نشریه علمی- ترویجی محاسبات نرم
شمارهٔ چهارم/ پاییز و زمستان 92/ صفحهٔ 16ـ25

دریافت مقاله: 92/2/14
پذیرش مقاله: 93/2/6

بهبود کارایی پروتکلSIP در شرایط اضافه‌بار با استفاده از قابلیت مبتنی بر پنجره

احمدرضا منتظرالقائم[1*]، محمدحسین یغمایی مقدم[2]

[1] دانشجوی دکتری، دانشکده مهندسی، دانشگاه فردوسی مشهد، مشهد، ایران
Ahmadreza.montazerolghaem@stu.um.ac.ir
[2] استاد، دانشکده مهندسی، دانشگاه فردوسی مشهد، مشهد، ایران
hyaghmae@ferdowsi.um.ac.ir

چکیده: پروتکل Session Initiation Protocol (SIP)، پروتکل سیگنالینگ لایهٔ کاربرد برای آغاز، مدیریت و پایان جلسات صدا و تصویر از طریق بسته‌ها می‌باشد. این پروتکل به‌عنوان اصلی‌ترین پروتکل سیگنالینگ در IMS در نظر گرفته شده است؛ بنابراین، با توجه به استفادهٔ وسیع از آن و وجود کاربران میلیونی در آیندهٔ نزدیک، لازم است که رفتار و کارایی سرورهای SIP در حالت‌های اضافه‌بار، بررسی شود. نشان داده شده است که بازدهی سرورهای SIP هنگام بازه‌های اضافه‌بار به‌شدت کاهش می‌یابد که دلیل آن سازوکار تعبیه‌شده در داخل SIP برای ارسال مجدد پیام‌هاست. ما در این مقاله یک الگوریتم تطبیقیِ توزیع‌شده و انتها به انتهای مبتنی بر پنجره را برای کنترل اضافه‌بار پیشنهاد می‌کنیم. این الگوریتم از بازخورد صریح از سرور پایین‌دستی استفاده نمی‌کند. سرورهای بالادستی از تعداد پیام‌های تأیید، به‌عنوان معیاری از میزان بار سرور پایین‌دستی استفاده می‌کنند؛ لذا الگوریتم پیشنهادی هیچ پیچیدگی یا پردازش اضافی را بر سرور تحت اضافه‌بار پایین‌دستی تحمیل نمی‌کند و در نتیجه رویکردی بسیار قوی می‌باشد. با استفاده از نتایج پیاده‌سازی در یک بستر واقعی، نشان می‌دهیم که روش پیشنهادی ما در عمل، می‌تواند باعث حفظ گذردهی در شرایط اضافه‌بار شود.

واژه‌های کلیدی: سرور **SIP**، کنترل اضافه‌بار مبتنی بر پنجره، پروکسی **Asterisk**، زیرسیستم چند رسانه‌ای مبتنی بر **IP.**

1. مقدمه

پروتکل آغاز جلسه (SIP) یک پروتکل سیگنالینگ لایهٔ کاربردی است که توسط IETF[1] استاندارد شده است و برای شروع، تصحیح و اتمام جلسات چندرسانه‌ای استفاده می‌شود. این باور وجود دارد که SIP می‌تواند ایجاد خدمات در شبکه‌های نسل بعد (NGN[2]) را آسان کند. در واقع پروژهٔ 3GP اخیراً از SIP به‌عنوان پروتکل اصلی سیگنالینگ در IMS [3] استفاده کرده است.

فراهم‌کردن کیفیت سرویس در شبکه‌های NGN، تعبیهٔ سازوکارهایی را برای برخورد با افزایش ناگهانی ترافیک SIP ملزم می‌کند که باعث اضافه‌بار در سرورهای SIP می‌شوند. فهرستی از دلایل اضافه‌بار در RFC5390 آورده شده است که از آن دسته می‌توان به کاهش ناگهانی ظرفیت پردازشی، خطای اجزا، شروع به کار یک‌باره، ازدحام آنی و حملات DOS [4] اشاره کرد. شرایط اضافه‌بار زمانی تشدید می‌شود که SIP از UDP[5] استفاده کند و همچنین از طریق ارسال مجدد به تمامی تقاضاهای پاسخ داده نشده، قصد افزایش قابلیت اطمینان را داشته باشد. هرچند که استفاده از TCP[6] تاحدودی عملکرد سرور را در اضافه‌بار بهبود می‌بخشد، همان‌طور که در [6و7] نشان داده شده، مشکلات مقیاس‌پذیری را به‌وجود می‌آورد. تاکنون UDP متداول‌ترین گزینه برای SIP بوده است و ما در این مقاله فرض می‌کنیم که SIP بر روی پروتکل لایهٔ انتقال UDP اجرا می‌شود.

در این مقاله، یک الگوریتم جدید توزیع‌شدهٔ کنترل اضافه‌بار SIP را ارائه می‌کنیم که در آن، نیازی به بازخورد صریح نیست. در این رویکرد، از تعداد پیام‌های تأیید به‌عنوان نشانه‌ای از اضافه‌بار در برخی از سرورهای پایین‌دست استفاده می‌شود. این پارامتر توسط سرور بالادست اندازه‌گیری می‌شود. سرور بالادست با استفاده از این اطلاعات، بار سیگنالینگ تحمیل‌شده بر سرور پایین‌دستی را با استفاده از یک رویکرد تطبیقی مبتنی بر پنجره کنترل می‌کند. نشان داده می‌شود که این رویکرد هم گذردهی بسیار بالایی دارد و هم پویا است. رویکرد ما تأثیری بر سرور تحت اضافه‌بار ندارد؛ به‌خصوص اینکه سرور تحت اضافه‌بار، بار اضافی برای تولید بازخورد و یا رد احتمالی تماس‌ها را به‌وجود نمی‌آورد.

همچنین این الگوریتم به‌سبب آنکه مبتنی بر پنجره است، دارای ویژگی self-clocking (روشی برای هم‌زمان کردن دو دستگاه فرستنده و گیرندهٔ مخابراتی) است. ما طرح پیشنهادی خود را تحت توپولوژی و شرایط مختلف شبکهٔ SIP آزمایش می‌کنیم. همچنین ما این الگوریتم پیشنهادی کنترل اضافه‌بار را در یک پراکسی متن باز SIP به نام Asterisk [9] پیاده‌سازی کردیم و دیدیم که عملکرد آن مطابق انتظار است. ادامهٔ این مقاله به‌صورت زیر سازمان‌دهی شده است: بخش 2 شامل روش‌های موجود کنترل اضافه‌باراست. در بخش 3 جزئیات این الگوریتم پیشنهادی کنترل اضافه‌بار آورده شده است. در بخش 4 به ارائهٔ مدل پیاده‌سازی، توپولوژی شبکه و نتایج ارزیابی عملکرد مکانیزم پیشنهادی پرداخته شده و در نهایت، در بخش 5 نتیجه‌گیری این مقاله و کارهای آتی به‌صورت مختصر بیان شده است.

2. کارهای مرتبط با موضوع کنترل اضافه‌بار SIP

هدف کنترل اضافه‌بار SIP، نگه‌داشتن گذردهی سرور در مقدار نزدیک به ظرفیت آن، در شرایط وقوع اضافه‌بار است. در شکل 4، منحنی Theoretical چگونگی مکانیزم کنترل اضافه‌بار ایده آل را در زمانی که گذردهی سرور برابر 700 cps است، نشان می‌دهد. یک مدل سیستماتیک برای کنترل اضافه‌بار SIP در [10] معرفی شده است. در این مدل، اجزای حلقه کنترلی و درجه همکاری بین سرورهای SIP در شرایط اضافه‌بار تعیین شده است. به‌طور کلی، دو روش برای کنترل اضافه‌بار وجود دارد: محلی و توزیع‌شده. در کنترل اضافه‌بار محلی، حلقهٔ کنترلی به‌طور داخلی بر روی سرور تحت اضافه‌بار پیاده می‌شود [10]؛

1. Internet Engineering Task Force
2. Next-generation network
3. IP Multimedia Subsystem
4. Denial-of-service
5. User Datagram Protocol
6. Transmission Control Protocol

لذا یک سرور SIP زمانی که به حد ظرفیت خود نزدیک می‌شود، شروع به رد تقاضاهای اضافی می‌کند. این امر با ارسال یک پیام 503 Service Unavailable در پاسخ به یک INVITE انجام می‌شود [1]. تعدادی از روش‌های کنترل اضافه‌بار محلی که در سیاست ردّ تماس و معیار تشخیص اضافه‌بار با هم متفاوت‌اند، در بسیاری از مقالات ارائه و ارزیابی شده‌اند. در [3، 4، 11و12] الگوریتم‌های مبتنی بر طول صف پیشنهاد شده‌اند. همچنین در [3و11] الگوریتم‌های مبتنی بر اشغال، به نام OCC[1] معرفی شده‌اند که از میزان استفاده از واحد پردازندهٔ مرکزی[2] به‌عنوان محرکی برای ردّ تماس‌ها استفاده می‌کنند (Local[3] در شکل 4).

علاوه بر این، به ترتیب در [6و13] تأثیر صف اولویت و پروتکل لایهٔ انتقال بر عملکرد سیگنالینگ SIP ارائه شده است. همچنین Garroppo و همکارانش [14] یک استراتژی هوشمند صف‌بندی با اولویت ترکیب‌شده با تشخیص اضافه‌بار مبتنی بر طول صف را به‌عنوان یک سیاست کنترل محلی پیشنهاد کرده است. به هر حال، تحت اضافه‌بار سنگین، سرور تحت اضافه‌بار بسیاری از منابع خود را به ردّ تقاضاهای اضافی اختصاص می‌دهد که این باعث افت گذردهی می‌شود. این امر به‌طور واضح، در منحنی‌ای تحت نام Local در شکل 4 قابل مشاهده است. همچنین در [3] می‌توان دید که برای بار 100 درصد (به‌عبارت دیگر 400 تماس بر ثانیه) گذردهی سیستم به اندازهٔ 25% افت می‌کند و از 200cps به 160cps تغییر می‌یابد. این افت گذردهی را می‌توان به‌عنوان هزینهٔ واحد پردازندهٔ مرکزی برای اجرای الگوریتم کنترل اضافه‌بار در نظر گرفت. همچنین می‌توان آن را در بسیاری از موارد که شرایط اضافه‌بار از سرورهای SIP بالادستی ناشی می‌شود، کم کرد. به این شرایط، اضافه‌بار «سرور به سرور» گفته می‌شود. سازوکارهای محلی ردّ تماس با کنترل اضافه‌بار توزیع‌شده، همراه می‌شوند و بدین صورت سرورهای بالادستی، بار سرور پایین‌دستی را کنترل می‌کنند و تا حدّ ممکن، آن را نزدیک به ظرفیت سرور نگه می‌دارند. عموماً سرور تحت اضافه‌بار منابع خود را رصد می‌کند و به تمام سرورهای بالادست، یک بازخورد صریح ارسال می‌کند و طی آن به آنان اطلاع می‌دهد که اضافه‌بار رخ داده است و نیز احتمالاً مقدار بار قابل پذیرش خود را به آنان مخابره می‌کند. متعاقباً سرورهای بالادست نرخ ارسال خود را کاهش می‌دهند. بر مبنای نوع بازخورد انتقال‌یافته و محل استقرار تابع کنترل، روش‌های کنترل اضافه‌بار صریح را می‌توان به الگوریتم‌های مبتنی بر نرخ، سیگنال، اتلاف و پنجره طبقه‌بندی کرد [10]. Sun و همکارانش [17و18] یک روش مدیریت جریان را ارائه می‌کنند که در آن، سرور ارسال‌کنندهٔ بار، میزان به‌کارگیری از واحد پردازندهٔ مرکزیِ سرورهای دریافت‌کننده را رصد می‌کند و از طریق طبقه‌بندی بسته‌ها، حذف ارسال مجدد و عبور تدریجی تماس‌های اضطراری ، مانع از بروز اضافه‌بار می‌شود.

در [6] تأثیر به‌کارگیری پروتکل‌های لایهٔ انتقال مختلف، به‌خصوص تأثیر مکانیزم کنترل پنجره در TCP، روی گذردهی SIP و تأخیر برقراری تماس، بررسی شده است.

در [19] و [25] سه روش مبتنی بر پنجره پیشنهاد شده است که در آن‌ها، پروکسی تحت اضافه‌بار به‌طور پویا و مستمر، ظرفیت پاسخ‌گویی کنونی خود را تخمین می‌زند و تحت عنوان طول پنجره، بار تحمل‌شدنی خود را (یا تعداد تماس‌هایی را که می‌تواند پردازش کند) به پروکسی‌های بالادست اطلاع می‌دهد.

Noel و Johnson [21] دو الگوریتم مبتنی بر بازخورد صریح را ارائه می‌کنند که یکی میزان استفاده از CPU و تأخیر صف‌بندی را به‌عنوان معیار ازدحام در نظر می‌گیرد و دیگری ترکیبی از الگوریتم‌های مبتنی بر پنجره و سیگنال است.

در مقاله [26] یک الگوریتم کنترل اضافه‌بار توزیع‌شده مبتنی بر پنجره در پروتکل SIP را پیشنهاد می‌کند که برای اضافه‌بار سرور به سرور کاربردی است. این مقاله از تأخیر برقراری تماس (منظور زمان بین پیام‌های INVITE ارسالی تا OK دریافتی است) که در سرور بالادست اندازه‌گیری می‌شود، به‌عنوان نشانگری از بروز اضافه‌بار در جایی در طول مسیر

1. Occupancy
2. CPU
3. مکانیزم محلی

منتهی به UAS استفاده می‌کند. نظریهٔ صف‌بندی بیان می‌کند که هنگامی که اضافه‌بار در سرور در حال وقوع است، افزایش قابل توجهی در تأخیر رخ می‌دهد.

مقاله [27] کنترل اضافه‌بار مبتنی بر پنجره را با استفاده از روابط ریاضی مدل می‌کند تا بتواند از این روابط، در بررسی این روش استفاده کند. برای این کار یک شبکهٔ ذوزنقه‌ای SIP را با استفاده از روابط ریاضی مدل می‌کند.

نویسندگان مقاله [28] یک مکانیزم سادهٔ کنترل اضافه‌بار مبتنی بر پنجره را بر روی پروکسی OpenSIPS توسعه داده‌اند.

درحالی‌که روش‌های کنترل اضافه‌بار محلی از هزینهٔ هنگفت ردّ تماس رنج می‌برند، بیشتر الگوریتم‌های توزیع‌شدهٔ پیشنهادی، به‌دلیل نیاز به رصد بار و محاسبهٔ یک بازخورد صریح به پیچیدگی سرور تحت اضافه‌بار می‌افزایند. عیب دیگر استفاده از بازخورد صریح، تأخیری است که بازخورد برای رسیدن به سرورهای بالادستی متحمل می‌شود. این تأخیر ممکن است باعث ناپایداری یا دست کم نوسان عملکرد الگوریتم شود. این موضوع در بحث الگوریتم‌های کنترل ازدحام اینترنت، پدیده‌ای شناخته‌شده است. پس در نتیجه معایب اصلی طرح‌های کنونی کنترل اضافه‌بار این است که اولاً اتکای صرفاً به ردّ کردن تماس‌ها به‌صورت محلی، باعث کاهش گذردهی می‌شود؛ ثانیاً شناسایی اضافه‌بار و تولید بازخورد، هزینهٔ استفاده از واحد پردازندهٔ مرکزی را در بر داشته و بر گذردهی تأثیر می‌گذارد. شرایط اضافه‌بار وضعیتی پیچیده است که می‌تواند به دلایل بسیاری رخ دهد.

3. الگوریتم پیشنهادی: کنترل تطبیقی اضافه‌بار مبتنی بر پنجره

در این بخش، یک روش کارای مبتنی بر بازخورد ضمنی برای کنترل اضافه‌بار مبتنی بر پنجره را پیشنهاد می‌کنیم که در آن، تعدادِ پیام‌های تأیید به‌منظور حل و فصل این مشکلات در نظر گرفته شده است. همچنین انواع پیام‌های درخواست را فقط به 5 نوع (INVITE,BYE,CANCEL,REGISTER,OPTIONS) محدود می‌کنیم. این 5 پیام درخواست، نقشی اساسی را به‌عنوان سیگنالینگ SIP بین پراکسی سرورها بازی می‌کنند. دلیل اصلی اضافه‌بار سرور SIP نیز همین 5 نوع پیام درخواست است. در روش سنتی، کنترل اضافه‌بار مبتنی بر پنجره، از بازخورد برای تجدید بیشینهٔ اندازهٔ پنجره استفاده می‌شود. با این حال، روش ارائه‌شده در این مقاله، با هدف حفاظت از منابع سرور، از بازخورد استفاده نمی‌کند. در این روش به جای بازخورد، از تعداد پیام‌های درخواست و تأیید استفاده می‌شود. ایدهٔ اصلی این روش، محاسبهٔ بیشینهٔ جدید اندازهٔ پنجره با تحلیل نسبت پیام‌های درخواست به پیام‌های تأیید است. برای محاسبهٔ این نسبت، از پاسخ‌ها به‌عنوان پیام تأیید استفاده می‌شود.

مسئلهٔ اساسی در الگوریتم‌های مبتنی بر پنجره، طول پنجره‌هاست که باید با بازخورد پروکسی پایین‌دست تنظیم شود. پس می‌توان نتیجه گرفت که می‌توان با محدودکردن طول پنجره، از بروز اضافه‌بار جلوگیری کرد [2]. در این الگوریتم‌ها سرور بالادست، پنجره‌ای از تراکنش‌های فعال را در خود نگه می‌دارد. تماس‌های جدید در صورتی که پنجره جای خالی داشته باشد، پذیرفته می‌شود. شکل 1 طرح ما از یک سرور بالادست را نشان می‌دهد که در آن، از الگوریتم کنترل اضافه‌بار پیشنهادی ما استفاده شده است. سرور بالادست یک پنجرهٔ کنترل جریان سیگنالینگ به طول Wmax را نگهداری می‌کند که تراکنش‌های آن با یک سرور مقصد شروع می‌شود. به‌عبارت دیگر، به‌ازای هر سرور SIP مقصد یک پنجره وجود دارد. توجه کنید که سرور پایین‌دست (یا مقصد) یک پراکسی SIP ساده است، نیازی به اصلاح ندارد. در صفحهٔ بعد نیز شبه کد توصیفی از روند پردازش پیام برای سرور بالادستی آورده شده است که در ادامه به توضیح آن می‌پردازیم.

پیام‌های «INVITE» که تقاضای ایجاد تماس جدید را دارند و از طرف UACها (یا سایر سرورها) می‌رسند، در صف پیام‌های ایجاد تراکنش قرار می‌گیرند. سرور تنها در صورتی که اسلات‌های خالی در پنجره وجود داشته باشد، تقاضاهای جدید (یا به‌عبارت دیگر ارسال مجدد نشده) «INVITE» را می‌پذیرد و ارسال می‌کند؛ در غیر این صورت، پیام‌های جدید با ارسال

یک پاسخ «503 Service Unavailable» رد می‌شوند. Wmax بیشینهٔ اندازهٔ پنجره برای محدودکردن تعداد پیام‌ها را مشخص می‌کند. N*request* تعداد تقاضاها و N*response* تعداد پاسخ‌ها در یک پروکسی فرستنده بار را مشخص می‌کنند. Rconfirmation (R*con*) از نسبت N*response* بر N*request* به‌دست می‌آید (N*response*/N*request*). مقدار Wmax در مرحلهٔ اول، در مقدار از پیش تعیین‌شدهٔ 1 تعریف می‌شود (خط 1). می‌توان با تنظیم بیشینهٔ اندازهٔ پنجره، بار ارسالی به پروکسی مقصد را کنترل کرد. پس از مقداردهی اولیه برای Wmax، پروکسی فرستنده به‌طور مکرر، نوع پیام را بررسی می‌کند. در صورتی که پروکسی فرستنده، پیام پاسخ «503 Service Unavailable» دریافت کند، بیانگر آن است که سرور SIP پایین‌دست دچار اضافه‌بار شده است. پس چنانچه یک پیام بررسی و به‌عنوان پاسخ « 503 Service Unavailable» درک شود، مقدار Wmax به مقدار یک به‌روزرسانی می‌شود (خط 23)؛ لذا می‌توان اضافه‌بار سرور را کاهش داد. در شرایط نرمال، مقدار Wmax پس از مقداردهی اولیه، افزایش می‌یابد. با این حال، در راستای سرعت‌بخشیدن به افزایش پنجره و به‌دست‌آوردن گذردهی بالاتر، اندازهٔ پنجره برای برقراری هر تماس، دو برابر می‌شود تا زمانی که به یک آستانه مشخص SSTH [1](خط 35) برسد. از منظر اضافه‌بار سرور، مقدار کم R*con* به‌معنای بالابودن بار در سرور است. بدین منظور زمانی که R*con* از یک مقدار آستانه کمتر شود، Wmax به نصف مقدار خود کاهش می‌یابد. این مقدار آستانه با نماد R*th* مشخص می‌شود. پس به محض تشخیص غریب الوقوع اضافه‌بار، سرور بالادست اندازهٔ پنجرهٔ خود را نصف و نیز آستانهٔ SSTH را نیز به مقدار نصف اندازهٔ کنونی پنجره، تغییر می‌دهد. این امر باعث کاهش بار سیگنالینگ تحمیل‌شده بر سرور پایین‌دست می‌شود. نصف‌کردن آستانهٔ SSTH، به پنجره اجازه می‌دهد که با سرعت به مقدار نصف اندازهٔ قبلی خود برگردد و متعاقب آن، مرحلهٔ افزایش جمعی [2] رخ می‌دهد. این امر یک مکانیزم کنترل جریان افزایش جمعی و کاهش ضربی (AIMD) است. نشان داده شده است که این مکانیزم به بیشینهٔ گذردهی همگرا شده است [24].

Overload Control Algorithm

Upstream SIP Server Behavior with request message:

```
1.  Wmax = 1,
2.  If INVITE Then
3.  /* new session */
4.  If Wmax = full Then
5.  Reject (request)
6.  /*503 Service Unavailable*/
7.  Else
8.  Update Cseq list
9.  Nrequest = Nrequest + 1
10. Forward (Message)
11. Else If Cancel Then
12. Update Cseq list
13. Nrequest = Nrequest + 2
14. Forward (Message)
15. Else
16. Update Cseq list
17. Nrequest = Nrequest + 1
18. Forward (Message)
```

19. Upstream SIP Server Behavior with response message:

```
20. If 503 Service Unavailable Then
21. /* overloaded */
22. SSTH = (Wmax / 2)
23. Wmax = 1
24. Else
25. Check the Cseq list
26. /*this is an existing session*/
27. Update Cseq list
28. Nresponse = Nresponsese + 1
29. Rcon = Nresponse / Nrequest
30. If Rcon < Rth Then
31. SSTH = (Wmax / 2)
32. Wmax = (Wmax / 2)
33. Else
34. If Wmax < SSTH Then
35. /*Slow Start*/
36. Wmax = (Wmax * 2)
37. Else
38. /*Overload Avoidance*/
39. Wmax = (Wmax + 1)
```

Parameters:

```
Wmax: Maximum Windows Size
Nrequest: The Number of Request
Nresponse: The Number of Response
Rcon: Nresponse/Nrequest
SSTH: Slow Start Threshold
```

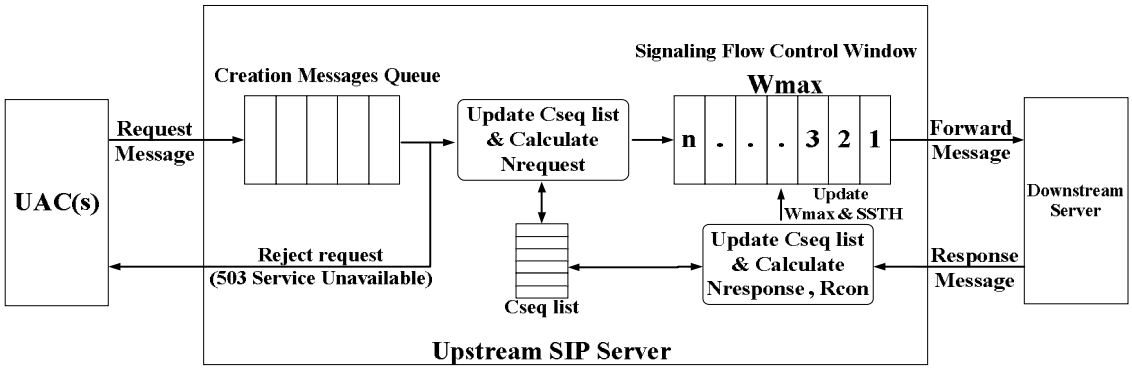
1. Slow-Start Threshold
2. additive increase phase

در ادامه، به چگونگی محاسبه و تطبیق دو مقدار N*response* و N*request* (برای به‌دست‌آوردن مقدار R*con*)، توسط پروکسی فرستندهٔ بار می‌پردازیم. تمامی پیام‌های SIP دارای فیلد هدر Cseq هستند. Cseq از یک عدد توالی[1] و نوع متد[2] تشکیل شده است. عدد توالی، تشخیص و مرتب‌کردن تراکنش‌ها را تسهیل می‌کند. ما از فیلد هدر Cseq به‌عنوان وسیله‌ای برای نگاشتن یک درخواست بر روی یک پاسخ استفاده می‌کنیم. بدین منظور، جدولی از مقادیر Cseq در پروکسی فرستندهٔ بار ساخته می‌شود. پس می‌توان با توجه به نسبت بین درخواست و پاسخ، به‌طور پویا بار سرور را کنترل کرد. هر زمان که یک پیام درخواست وارد پروکسی فرستنده می‌شود، پیش از هرچیز، نوع پیام درخواست بررسی می‌شود. در این بررسی، بین یک پیام «CANCEL» و سایر پیام‌ها تمایز قائل می‌شود. پیام‌های عمومی به غیر از پیام «CANCEL» در جدول Cseq ثبت می‌شوند؛ سپس مقدار N*request* جدید محاسبه می‌شود (N*request* +1). علاوه بر این، در زمانی که تماس‌گیرنده قصد لغو یک پیام تقاضای ارسال‌شده به طرف مقابل را دارد یا زمانی که پذیرندهٔ تماس قصد رد یک تقاضای ارسال‌شده از سوی تماس‌گیرنده را دارد، لغو جلسه می‌تواند رخ دهد. به‌عبارت دیگر، برای لغو جلسه و تقاضای قبلی ارسال‌شده از سوی مشتری، از پیام «CANCEL» استفاده می‌شود. با این حال، نباید از پیام «CANCEL» برای لغوکردن تقاضایی غیر از پیام «INVITE» استفاده کرد. می‌توان با استفاده از سازوکار شمارش که در قبل توصیف شد، با پیام‌های پاسخ رفتار کرد؛ اما نمی‌توان پیام «CANCEL» را در روندی معادل اعمال کرد؛ زیرا مشکل تکرار Cseq به‌وجود می‌آید. چنانچه پیش از رسیدن پیام تأیید یک تقاضا، یک پیام «CANCEL» به پروکسی مقصد برسد، Cseq مکرراً در جدول ثبت می‌شود. پروکسی مقصد یک پیام تأیید را فقط برای پیام «CANCEL» تولید می کند؛ بنابراین، یکی از دو Cseq تکراری حذف می‌شود. چنانچه مقادیر تکراری در این فهرست انباشته شوند، بر R*con* تأثیر خواهد گذاشت. برای حل مشکل فوق، به محض ورود یک پیام تقاضای «CANCEL» به یک پروکسی فرستندهٔ بار، یک Cseq از پیام هدف که قرار بود لغو شود، از جدول Cseq حذف می‌گردد؛ سپس یک Cseq مرتبط با پیام «CANCEL» در جدول ثبت و مقدار جدید N*request* محاسبه می‌شود (N*request* +2- خط 13). این روند در مورد پیام «CANCEL» باعث می‌شود که R*con* دقیق‌تر به‌دست آید. پس از ارسال بیش از یک تقاضا، پروکسی فرستندهٔ بار، منتظر پیام‌های پاسخ به‌منظور تأیید می‌ماند. به محض دریافت یک پیام پاسخ، فیلد هدر Cseq پیام بررسی می‌شود. چنانچه شمارهٔ Cseq در جدول Cseq که قبلاً ساخته شده است، موجود باشد، پروکسی فرستندهٔ بار، پیام پاسخ را تشخیص می‌دهد، آن را از جدول مذکور حذف می‌کند و مقدار N*response* یکی زیاد می‌شود (خط 28).

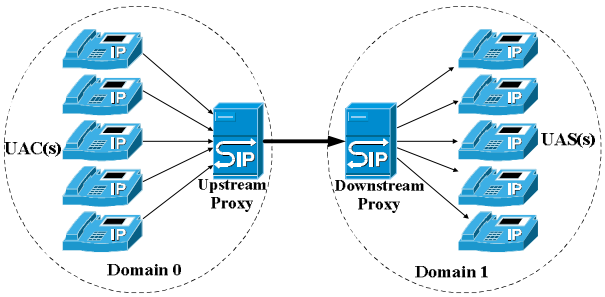
4. پیکربندی و ارزیابی عملکرد

به‌منظور بررسی کارایی الگوریتم بهبودیافته کنترل اضافه‌بار مبتنی بر پنجره، هنگام مواجهه با اضافه‌بار، از توپولوژی مرسوم شکل 2 استفاده شده است. در این توپولوژی، پیام‌های طرفین تماس از طریق دو پروکسی میانی رد و بدل می‌شوند. اضافه‌باری که پروکسی قرارگرفته در این توپولوژی با آن مواجه است، از نوع «سرور به سرور» است. الگوریتم پیشنهادی در این مقاله با الگوریتم‌های مشابه و کارآمد win-cont، win-auto و win-disc معرفی شده در مقالهٔ [25] و [19] مقایسه شده که نتایج آن در شکل‌های 4 و 5 و جدول 1 آمده و در ادامهٔ آن بحث شده است.

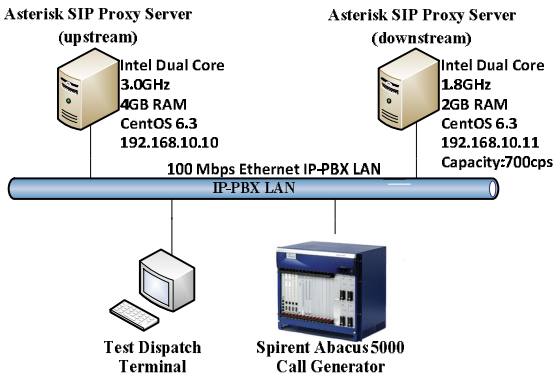
1. Sequence number
2. Method type



شکل (1): معماری پایه از کنترلر جریان پیشنهادی



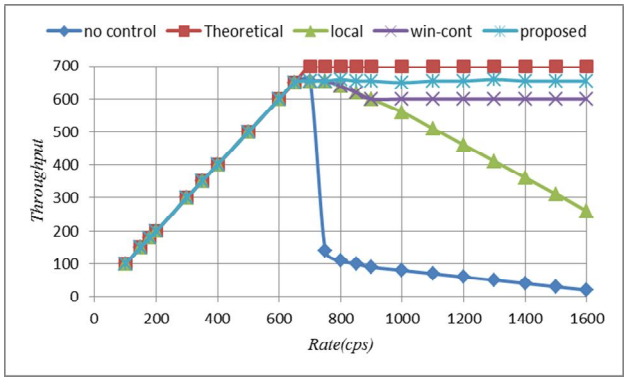
شکل (2): توپولوژی مورد آزمایش



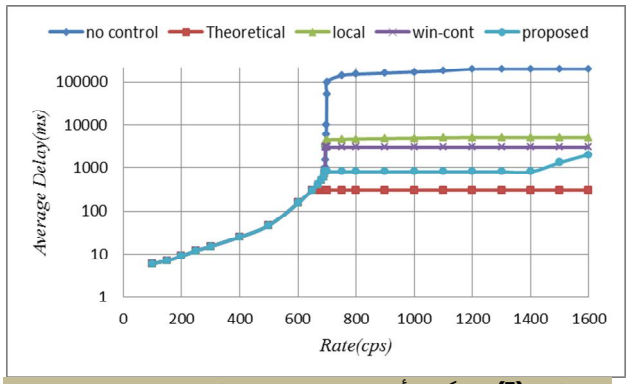
شکل (3): بستر آزمایش

برای پیاده‌سازی پروکسی سرورها از نرم‌افزار Asterisk و برای پیاده‌سازی عامل‌های کاربر از دستگاه تستر Spirent Abacus 5000 استفاده شده است. سرور بالادست یک pc با پردازندهٔ INTEL Dual Core 3GHZ با حافظهٔ 4GB و سرور پایین‌دست یک pc با پردازندهٔ INTEL 1.8 GHz با حافظهٔ 2GB است که از سیستم عامل لینوکس CentOS نسخهٔ 6.3 استفاده می‌کنند. از دستگاه Spirent Abacus 5000 برای ایجاد ترافیک با نرخ‌های ارسال متفاوت و توزیع‌های گوناگون استفاده کرده‌ایم. این دستگاه برای تست‌های مختلف شامل قابلیت همکاری[1]، کارایی[2]، مقیاس‌پذیری[3] و تست کیفیت صوت و ویدئو بر روی شبکه‌های IP استفاده می‌شود. این محصول با تولید صدها تا هزاران تماس قادر به تست کارایی و توسعه‌پذیری پروکسی تست‌شونده است. برای بررسی زمان و نوع پیام‌های ارسالی و دریافتی توسط کاربرها از گزارشاتی که دستگاه Spirent Abacus 5000 تولید می‌کند، استفاده شده است. همچنین از گزارشات نرم‌افزار asterisk برای اندازه‌گیری وضعیت پیشرفت تماس‌ها و تراکنش‌های روی پروکسی استفاده شده و نرم‌افزار oprofile نیز جهت اندازه گیری بار پردازشی پروکسی به‌کار رفته است. معیارهای متعددی برای تعیین کارایی SIP وجود دارد [8] که از این میان، در این پژوهش، روی تأخیر برقراری تماس (زمان بین ارسال INVITE از طرف UAC تا زمان دریافت OK از پروکسی)، نرخ ارسال مجدد و گذردهی پروکسی (تعداد تماس‌های موفق در واحد زمان- در اینجا یعنی تعداد «200 OK»هایی که کمتر از 10 ثانیه دریافت می‌کنیم) متمرکز شده‌ایم. نرخ تولید مکالمه از مقدار کم 100 تماس در ثانیه شروع می‌شود و تا نرخ مکالمه 1600 تماس در ثانیه و با توزیع پواسن ادامه می‌یابد. تأخیر لینک و احتمال گم‌شدن بسته‌ها در حالت کلی، ناچیز و ظرفیت پروکسی پایین‌دست در حدود 700 تماس در ثانیه محاسبه شده است.

1. interoperability
2. performance
3. scalability



شکل (4): گذردهی مکانیزم کنترل اضافه‌بار پیشنهادی

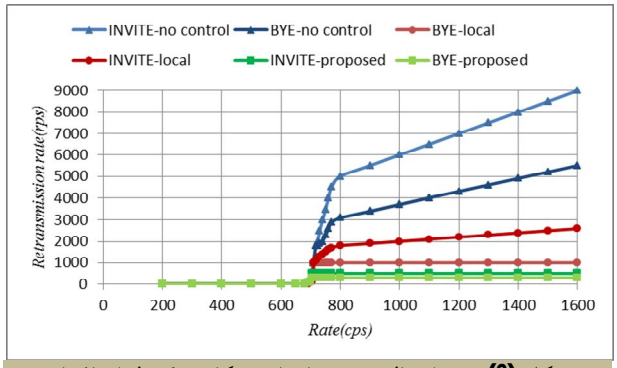


شکل (5): میانگین تأخیر برقراری تماس مکانیزم کنترل اضافه‌بار پیشنهادی

شکل 4 گذردهی را به‌صورت تابعی از نرخ درخواست تماس ورودی برای مکانیزم بهبودیافته و در حالتی که از هیچ مکانیزم کنترل اضافه‌باری استفاده نمی‌شود، نشان می‌دهد. این نمودار بیانگر آن است که با وجود مکانیزم کنترل اضافه‌بار، می‌توان گذردهی پروکسی را در حداکثر ظرفیت نگه داشت. همان‌طور که در این شکل مشخص است، با مکانیزم کنترل اضافه‌بار، پروکسی بالادست توانسته است گذردهی خود را بدون افت، تقریباً تا نرخ ارسال 1500 تماس در ثانیه (حدود دو برابر ظرفیت پروکسی پایین‌دست) حفظ کند؛ درحالی‌که اگر از الگوریتم کنترل اضافه‌بار استفاده نمی‌شد، گذردهی پروکسی بالادست تقریباً برابر با 700cps می‌ماند. اندازهٔ پنجره با دریافت درخواست‌های جدید شروع به افزایش می‌کند؛ بنابراین، تأخیر نیز افزایش پیدا می‌کند. همان‌طور که در شکل 5 نشان داده شده، میانگین زمان برقراری تماس در این پروکسی، تا حدود 1500cps به‌صورت خطی و با رشد بسیار کمتر از حالت بدون کنترل اضافه‌بار، افزایش می‌یابد. در نرخ‌های بالاتر از ظرفیت سرور پایین‌دست (700cps)، حجم بالای درخواست‌های رسیده، سبب تحریک حسگر CPU شده و تماس‌های زیادی رد خواهند شد. رد یک‌بارهٔ حجم عظیمی از تماس‌ها، سبب خواهد شد تعداد زیادی بستهٔ Ack در فاصلهٔ زمانی بسیار اندک، به پروکسی برسند و در نتیجه، صف پروکسی را پر کنند تا آنجا که جایی برای بسته‌های پاسخ مربوط به تماس‌های در جریان، در صف پروکسی باقی نماند. گم‌شدن بسته‌های پاسخ، محرکی برای فعال‌کردن مکانیزم ارسال مجدد در هر دو سرور و مشتری خواهد شد که به وخیم‌ترشدن اوضاع می‌انجامد؛ بنابراین، تماس‌هایی که در این حالت در پروکسی پذیرفته می‌شوند، با تأخیر بسیار زیاد پذیرفته خواهند شد. نمودار شکل 6 نرخ ارسال مجدد را برای درخواست‌های INVITE و BYE در سمت کاربر نشان می‌دهد. همان‌طور که انتظار می‌رود، در زمانی که از مکانیزم کنترل اضافه‌بار پیشنهادی در سرور بالادست استفاده می‌کنیم، نرخ ارسال مجدد پیام‌ها به میزان چشمگیری کاهش پیدا می‌کند. اضافه‌بار سبب گم‌شدن بسته‌های OK مربوط به تماس‌های گذرداده شده می‌شود. در نتیجه، پروکسی مجبور به ارسال مجدد درخواست INVITE مربوط به بسته‌های گم‌شده خواهد بود. بالارفتن نرخ ارسال مجدد در پروکسی در این حالت، سبب خواهد شد پروکسی بخش زیادی از وقت خود را صرف ارسال مجدد درخواست‌های مربوط به تماس‌های در جریان کند و نرخ گذردهی آن، افت قابل توجهی داشته باشد.

جدول (1): نتایج به‌دست‌آمده در حضور و عدم حضور مکانیزم کنترل اضافه‌بار پیشنهادی و الگوریتم‌های [19] و [25]

میزان بار (CPS)	200	400	600	800	1000	1200	1400	1600
گذردهی مکانیزم پیشنهادی	200	400	600	640	650	655	655	655
گذردهی مکانیزم win-cont	200	400	600	640	600	600	600	600
گذردهی مکانیزم win-disc	200	400	600	635	595	600	590	600
گذردهی مکانیزم win-auto	200	400	600	640	590	595	590	590



شکل (6): نرخ ارسال مجدد پیام‌ها در مکانیزم کنترل اضافه‌بار پیشنهادی

در جدول 1 نیز به‌صورت جامع و خلاصه می‌توانید گذردهی روش پیشنهادی در مقایسه با مکانیزم‌های مبتنی بر پنجرهٔ دیگر را مشاهده کنید که حاکی از عملکرد بهتر روش پیشنهادی با افزایش بارِ تحمیلی به پروکسی سرور است.

5. نتیجه‌گیری و پیشنهادات

حال که در کشور ما NGN به‌طور کامل پیاده‌سازی نشده است، باید در هنگام گسترش آن با نگاهی به IMS، بسترهای مورد نیاز آن را بیشتر مدّ نظر قرار داد تا بتوان در آینده، به فناوری IMS سریع‌تر و بهتر دست پیدا کرد. پروتکل SIP در مواجهه با ازدحام چندان کارآمد نیست، به طوری که با بالارفتن نرخ درخواست تماس، تأخیر ایجاد تماس به‌طور ناگهانی افزایش یافته، گذردهی پروکسی افت و در نهایت، نرخ ارسال مجدد پیام‌ها و در نتیجه، تماس‌های ناموفق افزایش می‌یابد. در این مقاله، روش کنترل بار مبتنی بر پنجره به‌طوری که در آن، تعداد پیام‌های تأیید برای تشخیص اضافه‌بار، در نظر گرفته می‌شود، در یک بستر واقعی توسعه، پیاده‌سازی و آزمایش شد و کارایی این مکانیزم بر روی پروکسی متن باز Asterisk، مورد بررسی قرار گرفت. بررسی نمودارهای گذردهی، تأخیر و نرخ ارسال مجدد نشان می‌دهد که این الگوریتم بر روی پروکسی Asterisk توانسته است به‌خوبی گذردهی را تا حداکثر دو برابر ظرفیت پروکسی پایین‌دست حفظ کند. در کارهای آتی، علاقه به مطالعهٔ استراتژی‌های پیچیده‌تر برای به‌روزکردن اندازهٔ پنجره و نیز درنظرگرفتن عملگرهای پیشرفته‌ترِ تشخیص اضافه‌بار را داریم. علاوه بر این، می‌توان یک مدل تحلیلی از شبکهٔ SIP به‌دست آورد و تحلیل پایداری آن را نیز انجام داد.

سپاسگزاری

این کار با پشتیبانی مادی و معنوی آزمایشگاه تأیید نمونهٔ تجهیزات IP-PBX دانشگاه فردوسی مشهد انجام شده است. بدین‌وسیله از اعضای این آزمایشگاه کمال تشکر به‌عمل می‌آید.

■

مراجع

1. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M.Handley, E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, December, 2002.
2. J. Rosenberg, "Requirements for management of overload in the session initiation protocol", RFC5390, December, 2008.
3. V. Hilt. I. Widjaja, "Controlling overload in networks of sip servers", in: *IEEE International Conference on Network Protocols*, 2008, pp. 83–93.
4. E.C. Noel, C.R. Johnson, "Initial simulation results that analyse sip based VoIP networks under overload", *International Teletraffic Congress* (2007) 54–64.
5. M. Homayouni, S. Azhari, M. Jahanbakhsh, A. Akbari, A. Mansoori, N. Amani, "Configuration of a sip signaling network: an experimental analysis", in: *Fifth International Joint Conference on INC, IMS and IDC* 2009, pp. 76 –81.
6. M. Ohta, "Performance comparisons of transport protocols for session initiation protocol signaling", in: *Telecommunication Networking Workshop on QoS in Multiservice IP Networks*, 2008, pp. 148–153.
7. K.K. Ram, I.C. Fedeli, A.L. Cox, S. Rixner, "Explaining the impact of network transport protocols on sip proxy performance", in: *ISPASS '08: Proceedings of the ISPASS* 2008 – *IEEE International Symposium on Performance Analysis of Systems and software, IEEE Computer Society*, Washington *DC*, USA, 2008, pp.75–84.

8. S. Low, F. Paganini, J. Doyle, "Internet congestion control", *IEEE Control Systems Magazine* 22 (1) (2002) 28–43.
9. The Asterisk Project, <http:// http://www.asterisk.org> last checked may 2011.
10. V. Hilt, E. Noel, C. Shen, A. Abdelal, "Design consideration for session initiation protocol (sip) overload control", SOC working group, *Internet draft,draft-ietfsoc-overload-design*, Work in progress (May 2011).
11. S. Montagna, M. Pignolo, "Performance evaluation of load control techniques in sip signaling servers", in: *ICONS*, 2008, pp. 51–56.
12. M. Ohta, "Overload control in a sip signaling network", *Transactions on Engineering Computing and Technology 12* (2006) 2006.
13. M. Ohta, "Overload protection in a sip signaling network", in: *International Conference on Internet Surveillance and Protection*, 2006, pp. 11–11.
14. R.G. Garroppo, S. Giordano, S. Spagna, S. Niccolini, "Queueing strategies for local overload control in sip server", in: *Proceedings of the 28th IEEE Conference on Global Telecommunications, GLOBECOM'09, IEEE Press*, Piscataway, NJ, USA, 2009, pp. 466–471.
15. Y. Hong, C. Huang, J. Yan, "Mitigating sip overload using a control-theoretic approach", in: *Global Telecommunications Conference (GLOBECOM 2010), IEEE*, 2010, pp. 1–5.
16. T.-Y. Chi, C.-H. Chen, H.-C. Chao, S.Y. Kuo, "An efficient earthquake early warning message delivery algorithm using an in time control-theoretic approach", in: Ubiquitous Intelligence and Computing, *Lecture Notes in Computer Science*, 6905, Springer, Berlin, Heidelberg, 2011, pp. 161–173.
17. J. Sun, J. Hu, R. Tian, B. Yang, "Flow management for sip application servers", in: *ICC*, 2007, pp. 646–652.
18. J. Sun, H. Yu, W. Zheng, "Flow management with service differentiation for sip application servers", in: *CHINAGRID '08: Proceedings of the The Third ChinaGrid Annual Conference, IEEE Computer Society*, Washington, DC, USA, 2008, pp. 272–277.
19. C. Shen, H. Schulzrinne, E. M. Nahum, "Session initiation protocol (sip) server overload control: design and evaluation", in: *IPTComm*, 2008, pp. 149– 173.
20. S. Montagna, M. Pignolo, "Comparison between two approaches to overload control in a real server: local or hybrid solutions", in: *Proceedings of the 15th IEEE Mediterranean Electrotechnical Conference*, MELECON 2010, IEEE Press, Piscataway, NJ, USA, 2010, pp. 845–849.
21. E. Noel, C. Johnson, "Novel overload controls for sip networks", in: *21st International Teletraffic Congress* 2009, pp. 1–8.
22. R.G. Garroppo, S. Giordano, S. Niccolini, S. Spagna, "A prediction-based overload control algorithm for sip servers", in: *IEEE Transactions on Network and Service Management*, IEEE Press, Piscataway, NJ, USA, vol. 8, 2011, pp. 39–51.
23. A. Abdelal, W. Matragi, "Signal-based overload control for sip servers", in: *Proceedings of the 7th IEEE Conference on Consumer Communications and Networking Conference, CCNC'10*, IEEE Press, Piscataway, NJ, USA, 2010, pp. 990–996.
24. D.-M. Chiu, R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance" in *computer networks, Computing Network ISDN System* 17 (1) (1989) 1–14.
25. C.Shen, H.Schulzrinne. "Application Layer Feedback-based SIP Server Overload Control", *Columbia University Computer Science Technical Report* CUCS-031-08, June 2008.
26. Azhari SV, Homayouni M, Nemati H, Enayatizadeh J, Akbari A. "Overload control in SIP networks using no explicit feedback: a window based approach". *Comput Commun* 2012;35 (12):1472–83.
27. نعمتی، ازهری، «مدل‌سازی روش پنجره در کنترل اضافه‌بار پروکسی‌های SIP»، *سومین کنفرانس مهندسی برق ایران*، دانشگاه آزاد اسلامی گناباد، تیرماه 1390.
28. عنایتی‌زاده، ازهری، «پیاده‌سازی روش مبتنی بر پنجره در پروکسی OpenSIPS»، *سومین کنفرانس مهندسی برق ایران*، دانشگاه آزاد اسلامی گناباد، تیرماه 1390.