UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
14/741,066	06/16/2015	James Kwon	12172US05CON	1051

157376 7590 02/03/2020
Xsensus/Broadcom
200 Daingerfield Road, Suite 201
Alexandria, VA 22314

EXAMINER
CUNNINGHAM, KEVIN M

ART UNIT	PAPER NUMBER
2461	

NOTIFICATION DATE	DELIVERY MODE
02/03/2020	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

Faith.Baggett@Xsensus.com
Sandy.Miles@Xsensus.com
anaquadocketing@xsensus.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte JAMES KWON and JOSEPH AMMIRATION

Appeal 2019-000724
Application 14/741,066
Technology Center 2400

Before CATHERINE SHIANG, JAMES W. DEJMEK, and
JOYCE CRAIG, *Administrative Patent Judges*.

SHIANG, *Administrative Patent Judge*.

DECISION ON APPEAL

Appellant[1] appeals under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1–14, which are all the claims pending and rejected in the application. We have jurisdiction under 35 U.S.C. § 6(b).

We reverse.

[1] We use "Appellant" to refer to "applicant" as defined in 37 C.F.R. § 1.42. Appellant identifies Brocade Communications Systems LLC as the real party in interest. Appeal. Br. 3.

STATEMENT OF THE CASE

Introduction

The present invention relates to "network switches and management tools, and more particularly to switches and management tools for executing and deploying network services." Spec. ¶ 4.

> In preferred embodiments according to the present invention, virtual machine environments are provided in the switches that form a network. The virtual machines are used to execute network services previously performed by dedicated appliances. The virtual machines can be executed on a single multi-core processor in combination with normal switch functions or on services processor boards added for the purpose of executing the services. The packet processors in the switch ports analyze incoming packets and add a services tag containing services entries to any packets requiring available network services. Each switch reviews the services tag and performs any network services resident on that switch, removing the services entry for that service. This allows services to be deployed at the optimal locations in the network, such as the edges or the core, rather than requiring multiple traverses of links to use dedicated appliances. The network services may be deployed to the switches by use of a graphical user interface and drag and drop operations. A topology view of the network is presented, along with network services that may be deployed. Multiple services may be selected and dragged to a single switch or multiple switches may be selected and then the services selected and dragged to the selected switches. The management tool deploys the network services software, with virtual machines being instantiated on the switches as needed to support the network services.

Spec. ¶ 7. Claim 1 is exemplary:

> 1. A network device comprising:
> at least one processor core and associated memory;
> a plurality of service software instances executing on said at least one processor core and stored in said associated

memory, at least one of said plurality of service software instances providing a network service;

a memory storing a local network services table; and

a packet analyzer coupled to said at least one processor core and said local network service table which reviews received packets for a services tag, the services tag specifying network services to be performed on the packet, compares the services tag to said local network service table to determine network services to be performed by the network device and directs the received packet to a service software instance of said plurality of service software instances which is providing one of the determined network services,

wherein said at least one service software instance providing network services updates the services tag in the received packet after performing said at least one service software instance's operations.

References and Rejections[2]

Claims Rejected	**35 U.S.C. §**	**References**
1–5, 7–12, 14	103	Aybay (US 8,284,664 B1; issued October 9, 2012), Kakadia (US 7,382,725 B1; issued June 3, 2008)
6, 13	103	Aybay, Kakadia, Yun (US 2007/0130309 A1; published June 7, 2007)

[2] Throughout this opinion, we refer to the (1) Final Office Action dated April 12, 2018 ("Final Act."); (2) Appeal Brief dated June 8, 2018 ("Appeal Br."); (3) Examiner's Answer dated September 6, 2018 ("Ans."); and (4) Reply Brief dated November 6, 2018 ("Reply Br.").

ANALYSIS

Obviousness

We have reviewed the Examiner's rejection in light of Appellant's contentions and the evidence of record. We concur with Appellant's contentions that the Examiner erred in determining the cited reference portions teach

> *a packet analyzer* coupled to said at least one processor core and said local network service table *which reviews received packets for a services tag*, the services tag specifying network services to be performed on the packet, compares the services tag to said local network service table to determine network services to be performed by the network device and directs the received packet to a service software instance of said plurality of service software instances which is providing one of the determined network services,

as recited in independent claim 1 (emphases added). *See* Appeal Br. 7–26; Reply Br. 3–21.

The Examiner cites Aybay and finds

> a packet analyzer coupled to said at least one processor core and said local network service table which reviews packets for a services tag (forwarding module using classification and ACL tables to obtain service tag from packet, C: 4 R: 18-28).

Final Act. 2 (emphasis omitted). In response to Appellant's arguments, the Examiner further finds:

> The claim does not explicitly state the received packet has a service tag already but reviews the packet for a service tag (could be seen as reviewing the packet to determine a service tag). Aybay can be seen to disclose this alone, see C: 4 R: 17-29, the forwarding module may obtain a service tag for the data unit that is received at the line interface based on classification and ACL lookup, so Aybay is reviewing received packets for a

> service tag. Kakadia discloses attaching a tag, which identifies services to be performed and schedules the packet on the switch fabric, C: 2 R: 15-19, meaning another switch in the fabric received a tagged packet. Therefore whether the claim limitation is interpreted to be seen as the received packet already having the service tag or the packet is reviewed to determine a tag, the limitation is disclosed by the references.

Ans. 3–4; *see also* Ans. 10–11.

We begin by noting the Examiner cites (i) Aybay's "forwarding module **302**" for teaching the claimed "packet analyzer," and (ii) Aybay's "data unit" for teaching the claimed "received packets." *See* Final Act. 2; Ans. 3. Aybay explains the service tag, classification table, and ACL [access control list] as follows:

> A service tag may be obtained based on the class of the data unit (block 508). In one implementation, the service tag may be obtained by looking up ACL 312 using the class as a key, by retrieving information from the ACL lookup, and by creating the service tag in accordance with the retrieved information.

Aybay 6:4–9.

> Classification table 306 may include rules for categorizing data units based on data unit headers. Examples of classification rules may include rules for performing an ACL lookup (e.g., if a field in a data unit header is one of specified values, perform a lookup into ACL 312), for performing a policy based routing (e.g., if a data unit header is a telephony data unit, route the data unit from X to Y via asynchronous transfer mode (ATM)), and for rendering differentiated quality of service (QoS).

Aybay 4:37–45.

> ACL 312 may include a list of rules that detail services or service ports that are available on network element 102. By performing an ACL lookup based on the data unit's class, a service tag may be obtained.

Aybay 4:59–62.

Contrary to the Examiner's assertion, Aybay teaches using the "forwarding module **302**" to obtain a service tag by "looking up ACL 312 using the class as a key, by retrieving information from the ACL lookup, and by creating the service tag in accordance with the retrieved information." Aybay 4:24–25, 6:4–9. After obtaining the service tag, the "forwarding module **302**" may "augment the data unit by appending the service tag . . . to the data unit." Aybay 4:24–27. Therefore, the cited Aybay portions teach using the "forwarding module **302**" to look up information from the classification table and ACL—not the claimed "received packets"—to obtain the service tag and append the service tag to the data unit. Because the Examiner cites Aybay's data unit for teaching the claimed "received packets," the cited Aybay portions teach obtaining the service tag and *appending* the service tag to the claimed "received packets"—not "review[ing] received packets for a services tag," as required by claim 1.

Further, as acknowledged by the Examiner (Ans. 4), the cited Kakadia portions teach "attach[ing] a tag to the packet" (Kakadia 2:16–17)—not "review[ing] received packets for a services tag," as required by claim 1.

Because the Examiner fails to provide sufficient evidence or explanation to support the rejection, we are constrained by the record to reverse the Examiner's rejection of claim 1.

Independent claim 8 recites "analyzing received packets for a services tag." For similar reasons, the Examiner cites the references for teaching appending (or attaching) the services tag to the received packets, not "analyzing received packets for a services tag," as required by claim 8. Therefore, for similar reasons, we reverse the Examiner's rejection of independent claim 8.

We also reverse the Examiner's rejection of corresponding dependent claims 2–7 and 9–14. Although the Examiner cites an additional reference for rejecting some dependent claims, the Examiner has not shown the additional reference overcomes the deficiency discussed above in the rejection of claims 1 and 8.

CONCLUSION

We reverse the Examiner's decision rejecting claims 1–14 under 35 U.S.C. § 103.

In summary:

Claims Rejected	**35 U.S.C. §**	**Reference(s)/Basis**	**Affirmed**	**Reversed**
1–5, 7–12, 14	103	Aybay, Kakadia		1–5, 7–12, 14
6, 13	103	Aybay, Kakadia, Yun		6, 13
Overall Outcome				1–14

REVERSED