شناسایی جریان‌های فیل‌آسا در شبکه‌ها با الگوریتم برخط، تصادفی و موازی

سید محمد رضوی، دانشجوی دکتری دانشکده مهندسی کامپیوتر، دانشگاه فردوسی ، مشهد
seyyed.razavi@stu-mail.um.ac.ir

علی معینی، دانشیار دانشکده علوم مهندسی، پردیس دانشکده های فنی، دانشگاه تهران، تهران،
moeini@ut.ac.ir

چکیده

این مقاله به مساله شناسایی جریان‌های فیل‌آسا در شبکه‌های کامپیوتری می‌پردازد. به دلیل اینکه شناسایی این نوع از جریان‌ها باید به صورت برخط انجام شود، ارائه الگوریتمی بسیار کارا و سریع برای شناسایی این جریان‌ها بسیار ضروری می‌باشد. الگوریتم ارائه شده دارای ساختاری موازی تصادفی است. به عبارت دیگر برای حل مساله از رویکرد تصادفی بر مبنای الگوریتم موازی با معماری هرم استفاده شده است. مرتبه زمانی الگوریتم ارائه شده در مرحله تحلیل بسته‌ها $O(1)$ و در مرحله ادغام نتایج و بازنشانی ساختمان داده $O(lg\ n)$ است. بهبود عملکرد الگوریتم چه از نظر کاهش میزان خطای شناسایی جریان‌های فیل‌آسا چه از نظر کاهش فرکانس ساز و کار بازنشانی با آزمایش بر روی داده‌های نمونه مورد ارزیابی و تایید قرار گرفته است.

کلمات کلیدی

جریان‌های فیل‌آسا، الگوریتم‌های موازی، الگوریتم‌های تصادفی، شبکه کامپیوتری

۱- مقدمه

در شبکه‌های کامپیوتری، حرکت دنباله‌ای از بسته‌ها از یک کامپیوتر به کامپیوتر دیگر را جریان شبکه می‌گویند. ویژگی مهم این جریان‌ها وجود دو نوع خاص از جریان‌ها در میان آن‌ها می‌باشد[1]، جریان‌های بسیار کم فیل-آسا[1] و جریان‌های بسیار زیاد موش‌آسا[2][2]. یکی از اهداف اصلی تحقیقات بر روی شناسایی نوع جریان‌های یک شبکه، مخصوصا جریان‌های فیل‌آسا، مدیریت بهتر و کارآمدتر پهنای باند و پیش‌بینی مدل مصرف در یک شبکه کامپیوتری می‌باشد[3,4] . هدف اصلی دیگر شناسایی جریان‌های انکار سرویس[3] در شبکه‌های کامپیوتری می‌باشد[5,6] . برای مثال، مدیران شبکه تمایل دارند تا در شبکه خود به جریان‌های موش‌آسا اولویت بیشتری نسبت به جریان‌های فیل‌آسا برای جریان در داخل شبکه بدهند، بنابراین بسیار ضروری است که بتوان جریان‌های فیل‌آسای یک شبکه را به صورت برخط و بسیار سریع شناسایی کرد[7].

الگوریتم‌های شناسایی جریان‌های فیل‌آسا در یک شبکه بر اساس ویژگی‌ها و خصوصیات ترافیک شبکه‌های کامپیوتری طراحی شده‌اند. برای مثال برای لینک‌هایی که حجم بسیار زیادی از داده را انتقال می‌دهند (40 GB/s in OC-768) الگوریتم‌های مقیاس‌پذیر، کارآمد، بسیار سریع و برخط مورد نیاز است [8].

به طور کلی برای شناسایی جریان‌های فیل‌آسا در یک شبکه کامپیوتری انواع مختلفی از الگوریتم‌ها ارائه شده است، از جمله الگوریتم‌های طبقه‌بندی [9,10]، الگوریتم‌های نمونه‌برداری [11,12] و الگوریتم‌های شمارشی [13,14]. در اکثر الگوریتم‌های شمارشی، شمارنده ای وجود دارد که به عنوان معیاری برای شناسایی جریان‌های فیل‌آسا عمل می‌کند. به دلیل نرخ بالای انتقال جریان و تعداد بسیار زیاد جریان‌های رد و بدل شده در یک شبکه

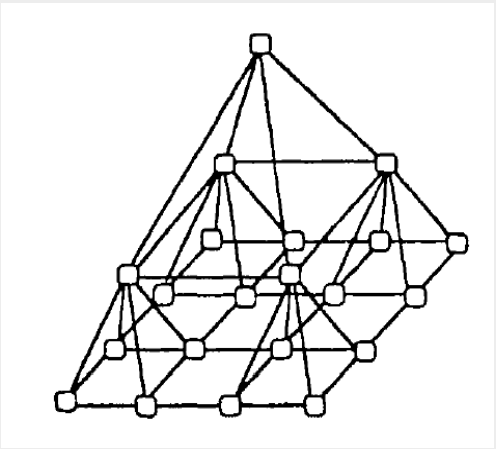
کامپیوتری، نگهداری لیست جریان‌های جاری و بروزرسانی شمارنده‌ها یکی از چالش‌های مهم در حل مساله شناسایی جریان‌های فیل‌آسا می‌باشد.
Estan و Verghase در [15] الگوریتمی را بر اساس فیلتر بلوم [۴] ارائه کردند. این الگوریتم به اندازه کافی سریع می‌باشد و از حافظه محدودی برای شناسایی جریان‌های فیل‌آسا استفاده می‌کند اما برای ترافیک‌های مختلف مناسب نمی‌باشد به این معنی که در آن از پارامتر ثابتی متناسب با ترافیک جریان استفاده شده است که ممکن است در هر ترافیکی متفاوت باشد. Chabchoub و بقیه در [16] بهبودی برای این الگوریتم با اضافه کردن ساز و کار بازنشانی[۵] ارائه دادند. ایده اصلی این الگوریتم به این صورت می‌باشد: فیلتر استفاده شده از d مرحله تشکیل شده است. هر مرحله شامل m شمارنده می‌باشد. زمانی که بسته ای می‌رسد IPی هدر بسته توسط d تابع مستقل، درهم‌سازی می‌شود، سپس بر اساس مدل سوپر مارکت کوچکترین مقدار از میان مقادیر درهم‌سازی شده انتخاب و شمارنده متناظر با آن در هر مرحله یک واحد اضافه می‌شود. زمانی‌که شمارنده به مقدار k می رسد (کوچکترین مقدار برای اینکه یک جریان به عنوان یک جریان فیل‌آسا شناسایی شود) جریان متناظر با آن به عنوان فیل‌آسا شناسایی می‌شود. با توجه به سنگین بودن ترافیک‌های شبکه‌ای، فیلتر نیاز دارد تا در برخی اوقات بازنشانی شود، در غیر اینصورت پس از مدتی تمام شمارنده‌ها مقدارشان از K تجاوز کرده و از آن به بعد هر بسته‌ای که بیاید مقدار شمارنده‌ها را افزایش داده و به عنوان جریان فیل‌آسا به اشتباه شناسایی می‌شود در حالیکه اولین بسته از یک جریان دیگر می‌باشد. ایده اصلی برای مقابله با این مشکل استفاده از ساز و کار بازنشانی است به این صورت که زمانیکه نسبت شمارنده‌های غیر تهی به نسبت کل شمارنده‌ها از مرز r عبور کرد تمام شمارنده‌ها یک واحد کاهش می‌یابد. این روند تا زمانیکه نسبت فوق کمتر از r شود ادامه می‌یابد. در این روش فرکانس تکرار ساز و کار بازنشانی به شدت به چگالی ترافیک شبکه وابسته می‌باشد. به عبارت دیگر هر چقدر ترافیک شبکه سنگین‌تر باشد تعداد دفعات اجرای ساز و کار بازنشانی بیشتر می‌شود.
در این مقاله، شناسایی جریان‌های فیل‌آسا به صورت موازی ارائه می‌شود به این صورت که اگر بخشی از الگوریتم درگیر انجام ساز و کار بازنشانی است، بخش دیگر به تحلیل بسته‌ها ادامه می‌دهد و به این صورت تعداد بسته‌های از دست رفته را به حداقل کاهش می‌دهد.
ساختار کلی مقاله به صورت زیر می‌باشد: بعد از این مقدمه، در قسمت بعدی به معماری موازی ارائه شده برای الگوریتم خواهیم پرداخت. در قسمت بعد از آن الگوریتم را ارائه خواهیم کرد و در قسمت آخر نیز الگوریتم را بر روی داده-های نمونه اجرا و نتایج مقایسه با کارهای قبلی را ارائه خواهیم کرد.

۲- آلگوریتم شناسائی جریان های فیل آسا

معماری استفاده شده برای الگوریتم موازی ارائه شده معماری هرم می‌باشد. هرم در واقع پشته‌ای از صفحات است که بر روی یکدیگر قرار می‌گیرند، اولین صفحه هرم قاعده نامیده می‌شود. تعداد عناصر پردازشی صفحات بعدی هر کدام نصف صفحه قبلی است و در نهایت صفحه آخر شامل فقط یک عنصر پردازشی است. آخرین سطح هرم که شامل یک عنصر محاسباتی است قله یا نوک هرم نامیده می‌شود. این ساختار معماری بسیار مناسب برای انجام اعمال جداگانه و ترکیب نتایج می‌باشد و با توجه به اینکه دورترین فاصله عناصر پردازشی در این معماری حداکثر $O(\log n)$ می‌باشد که در آن n تعداد کل عناصر پردازشی در هرم است، ارتباطات بین عناصر نیز به صورت کارا صورت می‌گیرد [17]. از طرف دیگر با توجه به ساختار سلسله مراتبی بودن این معماری، هر سطح می‌تواند به عنوان ترکیب کننده و تحلیل کننده نتایج به دست آمده از سطح پایین خود باشد. شکل (۱) شمای کلی این معماری را نشان می‌دهد.

۲-۱- معرفی الگوریتم

هدف اصلی استفاده از معماری هرم برای پیاده‌سازی الگوریتم تصادفی موازی برای شناسایی جریان‌های فیل‌آسا، استفاده از ساختار سلسله مراتبی این معماری می‌باشد. ایده اصلی الگوریتم ارائه شده به این صورت می‌باشد که بر روی هر عنصر پردازشی یک فیلتر با d مرحله و d تابع درهم‌سازی قرار دارد. به ازای هر بسته‌ای که می‌رسد IP مقصد بسته استخراج شده و توسط d تابع درهم‌سازی می‌شود و شمارنده متناظر با آن در فیلتر یک واحد افزایش می‌یابد. ساز و کار بازنشانی نیز مانند آنچه در بالا گفته شد عمل می‌کند فقط با این تفاوت که زمانیکه یک عنصر پردازشی درگیر عملیات بازنشانی می‌باشد پردازنده‌های دیگر بسته‌های ورودی را دریافت و تحلیل می‌کنند. با توجه به وجود چندین عنصر پردازشی احتمال همزمانی ساز و کار های بازنشانی کاهش می‌یابد و لذا تعداد بسته کمتری از دست می‌رود. این امر باعث می‌شود تا تعداد جریان‌های فیل‌آسا دقیق‌تر و با خطای کمتری تعیین شوند.



شکل (۱) : هرم با قاعده 4×4 عنصر پردازشی

شبه کد الگوریتم در شکل (۲) آورده شده است. ساختار الگوریتم به این صورت می‌باشد که هر سطح به فیلترهای سطح پایینی خود و هر عنصر پردازشی به فیلترهای عنصرهای پردازشی هم سطح خود دسترسی دارد. این رویکرد این امکان را می‌دهد تا طول فیلتر را بزرگ‌تر در نظر گرفت و آن را به بخش‌های مساوی میان عناصر پردازشی قاعده هرم تقسیم کرد. ویژگی مهم این روش در این می‌باشد که احتمال خطا در تعیین جریان‌های فیل‌آسا را به دو دلیل کاهش می‌دهیم: (۱) چون طول فیلتر در نظر گرفته شده بیشتر است لذا خود خطای فیلتر بلوم کاهش می‌یابد (۲) چون در حین انجام ساز و کار بازنشانی بسته‌ها توسط عنصر پردازشی دیگر تحلیل می‌شوند خطای از دست دادن بسته کاهش می‌یابد. در شبه کد بالا ساختار عناصر پردازشی هرم را به صورت ۳ بعدی در نظر گرفته شده است یعنی هر سطح هرم به صورت دو بعدی و سطوح هرم به عنوان بعد سوم در نظر گرفته شده است.

```
1.  IdentifyElephants ():
2.  begin
3.  for P_{i,j,k} where k = 0   do  //{processors in base level}
4.        get received packet
5.        fetch it's header IP
6.        hash IP with d hash_functions
7.        increase appropriate counters in filter
8.        if appropriate counter ≥ k do
9.            mark flow as a elephant
10.       end for
11.   end for
12.   for P_{i,j,k}  do
13.       if 1 ≤ k ≤ lg n  do
14.           if P_{i,j,k} needs refreshment mechanism do
15.               parent(P_{i,j,k}) make refreshment mechanism
16.           end if
17.       end if
18.   end for
19.   end
```

شکل (۲) : شبه کد الگوریتم

در خطوط ۳ تا ۸ برای قاعده هرم عملیات پردازش بسته‌ها و اضافه کردن شمارنده‌های متناظر آن در فیلتر انجام می‌شود. در خطوط ۹ تا ۱۵ در سطوح بالاتر از قاعده چک می‌شود که اگر عنصر پردازشی سطح پایین‌تر نیاز به ساز و کار بازنشانی داشته باشد این ساز و کار توسط عنصر والدش در سطح بالاتر انجام می‌شود.

۲-۲ تحلیل الگوریتم

روند انتصاب بسته‌های دریافتی به عناصر پردازشی قاعده به این صورت می‌باشد که عنصر پردازشی که مقدار r فیلتر آن کوچکترین مقدار باشد هر بار انتخاب می‌شود و برای r های با مقدار یکسان یک کدام به صورت تصادفی انتخاب می‌شود که r نسبت خانه‌های غیر تهی به کل تعداد خانه‌های فیلتر می‌باشد.

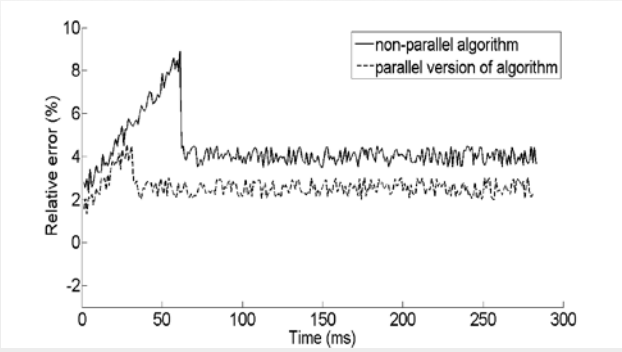
تحلیل زمانی الگوریتم به این صورت می‌باشد که برای هر بسته دریافتی در هر عنصر پردازشی زمان $O(1)$ برای تحلیل بسته و درهم‌سازی IP مقصد و افزایش شمارنده متناظر آن مورد نیاز است. برای ساز و کار بازنشانی نیز زمان $O(\lg n)$ با توجه به ساختار هرمی معماری و متناسب با ارتفاع هرم زمان نیاز می‌باشد.

۲-۳ آزمایشها

برای آزمایش الگوریتم ارائه شده از داده‌ها و کتابخانه libtrace که توسط دانشگاه وایکاتو[۶] ارائه شده است، استفاده می‌شود[18]. هرم آزمایش دو سطحی با ۴ عنصر پردازشی در قاعده است. طول فیلتر در هر عنصر پردازشی موسوم به m برابر $m = 2^{20}$، تعداد توابع درهم‌سازی موسوم به d برابر $d = 2$ ، مقدار آستانه شناسائی موسوم به k برابر $k = 20$، و نهایتا مقدار $r = 50\%$ در نظر گرفته شده است.

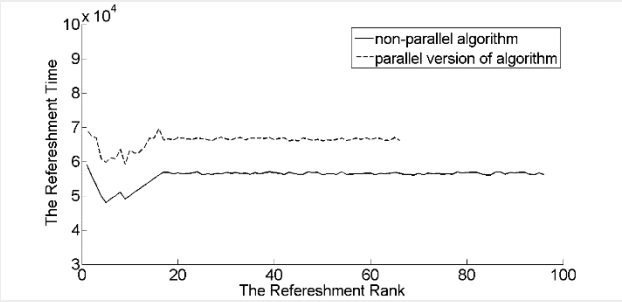
شکل (۳) نتیجه اجرای الگوریتم بر روی ترافیک [19] ارائه نشان می دهد. در این آزمایش میزان خطای الگوریتم ارائه شده با الگوریتم ارائه شده در [16] از جنبه میزان خطای موجود در شناسایی جریان‌های فیل‌آسا مورد بررسی قرار گرفته است. تاثیر استفاده از الگوریتم موازی مبتنی بر ساختار هرم که همان کاهش تعداد بسته‌های از دست رفته است به وضوح در آزمایش دیده می‌شود.

دلیل خطای کمتر هم موازی بودن عملیات پردازش بسته‌ها و انجام ساز و کار بازنشانی توسط عنصر پردازشی والد می‌باشد.



شکل (۳) : تاثیر موازی سازی بر کاهش میزان خطای شناسایی جریان‌های فیل‌آسا با $r = 50\%$

برای روشن شدن اثر الگوریتم موازی ارائه شده در کاهش فرکانس ساز و کار بازنشانی در طول اجرای الگوریتم، عملکرد الگوریتم ارائه شده با عملکرد الگوریتم بدون موازی سازی در شکل (۴) مقایسه شده است. در این آزمایش تعداد ساز و کار های بازنشانی بر اساس زمان آن‌ها برای هر دو الگوریتم ارائه شده است. همانطور که در شکل نیز مشخص است در الگوریتم موازی ارائه شده با توجه به اینکه طول فیلتر به صورت مجموع می‌تواند به صورت بزرگتر در نظر گرفته شود لذا علاوه بر کاهش خطای احتمالاتی فیلتر فرکانس ساز و کار بازنشانی در حالت کلی نیز کاهش می‌یابد.



شکل (۴) : تاثیر موازی سازی بر کاهش فرکانس ساز و کار بازنشانی در طول اجرای الگوریتم (زمان محاسبه شده در محور عمودی برچسب زمانی سیستم عامل لینوکس می‌باشد)

۳- نتیجه

در این مقاله الگوریتمی تصادفی موازی برای شناسایی جریان‌های حجیم به صورت آنلاین ارائه شده است. این الگوریتم بر مبنای استفاده از بلوم فیلتر به همراه ساز و کار بازنشانی می‌باشد. معماری موازی استفاده شده برای این الگوریتم ساختار هرم می‌باشد. دلیل استفاده از این ساختار سلسله مراتبی بودن ساختار می‌باشد و این ویژگی برای اعمال ساز و کار بازنشانی بسیار مناسب می‌باشد. روند کار به این صورت است که هر والد وظیفه انجام عمل بازنشانی فرزندان خود در سطح پایین‌تر را دارد. کارآیی الگوریتم ارائه شده در آزمایش از دو نظر کاهش میزان خطای شناسایی جریان‌های فیل‌آسا و کاهش فرکانس عملیات بازنشانی مورد ارزیابی و مقایسه با نسخه غیر موازی الگوریتم قرار گرفته است.

مراجع

[1] Shao, Yiyang, et al. "Emilie: Enhance the power of traffic identification. “Computing, Networking and Communications (ICNC), 2014 International Conference on. IEEE, 2014.
[2] Li, Qiang, et al. "Empirical analysis and comparison of IPv4-IPv6 traffic: A case study on the campus network." Networks (ICON), 2012 18th IEEE International Conference on. IEEE, 2012.
[3] Mori, Tatsuya, et al. "Identifying elephant flows through periodically sampled packets." Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. ACM, 2004.
[4] Lu, Yi, et al. "Elephant Trap: A low cost device for identifying large flows." High-Performance Interconnects, HOTI 2007. 15th Annual IEEE Symposium on. IEEE, 2007.
[5] Curtis, Andrew R., Wonho Kim, and Praveen Yalagandula. "Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection." INFOCOM, 2011 Proceedings IEEE. IEEE, 2011.
[6] Antikainen, M.; Aura, T.; Sarela, M., "Denial-of-Service Attacks in Bloom-Filter-Based Forwarding," Networking, IEEE/ACM Transactions on, vol.22, 2014, 1463-1476.
[7] Kun-chan Lan, John Heidemann, A measurement study of correlations of Internet flow characteristics, Computer Networks, Volume 50, 2006, Pages 46-62.
[8] Megyesi, Péter, and Sándor Molnár. "Analysis of Elephant Users in Broadband Network Traffic." Advances in Communication Networking. Springer Berlin Heidelberg, 2013. 37-45.
[9] Wu, Di, et al. "On Addressing the Imbalance Problem: A Correlated KNN Approach for Network Traffic Classification." Network and System Security. Springer International Publishing, 2014. 138-151.
[10] Tammaro, Davide, et al. "Exploiting packet-sampling measurements for traffic characterization and classification." International Journal of Network Management 22.6 (2012): 451-476.
[11] Wang, Xiaoming, Xiaoyong Li, and Dmitri Loguinov. "Modeling residual-geometric flow sampling." IEEE/ACM Transactions on Networking (TON) 21.4 (2013): 1090-1103.
[12] Wang, Xiaoming, Xiaoyong Li, and Dmitri Loguinov. "Modeling residual-geometric flow sampling." IEEE/ACM Transactions on Networking (TON) 21.4 (2013): 1090-1103.
[13] Bai, Lei, and Dong Min Li. "Using Sample and Multilayer Compressed Counting Bloom Filter Algorithm to Realize Elephant Flows Identification." Applied Mechanics and Materials. Vol. 651. 2014.
[14] Li, Tao, Shigang Chen, and Yibei Ling. "Fast and compact per-flow traffic measurement through randomized counter sharing." INFOCOM, 2011 Proceedings IEEE. IEEE, 2011.
[15] C. Estan and G. Varghese, "New directions in traffic measurement and accounting," in Proc. ACM Sigcomm, Pittsburgh, Pennsylvania, USA, August 19-23 2002.
[16] Chabchoub, Yousra, Christine Fricker, and Hanene Mohamed. "Analysis of a Bloom filter algorithm via the supermarket model." Tele traffic Congress, 2009. ITC 21 2009. 21st International. IEEE, 2009.
[17] Bongiovanni, G., Concettina Guerra, and Stefano Levialdi. "Computing the Hough transform on a pyramid architecture." *Machine Vision and Applications*3.2 (1990): 117-123.
[18] Alcock, S., Lorier, P., & Nelson, R. (2012). Libtrace: a packet capture and analysis library. ACM SIGCOMM Computer Communication Review, 42(2), 42-48.
[19] The trace “ISPDSL-I/20090105-220000-0.dsl”, URL: http://wand.net.nz/wits/ispdsl/1/20090105-215411-0.dsl.php, Waikato internet traffic storage.

زیرنویس‌ها

[1] Elephants
[2] Mice
[3] Denial of Service (DoS)
[4] Bloom Filter
[5] Refreshment Mechanisms
[6] University of Waikato