



(12) **United States Patent**
Vobbilisetty et al.

(10) **Patent No.:** **US 9,270,486 B2**
(45) **Date of Patent:** **Feb. 23, 2016**

(54) **NAME SERVICES FOR VIRTUAL CLUSTER SWITCHING**

(75) Inventors: **Suresh Vobbilisetty**, San Jose, CA (US);
Phanidhar Koganti, Sunnyvale, CA (US); **Jesse B. Willeke**, Broomfield, CO (US)

(73) Assignee: **BROCADE COMMUNICATIONS SYSTEMS, INC.**, San Jose, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 136 days.

(21) Appl. No.: **13/092,752**

(22) Filed: **Apr. 22, 2011**

(65) **Prior Publication Data**

US 2011/0299535 A1 Dec. 8, 2011

Related U.S. Application Data

(60) Provisional application No. 61/352,264, filed on Jun. 7, 2010, provisional application No. 61/380,803, filed on Sep. 8, 2010.

(51) **Int. Cl.**

H04L 12/46 (2006.01)
H04L 12/701 (2013.01)
H04L 12/931 (2013.01)
H04L 12/939 (2013.01)

(52) **U.S. Cl.**

CPC **H04L 12/4633** (2013.01); **H04L 12/4625** (2013.01); **H04L 45/00** (2013.01); **H04L 49/555** (2013.01); **H04L 49/70** (2013.01)

(58) **Field of Classification Search**

CPC H04L 49/00; H04L 45/00; H04L 12/4625; H04L 45/66; G06F 9/3885

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,390,173 A 2/1995 Spinney
5,802,278 A 9/1998 Isfeld
5,959,968 A 9/1999 Chin
5,973,278 A 10/1999 Wehrli, III
5,983,278 A 11/1999 Chong
6,041,042 A 3/2000 Bussiere
6,085,238 A 7/2000 Yuasa
6,104,696 A 8/2000 Kadambi

(Continued)

FOREIGN PATENT DOCUMENTS

CN 102801599 A 11/2012
EP 1398920 A2 3/2004

(Continued)

OTHER PUBLICATIONS

“Switched Virtual Internetworking moved beyond bridges and routers”, 8178 Data Communications Sep. 23, 1994, No. 12, New York.

(Continued)

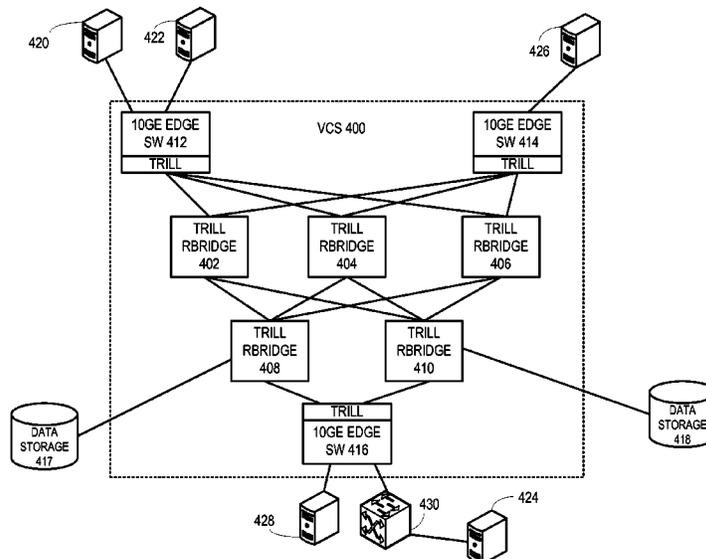
Primary Examiner — Jung Park

(74) *Attorney, Agent, or Firm* — Shun Yao; Park, Vaughan, Fleming & Dowler LLP

(57) **ABSTRACT**

One embodiment of the present invention provides a switch that facilitates name services in a virtual cluster switch. The switch includes a name service database indicating at least one media access control (MAC) address learned at a second switch. The switch also includes a control mechanism. During operation, the control mechanism distributes information on a locally learned MAC address to the second switch. In addition, the control mechanism receives information on a MAC address learned at the second switch.

22 Claims, 15 Drawing Sheets



(56)

References Cited

U.S. PATENT DOCUMENTS

6,185,214	B1	2/2001	Schwartz	8,369,347	B2	2/2013	Xiong
6,185,241	B1	2/2001	Sun	8,392,496	B2	3/2013	Linden
6,438,106	B1	8/2002	Pillar	8,462,774	B2	6/2013	Page
6,542,266	B1	4/2003	Phillips	8,467,375	B2	6/2013	Blair
6,633,761	B1	10/2003	Singhal	8,520,595	B2	8/2013	Yadav
6,873,602	B1	3/2005	Ambe	8,599,850	B2	12/2013	Jha
6,956,824	B2	10/2005	Mark	8,599,864	B2	12/2013	Chung
6,957,269	B2	10/2005	Williams	8,615,008	B2	12/2013	Natarajan
6,975,581	B1	12/2005	Medina	2001/0055274	A1	12/2001	Hegge
6,975,864	B2	12/2005	Singhal	2002/0019904	A1	2/2002	Katz
7,016,352	B1	3/2006	Chow	2002/0021701	A1	2/2002	Lavian
7,173,934	B2	2/2007	Lapuh	2002/0091795	A1	7/2002	Yip
7,197,308	B2	3/2007	Singhal	2003/0041085	A1	2/2003	Sato
7,206,288	B2	4/2007	Cometto	2003/0123393	A1	7/2003	Feuerstraeter
7,310,664	B1	12/2007	Merchant	2003/0174706	A1	9/2003	Shankar
7,313,637	B2	12/2007	Tanaka	2003/0189905	A1	10/2003	Lee
7,315,545	B1	1/2008	Chowdhury et al.	2004/0001433	A1	1/2004	Gram
7,316,031	B2	1/2008	Griffith	2004/0010600	A1	1/2004	Baldwin
7,330,897	B2	2/2008	Baldwin	2004/0049699	A1	3/2004	Griffith
7,380,025	B1	5/2008	Riggins	2004/0117508	A1	6/2004	Shimizu
7,430,164	B2	9/2008	Bare	2004/0120326	A1	6/2004	Yoon
7,453,888	B2	11/2008	Zabihi	2004/0156313	A1	8/2004	Hofmeister et al.
7,477,894	B1	1/2009	Sinha	2004/0165595	A1	8/2004	Holmgren
7,480,258	B1	1/2009	Shuen	2004/0213232	A1	10/2004	Regan
7,508,757	B2	3/2009	Ge	2005/0007951	A1	1/2005	Lapuh
7,558,195	B1	7/2009	Kuo	2005/0044199	A1	2/2005	Shiga
7,558,273	B1	7/2009	Grosser, Jr.	2005/0094568	A1	5/2005	Judd
7,571,447	B2	8/2009	Ally	2005/0094630	A1	5/2005	Valdevit
7,599,901	B2	10/2009	Mital	2005/0122979	A1	6/2005	Gross
7,688,736	B1	3/2010	Walsh	2005/0157645	A1	7/2005	Rabie et al.
7,688,960	B1	3/2010	Aubuchon	2005/0157751	A1	7/2005	Rabie
7,690,040	B2	3/2010	Frattura	2005/0169188	A1	8/2005	Cometto
7,706,255	B1	4/2010	Kondrat et al.	2005/0195813	A1	9/2005	Ambe
7,716,370	B1	5/2010	Devarapalli	2005/0213561	A1	9/2005	Yao
7,729,296	B1	6/2010	Choudhary	2005/0265356	A1	12/2005	Kawarai
7,787,480	B1	8/2010	Mehta	2005/0278565	A1	12/2005	Frattura
7,792,920	B2	9/2010	Istvan	2006/0018302	A1	1/2006	Ivaldi
7,796,593	B1	9/2010	Ghosh	2006/0023707	A1	2/2006	Makishima et al.
7,808,992	B2	10/2010	Homchaudhuri	2006/0034292	A1	2/2006	Wakayama
7,836,332	B2	11/2010	Hara	2006/0059163	A1	3/2006	Frattura
7,843,906	B1	11/2010	Chidambaram et al.	2006/0062187	A1	3/2006	Rune
7,843,907	B1	11/2010	Abou-Emara	2006/0072550	A1	4/2006	Davis
7,860,097	B1	12/2010	Lovett	2006/0083254	A1	4/2006	Ge
7,898,959	B1	3/2011	Arad	2006/0168109	A1	7/2006	Warmenhoven
7,924,837	B1	4/2011	Shabtay	2006/0184937	A1	8/2006	Abels
7,937,756	B2	5/2011	Kay	2006/0221960	A1	10/2006	Borgione
7,949,638	B1	5/2011	Goodson	2006/0235995	A1	10/2006	Bhatia
7,957,386	B1	6/2011	Aggarwal	2006/0242311	A1	10/2006	Mai
8,027,354	B1	9/2011	Portolani	2006/0245439	A1	11/2006	Sajassi
8,054,832	B1	11/2011	Shukla	2006/0251067	A1	11/2006	DeSanti
8,068,442	B1	11/2011	Kompella	2006/0256767	A1	11/2006	Suzuki
8,078,704	B2	12/2011	Lee	2006/0265515	A1	11/2006	Shiga
8,102,781	B2	1/2012	Smith	2006/0285499	A1	12/2006	Tzeng
8,102,791	B2	1/2012	Tang	2006/0291388	A1	12/2006	Amdahl
8,116,307	B1	2/2012	Thesayi	2007/0036178	A1	2/2007	Hares
8,125,928	B2	2/2012	Mehta	2007/0086362	A1	4/2007	Kato
8,134,922	B2	3/2012	Elangovan	2007/0094464	A1	4/2007	Sharma
8,155,150	B1	4/2012	Chung	2007/0097968	A1	5/2007	Du
8,160,063	B2	4/2012	Maltz	2007/0116224	A1	5/2007	Burke
8,160,080	B1	4/2012	Arad	2007/0177597	A1	8/2007	Ju
8,170,038	B2	5/2012	Belanger	2007/0183313	A1	8/2007	Narayanan
8,194,674	B1	6/2012	Pagel	2007/0211712	A1	9/2007	Fitch
8,195,774	B2	6/2012	Lambeth	2007/0274234	A1	11/2007	Kubota
8,204,061	B1	6/2012	Sane	2007/0289017	A1	12/2007	Copeland, III
8,213,313	B1	7/2012	Doiron	2008/0052487	A1	2/2008	Akahane
8,213,336	B2	7/2012	Smith	2008/0065760	A1	3/2008	Damm
8,230,069	B2	7/2012	Korupolu	2008/0080517	A1	4/2008	Roy
8,239,960	B2	8/2012	Frattura	2008/0101386	A1	5/2008	Gray
8,249,069	B2	8/2012	Raman	2008/0112400	A1	5/2008	Dunbar et al.
8,270,401	B1	9/2012	Barnes	2008/0133760	A1	6/2008	Berkvens et al.
8,295,291	B1	10/2012	Ramanathan	2008/0159277	A1	7/2008	Vobbilisetty
8,301,686	B1	10/2012	Appajodu	2008/0172492	A1	7/2008	Raghunath
8,339,994	B2	12/2012	Gnanasekaran	2008/0181196	A1	7/2008	Regan
8,351,352	B1	1/2013	Eastlake, III	2008/0181243	A1	7/2008	Vobbilisetty
8,369,335	B2	2/2013	Jha	2008/0186981	A1	8/2008	Seto
				2008/0205377	A1	8/2008	Chao
				2008/0219172	A1	9/2008	Mohan
				2008/0225853	A1	9/2008	Melman
				2008/0228897	A1	9/2008	Ko

(56)

References Cited

U.S. PATENT DOCUMENTS

2008/0240129 A1 10/2008 Elmeleegy
 2008/0267179 A1 10/2008 LaVigne
 2008/0285555 A1* 11/2008 Ogasahara 370/389
 2008/0298248 A1 12/2008 Roeck
 2008/0310342 A1 12/2008 Kruys
 2009/0037607 A1* 2/2009 Farinacci et al. 709/249
 2009/0044270 A1 2/2009 Shelly
 2009/0067422 A1 3/2009 Poppe
 2009/0067442 A1 3/2009 Killian
 2009/0079560 A1 3/2009 Fries
 2009/0080345 A1 3/2009 Gray
 2009/0083445 A1 3/2009 Ganga
 2009/0092042 A1 4/2009 Yuhara
 2009/0092043 A1* 4/2009 Lapuh et al. 370/228
 2009/0106405 A1 4/2009 Mazarick
 2009/0116381 A1 5/2009 Kanda
 2009/0129384 A1 5/2009 Regan
 2009/0138752 A1 5/2009 Graham
 2009/0161670 A1 6/2009 Shepherd
 2009/0168647 A1 7/2009 Holness
 2009/0199177 A1 8/2009 Edwards
 2009/0204965 A1 8/2009 Tanaka
 2009/0213783 A1 8/2009 Moreton
 2009/0222879 A1 9/2009 Kostal
 2009/0245137 A1 10/2009 Hares
 2009/0245242 A1 10/2009 Carlson
 2009/0252049 A1 10/2009 Ludwig
 2009/0260083 A1 10/2009 Szeto
 2009/0279558 A1 11/2009 Davis
 2009/0292858 A1 11/2009 Lambeth
 2009/0316721 A1 12/2009 Kanda
 2009/0323708 A1 12/2009 Ihle
 2009/0327392 A1 12/2009 Tripathi
 2009/0327462 A1 12/2009 Adams
 2010/0027420 A1 2/2010 Smith
 2010/0054260 A1 3/2010 Pandey
 2010/0061269 A1 3/2010 Banerjee
 2010/0074175 A1 3/2010 Banks
 2010/0097941 A1 4/2010 Carlson
 2010/0103813 A1 4/2010 Allan
 2010/0103939 A1 4/2010 Carlson
 2010/0131636 A1 5/2010 Suri
 2010/0158024 A1 6/2010 Sajassi
 2010/0165877 A1 7/2010 Shukia
 2010/0165995 A1 7/2010 Mehta
 2010/0169467 A1 7/2010 Shukla
 2010/0169948 A1 7/2010 Budko
 2010/0182920 A1 7/2010 Matsuoka
 2010/0215049 A1 8/2010 Raza
 2010/0220724 A1 9/2010 Rabie
 2010/0226368 A1 9/2010 Mack-Crane
 2010/0226381 A1 9/2010 Mehta
 2010/0246388 A1 9/2010 Gupta
 2010/0257263 A1 10/2010 Casado
 2010/0271960 A1 10/2010 Krygowski
 2010/0281106 A1 11/2010 Ashwood-Smith
 2010/0284418 A1 11/2010 Gray
 2010/0287262 A1 11/2010 Elzur
 2010/0287548 A1 11/2010 Zhou
 2010/0290473 A1 11/2010 Enduri
 2010/0303071 A1 12/2010 Kotalwar
 2010/0303075 A1 12/2010 Tripathi
 2010/0303083 A1 12/2010 Belanger
 2010/0309820 A1 12/2010 Rajagopalan
 2011/0019678 A1 1/2011 Mehta
 2011/0032945 A1 2/2011 Mullooly
 2011/0035498 A1 2/2011 Shah
 2011/0044339 A1 2/2011 Kotalwar
 2011/0064086 A1 3/2011 Xiong
 2011/0072208 A1 3/2011 Gulati
 2011/0085560 A1 4/2011 Chawla
 2011/0085563 A1 4/2011 Kotha
 2011/0110266 A1 5/2011 Li
 2011/0134802 A1 6/2011 Rajagopalan
 2011/0134803 A1 6/2011 Dalvi

2011/0134925 A1 6/2011 Safrai
 2011/0142053 A1 6/2011 Van Der Merwe
 2011/0142062 A1 6/2011 Wang
 2011/0161494 A1 6/2011 Mcdysan
 2011/0161695 A1 6/2011 Okita
 2011/0188373 A1 8/2011 Saito
 2011/0194403 A1 8/2011 Sajassi
 2011/0194563 A1 8/2011 Shen
 2011/0228780 A1 9/2011 Ashwood-Smith
 2011/0231574 A1 9/2011 Saunderson
 2011/0235523 A1 9/2011 Jha
 2011/0243133 A9 10/2011 Villait
 2011/0243136 A1* 10/2011 Raman et al. 370/392
 2011/0246669 A1 10/2011 Kanada
 2011/0255538 A1 10/2011 Srinivasan
 2011/0255540 A1 10/2011 Mizrahi
 2011/0261828 A1 10/2011 Smith
 2011/0268120 A1 11/2011 Vobbilisetty
 2011/0273988 A1 11/2011 Tourrilhes
 2011/0274114 A1 11/2011 Dhar
 2011/0286457 A1 11/2011 Ee
 2011/0296052 A1 12/2011 Guo
 2011/0299391 A1 12/2011 Vobbilisetty
 2011/0299414 A1 12/2011 Yu
 2011/0299527 A1 12/2011 Yu
 2011/0299528 A1 12/2011 Yu
 2011/0299531 A1 12/2011 Yu
 2011/0299532 A1 12/2011 Yu
 2011/0299533 A1 12/2011 Yu
 2011/0299534 A1 12/2011 Koganti
 2011/0299535 A1 12/2011 Vobbilisetty
 2011/0299536 A1 12/2011 Cheng
 2011/0317703 A1 12/2011 Dunbar et al.
 2012/0011240 A1 1/2012 Hara
 2012/0014261 A1 1/2012 Salam
 2012/0014387 A1 1/2012 Dunbar
 2012/0020220 A1 1/2012 Sugita
 2012/0027017 A1 2/2012 Rai
 2012/0033663 A1 2/2012 Guichard
 2012/0033665 A1 2/2012 Jacob Da Silva
 2012/0033669 A1 2/2012 Mohandas
 2012/0099602 A1 4/2012 Nagapudi
 2012/0106339 A1 5/2012 Mishra
 2012/0131097 A1 5/2012 Baykal
 2012/0131289 A1 5/2012 Taguchi
 2012/0163164 A1 6/2012 Terry
 2012/0177039 A1 7/2012 Berman
 2012/0243539 A1 9/2012 Keesara
 2012/0275347 A1 11/2012 Banerjee
 2012/0294192 A1 11/2012 Masood
 2012/0294194 A1 11/2012 Balasubramanian
 2012/0320800 A1 12/2012 Kamble
 2012/0320926 A1 12/2012 Kamath et al.
 2012/0327937 A1 12/2012 Melman et al.
 2013/0028072 A1 1/2013 Addanki
 2013/0034015 A1 2/2013 Jaiswal
 2013/0067466 A1 3/2013 Combs
 2013/0127848 A1 5/2013 Joshi
 2013/0194914 A1 8/2013 Agarwal
 2013/0250951 A1 9/2013 Koganti
 2013/0259037 A1 10/2013 Natarajan
 2013/0272135 A1 10/2013 Leong
 2014/0105034 A1 4/2014 Sun

FOREIGN PATENT DOCUMENTS

EP 1916807 A2 4/2008
 EP 2001167 A1 12/2008
 WO 2009042919 4/2009
 WO 2010111142 A1 9/2010

OTHER PUBLICATIONS

S. Night et al., "Virtual Router Redundancy Protocol", Network Working Group, XP-002135272, Apr. 1998.
 Eastlake 3rd., Donald et al., "RBridges: TRILL Header Options", Draft-ietf-trill-rbridge-options-00.txt, Dec. 24, 2009.
 J. Touch, et al., "Transparent Interconnection of Lots of Links (TRILL): Problem and Applicability Statement", May 2009.

(56)

References Cited

OTHER PUBLICATIONS

- Perlman, Radia et al., "RBridge VLAN Mapping", Draft-ietf-trill-rbridge-vlan-mapping-01.txt, Dec. 4, 2009.
- Brocade Fabric OS (FOS) 6.2 Virtual Fabrics Feature Frequently Asked Questions, (2009).
- Perlman, Radia "Challenges and Opportunities in the Design of TRILL: a Routed layer 2 Technology", XP-002649647, 2009.
- Nadas, S. et al., "Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6", Mar. 2010.
- Perlman, Radia et al., "RBridges: Base Protocol Specification", draft-ietf-trill-rbridge-protocol-16.txt, Mar. 3, 2010.
- Christensen, M. et al., "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", May 2006.
- Lapuh, Roger et al., "Split Multi-link Trunking (SMLT)", Oct. 2002.
- Lapuh, Roger et al., "Split Multi-link Trunking (SMLT) draft-lapuh-network-smlt-08", 2008.
- Brocade Unveils, "The Effortless Network" Mar. 2012.
- Foundry FastIron Configuration Guide, Software Release FSX 04.2.00b, Software Release FWS 04.3.00, Software Release FGS 05.0.00a, Sep. 2008.
- Brocade, "FastIron and TurboIron 24x Configuration Guide", Feb. 16, 2010.
- Brocade, "FastIron Configuration Guide" Dec. 18, 2009.
- Brocade, "The Effortless Network: Hyperedge Technology for the Campus LAN" 2012.
- Narten, T. et al., "Problem Statement: Overlays for Network Virtualization draft-narten-nvo3-overlay-problem-statement-01", Oct. 31, 2011.
- Knight, Paul et al., "Layer 2 and 3 Virtual Private Networks: Taxonomy, Technology, and Standardization Efforts", Jun. 2004.
- Brocade "An Introduction to Brocade VCS Fabric Technology", Dec. 3, 2012.
- Kreeger, L. et al., "Network Virtualization Overlay Control Protocol Requirements draft-kreeger-nvo3-overlay-cp-00", Jan. 30, 2012.
- Knight, Paul et al., "Network Based IP VPN Architecture using Virtual Routers", May 2003.
- Louati, Wajdi et al., "Network-Based Virtual Personal Overlay Networks Using Programmable Virtual Routers", Jul. 2005.
- Office Action for U.S. Appl. No. 13/533,843, filed Jun. 26, 2012, dated Oct. 21, 2013.
- Office Action for U.S. Appl. No. 13/312,903, filed Dec. 6, 2011, dated Nov. 12, 2013.
- Office Action for U.S. Appl. No. 13/042,259, filed Mar. 7, 2011, dated Jan. 16, 2014.
- Office Action for U.S. Appl. No. 13/092,580, filed Apr. 22, 2011, dated Jan. 10, 2014.
- Office Action for U.S. Appl. No. 13/092,877, filed Apr. 22, 2011, dated Jan. 6, 2014.
- U.S. Appl. No. 12/312,903 Office Action dated Jun. 13, 2013.
- U.S. Appl. No. 13/365,808 Office Action dated Jul. 18, 2013.
- U.S. Appl. No. 13/365,993 Office Action dated Jul. 23, 2013.
- U.S. Appl. No. 13/092,873 Office Action dated Jun. 19, 2013.
- U.S. Appl. No. 13/184,526 Office Action dated May 22, 2013.
- U.S. Appl. No. 13/184,526 Office Action dated Jan. 28, 2013.
- U.S. Appl. No. 13/050,102 Office Action dated May 16, 2013.
- U.S. Appl. No. 13/050,102 Office Action dated Oct. 26, 2012.
- U.S. Appl. No. 13/044,301 Office Action dated Feb. 22, 2013.
- U.S. Appl. No. 13/044,301 Office Action dated Jun. 11, 2013.
- U.S. Appl. No. 13/030,688 Office Action dated Apr. 25, 2013.
- U.S. Appl. No. 13/030,806 Office Action dated Dec. 3, 2012.
- U.S. Appl. No. 13/030,806 Office Action dated Jun. 11, 2013.
- U.S. Appl. No. 13/098,360 Office Action dated May 31, 2013.
- U.S. Appl. No. 13/092,864 Office Action dated Sep. 19, 2012.
- U.S. Appl. No. 12/950,968 Office Action dated Jun. 7, 2012.
- U.S. Appl. No. 12/950,968 Office Action dated Jan. 4, 2013.
- U.S. Appl. No. 13/092,877 Office Action dated Mar. 4, 2013.
- U.S. Appl. No. 12/950,974 Office Action dated Dec. 20, 2012.
- U.S. Appl. No. 12/950,974 Office Action dated May 24, 2012.
- U.S. Appl. No. 13/092,752 Office Action dated Feb. 5, 2013.
- U.S. Appl. No. 13/092,752 Office Action dated Jul. 18, 2013.
- U.S. Appl. No. 13/092,701 Office Action dated Jan. 28, 2013.
- U.S. Appl. No. 13/092,701 Office Action dated Jul. 3, 2013.
- U.S. Appl. No. 13/092,460 Office Action dated Jun. 21, 2013.
- U.S. Appl. No. 13/042,259 Office Action dated Mar. 18, 2013.
- U.S. Appl. No. 13/042,259 Office Action dated Jul. 31, 2013.
- U.S. Appl. No. 13/092,580 Office Action dated Jun. 10, 2013.
- U.S. Appl. No. 13/092,724 Office Action dated Jul. 16, 2013.
- U.S. Appl. No. 13/092,724 Office Action dated Feb. 5, 2013.
- U.S. Appl. No. 13/098,490 Office Action dated Dec. 21, 2012.
- U.S. Appl. No. 13/098,490 Office Action dated Sep. 12, 2013.
- U.S. Appl. No. 13/087,239 Office Action dated May 22, 2013.
- U.S. Appl. No. 13/087,239 Office Action dated Dec. 5, 2012.
- U.S. Appl. No. 12/725,249 Office Action dated Apr. 26, 2013.
- U.S. Appl. No. 12/725,249 Office Action dated Sep. 12, 2012.
- Brocade Unveils "The Effortless Network", <http://newsroom.brocade.com/press-releases/brocade-unveils-the-effortless-network--nasdaq-bred-0859535>, 2012.
- Foundry FastIron Configuration Guide, Software Release FSX 04.2.00b, Software Release FWS 04.3.00, Software Release FGS 05.0.00a, Sep. 26, 2008.
- FastIron and TurboIron 24X Configuration Guide Supporting FSX 05.1.00 for FESX, FWSX, and FSX; FGS 04.3.03 for FGS, FLS and FWS; FGS 05.0.02 for FGS-STK and FLS-STK, FCX 06.0.00 for FCX; and TIX 04.1.00 for TI24X, Feb. 16, 2010.
- FastIron Configuration Guide Supporting Ironware Software Release 07.0.00, Dec. 18, 2009.
- "The Effortless Network: HyperEdge Technology for the Campus LAN", 2012.
- Zhai F. Hu et al. "RBridge: Pseudo-Nickname; draft-hu-trill-pseudonode-nickname-02.txt", May 15, 2012.
- Huang, Nen-Fu et al., "An Effective Spanning Tree Algorithm for a Bridged LAN", Mar. 16, 1992.
- Office Action dated Jun. 6, 2014, U.S. Appl. No. 13/669,357, filed Nov. 5, 2012.
- Office Action dated Feb. 20, 2014, U.S. Appl. No. 13/598,204, filed Aug. 29, 2012.
- Office Action dated May 14, 2014, U.S. Appl. No. 13/533,843, filed Jun. 26, 2012.
- Office Action dated May 9, 2014, U.S. Appl. No. 13/484,072, filed May 30, 2012.
- Office Action dated Feb. 28, 2014, U.S. Appl. No. 13/351,513, filed Jan. 17, 2012.
- Office Action dated Jun. 18, 2014, U.S. Appl. No. 13/440,861, filed Apr. 5, 2012.
- Office Action dated Mar. 6, 2014, U.S. Appl. No. 13/425,238, filed Mar. 20, 2012.
- Office Action dated Jun. 20, 2014, U.S. Appl. No. 13/092,877, filed Apr. 22, 2011.
- Office Action dated Apr. 9, 2014, U.S. Appl. No. 13/092,752, filed Apr. 22, 2011.
- "RBridges: Base Protocol Specification", IETF Draft, Perlman et al., Jun. 26, 2009.
- Office action dated Aug. 29, 2014, U.S. Appl. No. 13/042,259, filed Mar. 7, 2011.
- Office action dated Aug. 21, 2014, U.S. Appl. No. 13/184,526, filed Jul. 16, 2011.
- Office Action for U.S. Appl. No. 13/042,259, filed Mar. 7, 2011, from Jaroenchonwanit, Bunjob, dated Jan. 16, 2014.
- Office Action for U.S. Appl. No. 13/365,993, filed Feb. 3, 2012, dated Jul. 23, 2013.
- Office Action for U.S. Appl. No. 13/030,806, filed Feb. 18, 2011, dated Dec. 3, 2012.
- Office Action for U.S. Appl. No. 13/044,326, filed Mar. 9, 2011, dated Oct. 2, 2013.
- Office Action for U.S. Appl. No. 13/092,877, filed Apr. 22, 2011, dated Sep. 5, 2013.
- Office Action for U.S. Appl. No. 13/098,490, filed May 2, 2011, dated Mar. 27, 2014.
- Office Action for U.S. Appl. No. 13/312,903, filed Dec. 6, 2011, dated Jun. 13, 2013.
- Office Action for U.S. Appl. No. 13/092,873, filed Apr. 22, 2011, dated Nov. 29, 2013.

(56)

References Cited

OTHER PUBLICATIONS

Office Action for U.S. Appl. No. 13/184,526, filed Jul. 16, 2011, dated Dec. 2, 2013.

Office Action for U.S. Appl. No. 13/598,204, filed Aug. 29, 2012, dated Feb. 20, 2014.

'An Introduction to Brocade VCS Fabric Technology', BROCADE white paper, <http://community.brocade.com/docs/DOC-2954>, Dec. 3, 2012.

Office Action for U.S. Appl. No. 13/092,887, dated Jan. 6, 2014.

Office action dated Apr. 26, 2012, U.S. Appl. No. 12/725,249, filed Mar. 16, 2010.

Office action dated Sep. 12, 2012, U.S. Appl. No. 12/725,249, filed Mar. 16, 2010.

Office action dated Dec. 21, 2012, U.S. Appl. No. 13/098,490, filed May 2, 2011.

Office action dated Mar. 27, 2014, U.S. Appl. No. 13/098,490, filed May 2, 2011.

Office action dated Jul. 9, 2013, U.S. Appl. No. 13/098,490, filed May 2, 2011.

Office action dated May 22, 2013, U.S. Appl. No. 13/087,239, filed Apr. 14, 2011.

Office action dated Dec. 5, 2012, U.S. Appl. No. 13/087,239, filed Apr. 14, 2011.

Office action dated Apr. 9, 2014, U.S. Appl. No. 13/092,724, filed Apr. 22, 2011.

Office action dated Feb. 5, 2013, U.S. Appl. No. 13/092,724, filed Apr. 22, 2011.

Office action dated Jun. 10, 2013, U.S. Appl. No. 13/092,580, filed Apr. 22, 2011.

Office action dated Mar. 18, 2013, U.S. Appl. No. 13/042,259, filed Mar. 7, 2011.

Office action dated Mar. 14, 2014, U.S. Appl. No. 13/092,460, filed Apr. 22, 2011.

Office action dated Jun. 21, 2013, U.S. Appl. No. 13/092,460, filed Apr. 22, 2011.

Office action dated Jan. 28, 2013, U.S. Appl. No. 13/092,701, filed Apr. 22, 2011.

Office action dated Mar. 26, 2014, U.S. Appl. No. 13/092,701, filed Apr. 22, 2011.

Office action dated Jul. 3, 2013, U.S. Appl. No. 13/092,701, filed Apr. 22, 2011.

Office action dated Jul. 18, 2013, U.S. Appl. No. 13/092,752, filed Apr. 22, 2011.

Office action dated Dec. 20, 2012, U.S. Appl. No. 12/950,974, filed Nov. 19, 2010.

Office action dated May 24, 2012, U.S. Appl. No. 12/950,974, filed Nov. 19, 2010.

Office action dated Sep. 5, 2013, U.S. Appl. No. 13/092,877, filed Apr. 22, 2011.

Office action dated Mar. 4, 2013, U.S. Appl. No. 13/092,877, filed Apr. 22, 2011.

Office action dated Jan. 4, 2013, U.S. Appl. No. 12/950,968, filed Nov. 19, 2010.

Office action dated Jun. 7, 2012, U.S. Appl. No. 12/950,968, filed Nov. 19, 2010.

Office action dated Sep. 19, 2012, U.S. Appl. No. 13/092,864, filed Apr. 22, 2011.

Office action dated May 31, 2013, U.S. Appl. No. 13/098,360, filed Apr. 29, 2011.

Office action dated Dec. 3, 2012, U.S. Appl. No. 13/030,806, filed Feb. 18, 2011.

Office action dated Apr. 22, 2014, U.S. Appl. No. 13/030,806, filed Feb. 18, 2011.

Office action dated Jun. 11, 2013, U.S. Appl. No. 13/030,806, filed Feb. 18, 2011.

Office action dated Apr. 25, 2013, U.S. Appl. No. 13/030,688, filed Feb. 18, 2011.

Office action dated Feb. 22, 2013, U.S. Appl. No. 13/044,301, filed Mar. 9, 2011.

Office action dated Jun. 11, 2013, U.S. Appl. No. 13/044,301, filed Mar. 9, 2011.

Office action dated Oct. 26, 2012, U.S. Appl. No. 13/050,102, filed Mar. 17, 2011.

Office action dated May 16, 2013, U.S. Appl. No. 13/050,102, filed Mar. 17, 2011.

Office action dated Aug. 4, 2014, U.S. Appl. No. 13/050,102, filed Mar. 17, 2011.

Office action dated Jan. 28, 2013, U.S. Appl. No. 13/148,526, filed Jul. 16, 2011.

Office action dated May 22, 2013, U.S. Appl. No. 13/148,526, filed Jul. 16, 2011.

Office action dated Jun. 19, 2013, U.S. Appl. No. 13/092,873, filed Apr. 22, 2011.

Office action dated Jul. 18, 2013, U.S. Appl. No. 13/365,808, filed Feb. 3, 2012.

Office action dated Jun. 13, 2013, U.S. Appl. No. 13/312,903, filed Dec. 6, 2011.

Lapuh, Roger et al., 'Split Multi-link Trunking (SMLT) draft-lapuh-network-smlt-08', Jan. 2009.

Office Action for U.S. Appl. No. 13/030,688, filed Feb. 18, 2011, dated Jul. 17, 2014.

Office Action for U.S. Appl. No. 13/044,326, filed Mar. 9, 2011, dated Jul. 7, 2014.

Office Action for U.S. Appl. No. 13/092,752, filed Apr. 22, 2011, dated Apr. 9, 2014.

Office Action for U.S. Appl. No. 13/092,873, filed Apr. 22, 2011, dated Jul. 25, 2014.

Office Action for U.S. Appl. No. 13/092,877, filed Apr. 22, 2011, dated Jun. 20, 2014.

Office Action for U.S. Appl. No. 13/351,513, filed Jan. 17, 2012, dated Jul. 24, 2014.

Office Action for U.S. Appl. No. 13/425,238, filed Mar. 20, 2012, dated Mar. 6, 2014.

Office Action for U.S. Appl. No. 13/556,061, filed Jul. 23, 2012, dated Jun. 6, 2014.

Office Action for U.S. Appl. No. 13/742,207 dated Jul. 24, 2014, filed Jan. 15, 2013.

Office Action for U.S. Appl. No. 13/950,974, filed Nov. 19, 2010, dated Dec. 2, 2012.

Office Action for U.S. Appl. No. 13/087,239, filed Apr. 14, 2011, dated Dec. 5, 2012.

Office Action for U.S. Appl. No. 13/351,513, filed Jan. 17, 2012, dated Feb. 28, 2014.

Perlman R: 'Challenges and opportunities in the design of TRILL: a routed layer 2 technology', 2009 IEEE GLOBECOM Workshops, Honolulu, HI, USA, Piscataway, NJ, USA, Nov. 30, 2009, pp. 1-6, XP002649647, DOI: 10.1109/GLOBECOM.2009.5360776 ISBN: 1-4244-5626-0 [retrieved on Jul. 19, 2011].

TRILL Working Group Internet-Draft Intended status: Proposed Standard RBridges: Base Protocol Specification Mar. 3, 2010.

Office action dated Aug. 14, 2014, U.S. Appl. No. 13/092,460, filed Apr. 22, 2011.

Office action dated Jul. 7, 2014, for U.S. Appl. No. 13/044,326, filed Mar. 9, 2011.

Office Action dated Dec. 19, 2014, for U.S. Appl. No. 13/044,326, filed Mar. 9, 2011.

Office Action for U.S. Appl. No. 13/092,873, filed Apr. 22, 2011, Nov. 7, 2014.

Office Action for U.S. Appl. No. 13/092,877, filed Apr. 22, 2011, dated Nov. 10, 2014.

Office Action for U.S. Appl. No. 13/157,942, filed Jun. 10, 2011.

Mckeown, Nick et al. "OpenFlow: Enabling Innovation in Campus Networks", Mar. 14, 2008, www.openflow.org/documents/openflow-wp-latest.pdf.

Office Action for U.S. Appl. No. 13/044,301, dated Mar. 9, 2011.

Office Action for U.S. Appl. No. 13/184,526, filed Jul. 16, 2011, dated Jan. 5, 2015.

Office Action for U.S. Appl. No. 13/598,204, filed Aug. 29, 2012, dated Jan. 5, 2015.

Office Action for U.S. Appl. No. 13/669,357, filed Nov. 5, 2012, dated Jan. 30, 2015.

(56)

References Cited

OTHER PUBLICATIONS

Office Action for U.S. Appl. No. 13/851,026, filed Mar. 26, 2013, dated Jan. 30, 2015.

Office Action for U.S. Appl. No. 13/786,328, filed Mar. 5, 2013, dated Mar. 13, 2015.

Office Action for U.S. Appl. No. 13/092,460, filed Apr. 22, 2011, dated Mar. 13, 2015.

Office Action for U.S. Appl. No. 13/425,238, dated Mar. 12, 2015.

Office Action for U.S. Appl. No. 13/092,752, filed Apr. 22, 2011, dated Feb. 27, 2015.

Office Action for U.S. Appl. No. 13/042,259, filed Mar. 7, 2011, dated Feb. 23, 2015.

Office Action for U.S. Appl. No. 13/044,301, filed Mar. 9, 2011, dated Jan. 29, 2015.

Office Action for U.S. Appl. No. 13/050,102, filed Mar. 17, 2011, dated Jan. 26, 2015.

Office action dated Oct. 2, 2014, for U.S. Appl. No. 13/092,752, filed Apr. 22, 2011.

Kompella, Ed K. et al., 'Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling' Jan. 2007.

Rosen, E. et al., "BGP/MPLS VPNs", Mar. 1999.

* cited by examiner

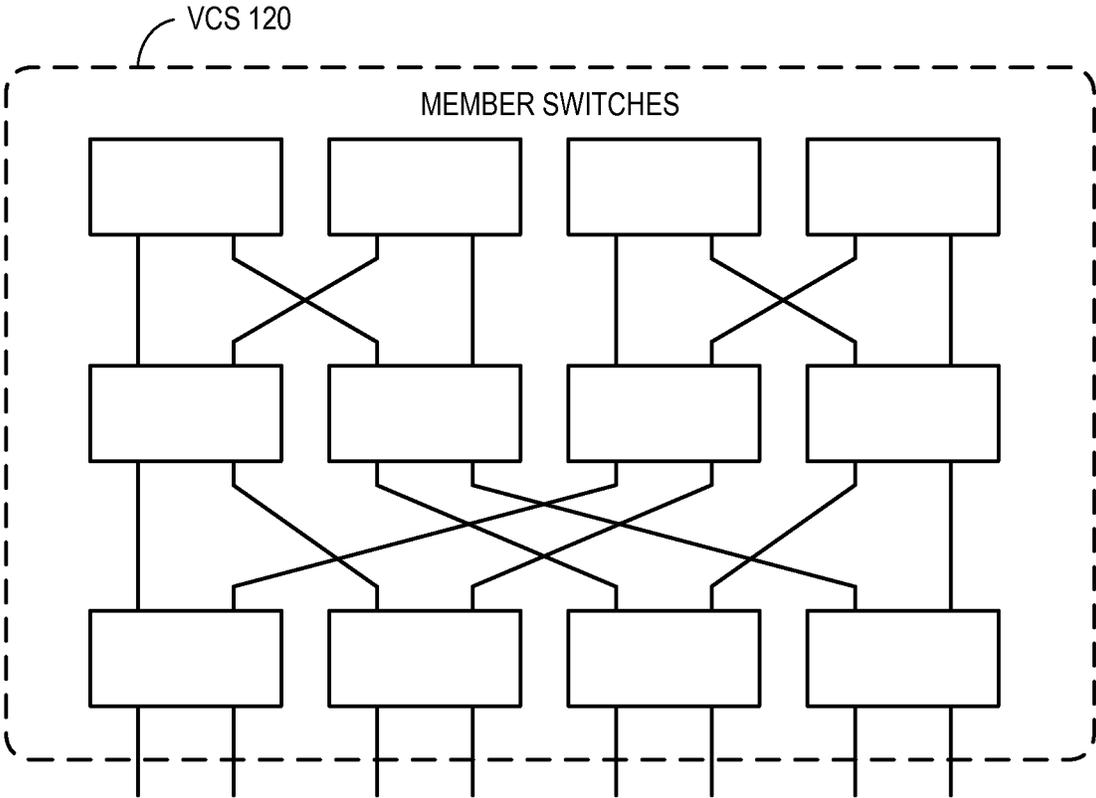


FIG. 1B

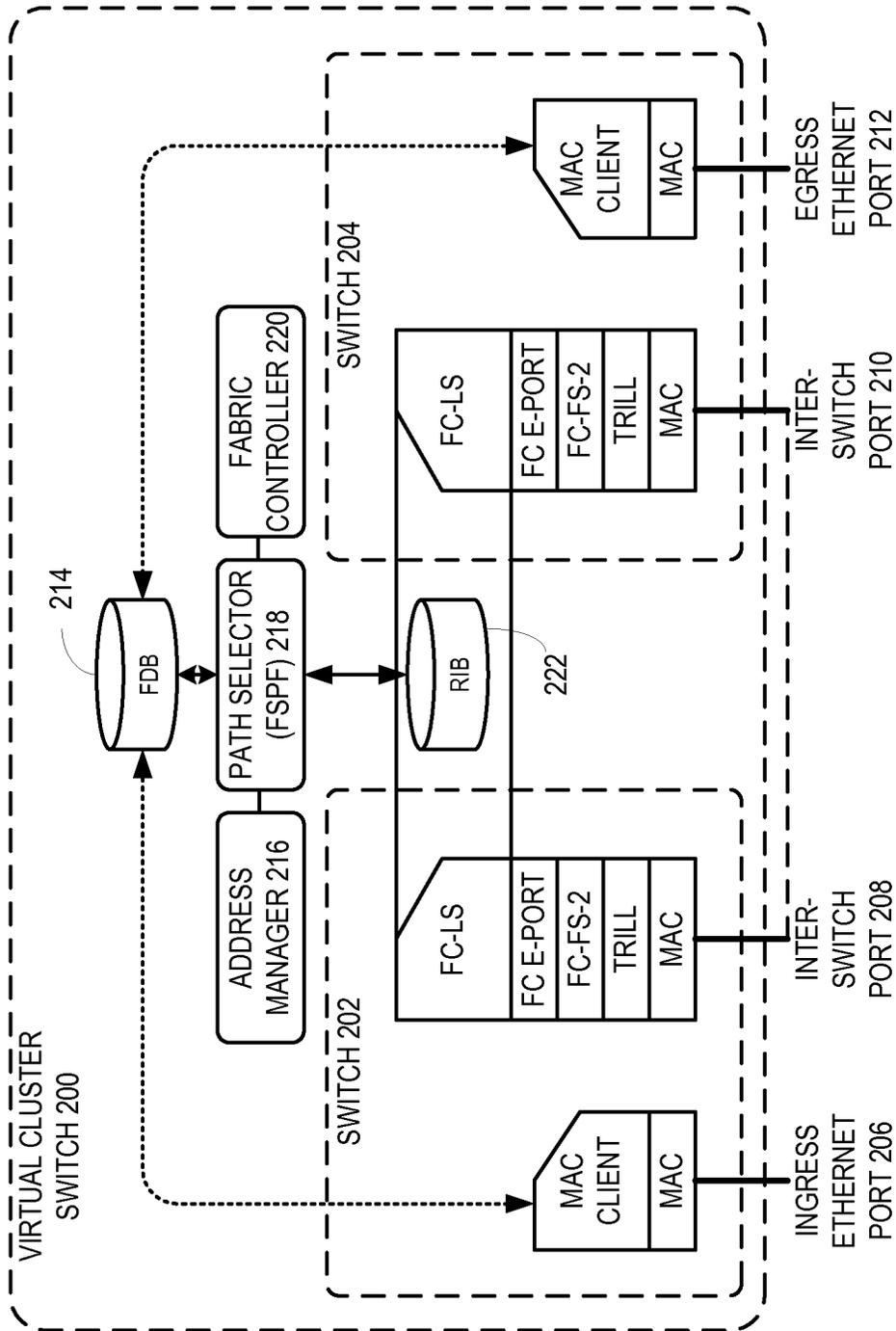


FIG. 2

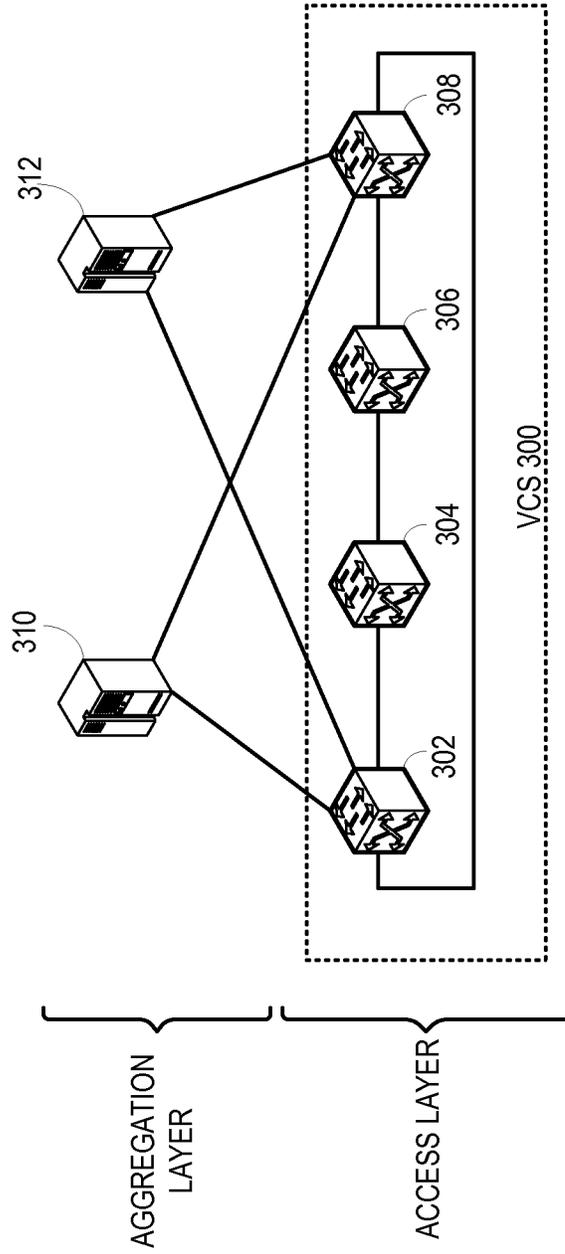


FIG. 3

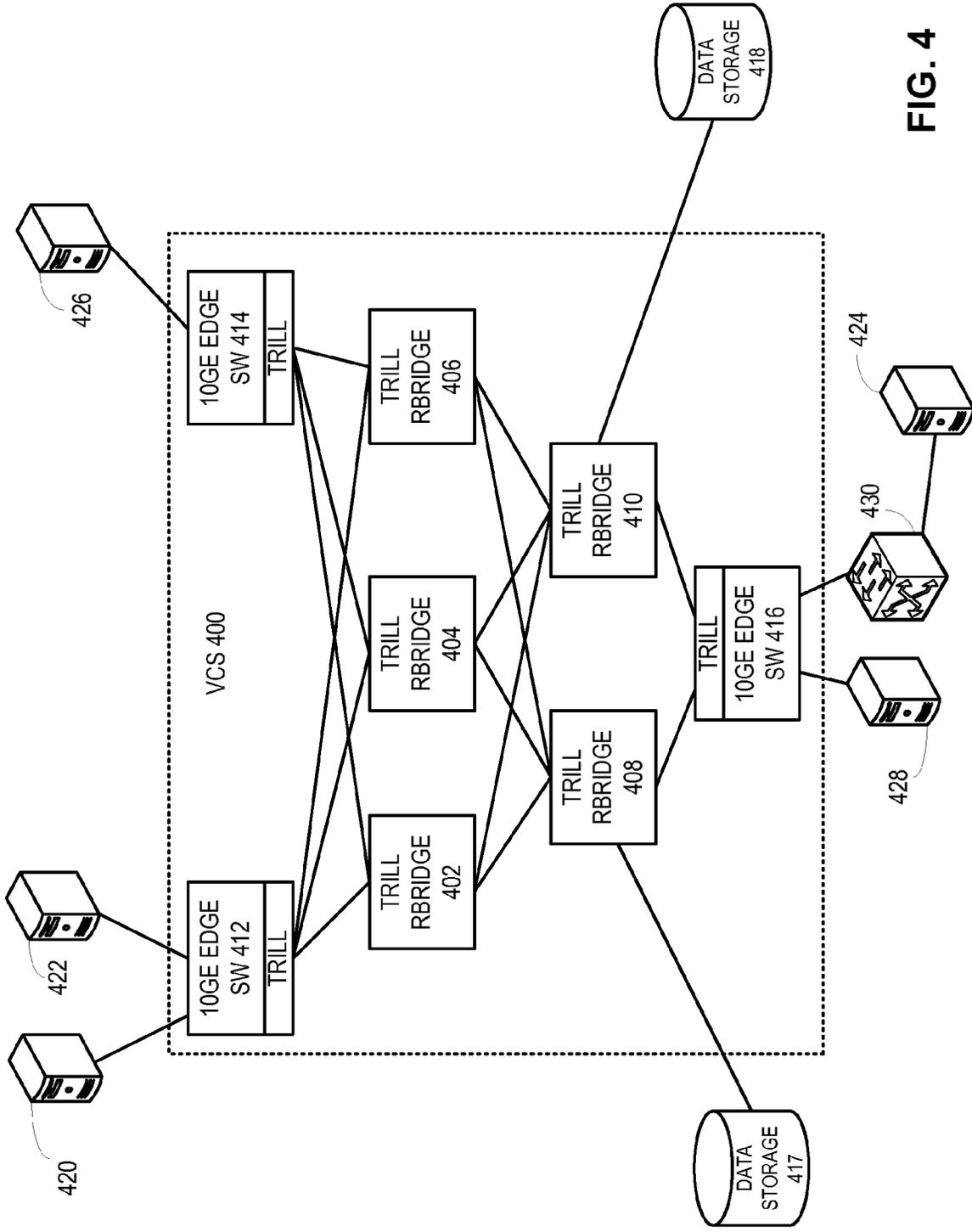


FIG. 4

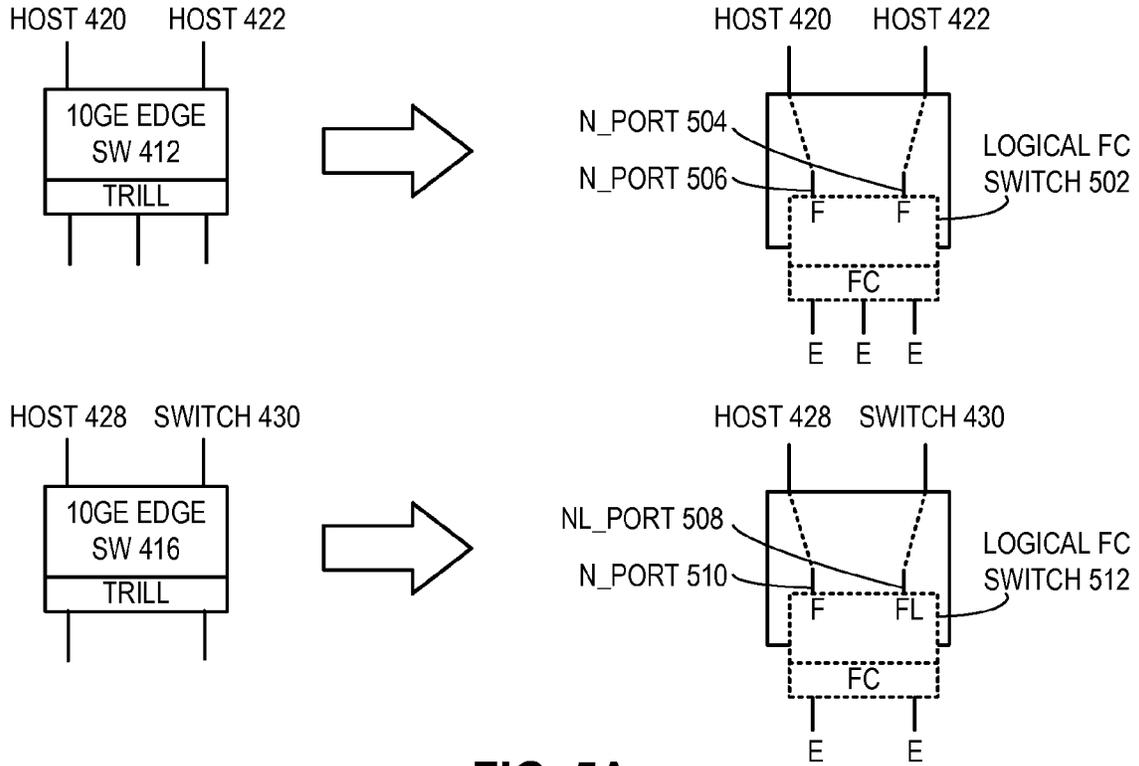


FIG. 5A

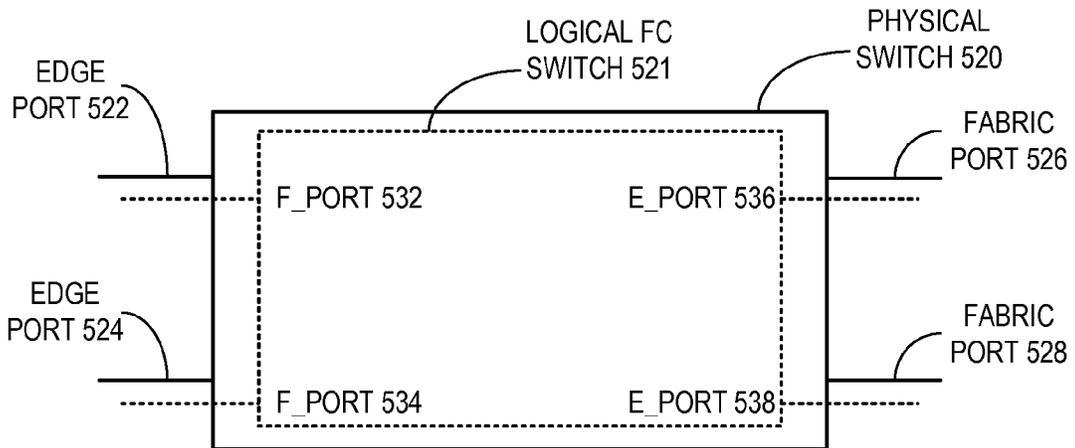


FIG. 5B

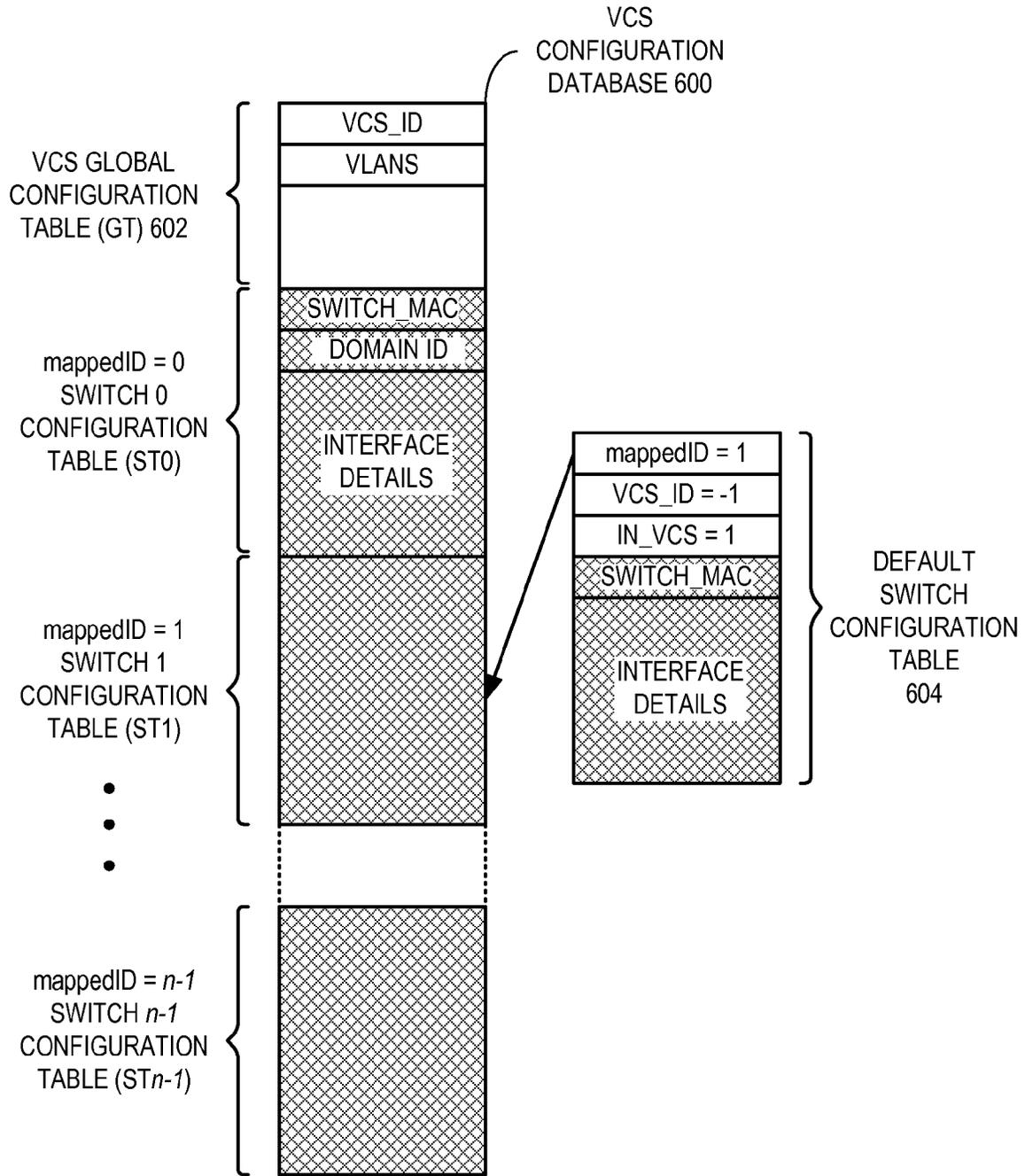


FIG. 6

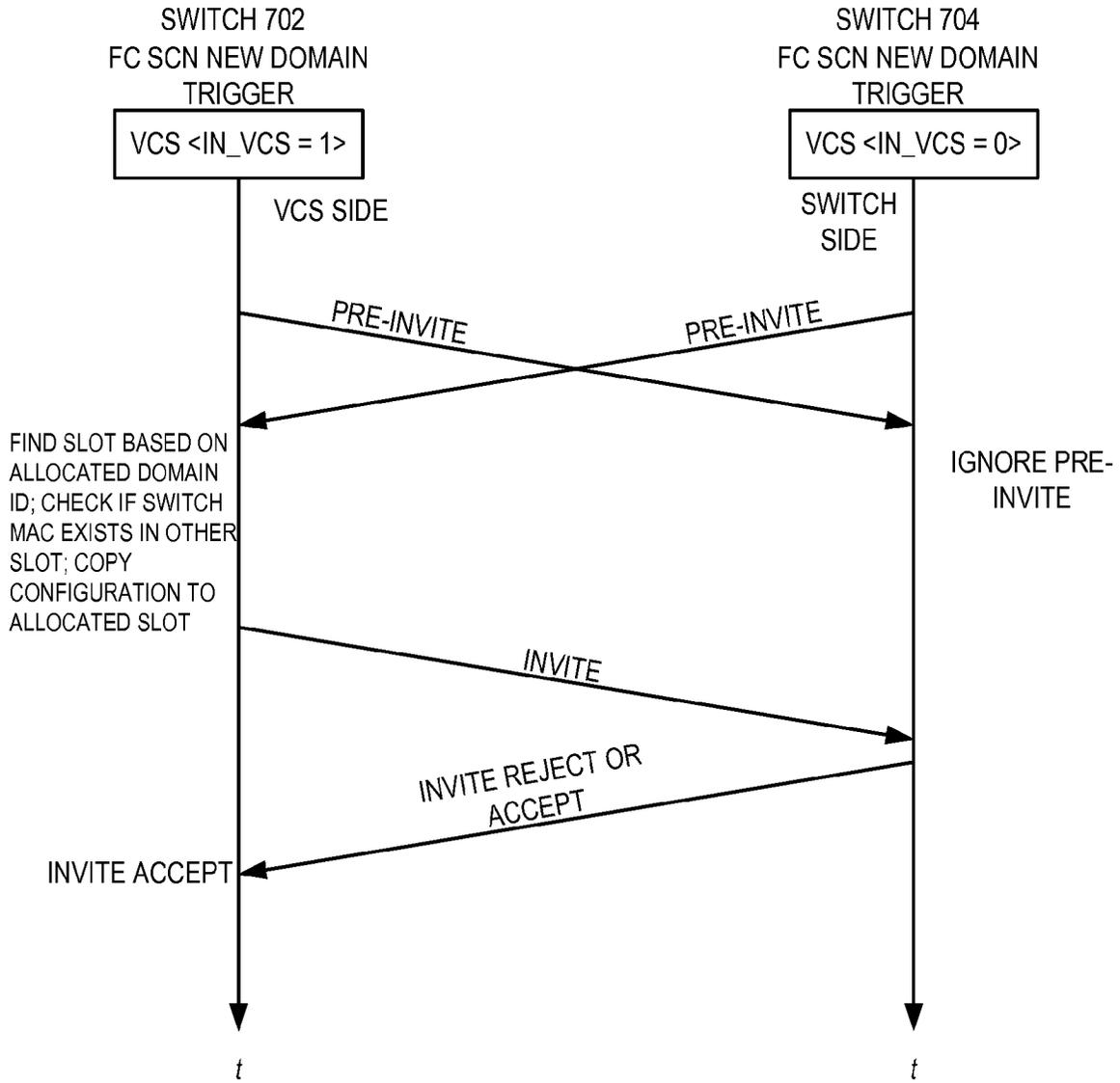


FIG. 7

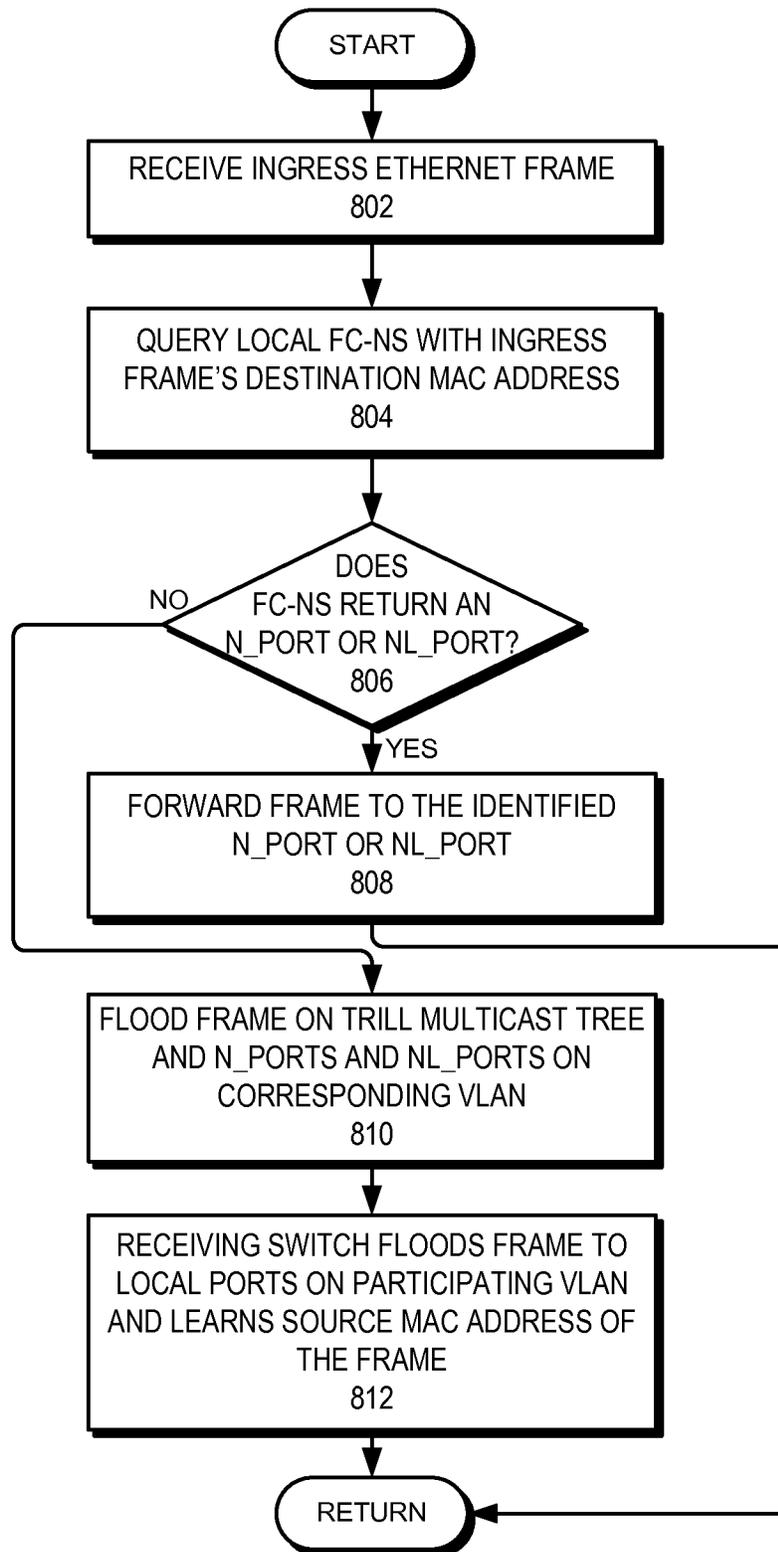


FIG. 8

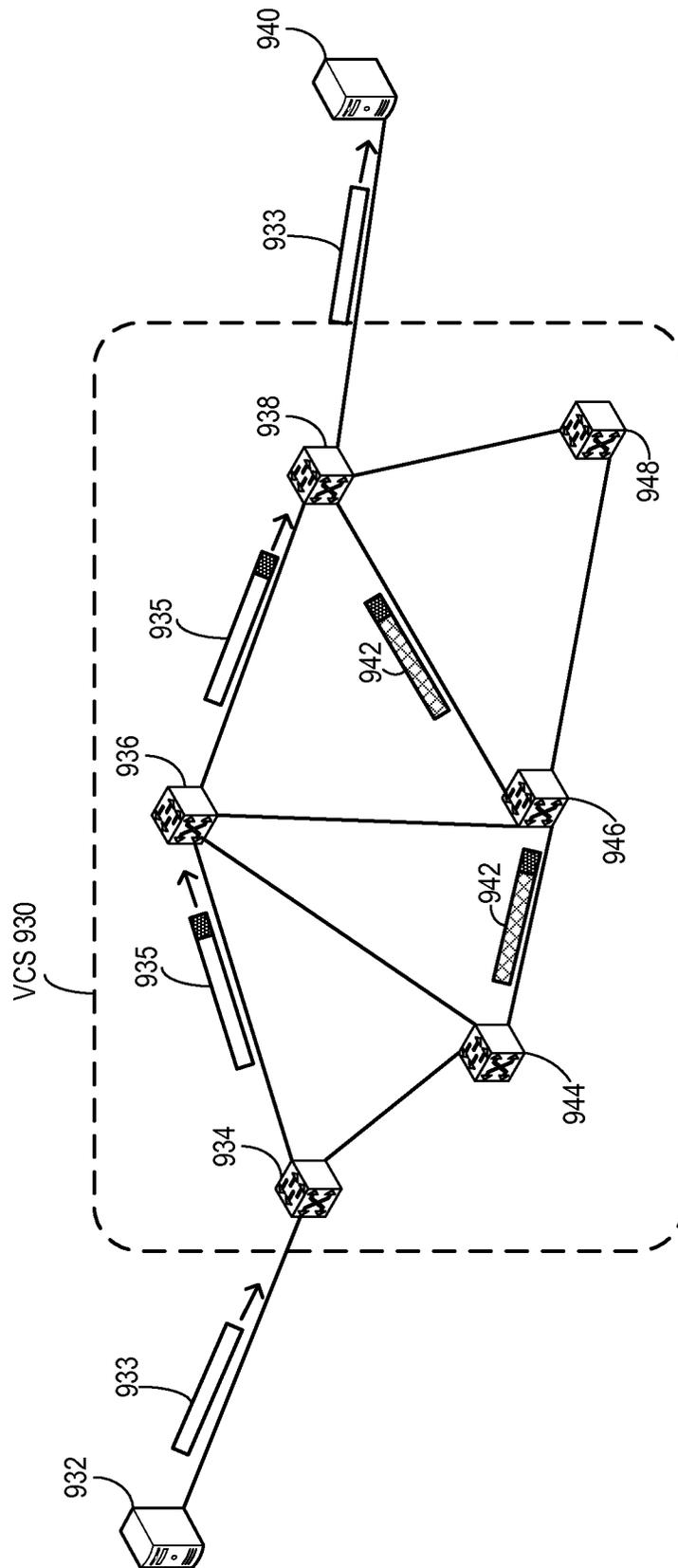


FIG. 9

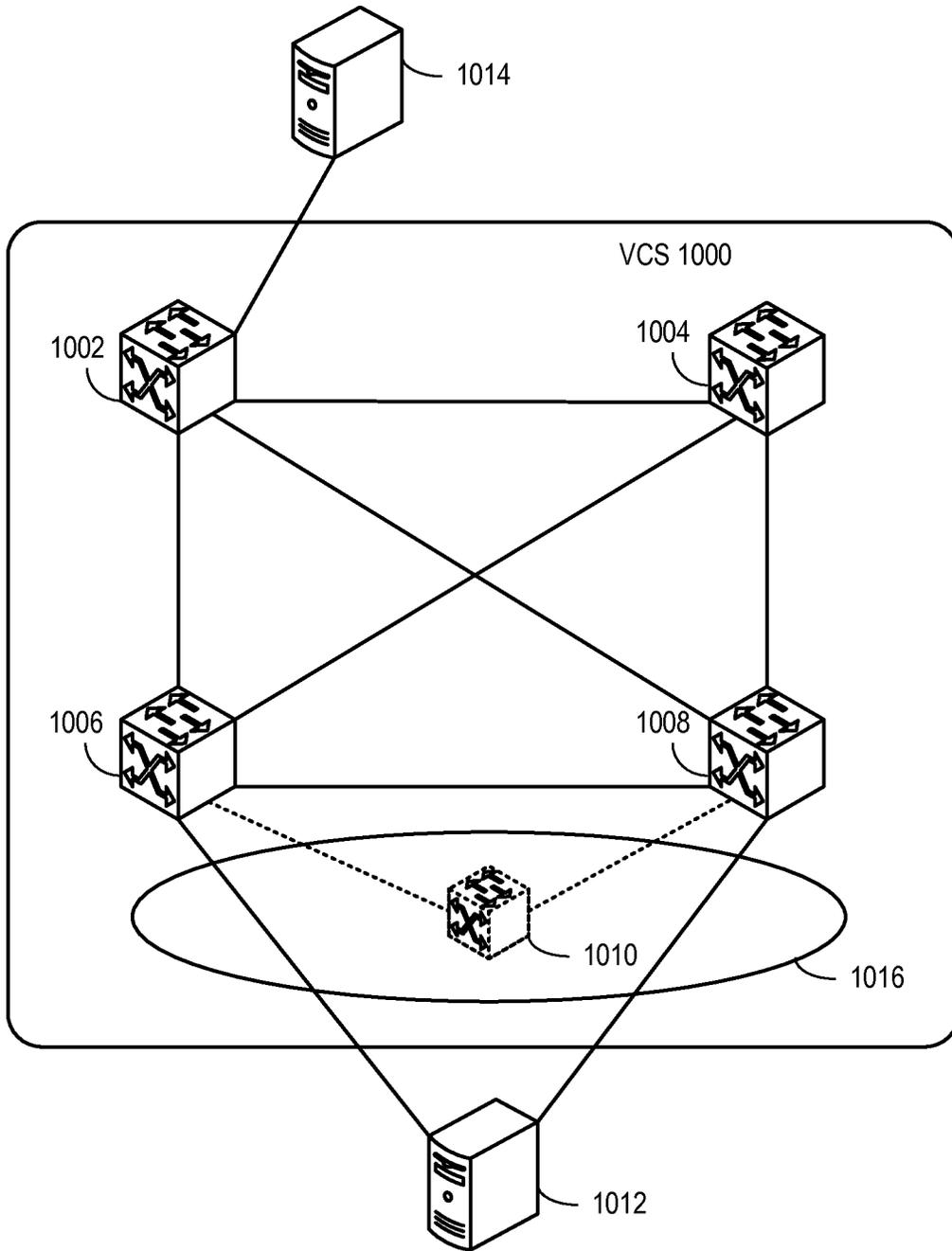


FIG. 10

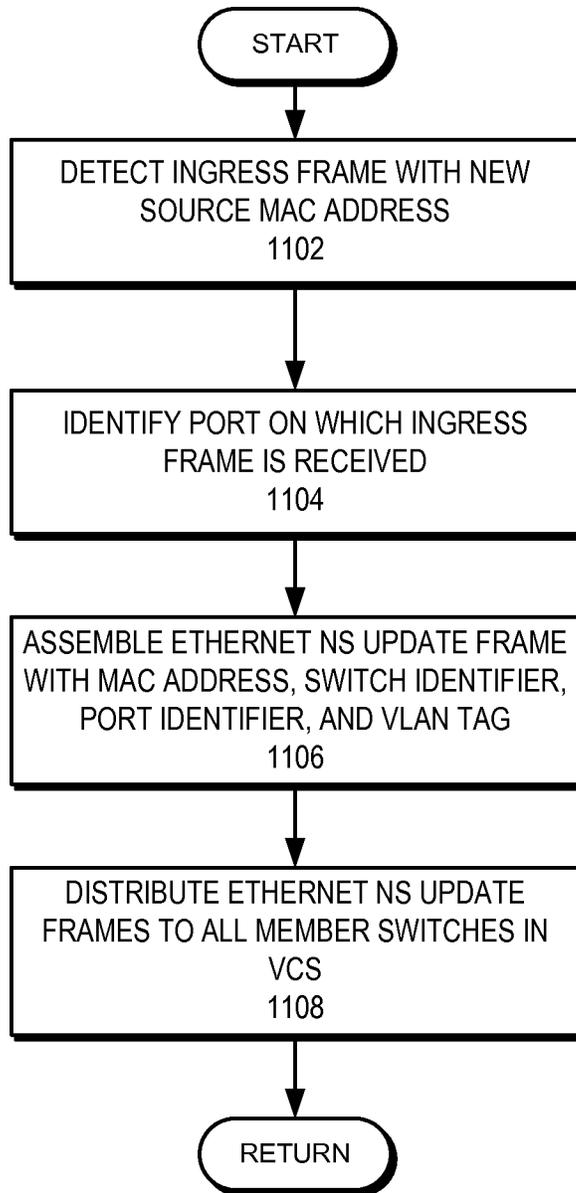


FIG. 11

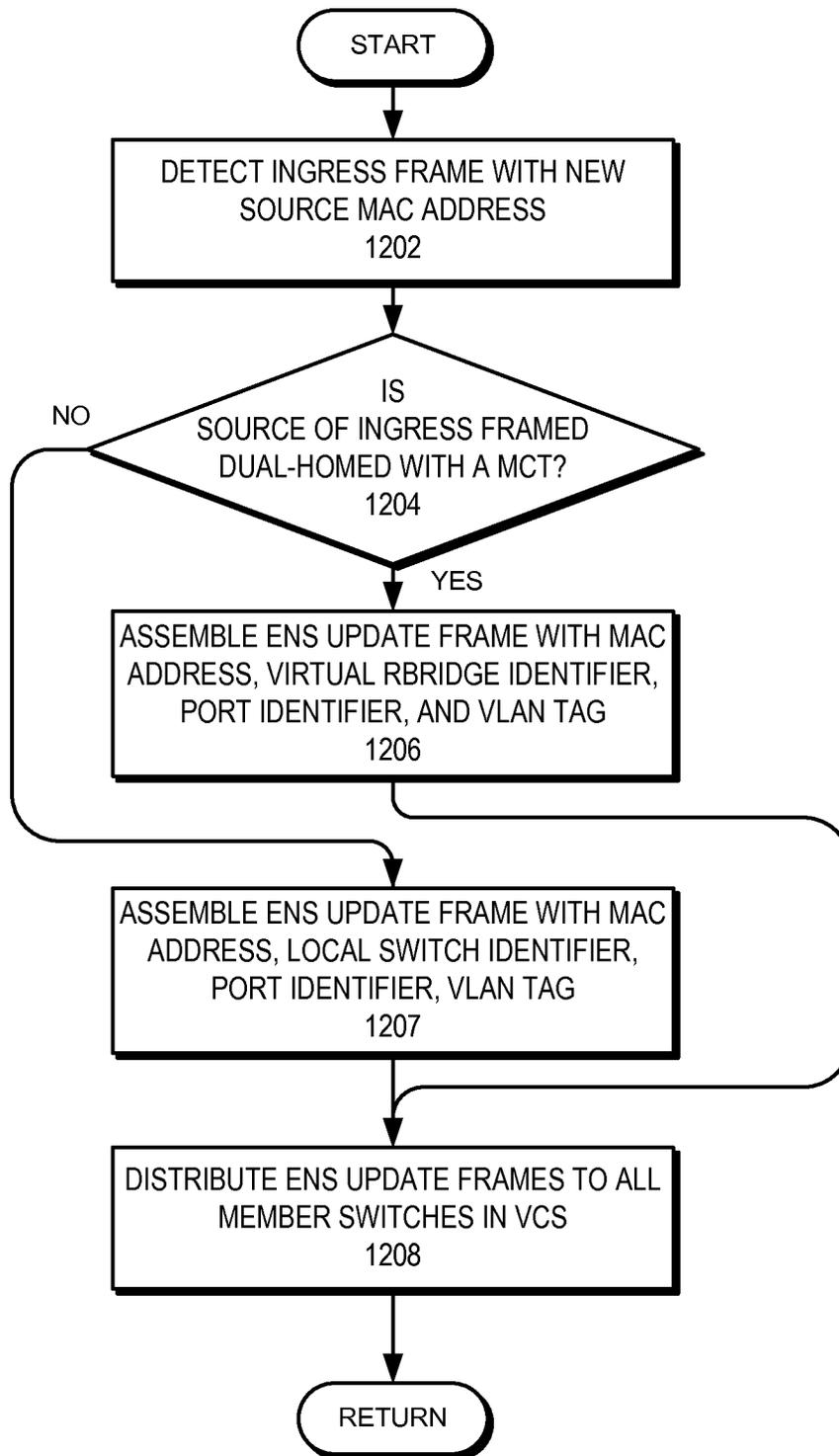


FIG. 12

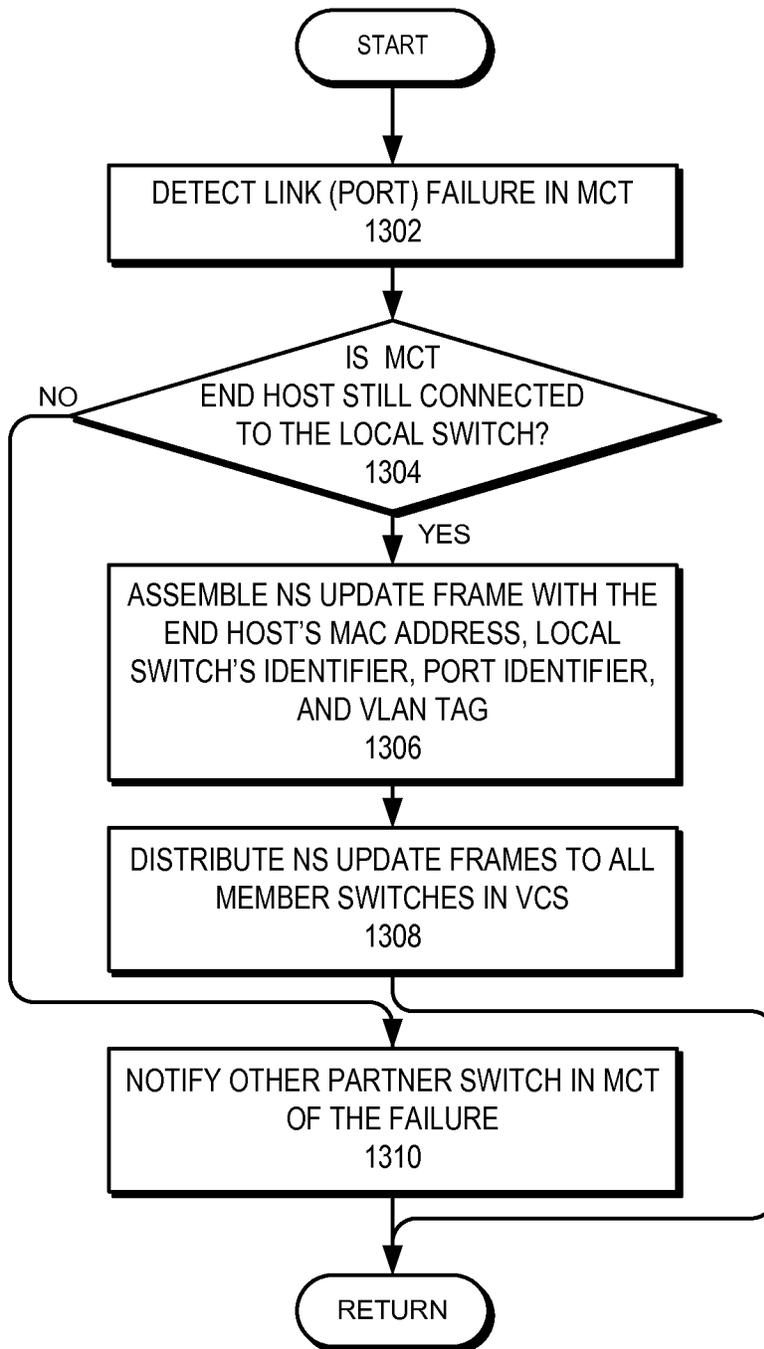


FIG. 13

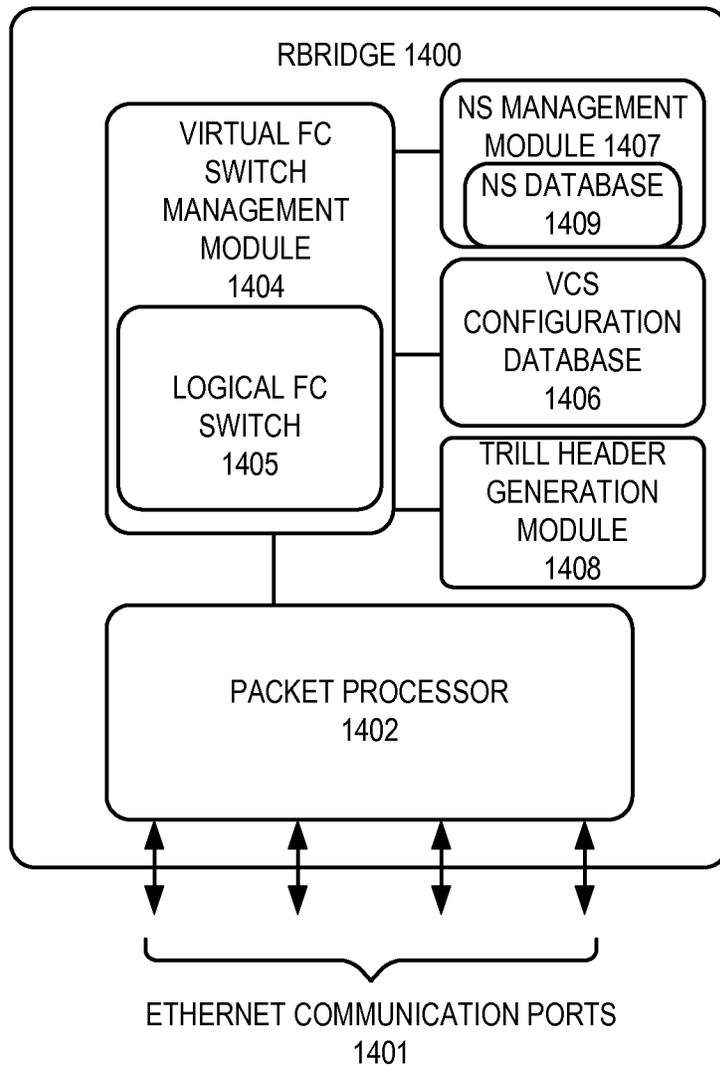


FIG. 14

NAME SERVICES FOR VIRTUAL CLUSTER SWITCHING

RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Application No. 61/352,264, entitled "Name Services for Virtual Cluster Switching," by inventors Suresh Vobbilisetty, Phanidhar Koganti, and Jesse B. Willeke, filed 07 Jun. 2010, and U.S. Provisional Application No. 61/380,803, entitled "Name Services for Virtual Cluster Switching," by inventors Suresh Vobbilisetty, Phanidhar Koganti, and Jesse B. Willeke, filed 08 Sep. 2010, the disclosure of which is incorporated by reference herein.

The present disclosure is related to U.S. patent application Ser. No. 12/725,249, entitled "REDUNDANT HOST CONNECTION IN A ROUTED NETWORK," by inventors Somesh Gupta, Anoop Ghanwani, Phanidhar Koganti, and Shunjia Yu, filed 16 Mar. 2010; and

U.S. patent application Ser. No. 13/087,239, entitled "VIRTUAL CLUSTER SWITCHING," by inventors Suresh Vobbilisetty and Dilip Chatwani, filed 14 Apr. 2011;

the disclosures of which are incorporated by reference herein.

BACKGROUND

1. Field

The present disclosure relates to network design. More specifically, the present disclosure relates to a method for a constructing a scalable switching system that facilitates automatic configuration.

2. Related Art

The relentless growth of the Internet has brought with it an insatiable demand for bandwidth. As a result, equipment vendors race to build larger, faster, and more versatile switches to move traffic. However, the size of a switch cannot grow infinitely. It is limited by physical space, power consumption, and design complexity, to name a few factors. More importantly, because an overly large system often does not provide economy of scale due to its complexity, simply increasing the size and throughput of a switch may prove economically unviable due to the increased per-port cost.

One way to increase the throughput of a switch system is to use switch stacking. In switch stacking, multiple smaller-scale, identical switches are interconnected in a special pattern to form a larger logical switch. However, switch stacking requires careful configuration of the ports and inter-switch links. The amount of required manual configuration becomes prohibitively complex and tedious when the stack reaches a certain size, which precludes switch stacking from being a practical option in building a large-scale switching system. Furthermore, a system based on stacked switches often has topology limitations which restrict the scalability of the system due to fabric bandwidth considerations.

In addition, the evolution of virtual computing has placed additional requirements on the network. For example, as the locations of virtual servers become more mobile and dynamic, it is often important for the network to update its knowledge of the location of these virtual servers quickly.

SUMMARY

One embodiment of the present invention provides a switch that facilitates name services in a virtual cluster switch. The switch includes a name service database indicating at least one media access control (MAC) address learned

at a second switch. The switch also includes a control mechanism. During operation, the control mechanism distributes information on a locally learned MAC address to the second switch. In addition, the control mechanism receives information on a MAC address learned at the second switch.

In a variation on this embodiment, the switch and the second switch are members of a virtual cluster switch comprising one or more physical switches which are allowed to be coupled in an arbitrary topology. Furthermore, the virtual cluster switch appears to be one single switch.

In a variation on this embodiment, while distributing information to the second switch, the control mechanism constructs a Fibre Channel registered state change notification (RSCN) encapsulated in a transparent interconnection of lots of links (TRILL) header.

In a variation on this embodiment, the distributed information to the second switch includes the MAC address and an identifier of the switch.

In a further variation, the distributed information further includes an identifier of a port to which a host corresponding to the MAC address is coupled and a virtual local area network (VLAN) tag associated with the MAC address.

In a variation on this embodiment, the control mechanism further sends an update to second switch when a link or port within a multi-chassis trunk fails.

In a further variation, the update indicates that an end host previously connected via a multi-chassis trunk is now connected with a physical switch.

BRIEF DESCRIPTION OF THE FIGURES

FIG. 1A illustrates an exemplary virtual cluster switch (VCS) system, in accordance with an embodiment of the present invention.

FIG. 1B illustrates an exemplary VCS system where the member switches are configured in a CLOS network, in accordance with an embodiment of the present invention.

FIG. 2 illustrates the protocol stack within a virtual cluster switch, in accordance with an embodiment of the present invention.

FIG. 3 illustrates an exemplary configuration of a virtual cluster switch, in accordance with an embodiment of the present invention.

FIG. 4 illustrates an exemplary configuration of how a virtual cluster switch can be connected to different edge networks, in accordance with an embodiment of the present invention.

FIG. 5A illustrates how a logical Fibre Channel switch fabric is formed in a virtual cluster switch in conjunction with the example in FIG. 4, in accordance with an embodiment of the present invention.

FIG. 5B illustrates an example of how a logical FC switch can be created within a physical Ethernet switch, in accordance with one embodiment of the present invention.

FIG. 6 illustrates an exemplary VCS configuration database, in accordance with an embodiment of the present invention.

FIG. 7 illustrates an exemplary process of a switch joining a virtual cluster switch, in accordance with an embodiment of the present invention.

FIG. 8 presents a flowchart illustrating the process of looking up an ingress frame's destination MAC address and forwarding the frame in a VCS, in accordance with one embodiment of the present invention.

FIG. 9 illustrates how data frames and control frames are transported through a VCS, in accordance with one embodiment of the present invention.

FIG. 10 illustrates an example of name service operation in a VCS, in accordance with one embodiment of the present invention.

FIG. 11 presents a flowchart illustrating the process of distributing learned MAC information by the Ethernet name service in a VCS, in accordance with one embodiment of the present invention.

FIG. 12 presents a flowchart illustrating the process of distributing information of a learned MAC address via an MCT, in accordance with one embodiment of the present invention.

FIG. 13 presents a flowchart illustrating the process of updating the link state in an MCT group, in accordance with one embodiment of the present invention.

FIG. 14 illustrates an exemplary switch that facilitates formation of a virtual cluster switch with Ethernet and MCT name services, in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not limited to the embodiments shown, but is to be accorded the widest scope consistent with the claims.

Overview

In embodiments of the present invention, the problem of facilitating fast distribution of the location information of each end host is solved by providing a distributed name service throughout a virtual cluster switch. The virtual cluster switch allows a number of switches to be inter-connected to form a single, scalable logical switch without requiring burdensome manual configuration. As a result, one can form a large-scale logical switch (referred to as a “virtual cluster switch” or VCS herein) using a number of smaller physical switches. The automatic configuration capability provided by the control plane running on each physical switch allows any number of switches to be connected in an arbitrary topology without requiring tedious manual configuration of the ports and links. This feature makes it possible to use many smaller, inexpensive switches to construct a large cluster switch, which can be viewed as a single logical switch externally.

In a VCS, each member switch performs source media access control (MAC) address learning. Once a new MAC address is observed and learned from a port, the corresponding MAC address and switch/port identifier information is distributed throughout the VCS. In this way, each VCS member switch can maintain a complete set of knowledge of the location of all the end-hosts (including virtual machines). This knowledge allows a frame destined to any MAC address to be properly routed to the correct switch, even if the MAC address is not directly learned by the local switch. (Note that, in a conventional Ethernet switch, a frame with an unknown destination MAC address is flooded to all the ports.) In this disclosure, the description in conjunction with FIGS. 1-9 is associated with the general architecture of VCS, and the description in conjunction with FIG. 10 and onward provide more details on the advanced link tracking mechanism.

It should be noted that a virtual cluster switch is not the same as conventional switch stacking. In switch stacking, multiple switches are interconnected at a common location

(often within the same rack), based on a particular topology, and manually configured in a particular way. These stacked switches typically share a common address, e.g., IP address, so they can be addressed as a single switch externally. Furthermore, switch stacking requires a significant amount of manual configuration of the ports and inter-switch links. The need for manual configuration prohibits switch stacking from being a viable option in building a large-scale switching system. The topology restriction imposed by switch stacking also limits the number of switches that can be stacked. This is because it is very difficult, if not impossible, to design a stack topology that allows the overall switch bandwidth to scale adequately with the number of switch units.

In contrast, a VCS can include an arbitrary number of switches with individual addresses, can be based on an arbitrary topology, and does not require extensive manual configuration. The switches can reside in the same location, or be distributed over different locations. These features overcome the inherent limitations of switch stacking and make it possible to build a large “switch farm” which can be treated as a single, logical switch. Due to the automatic configuration capabilities of the VCS, an individual physical switch can dynamically join or leave the VCS without disrupting services to the rest of the network.

Furthermore, the automatic and dynamic configurability of VCS allows a network operator to build its switching system in a distributed and “pay-as-you-grow” fashion without sacrificing scalability. The VCS’s ability to respond to changing network conditions makes it an ideal solution in a virtual computing environment, where network loads often change with time.

Although this disclosure is presented using examples based on the Transparent Interconnection of Lots of Links (TRILL) as the transport protocol and the Fibre Channel (FC) fabric protocol as the control-plane protocol, embodiments of the present invention are not limited to TRILL networks, or networks defined in a particular Open System Interconnection Reference Model (OSI reference model) layer. For example, a VCS can also be implemented with switches running multi-protocol label switching (MPLS) protocols for the transport. In addition, the terms “RBridge” and “switch” are used interchangeably in this disclosure. The use of the term “RBridge” does not limit embodiments of the present invention to TRILL networks only. The TRILL protocol is described in IETF draft “RBridges: Base Protocol Specification,” available at <http://tools.ietf.org/html/draft-ietf-trill-rbridge-protocol>, which is incorporated by reference herein.

The terms “virtual cluster switch,” “virtual cluster switching,” and “VCS” refer to a group of interconnected physical switches operating as a single logical switch. The control plane for these physical switches provides the ability to automatically configure a given physical switch, so that when it joins the VCS, little or no manual configuration is required.

The term “RBridge” refers to routing bridges, which are bridges implementing the TRILL protocol as described in IETF draft “RBridges: Base Protocol Specification.” Embodiments of the present invention are not limited to the application among RBridges. Other types of switches, routers, and forwarders can also be used.

The terms “frame” or “packet” refer to a group of bits that can be transported together across a network. “Frame” should not be interpreted as limiting embodiments of the present invention to layer-2 networks. “Packet” should not be interpreted as limiting embodiments of the present invention to layer-3 networks. “Frame” or “packet” can be replaced by other terminologies referring to a group of bits, such as “cell” or “datagram.”

VCS Architecture

FIG. 1A illustrates an exemplary virtual cluster switch system, in accordance with an embodiment of the present invention. In this example, a VCS **100** includes physical switches **101**, **102**, **103**, **104**, **105**, **106**, and **107**. A given physical switch runs an Ethernet-based transport protocol on its ports (e.g., TRILL on its inter-switch ports, and Ethernet transport on its external ports), while its control plane runs an FC switch fabric protocol stack. The TRILL protocol facilitates transport of Ethernet frames within and across VCS **100** in a routed fashion (since TRILL provides routing functions to Ethernet frames). The FC switch fabric protocol stack facilitates the automatic configuration of individual physical switches, in a way similar to how a conventional FC switch fabric is formed and automatically configured. In one embodiment, VCS **100** can appear externally as an ultra-high-capacity Ethernet switch. More details on FC network architecture, protocols, naming/address conventions, and various standards are available in the documentation available from the NCITS/ANSI T11 committee (www.t11.org) and publicly available literature, such as "Designing Storage Area Networks," by Tom Clark, 2nd Ed., Addison Wesley, 2003, the disclosures of which are incorporated by reference in their entirety herein.

A physical switch may dedicate a number of ports for external use (i.e., to be coupled to end hosts or other switches external to the VCS) and other ports for inter-switch connection. Viewed externally, VCS **100** appears to be one switch to a device from the outside, and any port from any of the physical switches is considered one port on the VCS. For example, port groups **110** and **112** are both VCS external ports and can be treated equally as if they were ports on a common physical switch, although switches **105** and **107** may reside in two different locations.

The physical switches can reside at a common location, such as a data center or central office, or be distributed in different locations. Hence, it is possible to construct a large-scale centralized switching system using many smaller, inexpensive switches housed in one or more chassis at the same location. It is also possible to have the physical switches placed at different locations, thus creating a logical switch that can be accessed from multiple locations. The topology used to interconnect the physical switches can also be versatile. VCS **100** is based on a mesh topology. In further embodiments, a VCS can be based on a ring, fat tree, or other types of topologies.

In one embodiment, the protocol architecture of a VCS is based on elements from the standard IEEE 802.1Q Ethernet bridge, which is emulated over a transport based on the Fibre Channel Framing and Signaling-2 (FC-FS-2) standard. The resulting switch is capable of transparently switching frames from an ingress Ethernet port from one of the edge switches to an egress Ethernet port on a different edge switch through the VCS.

Because of its automatic configuration capability, a VCS can be dynamically expanded as the network demand increases. In addition, one can build a large-scale switch using many smaller physical switches without the burden of manual configuration. For example, it is possible to build a high-throughput fully non-blocking switch using a number of smaller switches. This ability to use small switches to build a large non-blocking switch significantly reduces the cost associated with switch complexity. FIG. 1B presents an exemplary VCS with its member switches connected in a CLOS network, in accordance with one embodiment of the present invention. In this example, a VCS **120** forms a fully non-blocking 8x8 switch, using eight 4x4 switches and four 2x2

switches connected in a three-stage CLOS network. A large-scale switch with a higher port count can be built in a similar way.

FIG. 2 illustrates the protocol stack within a virtual cluster switch, in accordance with an embodiment of the present invention. In this example, two physical switches **202** and **204** are illustrated within a VCS **200**. Switch **202** includes an ingress Ethernet port **206** and an inter-switch port **208**. Switch **204** includes an egress Ethernet port **212** and an inter-switch port **210**. Ingress Ethernet port **206** receives Ethernet frames from an external device. The Ethernet header is processed by a media access control (MAC) layer protocol. On top of the MAC layer is a MAC client layer, which hands off the information extracted from the frame's Ethernet header to a forwarding database (FDB) **214**. Typically, in a conventional IEEE 802.1Q Ethernet switch, FDB **214** is maintained locally in a switch, which would perform a lookup based on the destination MAC address and the VLAN indicated in the Ethernet frame. The lookup result would provide the corresponding output port. However, since VCS **200** is not one single physical switch, FDB **214** would return the egress switch's identifier (i.e., switch **204**'s identifier). In one embodiment, FDB **214** is a data structure replicated and distributed among all the physical switches. That is, every physical switch maintains its own copy of FDB **214**. When a given physical switch learns the source MAC address and VLAN of an Ethernet frame (similar to what a conventional IEEE 802.1Q Ethernet switch does) as being reachable via the ingress port, the learned MAC and VLAN information, together with the ingress Ethernet port and switch information, is propagated to all the physical switches so every physical switch's copy of FDB **214** can remain synchronized. This prevents forwarding based on stale or incorrect information when there are changes to the connectivity of end stations or edge networks to the VCS.

The forwarding of the Ethernet frame between ingress switch **202** and egress switch **204** is performed via inter-switch ports **208** and **210**. The frame transported between the two inter-switch ports is encapsulated in an outer MAC header and a TRILL header, in accordance with the TRILL standard. The protocol stack associated with a given inter-switch port includes the following (from bottom up): MAC layer, TRILL layer, FC-FS-2 layer, FC E-Port layer, and FC link services (FC-LS) layer. The FC-LS layer is responsible for maintaining the connectivity information of a physical switch's neighbor, and populating an FC routing information base (RIB) **222**. This operation is similar to what is done in an FC switch fabric. The FC-LS protocol is also responsible for handling joining and departure of a physical switch in VCS **200**. The operation of the FC-LS layer is specified in the FC-LS standard, which is available at <http://www.t11.org/ftp/t11/member/fc/ls/06-393v5.pdf>, the disclosure of which is incorporated herein in its entirety.

During operation, when FDB **214** returns the egress switch **204** corresponding to the destination MAC address of the ingress Ethernet frame, the destination egress switch's identifier is passed to a path selector **218**. Path selector **218** performs a fabric shortest-path first (FSPF)-based route lookup in conjunction with RIB **222**, and identifies the next-hop switch within VCS **200**. In other words, the routing is performed by the FC portion of the protocol stack, similar to what is done in an FC switch fabric.

Also included in each physical switch are an address manager **216** and a fabric controller **220**. Address manager **216** is responsible for configuring the address of a physical switch when the switch first joins the VCS. For example, when switch **202** first joins VCS **200**, address manager **216** can

negotiate a new FC switch domain ID, which is subsequently used to identify the switch within VCS **200**. Fabric controller **220** is responsible for managing and configuring the logical FC switch fabric formed on the control plane of VCS **200**.

One way to understand the protocol architecture of VCS is to view the VCS as an FC switch fabric with an Ethernet/TRILL transport. Each physical switch, from an external point of view, appears to be a TRILL RBridge. However, the switch's control plane implements the FC switch fabric software. In other words, embodiments of the present invention facilitate the construction of an "Ethernet switch fabric" running on FC control software. This unique combination provides the VCS with automatic configuration capability and allows it to provide the ubiquitous Ethernet services in a very scalable fashion.

FIG. 3 illustrates an exemplary configuration of a virtual cluster switch, in accordance with an embodiment of the present invention. In this example, a VCS **300** includes four physical switches **302**, **304**, **306**, and **308**. VCS **300** constitutes an access layer which is coupled to two aggregation switches **310** and **312**. Note that the physical switches within VCS **300** are connected in a ring topology. Aggregation switch **310** or **312** can connect to any of the physical switches within VCS **300**. For example, aggregation switch **310** is coupled to physical switches **302** and **308**. These two links are viewed as a trunked link to VCS **300**, since the corresponding ports on switches **302** and **308** are considered to be from the same logical switch, VCS **300**. Note that, without VCS, such topology would not have been possible, because the FDB needs to remain synchronized, which is facilitated by the VCS.

FIG. 4 illustrates an exemplary configuration of how a virtual cluster switch can be connected to different edge networks, in accordance with an embodiment of the present invention. In this example, a VCS **400** includes a number of TRILL R Bridges **402**, **404**, **406**, **408**, and **410**, which are controlled by the FC switch-fabric control plane. Also included in VCS **400** are R Bridges **412**, **414**, and **416**. Each R Bridge has a number of edge ports which can be connected to external edge networks.

For example, R Bridge **412** is coupled with hosts **420** and **422** via 10GE ports. R Bridge **414** is coupled to a host **426** via a 10GE port. These R Bridges have TRILL-based inter-switch ports for connection with other TRILL R Bridges in VCS **400**. Similarly, R Bridge **416** is coupled to host **428** and an external Ethernet switch **430**, which is coupled to an external network that includes a host **424**. In addition, network equipment can also be coupled directly to any of the physical switches in VCS **400**. As illustrated here, TRILL R Bridge **408** is coupled to a data storage **417**, and TRILL R Bridge **410** is coupled to a data storage **418**.

Although the physical switches within VCS **400** are labeled as "TRILL R Bridges," they are different from the conventional TRILL R Bridge in the sense that they are controlled by the FC switch fabric control plane. In other words, the assignment of switch addresses, link discovery and maintenance, topology convergence, routing, and forwarding can be handled by the corresponding FC protocols. Particularly, each TRILL R Bridge's switch ID or nickname is mapped from the corresponding FC switch domain ID, which can be automatically assigned when a switch joins VCS **400** (which is logically similar to an FC switch fabric).

Note that TRILL is only used as a transport between the switches within VCS **400**. This is because TRILL can readily accommodate native Ethernet frames. Also, the TRILL standards provide a ready-to-use forwarding mechanism that can be used in any routed network with arbitrary topology (al-

though the actual routing in VCS is done by the FC switch fabric protocols). Embodiments of the present invention should be not limited to using only TRILL as the transport. Other protocols (such as multi-protocol label switching (MPLS) or Internet Protocol (IP)), either public or proprietary, can also be used for the transport.

VCS Formation

In one embodiment, a VCS is created by instantiating a logical FC switch in the control plane of each switch. After the logical FC switch is created, a virtual generic port (denoted as G_Port) is created for each Ethernet port on the R Bridge. A G_Port assumes the normal G_Port behavior from the FC switch perspective. However, in this case, since the physical links are based on Ethernet, the specific transition from a G_Port to either an FC F_Port or E_Port is determined by the underlying link and physical layer protocols. For example, if the physical Ethernet port is connected to an external device which lacks VCS capabilities, the corresponding G_Port will be turned into an F_Port. On the other hand, if the physical Ethernet port is connected to a switch with VCS capabilities and it is confirmed that the switch on the other side is part of a VCS, then the G_Port will be turned into an E_port.

FIG. 5A illustrates how a logical Fibre Channel switch fabric is formed in a virtual cluster switch in conjunction with the example in FIG. 4, in accordance with an embodiment of the present invention. R Bridge **412** contains a virtual, logical FC switch **502**. Corresponding to the physical Ethernet ports coupled to hosts **420** and **422**, logical FC switch **502** has two logical F_Ports, which are logically coupled to hosts **420** and **422**. In addition, two logical N_Ports, **506** and **504**, are created for hosts **420** and **422**, respectively. On the VCS side, logical FC switch **502** has three logical E_Ports, which are to be coupled with other logical FC switches in the logical FC switch fabric in the VCS.

Similarly, R Bridge **416** contains a virtual, logical FC switch **512**. Corresponding to the physical Ethernet ports coupled to host **428** and external switch **430**, logical FC switch **512** has a logical F_Port coupled to host **428**, and a logical FL_Port coupled to switch **430**. In addition, a logical N_Port **510** is created for host **428**, and a logical NL_Port **508** is created for switch **430**. Note that the logical FL_Port is created because that port is coupled to a switch (switch **430**), instead of a regular host, and therefore logical FC switch **512** assumes an arbitrated loop topology leading to switch **430**. Logical NL_Port **508** is created based on the same reasoning to represent a corresponding NL_Port on switch **430**. On the VCS side, logical FC switch **512** has two logical E_Ports, which to be coupled with other logical FC switches in the logical FC switch fabric in the VCS.

FIG. 5B illustrates an example of how a logical FC switch can be created within a physical Ethernet switch, in accordance with one embodiment of the present invention. The term "fabric port" refers to a port used to couple multiple switches in a VCS. The clustering protocols control the forwarding between fabric ports. The term "edge port" refers to a port that is not currently coupled to another switch unit in the VCS. Standard IEEE 802.1Q and layer-3 protocols control forwarding on edge ports.

In the example illustrated in FIG. 5B, a logical FC switch **521** is created within a physical switch (R Bridge) **520**. Logical FC switch **521** participates in the FC switch fabric protocol via logical inter-switch links (ISLs) to other switch units and has an FC switch domain ID assigned to it just as a physical FC switch does. In other words, the domain allocation, principal switch selection, and conflict resolution work just as they would on a physical FC ISL.

The physical edge ports **522** and **524** are mapped to logical F_Ports **532** and **534**, respectively. In addition, physical fabric ports **526** and **528** are mapped to logical E_Ports **536** and **538**, respectively. Initially, when logical FC switch **521** is created (for example, during the boot-up sequence), logical FC switch **521** only has four G_Ports which correspond to the four physical ports. These G_Ports are subsequently mapped to F_Ports or E_Ports, depending on the devices coupled to the physical ports.

Neighbor discovery is the first step in VCS formation between two VCS-capable switches. It is assumed that the verification of VCS capability can be carried out by a handshake process between two neighbor switches when the link is first brought up.

In general, a VCS presents itself as one unified switch composed of multiple member switches. Hence, the creation and configuration of VCS is of critical importance. The VCS configuration is based on a distributed database, which is replicated and distributed over all switches.

In one embodiment, a VCS configuration database includes a global configuration table (GT) of the VCS and a list of switch description tables (STs), each of which describes a VCS member switch. In its simplest form, a member switch can have a VCS configuration database that includes a global table and one switch description table, e.g., [**GT**][**ST**]. A VCS with multiple switches will have a configuration database that has a single global table and multiple switch description tables, e.g., [**GT**][**ST0**][**ST1**] . . . [**STn-1**]. The number *n* corresponds to the number of member switches in the VCS. In one embodiment, the GT can include at least the following information: the VCS ID, number of nodes in the VCS, a list of VLANs supported by the VCS, a list of all the switches (e.g., list of FC switch domain IDs for all active switches) in the VCS, and the FC switch domain ID of the principal switch (as in a logical FC switch fabric). A switch description table can include at least the following information: the IN_VCS flag, indication whether the switch is a principal switch in the logical FC switch fabric, the FC switch domain ID for the switch, the FC world-wide name (WWN) for the corresponding logical FC switch; the mapped ID of the switch, and optionally the IP address of the switch.

In addition, each switch's global configuration database is associated with a transaction ID. The transaction ID specifies the latest transaction (e.g., update or change) incurred to the global configuration database. The transaction IDs of the global configuration databases in two switches can be compared to determine which database has the most current information (i.e., the database with the more current transaction ID is more up-to-date). In one embodiment, the transaction ID is the switch's serial number plus a sequential transaction number. This configuration can unambiguously resolve which switch has the latest configuration.

As illustrated in FIG. 6, a VCS member switch typically maintains two configuration tables that describe its instance: a VCS configuration database **600**, and a default switch configuration table **604**. VCS configuration database **600** describes the VCS configuration when the switch is part of a VCS. Default switch configuration table **604** describes the switch's default configuration. VCS configuration database **600** includes a GT **602**, which includes a VCS identifier (denoted as VCS_ID) and a VLAN list within the VCS. Also included in VCS configuration database **600** are a number of STs, such as ST0, ST1, and STn. Each ST includes the corresponding member switch's MAC address and FC switch

domain ID, as well as the switch's interface details. Note that each switch also has a VCS-mapped ID which is a switch index within the VCS.

In one embodiment, each switch also has a VCS-mapped ID (denoted as "mappedID"), which is a switch index within the VCS. This mapped ID is unique and persistent within the VCS. That is, when a switch joins the VCS for the first time, the VCS assigns a mapped ID to the switch. This mapped ID persists with the switch, even if the switch leaves the VCS. When the switch joins the VCS again at a later time, the same mapped ID is used by the VCS to retrieve previous configuration information for the switch. This feature can reduce the amount of configuration overhead in VCS. Also, the persistent mapped ID allows the VCS to "recognize" a previously configured member switch when it re-joins the VCS, since a dynamically assigned FC fabric domain ID would change each time the member switch joins and is configured by the VCS.

Default switch configuration table **604** has an entry for the mappedID that points to the corresponding ST in VCS configuration database **600**. Note that only VCS configuration database **600** is replicated and distributed to all switches in the VCS. Default switch configuration table **604** is local to a particular member switch.

The "IN_VCS" value in default switch configuration table **604** indicates whether the member switch is part of a VCS. A switch is considered to be "in a VCS" when it is assigned one of the FC switch domains by the FC switch fabric with two or more switch domains. If a switch is part of an FC switch fabric that has only one switch domain, i.e., its own switch domain, then the switch is considered to be "not in a VCS."

When a switch is first connected to a VCS, the logical FC switch fabric formation process allocates a new switch domain ID to the joining switch. In one embodiment, only the switches directly connected to the new switch participate in the VCS join operation.

Note that in the case where the global configuration database of a joining switch is current and in sync with the global configuration database of the VCS based on a comparison of the transaction IDs of the two databases (e.g., when a member switch is temporarily disconnected from the VCS and reconnected shortly afterward), a trivial merge is performed. That is, the joining switch can be connected to the VCS, and no change or update to the global VCS configuration database is required.

FIG. 7 illustrates an exemplary process of a switch joining a virtual cluster switch, in accordance with an embodiment of the present invention. In this example, it is assumed that a switch **702** is within an existing VCS, and a switch **704** is joining the VCS. During operation, both switches **702** and **704** trigger an FC State Change Notification (SCN) process. Subsequently, both switches **702** and **704** perform a PRE-INVITE operation. The pre-invite operation involves the following process.

When a switch joins the VCS via a link, both neighbors on each end of the link present to the other switch a VCS four-tuple of <Prior VCS_ID, SWITCH_MAC, mappedID, IN_VCS> from a prior incarnation, if any. Otherwise, the switch presents to the counterpart a default tuple. If the VCS_ID value was not set from a prior join operation, a VCS_ID value of -1 is used. In addition, if a switch's IN_VCS flag is set to 0, it sends out its interface configuration to the neighboring switch. In the example in FIG. 7, both switches **702** and **704** send the above information to the other switch.

After the above PRE-INVITE operation, a driver switch for the join process is selected. By default, if a switch's IN_VCS

value is 1 and the other switch's IN_VCS value is 0, the switch with IN_VCS=1 is selected as the driver switch. If both switches have their IN_VCS values as 1, then nothing happens, i.e., the PRE-INVITE operation would not lead to an INVITE operation. If both switches have their IN_VCS values as 0, then one of the switches is elected to be the driving switch (for example, the switch with a lower FC switch domain ID value). The driving switch's IN_VCS value is then set to 1 and drives the join process.

After switch **702** is selected as the driver switch, switch **702** then attempts to reserve a slot in the VCS configuration database corresponding to the mappedID value in switch **704**'s PRE-INVITE information. Next, switch **702** searches the VCS configuration database for switch **704**'s MAC address in any mappedID slot. If such a slot is found, switch **702** copies all information from the identified slot into the reserved slot. Otherwise, switch **702** copies the information received during the PRE-INVITE from switch **704** into the VCS configuration database. The updated VCS configuration database is then propagated to all the switches in the VCS as a prepare operation in the database (note that the update is not committed to the database yet).

Subsequently, the prepare operation may or may not result in configuration conflicts, which may be flagged as warnings or fatal errors. Such conflicts can include inconsistencies between the joining switch's local configuration or policy setting and the VCS configuration. For example, a conflict arises when the joining switch is manually configured to allow packets with a particular VLAN value to pass through, whereas the VCS does not allow this VLAN value to enter the switch fabric from this particular RBridge (for example, when this VLAN value is reserved for other purposes). In one embodiment, the prepare operation is handled locally and/or remotely in concert with other VCS member switches. If there is an un-resolvable conflict, switch **702** sends out a PRE-INVITE-FAILED message to switch **704**. Otherwise, switch **702** generates an INVITE message with the VCS's merged view of the switch (i.e., the updated VCS configuration database).

Upon receiving the INVITE message, switch **704** either accepts or rejects the INVITE. The INVITE can be rejected if the configuration in the INVITE is in conflict with what switch **704** can accept. If the INVITE is acceptable, switch **704** sends back an INVITE-ACCEPT message in response. The INVITE-ACCEPT message then triggers a final database commit throughout all member switches in the VCS. In other words, the updated VCS configuration database is updated, replicated, and distributed to all the switches in the VCS.

Layer-2 Services in VCS

In one embodiment, each VCS switch unit performs source MAC address learning, similar to what an Ethernet bridge does. Each {MAC address, VLAN} tuple learned on a physical port on a VCS switch unit is registered into the local Fibre Channel Name Server (FC-NS) via a logical Nx_Port interface corresponding to that physical port. This registration binds the address learned to the specific interface identified by the Nx_Port. Each FC-NS instance on each VCS switch unit coordinates and distributes all locally learned {MAC address, VLAN} tuples with every other FC-NS instance in the fabric. This feature allows the dissemination of locally learned {MAC addresses, VLAN} information to every switch in the VCS. In one embodiment, the learned MAC addresses are aged locally by individual switches.

FIG. 8 presents a flowchart illustrating the process of looking up an ingress frame's destination MAC address and forwarding the frame in a VCS, in accordance with one embodiment of the present invention. During operation, a VCS

switch receives an Ethernet frame at one of its Ethernet ports (operation **802**). The switch then extracts the frame's destination MAC address and queries the local FC Name Server (operation **804**). Next, the switch determines whether the FC-NS returns an N_Port or an NL_Port identifier that corresponds to an egress Ethernet port (operation **806**).

If the FC-NS returns a valid result, the switch forwards the frame to the identified N_Port or NL_Port (operation **808**). Otherwise, the switch floods the frame on the TRILL multi-cast tree as well as on all the N_Ports and NL_Ports that participate in that VLAN (operation **810**). This flood/broadcast operation is similar to the broadcast process in a conventional TRILL RBridge, wherein all the physical switches in the VCS will receive and process this frame, and learn the source address corresponding to the ingress RBridge. In addition, each receiving switch floods the frame to its local ports that participate in the frame's VLAN (operation **812**). Note that the above operations are based on the presumption that there is a one-to-one mapping between a switch's TRILL identifier (or nickname) and its FC switch domain ID. There is also a one-to-one mapping between a physical Ethernet port on a switch and the corresponding logical FC port. End-to-End Frame Delivery and Exemplary VCS Member Switch

FIG. 9 illustrates how data frames and control frames are transported in a VCS, in accordance with an embodiment of the present invention. In this example, a VCS **930** includes member switches **934**, **936**, **938**, **944**, **946**, and **948**. An end host **932** is communicating with an end host **940**. Switch **934** is the ingress VCS member switch corresponding to host **932**, and switch **938** is the egress VCS member switch corresponding to host **938**. During operation, host **932** sends an Ethernet frame **933** to host **940**. Ethernet frame **933** is first encountered by ingress switch **934**. Upon receiving frame **933**, switch **934** first extracts frame **933**'s destination MAC address. Switch **934** then performs a MAC address lookup using the Ethernet name service, which provides the egress switch identifier (i.e., the RBridge identifier of egress switch **938**). Based on the egress switch identifier, the logical FC switch in switch **934** performs a routing table lookup to determine the next-hop switch, which is switch **936**, and the corresponding output port for forwarding frame **933**. The egress switch identifier is then used to generate a TRILL header (which specifies the destination switch's RBridge identifier), and the next-hop switch information is used to generate an outer Ethernet header. Subsequently, switch **934** encapsulates frame **933** with the proper TRILL header and outer Ethernet header, and sends the encapsulated frame **935** to switch **936**. Based on the destination RBridge identifier in the TRILL header of frame **935**, switch **936** performs a routing table lookup and determines the next hop. Based on the next-hop information, switch **936** updates frame **935**'s outer Ethernet header and forwards frame **935** to egress switch **938**.

Upon receiving frame **935**, switch **938** determines that it is the destination RBridge based on frame **935**'s TRILL header. Correspondingly, switch **938** strips frame **935** of its outer Ethernet header and TRILL header, and inspects the destination MAC address of its inner Ethernet header. Switch **938** then performs a MAC address lookup and determines the correct output port leading to host **940**. Subsequently, the original Ethernet frame **933** is transmitted to host **940**.

As described above, the logical FC switches within the physical VCS member switches may send control frames to one another (for example, to update the VCS global configuration database or to notify other switches of the learned MAC addresses). In one embodiment, such control frames can be FC control frames encapsulated in a TRILL header and

an outer Ethernet header. For example, if the logical FC switch in switch **944** is in communication with the logical FC switch in switch **938**, switch **944** can send a TRILL-encapsulated FC control frame **942** to switch **946**. Switch **946** can forward frame **942** just like a regular data frame, since switch **946** is not concerned with the payload in frame **942**.

VCS Name Services

VCS allows an interconnected fabric of Rbridges to function as a single logical switch. The VCS name services facilitate fast distribution of run-time network state changes, including newly learned MAC addresses (which is referred to as “Ethernet name service” or “Ethernet NS” in this disclosure) and multi-chassis trunk (MCT) port state updates (which is referred to as “MCT name service” or “MCT NS” in this disclosure). More details on MCT are provided in U.S. patent application Ser. No. 12/725,249, entitled “REDUNDANT HOST CONNECTION IN A ROUTED NETWORK,” by inventors Somesh Gupta, Anoop Ghanwani, Phanidhar Koganti, and Shunjia Yu, filed 16 Mar. 2010, the disclosure of which is incorporated by reference herein.

The Ethernet NS provides the ability to distribute various information across the VCS. The MAC information learned at one member switch is distributed to all other member switches, which facilitates fast MAC moves (for example, during migration of virtual machines) and global MAC learning. In some embodiments, layer-2 multicast information, which can be a multicast MAC address with corresponding switch/port identifiers and VLAN tag, can be distributed to facilitate efficient VCS-wide multicast. Optionally, Ethernet NS provides a distribution mechanism and does not maintain a central storage of the MAC-related knowledge base. In other words, the Ethernet NS knowledge database is replicated and stored distributively among all the VCS member switches.

Each member switch maintains a database of all the MAC addresses learned throughout the VCS. This database can be used to minimize the amount of flooding (a default behavior of Ethernet switch when a frame’s destination MAC address is not recognized). Ethernet NS also provides VCS-wide distribution of multicast MAC-to-RBridge/Port mapping information which can be obtained by Internet Group Management Protocol (IGMP) snooping. (Details about IGMP and IGMP snooping can be found at IETF RFC 3376 available at <http://tools.ietf.org/html/rfc3376> and IETF RFC 4541 available at <http://tools.ietf.org/html/rfc4541>.) Ethernet NS distributes this information to all Rbridges, thereby allowing the VCS to behave as a single switch. By tracking and forwarding IGMP join and leave information, the Ethernet NS can efficiently track the multicast MAC information and maintain an accurate layer-2 multicast group.

One of the requirements of presenting a VCS as a single switch is to support connection of trunked links from external hosts to different Rbridges within the VCS fabric. Such trunking which involves connection to different Rbridges is referred to as multi-chassis trunking (MCT). Conceptually, support within the VCS fabric for routing to a MCT destination is achieved by presenting each MCT group (i.e., each trunk) as a virtual RBridge. In some embodiments, the virtual RBridge is not assigned a domain ID and thus does not utilize FSPF for routing setup. Instead, the primary RBridge hosting the MCT distributes the virtual RBridge ID and the corresponding link state updates to the VCS fabric. The primary RBridge is responsible for learning a new MAC via an MCT and distributing the new MAC information to the VCS.

When an RBridge joins the VCS it will request a dump of the local NS database from the remote RBridge. It will not respond to individual updates from the remote RBridge until

the DB dump has been received. After the database is in sync between two Rbridges, individual changes are detected locally and pushed remotely. If a local database receives domain unreachable it is responsible for removing all records for that remote domain and doing any local notification that this removal implies.

FIG. 10 illustrates an example of name service operation in a VCS, in accordance with one embodiment of the present invention. In this example, a VCS **1000** includes four member switches (Rbridges), **1002**, **1004**, **1006**, and **1008**. Assume that an end host **1014** is coupled to switch **1002** during operation. When end host **1014** sends its first Ethernet frame, switch **1002** would not recognize the source MAC address of this ingress frame. Upon receiving this ingress frame, switch **1002** then determines the port (or interface) on which the frame arrives and the frame’s VLAG tag. Subsequently, switch **1002** assembles an Ethernet NS update frame which indicates the learned MAC address (which corresponds to end host **1014**), its switch identifier (which in one embodiment is the RBridge ID of switch **1002**), the port identifier, and the VLAG tag for the frame. In one embodiment, this frame is an FC registered state change notification (RSCN) encapsulated in a TRILL header. Note that switch **1002** can obtain the information of all other member switches in the VCS by looking up the global configuration database. Subsequently, switch **1002** can send the Ethernet NS update frame to switches **1004**, **1008**, and **1006**, respectively. Upon receiving the Ethernet NS update frame, each member switch updates its own MAC database accordingly. In this way, when one of the member switches receives an Ethernet frame destined to end-host **1014**, it can forward that frame to switch **1002** (instead of flooding the frame to all of its ports).

Also shown in the example in FIG. 10 is an MCT group **1016**. MCT group **1016** is formed by an end host **1012** which is dual-homed with switches **1006** and **1008**. Assume that switch **1006** is the primary RBridge in MCT group **1016**. When end host **1012** and MCT group **1010** is first configured, switch **1006** assigns a virtual RBridge **1010** to MCT group **1010**. In addition, switch **1006** notifies the rest of VCS **1000** about the MAC address of end host **1012**. Note that the NS update associated the MAC address of end host **1012** indicates the identifier of virtual RBridge **1010** (instead of the identifier of either switch **1006** or switch **1008**). In this way, the rest of VCS **1000** can associate end host **1012** with virtual RBridge **1010**. When forwarding a frame destined to end host **1012**, a member switch in VCS **1000** would forward the frame toward virtual RBridge **1010** (i.e., by setting RBridge **1010** as the destination RBridge in the TRILL header). Note that switch **1006** is also responsible for distributing the link state information with respect to the virtual connectivity between virtual RBridge **1010** and switches **1006** and **1008** (indicated by the dotted lines).

In case when one of the links (i.e., either the link between switch **1006** and end host **1012**, or the link between switch **1008** and end host **1012**) fails, as part of the MCT NS, in one embodiment, primary RBridge **1006** is responsible for updating the rest of the VCS **1000** that host **1012**’s MAC address is no longer associated with virtual RBridge **1010**. Instead, the MAC address of host **1012** is now associated with the switch to which host **1012** remains connected. In a further embodiment, it can be the responsibility of the switch that remains connected to host **1012** to distribute the updated MAC address association to the rest of VCS **1000**.

FIG. 11 presents a flowchart illustrating the process of distributing learned MAC information by the Ethernet name service in a VCS, in accordance with one embodiment of the present invention. During operation, a VCS member switch

15

detects an ingress frame with a new source MAC address (operation 1102). The switch then identifies the port on which the ingress frame is received (operation 1104). Subsequently, the switch assembles an Ethernet NS update frame with the learned MAC address, the switch identifier, port identifier, and VLAN tag (operation 1106). The switch then distributes the Ethernet NS update frames to all member switches in the VCS (operation 1108).

FIG. 12 presents a flowchart illustrating the process of distributing information of a learned MAC address via an MCT, in accordance with one embodiment of the present invention. During operation, assume that one of the switches in a MCT group detects an ingress frame with a new source MAC address (operation 1202). The switch then determines whether the end host which generates the frame is dual-homed with the MCT group (operation 1204). In one embodiment, the switch can make this determination by communicating with the other switch of the MCT group. In a further embodiment, the switch can inspect the link aggregation group (LAG) ID of the ingress frame to determine whether the end host is transmitting using a LAG. If the frame is an MCT frame, the switch then assembles an Ethernet NS update frame with the MAC address, the virtual RBridge identifier corresponding to the MCT, a port identifier, and the VLAG tag of the frame (operation 1206).

If the frame is determined to be from a regular end host (i.e., not a dual-homed host), the switch assembles an Ethernet NS updated frame with the MAC address, the local physical switch identifier (as opposed to the virtual RBridge ID), the identifier of the port on which the frame is received, and the frame's VLAN tag (operation 1207). The switch then distributes the Ethernet NS update frames to all the member switches in the VCS (operation 1208).

FIG. 13 presents a flowchart illustrating the process of updating the link state in an MCT group, in accordance with one embodiment of the present invention. During operation, assume one of the MCT partner switches detects a link or port failure which is part of the MCT group (operation 1302). Note that this failure can be detected locally (which means a port on the local switch or a link coupled to a local port has failed), or be detected remotely (which means that the failure occurs on the partner switch and the local switch is notified of the failure by the partner switch). The switch then determines whether the MCT end host is still connected to the local switch (operation 1304). If the end host is no longer connected to the local switch, the local switch optionally notifies the other partner switch in the MCT of the failure (operation 1310) and takes no further actions, assuming that the partner switch will assume responsibility of updating the link state (using, for example, the same procedure illustrated in FIG. 13).

If the MCT end host is still connected to the local switch, the switch then assembles an NS update frame with the end host's MAC address, the local switch's identifier (e.g., the physical RBridge ID of the local switch), the identifier of the port through which the end host is connected, and the proper VLAN tag (operation 1306). The switch then distributes the NS update frames to all member switches in the VCS (operation 1308).

Exemplary VCS Member Switch

FIG. 14 illustrates an exemplary switch that facilitates formation of a virtual cluster switch with Ethernet and MCT name services, in accordance with an embodiment of the present invention. The VCS member switch is a TRILL RBridge 1400 running special VCS software. RBridge 1400 includes a number of Ethernet communication ports 1401, which can transmit and receive Ethernet frames and/or TRILL encapsulated frames. Also included in RBridge 1400

16

is a packet processor 1402, a virtual FC switch management module 1404, a logical FC switch 1405, a VCS configuration database 1406, a name services management module 1407, and a TRILL header generation module 1408.

During operation, packet processor 1402 extracts the source and destination MAC addresses of incoming frames, and attaches proper Ethernet or TRILL headers to outgoing frames. Virtual FC switch management module 1404 maintains the state of logical FC switch 1405, which is used to join other VCS switches using the FC switch fabric protocols. VCS configuration database 1406 maintains the configuration state of every switch within the VCS. TRILL header generation module 1408 is responsible for generating property TRILL headers for frames that are to be transmitted to other VCS member switches. Based on the extracted MAC addresses of incoming frames, NS management module 1407 distributes the NS update frames to the rest of the VCS. NS management module 1407 also maintains a copy of NS database 1409. NS database 1409 stores all the learned MAC address information from every member switch in the VCS.

The methods and processes described herein can be embodied as code and/or data, which can be stored in a computer-readable non-transitory storage medium. When a computer system reads and executes the code and/or data stored on the computer-readable non-transitory storage medium, the computer system performs the methods and processes embodied as data structures and code and stored within the medium.

The methods and processes described herein can be executed by and/or included in hardware modules or apparatus. These modules or apparatus may include, but are not limited to, an application-specific integrated circuit (ASIC) chip, a field-programmable gate array (FPGA), a dedicated or shared processor that executes a particular software module or a piece of code at a particular time, and/or other programmable-logic devices now known or later developed. When the hardware modules or apparatus are activated, they perform the methods and processes included within them.

The foregoing descriptions of embodiments of the present invention have been presented only for purposes of illustration and description. They are not intended to be exhaustive or to limit this disclosure. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. The scope of the present invention is defined by the appended claims.

What is claimed is:

1. A first switch, comprising:

a processor;

memory;

one or more local ports of the switch; and

a non-transitory computer-readable storage medium storing instructions which when executed by the processor causes the processor to perform a method, the method comprising:

extracting from a message information associated with a media access control (MAC) address learned at a port of a second switch, wherein the first and second switches are member switches of a fabric switch;

storing the extracted information associated with the MAC address in association with an identifier of the second switch in a name service database in the memory of the switch, wherein the identifier of the second switch uniquely identifies the second switch in the fabric switch; and

constructing a message for distributing to the second switch information associated with a MAC address learned at a said local port.

17

2. The first switch of claim 1, wherein the fabric switch is an Ethernet fabric switch comprising one or more physical switches; and wherein the Ethernet fabric switch operates as one single Ethernet switch.

3. The first switch of claim 1, wherein the message constructed for distributing to the second switch is a Fibre Channel registered state change notification (RSCN) encapsulated in a transparent interconnection of lots of links (TRILL) header.

4. The first switch of claim 1, wherein the message constructed for distributing to the second switch includes the MAC address and an identifier of the first switch, wherein the identifier of the first switch uniquely identifies the first switch in the fabric switch.

5. The first switch of claim 4, wherein the message constructed for distributing to the second switch further includes: an identifier of the local port from which the MAC address is learned; and a virtual local area network (VLAN) tag associated with the learned MAC address.

6. The first switch of claim 1, wherein the method further comprises constructing an update message destined for the second switch in response to a failure to a link or port within a multi-chassis trunk.

7. The first switch of claim 6, wherein the update message indicates that an end host previously coupled via the multi-chassis trunk is now coupled with a physical switch.

8. A switching system, comprising: a plurality of member switches; memory in a first member switch of the plurality of member switches; one or more local ports in the first member switch; and a non-transitory computer-readable storage medium residing in the first member switch and storing instructions which when executed by a processor in first member switch causes the processor to perform a method, the method comprising:

extracting from payload of a message information associated with a media access control (MAC) address learned at a port of a second member switch of the plurality of member switches;

storing the extracted information associated with the MAC address in association with an identifier of the second member switch in a name service database in the memory of the first member switch, wherein the identifier of the second member switch uniquely identifies the second member switch in the switching system; and

constructing a message for distributing to the second member switch information associated with a MAC address learned at a said local port of the first member switch.

9. The switching system of claim 8, wherein the switching system is an Ethernet fabric switch comprising one or more physical switches; and wherein the Ethernet fabric switch operates as one single Ethernet switch.

10. The switching system of claim 8, wherein the message constructed for distributing to the second member switch is a Fibre Channel registered state change notification (RSCN) encapsulated in a transparent interconnection of lots of links (TRILL) header.

11. The switching system of claim 8, wherein the message constructed for distributing to the second member switch includes the MAC address and an identifier of the first mem-

18

ber switch, wherein the identifier of the first member switch uniquely identifies the first member switch in the switching system.

12. The switching system of claim 11, wherein message constructed for distributing to the second member switch further includes:

an identifier of the local port of the first member switch from which the MAC address is learned; and a virtual local area network (VLAN) tag associated with the MAC address.

13. The switching system of claim 8, wherein the method further comprises constructing an update message destined for the second member switch in response to a failure to a link or port within a multi-chassis trunk.

14. The switching system of claim 13, wherein the update message indicates that an end host previously coupled via the multi-chassis trunk is now coupled with a physical switch.

15. A method, comprising:

extracting, by a first switch, from payload of a message information associated with a media access control (MAC) address learned at a port of a second switch, wherein the first and second switches are member switches of a fabric switch;

storing the extracted information associated with the MAC address in association with an identifier of the second switch in a name service database in memory of the first switch, wherein the identifier of the second switch uniquely identifies the second switch in the fabric switch; and

constructing a message for distributing to the second switch through a local port of the first switch information associated with a MAC address learned at a local port of the first switch.

16. The method of claim 15, wherein the fabric switch is an Ethernet fabric switch comprising one or more physical switches; and

wherein the Ethernet fabric switch operates as one single Ethernet switch.

17. The method of claim 15, wherein the message constructed for distributing to the second switch is a Fibre Channel registered state change notification (RSCN) encapsulated in a transparent interconnection of lots of links (TRILL) header.

18. The method of claim 15, wherein message constructed for distributing to the second switch includes the MAC address and an identifier of the first switch, wherein the identifier of the first switch uniquely identifies the first switch in the fabric switch.

19. The method of claim 18, wherein the message constructed for distributing to the second switch further includes: an identifier of a the local port of the first switch from which the MAC address is learned; and a virtual local area network (VLAN) tag associated with the MAC address.

20. The method of claim 18, further comprising constructing an update message destined for the second switch in response to a failure to a link or port within a multi-chassis trunk fails.

21. The method of claim 20, wherein the update message indicates that an end host previously coupled via the multi-chassis trunk is now coupled with a physical switch.

22. A first switch means, comprising:

a control means for:

extracting from payload of a message information associated with a media access control (MAC) address

learned at a port of a second switch means, wherein
the first and second switch means are members of a
fabric switch means;
storing the extracted information associated with the
MAC address in association with an identifier of the 5
second switch means in a name service database,
wherein the identifier of the second switch means
uniquely identifies the second switch means in the
fabric switch means; and
constructing a message for distributing to the second 10
switch means information associated with a MAC
address learned at a local port.

* * * * *