In Network Volumetric DDoS Victim Identification using Programmable Commodity Switches

IEEE Transactions on Network & System Management 2021

Damu Ding(anomaly detection,P4) Macro Savi(NFV, Network Function)

رویکرد روش‌های شناسایی حملات منع خدمت می‌تواند چندین نوع باشد، مثلا: مشاهده کاهش آنتروپی برای آدرس‌های مقصد متمایز یا محاسبه کاردینالیتی جریان‌های متمایز ارتباط برقرار کننده با یک آدرس مقصد مشخص. روش دوم علاوه بر شناسایی حملات منع خدمت، می‌تواند به شناسایی قربانیان نیز کمک کند. مدل حمله ما حملات منع خدمت حجیم می‌باشد که تمییز آنها از flash crowd ها دشوار می‌باشد. یکی از راه‌های تمایز بین آنها، بررسی ارتباط بین جریان‌هاست. روشهایی شناسایی قبلی به یک مکان متمرکز اطلاعات ترافیک را ارسال می‌کردند و در آنجا عملیات مانیتورینگ و بازخورد[1] انجام می‌شد، که این عملیات تاخیر زیادی به همراه داشت. یک دسته از راهکارهای جدید، DataPlaneProgrammibility می‌باشد که تنها ترافیک را برای بررسی عمیق تر به کنترلر می‌فرستد و خود سوییچها بر اساس آستانه از پیش تعریف می‌توانند تصمیم‌گیری کنند. این مقاله روشی به نام INDDoS ارایه می‌دهد که بر مبنای یک داده ساختار اسکچ به نام BACON می‌باشد. اطلاعات جریانها به مقصدهای مختلف را ذخیره می‌کندو تنها زمانی که کاردینالیتی یک مقصد[2] از مقدار آستانه بیشتر شد، به کنترلر اطلاع می‌دهد. سخت‌افزار برنامه‌پذیر Tofino و زبان خاص منظوره p4 استفاده می‌کند. زبان p4 نیز از یک سری عملیات زمان‌بر پشتیبانی نمی‌کند لذا عملیات روی هر بسته محدود می‌باشد. کد p4 را نیز روی سوییچهای واقعی پیاده می‌کند، برخلاف روش‌های قبلی که روی شبیه‌ساز Behavioural Model – p4 پیاده می‌شده است.

راهکارما: Bacon یک ساختار داده اسکچ می‌باشد که از direct bitmap و Count-Min Sketch استفاده می‌کند تا تعداد جریانات منحصر به فرد به سمت یک مقصد را تخمین می‌زند. حالا از مقایسه این مقادیر با آستانه IP های قربانی در هر دوره زمانی را شناسایی می‌کند.

Direct BitMap: یک آرایه بیتی است بر مبنای هش که هش کلید همه جریان‌ها را با استفاده از یک یا چندین تابع محاسبه می‌کند، و مقادیر خانه‌ها را برابر ۱ یا صفر قرار می‌دهد. از این ساختار برای شمارش جریانهای متمایز استفاده می‌شود.

Count-Min Sketch: d تا تابع هش متفاوت با اندازه خروجی w می‌باشد که برای تخمین سایز هر جریان استفاده می‌شود.

اگر مقدار تخمینی یک مقصد از مقدار آستانه بیشتر شد، یک digest آدرس مقصد را به کنترلر می‌فرستد. درآنجا نیز می‌تواند از روشهای رفع مخاطره مثل rate limit و یا drop استفاده کند. این فیلترها نیز روی خروجی egress سوییچها پیاده شده اند. در پایان هردوره زمانی نیز رجیسترها ریست می‌شوند. روش‌های دیگر mitigate بدون استفاده از سوییچهای برنامه پذیر می‌باشند:

- Bohatei: یک سری ماشین‌های مجازی بر روی سرور ها استفاده می‌کند.
- Poseidon؛ از سرورها هم به عنوان کمکی برای مقابله با انواع مختلف حملات منع خدمت استفاده می‌کند.

اما این دو روش مشکلشان اینست که فرض می‌کنند مهاجم از قبل شناسایی شده است (DDOS defense as Service)، اما INDDOS مکمل ایناست و قربانی را شناسایی می‌کند و حجم مدیریت شده ای از ترافیک را هدایت می‌کند.

---

[1] Mitigation
[2] Flow cardinality

پیاده‌سازی: در مقایسه با دیگر اسکچ‌ها، پیاده سازی الگوریتم‌هایمان بر روی ASIC از چیپ‌های دیگر NPU,NIC,FPGA بهتر می‌باشد. روش‌مان را بر روی لبه ISP پیاده می‌کنیم، چون حداقل یک سوییچ که به تمامی جریان‌های عبوری دسترسی داشته باشد. معماری سوییچ‌هایمان نیز به این صورت هست که داری چندین pipe بر روی هر پورت ورودی و خروجی (egress,ingress) می‌باشد. هر کدام از این خطوط لوله نیز یک سری stage دارند که بلاک‌های کدی هستند که مسیول انجام عملیات می‌باشند.

بررسی کراندار بودن میزان خطا:

یکی از قضایایی که در این مورد استفاده می‌شود، قضیه مارکو می‌باشد:

$$P(x \geq a) \leq \frac{E[x]}{a}$$

قسمت استفاده از آمار و احتمال برای اثبات کران دار بودن تخمین‌ها دقیق‌تر خوانده شود.

بررسی کارایی و ارزیابی دقت سیستم: همانطور که می‌دانیم انتخاب مقدار حد آستانه، دلیل اصلی کارایی سیستم می‌باشد.(این که از چه روشی برای دینامیک بودن آن نیز استفاده کنیم، نیز جای بحث دارد). می‌توان یک مدت ترافیک را مانیتور کرد و بهینه آستانه را یافت. یا از یک مقدار خیلی پایین شروع کرد و سپس مقاصد را پرچم متخاصم زد و به کنترلر بگوییم آن را بررسی کند و بر اساس آن مقدار آستانه را تنظیم کنیم. برای ترافیک واقعی حملات، از سرویس Booster استفاده می‌کنیم. یکی از معیارهای مهمی که اندازه می‌گیریم F1 می‌باشد که ترکیبی از recall (شناسایی FP+TP) و precision (مثل recall اما مثبت کاذب‌ها چقدر است) را محاسبه می‌کنیم:

Recall $$R = \frac{Count_{DDoSvictim}^{detected/true}}{Count_{DDoSvictim}^{detected/true} + Count_{DDoSvictim}^{undetected/true}}$$

Precision $$Pr = \frac{Count_{DDoSvictim}^{detected/true}}{Count_{DDoSvictim}^{detected/true} + Count_{DDoSvictim}^{detected/false}}$$

$$F1 = \frac{2 \cdot Pr \cdot R}{Pr + R}$$

پارامترهای اسکچ نیز به طور پیشفرض m=1024, w=1024 , d=3 می‌باشد. و آستانه نیز برابر Θ=0.5% و در آزمایش دوم برابر 1% می‌باشد.

کارایی روش‌مان در پردازش بسته‌ها با سرعت linerate را تایید می‌کنیم. همچنین دقت تشخیص بالا و مقادیر F1 بالای ۹۵ درصد، در صورتی که پارامترها به درستی تنظیم شده باشد در ترافیک ۱۰ گیگابیت بر ثانیه قابل دسترسی می‌باشد.

درصد خطای $\hat{E}_{dst}$ نیز به مقادیر w,m بستگی دارد. برای محاسبه تعداد ها نیز از CMS کمکی استفاده می‌کند.

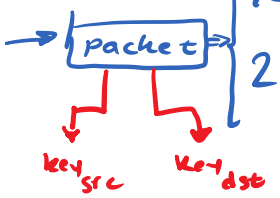
مقایسه: چون روش‌های دیگر مبتنی بر اسکچ، از الگوریتم‌ها و عملیاتی استفاده می‌کنند که بر روی سوییچ‌های Tofino نمی‌تواند پیاده شود، لذا روش‌مان را با آنها مقایسه نمی‌کنیم هر چند که از نظر عملکرد و پیچیدگی بهتر باشند.

برای مقایسه، BACON را با spread sketch که یک اسکچ سه بعدی می‌باشد، مقایسه کرده ایم.f1 ما از اون در شرایطی که تصادم نباشد، بهتر می‌باشد. روش ما از direct bitmap و اون از multi resolutional bitmap استفاده می‌کند. با netflow که آن نیز تشخیص حملات در کنترلر صورت می‌گیرد نیز مقایسه می‌کنیم. تاخیر را نیز محاسبه کرده‌ایم، در روش ما چون تنها آی‌پی را می‌فرستیم، کمتر می‌باشد در حالی که در spread sketch کل اسکچ را برای پردازش بیشتر به کنترلر ارسال می‌کند. از ابزار iperf نیز به منظور تولید ترافیک برای بررسی تاخیر پردازشی بسته‌ها استفاده می‌کنیم.

در پایان این مقاله و پژوهش به شناسایی و کاهش حملات DDoS، که یکی از اهداف پروژه GN4-3 می‌باشد-سرمایه موردنیاز نیز توسط این پروژه فراهم شده است،برای ارتقاء شبکه GEANT سراسر اروپا کمک می کند.

طریقه کار الگوریتم

Insert:



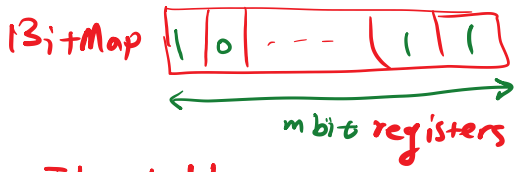
1_ $hb_m(key_{src})$: bucket i of BitMap BACON :

2_ Foreach i in d:

$1 = [h^i_{cm}(key_{dst}) \% w]$ (m+bucket)

Query: $\hat{E}_{dst} = \min(E_{i,j})$

E_i = Count_ones



S : بیت دوری

m: سایز Bitmap

d :cms تعداد توابع هش

w : cms سایز دوبعدی

T_{int} : interval timeout

Θ : Threshold

$BACON_i$: ith row of BACON

n : distinict source IPs

E_{dst} : تعداد جریان های یکتا با key_{src} که به یک key_{dst} رفته اند