

## อัปเดตประสิทธิภาพ Apache ให้แรงด้วยการใช้ Reverse Proxies - ฉบับที่ 132 พฤศจิกายน 2006

- 
- อ่าน 9438 ครั้ง

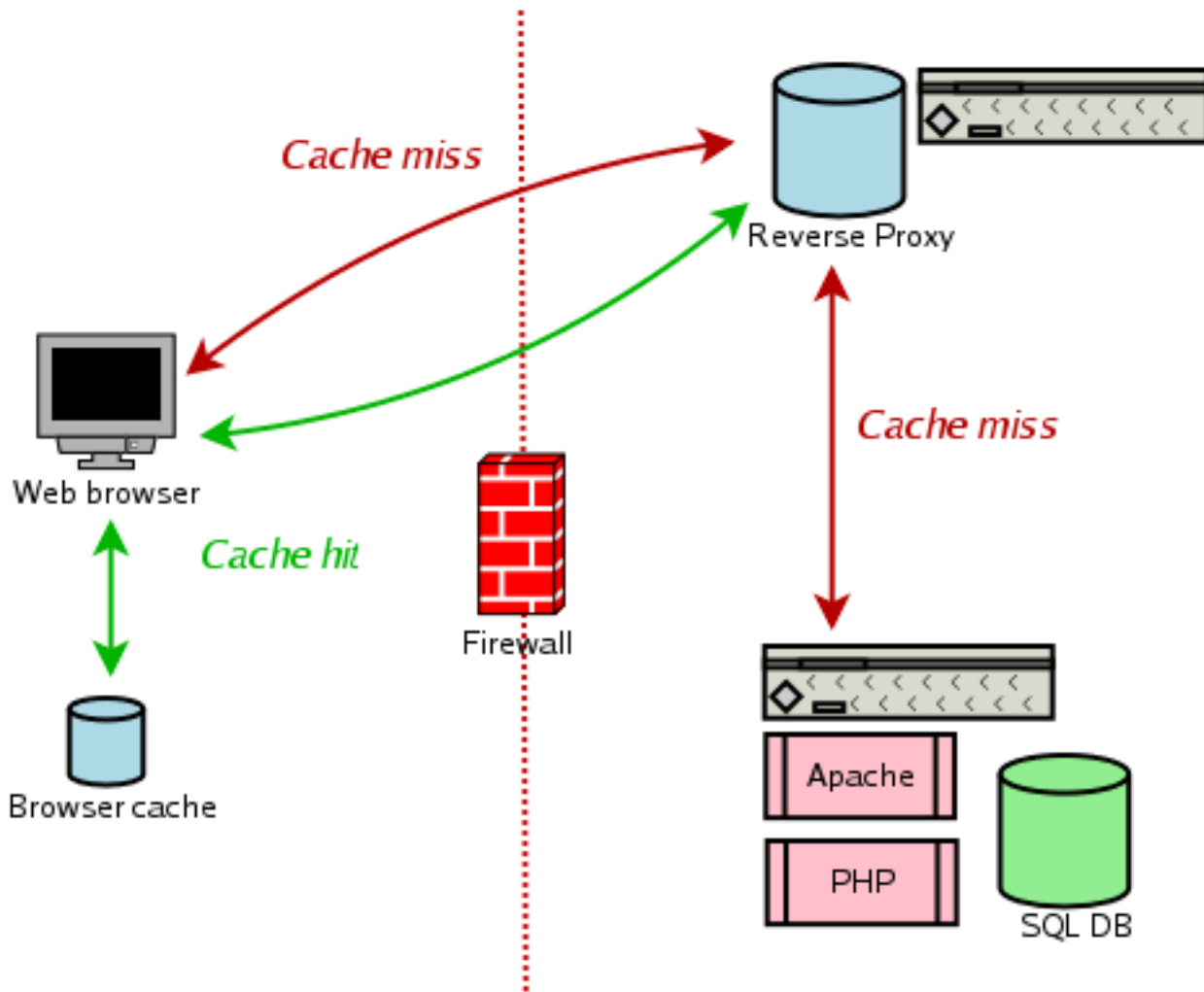
### อัปเดตประสิทธิภาพ Apache ให้แรงด้วยการใช้ Reverse Proxies

โดย René Pfeiffer และ pooz

กาลครั้งหนึ่งแต่ไม่นานซักเท่าไร, เมื่อครั้งเว็บเซิร์ฟเวอร์ผู้เดียวดายยังคงเศร้าหมอง. เว็บเบราว์เซอร์สุดคณานับ ต่างมุ่งโอบล้อมพอร์ตของมัน แบนด์วิทกำลังอ่อนล้า; ส่วนประมวลผลกลางต่างกำลังวุ่น; ฐานข้อมูลก็กำลังครวญคราง. หัวหน้าฝ่ายสารสนเทศได้เข้าพบ Pooz กับผม, ถามถึงการแก้ไข. ปรับปรุงทั้งฮาร์ดแวร์หรือแม้แต่การเชื่อมต่อเครือข่ายอินเทอร์เน็ตก็ไม่ใช่ทางเลือก, ดังนั้นเราจึงลองหาทางว่าอะไรที่เราควรทำ - ทำแค่ขการช่วยชีวิต!

### แคชอยู่ทุกหนแห่งที่คุณไป

คอมพิวเตอร์ทุกเครื่องต่างอยู่บนการแคช. ส่วนประมวลผลกลางของคุณก็หนึ่งละ, ดิสก์ไดร์ฟของคุณ, ระบบปฏิบัติการของคุณ, การ์ดจอของคุณ, หรือแม้กระทั่งเว็บเบราว์เซอร์ของคุณด้วย. แคชถูกออกแบบมาเพื่อเก็บรักษาสำเนาของข้อมูลที่มีการเข้าถึงบ่อย ๆ. แคชของส่วนประมวลผลกลางสามารถเก็บรักษาทั้งคำสั่งและข้อมูลได้. แทนที่จะเข้าถึงข้อมูลจากหน่วยความจำของระบบ เพื่ออ่านคำสั่งต่อไปหรือชิ้นส่วนของข้อมูล, มันจำทำการอ่านจากแคชแทน. เว็บเบราว์เซอร์ก็เป็นอีกตัวหนึ่งที่น่าสนใจมาก ในการแคชไฟล์อย่างรูปภาพ สไลด์ซีท, เอกสาร, และสิ่งอื่นที่ใกล้เคียง. ซึ่งจะช่วยให้การท่องเว็บรวดเร็วขึ้น. เนื่องจากรูปแบบขององค์ประกอบในหน้าเว็บต่างก็คล้าย ๆ กัน. แทนที่จะดาวน์โหลดรูปภาพหรือไฟล์เดิมซ้ำอีกรอบ มันจะใช้ข้อมูลที่พบในแคช ซึ่งเป็นจริงอย่างยิ่ง ถ้าคุณดูหน้าที่สร้างขึ้นจากระบบจัดการเนื้อหา (CMS). ตอนนี้, ถ้าเราสามารถหาทางบอกกับเว็บเบราว์เซอร์ว่า สำเนาที่แคชนั้นถูกต้อง, ดังนั้นเราจะสามารถประหยัดแบนด์วิทของเว็บเซิร์ฟเวอร์ได้. ในกรณี CMS ของเรา, ซึ่งใช้ Typo3, เราสามารถประหยัดทั้งเวลาทำงานของ CPU และการเข้าถึงข้อมูล, เพียงแต่เราสามารถส่งเวลาหมดอายุของ HTML ที่ถูกสร้างขึ้นได้อย่างถูกต้อง. คุณสามารถเพิ่มแคชระหว่างเว็บเบราว์เซอร์และเครื่องแม่ข่ายของคุณ, เพื่อลดการร้องขอจากเครื่องแม่ข่ายในครั้งต่อไป ซึ่งแคชนี้เรียกว่า *reverse proxy*, บางครั้งเรียก *gateway* หรือ *surrogate cache*. ฟร็อกซีดั้งเดิมทำงานเพื่อลูกข่าย, แต่ reverse proxy ทำงานเพื่อเครื่องแม่ข่ายแทน. ฟร็อกซีนี้ก็มีฮาร์ดดิสและหน่วยความจำแคชเช่นเดียวกัน, ซึ่งสามารถถูกใช้เพื่อเก็บข้อมูลเนื้อหาจากเครื่องแม่ข่าย Apache. ภาพข้างล่างนี้ แสดงให้เห็นว่า แคชอยู่ที่ไหน และกำลังทำอะไรอยู่.



เส้นสีเขียวที่เขียนว่า *cache hits*. เอกสารที่แคชร้องขอถูกต้อง (นั่นคือ ยังไม่หมดอายุ) ซึ่งถูกพบในแคช และสามารถคัดลอกได้จากที่นี่. การร้องขอต่าง ๆ ส่วนมากจะไม่ต้องเข้าถึงเครื่องเว็บแม่ข่าย. มีเพียงเครื่องลูกข่ายบางเครื่องอาจถามกับเครื่องแม่ข่าย ถ้าเนื้อหาได้ถูกเปลี่ยนแปลง แต่คำถามสั้น ๆ นี้ ก็ไม่ได้สร้างการจราจรที่มากมายนัก. เว็บเซิร์ฟเวอร์อาจจะตอบด้วย ส่วนหัว "HTTP/1.x 304 Not Modified" และไม่ต้องใส่ข้อมูลตาม. เส้นแดงที่เขียนว่า *Cache Miss (cache misses)*. เกิดเมื่อการที่พบแคชเกิดขึ้น เมื่อแคชไม่พบเป้าหมายที่ถูกร้องขอ จะทำการร้องขอจากเครื่องแม่ข่ายที่ถูกต้อง จากนั้นจำเป็นเข้าสู่ดีสก์หรือหน่วยความจำ เพื่อให้บริการกับเครื่องลูกข่ายต่อไป. เมื่อใดก็ตามที่การร้องขอถูกส่งออกไปยังแคช สำเนานี้จะถูกใช้ไปจนกว่ายังถูกต้องอยู่.

### ส่วนหัวที่ควบคุมแคช

แคชรู้ได้อย่างไรว่า เมื่อใดควรใช้สำเนาที่เครื่อง หรือเมื่อใดควรร้องขอต่อเครื่องแม่ข่าย อืม, มันก็แล้วแต่. แคชของเว็บเบราว์เซอร์จะดูข้อความจากเครื่องเว็บแม่ข่าย. เครื่องแม่ข่ายอาจจะใช้ส่วนหัว ที่ควบคุมแคชในการบอกกล่าว. มาดูตัวอย่างกัน. การร้องขอ "GET <http://www.luchs.at/linuxgazette/index.html> [1] HTTP/1.1" จะได้ข้อมูลส่วนหัว HTTP ที่มีหน้าตาแบบนี้.

```
HTTP/1.x 200 OK
Date: Tue, 03 Oct 2006 10:24:35 GMT
Server: Apache
Last-Modified: Mon, 02 Oct 2006 02:04:36 GMT
Etag: "e324ac5-6d7d5500"
Accept-Ranges: bytes
Cache-Control: max-age=142800
Expires: Thu, 05 Oct 2006 02:04:36 GMT
```

```
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 3028
Content-Type: text/html; charset=ISO-8859-1
X-Cache: MISS from bazaar.office.lan
X-Cache-Lookup: MISS from bazaar.office.lan:3128
Via: 1.0 bazaar.office.lan:3128 (squid/2.6.STABLE1)
Proxy-Connection: keep-alive
```

เครื่องแม่ข่ายจะให้เอกสาร HTML กับคุณ. ซึ่งส่วนหัวของ HTTP จะประกอบด้วยส่วนข้างล่างนี้ :

- *Last-Modified*: บ่งบอกว่าเนื้อหาถูกแก้ไขล่าสุดเมื่อใด.
- *Cache-Control*: บอกว่าทุก ๆ แคลระหว่างเครื่องแม่ข่ายและเว็บเบราว์เซอร์ จะถูกทำการแคชเป็นเวลา 142800 วินาที.
- *Expires*: บอกกับทุกแคชถึงวันเวลาที่แน่นอนที่แคชหมดอายุ.

*Cache-Control*: จะดีกว่า *Expires*: เนื่องจากอย่างหลังเครื่องจำเป็นต้องปรับเวลาให้สอดคล้องกับต้นฉบับ *Cache-Control*: จะใช้ได้ทั่วไปกว่า เพียงแต่ใช้ได้เฉพาะ HTTP1.1 เท่านั้น. มีข้อมูลบางอย่างที่ถูกรวมเข้าไปโดยที่ไม่ได้ถูกส่งจากเครื่อง Apache. คือส่วนหัว HTTP สี่ส่วนท้ายสุด ถูกใส่เข้าไปโดย Squid proxy ในที่ทำงานของเรา. ซึ่งบอกว่าแคชไม่พบสิ่งที่เราค้นหา.

## การปรับแต่งแคชด้านเครื่องแม่ข่าย

เอาละมาดูเครื่องแม่ข่ายของเรา, แล้วมาดูว่าเราจะปรับแต่งอะไรกันบ้าง.

### mod\_expires ของ Apache

ถึงแม้ว่า *Cache-Control*: จะดีกว่า, เราเริ่มจากการหาทางสร้างส่วนหัว *Expires*: สำหรับเนื้อหา. โปรแกรมเว็บแม่ข่าย Apache มีโมดูลที่ทำหน้าที่นี้อยู่แล้วเรียกว่า mod\_expires. ซึ่งดีสทริบิวชันส่วนใหญ่จะรวมไว้ใน Apache อยู่แล้ว. คุณสามารถคอมไพล์มันเป็นโมดูลและใส่มันเข้าไปหลังจากติดตั้ง Apache แล้ว. อีกทางหนึ่ง, คุณสามารถที่จะสร้างส่วนหัว *Expires*: , ทั้งในไฟล์ปรับแต่งหลักหรือแต่ละโฮสต์เสมือนก็ได้. ตัวอย่างการติดตั้งจะหน้าคล้าย ๆ อย่างนี้ (สำหรับ Apache 2.0.x):

```
<IfModule mod_expires.c>
    ExpiresActive On
    ExpiresByType text/html "modification plus 3 days"
    ExpiresByType text/xml "modification plus 3 days"
    ExpiresByType image/gif "access plus 4 weeks"
    ExpiresByType image/jpg "access plus 4 weeks"
    ExpiresByType image/png "access plus 4 weeks"
    ExpiresByType video/quicktime "access plus 2 months"
    ExpiresByType audio/mpeg "access plus 2 months"
    ExpiresByType application/pdf "modification plus 2 months"
    ExpiresByType application/ps "modification plus 2 months"
    ExpiresByType application/xml "modification plus 2 weeks"
</IfModule>
```

จะคำนวณและใส่ค่าส่วนหัว *Cache-Control*: ตามความเหมาะสมโดยอัตโนมัติ, ซึ่งเป็นการดี. คุณสามารถใช้ "modification plus ..." หรือ "access plus ...". "modification" ให้ทำงานเพียงไฟล์ที่ Apache อ่านจากดิสก์. ซึ่งหมายความว่า คุณสามารถใช้ "access" ถ้าคุณต้องการตั้งค่าหมดอายุส่วนหัวแบบอัตโนมัติ สำหรับการสร้างเนื้อหา แบบเปลี่ยนแปลงตลอดเวลา ได้อย่างง่ายดาย. **โปรดระวัง!!** ถึงแม้ว่าสคริป CGI ต่างต้องการที่จะตั้งค่า วันหมดอายุด้วยตนเอง ในการการันตีการโหลดซ้ำทันที - ผู้พัฒนาบางคนก็ไม่สนใจ. *mod\_expires* จะทำงานอย่างยอดเยี่ยมในการสร้าง CGIs - แบบไม่น่าดูเลย. แต่ก่อน, ผมเสียเวลาเป็นชั่วโมง ๆ ในการชูดั๊กโค้ดที่ยอดเยี่ยม เพื่อหาว่าทำไมสคริปเข้ารหัสถึงใช้การไม่ได้สักที. แต่เป็นเพราะผู้พัฒนาที่เจตนาจะตั้งค่าการหมดอายุอย่างถูกต้อง, ดังนั้นผมจึงปรับการตั้งค่าสำหรับ Virtual Host นี้เพื่อให้สามารถใช้งานได้. โดยเฉพาะอย่างยิ่ง, ต้องแน่ใจว่าตั้งค่าวันหมดอายุให้ถูกต้อง. ค่าข้างบนเป็นตัวอย่าง. ซึ่งคุณอาจจะต้องการค่าที่ต่างออกไป, ขึ้นกับว่าเนื้อหาของคุณเปลี่ยนแปลงบ่อยเพียงใด.

## Squid Reverse Proxy

Squid proxy มี directives สำหรับการตั้งค่าเป็นต้น. ถ้าคุณไม่มีประสบการณ์เกี่ยวกับ Squid, มันอาจจำยากอย่างมากในตอนแรกสำหรับคุณ. เพราะฉะนั้น, ผมจะแสดงให้เห็นถึงไฟล์การตั้งค่าบางส่วนเท่านั้น, ซึ่งแสดงถึงสิ่งที่ผมกำลังจะทำ. ประสิทธิภาพของมีประโยชน์มากทั้งสองด้าน. ผมจะสมมติว่าเราได้ติดตั้ง Squid proxy 2.6.x จากต้นฉบับไว้ที่ `/usr/local/squid/`.

reverse proxy สมมติถึงที่อยู่ของเว็บเซิร์ฟเวอร์จริง ๆ. มันมีไว้สำหรับขวางกั้นทุก ๆ การร้องขอ, แล้วทำการเปรียบเทียบกับแคชที่มีอยู่. สมมติว่าเรามีเครื่องอยู่สองเครื่อง:

- `stingray.example.net` serving <http://www.example.net/> [2] (172.16.23.42)
- `squid.example.net` (172.16.23.43)

เครื่องท้องถิ่น `/usr/local/squid/etc/squid.conf` กำหนด Squid ของเรามาจะทำอะไร. เราเริ่มจาก IP addresses, และบอกให้มันรอฟังการเชื่อมต่อเข้ามาจากพอร์ต 80.

```
http_port      172.16.23.43:80 vhost vport
http_port      127.0.0.1:80
icp_port       0
cache_peer     172.16.23.42 parent 80 0 originserver default
```

ICP หมายถึง Internet Cache Protocol. เราไม่จำเป็นต้องใช้มัน, และปิดการใช้งานโดยตั้งเป็นพอร์ต 0. *cache\_peer* บอกถึง reverse proxy ของเรา เพื่อส่งต่อทุก ๆ การร้องขอที่ไม่สามารถจัดการได้ของเครื่องเว็บแม่ข่ายของเรา. จากนั้น, เราได้กำหนดกฎการเข้าถึง. โดยคำนึงถึงลักษณะสถานะการณ์ของ พร็อกซีเครื่องลูกข่าย, reverse proxy สำหรับเครื่องเว็บแม่ข่ายสาธารณะในการตอบการร้องขอของทุกคน. **ระวัง:** นี่เป็นเหตุผลว่าทำไมไม่รวม `forward` และ reverse proxies ด้วยกัน, หรือคุณจะหยุดการทำงานของพร็อกซีที่เปิดอยู่ ซึ่งเป็นสิ่งที่ไม่ดีเลย.

```
acl            all src 0.0.0.0/0.0.0.0
acl            manager proto cache_object
acl            localhost src 127.0.0.1/255.255.255.255
acl            accel_hosts dst 172.16.23.42 172.16.23.43
http_access    allow accel_hosts
http_access    allow manager localhost
http_access    deny manager
http_access    deny all
deny_info      http://www.example.net/ all
```

บรรทัด `acl` กำหนดกลุ่ม. `accel_hosts` คือเครื่องแม่ข่ายของเราสองเครื่อง. `http_access allow accel_hosts` อนุญาตให้ทุก ๆ คนสามารถเข้าถึงเครื่องแม่ข่ายนี้ได้. บรรทัดอื่น ๆ เป็นค่ามาตรฐานของการตั้งค่า Squid, และหยุดการทำงานของตัวจัดการ URL. เราไม่จำเป็นต้องใช้ในตอนนี้. บรรทัดสุดท้ายเป็นเครื่องป้องกันสำหรับหน้าแสดงข้อผิดพลาดไม่พึงประสงค์. (Squid มีชุดของมันเอง: ซึ่งต่างจากหน้าแสดงข้อผิดพลาดของ Apache.) ผู้ใช้ต่าง ๆ ถูกส่งเข้าสู่หน้าแรกในกรณีที่มีปัญหาในการร้องขอ. คุณสามารถเห็น [squid.conf](#) [ตัวเต็ม](#) [3] แบบแยกออกจากกัน, เพราะการหยุด "เพียงแค" จัดการกับการติดตั้งแคชและปรับให้เข้ากัน. (พิจารณา: ไฟล์การปรับแต่งนำมาจากเครื่องแรม 2 GB และดิสก์จำนวนมาก. บางทีคุณอาจจะต้องการลดขนาดของแคชลง.) อย่างที่ผมพูด, Squid สามารถทำสิ่งที่ยอดเยี่ยม. ตราบเท่าที่ Squid เปิดและทำงาน, เราพร้อมที่จะส่งผู้ใช้งานสู่ reverse proxy แล้ว.

## ข้อมูลสถิติ

คุณจะต้องระมัดระวัง, ถ้าคุณจะดูในเรื่องความแม่นยำของสถิติที่ได้จากล็อกของ Apache. การจัดการการร้องขอ HTTP จะถูกกรองโดย Squid reverse proxy. นั่นแสดงว่าเครื่องแม่ข่าย Apache จะเห็นการร้องขอที่น้อยลง, และหมายเลข IP จะมาจากเครื่องแม่ข่ายที่เป็นพร็อกซี. ซึ่งเป็นแนวคิดของการติดตั้ง. คุณสามารถรวบรวมล็อกที่คล้ายกับ Apache บน Squid, ถ้าคุณปรับเปลี่ยนรูปแบบ.

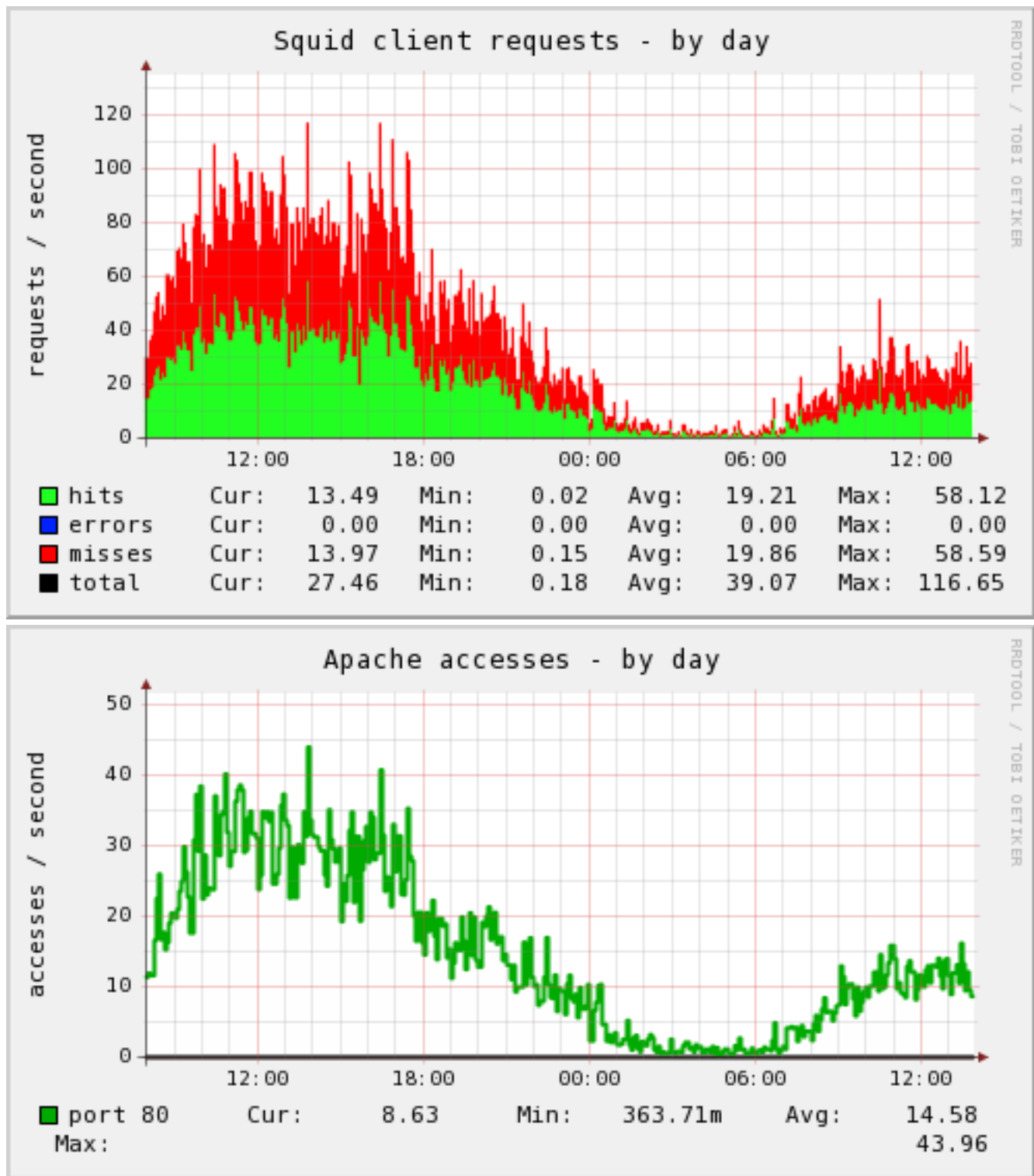
```
logformat combined %{Host}>h %>a %ui %un [%tl] "%rm %ru HTTP/%r
v" %Hs %<st "%{Referer}>h" "%{User-Agent}>h" %Ss:%Sh
logformat vcombined %{Host}>h %>a %ui %un [%tl] "%rm %ru HTTP/%
rv" %Hs %<st "%{Referer}>h" "%{User-Agent}>h"
access_log /var/log/squid/access.log combined
access_log /var/log/squid/vaccess.log vcombined
```

ในการรวบรวมสู่การวิเคราะห์ไฟล์ล็อก, คุณจะต้องทำสำเนาล็อกบน reverse proxy แล้วรวมเข้ากับล็อกของ Apache. ตราบเท่าที่เว็บยังใช้งานพร็อกซี หรือยังทำการใช้เทคนิคสมดุลการทำงาน การบำรุงรักษาความแม่นยำของข้อมูลสถิติจะทำให้ค่อนข้างยุ่งยาก.

## กระตุ้นการทำงานของแคช

หลังจากที่คุณทำการปรับแต่งการทำงานของ Apache และ Squid, คุณพร้อมที่จะทดสอบทุกอย่างแล้ว. เริ่มจากโฮสต์เสมือนที่วัตถุประสงค์ใช้ในการทดสอบ. เปลี่ยนข้อมูล DNS เพื่อชี้ไปยังเครื่อง reverse proxy. ตรวจสอบล็อก. ลองเล่นดู. วิเคราะห์ส่วนหัว. เมื่อคุณมั่นใจแล้ว, เปลี่ยนข้อมูล DNS อื่น ๆ. บันทึกสำหรับการวิเคราะห์ปัญหา: คุณสามารถบังคับการโหลดใหม่ "จริง ๆ" ใน Internet Explorer และ Mozilla Firefox ถ้าคุณกดปุ่ม Shift ค้างไว้แล้วกดปุ่ม Reload "Reload". เนื่องจากการโหลดแบบปกติจะได้รับแคชจากสำเนาในเครื่องแล้วตอนนี้.

คุณอาจจะไม่ประทับใจสำหรับอะไรที่เปลี่ยนแปลง, ให้ดูที่ล็อก. ผมแนะนำให้ทำการเฝ้าดูระบบจากข้อมูลสถิติ, ซึ่งอาจใช้ Munin เป็นต้น, แล้วคุณก็จะเห็นอย่างชัดเจนว่าเครื่องแม่ข่ายคุณทำอะไรอยู่. ผมมีกราฟที่อยู่สองภาพจากเครื่องแม่ข่ายทดสอบ, ได้ระหว่างการทำโหลดทดลอง.



กราฟแรก, สีแดงแสดงให้เห็นแคชที่ไม่พบ; สีเขียวแสดงแคชที่พบ. ภาพข้างล่าง, คุณสามารถเห็นการค้นพบบนเครื่องแม่ข่าย Apache ภายหลัง reverse proxy. รูปร่างมีลักษณะคล้าย ๆ กัน, แต่อย่าลืมว่าการร้องขอที่เป็นสีเขียวใน Squid server จะไม่เข้าถึงยัง Apache, และทำให้ช่วยลดภาระงานที่เกิดขึ้น. ถ้าคุณเปรียบเทียบผลที่ได้, คุณจะเห็นเพียง 1 ใน 2 ของการร้องขอที่ถูกส่งไปยังเครื่องแม่ข่าย Apache .

## Summary

ตอนนี้, คุณรู้แล้วว่าความสามารถบรรลุผล จากการใช้ทรัพยากรของ Apache และ Squid. เครื่องแม่ข่ายเว็บของเราจัดการโดยใช้การจราจรที่ดีขึ้น, ภาระงานของ CPU ลดลงถึง 50%, และผู้ใช้งานทุกคนเว็บต่างมีความสุขอีกครั้ง . คุณอาจจะต้องทำอะไรมากกว่านี้, ถ้าคุณใช้การเชื่อมต่ออินเทอร์เน็ตหลายทาง และทำการสมดุลภาระงานบนไฟวอลล์หรือเราเตอร์ของคุณ. โชคดี, ที่เราไม่จำเป็นต้องทำมันในกรณีนี้.

## ลิงค์ที่เป็นประโยชน์

ไม่มีสั้วหรือซอฟต์แวร์ที่ทำอันตราย เมื่อขณะที่ใช้เตรียมบทความนี้. บางทีคุณอาจจะหวังที่จะติดตามเครื่องมือและบทความต่าง ๆ. ซึ่งอาจจะช่วยรักษาเว็บไซต์ของคุณได้.

- [Apache's mod\\_expires](#) [4]
- [เอกสารการสอนเรื่องแคชสำหรับเว็บมาสเตอร์](#) [5]
- [LiveHTTPHeader](#) [6] - ส่วนเพิ่มเติมของ Firefox สำหรับดูส่วนหัวของ HTTP
- [Munin Project](#) [7] - โปรแกรมเฝ้าดูเครื่องแม่ข่ายน้ำหนักเบา
- [Squid Proxy](#) [8]
- [คู่มือ Squid proxy 2.6/3.0](#) [9]

อภิปรายปัญหา: [อภิปรายบทความนี้กับ The Answer Gang](#) [10]

René Pfeiffer



René เกิดในปีของการค้นพบ Atari และ ออกวางเกม Pong. เมื่อเริ่มเป็นหนุ่มเขาเริ่มที่จะแยกชิ้นส่วนสิ่งต่าง ๆ เพื่อให้เห็นว่าเขาทำมันได้อย่างไร. เขาไม่สามารถที่จะผ่านสถานที่ก่อสร้างได้ โดยไม่มองสายไฟฟ้า ที่ดูทำทางน่าสนใจ. ความสนใจเกี่ยวกับคอมพิวเตอร์เริ่มขึ้น เมื่อคุณตาของเขาซื้อไมโครคอนโทรลเลอร์ขนาด 4 บิต ด้วยหน่วยความจำ 256 ไบต์ และระบบปฏิบัติการ 4096 ไบต์ ส่งผลให้เขาเรียนรู้ภาษาแอสเซมบลีก่อนภาษาอื่น ๆ.

หลังจากเรียนจบชั้นมัธยม เขาก็เข้ามหาวิทยาลัยเพื่อที่จะศึกษาด้วยฟิสิกส์. จากนั้น เขาก็เริ่มเก็บประสบการณ์กับ C64, C128, Amigas สองแบบ, Ultrix ของ DEC, OpenVMS และ ท้ายสุด GNU/Linux บนเครื่องคอมพิวเตอร์ส่วนบุคคลในปี 1997. เขาเริ่มใช้ Linux จนกระทั่งถึงทุกวันนี้ และเขายังชอบที่จะแยกชิ้นส่วนต่าง ๆ และประกอบมันเข้าเหมือนเดิม. อิสระของช่างซ่อม นำเขาใกล้ชิดกับการปรับเปลี่ยนของ Free Software, ที่ซึ่งเขาได้ใส่ความมุ่งมั่นในการทำสิ่งที่ถูก. เพื่อเข้าใจว่าสิ่งต่าง ๆ ทำงานได้อย่างไร. เขายังได้เข้าร่วมกลุ่มสิทธิส่วนบุคคลเกี่ยวกับลิขสิทธิ์ดีจิดี้อีกด้วย.

กระทั่งปี 1999 เขาได้ทำการเสนอความสามารถในลักษณะผู้ประกอบการอิสระ. กิจกรรมหลักของเขา รวมถึงการเป็นผู้ดูแลระบบ/เครือข่าย, การเขียนต้นร่างและการให้คำปรึกษา. ในปี 2001 เขาเริ่มที่จะบรรยายเกี่ยวกับความปลอดภัยของระบบคอมพิวเตอร์ที่ Technikum Wien. ซึ่งแยกจากกันกับการเริ่มต้นที่จะตรวจสอบระบบคอมพิวเตอร์, ตรวจสอบฮาร์ดแวร์อย่างละเอียดและการพูดคุยเกี่ยวกับอุปกรณ์ด้านเครือข่าย เขารักในการดำน้ำ, งานเขียน, หรือการถ่ายรูปด้วยกล้องดิจิทัลของเขา. เขาอยากที่จะมีเรื่องเล่าและบทบาที่อีก トラบเท่าที่เค้าหาเวลาว่างได้ บนเครื่องบันทึกข้อมูลสำรองของเขา.

pooz

---



pooz เป็นผู้ดูแลระบบ/นักแก้ไขปรับปรุงโปรแกรมเว็บ ทำงานที่กรุงเวียนนา ประเทศออสเตรีย. ซอฟต์แวร์ Free/Open Source ได้เป็นตัวเลือกของเครื่องมือในการพัฒนาของเขาตั้งแต่ช่วงปี 90.

สงวนลิขสิทธิ์ ปี 2006, René Pfeiffer and pooz. ออกวางภายใต้สัญญาอนุญาต [Open Publication license](#) [11] เว้นแต่บันทึกภายในบทความบอกเป็นอย่างอื่น. Linux Gazette ไม่ได้ถูกสร้างขึ้น, ได้รับการสนับสนุน, หรือได้รับการรับรอง จากผู้ให้โฮสต์, SSC, Inc.

ตีพิมพ์ในเล่มที่ 132 ของ Linux Gazette, พฤศจิกายน ปี 2006



SiteTags: [Linux Gazette](#) [12]

[Reverse Proxy](#) [13]

[Apache](#) [14]

Source URL (modified on 2009-08-15 10:38): <https://sake.in.th/node/9>

#### Links

[1] <http://www.luchs.at/linuxgazette/index.html>

[2] <http://www.example.net/>

[3] <https://sake.in.th/lg/132/misc/pfeiffer/squid.conf.txt>

[4] [http://httpd.apache.org/docs/2.0/mod/mod\\_expires.html](http://httpd.apache.org/docs/2.0/mod/mod_expires.html)

[5] [http://www.mnot.net/cache\\_docs/](http://www.mnot.net/cache_docs/)

[6] <http://livehttpheaders.mozdev.org/>

[7] <http://munin.projects.linpro.no/>

[8] <http://www.squid-cache.org/>



[9] <http://www.visolve.com/squid/squid30/contents.php>

[10] <mailto:tag@lists.linuxgazette.net?subject=Talkback:132/pfeiffer.html>

[11] <http://linuxgazette.net/copying.html>

[12] <https://sake.in.th/category/sitetags/linux-gazette>

[13] <https://sake.in.th/category/sitetags/reverse-proxy>

[14] <https://sake.in.th/category/sitetags/apache>