5; EX1003, ¶¶166-167, 281-282.

For these reasons, a PHOSITA would understand that Rajan discloses claim limitation 1[c].

[1d.] a security context providing security information about the application, wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components;

Rajan is directed toward providing trustworthy services in a service-oriented architecture based on the standard published by the Trusted Computing Group (TCG). EX1006, p.2. Rajan explains that its requirements monitor traces the operations of running programs and traces variables, methods, and arguments of the programs (which is similar to the IML information of Munetoh) as they execute to ensure information security during runtime (e.g., credit card numbers) (corresponding to “an execution analysis”). EX1006, pp.6,8, Fig.4-5; EX1003, ¶¶176-178. Finally, Rajan discloses sending the trace data (i.e., the execution

analysis) along with the static attestation of the software stack (i.e., the runtime execution context). EX1003, ¶¶178-179,286; EX1006, p.7.

Further, Munetoh teaches “a security context providing security information about the application.” EX1005, pp.1,4-8; EX1003, ¶172. Munetoh describes collecting client-side integrity data into an integrity measurement log (IML) that includes a real time log of state changes in applications of the platform (corresponding to “a security context providing security information about the application”). EX1003, ¶172; EX1005, pp.1,4-5. Using the TPM module on the client, “the integrity data for each component running on the platform is recorded in the security chip and securely checked by remote attestation.” *Id.* In Munetoh, “[t]he term ‘integrity’ is used to denote the state of the components within a computing platform, and the ultimate objective of Trusted Computing is to ascertain that the components in a computing platform are in known and trustworthy states, and have not been tampered with.” EX1005, p.4; EX1003, ¶¶173-175.

Finally, Munetoh teaches “wherein the security information comprises an execution analysis of the one or more executable file binaries and the loaded components.” EX1005, pp.3-6; EX1003, ¶¶171,174. Munetoh states that the Integrity Measurement Log (IML) “includes a list of components loaded on the platform and measured by the TPM and how they are composed.” EX1005, p.3;

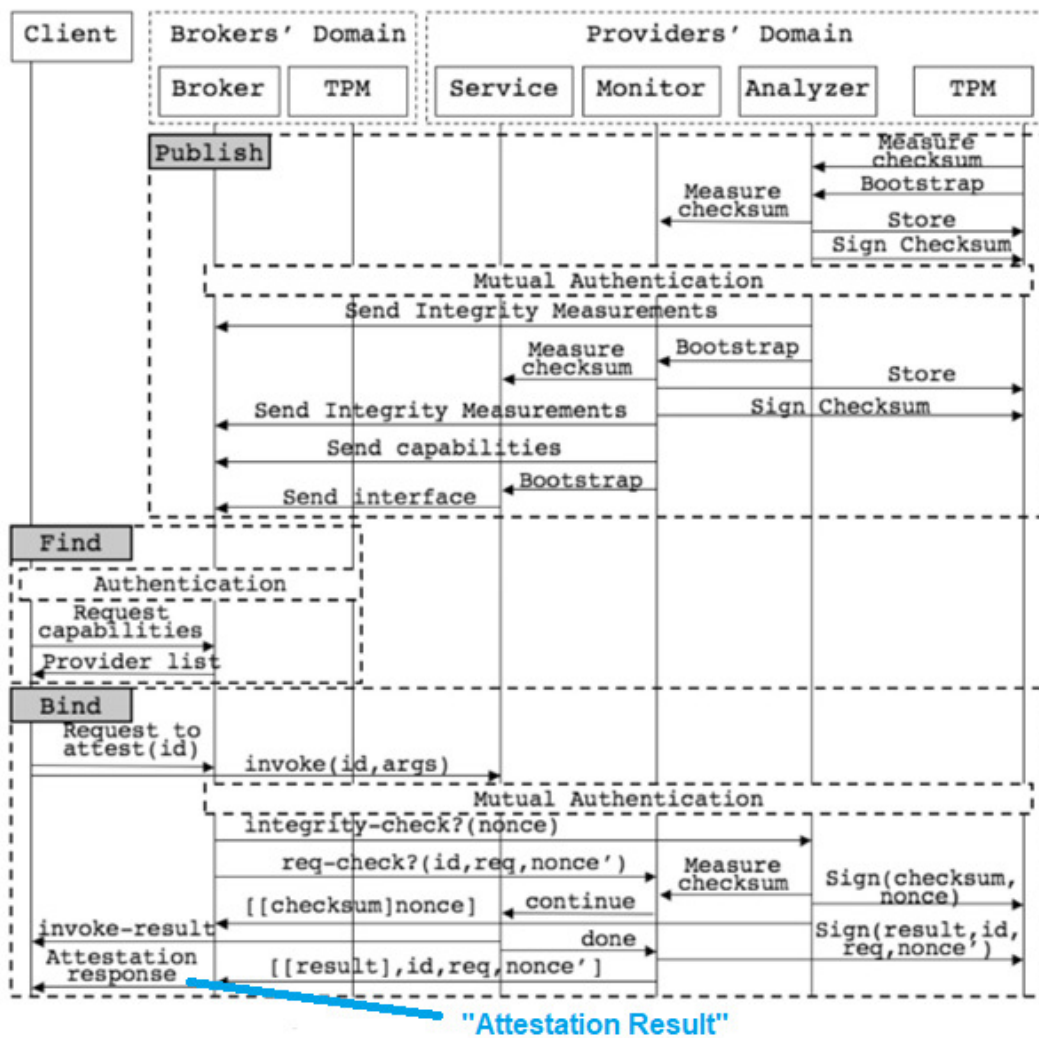
EX1003, ¶¶171,174. Munetoh discloses a finite state machine (FSM) that analyzes the IML and IR based on a behavioral model that represents platform states and “the integrity of the component that is loaded and executed”, which corresponds to “the security information comprises an execution analysis of the one or more executable file binaries and the loaded components”. EX1005, pp.5-6; EX1003, ¶¶172-174. The remote verifier (PVA) server uses this information to “verif[y] the security state from the latest components.” EX1005, p.7. A PHOSITA would have been motivated to combine this security contextual information into Rajan so that Rajan’s broker, which verifies the attestation data, would be able to know which components are loaded and how they are composed. EX1003, ¶287.

For these reasons, a PHOSITA would understand that the combination of Rajan and Munetoh discloses claim limitation 1[d]. EX1003, ¶288.

[1e.] generating, by the attestation server, a report indicating security risks associated with the application based on the received runtime execution context and the received security context, as an attestation result; and

As discussed above, Rajan teaches a broker (“attestation server”) for assessing a service provider’s compliance with security requirements (corresponding to “security risk”). Further, Rajan teaches the runtime execution context as described regarding ground 2, claim 1[c] and Rajan in view of Munetoh teaches the security context as described regarding ground 2, claim 1[d].

Rajan teaches determining an attestation response (“attestation result”) specifically for a service or application based on operational traces transmitted from the service to the broker (corresponding to “attestation server”). EX1003, ¶¶194; EX1006, p.7, Fig.2.



EX1006, Fig. 2 (annotated)

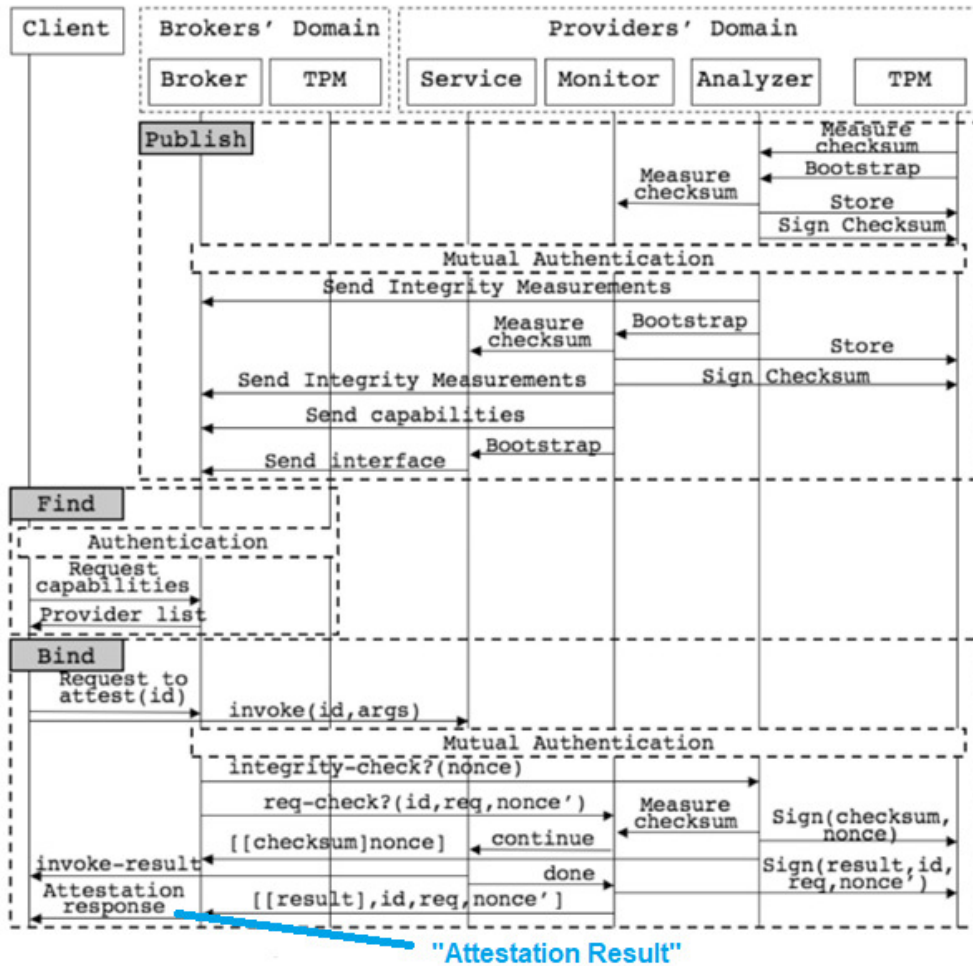
Rajan teaches the attestation result of the broker is calculated during runtime based on both the static hash of the data and the trace data of the application

(corresponding to “generating...a report indicating security risks associated with the application based on the received runtime execution context and the received security context” as combined regarding limitations 1[c] and 1[d]). EX1003, ¶¶194; EX1006, pp.5-7. A PHOSITA would have been motivated to combine Munetoh’s validation result report having vulnerability data with Rajan’s determination of violations of the runtime execution of the application to obtain the fine-grained, operation-level vulnerability information and obtain static and dynamic analysis of the system. EX1003, ¶¶194-195,291.

For these reasons, a PHOSITA would understand that the combination of Rajan and Munetoh discloses claim limitation 1[e]. EX1003, ¶293.

[1f.] sending, by the attestation server, the attestation result associated with the application.

Rajan teaches that the attestation result is measured and analyzed at an application level (e.g., web service) as explained regarding claim limitation 1[e]. EX1006, p.8. Rajan also discloses that the broker notifies the service provider being attested of violations such as an API call (corresponding to “sending...the attestation result associated with the application”). EX1003, ¶205; EX1006, p.7. The broker also notifies (sends) the attestation result to the client to which the service is to be provided. *Id.*



EX1006, Fig.2 (annotated)

For these reasons, a PHOSITA would understand that the combination of Rajan and Munetoh discloses claim limitation 1[f]. EX1003, ¶297.

Therefore, a PHOSITA would have found the limitations recited in claim 1 to be obvious based on Rajan in view of Munetoh. EX1003, ¶298. Accordingly, claim 1 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

2. Claim 2

The method of claim 1, further comprising:

[2a.] generating, by the attestation server, an application artifact as a reference for changes in a subsequent execution context; and

Rajan teaches “generating...an application artifact as a reference for changes in a subsequent execution context” because it describes generating a reference hash of the software stack of the service provider. EX1006, pp.6-7; EX1003, ¶212,301. Rajan discloses that this reference hash is used for comparison throughout the execution of the service to ensure that the software stack of the service provider has not changed. EX1006, pp.6-7; EX1003, ¶¶212-213. Rajan teaches that the reference hash (corresponding to “application artifact” is generated at the broker server (corresponding to “attestation server”). EX1003, ¶212; EX1006, p.7.

Rajan discloses:

During service execution, the broker and the provider must communicate to verify that: the provider and the broker are indeed the same principals [sic] that agreed to the contract during the publish step, the requirements monitor as well as the software stack on the provider’s end is not compromised, and the trace provided by this monitor does not violate the properties desired by the client.

The trust analyzer component, which derives its root of trust from the TPM on the provider’s side, serves to verify the integrity of the requirements monitor. It provides a report that includes an immutable

integrity measurement of the local architecture along with a time stamp to indicate the freshness of the measurement.

EX1006, p.6.

This disclosure in Rajan makes clear that the reference hash is used for comparison throughout the execution of the service to ensure that the software stack of the service provider has not changed and is created to enable measurement of the integrity of the application's environment in the future (corresponding to "an application artifact as a reference for changes in a subsequent execution context"). *Id.*; EX1003, ¶¶214-215.

Thus, Rajan teaches an application artifact as a reference for changes in a subsequent execution context as claimed. EX1003, ¶¶214-215. Further, based on this use case, a PHOSITA would have understood "reference measurements" to be used for comparison against subsequent measurements sent during service execution. EX1003, ¶215(citing EX1011).

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses claim limitation 2[a]. EX1003, ¶217,302.

[2b.] sending the generated application artifact such that subsequent changes to the runtime execution context are tracked based on the generated application artifact.

Rajan discloses generating a reference hash (i.e., computeSHA1Hash) for the software stack up to the requirements monitor in order to identify unauthorized changes in the stack in the future. EX1003, ¶¶214-215; EX1006, p.7. Based on this

reference hash, subsequent changes to the runtime execution context of the software stack are tracked. *Id.* Further, as discussed with respect to limitation 1[e], Munetoh discloses sending Validation Results (VR) to the platform seeking attestation. EX1005, p.5, Fig.-1; EX1003, ¶221. A PHOSITA would have been motivated to combine the current static hash disclosed by Rajan with the sending of the Validation Result (VR) in Munetoh to reduce errors and enable policy negotiation. EX1003, ¶¶222, 304; EX1005, p.6; EX1006, p.4.

Specifically, Munetoh discloses that re-initiated authentication is expected between client and server as indicated by having the client do its own initial checks to reduce networking load caused by obvious errors that would cause attestation to fail. EX1003, ¶¶221-222; EX1005, p.6. Likewise, Rajan anticipates a back-and-forth negotiation over security/requirements policy before a valid attestation is issued. EX1006, p.4. Thus, it would have been obvious to a PHOSITA to include Rajan's computed hash of the attestation checksum in the VR sent back to the attestation client so that the client could know what its current state is recorded as at the attestation server and potentially fix errors if there are any. EX1003, ¶222.

To the extent that the patent owner asserts that logging data corresponds to the "sending" recited in this claim, Rajan also teaches storing the application artifact on the attestation server. EX1003, ¶223(citing 1010, p.20).

For these reasons, a PHOSITA would understand that the combination of Munetoh and Rajan discloses the limitations recited in claim limitation 2[b]. EX1003, ¶¶224-225,304.

Therefore, a PHOSITA would have found the limitations recited in claim 2 to be obvious based on Rajan in view of Munetoh. EX1003, ¶305. As shown, the Rajan-Munetoh combination discloses all limitations of claim 1 from which claim 2 depends. Accordingly, claim 2 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

3. Claim 3

[3a.] The method according to claim 2, further comprising digitally signing the attestation results:

As shown regarding Ground 1, claim limitation 3[a], Munetoh and Rajan disclose the 3[a] limitations. EX1003, ¶308. Further, Munetoh discloses that HTTPS is an alternative for secure communication with WS-Trust, which both Rajan and Munetoh disclose. Therefore, a PHOSITA would have been motivated to combine Munetoh's HTTPS communications with Rajan's disclosures as they are "complementary." EX1003, ¶309(citing EX1008).

[3b.] wherein the attributes further comprise parent-child process associations of the application.

As described in Ground 1, claim limitation 3[b], Munetoh discloses analyzing runtime attributes that include relationships between pieces of software components. EX1003, ¶231-233. Likewise, a PHOSITA would incorporate the

IML data of Munetoh that describes the composition of components since Rajan would be able to determine process relationships. EX1003, ¶310.

Therefore, a PHOSITA would have found the limitations recited in claim 3 to be obvious based on Rajan in view of Munetoh. EX1003, ¶311. As shown, the Rajan-Munetoh combination discloses all limitations of claims 1 and 2 from which claim 3 depends. Accordingly, claim 3 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

4. Claim 4

The method of claim 1, wherein the received security context is an introspective security context, and wherein the execution analysis is a static, dynamic, or virtual analysis of the one or more executable file binaries and the loaded components.

As shown regarding Ground 1, claim 4, Rajan and Munetoh disclose the limitations of claim 4. EX1003, ¶¶239-245, 314-316. As shown, the Rajan-Munetoh combination discloses all limitations of claim 1 from which claim 4 depends. Accordingly, claim 4 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

5. Claim 5

The method of claim 4, wherein the generating of the report indicating security risks associated with the application includes generating, by the attestation server, one or more security assertions that pertain to the received runtime execution context and the received introspective security context.

As shown regarding Ground 1, claim 5, Rajan and Munetoh disclose the limitations of claim 5. EX1003, ¶¶249-251, 318-320. As shown, the Rajan-

Munetoh combination discloses all limitations of claims 1 and 4 from which claim 5 depends. Accordingly, claim 5 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

6. Claim 6

The method according to claim 1, further comprising authenticating the application using a plurality of collaboration services.

As shown regarding Ground 1, claim 6, Rajan and Munetoh disclose the limitations of claim 6. EX1003, ¶¶255-257,322-324. As shown, the Rajan-Munetoh combination discloses all limitations of claim 1 from which claim 6 depends. Accordingly, claim 6 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

7. Claim 7

The method according to claim 1, further comprising controlling a user's transaction with the application by applying a set of authorization rules in accordance with the attestation results.

As shown regarding Ground 1, claim 7, Rajan and Munetoh disclose the limitations of claim 7. EX1003, ¶¶194,261-262,326. As shown, the Rajan-Munetoh combination discloses all limitations of claim 1 from which claim 7 depends. Accordingly, claim 7 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

9. Claim 9

The method according to claim 1, further comprising providing confidence metrics in the attestation results indicating a level of security risk by different classifications such that a restriction on a user's transaction with the application are applied based on the level of security risk indicated by the confidence metrics in the attestation results.

As shown regarding Ground 1, claim 9, Rajan and Munetoh disclose the limitations of claim 9. EX1003, ¶¶265-267, 328-330. As shown, the Rajan-Munetoh combination discloses all limitations of claim 1 from which claim 9 depends. Accordingly, claim 9 should be found invalid as unpatentable under 35 U.S.C. § 103(a) based on Rajan in view of Munetoh.

VI. NO BASIS FOR DISCRETIONARY DENIAL

There is no basis to deny institution under 35 U.S.C. § 325(d).

There is parallel court litigation, so Fintiv factors are addressed preliminarily. First, no stay has been entered in the parallel litigation. Second, trial is currently set to begin February 5, 2024. Third, it is early in the parallel litigation as neither party has responded to written fact discovery nor taken any other type of discovery, nor have the parties begun the Markman process. Fourth, Defendants in the parallel litigation served preliminary invalidity contentions on March 28, 2023. Fifth, Petitioner and Taasera are parties in the parallel litigation as identified in the Mandatory Notices below and the Challenged Claims are the

same as the claims identified in Taasera infringement contentions. As such, the first five Fintiv factors do not favor discretionary denial. Moreover, even if that were not the case, the merits of the grounds raised in this Petition are particularly strong for all of the reasons discussed above.

Discretionary denial is strongly disfavored here because the “same or substantially the same art or arguments” were not presented to the Office, and the Office “erred in a manner material” to patentability by not asserting the Halperin, Inoue, and Patil references. *See Advanced Bionics, LLC v. MED-EL Elektromedizinische Gerate GmbH*, IPR2019-01469, Paper 6, 8-9 (Feb. 13, 2020). Analysis of this issue is guided by the “*Becton*” factors. *Id.*, at 9-11.

Munetoh and Rajan were not cited, individually or in combination, in any Office communications. *See* EX1001, p.1-2. Regarding *Becton* factors (a) (material differences between asserted art and prior art) and (b) (cumulative nature of the asserted art and prior art), Munetoh and Rajan are materially different from the combination of references asserted in Office communications because Munetoh and Rajan disclose the independent claim limitations that attributes comprise one or more executable file binaries of the application and loaded components of the application and the security information comprises an execution analysis of the one or more executable file binaries and the loaded components. Thus, the individual references Munetoh and Rajan are not merely cumulative of references the Office

relied on. Regarding *Becton* factor (d) (extent of overlap), the arguments in this Petition do not overlap with arguments made during examination.

Even if the Board finds that the “same or substantially the same” art or arguments were presented to the Office, discretionary denial under § 325(d) is still not proper because the Office made a “material error” in not considering the Munetoh and Rajan references.

Regarding *Becton* factors (c) (evaluation of art during examination), none of Munetoh and Rajan were used in any claim rejections, weighing strongly against exercising discretion. *See Celco Partnership v. Huawei Device Co.*, IPR2020-01117, Paper 10, 13 (Feb. 3, 2021).

Regarding *Becton* factor (e) (examiner error) and (f) (other facts), this petition specifically points out and explains how Munetoh and Rajan discloses all limitations of the challenged claims.

VII. CONCLUSION

Inter partes review of claims 1-7 and 9 of U.S. Patent No. 8,327,441 is requested based on the grounds described above.

VIII. Mandatory Notices (37 C.F.R. § 42.8(a)(1))

REAL PARTY IN INTEREST:

The real parties-in-interest is Trend Micro, Inc., Trend Micro America, Inc., and Trend Micro Incorporated. Trend Micro, Inc. is a wholly owned subsidiary of

Trend Micro America, Inc., which is a wholly owned subsidiary of Trend Micro Incorporated, a Japanese Corporation.

RELATED MATTERS:

Pursuant to 37 C.F.R. § 42.8(b)(2), Petitioner identifies the following related matters.

The ‘441 patent is at issue in the following district court litigations

- *In re: Taasera Licensing LLC, Patent Litigation*, 2:22-md-03042 (E.D. Tex.), which was consolidated by the Judicial Panel on Multidistrict Litigation on August 3, 2022.

LEAD AND BACKUP COUNSEL:

Pursuant to 37 C.F.R. § 42.8(b)(3) and 37 C.F.R. § 42.10(a), Petitioner appoints:

Robert M. Hansen (Reg. No. 43,656) as lead counsel,

Scott B. Amankwatia (Reg. No. 78,226) as back-up counsel.

Matthew Anderson (Reg. No. 78,257) as back-up counsel, and

Pursuant to 37 C.F.R. § 42.10(b), a power of attorney is being filed with this designation of counsel.

SERVICE INFORMATION:

Petitioner provides the following service information for designated counsel:

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive

15th Floor
Tel: (703)-391-2900
Fax: (703)-391-2900

Petitioner consents to electronic service by email to the email addresses:

rhansen@marburylaw.com;

samankwatia@marburylaw.com;

manderson@marburylaw.com; and

pat-docketing@marburylaw.com.

PAYMENT OF FEES

The PTO is authorized to charge any fees due during this proceeding to
Deposit Account No. 18-1579.

Respectfully submitted,
The Marbury Law Group, PLLC



Robert M. Hansen, Reg. No. 43,656
Scott B. Amankwatia, Reg. No. 78,257

Attorneys for Petitioner

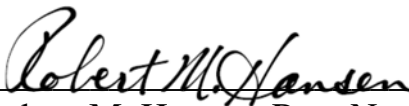
Date: April 6, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703) 391-2900
Fax: 703-391-2901

CERTIFICATION OF SERVICE (37 C.F.R. §§ 42.6(e), 42.105(a))

The undersigned hereby certifies that the above-captioned **PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO. 8,327,441**, Petitioner's power of attorney, and all associated exhibits were served in their entireties on April 6, 2023, on the following party via FedEx:

Buchanan, Ingersoll & Rooney PC
1737 King Street
Suite 500
Alexandria, VA 22314


Robert M. Hansen, Reg. No. 43,656
Attorney for Petitioner

Date: April 6, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703)-391-2900
Fax: 703-391-2901

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION

Pursuant to 37 C.F.R. § 42.24 *et seq.*, the undersigned certifies that this Petition complies with the 14,000-word type-volume limitation. The Petition contains **11,928** words, excluding the table of contents, table of authorities, Table of Exhibits, mandatory notices under § 42.8, the signature block, the claims appendix, certificate of service, certificate of word count, appendix of exhibits, and words that appear in figures copied from the cited prior art.

The Marbury Law Group, PLLC

A handwritten signature in black ink, reading "Robert M. Hansen", written over a horizontal line.

Robert M. Hansen, Reg. 43,656
Attorney for Petitioner

Date: April 6, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: 703-391-2900
Fax: 703-391-2901