

Juniper Networks, Inc., v. Portsmouth Network Corporation

PATENT OWNER *Portsmouth's* **PRESENTATION**

Inter Partes Review of
U.S. Patent No. 6,917,986

IPR2024-00893

1

Grounds

Ground	Claims	Basis
#1	1-3, 6-15, 18-24	Gai in view of IEEE 1999
#2	4-5, 16-17	Gai in view of IEEE 1999 and U.S. Patent No. 6,330,229 to Jain

Roadmap

Ground 1

Independent claims 1, 12, 24 (Dependent claims 2, 7, 9-11, 13, 14, 16, 18, 19, and 21-23)

*Gai and IEEE
1999*

- No Motivation To Combine - IEEE 1999 Tear Down And Gai's Dummy Messages Function Differently
- Neither Gai or IEEE 1999 Disclose Sending Dummy Messages Downstream
- Gai Does Not Disclose The "Data Frames" Limitation Of Dependent Claims 6 and 18

Ground 2

Dependent claims 4, 5, 6, 17

*Gai, IEEE
1999, and
Jain*

- Fail For The Same Reason As The Independent Claims They Rely On
- No Motivation To Combine – Impermissible Hindsight

Ground 1

Independent claims 1, 12, 24 (Dependent claims 2, 7, 9-11, 13, 14, 16, 18, 19, and 21-23)

Gai and IEEE 1999

No Motivation To Combine - IEEE 1999 Tear Down And Gai's Dummy Messages Function Differently

U.S. Patent No. 6,917,986

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

(10) Patent No.: **US 6,917,986 B2**
(45) Date of Patent: **Jul. 12, 2005**

(54) **FAST FAILURE PROTECTION USING
REDUNDANT NETWORK EDGE PORTS**

6,917,986 B2 1/2004 Nguyen

(75) Inventors: **Gal Mor, Rishon le Zion (IL); Shmuel
Ilan, Ra'anana (IL)**

(73) Assignee: **Corrigent Systems Ltd., Tel Aviv (IL)**

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 734 days.

(21) Appl. No.: **10/036,518**

(22) Filed: **Jan. 7, 2002**

(65) **Prior Publication Data**

US 2003/028618 A1 Nov. 6, 2003

(51) Int. Cl. **G06F 15/173**

(52) U.S. Cl. **709/238, 709/208, 709/226,**

709/209, 709/259, 707/104, 1, 370/396,

370/252, 370/352, 370/389

(58) Field of Search **709/238, 259,**

709/208, 209, 226; 707/104, 1, 370/396,

252, 352, 389

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,159,595 A 10/1992 Huang et al.

5,307,353 A 4/1994 Yamashita et al.

5,925,137 A 7/1999 Okamoto et al.

6,246,697 B1 6/2001 Bellare et al.

OTHER PUBLICATIONS

"IEEE Standard for Information Technology, Telecommunications and Information Exchange between Systems, Local and Metropolitan Area Networks, Common Specifications, Part 3: Media Access Control (MAC) Bridges", Published as ANSI/IEEE Standard 802.1D (1998). Available at: standards.ieee.org/catalog/IEEE802.1.html.

Primary Examiner—William A. Cechinski, Jr.

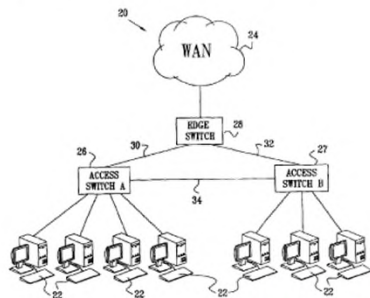
Assistant Examiner—Thanh T. Nguyen

(74) Attorney, Agent, or Firm—Nixon & Vanderhye P.C.

(57) **ABSTRACT**

A method for fast protection in a fault-tolerant network includes coupling an edge switch in a network to one or more access switches via at least first and second redundant links, respectively connected to first and second ports of the edge switch. The first port is placed in an active state, while the second port is placed in a blocking state, so that communication traffic is conveyed over the first link, and each of the switches builds a respective database for use in forwarding the traffic. Responsive to a failure associated with the first link, the second port is placed in the active state, and the first port is placed in the blocking state. Dummy traffic is then sent from the edge switch over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.

24 Claims, 4 Drawing Sheets



IPR2024-00505
Cisco Systems, Inc. v. Portsmouth Network Corp.
Cisco's Exhibit 1001
Ex. 1001, Page 1

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

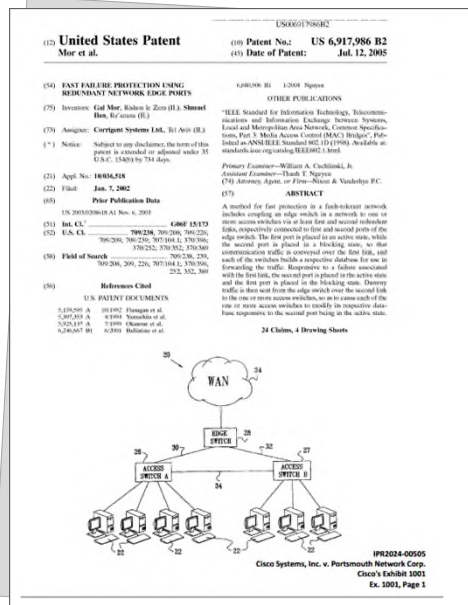
(10) Patent No.: **US 6,917,986 B2**
(45) Date of Patent: **Jul. 12, 2005**

(54) **FAST FAILURE PROTECTION USING
REDUNDANT NETWORK EDGE PORTS**

(75) Inventors: **Gal Mor, Rishon le Zion (IL); Shmuel
Ilan, Ra'anana (IL)**

(73) Assignee: **Corrigent Systems Ltd., Tel Aviv (IL)**

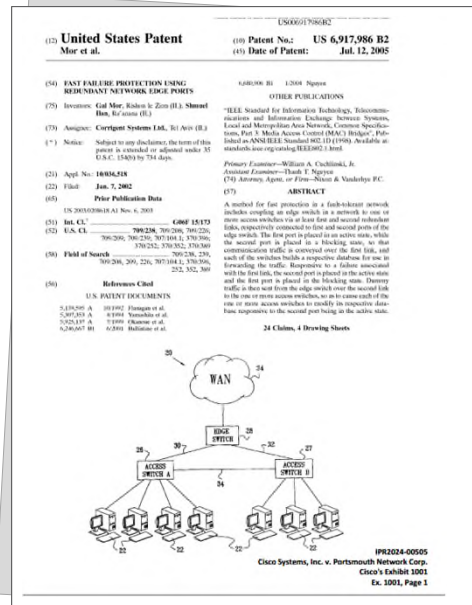
U.S. Patent No. 6,917,986 – Independent Claim 1



What is claimed is:

1. A method for network communication, comprising:
coupling an edge switch in a network to one or more access switches via at least first and second redundant links, respectively connected to first and second ports of the edge switch;
placing the first port in an active state, while placing the second port in a blocking state;
conveying communication traffic over the first link while the first port is in the active state, thereby causing each of the switches to build a respective database for use in forwarding the traffic;
responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and
sending dummy traffic from the edge switch over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.

U.S. Patent No. 6,917,986 – Independent Claim 12



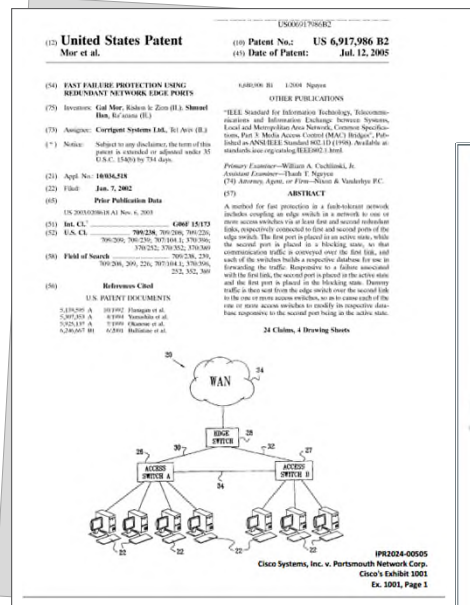
12. A method for network communication, comprising:
coupling a first bridge in a network to a second bridge located downstream of the first bridge, by connecting at least first and second redundant links to respective first and second ports of the first bridge so as to communicate with the second bridge;

placing the first port in an active state, while placing the second port in a blocking state;

conveying communication traffic over the first link while the first port is in the active state, thereby causing the bridges to build respective databases for use in forwarding the traffic;

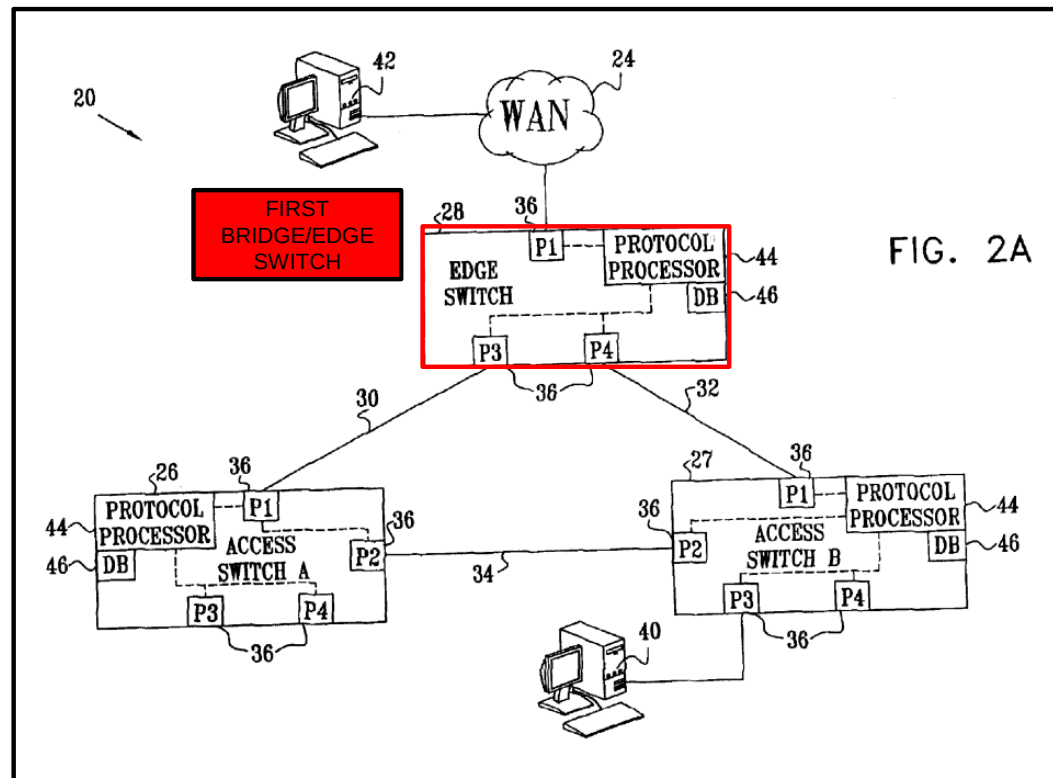
responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and sending dummy traffic from the first bridge over the second link downstream to the second bridge, so as to cause the second bridge to modify its database responsive to the second port being in the active state.

The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover

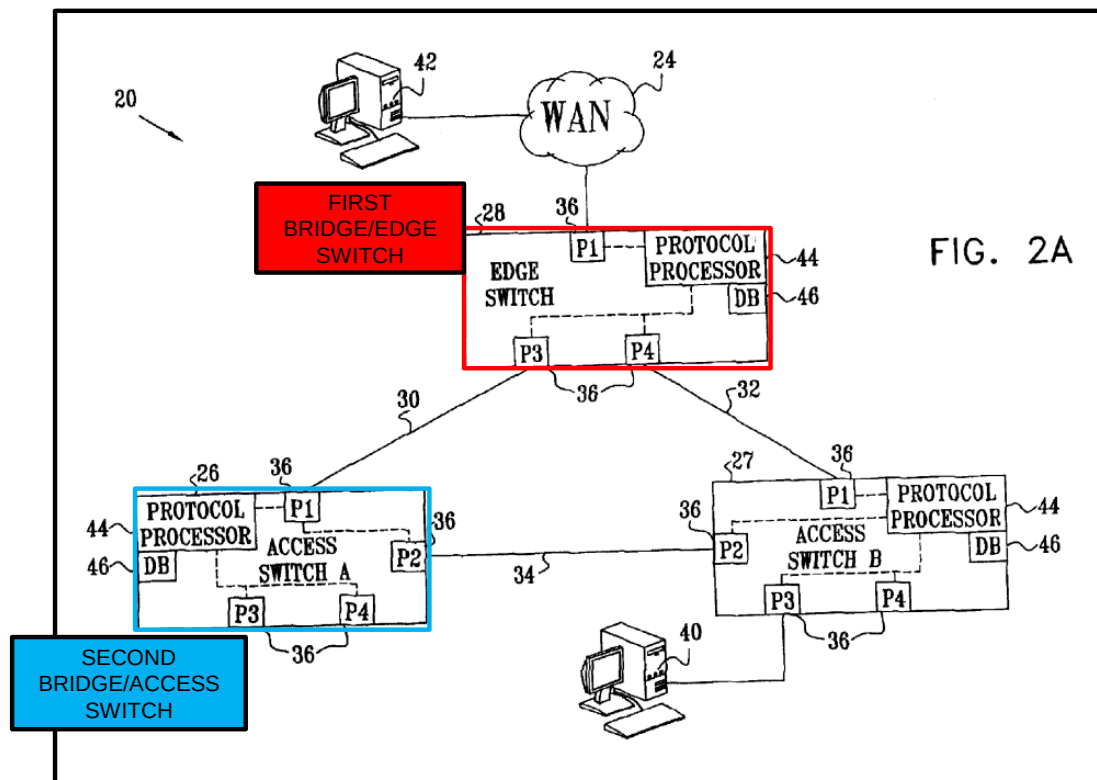


Thus, preferred embodiments of the present invention provide a method for fast changeover between redundant network links that can be implemented by modifying a single MAC bridge, without the need for reprogramming or replacing other items of equipment in the network. The bridge is modified, as described above, so as to manage its redundant downstream ports in a novel way. This modifi-

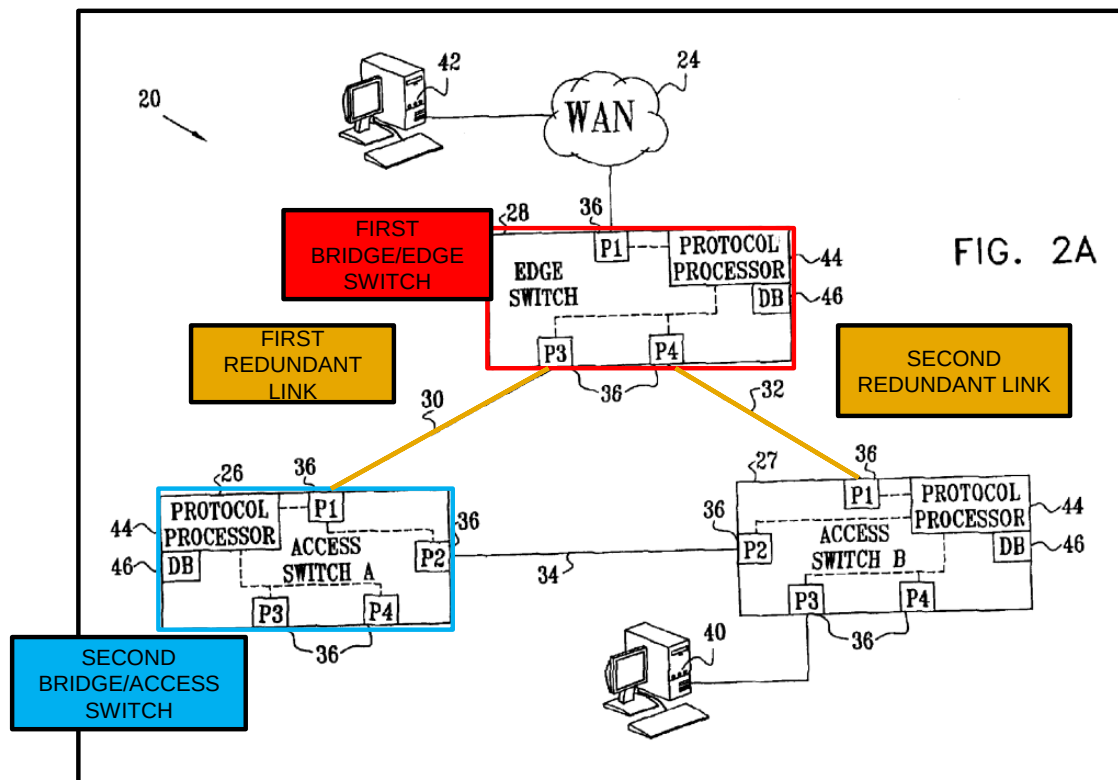
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



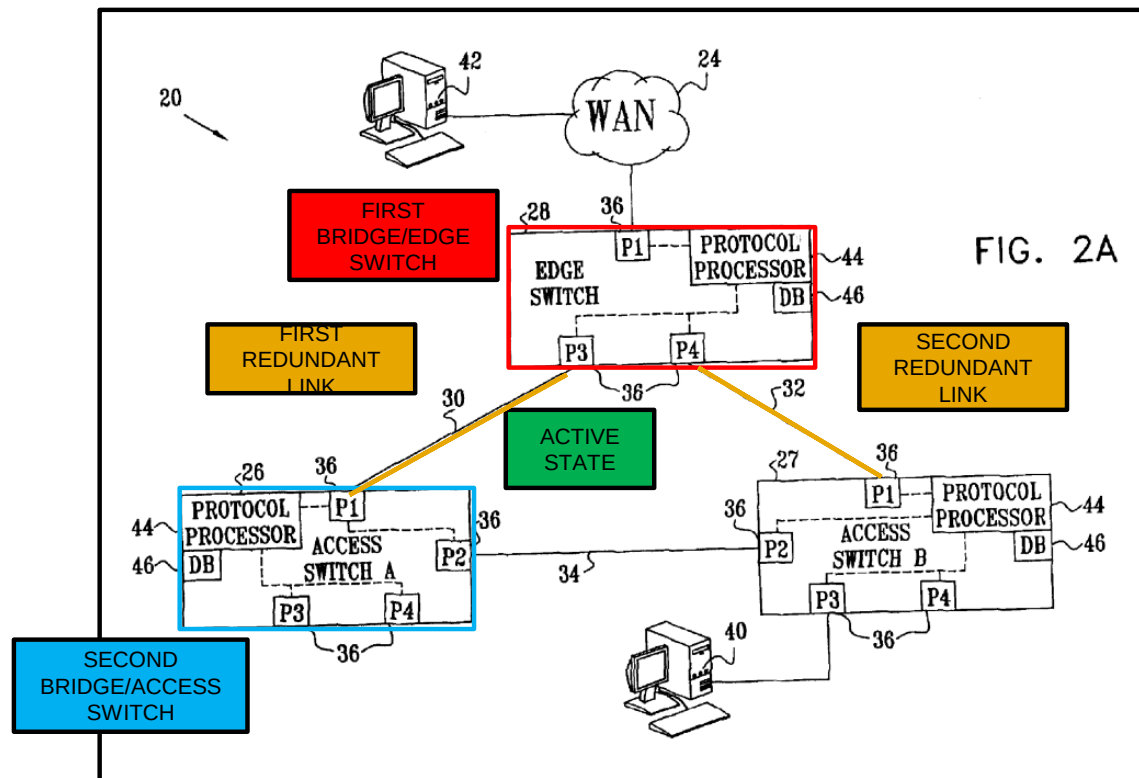
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



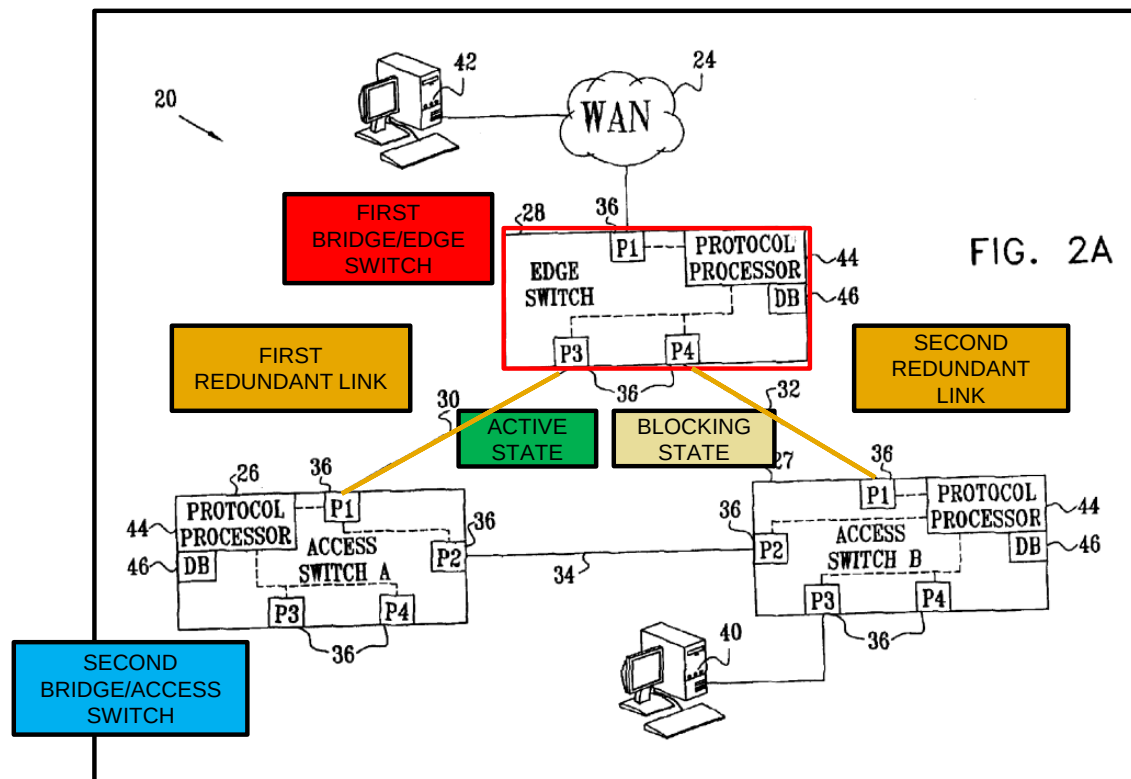
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



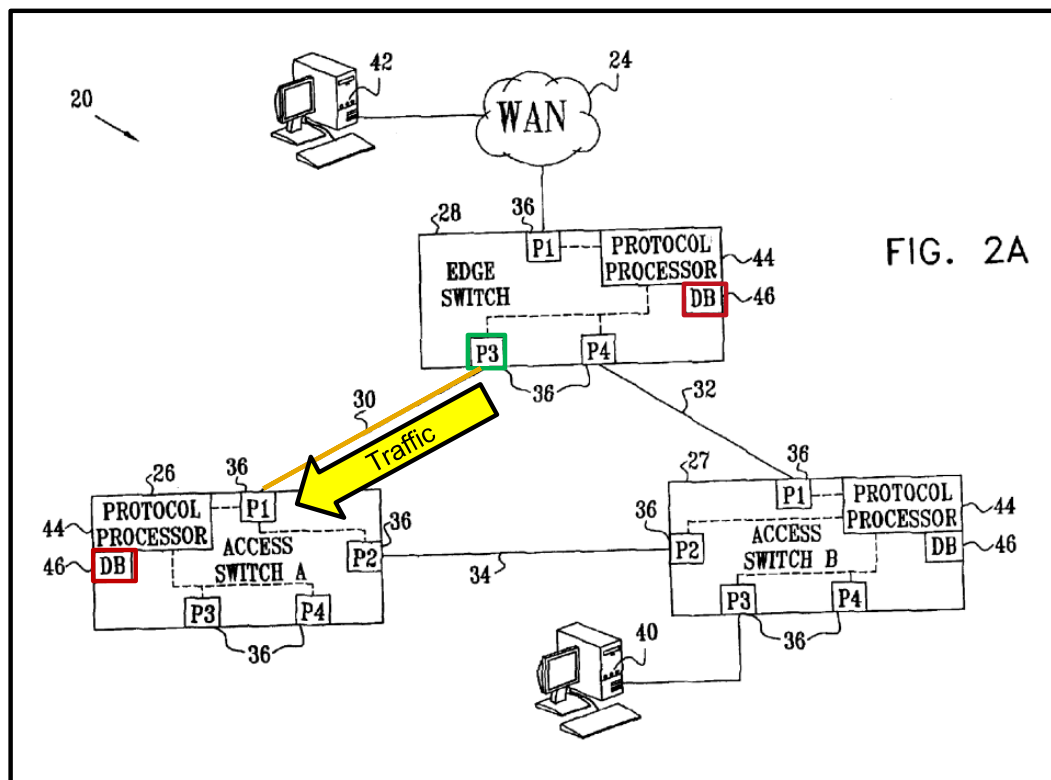
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



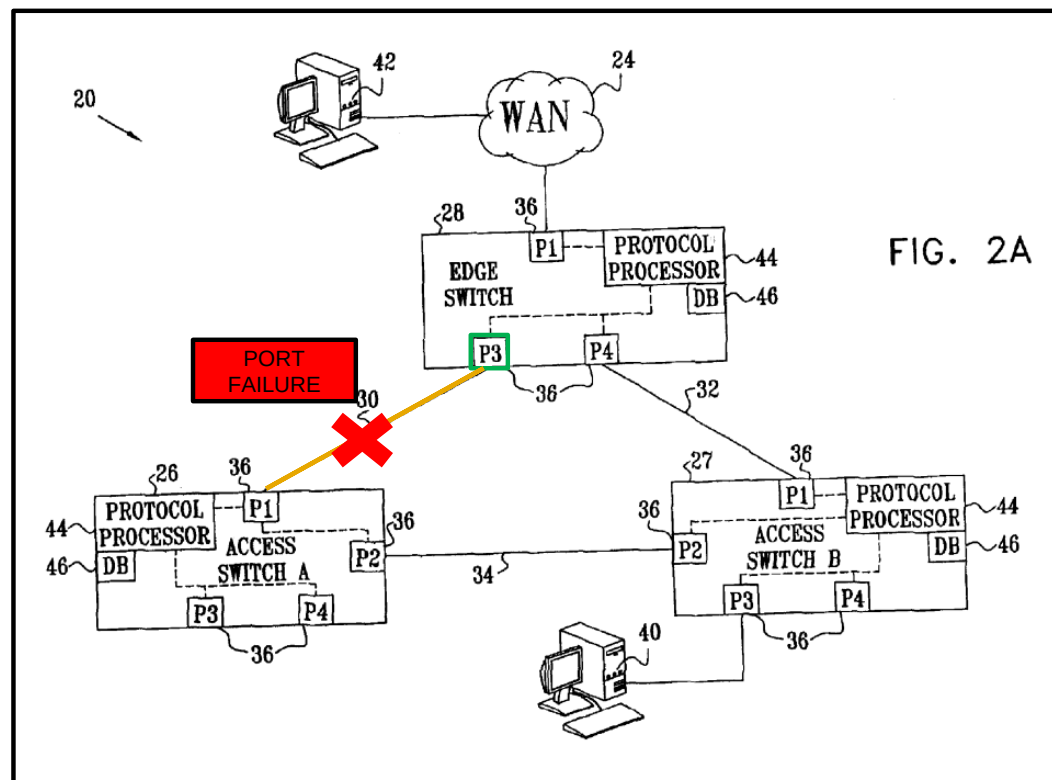
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



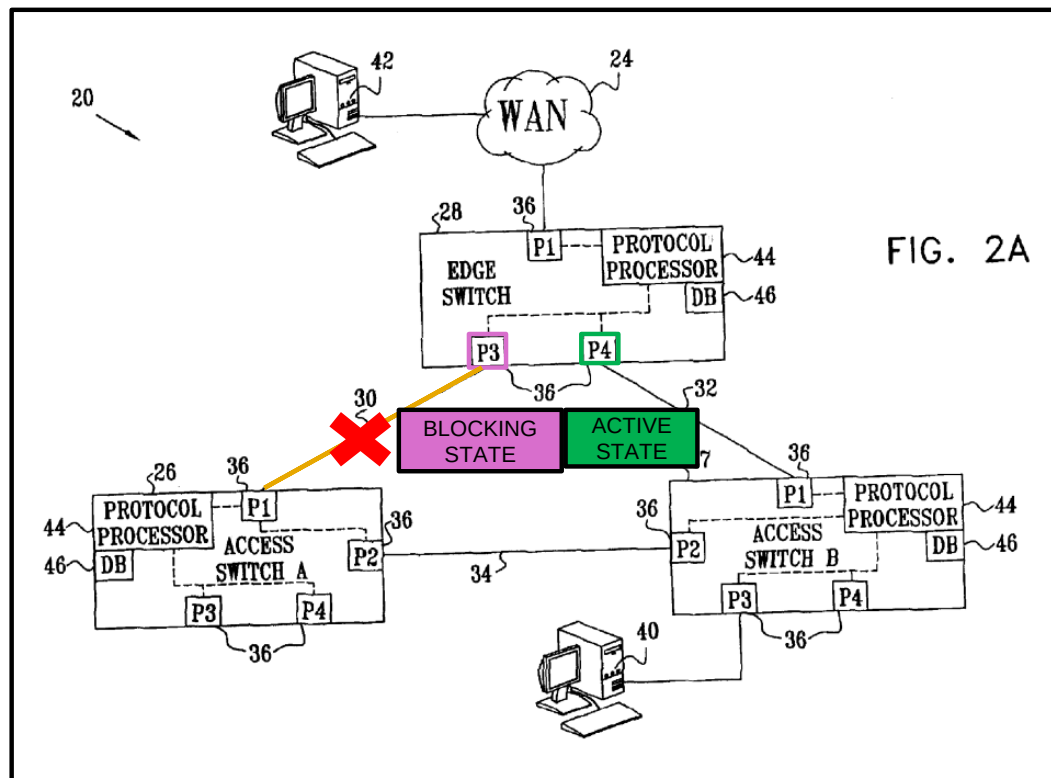
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



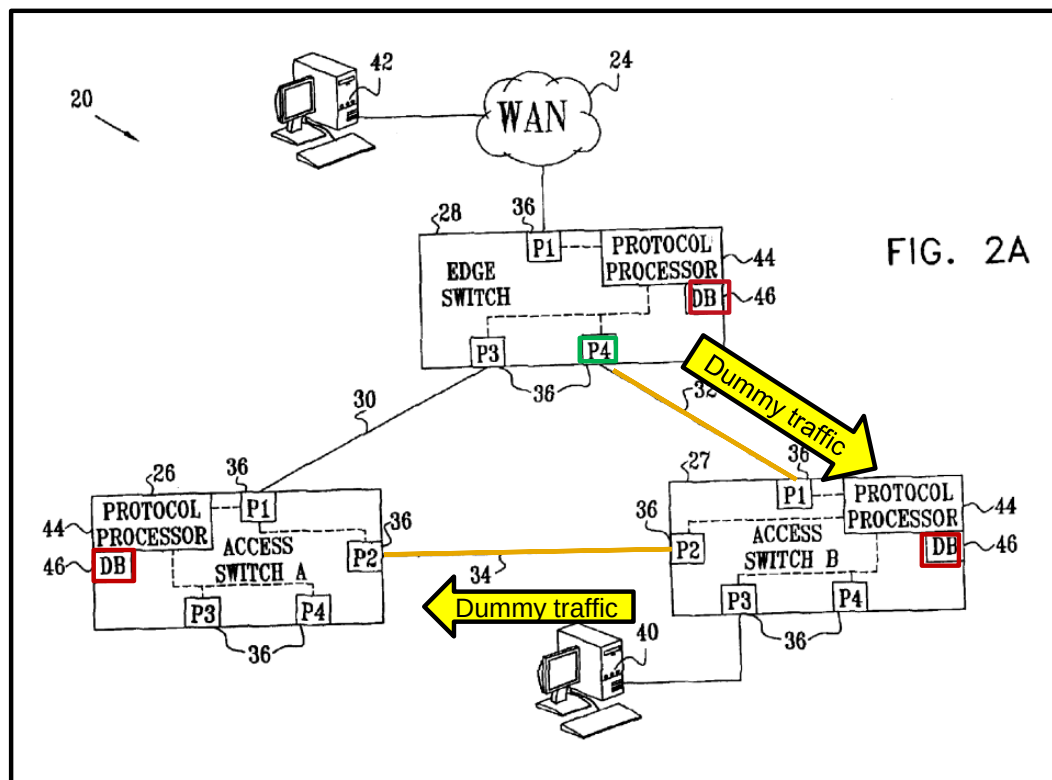
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



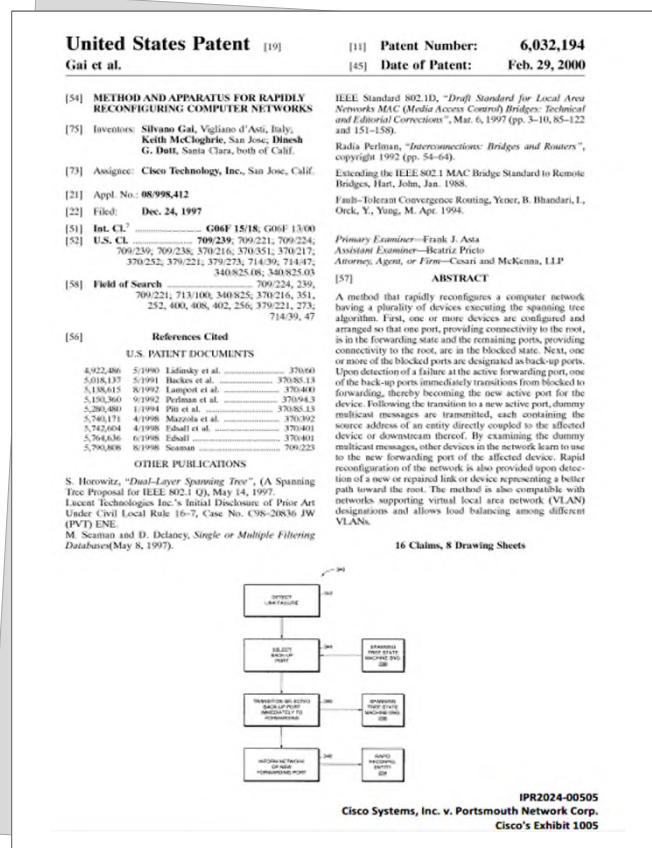
The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



The '986 Patent Sends Dummy Traffic *Downstream* to Notify of a Port Changeover



U.S. Patent No. 6,032,194 – *Gai*



United States Patent [19]
Gai et al.

[11] **Patent Number:** **6,032,194**
[45] **Date of Patent:** **Feb. 29, 2000**

[54] **METHOD AND APPARATUS FOR RAPIDLY RECONFIGURING COMPUTER NETWORKS**

[75] Inventors: **Silvano Gai**, Vigliano d'Asti, Italy; **Keith McCloghrie**, San Jose; **Dinesh G. Dutt**, Santa Clara, both of Calif.

[73] Assignee: **Cisco Technology, Inc.**, San Jose, Calif.

Gai Teaches Transmitting Dummy Traffic *Upstream* Towards the Root

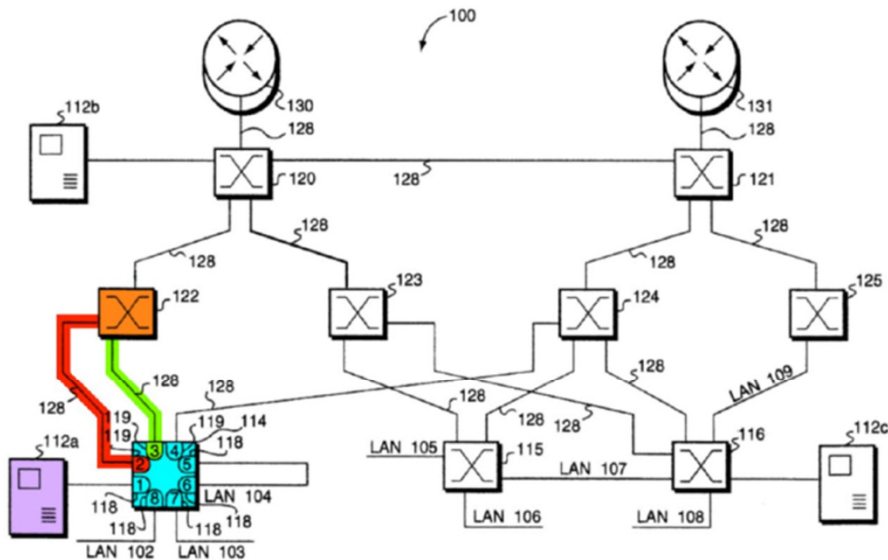


FIG. 1

Annotated Figure 1 of Gai before link failure

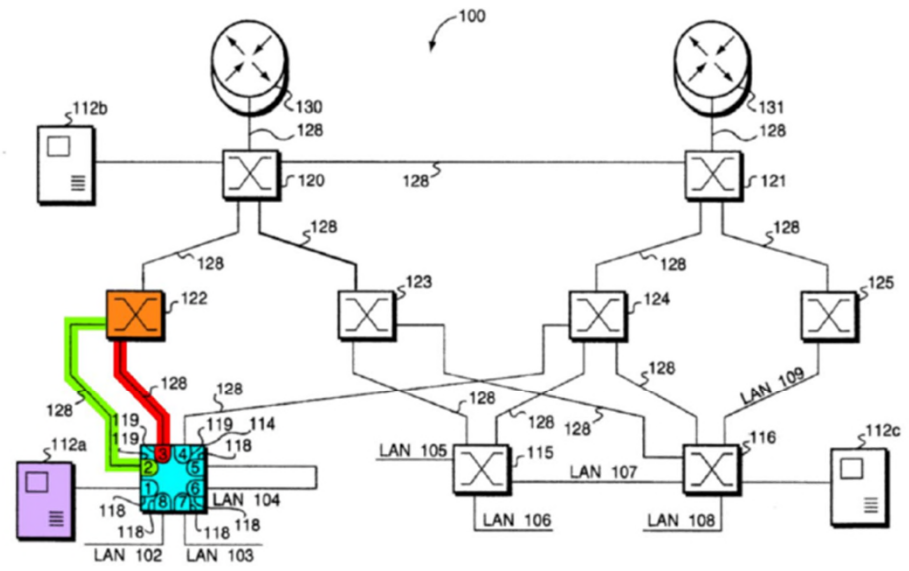
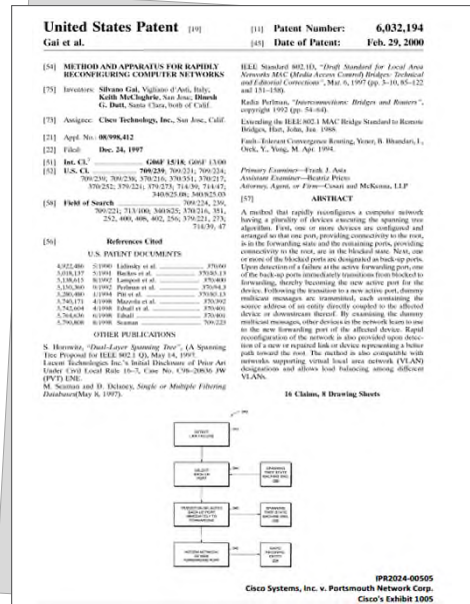


FIG. 1

Annotated Figure 1 of Gai after link failure

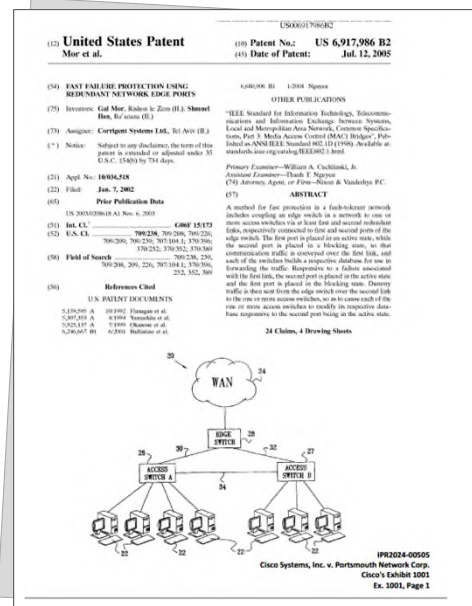
Gai* Teaches Transmitting Dummy Traffic *Upstream



state. Additionally, one or more of the blocked ports are designated as back-up ports. Upon detection of a failure at the active forwarding port, the state of one of the back-up ports immediately transitions from blocked to forwarding, thereby becoming the new active port for the device.

Upon transition to the new forwarding port, the device begins transmitting “dummy” multicast messages through the new port. These dummy multicast messages carry the source address of each entity that is directly coupled to the device with the new active port or downstream thereof (relative to the root) and are received by other devices in the computer network. Upon receipt, the other devices examine

'986 Patent Confirms Gai is Limited to Sending Dummy Traffic Upstream



U.S. Pat. No. 6,032,194, whose disclosure is incorporated herein by reference, describes a method for rapidly reconfiguring a computer network, based on certain modifications to the standard STP. The method assumes that the switches in the network, including both core and access switches, have multiple trunk ports providing connectivity to the STP root via trunk switches. (These trunk switches are computers are designated as back-up ports. Upon detection of a failure at the active forwarding port, the switch immediately places one of the back-up trunk ports in the forwarding state, so that it becomes the new active port. The switch then transmits dummy multicast messages toward the root (upstream) through the newly-active trunk port, each containing the source address of a network entity that is coupled to the switch itself or is downstream of the switch. Upon receiving these messages, other devices in the network learn to use the new active port. In order to implement this rapid

Petitioner's Expert Admits Gai's Dummy Messages Are Sent Upstream

18 Q. So, here Gai is discussing sending
19 dummy multicast messages upstream, correct?

20 A. That's correct.

2 The portion I just read does not
3 discuss Gai transmitting dummy multicast
4 messages downstream, correct?

5 A. That's correct.

Gai's Dummy Messages Sent After Port Transition During Reconfiguration

United States Patent [19]
Gai et al.

[11] Patent Number: **6,032,194**
[45] Date of Patent: **Feb. 29, 2000**

[54] **METHOD AND APPARATUS FOR RAPIDLY RECONFIGURING COMPUTER NETWORKS**

[75] Inventors: **Sharon Gai, Vigliani d'Amico, Indy, Keith McChesney, San Jose, Blum G. Dutt, Santa Clara, both of Calif.**

[73] Assignee: **Cisco Technology, Inc., San Jose, Calif.**

[21] Appl. No.: **08/99,412**

[22] Filed: **Dec. 24, 1997**

[51] Int. Cl.⁷: **G06F 15/16, G06F 13/00**

[52] U.S. Cl.: **709/229, 709/231, 709/234, 709/236, 709/238, 709/239, 709/241, 370/252, 370/253, 370/273, 714/96, 714/97, 340/825.06, 340/825.03**

[56] **Field of Search**: **709/229, 709/234, 252, 400, 406, 408, 402, 256, 379/221, 273, 714/96, 47**

References Cited

U.S. PATENT DOCUMENTS

4,922,486	01/09	Lafont et al.	07/090
5,015,117	01/09	Becker et al.	07/043.13
5,110,415	03/92	Lampert et al.	07/044
5,135,360	01/92	Reimer et al.	07/043.13
5,266,487	11/94	Pill et al.	07/043.13
5,740,171	4/1998	Moskowitz et al.	07/043.13
5,742,644	4/1998	Edell et al.	07/043.13
5,742,645	4/1998	Edell et al.	07/043.13
5,742,646	4/1998	Edell et al.	07/043.13

OTHER PUBLICATIONS

S. Horowitz, "Dual-Layer Spanning Tree", (A Spanning Tree Protocol for IEEE 802.1Q), May 14, 1997, Lucent Technologies Inc.'s Initial Disclosure of Prior Art Under Title Local Rule 16-5, Case No. C98-2065, PW (PVT) LINE.

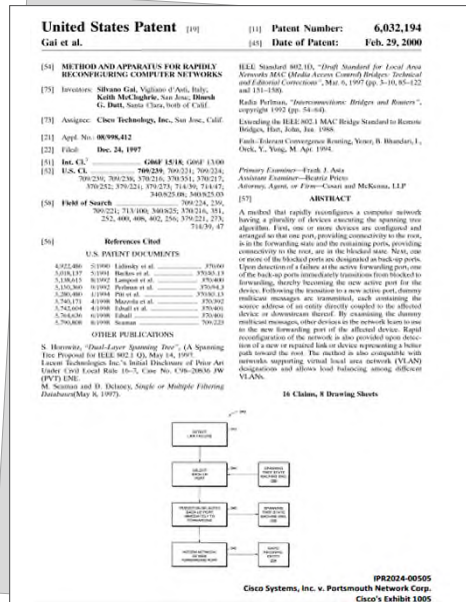
M. Scharf and D. DeSousa, "Single or Multiple Filtering Database(s)", May 8, 1997.

16 Claims, 8 Drawing Sheets

IPR2024-00505
Cisco Systems, Inc. v. Portsmouth Network Corp.
Cisco's Exhibit 1005

Upon transition to the new forwarding port, the device begins transmitting "dummy" multicast messages through the new port. These dummy multicast messages carry the source address of each entity that is directly coupled to the device with the new active port or downstream thereof (relative to the root) and are received by other devices in the computer network. Upon receipt, the other devices examine the contents of these messages and note the port on which they were received, which may differ from the port on which messages from these entities were previously received (i.e., before the failure and subsequent replacement of the device's active port). It is through this process that other devices within the network learn to utilize the new forwarding port, rather than the failed port, when directing messages to these entities. Notably, the transition to a new forwarding port is accomplished without other devices having to discard the contents of their filtering databases and, thus, the flooding of messages following a network change is substantially reduced.

Gai's Dummy Messages Sent To Other Devices So They Can Learn And Use the New Forwarding Port



Upon transition to the new forwarding port, the device begins transmitting “dummy” multicast messages through the new port. These dummy multicast messages carry the source address of each entity that is directly coupled to the device with the new active port or downstream thereof (relative to the root) and are received by other devices in the computer network. Upon receipt, the other devices examine the contents of these messages and note the port on which they were received, which may differ from the port on which messages from these entities were previously received (i.e., before the failure and subsequent replacement of the device’s active port). It is through this process that other devices within the network learn to utilize the new forwarding port, rather than the failed port, when directing messages to these entities. Notably, the transition to a new forwarding port is accomplished without other devices having to discard the contents of their filtering databases and, thus, the flooding of messages following a network change is substantially reduced.

IEEE 1999 Fault Detection Mechanism

Automatic Fault Detection and Recovery in Real Time Switched *Ethernet* Networks

Srinidhi Varadarajan Tzi-cker Chiueh

Department of Computer Science
State University of New York,
Stony Brook NY 11794-4400
(srinidhi, chiueh)@cs.sunysb.edu

ABSTRACT

EtherReal is a real-time Fast *Ethernet* switch architecture that provides bandwidth guarantees to distributed multimedia applications without OS or hardware modifications on the host machines. It implements true link-layer multicast, and offers a natural match to support network-layer QoS protocols such as RSVP. Because real-time performance guarantees fundamentally require state to be installed inside the network, link/switch failures could lead to significant disruption to the QoS promised to the user applications. This paper describes the fault detection and recovery mechanism supported by the *EtherReal* architecture, and reports on the performance measurements of the initial prototype implementation. The heart of *EtherReal*'s fault detection and recovery mechanism is a fast spanning tree reconfiguration algorithm to reduce the total fault recovery time, and a delayed link inactivation scheme that allows real-time connections which are not affected by the failed links/switches to continue to exist, even though some of the links are marked as "blocked" in the new spanning tree topology. Measurements on the prototype show that the fault detection and recovery time on a network whose diameter is 10 hops are 220ms and 31ms, respectively. This combined delay corresponds to a minor jitter in real-time audio/video communication, and is a significant improvement over the standard IEEE 802.1d implementation, which takes on the order of 30 sec.

1 INTRODUCTION

With the availability of powerful desktop computers, distributed multimedia applications are becoming commonplace. In contrast to best effort traffic, multimedia traffic places strict resource requirements in terms of bandwidth, end to end delays and delay variance. A large body of the research effort in networks is concentrated on providing Quality of Service (QoS) guarantees over wide area networks at the internetwork layer, assuming that the underlying medium access layer is capable of delivering such guarantees. *Ethernet*, which forms the last mile of the network, is incapable of guaranteeing any performance bound. QoS guarantee mechanisms over *Ethernet* are needed for an end-to-end resource guarantees.

A connection-oriented mechanism is needed to deliver hard resource guarantees, since it ensures that traffic flows through a fixed path on which resources have been reserved to deliver the guarantee. Our *EtherReal*[11] architecture provides a transparent connection-oriented mechanism for delivering

bandwidth guarantees over switched *Ethernet* networks without any changes to the operating system/device drivers or network hardware on the hosts connected to the network. The functionality for establishing real time connections, maintaining resource reservations and transferring best effort traffic is implemented in *EtherReal* switches.

A connection-oriented network maintains distributed state associating connection identifiers with their resource reservations. In the case of network failures, automatic fault detection and recovery mechanisms are needed to recover the distributed state and reestablish broken connections. However, real time connection oriented networks place additional constraints on their fault detection and recovery mechanisms. First, to minimize the effect of a fault, the fault detection mechanisms have to be responsive. Long fault detection times may void any resource guarantees offered by the network. Secondly, the fault recovery mechanism should constrain the effect of the fault to the affected link(s)/switch(es) and the connections through them, while allowing normal operation in the rest of the network. Finally, the fault recovery mechanism has to be fast, so that the affected connections can be reestablished with minimal interruption in service.

Unlike the case for layer 3 devices, routing loops in transparent switched *Ethernet* networks pose serious problems. Because transparent layer 2 devices cannot maintain a hop count field to avoid loops, they superimpose a logical spanning tree upon the physical topology and transfer data only on the links belonging to the spanning tree. When new links are detected or links in the spanning tree fail, the spanning tree has to be rebuilt. Since routing has to wait till the spanning tree has been built, spanning tree rebuild constitutes the major part of the time spent in fault recovery.

Current transparent switched *Ethernet* networks use the IEEE 802.1d distributed spanning tree algorithm to discover a loop free network topology and recreate it, in the event of changes. IEEE 802.1d was written with best effort traffic in mind and has several drawbacks in a real time connection oriented environment. First, it uses global timers to detect the completion of the spanning tree build phase. Because the size

3.4 FAULT RECOVERY

When the fault detection mechanism detects a change in the network topology, it notifies the fault recovery mechanism and indicates the offending link. In the interior of the network, the fault would be independently detected by switches both upstream as well as downstream of the fault. The fault recovery mechanism on each switch that detects the fault, performs the following functions:

1. Discard all best effort traffic
2. Terminate all connections going through the offending link.
3. Rebuild the spanning tree
4. Reestablish the connections that were torn down

IEEE 1999 Discloses Tear Down Messages Notifying Hosts of Network Instability and Connection Termination

Automatic Fault Detection and Recovery in Real Time Switched *Ethernet* Networks

Srinidhi Varadarajan Tzi-cker Chiueh

Department of Computer Science
State University of New York,
Stony Brook NY 11794-4400
(srinidhi, chiueh)@cs.sunysb.edu

ABSTRACT

EtherReal is a real-time Fast *Ethernet* switch architecture that provides bandwidth guarantees to distributed multimedia applications without OS or hardware modifications on the host machines. It implements true link-layer multicast, and offers a natural match to support network-layer QoS protocols such as RSVP. Because real-time performance guarantees fundamentally require state to be installed inside the network, link/switch failures could lead to significant disruption to the QoS promised to the user applications. This paper describes the fault detection and recovery mechanism supported by the *EtherReal* architecture, and reports on the performance measurements of the initial prototype implementation. The heart of *EtherReal*'s fault detection and recovery mechanism is a fast spanning tree reconfiguration algorithm to reduce the total fault recovery time, and a delayed link inactivation scheme that allows real-time connections which are not affected by the failed links/switches to continue to exist, even though some of the links are marked as "blocked" in the new spanning tree topology. Measurements on the prototype show that the fault detection and recovery time on a network whose diameter is 10 hops are 220ms and 31ms, respectively. This combined delay corresponds to a minor jitter in real-time audio/video communication, and is a significant improvement over the standard IEEE 802.1d implementation, which takes on the order of 30 sec.

1 INTRODUCTION

With the availability of powerful desktop computers, distributed multimedia applications are becoming commonplace. In contrast to best effort traffic, multimedia traffic places strict resource requirements in terms of bandwidth, end to end delays and delay variance. A large body of the research effort in networks is concentrated on providing Quality of Service (QoS) guarantees over wide area networks at the internetwork layer, assuming that the underlying medium access layer is capable of delivering such guarantees. *Ethernet*, which forms the last mile of the network, is incapable of guaranteeing any performance bound. QoS guarantee mechanisms over *Ethernet* are needed for an end-to-end resource guarantees.

A connection-oriented mechanism is needed to deliver hard resource guarantees, since it ensures that traffic flows through a fixed path on which resources have been reserved to deliver the guarantee. Our *EtherReal*[11] architecture provides a transparent connection-oriented mechanism for delivering

bandwidth guarantees over switched *Ethernet* networks without any changes to the operating system/device drivers or network hardware on the hosts connected to the network. The functionality for establishing real time connections, maintaining resource reservations and transferring best effort traffic is implemented in *EtherReal* switches.

A connection-oriented network maintains distributed state associating connection identifiers with their resource reservations. In the case of network failures, automatic fault detection and recovery mechanisms are needed to recover the distributed state and reestablish broken connections. However, real time connection oriented networks place additional constraints on their fault detection and recovery mechanisms. First, to minimize the effect of a fault, the fault detection mechanisms have to be responsive. Long fault detection times may void any resource guarantees offered by the network. Secondly, the fault recovery mechanism should constrain the effect of the fault to the affected link(s)/switch(es) and the connections through them, while allowing normal operation in the rest of the network. Finally, the fault recovery mechanism has to be fast, so that the affected connections can be reestablished with minimal interruption in service.

Unlike the case for layer 3 devices, routing loops in transparent switched *Ethernet* networks pose serious problems. Because transparent layer 2 devices cannot maintain a hop count field to avoid loops, they superimpose a logical spanning tree upon the physical topology and transfer data only on the links belonging to the spanning tree. When new links are detected or links in the spanning tree fail, the spanning tree has to be rebuilt. Since routing has to wait till the spanning tree has been built, spanning tree rebuild constitutes the major part of the time spent in fault recovery.

Current transparent switched *Ethernet* networks use the IEEE 802.1d distributed spanning tree algorithm to discover a loop free network topology and recreate it, in the event of changes. IEEE 802.1d was written with best effort traffic in mind and has several drawbacks in a real time connection oriented environment. First, it uses global timers to detect the completion of the spanning tree build phase. Because the size

The fault recovery mechanism then terminates all real time connections going through the offending link and informs the source of the connection that the network is unstable. The switches that detect the failure then initiate a spanning tree reconfiguration, with themselves as root. At the end of the spanning tree rebuild phase, all the hosts connected to the network are informed that the network is stable. Hosts that had their connections torn down due to network failure now initiate a new connection to their destination. In our implementation on *EtherReal*, reestablishment of torn down connections is done by the Real Time Communication Daemon, and is transparent to the host application, which merely sees a small interruption in service.

IEEE 1999 Discloses Tear Down Messages Notifying Hosts of Network Instability and Connection Termination

Automatic Fault Detection and Recovery in Real Time Switched *Ethernet* Networks

Srinidhi Varadarajan Tzi-cker Chiueh

Department of Computer Science
State University of New York,
Stony Brook NY 11794-4400
(srinidhi, chiueh)@cs.sunysb.edu

ABSTRACT

EtherReal is a real-time Fast *Ethernet* switch architecture that provides bandwidth guarantees to distributed multimedia applications without OS or hardware modifications on the host machines. It implements true link-layer multicast, and offers a natural match to support network-layer QoS protocols such as RSVP. Because real-time performance guarantees fundamentally require state to be installed inside the network, link/switch failures could lead to significant disruption to the QoS promised to the user applications. This paper describes the fault detection and recovery mechanism supported by the *EtherReal* architecture, and reports on the performance measurements of the initial prototype implementation. The heart of *EtherReal*'s fault detection and recovery mechanism is a fast spanning tree reconfiguration algorithm to reduce the total fault recovery time, and a delayed link inactivation scheme that allows real-time connections which are not affected by the failed links/switches to continue to exist, even though some of the links are marked as "blocked" in the new spanning tree topology. Measurements on the prototype show that the fault detection and recovery time on a network whose diameter is 10 hops are 220ms and 31ms, respectively. This combined delay corresponds to a minor jitter in real-time audio/video communication, and is a significant improvement over the standard IEEE 802.1d implementation, which takes on the order of 30 sec.

1 INTRODUCTION

With the availability of powerful desktop computers, distributed multimedia applications are becoming commonplace. In contrast to best effort traffic, multimedia traffic places strict resource requirements in terms of bandwidth, end to end delays and delay variance. A large body of the research effort in networks is concentrated on providing Quality of Service (QoS) guarantees over wide area networks at the internetwork layer, assuming that the underlying medium access layer is capable of delivering such guarantees. *Ethernet*, which forms the last mile of the network, is incapable of guaranteeing any performance bound. QoS guarantee mechanisms over *Ethernet* are needed for an end-to-end resource guarantees.

A connection-oriented mechanism is needed to deliver hard resource guarantees, since it ensures that traffic flows through a fixed path on which resources have been reserved to deliver the guarantee. Our *EtherReal*[11] architecture provides a transparent connection-oriented mechanism for delivering

bandwidth guarantees over switched *Ethernet* networks without any changes to the operating system/device drivers or network hardware on the hosts connected to the network. The functionality for establishing real time connections, maintaining resource reservations and transferring best effort traffic is implemented in *EtherReal* switches.

A connection-oriented network maintains distributed state associating connection identifiers with their resource reservations. In the case of network failures, automatic fault detection and recovery mechanisms are needed to recover the distributed state and reestablish broken connections. However, real time connection oriented networks place additional constraints on their fault detection and recovery mechanisms. First, to minimize the effect of a fault, the fault detection mechanisms have to be responsive. Long fault detection times may void any resource guarantees offered by the network. Secondly, the fault recovery mechanism should constrain the effect of the fault to the affected link(s)/switch(es) and the connections through them, while allowing normal operation in the rest of the network. Finally, the fault recovery mechanism has to be fast, so that the affected connections can be reestablished with minimal interruption in service.

Unlike the case for layer 3 devices, routing loops in transparent switched *Ethernet* networks pose serious problems. Because transparent layer 2 devices cannot maintain a hop count field to avoid loops, they superimpose a logical spanning tree upon the physical topology and transfer data only on the links belonging to the spanning tree. When new links are detected or links in the spanning tree fail, the spanning tree has to be rebuilt. Since routing has to wait till the spanning tree has been built, spanning tree rebuild constitutes the major part of the time spent in fault recovery.

Current transparent switched *Ethernet* networks use the IEEE 802.1d distributed spanning tree algorithm to discover a loop free network topology and recreate it, in the event of changes. IEEE 802.1d was written with best effort traffic in mind and has several drawbacks in a real time connection oriented environment. First, it uses global timers to detect the completion of the spanning tree build phase. Because the size

3.4 FAULT RECOVERY

When the fault detection mechanism detects a change in the network topology, it notifies the fault recovery mechanism and indicates the offending link. In the interior of the network, the fault would be independently detected by switches both upstream as well as downstream of the fault. The fault recovery mechanism on each switch that detects the fault, performs the following functions:

1. Discard all best effort traffic
2. Terminate all connections going through the offending link.
3. Rebuild the spanning tree
4. Reestablish the connections that were torn down

Both Experts Confirm IEEE 1999 Tear Down Messages Notify Hosts of Network Instability and Connection Termination

22 Q. But the teardown message is not sent
23 to reconfigure the spanning tree, correct?
24 A. No, that's not the intent of the
25 teardown message. It is to tear down. It is a
1 teardown message. It is to tear down a
2 connection.

Bruce McNair Deposition at 39:22-40:2

Both Experts Confirm IEEE 1999 Tear Down Messages Notify Hosts of Network Instability and Connection Termination

42. To terminate the connections through a failed link, the switches first terminate all real time connections on the link and then inform the source of these connections (e.g. the host computer which initiated the connection) that the network is unstable. *Id.* at 166 (“The fault recovery mechanism then terminates all real time connections going through the offending link and **informs the source of the connection** that the network is **unstable**”) (emphasis added). An example of the

Dr. Olivier Declaration at ¶ 42

IEEE 1999 Tear Down Messages Are Sent Before The Spanning Tree Is Reconfigured

Automatic Fault Detection and Recovery in Real Time Switched *Ethernet* Networks

Srinidhi Varadarajan Tzi-cker Chiueh

Department of Computer Science
State University of New York,
Stony Brook NY 11794-4400
(srinidhi, chiueh)@cs.sunysb.edu

ABSTRACT

EtherReal is a real-time Fast *Ethernet* switch architecture that provides bandwidth guarantees to distributed multimedia applications without OS or hardware modifications on the host machines. It implements true link-layer multicast, and offers a natural match to support network-layer QoS protocols such as RSVP. Because real-time performance guarantees fundamentally require state to be installed inside the network, link/switch failures could lead to significant disruption to the QoS promised to the user applications. This paper describes the fault detection and recovery mechanism supported by the *EtherReal* architecture, and reports on the performance measurements of the initial prototype implementation. The heart of *EtherReal*'s fault detection and recovery mechanism is a fast spanning tree reconfiguration algorithm to reduce the total fault recovery time, and a delayed link inactivation scheme that allows real-time connections which are not affected by the failed links/switches to continue to exist, even though some of the links are marked as "blocked" in the new spanning tree topology. Measurements on the prototype show that the fault detection and recovery time on a network whose diameter is 10 hops are 220ms and 31ms, respectively. This combined delay corresponds to a minor jitter in real-time audio/video communication, and is a significant improvement over the standard IEEE 802.1d implementation, which takes on the order of 30 sec.

1 INTRODUCTION

With the availability of powerful desktop computers, distributed multimedia applications are becoming commonplace. In contrast to best effort traffic, multimedia traffic places strict resource requirements in terms of bandwidth, end to end delays and delay variance. A large body of the research effort in networks is concentrated on providing Quality of Service (QoS) guarantees over wide area networks at the internetwork layer, assuming that the underlying medium access layer is capable of delivering such guarantees. *Ethernet*, which forms the last mile of the network, is incapable of guaranteeing any performance bound. QoS guarantee mechanisms over *Ethernet* are needed for an end-to-end resource guarantees.

A connection-oriented mechanism is needed to deliver hard resource guarantees, since it ensures that traffic flows through a fixed path on which resources have been reserved to deliver the guarantee. Our *EtherReal*[11] architecture provides a transparent connection-oriented mechanism for delivering

bandwidth guarantees over switched *Ethernet* networks without any changes to the operating system/device drivers or network hardware on the hosts connected to the network. The functionality for establishing real time connections, maintaining resource reservations and transferring best effort traffic is implemented in *EtherReal* switches.

A connection-oriented network maintains distributed state associating connection identifiers with their resource reservations. In the case of network failures, automatic fault detection and recovery mechanisms are needed to recover the distributed state and reestablish broken connections. However, real time connection oriented networks place additional constraints on their fault detection and recovery mechanisms. First, to minimize the effect of a fault, the fault detection mechanisms have to be responsive. Long fault detection times may void any resource guarantees offered by the network. Secondly, the fault recovery mechanism should constrain the effect of the fault to the affected link(s)/switch(es) and the connections through them, while allowing normal operation in the rest of the network. Finally, the fault recovery mechanism has to be fast, so that the affected connections can be reestablished with minimal interruption in service.

Unlike the case for layer 3 devices, routing loops in transparent switched *Ethernet* networks pose serious problems. Because transparent layer 2 devices cannot maintain a hop count field to avoid loops, they superimpose a logical spanning tree upon the physical topology and transfer data only on the links belonging to the spanning tree. When new links are detected or links in the spanning tree fail, the spanning tree has to be rebuilt. Since routing has to wait till the spanning tree has been built, spanning tree rebuild constitutes the major part of the time spent in fault recovery.

Current transparent switched *Ethernet* networks use the IEEE 802.1d distributed spanning tree algorithm to discover a loop free network topology and recreate it, in the event of changes. IEEE 802.1d was written with best effort traffic in mind and has several drawbacks in a real time connection oriented environment. First, it uses global timers to detect the completion of the spanning tree build phase. Because the size

To terminate the connections through a failed link we need to (a) identify the connection identifiers of the failed connections (b) send a teardown message to the upstream/downstream sources of outgoing connections and (c) tear the connection to the upstream/downstream destinations if incoming connections. The CPU overhead of processing the connection queue is 95 ns/connection, which includes time spent in freeing up resources used by the connection. The cost of sending a teardown message to the next upstream hop involves formulating the message and setting up a DMA transaction to transfer the message, which accounts for a total of 2 μ s/message/hop.

A POSITA Wouldn't Combine IEEE Tear Down with Gai's Dummy Messages Due to Core Differences

Where, When, and Why	IEEE Tear Down Message	Gai Dummy Message
Sent to different destinations	Host computers	Other devices in the network
Sent at different times	Before Reconfiguration and Traffic Forwarding Begin	During Reconfiguration After Port Transition
Serves different purposes	Notify Hosts Computers of Network Instability and Connection Loss	Help Devices Use the New Forwarding Port

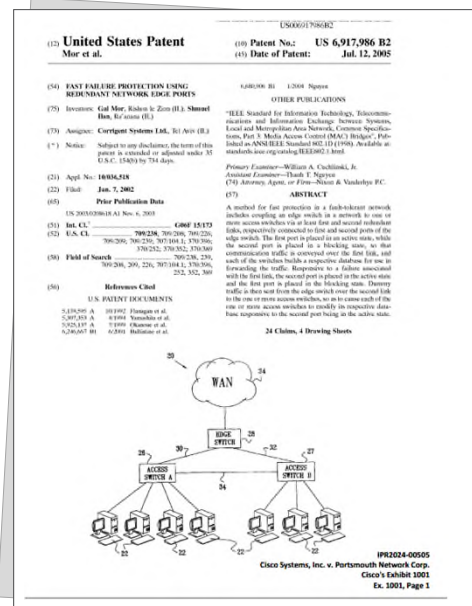
Ground 1

Independent claims 1, 12, 13 & 24 (Dependent claims 2, 4-7, 9-11, 14, 16-19, 21-23)

Gai and IEEE 1999

Neither Gai or IEEE 1999 Disclose Sending Dummy Messages Downstream

U.S. Patent No. 6,917,986 – Independent Claim 1



1. A method for network communication, comprising:
coupling an edge switch in a network to one or more access switches via at least first and second redundant links, respectively connected to first and second ports of the edge switch;
placing the first port in an active state, while placing the second port in a blocking state;
conveying communication traffic over the first link while the first port is in the active state, thereby causing each of the switches to build a respective database for use in forwarding the traffic;
responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and
sending dummy traffic from the edge switch over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.

No Support in *Gai* for Sending Traffic Downstream

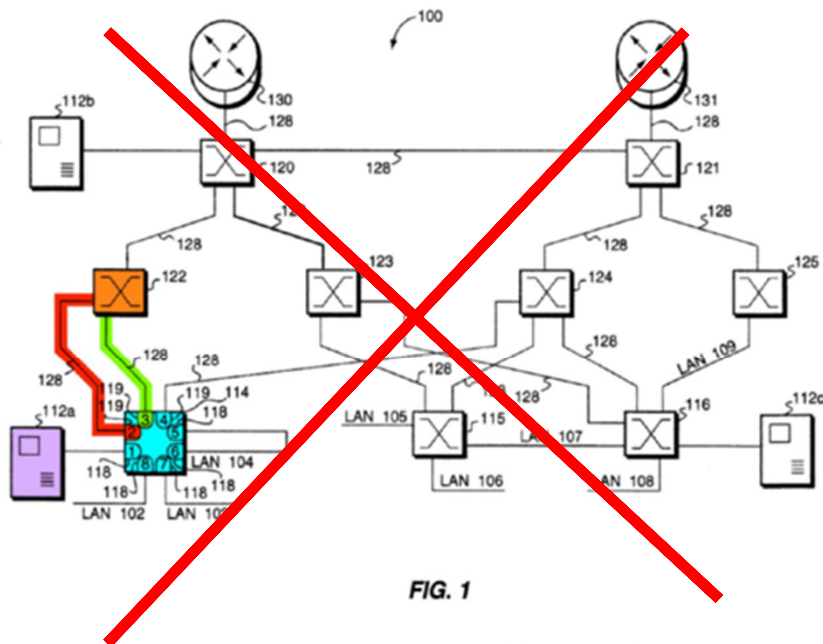


FIG. 1

Annotated Figure 1 of *Gai* before link failure

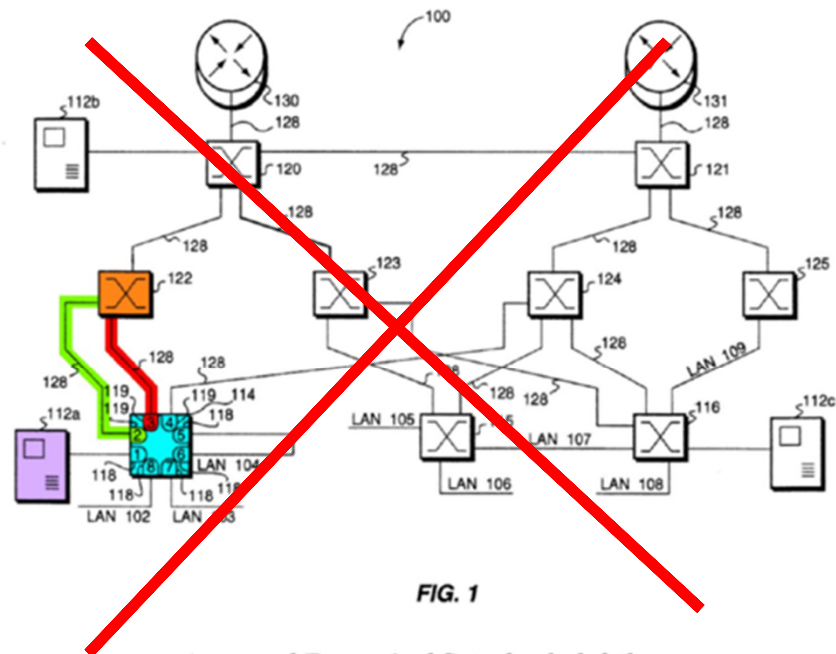


FIG. 1

Annotated Figure 1 of *Gai* after link failure

No Support in *Gai* for Sending Traffic Downstream

claim.” Reply at 2-3. But this point is irrelevant and incorrect. It is undisputed that *Gai* does not disclose sending dummy messages downstream. POR at 13-14. Nor

No Support in IEEE 1999 For Sending Dummy Messages

22 Q. But the teardown message is not sent
23 to reconfigure the spanning tree, correct?

24 A. No, that's not the intent of the
25 teardown message. It is to tear down. It is a
1 teardown message. It is to tear down a
2 connection.

No Support in IEEE 1999 For Sending Dummy Messages

To terminate the connections through a failed link we need to (a) identify the connection identifiers of the failed connections (b) send a teardown message to the upstream/downstream sources of outgoing connections and (c) tear the connection to the upstream/downstream destinations if incoming connections. The CPU overhead of processing the connection queue is 95 ns/connection, which includes time spent in freeing up resources used by the connection. The cost of sending a teardown message to the next upstream hop involves formulating the message and setting up a DMA transaction to transfer the message, which accounts for a total of 2us/message/hop.

The Federal Circuit Requires More Than A Commonsense Argument

So here, Petitioner appears to proffer a commonsense argument that because IEEE 1999 messages are sent downstream, a POSA could simply send Gai's dummy messages downstream too. Reply at 2-4. But the Federal Circuit has rejected similar arguments when they are unsupported by evidence and reasoned explanation. *Arendi S.A.R.L. v. Apple Inc.*, 832 F.3d 1355, 1363 (Fed. Cir. 2016) ("Although such

Petitioner Offers No Basis for POSITA to Combine the References

Record Lacks Evidence Explaining How Gai's Messages Could Be Repurposed In View Of IEEE 1999 Tear Down Messages

"It is undisputed that Gai does not disclose sending messages downstream." Sur-reply at 10 (citing POR at 13-14).

"IEEE teardown messages are transmitted to host computers. . . Gai's dummy messages are sent to other devices in the network (not host computer)" Sur-reply at 5 (citing POR at 8).

"IEEE 1999's tear down messages are sent before the spanning tree is reconfigured . . . Gai dummy messages are transmitted as part of the reconfiguration process and only transmitted once a device has already successfully transitioned to a new forwarding port." Sur-reply at 5-6 (citing POR at 8).

The Independent Claims Are Patentable

B. The Independent Claims Are Patentable

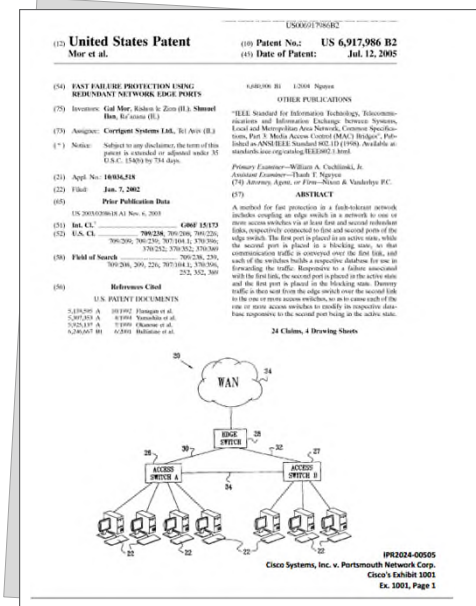
The Reply arguments apply to independent claims 1, 12, 13, and 24. Reply at 6-7. Therefore, for the reasons explained above, Petitioner failed to meet its burden of showing that Gai in view of IEEE 1999 renders the independent claims unpatentable.

***Gai* in view of IEEE 1999 does not teach sending data frames originating from an edge switch (claims 6 and 18)**

U.S. Patent No. 6,917,986 – Dependent Claims 6 and 18

6. A method according to claim 1, wherein sending the dummy traffic comprises sending data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.

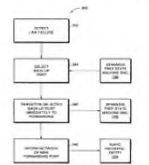
18. A device according to claim 13, wherein the dummy traffic comprises data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.



Gai Discloses Transmitting Traffic from Access Switches NOT Edge Switches

United States Patent [10] Patent Number: **6,032,194**
Gai et al. [45] Date of Patent: **Feb. 29, 2000**

[54] **METHOD AND APPARATUS FOR RAPIDLY RECONFIGURING COMPUTER NETWORKS**
 [57] Inventors: **Silvana Gai, Vigorino D'Adda, Eddy, Keith McCaughey, San Jose, Blanch G. Datt, Santa Clara, both of Calif.**
 [73] Assignee: **Cisco Technology, Inc., San Jose, Calif.**
 [21] Appl. No: **08/984,412**
 [22] Filed: **Dec. 24, 1997**
 [51] Int. Cl.⁷ **G06F 15/18, G06F 15/00**
 [52] U.S. Cl. **706/239, 706/231, 706/234, 706/230, 706/238, 706/236, 706/235, 706/237, 370/252, 379/221, 379/275, 714/39, 714/47, 360/200, 360/203**
 [56] **Field of Search** **706/234, 239, 706/231, 713/300, 360/200, 379/234, 311, 252, 400, 406, 402, 256, 379/221, 275, 714/39, 47**
 [56] **References Cited**
 U.S. PATENT DOCUMENTS
 4,022,486 5/1980 Lefkowitz et al. 770/60
 5,018,177 1/1991 Bollen et al. 770/61.9
 5,130,435 9/1992 Lampert et al. 770/60
 5,430,860 4/1995 Perreault et al. 770/61.3
 5,280,400 1/1994 Perreault et al. 770/61.3
 5,361,171 4/1995 Macdonald et al. 770/602
 5,742,094 1/1998 Gai et al. 770/601
 5,764,036 4/1998 Gai et al. 770/601
 5,766,408 4/1998 Saito et al. 770/623
 OTHER PUBLICATIONS
 S. Herwin, "Dual-Layer Spanning Tree", (A Spanning Tree Protocol for IEEE 802.3), May 14, 1994, Lucent Technologies Inc.'s Initial Disclosure of Prior Art Under Civil Court Rule 10-7, Case No. C96-20036 PH (PVT) LSI.
 M. Sedwani and D. Delaney, *Single or Multiple Filtering Database* (May 8, 1997).
 16 Claims, 8 Drawing Sheets



IPR2024-00505
 Cisco Systems, Inc. v. Portsmouth Network Corp.
 Cisco's Exhibit 1005

Rapid reconfiguration entity 234 similarly generates one or more separate multicast messages 400 for each remaining entity directly coupled to access switch 114 (i.e., entities or end stations on LANs 102-104). In each message 400, entity 234 loads SA field 414 with the address of the corresponding entity. Each of these messages are similarly forwarded by access switch 114 through the new forwarding port (i.e., port number four). In addition, entity 234 generates and transmits

No Portion Of Gai Discloses Backbone 122 Sending Dummy Messages

that are accessible to the access switches via the edge switch.” Mr. McNair states that “it would have been obvious for backbone switch 122 (i.e., the edge switch) to send dummy multicast messages and thereby originate them.” EX1003 at ¶ 179. However, no portion of Gai actually discloses this scenario, and Mr. McNair does not cite to any passage in Gai to support his argument. *Id.* In fact, Gai discloses the transition at access switch 114 from an initial root port number three to back-up root port number four and the dummy messages associated with the transmissions from the access switch. *See, e.g.*, EX1005 at 12:56–13:33.

No Explanation How Gai's Messages Could Be Modified In View Of IEEE 1999 to Meet "Data Frames" Limitation

why this disclosure satisfies these limitations. To the extent Petitioner is arguing Gai can be modified by IEEE 1999 to meet the "data frames" limitation, Petitioner failed to meet its burden of showing there was a motivation to combine (*supra* section II) and failed to show how or why Gai's dummy messages would be combined with IEEE 1999 to teach the "data frames" limitation. At bottom, Petitioner fails to meet

Ground 2

Dependent claims 4, 5, 6, 17

*Gai, IEEE 1999, and
Jain*

***Fail For The Same Reason As The
Independent Claims They Rely On And
Impermissible Hindsight Is Not A
Motivation To Combine***

Petitioner's Motivation To Combine Conclusion Is Based On Mr. McNair's Flawed Analysis

50. Mr. McNair asserts that “Gai expresses concern that messages may be misdirected or lost” and that “POSAs would have been motivated to modify Gai by incorporating Jain’s technique of updating a filtering database responsive to a link failure for the benefit of reducing the possibility of misdirected/lost data messages during the self-healing process.” EX1003 at ¶ 214. Mr. McNair specifies that the modifications to Gai “would be implemented by remapping entries from the failed port to a backup port” and that the update “would be implemented before receiving communication traffic on the backup port to reduce the possibility of the backup port misdirecting or losing received messages using outdated database information.” *Id.* But Gai already describes a solution to this problem by sending dummy multicast messages. EX1005 at 12:62–66 (“Switch 114, via rapid reconfiguration entity 234, preferably informs the other devices of its new forwarding port by transmitting dummy multicast packets through the new port, as indicated at block 348.”).

Petitioner's Motivation To Combine Conclusion Is Based On Mr. McNairs Flawed Analysis

51. Mr. McNair's proposed modifications are therefore unnecessary and taught away by Gai. While Gai does mention the risk of messages being lost or misdirected, Gai presents its own solution to avoid that outcome. For example, Gai discloses that when access switch 114 transitions from initial port number three to back-up root port number four, "the identity of the new root port (e.g., port number four) must be propagated to the other devices, such as access switches 115-116 and backbone switches 120-125 to prevent messages from being lost or misdirected." EX1005 at 12:59-62. Gai further discloses that when a dummy multicast message 400 is received at a new source port, "this new location information is entered by backbone switch 124 into its filtering database, replacing the previous information that was stored therein." *Id.* at 13:34-38. By doing so, Gai's dummy multicast message 400 "effectively apprises backbone switch 124 of the change in forwarding ports that occurred at access switch 114." *Id.* at 13:43-45. Gai's transmission of dummy multicast messages records the location information of the new root port and propagates that information to other devices in the system so that filtering databases can be updated. As a result, Gai already provides its own solution so that messages are not lost or misdirected when a switch transitions to a new port.

Petitioner Uses Impermissible Hindsight To Support A Motivation To Combine

because, as Patent Owner argued, Gai already describes a solution to the problem Petitioner raised regarding why a POSA would be motivated to modify Gai in view of IEEE 1999 and further in view of Jain. POR at 17. Petitioner ignores this and instead uses impermissible hindsight to argue “the Petition argued that Jain’s technique would have been obvious to use in the switch that sends Gai’s dummy messages – as claims 4, 5, 16, and 17 require.” Reply at 10 (emphasis added). *See*

The Reply Fails To Respond To POR's Motivation To Combine Argument

V. Gai in View of IEEE 1999 and Further in View of Jain Does Not Render Obvious Claims 4, 5, 16, and 17

Dependent claims 4, 5, 16, and 17 are valid based on Petitioner's failure to establish obviousness of the underlying independent claims from which they depend.

Reply at 16. Moreover, Petitioner failed to show there was a motivation to combine,

Conclusion

Ground 1

Independent claims 1, 12, 24 (Dependent claims 2, 7, 9-11, 13, 14, 16, 18, 19, and 21-23)

*Gai and IEEE
1999*

- No Motivation To Combine - IEEE 1999 Tear Down And Gai's Dummy Messages Function Differently
- Neither Gai or IEEE 1999 Disclose Sending Dummy Messages Downstream
- Gai Does Not Disclose The "Data Frames" Limitation Of Dependent Claims 6 and 18

Ground 2

Dependent claims 4, 5, 6, 17

*Gai, IEEE
1999, and
Jain*

- Fail For The Same Reason As The Independent Claims They Rely On
- No Motivation To Combine – Impermissible Hindsight