دوفصل‌نامه علمی منادی امنیت فضای تولید و تبادل اطلاعات (افتا)
سال ۱۴۰۰، جلد ۱۹، شماره ۱ (۱-۹)
https://monadi.isc.org.ir

ارائه‌شده در شانزدهمین کنفرانس بین المللی انجمن رمز ایران در دانشگاه فردوسی مشهد، ۶ و ۷ شهریور ۱۳۹۸

فانوس: راهکار مقابله با حملات انگشت‌نگاری وب‌سایت★

سعید شیروی*، امیرمهدی صادق‌زاده و رسول جلیلی

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران

اطلاعات مقاله

کلمات کلیدی:
سازوکارهای ارتقا حریم خصوصی
تُر
گمنامی
تحلیل ترافیک
انگشت‌نگاری وب‌سایت
شبکه عصبی عمیق
آزمایش انسداد

dor: 10.0000/000000000

نوع مقاله: پژوهشی

چکیده

حملات انگشت‌نگاری وب‌سایت از جمله حملات تحلیل ترافیک هستند که مهاجم با نظارت بر ترافیک کاربران به شناسایی فعالیت وب آنان می‌پردازد. این حملات حتی زمانی‌که کاربران از سازوکارهای ارتقا حریم خصوصی، مانند شبکه تُر بهره برده باشند نیز موثرند. تحقیقات اخیر نشان داده‌اند که مهاجم با استفاده از شبکه عصبی عمیق، قادر است با دقت ٪۹۸، وب‌سایت‌های بازدید شده توسط کاربر را شناسایی کند. این درحالیست که سازوکارهای ارائه شده به منظور مقابله با این حملات، یا سربار پهنای باند و زمانی بالایی به کاربران تحمیل می‌کنند یا آنکه در مقابل حملات اخیر، عملاً موثر نیستند. در این مقاله ساز و کار دفاعی جدیدی بر اساس آزمایش انسداد معرفی خواهیم کرد. در روش پیشنهادی آنچه یک شبکه عصبی به عنوان الگو از داده‌ها برداشت می‌کند را شناسایی خواهیم کرد و بر این اساس، الگوی ترافیک شبکه را به گونه‌ای تغییر خواهیم داد که شبکه عصبی در دسته‌بندی ترافیک کاربران با خطا مواجه شود. این روش با کاهش دقت مهاجم از ٪۹۸ به ٪۱۹ تنها با سربار پهنای باند ٪۴۷ و بدون داشتن سربار زمانی، در مقابل حملاتی که از شبکه عصبی بهره برده‌اند کاملاً موثر است.

۱ مقدمه

مسیریاب پیازی (تُر) [1]، ابزاری ارتباطی به منظور ارتقای حریم‌خصوصی کاربران در فضای اینترنت است که به کاربران امکان گمنامی در ارتباطات را می‌دهد [۱]. برای دستیابی به این مقصود، تُر علاوه بر رمزنگاری محتوای ارتباطات، اطلاعات را میان گره‌های شبکه خود به صورت تصادفی جابجا می‌کند. به این ترتیب هر گره این شبکه، تنها از گره بعدی و قبلی خود آگاه است و هیچ گره‌ای به طور همزمان از مبدأ و مقصد ارتباطات اطلاع ندارد.

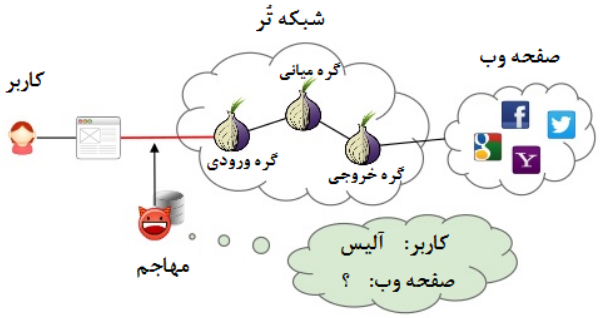
مطالعات اخیر نشان داده‌اند که این شبکه مستعد حملات تحلیل ترافیک، موسوم به حملات انگشت‌نگاری وب‌سایت [2] است که به موجب آن مهاجم، قادر به شناسایی وب‌سایت‌های بازدید شده توسط کاربر است. در این حملات، شنونده از طریق نظارت بر ترافیک ردوبدل شده کاربران و تطابق دادن این ترافیک با الگوی ترافیک وب‌سایت‌های از پیش ضبط شده به شناسایی فعالیت وب کاربران اقدام می‌کند. این حملات به عنوان تهدیدی جدی در مقابل سازوکارهای ارتقا حریم خصوصی مانند تُر هستند؛ سازوکارهایی که در تلاشند تا الگوی ترافیکی کاربران را مبهم سازند.

مهاجم در این حملات مابین کاربر و اولین گره شبکه تُر قرار می‌گیرد و هدف آن تشخیص وب‌سایت‌های بازدید شده توسط کاربر با استفاده

★از کمیته علمی شانزدهمین کنفرانس بین‌المللی انجمن رمز ایران برای داوری این مقاله تشکر می‌شود.
*نویسنده مسئول
آدرس‌های رایانامه: saeedshiravi@ce.sharif.edu (سعید شیروی)، amsadeghzadeh@ce.sharif.edu (امیرمهدی صادق‌زاده)، jalili@sharif.edu (رسول جلیلی)

[1] The Onion Routing (Tor)

[2] Website Fingerprinting



شکل ۱. مدل تهدید حملات انگشت‌نگاری وب‌سایت [۲]

از یک الگوریتم دسته‌بندی است. همانطور که در شکل ۱ مشخص است، در این حمله مهاجم از طریق شنود ارتباط بی‌سیم یا نفوذ به شبکه محلی و یا با داشتن دسترسی به ارائه‌کننده خدمات اینترنت [۱] کاربر، امکان شنود لینک ارتباطی مابین کاربر و شبکه تُر را دارد. تُر تمامی محتوای ارتباطی را رمزنگاری می‌کند؛ از این رو مهاجم تنها قادر به استخراج ویژگی‌های آماری است. طول بسته‌ها، تعداد بسته‌ها، جهت بسته‌ها، و زمان ما بین بسته‌ها از جمله این ویژگی‌ها هستند که به منظور دسته‌بندی ترافیک ردوبدل شده استفاده می‌شوند.

حملات انگشت‌نگاری وب‌سایت به دلیل هزینه پایین محاسباتی، نیاز به منابع کم، و ریسک پایین شناسایی، از جمله حملات خطرناک تحلیل ترافیک محسوب شده و کاربرانی که حریم خصوصی آنان برایشان حیاتی است نیاز به استفاده از روش مقابله با این حملات را دارند.

اخیراً حملاتی با استفاده از شبکه عصبی عمیق ارائه شده‌اند [۲، ۳] که علاوه بر حذف مرحله مهندسی ویژگی [۲] در مقایسه با مدل‌های پیشین مبتنی بر یادگیری ماشین [۴، ۵] دقت بالاتری را بدست آورده‌اند. در این مقاله با بررسی حملات مبتنی بر شبکه عصبی عمیق، سازوکار دفاعی برای مقابله با این حملات ارائه خواهد شد. روش‌های متعددی برای دفاع در مقابل حملات انگشت‌نگاری وب‌سایت تاکنون ارائه شده‌اند که حتی می‌توان به نمونه‌هایی از آنها که در شبکه تُر پیاده‌سازی شده‌اند اشاره کرد [۶، ۷]. اما این روش‌ها یا دارای سربار زمانی و پهنای باند بسیار بالایی بوده‌اند و یا آنکه در مقابل حملات اخیر مبتنی بر شبکه عصبی عمیق، کارا نیستند.

در این مقاله، روشی با نام فانوس (فریب‌دهنده انگشت‌نگاری وب‌سایت) پیشنهاد می‌شود که تنها با داشتن سربار پهنای باند ۴۷٪ و بدون سربار زمانی در مقابل حملات نوین انگشت‌نگاری وب‌سایت موثر بوده و دقت این حملات را از ۹۸٪ به ۱۹٪ کاهش می‌دهد. در ادامه این مقاله در بخش ۲، حملات انگشت‌نگاری وب‌سایت را بررسی خواهیم کرد. سپس در بخش ۳ به روش‌های دفاعی پیشین گریزی خواهیم زد. در بخش ۴ به معرفی روش پیشنهادی خواهیم پرداخت و به ارزیابی این روش در بخش ۵ می‌پردازیم و در پایان در بخش ۶ نتیجه‌گیری بر این مبحث خواهیم داشت.

[1] Internet Service Provider [2] Feature Engineering

۲ حملات انگشت‌نگاری وب‌سایت

اولین مطالعه در حوزه حملات انگشت‌نگاری وب‌سایت که به ارزیابی شبکه تُر پرداخت توسط هرمن [۳] در سال ۲۰۰۹ انجام شد [۸]. در این مطالعه، هرمن با استفاده از دسته‌بند بیز چندگانه [۴] بر روی ویژگی طول بسته‌ها، تنها با دقت ۳٪ توانست وب‌سایت‌های بازدید شده کاربر را شناسایی کند. باتوجه به نتیجه بدست آمده، بنظر شبکه‌های گمنام‌کننده در مقابل این حملات مصون بودند هرچند تحقیقات بعدی این ادعا را رد کردند. مشکل اصلی روش آنها تکیه بر طول بسته‌ها به عنوان ویژگی در دسته‌بندی بود در حالی که تُر تمامی بسته‌ها را در اندازه ثابت ۵۱۲ بایت به نام سلول ارسال می‌نماید و استفاده از این ویژگی را بدون استفاده می‌کند.

در سال ۲۰۱۱ پانچنکو و همکاران از دسته‌بند ماشین بردار پشتیبان [۵] و ویژگی بسته‌های قطاری [۶] (بسته‌های پشت سرهم دریافتی یا ارسالی) استفاده کردند؛ که نتیجه آن دستیابی به دقتی بیش از ۵۰٪ بود [۵]. در ادامه این تحقیقات وانگ و همکاران در حمله‌ای با استفاده از دسته‌بند کا-نزدیک‌ترین همسایه [۷] و در نظر گرفتن بیش از ۳۰۰۰ ویژگی ترافیک، توانستند دقت خود را تا ۹۱٪ افزایش دهند [۴]. تعداد زیاد این ویژگی‌ها بر اساس گسترش ویژگی‌های پایه بدست آمده است. بطور مثال تعداد بسته‌های ارسالی کاربر در بازه X بسته یا مجموع طول بسته‌های هم جهت در بازه Y بسته از جمله این ویژگی‌ها بودند. هرچند تحقیقات بعدی نشان دادند در عمل با تعداد کمتری می‌توان به دقت بالایی رسید [۲].

۱.۲ حملات انگشت‌نگاری وب‌سایت مبتنی بر شبکه عصبی عمیق

اولین تحقیقات با هدف بکاربردن شبکه عصبی عمیق در حوزه انگشت‌نگاری وب‌سایت توسط ایب و گوتو [۸] انجام شد [۹]. آنها نشان دادند که با استفاده از خودکدگذار نویزبر پشته‌ای [۹] می‌توان به دقت ۸۶٪ رسید. یکی از دلایل دقت پایین آنها، کمبود تعداد نمونه‌های وب‌سایت‌ها بود. در حالی که شبکه عصبی عمیق برای بدست آوردن نتایجی مطلوب، نیاز به نمونه‌های زیادی دارد. ریمر [۱۰] و همکاران نشان دادند که شبکه عصبی عمیق به عنوان ابزاری به منظور خودکارسازی مهندسی ویژگی‌ها قابل استفاده است. از این رو با خودکارسازی مهندسی ویژگی، نیاز به طراحی و استخراج ویژگی‌ها به صورت دستی را حذف کردند. آنها با استفاده از شبکه خودکدگذار به دقتی تا ۹۵/۳٪ دست یافتند [۲].

سرینم [۱۱] و همکاران با بهره بردن از شبکه عصبی پیچشی [۱۲] و طراحی یک مدل چندلایه عمیق، دقت روش‌های پیشین را بر روی شبکه تُر بدون دفاع تا ۹۸٪ افزایش دادند [۳]، همچنین به بالاترین دقت بر روی روش‌های مقابله [۷، ۱۰] دست یافتند.

[3] Herrmann [4] Multinomial naïve-bayes classifier [5] Support Vector Machine [6] Burst Packet [7] K-Nearest Neighbor [8] Abe and Goto [9] Stacked Denoising AutoEncoder [10] Rimmer [11] Sirinam [12] Convolutional Neural Network

۳ مقابله با حملات انگشت‌نگاری وب‌سایت

به منظور مقابله با حملات انگشت‌نگاری وب‌سایت، راهکارهای متعددی ارائه شده است. به طور کلی این روش‌ها با به تاخیر انداختن ارسال بسته‌ها، ایجاد بسته‌های ساختگی[۱]، یا تغییر در ترتیب ارسال بسته‌ها، سعی در غیرقابل تمایز ساختن ویژگی‌های ترافیک ردوبدل شده و در نتیجه افزایش نرخ خطای مهاجم در دسته‌بندی وب‌سایت‌ها را دارند. مشخص است که اعمال این روش‌ها با دو سربار زمانی و پهنای باند همراه خواهد بود. از این رو روش پیشنهادی باید به نحوی باشد تا مصالحه‌ای میان کاهش دقت مهاجم و سربار ایجاد شده همراه باشد. به طور کلی روش‌های مقابله به سه دسته، مبهم‌سازی در لایه کاربرد، لایه‌گذاری، و دگرریختی[۲] تقسیم می‌شوند.

۱.۳ مبهم‌سازی در لایه کاربرد

این روش‌ها سعی کرده‌اند با دستکاری در روال عادی پروتکل HTTP اقدام به مبهم‌سازی ترافیک عبوری نمایند. از جمله این روش‌ها HTTPOS است، که در سال ۲۰۱۱ توسط لو و همکاران ارائه شد [۱۱]. این روش با به تاخیر انداختن ارسال بسته‌ها، تغییر اندازه اشیا وب، و یا ارسال درخواست‌های HTTP به صورت لوله‌کشی[۳]، سعی در کنترل اندازه بسته‌های ورودی و خروجی و زمان انتقال آنها دارد. اگرچه لو و همکاران نشان دادند این روش در مقابل حملات پیشین موثر واقع شده است؛ اما محققان مشخص کردند که در مقابل حملات جدید بدون تاثیر است [۴].

توسعه‌دهندگان تُر به منظور مقابله با حمله معرفی شده توسط پانچنکو [۵] روشی ارائه کردند. در این روش با تغییر خط لوله HTTP، تعداد درخواست‌های ارسالی در هر لوله به صورت تصادفی انجام می‌شد. به این شکل، ترتیب ارسال درخواست‌ها با افزایش تعداد درخواست‌ها و پیشی گرفتن از اندازه لوله تغییر می‌کرد [۶]. اگرچه تُر بروزرسانی برای این روش ارائه کرده‌است اما در مقابل حملات جدید تاثیر چندانی ندارد [۳].

۲.۳ لایه‌گذاری

روش‌های دفاعی در این دسته با لایه‌گذاری بسته‌های واقعی، ارسال متعدد بسته‌های ساختگی، یا ایجاد تاخیر در ارسال بسته‌ها سعی دارند ویژگی‌های آماری ترافیک را تخریب کنند. دایر[۴] و همکاران روش بوفالو[۵] [۱۲] را معرفی کردند. این روش سعی در بهبود روش‌های پیشین که تنها به لایه‌گذاری هر بسته اکتفا می‌کردند داشت. بر اساس مشاهدات دایر، بسته‌های قطاری که در روش‌های قبلی مخفی نمی‌شدند ویژگی‌های مهم و قابل تمایز ترافیک هستند. در روش پیشنهادی بسته‌ها با نرخ ثابت در شبکه ردوبدل شده و طول تمامی بسته‌های ارسالی تا مقدار مشخصی (به طور پیش فرض MTU) لایه‌گذاری می‌شوند. هرچند به دلیل انتقال بسته‌ها با نرخ ثابت این روش از سربار پهنای باند و زمانی بالایی برخوردار است.

پانچنکو به منظور مقابله با حمله پیشنهادی خود، روش دکو[۶] [۵] را پیشنهاد داد. در این روش به ازای هر وب‌سایت هدف، وب‌سایتی دیگر بارگذاری می‌شود تا مهاجم قادر به تفکیک دو وب سایت از هم نباشد.

توسعه‌دهندگان تُر با ارائه روش WTF-PAD و بهره بردن از لایه‌گذاری وفقی[۷]، سعی در کاهش سربار پهنای باند ایجاد شده در شبکه را داشتند. در این روش در صورت کاهش نرخ انتقال بسته‌ها در شبکه، بسته‌های ساختگی در شبکه ارسال می‌شوند تا شکاف بوجود آمده میان بسته‌های قطاری پوشیده شود. به این دلیل، در قیاس با روش‌های پیشین لایه‌گذاری با نرخ ثابت، سربار پهنای باند کاهش چشمگیری خواهد داشت [۷].

۳.۳ دگرریختی

اساس کار روش‌های دفاعی در این دسته، تغییر شکل دادن ویژگی‌های آماری یک دنباله از وب‌سایت به وب‌سایتی دیگر است. در این روش‌ها یک دادگان از ویژگی‌های وب‌سایت‌های مختلف جمع‌آوری می‌شود. پس از آن، زمانی که کاربر در خواست بازدید وب‌سایتی را می‌دهد؛ باتوجه به دادگان موجود تغییراتی در دنباله بسته‌های رد و بدل شده ایجاد می‌شود. تغییراتی مانند اضافه کردن بسته ساختگی، یا تکه‌تکه کردن بسته‌های اصلی اعمال می‌شود تا ویژگی‌های دنباله ارسالی وب‌سایت هدف مشابه وب‌سایتی دیگر گردد.

روش واکی‌تاکی[۸] از جمله این روش‌ها است. در این روش، ارتباط به صورت یک‌طرفه انجام می‌شود و در هر زمان کاربر یا درخواستی ارسال می‌کند و یا جواب درخواست‌های خود را دریافت می‌کند. پس از یک‌طرفه شدن ارتباط، مجموع دنباله بسته‌های وب‌سایت هدف کاربر با یک وب‌سایت دیگر، به یک ابردنباله تبدیل می‌شود. از این رو مهاجم قادر به تفکیک وب‌سایت بازدید شده کاربر از این ابردنباله نیست و حداکثر دقت ٪۵۰ خواهد داشت [۱۰].

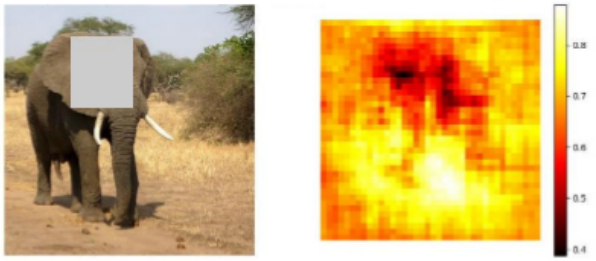
۴ روش پیشنهادی

روش پیشنهادی این مقاله رویکردی مشابه روش دکو [۵] دارد. اما برخلاف روش دکو که هر وب‌سایت هدف، با وب‌سایتی دیگر همزمان بارگذاری می‌شود. سعی داریم تا در عوض ارسال کل دنباله بسته‌ها، الگوی دنباله بسته وب‌سایت‌ها را تشخیص داده و بسته‌های ساختگی را مطابق با این الگو در شبکه انتقال دهیم که در نتیجه آن، سربار پهنای باند آن از روش دکو کمتر خواهد بود (سربار پهنای باند روش دکو حداقل ٪۱۰۰ است).

برای رسیدن به این منظور ابتدا لازم است آن قسمت از دنباله بسته‌های وب‌سایت که برای شبکه عصبی از ارزش اطلاعاتی بیشتری برخوردارند و شبکه عصبی برای دسته‌بندی دنباله بسته به آن تکیه می‌کند را تشخیص دهیم. پس از آن، الگوی هر دنباله بسته وب‌سایت اصلی را تخریب می‌کنیم و در نهایت با تزریق الگوی جدید که از دنباله بسته‌های یک

[1]Dummy packet [2]Morphing [3]Pipelining [4]Dyer [5]Buffered Fixed Length Obfuscation

[6]Decoy [7]Adaptive Padding [8]Walkie-Talkie



شکل ۲. مناطق حساس تصویر برای تصمیم‌گیری شبکه عصبی پیچشی [۱۳]

وب‌سایت دیگر دریافت شده است مدل یادگیری را برای دسته‌بندی به اشتباه می‌اندازیم.

۱.۴ شناسایی الگوی ترافیک

نحوه عملکرد و تصمیم‌گیری شبکه عصبی همچنان یک سوال اساسی است. زایلر[1] و همکاران در آزمایشی به نام انسداد[2] سعی کردند چگونگی شناسایی تصویر توسط شبکه عصبی پیچشی را بررسی کنند [۱۳]. در این آزمایش با حذف نواحی مختلف تصویر، سعی شد تا حساسیت شبکه عصبی به قسمت‌های مختلف تصویر بررسی شود.

به این منظور تصویری به عنوان ورودی شبکه عصبی پیچشی در نظر گرفته شد و با قراردادن یک پوشش خاکستری رنگ بر روی نواحی مختلف تصویر، حساسیت مدل به بخش‌های مختلف تصویر بررسی شد. به طور نمونه همانگونه که در نقشه حرارتی شکل ۲ مشخص است؛ مناطق با رنگ تیره‌تر (صورت فیل) برای مدل از اهمیت بالاتری به منظور دسته‌بندی برخوردار هستند و با تخریب این نواحی مدل در دسته‌بندی تصویر با خطا مواجه می‌شود و قسمت‌هایی که با رنگ روشن‌تر (پای فیل) مشخص شده‌اند تاثیر کمتری در دسته‌بندی تصویر داشته‌اند و حتی با حذف آنها و جایگزین کردن با پوشش خاکستری، شبکه عصبی پیچشی به درستی تصویر را در دسته فیل دسته‌بندی کرده است.

روش پیشنهادی ما شامل شناسایی بخش‌هایی از ترافیک است که شبکه عصبی عمیق مهاجم، برای دسته‌بندی وب‌سایت‌ها به آنها حساس است. در این قسمت روشی مشابه آزمون انسداد معرفی می‌کنیم که با تغییراتی جزئی بر روی دنباله ترافیک کاربر، باعث دسته‌بندی اشتباه توسط مهاجم شود. مشابه آزمون انسداد، پوششی بر روی دنباله بسته‌های هر وب‌سایت گذاشته می‌شود به صورتی که موجب خطا در دسته‌بندی شبکه عصبی شود.

۲.۴ شناسایی الگوی ترافیک با آزمایش انسداد

در انگشت‌نگاری وب‌سایت، دنباله بسته‌های یک وب‌سایت به صورت یک دوتایی $\{(t_k, \pm s_k)\}$ نمایش داده می‌شوند که در آن t_k نمایانگر زمان ارسال یا دریافت بسته k–ام توسط کاربر و s_k نمایانگر اندازه همان بسته در لایه IP است. همچنین علامت بسته k–ام در صورت مثبت بودن، نشانگر بسته ارسالی کاربر و منفی نمایانگر، بسته دریافتی کاربر است. سرینم و همکاران [۳]، با چشم‌پوشی کردن از اندازه و زمان ردوبدل شدن بسته‌ها و تنها با استفاده از جهت و تعداد سلول‌ها (بسته‌های شبکه تُر)، دنباله بسته‌ها را به مقدارهای $\{+۱, -۱\}$ ساده کردند و نشان دادند استفاده از این ویژگی‌ها برای متمایز کردن دنباله وب‌سایت‌ها کافی است.

فرض می‌کنیم مجموعه‌ای از وب‌سایت‌های مختلف به نام S داریم و مهاجم با در دسترس داشتن دسته‌بند $f(x)$ و داشتن نمونه i–ام از این دادگان سعی دارد دسته این نمونه، که سایت $s \in S$ است را تشخیص دهد. دنباله بسته Seq_s^i را به عنوان نمونه i، در نظر می‌گیریم که به عنوان سایت s دسته‌بندی می‌شود.

$$\mathrm{Seq}_s^i = \langle p_۰^i, p_۱^i, \dots, p_m^i \rangle, \quad p_k^i \in \{+۱, -۱\}.$$

همچنین دنباله بسته قطاری Burst_s^i را به عنوان دنباله بسته‌های قطاری ترافیک نمونه i، در نظر می‌گیریم که در آن بسته‌های هم جهت و پشت سرهم در دنباله بسته Seq_s^i به صورت تجمعی در نظر گرفته می‌شوند:

$$\mathrm{Burst}_s^i = \langle b_۰^i, \dots, b_n^i \rangle, \quad b_k^i \in [-m, +m],\ b_q^i = \sum_{k=r} p_k^i.$$

استفاده از دنباله بسته قطاری موجب می‌شود تا آزمون انسداد در ترافیک با سرعت بسیار بیشتری نتیجه دهد. هدف ما پیدا کردن بسته قطاری در دنباله Burst_s^i است؛ به صورتی که با پوشش آن، دسته‌بند $f(x)$ در دسته‌بندی نمونه i–ام به خطا افتد و $f(\mathrm{Burst}_s^i) \neq ۰$.

در آزمایش انسداد بر روی تصویر، پوشش مورد استفاده یک برچسب خاکستری بود. می‌توان در این آزمایش نیز از پوشش $\mathrm{cover}(b_k^i) = \langle ۰ \rangle$ استفاده کرد. اما حذف بسته‌های اصلی وب‌سایت، موجب از دست رفتن اطلاعات و عدم توانایی کاربران برای اتصال به وب‌سایت مورد نظرشان می‌شود. به این منظور بجای حذف بسته، هر بسته قطاری ارسالی از سمت کاربر را با استفاده از بسته‌های دریافتی کاربر به تکه بسته‌ها تغییر می‌دهیم و هر بسته دریافتی کاربر را با استفاده از بسته‌های ارسالی از سمت کاربر به صورت بسته‌های تکی جدا می‌کنیم به صورتی که برای بسته‌های قطاری ارسالی کاربر (با علامت مثبت) داریم (بسته‌های دریافتی نیز به‌صورتی مشابه قابل پوشش هستند):

$$\mathrm{cover}(b_k^i) = \begin{cases} \langle +۱, -۱ \rangle \cdot \mathrm{cover}(b_k^i - ۱) & \text{if } b_k^i > ۱ \\ \langle +۱, -۱ \rangle & \text{if } b_k^i = ۱ \end{cases}$$

با یک مثال چگونگی انجام آزمایش انسداد با پوششی به اندازه یک بسته قطاری بر روی دنباله ترافیک را شرح می‌دهیم. دنباله‌ای از سلول‌ها (بسته‌های شبکه تُر) را در نظر بگیرید که به این صورت هستند:

$$\mathrm{Seq}_{\mathrm{google}}^{۱} = \langle ۱, -۱, ۱, -۱, -۱, -۱, ۱, \dots, -۱ \rangle$$

همچنین دنباله بسته قطاری این نمونه به صورت زیر تشکیل می‌شود:

$$\mathrm{Burst}_{\mathrm{google}}^{۱} = \langle ۱, -۱, ۱, -۳, ۲, \dots, -۱ \rangle$$

[1] Zeiler [2] Occlusion Experiment

الگوریتم ۱ شناسایی الگوی ترافیک با آزمایش انسداد

Input: consist of HB set with m' sites and one instance of each such that: HB $= \{\text{HBurst}'\}$ where HBurst$' = \langle b_0^i, \ldots, b_y^i \rangle$

Output: is sensitive bursts of HB with m instances

For each instances in HB by padding (by 0) or truncating the instance, generate instances with fix length L
Construct Patterns set from browsing history of client as follows:
for $i = 1 : m'$ **do**
  for $j = 1 : L$ **do**
    cover = concatenate $[-1, 1]$, $|b_j^i|$ times
    $\text{HBurst}'^i = \text{HBurst}^i$ such that b_j^i replaced by cover
    if If $f(\text{HBurst}'^i_s) \neq f(\text{HBurst}^i_s)$ **then**
      insert to Patterns set
    end if
  end for
end for
Return(Patterns)

بسته قطاری چهارم با سه بسته ارسالی پوشیده شده و دنباله جدید به صورت:

$$\text{Burst}'^{۱}_{\text{google}} = \langle ۱, -۱, ۲, -۱, ۱, -۱, ۱, -۱, ۲, \ldots, -۱ \rangle$$

در صورتی که $f(\text{Burst}'^{۱}_{\text{google}}) \neq f([\text{Burst}^{۱}_{\text{google}})$ باشد؛ بسته قطاری چهارم به عنوان قسمتی از ترافیک که شبکه عصبی به آن حساس است در نظر گرفته می‌شود و در زمان اتصال کاربر به وب‌سایت گوگل، دنباله بسته را به صورتی تغییر می‌دهیم تا مطابق با دنباله $\text{Burst}'^{۱}_{\text{google}}$ شود.

پیاده‌سازی این روش در عمل با پیچیدگی‌هایی همراه خواهد بود. اولاً باتوجه به آنکه برای تبدیل هر دنباله بسته قطاری به دنباله‌ای از بسته‌های ارسالی و دریافتی، نیاز به تاخیر در ارسال بسته‌های وب‌سایت به اندازه $\text{RTT} \times \text{length}(b_k^i)$ است؛ سربار زمانی بالایی خواهد داشت. ثانیاً در صورت آشنایی مهاجم با روش مقابله استفاده شده و آموزش شبکه عصبی بر روی داده‌های جدید به دقت بالایی دست خواهد یافت. از این رو لازم است تا آزمایش انسداد ترافیک با رویکرد دیگری انجام شود.

۳.۴ ساخت و تزریق الگوی جدید ترافیک توسط فانوس

در بخش ۲.۴ با استفاده از آزمایش انسداد قسمت‌هایی از ترافیک که سهم عمده در دسته‌بندی ترافیک، توسط شبکه عصبی پیچشی داشتند مشخص شدند. در این بخش با پوششی متفاوت آزمایش انسداد را تکرار می‌کنیم. فرض ما از این قرار است که تاریخچه مرور وب کاربران، از

الگوریتم ۲ تزریق الگوی جدید به ترافیک با آزمایش انسداد

Input: consist of two datasets, Patterns and Bursts such that Patterns has m' instances and Bursts has m sites $\in S$ with n instances of each. $m' > m$, Patterns $= \{pattern^i\}$ where $pattern^i = \langle b_0^i, \ldots, b_y^i \rangle$, Bursts $= \{\text{Bursts}_s^i\}$ where $\text{Bursts}_s^i = \langle b_0^i, \ldots, b_z^i \rangle$

Output: is defined websites set B

For each instances in Bursts by padding (by 0) or truncating the instance, generate instances with fix length L
For each $pattern'$ in Patterns, assign $pattern'$ to $k \in S$ such that $pattern_k^i = \text{Assign}(pattern', k = \text{rand}(S))$
Occlude every Burst_k^i with $pattern_k$
for $i = 1 : m \times n$ **do**
  for $j = 1 : L$ **do**
    $\text{Burst}''^i_k = \text{Burst}_k^i$ such that replace b_j^i and $\max(pattern_k, b_j^i)$
    if $f(\text{Burst}''^i_k = \text{Burst}_k^i)$ **then**
      insert Burst''^i_k to B
    end if
  end for
end for
Return(B)

دو نظر پوششی مناسب است. اول آنکه این تاریخچه برای هر کاربر متفاوت است و دوم باتوجه به آنکه مهاجم با نظارت بر ترافیک کاربر، وب‌سایت‌های مرور شده کاربر را مشاهده کرده است؛ مشابه روش دکو این تاریخچه روشی مناسب برای فریب مهاجم است.

روش پیشنهادی با دو الگوریتم شرح داده می‌شود. در الگوریتم ۱ مطابق با بخش ۲.۴ به ازای هر دنباله بسته قطاری از تاریخچه وب‌سایت‌های بازدید شده کاربر، چکیده‌ای ذخیره می‌کنیم. این چکیده نمایانگر قسمت‌هایی از ترافیک است که شبکه عصبی برای دسته‌بندی به آن حساس است و در الگوریتم ۲ این چکیده‌ها را به ترافیک وب‌سایت کاربر تزریق می‌کنیم تا دسته‌بند با خطا مواجه شود. این دو الگوریتم را با استفاده از پوشش، یک بسته قطاری شرح می‌دهیم و می‌توان با تغییرات جزئی الگوریتم، آن را برای پوشش‌هایی با بیش از یک بسته قطاری توسعه داد.

فرض می‌کنیم دنباله بسته‌های قطاری وب‌سایت‌های بازدید شده کاربر، توسط افزونه‌ای در مرورگر کاربر ذخیره شده‌اند. مطابق با الگوریتم ۱، در گام نخست تمامی نمونه‌های این دادگان تا مقداری مشخص با ۰ لایه‌گذاری می‌شوند. سپس مطابق با آزمایش انسداد در بخش ۲.۴ قسمت‌های حساس هر دنباله استخراج شده و به دادگان Patterns اضافه می‌گردند.

در الگوریتم ۲ با استفاده از چکیده تاریخچه وب‌سایت‌های مروری کاربر (دادگان استخراجی از الگوریتم ۱)، دنباله بسته‌های وب‌سایت‌های هدف را می‌پوشانیم. به این منظور دو دادگان در اختیار داریم. دادگان Bursts شامل دنباله بسته‌های قطاری وب‌سایت‌های هدف و دادگان Patterns که چکیده یا الگوی تاریخچه وب‌سایت‌های مروری کاربر است. همچنین فرض می‌کنیم تعداد وب‌سایت‌های تاریخچه مرور کاربر بیش از وب‌سایت‌های هدف کاربر است. باتوجه به آنکه هر وب‌سایتی که کاربر آن را بازدید می‌کند در تاریخچه مرور کاربر ثبت می‌شود فرضیه‌ای دور از ذهن نیست.

در گام نخست دنباله بسته‌ها را تا مقداری مشخص لایه‌گذاری می‌کنیم، سپس در گام دوم هر نمونه از دادگان Patterns را به صورت تصادفی به یکی از دسته وب‌سایت‌های هدف تخصیص می‌دهیم تا در زمان بارگذاری وب‌سایت هدف، دنباله بسته‌ای از Patterns به آن اضافه گردد. در گام سوم هر دنباله بسته از وب‌سایت هدف را با دنباله‌ای از دادگان Patterns که به آن اختصاص داده شده است می‌پوشانیم و در صورتی که دسته‌بندی این وب‌سایت تغییر کرد به دادگان دنباله بسته‌های مقابله شده اضافه می‌کنیم و در زمان ارسال بسته‌های کاربر مطابق با الگوی جدید بسته‌ها را ارسال می‌نماییم. همچنین برای جلوگیری از دست رفتن بسته‌ها میان دنباله بسته و پوشش با استفاده از تابع max بیشینه بسته را در نظر گرفته‌ایم.

۵ ارزیابی

در این بخش ابتدا دادگان استفاده شده به منظور ارزیابی روش پیشنهادی معرفی شده، سپس معیارهای ارزیابی بررسی می‌شوند. در ادامه معماری برای پیاده‌سازی روش پیشنهادی در شبکه تُر معرفی می‌شود و در آخر نتایج روش پیشنهادی با بهترین روش‌های مقابله مطرح شده، مقایسه می‌شوند.

۱.۵ دادگان و معیار ارزیابی

به منظور ارزیابی روش پیشنهادی از دادگان سرینم استفاده کرده‌ایم. این دادگان شامل ۹۵ وب‌سایت است که هر وب‌سایت در دوره زمانی‌های غیر متوالی، ۱۰۰۰ بار بازدید شده است که در نتیجه آن ۹۵۰۰۰ نمونه دنباله بسته از مقدارهای $\{+1, -1\}$ در دسترس است. طول هر دنباله پس از لایه‌گذاری ۵۰۰۰ سلول است. همچنین ٪۸۰ دادگان به منظور آموزش، ٪۱۰ اعتبارسنجی، و ٪۱۰ برای آزمون در نظر گرفته شده‌اند. همچنین به منظور پیاده‌سازی روش پیشنهادی از Keras بر بستر Tensorflow استفاده شده است.

به منظور ارزیابی روش مقابله بطور کلی دو معیار، سربار تحمیلی به شبکه (پهنای باند و زمانی) و دقت مهاجم مطرح است. سربار پهنای باند روش مقابله D بر روی دادگان S که شامل دنباله سلول‌های P است، باتوجه به آنکه اندازه سلول‌ها برابر می‌باشد؛ مطابق فرمول (۱) محاسبه می‌شود.

$$O_D^B(S) = \frac{\sum_{P\in S}|D(P)| - \sum_{P\in S}|P|}{\sum_{P\in S}|P|} \qquad (۱)$$

سربار زمانی روش مقابله D بر روی دنباله سلول‌های P مقدار زمان اضافه به‌منظور انتقال دنباله بسته‌ها است که مطابق با فرمول (۲) محاسبه می‌گردد. همچنین مطابق با تحقیقات [۷، ۱۰] فرض می‌کنیم ارسال بسته‌های ساختگی در شبکه موجب ایجاد سربار زمانی در شبکه نمی‌شود (پهنای باند به اندازه‌ای در نظر گرفته می‌شود که ارسال بسته‌های ساختگی موجب به تاخیر افتادن بسته‌های اصلی نمی‌شود).

$$O_D^T(S) = \frac{\sum_{P\in S} T_{|D(P)|} - \sum_{P\in S} T_{|P|}}{\sum_{P\in S} T_{|P|}} \qquad (۲)$$

همچنین دقت مهاجم را در مدل محیط بسته [1] مطابق فرمول (۳) ارزیابی می‌کنیم. در مدل محیط بسته فرض می‌شود که کاربر از همان وب‌سایتی بازدید می‌کند که مهاجم بر روی آن دسته‌بند خود را آموزش داده است. Pcorrect مجموع تعداد پیش‌بینی‌های درست مدل یادگیری مهاجم است و N تعداد کل دنباله‌هاست.

$$\text{Accuracy} = \frac{P_{\text{correct}}}{N} \qquad (۳)$$

۲.۵ معماری روش پیشنهادی

در این بخش به منظور ارتباط کاربر با کارگزار نویز [2] (کارگزاری که برای پوشش ترافیک اقدام به ارسال بسته‌های ساختگی به سمت کاربر می‌کند) معماری پیشنهاد داده و با معماری روش‌های پیشین مقایسه می‌کنیم.

مهاجم در مدل تهدید ارائه شده توسط سرینم ، کنترل گره ورودی تُر را در دست گرفته است. در این صورت مهاجم با توجه به شناسه مدار [3] (به‌ازای هر وب‌سایت متمایز، مداری مجزا تشکیل می‌شود) امکان تفکیک ترافیک وب‌سایت‌هایی را که همزمان بازدید می‌شوند دارد. این مدل تهدید از دو جهت روش‌های دفاعی پیشین را مورد نقد قرار می‌دهد. روش‌هایی مانند [۷] که فرض کرده‌اند کارگزار نویز در گره ورودی تعبیه شده است. در این صورت مهاجم قادر به تفکیک ترافیک اصلی از ترافیک جعلی که خود اقدام به ساخت آن می‌کند خواهد بود و روش‌هایی مانند [۱۲] که فرض کرده‌اند کارگزار نویز خارج از شبکه است. در این صورت مهاجمی که بر روی گره ورودی قرار دارد با توجه به شناسه مدار، امکان تفکیک ترافیک ساختگی و اصلی را از هم دارد.

در روش پیشنهادی، کارگزار نویز در گره خروجی قرار گرفته است که از دو نظر نسبت به روش‌های قبلی ارجح است. اولاً حتی مهاجمی که کنترل گره ورودی را در دست گرفته است، امکان تفکیک بسته‌های اصلی و ساختگی ترافیک را از هم ندارد و ثانیاً نیاز به ذخیره دنباله بسته‌های وب‌سایت‌ها در سمت کاربر نیست.

[1] Close world [2] Noise Server [3] Circuit ID

جدول ۱. نتایج آزمایش انسداد با یک بسته قطاری

توضیحات	مقدار
دنباله بسته‌های قطاری که اشتباه دسته‌بندی شدند	۵۴٪
بسته‌های قطاری حساس که دریافتی کاربر بوده‌اند	۹۱٪
پوشش‌هایی که در ۳۰٪ ابتدایی دنباله موثر بودند	۴۵٪
پوشش‌هایی که در ۵۰٪ ابتدایی دنباله موثر بودند	۸۸٪
میانگین اندازه پوشش به اندازه کل دنباله بسته	۰٫۳٪
سربار پهنای باند	۱٫۶٪

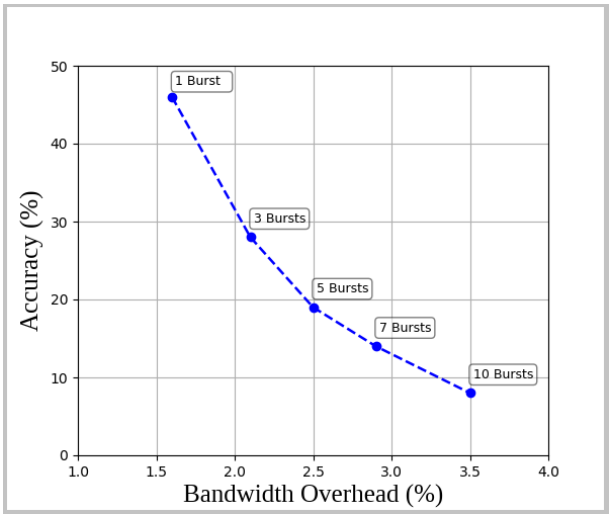
در فانوس نیازی به ذخیره دنباله بسته‌های وب‌سایت‌ها در سمت کاربر نیست. در این روش افزونه‌ای در سمت کاربر نصب شده و تنها دادگانی از چکیده دنباله تاریخچه وب‌سایت‌ها ذخیره می‌شود. زمانی‌که کاربر درخواست بازدید وب‌سایت هدف را می‌دهد همراه با ارسال درخواست GET، یک نمونه از دادگان چکیده مرور وب کاربر به سمت کارگزار هدف روانه می‌شود. گره خروجی با دریافت این درخواست، ابتدا چکیده مورد نظر را استخراج کرده و سپس درخواست GET را به سمت سایت مقصد ارسال می‌کند. در این بین با استفاده از الگوریتم ۲ اقدام به پوشش این دنباله می‌کند و دنباله خروجی را هم در سمت خود ذخیره و هم با پاسخ GET برای کاربر ارسال می‌کند.

باتوجه به دانش ما از دادگانی که در اختیار داریم. اندازه چکیده‌ای که از سمت کاربر ارسال می‌گردد به طور میانگین ۱۳ بایت است که باتوجه اندازه ۵۱۲ بایت سلول، مقداری ناچیز است و قابلیت ارسال با همان سلول اول را دارد. همچنین زمان پردازش آزمایش انسداد بر روی یک دنباله با پوشش ۱۰ بسته قطاری و با استفاده از یک پردازنده ۲٫۴ گیگاهرتزی نزدیک به ۹ میلی‌ثانیه است که نشان می‌دهد نیازی به تاخیر انداختن پاسخ کاربر برای پردازش در کارگزار نویز نیست و امکان پاسخ سریع درخواست‌ها وجود دارد.

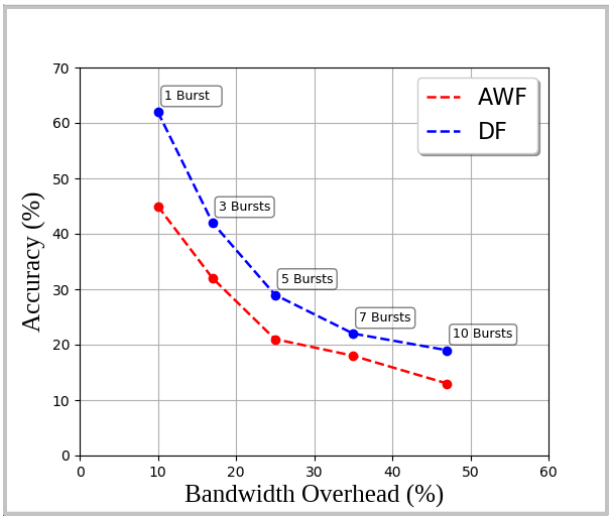
۳.۵ نتایج شناسایی الگوی ترافیک با انسداد

باتوجه به نتایج بدست آمده از روش شناسایی الگوی ترافیک با آزمایش انسداد در جدول ۱، با استفاده از پوششی به اندازه یک بسته قطاری، ۵۴٪ دنباله بسته‌ها به اشتباه دسته‌بندی می‌شوند. همچنین ۹۱٪ بسته‌های قطاری حساس، بسته‌های دریافتی کاربر بوده‌اند. از جمله نتایج مهم دیگر این آزمایش، حساسیت بالای شبکه عصبی پیچشی به ابتدای دنباله بسته‌ها است.

پس از انتخاب نوع پوشش، آزمایش را با استفاده از پوشش‌هایی با اندازه دیگر نیز بررسی کردیم که نتایج استفاده از پوشش‌های ۱، ۳، ۵، ۷، ۱۰ بسته قطاری در شکل ۳ مشخص است. بر اساس نتایج این آزمایش با استفاده از پوشش ۱۰ بسته قطاری و سربار پهنای باند ۳٫۵٪، می‌توان دقت مهاجم در تشخیص وب‌سایت‌های کاربر را به ۸٪ تقلیل داد.



شکل ۳. بررسی میزان موفقیت آزمایش انسداد در پوشش ترافیک



شکل ۴. سربار پهنای باند کاربر به نسبت میزان کاهش دقت مهاجمان [۲، ۳] با روش فانوس

۴.۵ نتایج روش فانوس در مقایسه با سایر روش‌ها

سزجدی[1] و همکاران در آزمایشی نشان دادند، داده‌هایی که تصمیم‌گیری شبکه عصبی مشخصی را با خطا مواجه می‌کنند. امکان فریب شبکه عصبی دیگر با معماری متفاوت را نیز دارند [۱۴]. قابلیت انتقال از آن جهت حائز اهمیت است که بدون دانش در خصوص معماری و پارامترهای شبکه عصبی استفاده شده، می‌توان داده‌هایی ساخت که شبکه‌های عصبی با معماری متفاوتی را فریب دهند. با استفاده از همین قابلیت، نمونه‌های ساخته شده بر روی شبکه عصبی پیچشی سرینم (DF) را بر روی شبکه عصبی خودکدگذار نویزبر پشته‌ای ریمر (AWF) آزمایش کردیم. همانگونه که از نتایج این آزمایش در شکل ۴ مشخص است. با استفاده از پوشش ۱۰ بسته قطاری و سربار پهنای باند ۴۷٪ می‌توان دقت روش سرینم را به ۱۹٪ و دقت روش ریمر را به ۱۳٪ تقلیل داد.

همچنین در جدول ۲ روش پیشنهادی خود را با روش‌هایی که بهترین نتایج را بر روی حملات اخیر داشته‌اند مقایسه کرده‌ایم. همانطور که

[1] Szegedy

جدول ۲. سربار پهنای باند و زمانی بهترین روش‌های مقابله با حملات انگشت‌نگاری در مقایسه با روش فانوس

روش مقابله	سربار زمانی	سربار پهنای باند	سربار دقت DF	دقت AWF
بوفالو [۱۲]	۱۳۷٪	۲۶۴٪	۱۲٫۶٪	۱۱٫۷٪
واکی تاکی [۱۰]	۳۴٪	۳۱٪	۴۹٫۷٪	۴۵٫۸٪
WTF-PAD [۷]	۰٪	۶۴٪	۹۰٫۷٪	۶۰٫۸٪
فانوس	۰٪	۴۷٪	۱۹٪	۱۳٪

مشخص است فانوس تنها با ۴۷٪ سربار پهنای باند، دقت مهاجمان را در مقایسه با سایر روش‌ها به نسبت سربار تحمیلی به شبکه به پایین‌تر مقدار رسانده است. باتوجه به آنکه در این روش نیازی به تاخیر انداختن بسته‌های واقعی نیست، برخلاف آزمایش نخست سربار زمانی به شبکه تحمیل نخواهد شد.

۶ نتیجه

در این مقاله بر اساس آزمایش انسداد، روش فانوس را معرفی کردیم که در مقایسه با روش‌های پیشین با سربار پایین، نرخ خطای مهاجم در دسته‌بندی را به بالاترین مقدار خود می‌رساند. این روش، با پوششی مناسب، قسمت‌هایی از ترافیک را که برای شبکه عصبی پیچشی از ارزش اطلاعاتی بالایی برخوردارند تغییر می‌دهد تا مهاجم در دسته‌بندی ترافیک با خطا روبرو شود. همچنین در ادامه نشان دادیم که بدون نیاز به دانشی در خصوص معماری شبکه عصبی استفاده شده، می‌توان داده‌های تغییریافته را بر روی معماری یک شبکه خودکدگذار نویزبر پشته‌ای آزمود. یکی از چالش‌های روش پیشنهادی نیاز به دادگانی آماده از دنباله بسته‌های وب‌سایت‌هاست. باتوجه به نتایج جدول ۱ و حساسیت دسته‌بند به ابتدای دنباله‌ها، در ادامه این تحقیق، می‌توان به نحوی پوشش‌ها را به صورت تصادفی به ابتدای دنباله‌ها افزود تا بدون نیاز به دادگان، روش پیشنهادی به صورت برخط قابل اجرا باشد.

مراجع

[1] Tor. https://torproject.org. Accessed: 2018.

[2] Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom Van Goethem, and Wouter Joosen. Automated website fingerprinting through deep learning. *arXiv preprint arXiv:1708.06376*, 2017.

[3] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1928–1943, 2018.

[4] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 143–157, 2014.

[5] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, pages 103–114, 2011.

[6] M. perry. experimental defense for website traffic fingerprinting. https://blog.torproject.org/blog/experimental-defense-website-traffic-fingerprinting. 2011.

[7] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *European Symposium on Research in Computer Security*, pages 27–46. Springer, 2016.

[8] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 31–42, 2009.

[9] Kota Abe and Shigeki Goto. Fingerprinting attack on tor anonymity using deep learning. *Proceedings of the Asia-Pacific Advanced Network*, 42:15–20, 2016.

[10] Tao Wang and Ian Goldberg. {Walkie-Talkie}: An efficient defense against passive website fingerprinting attacks. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1375–1390, 2017.

[11] Xiapu Luo, Peng Zhou, Edmond WW Chan, Wenke Lee, Rocky KC Chang, Roberto Perdisci, et al. Httpos: Sealing information leaks with browser-side obfuscation of encrypted flows. In *NDSS*, volume 11, 2011.

[12] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *2012 IEEE symposium on security and privacy*, pages 332–346. IEEE, 2012.

[13] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.