© ECTI-CARD 2009, May'09, Bangkok, Thailand. ISBN: 978-974-8285-62-7

การปรับเปลี่ยนระบบจัดการแฟ้มข้อมูล CODA เพื่อรองรับ IPv6

อากาศ กิตติวรเชฏฐ์
ภาควิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์
มหาวิทยาลัยเกษตรศาสตร์
nova_cengin@hotmail.com

นพปฎล พุมอาภรณ์
ภาควิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์
มหาวิทยาลัยเกษตรศาสตร์
noppapadol@hotmail.com

สุขุมาล กิติสิน
ภาควิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์
มหาวิทยาลัยเกษตรศาสตร์
sukumal.i@ku.ac.th

บทคัดย่อ

***ระบบจัดการแฟ้มข้อมูล** CODA[3] **เป็นระบบจัดการแฟ้มข้อมูล** กระจาย (Distributed File Systems)[1] **ที่มีจุดเด่นคือสามารถแก้ไขข้อมูลได้แม้จะขาดการติดต่อกับ** Server (Disconnected Operation) **และในอนาคตระบบเครือข่ายกำลังจะก้าวเข้าสู่การใช้งาน** IP version **ที่** 6 (IPv6)[2] **ซึ่งเป็นแบบแผนใหม่ที่ออกแบบมาเพื่อรองรับจำนวนผู้ใช้งานทางระบบเครือข่ายที่เพิ่มมากขึ้น ทางผู้คณะจัดทำจึงเล็งเห็นว่าหากระบบจัดการแฟ้มข้อมูล** CODA **ซึ่ง ณ ปัจจุบันทำงานอยู่บน** IPv4[5] **สามารถทำงานบนระบบ** IPv6 **ได้ก็จะสามารถขยายขอบเขตการใช้งานออกไปได้อย่างกว้างขวาง การทดสอบเพื่อหาแนวทางในการแก้ไขข้อจำกัดนี้จึงเป็นสิ่งสำคัญ**
แนวทางการปฏิบัติคือจะสร้างระบบทดสอบที่ประกอบด้วย client **และ** server **ที่สามารถทำงานได้ทั้งบนเครื่องที่ใช้** IPv4 **และ** IPv6 **แล้วทดสอบระบบจัดการแฟ้มข้อมูล** CODA **บนระบบทั้งสองแล้วหาข้อผิดพลาดเพื่อแก้ไข**
จากการทดสอบและแก้ไขขณะนี้ระบบจัดการแฟ้มข้อมูล CODA **สามารถทำการยืนยันตัวผู้ใช้งานระบบผ่านทาง** IPv6 **ได้แล้วแต่ว่ายังไม่สามารถแก้ไขแฟ้มข้อมูลของระบบได้ เนื่องจากกระบวนการอ้างอิงถึงตัว** server **ยังใช้กับ** IP address **เพื่อการติดต่อ ทางคณะผู้จัดทำเล็งเห็นว่าควรเปลี่ยนเป็นอ้างอิงเป็นใช้** domain name **ซึ่งสามารถอ้างอิงได้ทั้งบนระบบ** IPv4 **และ** IPv6 **จากการทดลองพบว่า** RPC2[6] **ซึ่งเป็น** module **สำหรับใช้ในการติดต่อระหว่าง** client **และ** server **สามารถรองรับการติดต่อแบบใช้** domain name **บน** IPv6 **ได้***

Abstract

Coda developed by the research group of M. Satyanarayanan at Carnegie Mellon University since 1987. It is a distributed file system providing flexibility to data server and network failures by using two essential methods -- disconnected operation and server replication. Nevertheless, Coda can recently be ported to many operating systems such as Linux, NetBSD, and FreeBSD etc. This notable file system is expected to become popular, widely used and full available for everyone. However Coda can be deployed on top of IPv4 networks which has been used for past twenty years and will be gradually replaced by IPv6 which can solve IPv4 address depletion. Our efforts are to port Coda to IPv6. This paper deeply represents how and where it is necessary Coda file system should be modified in order to support IPv6 and how Coda file system can be widely adopt in IPv6 world.

คำสำคัญ

CODA, IPv6, แฟ้มข้อมูล

1. บทนำ

ในปัจจุบันการแลกเปลี่ยนข้อมูลระหว่างผู้ร่วมงาน โดยทั่วไปแล้วข้อมูลต่างๆ ถูกเก็บอยู่ในรูปแบบของแฟ้มข้อมูลบนเครื่องคอมพิวเตอร์ เนื่องจากสามารถจัดเก็บข้อมูลได้เป็นจำนวนมาก สามารถเข้าและค้นหาข้อมูลที่ต้องการได้อย่างรวดเร็วและแก้ไขได้ง่าย แต่ในการทำงานร่วมกันมากกว่าหนึ่งบุคคลขึ้นไป แต่ละคนอาจจะใช้แฟ้มข้อมูลชุดเดียวกัน ดังนั้นการที่จะเก็บข้อมูลที่แต่ละคนต้องการให้แยกออกจากกันอาจจะทำให้เกิดปัญหาหลายประการอาทิเช่น การซ้ำซ้อนของข้อมูล กล่าวคือถ้ามีการแก้ไขแฟ้มข้อมูลบางส่วน แฟ้มข้อมูลที่เก็บข้อมูลชุดเดียวกัน ที่ผู้ใช้คนอื่นสำรองไปใช้งานที่เครื่องของตนจะต้องถูกแก้ไขด้วย เพื่อให้ทุกคนที่ใช้งานได้ข้อมูลที่ตรงกันจำเป็นต้องมีการติดตามว่าแฟ้มข้อมูลนั้นมีใครใช้งานบ้างแล้วตามไปแก้ไขทั้งหมด ซึ่งเกิดความยุ่งยากในการทำงาน

แต่ปัญหาที่เกิดขึ้นในระบบแฟ้มข้อมูลทั่วไปนั้น ถ้าหากส่วนผู้ให้บริการหรือ File Server เกิดปัญหาไม่สามารถติดต่อได้ Mahadev Satyanarayanan นักวิจัยจากมหาวิทยาลัย Carnegie Mellon ประเทศสหรัฐอเมริกาจึงได้มีการคิดค้นระบบจัดการแฟ้มข้อมูล Coda (Coda File System) ขึ้นมาเพื่อจัดการปัญหาการไม่สามารถติดต่อ Server ขณะทำงานได้ที่เรียกว่า Disconnected Operation เพื่อให้ผู้ใช้งานสามารถทำงานได้แม้ในขณะที่ส่วนให้บริการมีความขัดข้องได้อย่างมีประสิทธิภาพ สามารถมองแฟ้มข้อมูลที่อยู่บน Server เสมือนแฟ้มข้อมูลบนเครื่องตัวเอง และ ยังสามารถนำไปประยุกต์กับ

ระบบคอมพิวเตอร์ไร้สายแบบพกพาต่างๆ ซึ่งมีโอกาสที่จะหลุดจากการเชื่อมต่อกับส่วนให้บริการเมื่อไรก็ได้ ดังนั้นการแก้ปัญหาการไม่สามารถเชื่อมต่อกับ Server ชั่วคราวของระบบแฟ้มข้อมูล Coda จะช่วยให้คอมพิวเตอร์ไร้สายแบบพกพา สามารถติดต่อกับ Server ได้ตามที่ผู้ใช้งานต้องการ

คุณสมบัติอื่นๆของระบบแฟ้มข้อมูล Coda ยังมีความปลอดภัย เพราะก่อนการเข้าใช้งาน ผู้ใช้ต้องทำการยืนยันตนเอง (authentication) และข้อมูลที่ส่งจะถูกเข้ารหัสก่อนเพื่อป้องกันการถูกดักจับข้อมูลระหว่างทำงาน นอกจากนี้ยังมีการทำงานที่รวดเร็วเพราะระบบจะสำรองแฟ้มข้อมูลสำหรับใช้งานไว้บนเครื่องของผู้ใช้งาน ทำให้ไม่เสียเวลาในการส่งข้อมูลไปมาระหว่าง Server และ Client โดยไม่จำเป็น และยังปรับระดับการส่งข้อมูลให้เหมาะสมกับสภาพของระบบเครือข่ายในขณะนั้นได้อีกด้วย

2. ที่มาและแรงจูงใจของปัญหา

ในปัจจุบันในระบบเครือข่ายทั่วไปจะใช้โปรโตคอล IPv4 ในการอ้างอิง IP Address และมีการคาดการณ์ว่า IP Address ที่เราใช้กันในปัจจุบันนั้น อาจจะหมดลงในอนาคต จึงได้มีการนำ IPv6 ซึ่งเป็น Internet Protocol ที่สามารถรองรับผู้ใช้งานได้จำนวนมากกว่าด้วยขนาดของหน่วยความจำที่ใช้เก็บตำแหน่งที่อยู่ถึง 128 บิต (ใน IPv4 มีเพียง 32 บิต) และจะแทนที่ IPv4 ที่ใช้กันอยู่ในปัจจุบัน จึงมีความจำเป็นอย่างยิ่งที่จะต้องมีการพัฒนาระบบแฟ้มข้อมูล Coda File System ขึ้นเพื่อสนับสนุนโปรโตคอลรุ่นใหม่ที่จะนำมาใช้งานในอนาคต ดังกล่าวนี้ได้

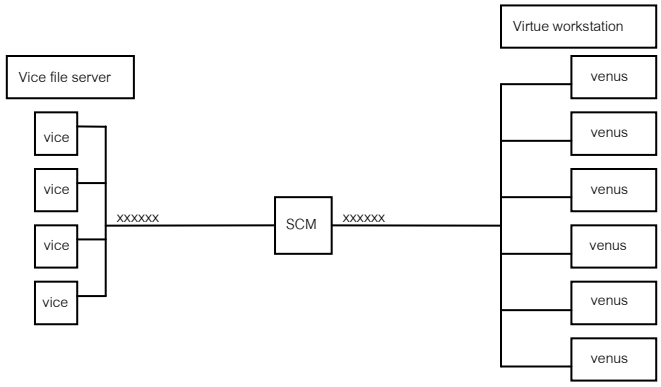
จากความนำที่กล่าวมาพบว่า Coda เป็นระบบจัดการแฟ้มข้อมูลที่มีประสิทธิภาพ เหมาะแก่การนำมาใช้สำหรับจัดการแฟ้มข้อมูลที่ใช้งานร่วมกันในองค์กร และ IPv6 ก็เป็นโปรโตคอลที่จะมีบทบาทสำคัญในโครงสร้างการสื่อสารในอนาคต แต่ Coda นั้นได้ถูกพัฒนาบน IPv4 จึงจำเป็นต้องมีการดัดแปลงเพื่อให้สามารถทำการติดต่อกับ IPv6 ได้ ทางผู้จัดทำจึงเล็งเห็นความสำคัญที่จะนำระบบแฟ้มข้อมูล Coda มาดัดแปลงให้สามารถทำงานได้บน IPv6 อย่างมีประสิทธิภาพในระบบเครือข่ายของอนาคตที่กำลังจะมาถึงนี้

3. งานและทฤษฎีที่เกี่ยวข้อง

3.1 Coda File System

ระบบแฟ้มข้อมูล Coda[15] ถูกพัฒนาโดย Carnegie Mellon University (CMU) ในปี 1990 และปัจจุบันได้รวมเข้ากับระบบปฏิบัติการ UNIX-base เช่น Linux ซึ่งระบบแฟ้มข้อมูล Coda จะอนุญาตให้ผู้ใช้งานสามารถติดต่อกับแฟ้มข้อมูลได้แม้จะขาดการติดต่อกับ server ไปแล้วก็ตาม นอกจากนี้ยังมีความยืดหยุ่น ปลอดภัย และสามารถใช้งานแฟ้มข้อมูลบน server เสมือนกับแฟ้มข้อมูลนั้นอยู่บนเครื่อง client เอง

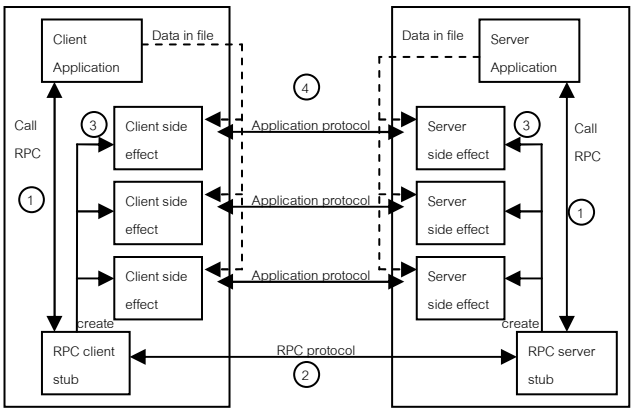
ระบบแฟ้มข้อมูล Coda เป็นเสมือนกับรุ่นที่สองของ ระบบแฟ้มข้อมูล Andrew (Andrew File Systems หรือ AFS[4]) ซึ่งแบ่งระบบออกเป็นสองกลุ่ม กลุ่มหนึ่งเป็นกลุ่มเล็กๆ ที่ทำหน้าที่เป็น server ของระบบชื่อว่า Vice โดยมี SCM (the master coda server) เป็นศูนย์กลางการจัดการกับ client รวมไปถึงการสำรองข้อมูลเพื่อให้สามารถกู้คืนได้ ส่วนอีกกลุ่มเป็นกลุ่มของ client เรียกว่า Virtue workstations จะเป็นส่วนให้ผู้ใช้ติดต่อเข้าไปใช้งานระบบได้ ดังรูปที่ 1 แสดงโครงสร้างของ ระบบจัดการแฟ้มข้อมูล Coda



รูปที่ 1 โครงสร้าง ระบบจัดการแฟ้มข้อมูล Coda

แต่ละ Virtue workstation จะมี user-level process ที่ชื่อว่า Venus มีหน้าที่ติดต่อแฟ้มข้อมูลที่ถูกเก็บไว้ใน Vice และจะอนุญาตให้ผู้ใช้สามารถติดต่อกับระบบได้ แม้จะขาดการติดต่อกับ server ไปแล้วก็ตาม นอกจากนี้ ยังคอยฟังการร้องขอข้อมูลภายในแฟ้มข้อมูลจากผู้ใช้งาน โดยเมื่อผู้ใช้งานเข้าไป directory /coda จะมีการติดต่อไปยัง server เพื่อขอข้อมูลภายในแฟ้มข้อมูล และเมื่อใช้คำสั่ง clog แล้วผู้ใช้งานจะสามารถแก้ไขข้อมูลภายในแฟ้มข้อมูลได้ แต่ถ้ายังไม่ได้ผ่าน

การใช้คำสั่ง แฟ้มข้อมูลจะเป็นแบบ Read Only หลังจากนั้นเมื่อมีการแก้ไขข้อมูลภายในระบบ venus จะทำการส่งข้อความที่แก้ไขกลับไปที่ server ผ่านทาง RPC2 ดังรูปที่ 2 แสดงการใช้งาน RPC2



รูปที่ 2 การใช้งาน RPC2

จากรูปที่ 2 ในขั้นแรกจะมีการสร้าง RPC2 module ขึ้นมาทั้งสองฝั่งเพื่อรอการสั่งการจากผู้ใช้งานระบบ เมื่อผู้ใช้งานต้องการติดต่อกับ server แบบเรียกใช้งานคำสั่งบน server ตามปกติ (Remote Procedure Call) ซึ่งการส่งข้อมูลกลับมาเป็น data type หรือ data structure ที่ทราบขนาดแน่นอน อย่างเช่นคำสั่ง clog ที่จะส่งผลกลับเพียงสถานะการยืนยันตัวผู้ใช้งานว่าถูกต้องหรือไม่ RPC2 ของ client จะติดต่อกับ server ผ่าน RPC Protocol เพื่อสั่งการและส่งผลกลับตามปกติตามหมายเลข 2 แต่หากผู้ใช้งานต้องการข้อมูลขนาดใหญ่ RPC2 ของ client ก็จะติดต่อกับ server แล้วทำการสร้าง side effect ดังหมายเลข 3 ซึ่ง side effect นั้นเป็นช่องสำหรับการส่งข้อมูลโดยสามารถเลือกได้ว่าจะใช้ protocol ใดในการส่งซึ่ง โดยปกติ RPC2 จะใช้ UDP เมื่อสร้าง side effect รอทั้ง 2 ฝั่งแล้วจะเริ่มส่งข้อมูลกันผ่าน protocol ที่กำหนดดังหมายเลข 4 เมื่อส่งข้อมูลครบก็จะปิด side effect นั้นและรอรับคำสั่งต่อไป

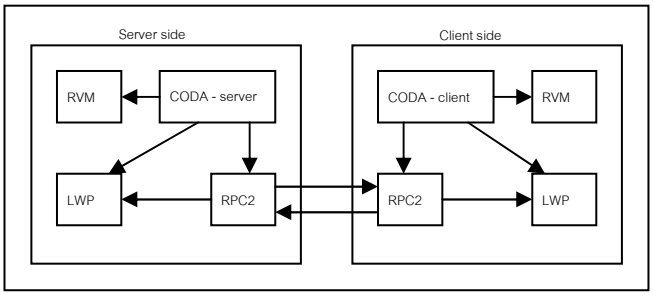
ความสามารถอื่นของ RPC2 คือ สามารถทำ multicast ได้ โดยเมื่อ client ทำการร้องขอข้อมูลไปใช้งาน Coda จะสร้างสำเนาไว้ในเครื่องของตนเองเก็บไว้ เมื่อมีการแก้ไขแฟ้มข้อมูลจะส่งกลับไปบอก server และ server จะ multicast ไปบอกกับ client อื่นที่ใช้งานข้อมูลเดียวกันว่า ข้อมูลชุดนี้มีการแก้ไข ถ้าหาก client เหล่านั้นยังติดต่อกับ server อยู่ จะสามารถรับข้อมูลและทำการแก้ไขสำเนาของไฟล์นั้นบนเครื่องตนเองและตอบกลับไปว่าได้รับการแก้ไขแล้ว แต่หาก client เครื่องใดไม่สามารถติดต่อกับ server ได้ในขณะนั้น server จะจดจำไว้ว่า client เครื่องนั้นยังไม่ได้รับข้อมูลการเปลี่ยนแปลงและเมื่อใดที่ client เครื่องนั้นได้เข้าติดต่อกับ server จึงจะทำการส่งข้อมูลการเปลี่ยนแปลงกลับไปอย่างเหมาะสม

4. รายละเอียดการพัฒนา

ในการดำเนินงานจะต้องสร้างเครือข่ายที่สามารถทำงานได้ทั้งบนระบบ IPv4 และ IPv6 สำหรับทดสอบระบบแฟ้มข้อมูล Coda แล้วจึงทดสอบหาข้อผิดพลาดโดยดูจาก output ของการทำงาน และ Log File นอกจากนี้ยังใช้โปรแกรมตรวจสอบความผิดพลาดในระหว่างการทำงานของระบบ (Debugger) เพื่อหาว่ามีความผิดปกติที่ส่วนใด จึงเข้าไปหาสาเหตุและแก้ไขพร้อมทั้งรายงานและสรุปผล

4.1 ภาพรวมของระบบ

ระบบแฟ้มข้อมูล Coda แบ่งออกเป็นส่วนย่อยเพื่อทำงานในส่วนต่างๆ คือ RVM[8], RPC2, LWP และตัว Coda เอง นอกจากนี้ Coda ยังแบ่งออกเป็น 2 ฝั่งคือ client กับ server ในการดัดแปลงระบบแฟ้มข้อมูล Coda ให้ทำงานได้บนระบบ IPv6 นั้นจำเป็นต้องเข้าใจการทำงานและการติดต่อระหว่างกันในส่วนต่างๆ เพื่อจะตรวจสอบและแก้ไขการทำงานได้ตรงจุด ภาพรวมของการติดต่อระหว่างส่วนย่อยต่างๆ ของ Coda แสดงในรูปที่ 3



รูปที่ 3 การติดต่อระหว่างส่วนย่อยภายในระบบแฟ้มข้อมูล CODA

4.2 การออกแบบและพัฒนาระบบ

4.2.1 LWP

Light-weight Process หรือ LWP เป็นส่วนที่ Coda และ RPC2 ใช้ในการจัดการเรื่อง Thread สำหรับการทำงานที่ต้องทำแบบคู่ขนาน (concurrent)

4.2.2 RVM

Recoverable Virtual Machine หรือ RVM ทำหน้าที่สำหรับกู้คืนข้อมูลในกรณีที่แฟ้มข้อมูลของ client นั้นไม่ตรงกัน

4.2.3 RPC2

RPC[7] มาจากคำว่า remote procedure call ซึ่งเป็นกลวิธีในการติดต่อกันระหว่าง Server และ Client โดยที่ผู้ติดต่อนั้นมีการเคลื่อนที่และระยะการติดต่อที่ขยายอยู่ตลอดเวลาซึ่งกระบวนการติดต่อนี้จะทำงานอยู่บน IP และ UDP Protocol ในระบบ RPC ชุดเริ่มแรกนั้นได้ถูกนำมาใช้ในการติดต่อสื่อสารระหว่าง Server และ Client ในระบบ Andrew File System หรือ AFS ซึ่งเป็นแม่แบบของ Coda File System จึงจำเป็นต้องมีการ implement RPC มาเป็น RPC2 นั่นเอง

4.2.4 CODA

ระบบจัดการแฟ้มข้อมูล CODA ถูกแบ่งออกเป็น 2 ส่วนใหญ่ๆ และมีชื่อเรียกเฉพาะ ดังนี้

4.2.4.1 Vice

เป็นชื่อของเครื่อง server ที่จะเป็นส่วนในการจัดการเรื่องการเก็บข้อมูลในแฟ้มข้อมูล การแบ่งสิทธิของผู้ใช้งานระบบ ฯลฯ

4.2.4.2 Venus

เป็นชื่อของ process ในฝั่ง client ซึ่งเริ่มต้นด้วยการสั่ง ./usr/local/etc/venus.init start จะคอยรับคำสั่ง clog สำหรับการยืนยันตัวตนผู้ใช้งานระบบ ถ้าหากผ่านจึงจะทำการเชื่อมต่อกับ vice server และคอยทำการ update แฟ้มข้อมูลไปที่ vice server โดยอัตโนมัติ

จากการทำงานดังกล่าว จึงสามารถสรุปได้ว่า การจะดัดแปลงการทำงานของระบบแฟ้มข้อมูล CODA ให้สามารถใช้งานได้บนระบบ IPv6 นั้นน่าจะเกี่ยวข้องกับการทำงานของส่วนที่ใช้ในการติดต่อระหว่าง server กับ server หรือ server กับ client นั่นคือคำสั่ง clog ในฝั่ง client และ auth2.init update.init codasrv.init ในฝั่งของ server โดยอาจจะมีส่วนของการเก็บข้อมูลเกี่ยวกับตำแหน่งและชื่อของเครื่องที่อยู่ในกลุ่ม server

5. การทดสอบการใช้งาน

5.1 ผลการทดสอบและการวิจารณ์ผล

5.1.1 เริ่มต้น IPv6 และทดสอบ

หลังจากการติดตั้งระบบ IPv6 โดยการแก้ไขข้อมูลในแฟ้มข้อมูล `/etc/sysconfig/network` และแฟ้มข้อมูล `/etc/sysconfig/network-scripts/ifcfg-eth0` แล้วทำการเริ่มการทำงานของ service network ใหม่ด้วยคำสั่ง `service network restart` จากนั้นจึงทดสอบด้วยคำสั่ง `dig` เพื่อ resolve domain name ที่กำหนดให้ด้วย DNS และจะตอบกลับมาว่า domain name นั้นได้ถูกกำหนดให้กับ IP address ใด หากค้นหาไม่พบจะได้คำตอบว่าไม่สามารถค้นหาได้ โดยทดสอบด้วยคำสั่ง `dig server.ipv6.net AAAA` (`AAAA` เป็นส่วนเสริมเพื่อบอกว่าเป็นการ resolve แบบ IPv6) ได้ผลลัพธ์กลับมาว่า domain name ชื่อ `server.ipv6.net` ที่อยู่ในหมวด `AAAA` มี IP Address เป็น `2002:9e6c:153a::1`

ต่อมาจึงทดสอบด้วยคำสั่ง `ping6` เพื่อทดลองส่งข้อมูลไปยังปลายทางที่กำหนดบนระบบ IPv6 แล้ววัดคุณภาพในการส่ง โดยครั้งแรก การใช้คำสั่ง `ping6` จาก client ไปยัง server พบว่าไม่สามารถส่ง packet จาก client ไปยัง server ได้ทาง คณะผู้จัดทำจึงลองใช้คำสั่ง `ping6` จาก server เข้าสู่เครื่อง server ปรากฏว่า `ping6` สามารถทำงานได้ตามปกติแล้ว คณะผู้จัดทำจึงทดลอง `ping6` แบบนี้อีกหลายครั้ง ได้ผลเช่นเดิม คณะผู้จัดทำคาดว่าน่าจะเกิดจากการที่ระบบเครือข่ายที่คณะผู้จัดทำใช้ไม่ได้เป็น IPv6 แท้ IP Address ที่กำหนดให้กับ domain name ของ client และ server ทางคณะผู้จัดการใช้ IP Address ของ Tunnel 6 to 4 (tun6to4) แทนเพื่อให้สามารถส่ง packet จากเครื่องที่ใช้ IPv6 ผ่านเข้าไปในเครือข่ายที่เป็น IPv4 ได้ มีผลทำให้การจะส่ง packet

ออกจาก tunnel นี้จะต้องมี packet เข้ามาก่อนจึงจะสามารถทำได้

5.1.2 ทดสอบ Coda client

- เข้าสู่ directory `/coda` ว่าตอนที่ยังไม่ได้ทำการใดๆกับระบบจัดการแฟ้มข้อมูล CODA เลย จะสามารถกระทำการใดกับแฟ้มข้อมูลได้บ้าง ได้ผลดังนี้

```
   [root@client ~]# cd /coda
   [root@client coda]# ls
   NOT_REALLY_CODA
   [root@client coda]# cd server.codomo.net
   -bash: cd: server.codomo.net: No such
file or directory
```

- ใช้คำสั่ง `venus-setup` ตามด้วย domain name ที่เป็น IPv6 ของ server เป็นการบอก venus ว่าจะเรียกใช้แฟ้มข้อมูลจาก server ด้วยการติดต่อแบบ IPv6 ได้ผลดังนี้

```
   [root@client /]# venus-setup
server.ipv6.net
   == Loading coda kernel module.
   /etc/services already has new services
registered! Good.
   /etc/services ready for Coda
```

- ใช้คำสั่ง `./usr/local/etc/venus.init start` แล้วกลับไปทำงานกับแฟ้มข้อมูลอีกครั้ง ได้ผลดังนี้

```
   [root@client /]#
./usr/local/etc/venus.init start
   Starting venus:
/usr/coda/venus.cache/pid exists, venus
already running?.
   done.
   [root@client /]# cd coda
   [root@client coda]# cd server.codomo.net
   [root@client server.codomo.net]# ls
   hello  hello_v2
   [root@client server.codomo.net]# vi
hello
   <file still read only>
```

ในขั้นนี้สามารถเข้าถึงแฟ้มข้อมูลที่เป็นสำเนาบนเครื่องได้แต่หากต้องการแก้ไขนั้นจำเป็นต้องผ่านการใช้คำสั่ง `clog` ก่อน

- ทำการยืนยันตัวตนผู้ใช้งานระบบด้วยคำสั่ง `clog` แล้วกลับไปทำงานกับแฟ้มข้อมูล ได้ผลดังนี้

```
   [root@client /]# clog admin
   username: admin@server.ipv6.net
   Unable to resolve addresses for Coda
auth2 servers in realm 'server.ipv6.net'
   Password:
   Invalid login (RPC2_FAIL (F)).
   [root@client /]# cd coda
   [root@client coda]# cd
server.codomo.net/
   [root@client server.codomo.net]# ls
   hello  hello_v2
   [root@client server.codomo.net]# vi
hello
   <file still read only>
```

จะเห็นได้ว่าไม่สามารถทำการ login ด้วยคำสั่ง clog ได้

5.1.3 ตรวจสอบ source code และ แก้ไข

เมื่อเข้าไปตรวจสอบ source code ของ function `U_GetAuthServers`(auser.c) ได้พบว่ามีการเรียกใช้ function `getaddrinfo` ซึ่งมี parameter ที่ชื่อว่า `structure addrinfo *hints` มีหน้าที่เป็นตัวกำหนดว่า `addrinfo` ที่จะได้เป็นผลลัพธ์จาก function นี้ จะมีรูปแบบใด ภายใน `addrinfo` จะมีสมาชิกตัวหนึ่งที่ชื่อ `ai_family` เป็นตัวบอกว่า `addrinfo` นี้อยู่ใน IP protocol ใดซึ่งมีสามแบบ คือ `PF_INET` `PF_INET6` และ `PF_UNSPEC` ซึ่งหมายถึงการทำงานบน IP protocol version 4 version 6 และ ไม่เจาะจงตามลำดับ

`struct addrinfo *hints` ถูกสร้างและกำหนดค่าที่ function `GetRealmServers` โดยให้ `ai_family` เป็น `PF_INET` ดังนั้น addrinfo ที่ได้ จะมีเฉพาะบน IPv4 เท่านั้น

คณะผู้จัดทำจึงทำการแก้ไขโดยให้ `ai_family` เป็น `PF_INET` ก่อน ถ้าหาก ผลลัพธ์จาก `coda_getaddrinfo` เป็น NULL แล้วจึง เปลี่ยนค่า `ai_family` เป็น `PF_INET6` แล้วก็ส่งเข้าไปใน `coda_getaddrinfo` อีกรอบ ผลการทำงานคือ `clog` ผ่าน และสามารถเข้าไปใน `directory /coda` ได้ และมองเห็น directory ที่เป็นชื่อของ domain name แบบ IPv4 และ IPv6 ของ server แต่ directory ที่เป็น domain name แบบ IPv6 นั้นเป็น link error ส่วน แบบ IPv4 นั้น สามารถเข้าไปอ่านได้ แต่ไม่สามารถแก้ไขได้

6. บทสรุป

6.1 ผลการปฏิบัติงานในส่วนของ Client

ในขณะนี้ client สามารถทำคำสั่ง `clog` ได้อย่างถูกต้องแล้ว แต่ในส่วนของ venus ที่ทำการติดต่อกับ server หลังจากทำการ `clog` แล้วนั้น ยังไม่สมบูรณ์โดย RPC2 นั้นสามารถทำการ bind ระหว่าง client กับ server ได้ 3 วิธีคือ

RPC2_HOSTBYNAME RPC2_HOSTBYADDRINFO และ RPC2_HOSTBYINETADDR แต่การติดต่อกับ server บน IPv6 โดยใช้ RPC2 นั้นที่เป็นไปได้มี 2 วิธีคือ RPC2_HOSTBYNAME และ RPC2_HOSTBYADDRINFO แต่ CODA เลือกใช้งาน RPC2_HOSTBYINETADDR ทางทีมงานจึงทดลองเปลี่ยนเป็นใช้งานแบบ RPC2_HOSTBYNAME ซึ่งบน IPV4 นั้นยังไม่เกิดข้อผิดพลาดใดๆ แต่บน IPv6 นั้นได้รับการตอบกลับจาก RPC2 Layer มาเป็น RPC2_NOBINDING จึงต้องทำการตรวจสอบแก้ไขต่อไป

6.2 ผลการปฏิบัติการในส่วน Server

ในส่วนของ Coda นั้น มีฟังค์ชั่นในส่วนของ Server ที่เรียกว่า ViceGetVolumeLocation(IN VolumeId volid, OUT RPC2_BoundedBS HostPort); ซึ่งถูกเรียกใช้ในกรณีที่คำร้องขอของฝั่ง Client ในการเข้าหาข้อมูลในส่วนของ replicated volume โดยหลังจากการทำงาน client จะทำการค้นหาและได้ลำดับรายการข้อมูลของ volume ในลักษณะ IPv4 หรือ hostname จากนั้นจะทำการสำรองเก็บข้อมูลไว้ที่ไฟล์ /vice/db/servers โดยเราสามารถใช้ hostname แทนได้ทั้ง IPv4 และ IPv6 แสดงให้เห็นว่า Vice Server สามารถหาและจัดเก็บ HostEntry, ClientEntry และ HostAddress ตาม Server แต่ละตัวใน SCM และ Venus Request จาก Client แต่ละคนที่อยู่ในรูปแบบ IPv6 ได้อย่างสมบูรณ์ หากแต่ปัญหาที่ยังเกิดขึ้นมาจากการที่ RPC2_GetRequest ไม่สามารถรองรับและ resolve IPv6 ใน filter ที่ Coda File Server ต้องการได้ซึ่งเป็นปัญหาที่รอการตรวจสอบต่อไป

7. กิตติกรรมประกาศ

โครงการนี้สามารถสำเร็จลุล่วงไปได้ด้วยความช่วยเหลือของ Rod Van Meter และ Jan Harkes ผู้ให้คำปรึกษาแนวทางการแก้ไขในการพัฒนาระบบบน IPv6 และขอขอบคุณทีมผู้พัฒนาระบบแฟ้มข้อมูล Coda แห่งมหาวิทยาลัย Carnegie Mellon ประเทศสหรัฐอเมริกาที่เปิดโอกาสให้มีการพัฒนาแฟ้มข้อมูลตัวนี้เพื่อให้เกิดประโยชน์ และเป็นรากฐานที่สำคัญในการวิจัยและพัฒนาระบบแฟ้มข้อมูลที่มีประสิทธิภาพต่อไปในอนาคต

8. เอกสารอ้างอิง

[1] Silberschatz, Galvin (1994). Operating System concepts, chapter 17 Distributed file systems. Addison-Wesley Publishing Company. ISBN 0-201-59292-4

[2] RFC 1752 (http://tools.ietf.org/html/rfc1752): The Recommendation for the IP Next Generation Protocol

[3] Andrew S. Tanenbaum, Maarten van Steen, Distributed Systems: Principles and Paradigms, page 604 – 623.

[4] Howard, J.H., Kazar, M.L., Nichols, S.G., Nichols, D.A., Satyanarayanan, M., Sidebotham, R.N., & West, M.J. (February 1988). "Scale and Performance in a Distributed File System". ACM Transactions on Computer Systems 6 (1): 51-81.

[5] RFC 791 (http://tools.ietf.org/html/rfc791) : Internet Protocol

[6] M. Satyanarayanan (Editor) Richard Draves, James Kistler, Anders Klemets, Qi Lu, Lily Mummert, David Nichols, Larry Raper, Gowthami Rajendran, Jonathan Rosenberg, Ellen Siegel, “RPC2 User Guide and Reference Manual” available at http://www.coda.cs.cmu.edu/doc/html/rpc2_manual.html

[7] RFC 1057 (http://tools.ietf.org/html/rfc1057) - Specifies version 1 of ONC RPC

[8] Henry M. Mashburn, Mahadev Satyanarayanan, David Steere, Yui W. Lee, School of Computer Science, CMU, “RVM: Recoverable Virtual Memory, Release 1.3” http://www.coda.cs.cmu.edu/doc/html/rvm_manual.html