Kỷ yếu Hội nghị Quốc gia lần thứ VIII về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR); Hà Nội, ngày 9-10/7/2015
DOI: 10.15625/vap.2015.000189

NHẬN DẠNG PAYLOAD ĐỘC VỚI HƯỚNG TIẾP CẬN TẬP MÔ HÌNH MÁY HỌC

Nguyễn Hữu Hòa, Đỗ Thanh Nghị, Phạm Nguyên Khang
Khoa Công nghệ Thông tin và Truyền thông, Trường Đại học Cần Thơ
nhhoa@ctu.edu.vn, dtnghi@cit.ctu.edu.vn, pnkhang@cit.ctu.edu.vn

TÓM TẮT *- Sự sinh sôi nẩy nở nhanh chóng của các payload độc (mã độc được ngụy trang trong nội dung gói tin) đang trở thành mối nguy hại trong vấn đề toàn dữ liệu và an ninh mạng. Trong số nhiều giải pháp được đề xuất bởi cộng đồng nghiên cứu nhằm đối phó với mối nguy hại gia tăng này, hướng tiếp cận tập mô hình máy học đã bộc lộ tính ưu việt đối với vấn đề cải thiện độ chính xác nhận dạng. Tuy nhiên, sức mạnh của một tập mô hình phụ thuộc lớn vào tính đa dạng của các mô hình thành viên. Trong ngữ cảnh này, chúng tôi đề xuất một phương pháp mới và hiệu quả để xây dựng tập mô hình máy học cho bài toán nhận dạng payload độc. Trong hướng tiếp cận của chúng tôi, các mô hình thành viên được đa dạng hóa bằng cách thay đổi tham số từ một kỹ thuật biểu diễn dữ liệu được đề xuất. Kết quả thực nghiệm chứng minh rằng phương pháp chúng tôi đề xuất cho kết quả tốt hơn so với những phương pháp thông dụng khác.*

Từ khóa *- Nội dung gói tin (payload), nhận dạng payload (payload detection), tập mô hình máy học (classifier ensemble), tính đa dạng mô hình (classifier diversity).*

I. GIỚI THIỆU

Việc nhận dạng sự hiện diện của mã độc trong nội dung (payload) của gói tin mạng hay Web, gọi tắt là *payload độc*, là một chủ đề thách thức thu hút sự quan tâm của cộng đồng nghiên cứu. Những phương pháp truyền thống sử dụng tập chữ ký của mã độc (worms, viruses, malware) và các kỹ thuật so khớp để phát hiện ra payload độc dựa trên một ngưỡng được định nghĩa trước. Tuy nhiên, tin tặc có thể dễ dàng đệm dữ liệu rác vào payload để biến mã độc thành mã vô hại và do đó đánh lừa các mô hình nhận dạng dựa trên tập chữ ký (signature-based models). *Obfuscation* là một kỹ thuật đệm được sử dụng phổ biến bởi cộng đồng tin tặc [11, 12]. Hình 1 minh họa payload độc được ngụy trang bởi kỹ thuật obfuscation. Một ví dụ khác được thể trong Hình 2, trong đó tác giả của sâu Code Red đệm rất nhiều ký tự "N" và "X" vào HTTP payloads để làm tràn bộ nhớ đệm của máy tính nạn nhân. Do đó, để có thể nhận dạng những mã độc và/hoặc những biến thể của mã độc như thế đòi hỏi một sự phân tích sâu vào bên trong payload, thay vì chỉ dựa vào thông tin từ tiêu đề gói tin.

Về căn bản, cộng đồng nghiên cứu thường sử dụng kỹ thuật phân tích *n*-grams để biểu diễn dữ liệu cho việc xây dựng mô hình máy học (machine learning). Tuy nhiên, tính hiệu quả của các mô hình này phụ thuộc lớn vào thứ bậc của *n*. Việc sử dụng *n*-grams bậc thấp (ví dụ: *n*=1 hoặc *n*=2) có thể làm mất thông tin và do đó giảm độ chính xác nhận dạng. Mặc khác, việc sử dụng *n*-grams bậc cao dẫn đến sự bùng nổ không gian chiều của tập dữ liệu mà vượt quá khả năng tính toán của máy tính. Hơn nữa, vấn đề cao chiều (the curse of dimensionality) thường làm giảm tính hiệu quả của mô hình nhận dạng [8]. Do đó, việc xây dựng một mô hình *n*-grams hiệu quả là rất khó và đòi hỏi những chiến lược hợp lý cho vấn đề biểu diễn và xử lý dữ liệu cũng như chọn giải thuật học/huấn luyện.

Từ bối cảnh trên, chúng tôi đề xuất một kỹ thuật mới lạ, gọi là n_p-grams, cho việc biểu diễn dữ liệu *n*-grams bậc cao. Dựa trên kỹ thuật n_p-grams, chúng tôi đề xuất một phương pháp tạo tập mô hình hiệu quả với số lượng lớn các mô hình thành viên khác nhau. Bên cạnh đó, vấn đề cao chiều cũng được xử lý. Thông qua kiểm chứng thực nghiệm, chúng tôi chứng minh rằng phương pháp đề xuất cho kết quả tốt hơn các phương pháp thông dụng khác.

```
var zngvwmyefzmh = 'nuVApEJa3cnuVApEJa69nuVApEJa66';var yoxhmlpkpmb = 'nuVApEJa72';var uigduejgjdu
=
'nuVApEJa61nuVApEJa6dnuVApEJa65nuVApEJa20nuVApEJa6enuVApEJa61nuVApEJa6dnuVApEJa65nuVApEJa3dnuVApEJa
vagpelortng =
'nuVApEJa6fnuVApEJa66nuVApEJa79nuVApEJa64nuVApEJa6dnuVApEJa79nuVApEJa6bnuVApEJa6anuVApEJa77nuVApEJa
xvscldsxeqf =
'nuVApEJa22nuVApEJa20nuVApEJa77nuVApEJa69nuVApEJa64nuVApEJa74nuVApEJa68nuVApEJa3dnuVApEJa22nuVApEJa
ptzzdgjgaah = 'nuVApEJa20nuVApEJa73nuVApEJa72nuVApEJa63nuVApEJa3dnuVApEJa22';var npmseelskcn =
'nuVApEJa68nuVApEJa74nuVApEJa74nuVApEJa70nuVApEJa3anuVApEJa2fnuVApEJa2f';var nfygxwlncgh =
'picturesuploadesonline.net/Proxyscanner/index.php';var rgkiqriqyog =
'nuVApEJa22nuVApEJa20nuVApEJa6dnuVApEJa61nuVApEJa72nuVApEJa67nuVApEJa69nuVApEJa6enuVApEJa77nuVApEJa
```

Hình 1. Payload độc được ngụy trang bởi kỹ thuật obfuscation

Hình 2. Sâu Code Red trong payload

Phần còn lại của bài báo này được cấu trúc như sau. Mục II mô tả bài toán nhận dạng payload, trong khi Mục III trình bày phương pháp đề xuất. Mục IV cụ thể hóa việc kiểm chứng thực nghiệm. Cuối cùng, chúng tôi kết thúc bài báo bằng cách đưa ra kết luận và hướng phát triển trong Mục V.

II. VẤN ĐỀ NHẬN DẠNG PAYLOAD

Trong số nhiều giải pháp hiệu quả được đề xuất trong các tài liệu khoa học, hướng tiếp cận khai khoáng dữ liệu văn bản (text mining) bộc lộ nhiều điểm mạnh. Vì thế, chúng tôi diễn đạt vấn đề nhận dạng payload (payload detection) dưới dạng bài toán phân loại văn bản (text classification), ở đó mỗi payload được xử lý như là một văn bản. Hình 3 khái quát hóa quy trình xây dựng mô hình nhận dạng, gồm 4 bước chính: thu thập dữ liệu, biểu diễn dữ liệu, xử lý đặc trưng và huấn luyện mô hình. Về phương diện lý thuyết, các bước này được mô tả sơ lược như sau.

A. Thu thập dữ liệu

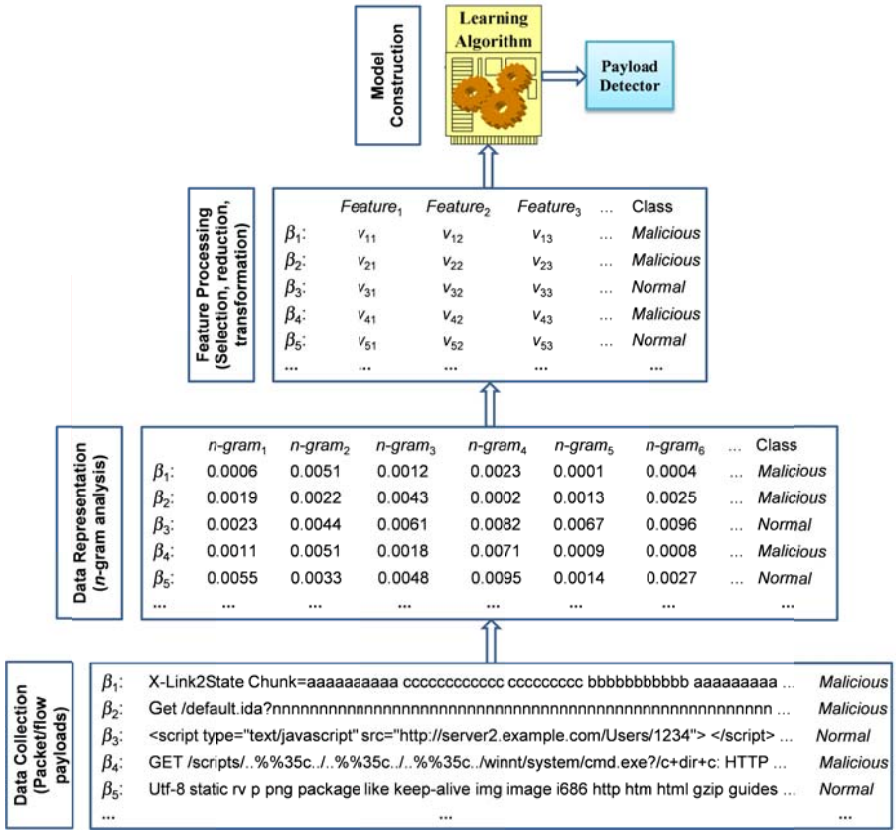
Việc thu thập dữ liệu thường được thực hiện thông qua các công cụ phân tích dữ liệu mạng, như Wireshark, Netflow và Tcpdump. Tập payloads có thể được nhãn hóa thành một hoặc nhiều lớp, sử dụng các phần mềm an ninh mạng (như Anti-Virus, Signature Detection) và/hoặc phương pháp thủ công. Mỗi payload là một chuỗi L bytes (hoặc L kí tự ASCII), trong đó L có thể dao động từ 0 đến vài chục ngàn bytes.

B. Biểu diễn dữ liệu

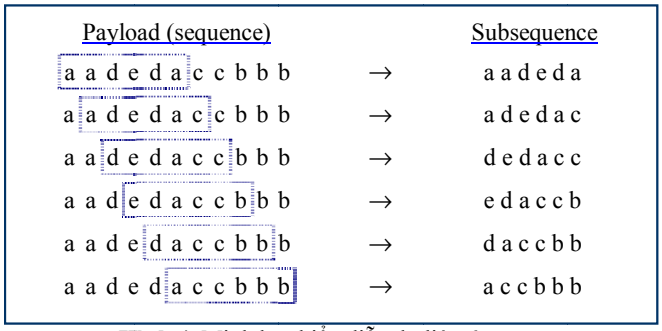
Về căn bản, n-grams là một kỹ thuật được sử dụng rộng rãi để biểu diễn dữ liệu cho bài toán phân loại văn bản. Kỹ thuật này sử dụng một cửa sổ trượt (sliding window) có chiều dài n để trích những chuỗi tuần tự của các bytes trong payloads (Hình 4). Tại mỗi bước trượt (mỗi lần một byte), thông tin thống kê về chuỗi n-grams được tính toán. Theo đó, mỗi chuỗi n-grams được xem như là một đặc trưng (feature) mà giá trị của nó được thống kê bằng các độ đo khác nhau, như *tần số tương đối và tần số xuất hiện*. Trong bài báo này, chúng tôi sử dụng độ đo tần số tương đối: $v_{i,j} = x_{i,j}/L$, trong đó $x_{i,j}$ là số lần xuất hiện chuỗi j trong payload i,và L là chiều dài của payload i. Những nghiên cứu thực nghiệm gần đây cũng chỉ ra rằng, độ đo tần số tương đối thường cho kết quả tốt đối với vấn đề nhận dạng payload [2, 6, 7].

C. Xử lý đặc trưng

Mặc dù có nhiều thuận lợi trong việc biểu diễn dữ liệu (ví dụ như không cần kiến thức chuyên gia), việc phân tích n-grams bậc cao dẫn tới sự bùng nổ không gian chiều mà có thể vượt quá khả năng tính toán của máy tính. Cụ thể hơn, đối với vấn đề nhận dạng payload, số chiều tối đa có thể là 256^n, vì mỗi payload là một chuỗi được biểu diễn từ tập 256 bytes ASCII. Tuy nhiên, trong không gian cao chiều thường tồn tại rất nhiều đặc trưng không phù hợp (irrelevant features). Những đặc trưng như thế cần được loại bỏ trước khi huấn luyện mô hình. Trong bài báo này, chúng tôi sử dụng độ đo *information gain* (độ lợi thông tin) để chọn một số lượng cố định các đặt trưng phù hợp (relevant features) nhằm giảm không gian chiều trong tập dữ liệu huấn luyện. Phương pháp giảm chiều mô tả chi tiết trong Mục III.



Hình 3. Quy trình xây dựng mô hình nhận dạng payload



Hình 4. Minh họa biểu diễn dữ liệu 6-grams

D. Huấn luyện mô hình

Về cơ bản, mô hình nhận dạng payload có thể được xây dựng dưới dạng huấn luyện có giám sát hoặc không giám sát (supervised or unsupervised training), tùy vào khả năng hiện có của dữ liệu, ví dụ như dữ liệu có nhãn lớp hay không. Trong khuôn khổ của bài báo này, chúng tôi xây dựng mô hình theo hướng huấn luyện có giám sát. Cụ thể hơn, chúng tôi sử dụng Linear Proximal Support Vector Machine (máy học véctơ hỗ trợ xấp xỉ tuyến tính) [13] như là một giải thuật cơ sở để xây dựng tập mô hình. Chi tiết về điều này được trình bày trong Mục III.

III. PHƯƠNG PHÁP ĐỀ XUẤT

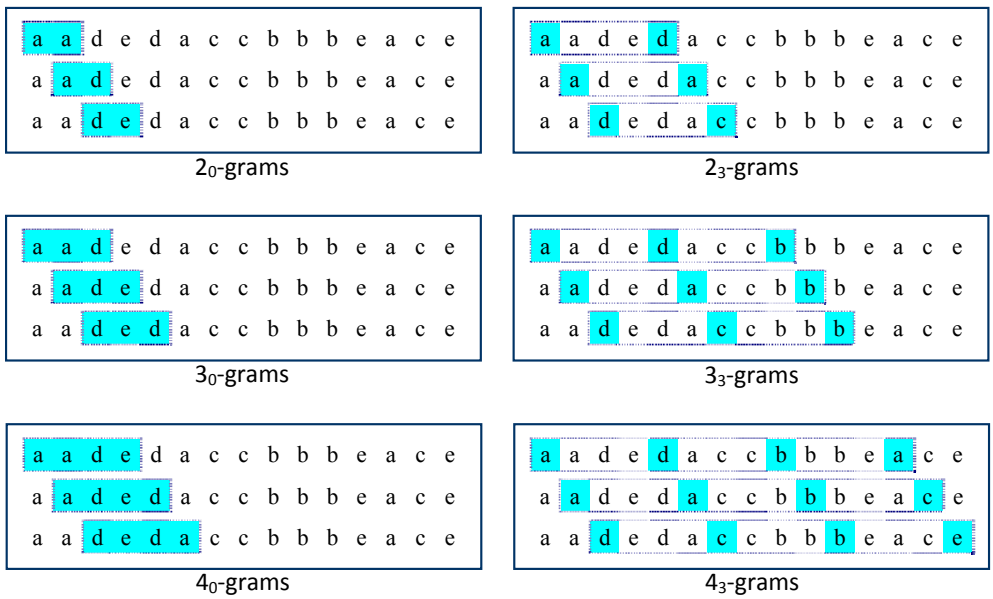
A. Biểu diễn dữ liệu

Kỹ thuật phân tích *n*-grams nhằm để trích phân phối tần số của những chuỗi bytes kề nhau trong payload. Về phương diện thống kê, giá trị của *n* càng cao thì lượng thông tin cấu trúc (structural information) càng lớn [4]. Chẳng hạn như, để lấy được thông tin cấu trúc của chuỗi "http://www", phân tích *n*-grams đòi hỏi phải thiết lập n=10, dẫn đến sự bùng nổ không gian chiều.

Để giải quyết vấn đề trên, chúng tôi đề xuất một biến thể của phân tích *n*-grams, được gọi là n_p-grams. Mục đích là để lấy được một lượng lớn thông tin cấu trúc, trong khi sử dụng giá trị *n* nhỏ nhằm kiềm chế sự bùng nổ không gian chiều. Nguyên lý của kỹ thuật trích đặc trưng n_p-grams được hình thức hóa thông qua Hình 5. Theo đó, thay vì sử dụng cửa sổ trượt chiều dài *n*, chúng tôi sử dụng cửa sổ trượt chiều dài $(n + np - p)$ để trích phân phối tần số của những chuỗi bytes không kề nhau trong payload. Tại mỗi bước trượt (mỗi lần một byte), kỹ thuật n_p-grams lấy *n* bytes mà vị

trí của chúng trong cửa sổ trượt cách khoảng với nhau p bytes. Quá trình trượt được thực hiện cho đến khi lề trái của cửa sổ chạm byte cuối cùng của payload.

Một cách trực quan, n_p-grams và n-grams có độ phức tạp tính toán tương đương nhau, đó là tuyến tính với chiều dài của payload. Cần chú ý rằng, khi $p = 0$, n_p-grams vận hành giống như n-grams. Khi $p > 0$, tần số của chuỗi n_p-grams có thể được xem như là xác suất "lề - trung tâm" được tính từ phân phối tần số của $(n + np - p)$-grams tương ứng. Khi kỹ thuật n_p-grams được tham số hóa với những giá trị khác nhau của n và p, phân phối tần số của n_p-grams chứa đựng nhiều thông tin cấu trúc khác nhau về dữ liệu payload. Việc tổng hợp thông tin cấu trúc của n_p-grams, phần nào, cho phép xây dựng lại thông tin cấu trúc của $(n + np - p)$-grams. Điều này thúc đẩy chúng tôi theo hướng tiếp cận tập mô hình, trong đó mỗi mô hình thành viên vận hành trên một tập đặc trưng n_p-grams khác nhau với sự thay đổi của cả hai giá trị n và p.



Hình 5. Minh họa kỹ thuật trích đặc trưng n_p-grams

B. Giảm chiều dữ liệu

Như đã được đề cập trong các Mục II, tập dữ liệu huấn luyện được trích từ kỹ thuật n-grams và n_p-grams có số chiều rất lớn, do đó cần thiết phải áp dụng một phương pháp giảm chiều trên tập dữ liệu ban đầu. Có nhiều phương pháp giảm chiều được đề xuất trong các tài liệu khoa học, sử dụng các độ đo khác nhau, như *correlation*, *information gain*, *consistency*, *chi-square* và *belief*. Nhằm tránh làm loãng vấn đề quan tâm, trong bài báo này chúng tôi chỉ chọn một độ đo cho mục đích giảm chiều, đó là *information gain* (*IG*) [1]. *IG* là một độ đo phổ biến, đơn giản và có độ phức tạp tính toán tuyến tính với số lượng đặc trưng.

Việc giảm chiều được thực hiện theo hướng xếp hạng các đặc trưng, cụ thể như sau. Đầu tiên, tính giá trị *IG* cho mỗi đặc trưng và rồi xếp hạng các đặc trưng theo giá trị *IG*. Đặc trưng có *IG* càng cao thì tầm quan trọng của nó càng lớn. Cuối cùng, chúng tôi chọn k đặc trưng dựa vào sự xếp hạng, với k là tham số được thiết lập trong thực nghiệm.

Giá trị *IG* của đặc trưng F_j, dựa trên biến lớp Y, được tính bằng các Công thức (1), (2) và (3). Trong đó, $H(Y)$ và $H(Y|F_j)$, tương ứng, là entropy của Y trước và sau khi quan sát F_j.

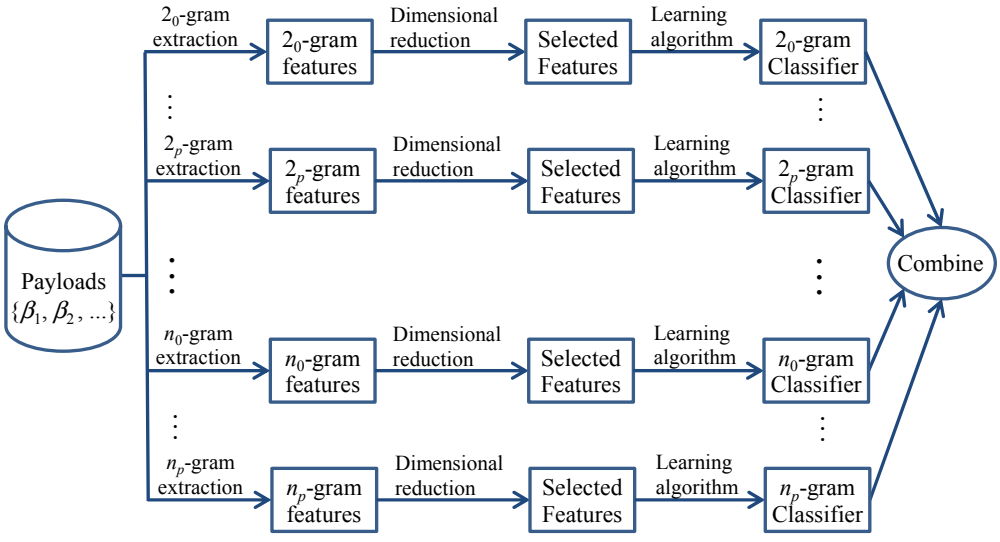
$$IG(F_j) = H(Y) - H(Y|F_j) \quad (1)$$

$$H(Y) = -\sum_{y \in Y} P(y) log_2 P(y) \quad (2)$$

$$H(Y|F_j) = -\sum_{x \in F_j} P(x) \sum_{y \in Y} P(y|x) log_2 P(y|x) \quad (3)$$

C. Tạo tập mô hình

Phương pháp tạo tập mô hình được ý tưởng hóa thông qua Hình 6. Theo đó, bằng cách thay đổi giá trị của hai tham số n và p trong kỹ thuật n_p-grams, chúng tôi thu được các tập đặc trưng khác nhau và do đó tạo ra các mô hình thành viên khác nhau. Hay nói cách khác, mỗi mô hình thành viên (classifier) được xây dựng theo cách thức song song, sử dụng sự biểu diễn dữ liệu khác nhau của payload. Như được thấy từ Hình 6, tổng số mô hình thành viên được tạo ra là $|n| \times |p|$. Trong đó, n được thiết lập với các giá trị nhỏ, nhằm tránh bùng nổ không gian chiều. Tuy nhiên, việc thiết lập giá trị cho p là không hạn chế, vì p không ảnh hưởng đến không gian chiều. Sau khi trích tập đặc trưng n_p-grams, chúng tôi áp dụng kỹ thuật giảm chiều như đã được mô tả trong Mục III.B.



Hình 6. Phương pháp tạo tập mô hình

Đối với việc xây dựng mô hình thành viên, chúng tôi chỉ sử dụng duy nhất một giải thuật học, đó là Linear Proximal Support Vector Machine (LP-SVM) [13]. Giải thuật này phù hợp với phương pháp đề xuất vì hai lý do chính yếu. Thứ nhất, LP-SVM hoạt động tốt trong không gian cao chiều. Thứ hai, LP-SVM có độ phức tạp tuyến tính với số lượng điểm dữ liệu huấn luyện, do đó thời gian xây dựng mô hình và phân loại là nhanh, thích hợp cho hướng tiếp cận tập mô hình [174]. Ở giai đoạn vận hành, kết quả của các mô hình thành viên được tổng hợp theo luật số đông.

IV. KIỂM CHỨNG THỰC NGHIỆM

A. Dữ liệu thực nghiệm

Chúng tôi kiểm chứng phương pháp đề xuất trên các tập dữ liệu được chia sẻ từ cộng nghiên cứu và từ sự thu thập riêng của chúng tôi. Mặc dù phương pháp mà chúng tôi đề xuất có thể áp dụng trên dữ liệu ở các tầng giao thức khác nhau (miễn là dữ liệu kiểu Text), chúng tôi giới hạn thực nghiệm trên tập dữ liệu giao thức HTTP, vì hai lý do chính. Thứ nhất, việc thu thập số lượng đủ lớn các payload độc trong những giao thức khác (như: SMTP, FTP) là rất khó, so với giao thức HTTP. Thứ hai, đa số các cuộc tấn công mạng nhắm đích vào giao thức HTTP [3, 4].

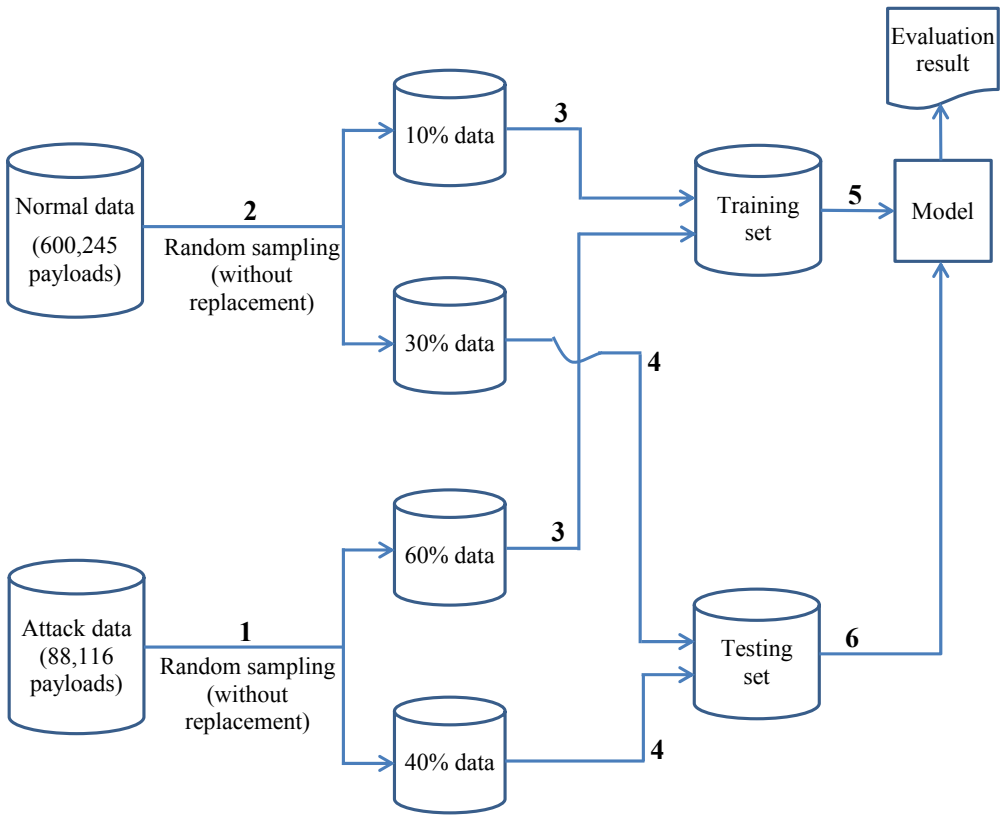
Như đã được đề cập trong Mục III, chúng tôi diễn đạt vấn đề nhận dạng payload dưới dạng bài toán phân loại hai lớp. Do đó, tập dữ liệu gồm payload độc (malicious payloads) và payload thường (normal payloads) được đòi hỏi cho việc tạo mô hình máy học. Đối với tập dữ liệu payload độc, chúng tôi thu thập từ 3 nguồn được chia sẻ từ cộng đồng nghiên cứu [15, 16, 17]. Tổng số lượng thu thập từ 3 nguồn này gồm 88,116 malicious payloads.

Đối với tập dữ liệu payload thường, chúng tôi thu thập từ 2 nguồn chính yếu. Nguồn thứ nhất là từ không gian mạng của trường đại học chúng tôi. Nguồn thứ hai là từ sự truy cập nhiều trang Web khác nhau (như Tin tức, Thể thao, Văn hóa, Khoa học, Giáo dục, Đời sống) để có được tập dữ liệu đa dạng hơn. Tổng số lượng thu thập từ 2 nguồn trên gồm 600,245 payloads. Mặc dù không được nhãn hóa, chúng tôi giả định rằng tập dữ liệu này có nhãn *thường* (normal), vì hai lý do cốt yếu. Thứ nhất, trong suốt thời gian thu thập dữ liệu, không gian mạng của chúng tôi được bảo vệ bằng những công cụ an ninh mạng, như Firewalls và Kaspersky Internet Security. Thứ hai, thậm chí nếu tồn tại những kiểu tấn công vụng trộm/dai dẳng trong quá trình thu thập dữ liệu, tỷ lệ của *dữ liệu tấn công* so với *dữ liệu thường* là không đáng kể. Tỷ lệ này được xem như là mức độ nhiễu có thể chịu đựng được (tolerable noise) trong tập dữ liệu lớn.

B. Bố trí thực nghiệm

Trên thực tế, số lượng payload độc ít hơn rất nhiều so với payload thường. Điều này dẫn đến vấn đề lệch lớp (imbalanced class), gây tác động không đúng đến các số đo thống kê của mô hình máy học. Vì thế, chúng tôi đánh giá thực nghiệm thông qua việc lấy mẫu dữ liệu gồm 6 bước như trong Hình 7. Theo đó, phần trăm mẫu trong các bước từ 1 đến 4 được xác định theo hai nguyên tắc: (1) cân bằng phân phối lớp đối với tập huấn luyện nhằm giải quyết vấn đề lệch lớp và (2) tạo phân phối lệch lớp đối với tập kiểm tra nhằm thể hiện bản chất của môi trường thực tiễn (đó là, tỷ lệ payload độc ít hơn nhiều so với payload thường). Trong bước 5, chúng tôi xây dựng mô hình máy học, sử dụng tập huấn luyện được lấy mẫu trước đó. Cuối cùng, mô hình máy học được đánh giá trong bước 6, sử dụng tập kiểm tra.

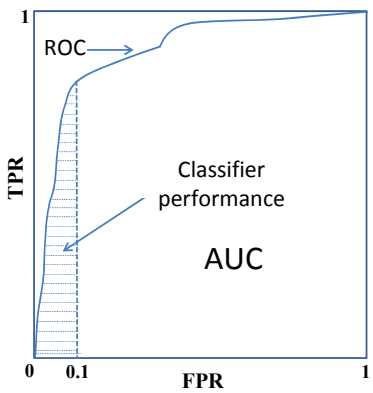
Chúng tôi đánh giá kết quả thực nghiệm dựa trên trung bình của 20 lần chạy. Cụ thể hơn, đối với mỗi giải thuật máy học, chúng tôi cho thực thi 20 lần các bước 1 – 6 trong Hình 7 và rồi lấy kết quả trung bình của 20 lần chạy.



Hình 7. Bố trí thực nghiệm

C. Độ đo đánh giá mô hình

Trong thực nghiệm của chúng tôi, các mô hình máy học được đánh giá thông qua độ đo AUC (Area Under Curve). Đây là một độ đo thông dụng lĩnh vực khai khoáng dữ liệu và máy học. Về cơ bản, AUC là tổng diện tích trong không gian ROC (Receiver Operating Characteristic) của tỷ lệ nhận dạng sai FPR (false positive rate) và tỷ lệ nhận dạng đúng TPR (true positive rate) trong cận [0, 1]. Tuy nhiên, trong thực tiễn, quản trị viên hệ thống mạng hiếm khi thiết lập tham số để chịu đựng tỷ lệ FPR cao, bởi vì việc xử lý số lượng lớn của các cảnh báo sai (false alarms or false positives) là một gánh nặng. Vì thế, chúng tôi tính AUC trong cận [0, 0.1] (gọi tắc là $AUC_{[0, 0.1]}$), thay vì cận [0, 1], như trong Hình 8. Cuối cùng, giá trị $AUC_{[0, 0.1]}$ được nhân cho 10 để chuẩn hóa thành cận [0, 1]. $AUC_{[0, 0.1]}$ cũng được sử dụng rộng rãi trong các bài toán liên quan [3, 9].



Hình 8. Độ đo $AUC_{[0, 0.1]}$

D. Thiết lập tham số

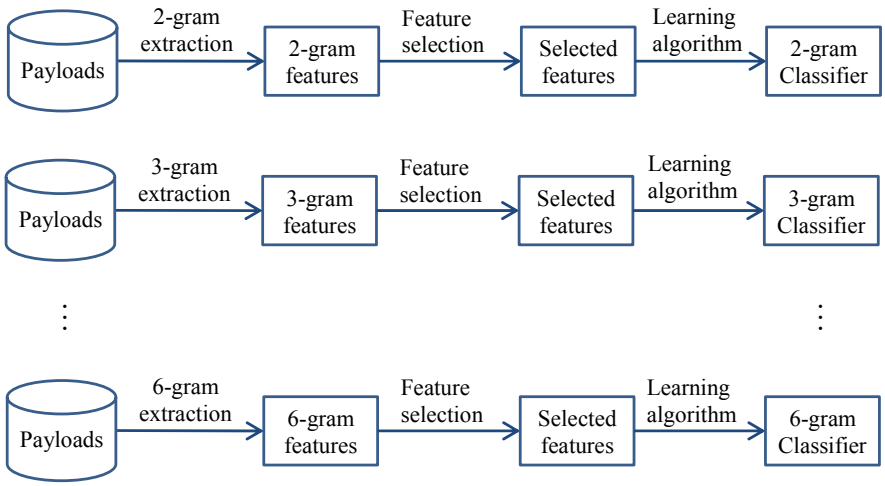
Phương pháp chúng tôi đề xuất đòi hỏi 3 tham số chính yếu. Trong đó, hai tham số đầu (n và p) liên quan đến kỹ thuật trích đặc trưng n_p-grams, tham số còn lại (k) được sử dụng để chọn đặc trưng. Đối với kỹ thuật n_p-grams, chúng tôi thay đổi n từ 2 đến 6 và p từ 0 đến 4 cho mỗi biểu diễn khác nhau của dữ liệu. Sự thiết lập này tạo ra ($|n|\times|p|$=25) mô hình thành viên, một số lượng vừa phải trong ngữ cảnh tập mô hình [10]. Như đã đề cập trong Mục III, n được tham số hóa với những giá trị nhỏ, trong khi p là không hạn chế. Tuy nhiên, thông qua một vài thực nghiệm thí điểm, chúng tôi nhận thấy rằng tập dữ liệu được thu thập có nhiều payloads với chiều dài tương đối nhỏ (ví dụ: khoảng vài trăm bytes). Đó cũng chính là lý do mà chúng tôi chọn p nhỏ, trong cận từ 0 đến 4.

Tham số *k* được chọn thông qua một vài quan sát từ thực nghiệm thí điểm, sử dụng độ đo *information gain* (*IG*) như đã mô tả trong Mục III. Chính xác hơn, thông qua thực nghiệm thí điểm, chúng tôi nhận thấy rằng tồn tại một số lượng lớn các đặc trưng có giá trị *IG* bằng 0. Những đặc trưng này bị loại bỏ, nhằm cải thiện tốc độ tính toán trong cả hai giai đoạn huấn luyện và phân loại. Chúng tôi cũng quan sát rằng có khoảng 10,000 đặc trưng với giá trị *IG* lớn hơn 0. Do đó, chúng tôi chọn *(k*=10,000), một con số tương đối lớn nhằm tránh mất nhiều thông tin.

E. Phương pháp so sánh

Chúng tôi so sánh phương pháp đề xuất với phương pháp sử dụng kỹ thuật *n*-grams truyền thống. Hình 9 mô tả tổng quan về phương pháp được so sánh. Như với kỹ thuật n_p-grams, chúng tôi tham số hóa n = {2, 3, …, 6} cho kỹ thuật *n*-grams. Bên cạnh đó, đặc trưng 1-gram cũng được loại bỏ, vì những nghiên cứu trước đây đã chỉ ra rằng mô hình máy học dựa trên tập đặc trưng 1-gram có độ chính xác thấp [2, 3, 6]. Sau khi trích các đặc trưng *n*-grams, chúng tôi sử dụng độ đo *IG* để chọn tập đặc trưng có giá trị *IG* cao. Số lượng đặc trưng *n*-grams được chọn bằng với số lượng đặc trưng n_p-grams, đó là *k*=10,000. Nhằm cô đọng trong diễn đạt, chúng tôi gọi các mô hình máy học được tạo từ tập đặc trưng *n*-grams là *n*-grams classifiers, ví dụ như 3-gram classifiers. Tương tự, chúng tôi gọi các mô hình máy học được tạo từ tập đặc trưng n_p-grams là n_p-grams classifiers, ví dụ như 3_p-gram classifiers. Ngoài ra, chúng tôi cũng mở rộng so sánh phương pháp đề xuất với tập mô hình *n*-grams (gọi tắc là *n*-grams ensemble).

Như đã được đề cập trong Mục III, chúng tôi chủ ý sử dụng Linear Proximal SVM [13] để xây dựng các n_p-grams classifiers. Tuy nhiên, để có cái nhìn rộng hơn, chúng tôi mở rộng thực nghiệm với ba giải thuật: Decision Tree [5], Naïve Bayesian [14] và Linear Proximal SVM [13]. Những giải thuật này có thời gian huấn luyện và phân loại tương đối nhanh. Cho mục đích đơn giản, mỗi giải thuật máy học được tham số hóa bởi các giá trị mặc định được thiết lập sẵn.



Hình 9. *n*-gram classifiers

F. Kết quả thực nghiệm

Kết quả thực nghiệm được chỉ ra chi tiết trong các Bảng 1 – 2 và các Hình 10 – 13. Những ký hiệu trong các bảng và hình mang ý nghĩa như sau:

- *E*(*n*-gram DTrees) là tập mô hình gồm 5 *n*-gram classifiers (*n*=2,3,...,6), mỗi classifier được xây dựng sử dụng giải thuật Decision Tree (DTree).
- *E*(*n*-gram NBs) là tập mô hình gồm 5 *n*-gram classifiers (*n*=2,3,...,6), mỗi classifier được xây dựng sử dụng giải thuật Naive Bayesian inducer (NB).
- *E*(*n*-gram SVMs) là tập mô hình gồm 5 *n*-gram classifiers (*n*=2,3,...,6), mỗi classifier được xây dựng sử dụng giải thuật Linear Proximal SVM (LP-SVM).
- *E*(*n*-gram DTrees, *n*-gram NBs, *n*-gram SVMs) là tập mô hình gồm 15 *n*-gram classifiers (với 5 *n*-gram DTrees, 5 *n*-gram NBs và 5 *n*-gram SVMs), trong đó *n*=2...6.
- *E*(n_p-gram SVMs) là tập mô hình gồm 25 n_p-gram classifiers (n=2…6; p =0…4), mỗi classifier được xây dựng sử dụng giải thuật LP-SVM.
- *E*(2_p-gram SVMs) là tập mô hình gồm 5 2_p-gram LP-SVM classifiers (p=0…4), *E*(3_p-gram SVMs) là tập mô hình gồm 5 3_p-gram LP-SVM classifiers (p = 0…4), và tương tự như thế.

Bảng 1. AUC of *n*-gram classifiers

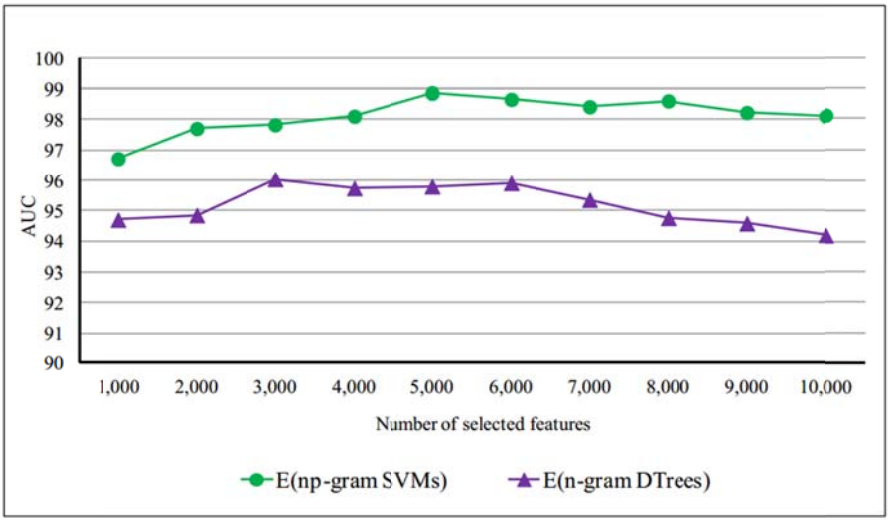
No.	Length of *n* -gram	AUC (%)		
		DTree	NB	SVM
1.	n=2	93.74	92.93	93.25
2.	n=3	93.97	93.76	93.47
3.	n=4	94.55	94.66	93.92
4.	n=5	94.44	94.31	94.78
5.	n=6	95.15	94.54	94.31

Bảng 2. AUC of *n*-gram and n_p-gram ensembles

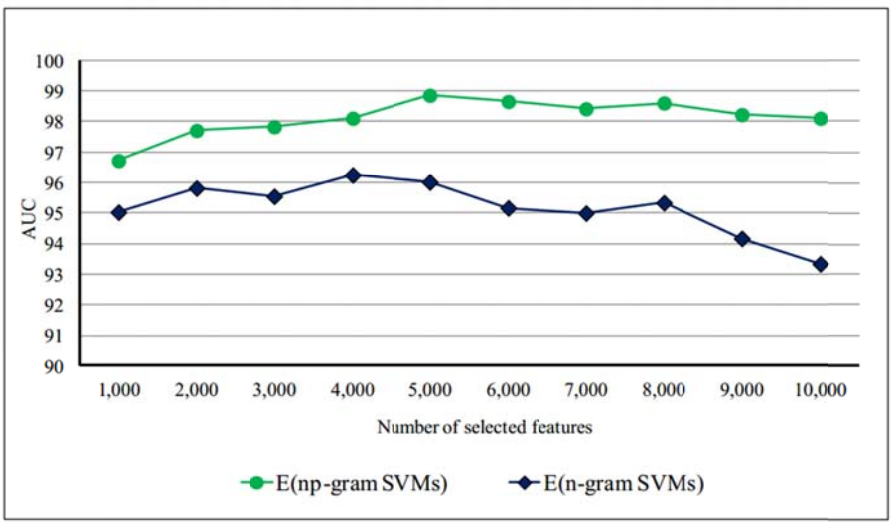
No.	Ensemble-based detector		AUC (%)
1.	E(*n* -gram DTrees)	n=2…6	95.80
2.	E(*n* -gram NBs)	n=2…6	95.42
3.	E(*n* -gram SVMs)	n=2…6	96.02
4.	$E(2_p$ -gram SVMs)	p=0…4	96.35
5.	$E(3_p$ -gram SVMs)	p=0…4	97.05
6.	$E(4_p$ -gram SVMs)	p=0…4	96.72
7.	$E(5_p$ -gram SVMs)	p=0…4	96.52
8.	$E(6_p$ -gram SVMs)	p=0…4	96.67
9.	E(*n* -gram DTrees, *n* -gram NBs, *n* -gram SVMs)	n=2…6	96.75
10.	$E(n_p$ -gram SVMs)	n=2…6, p=0…4	98.86

Như được thấy từ Bảng 1, AUC của các *n*-gram classifiers là tương đối thấp. Mặc dù *n*-gram classifiers bậc cao (*n* lớn) không luôn luôn sinh ra AUC tốt nhất, nhưng nhìn chung chúng có AUC cao hơn *n*-gram classifiers bậc thấp (*n* nhỏ). Tuy nhiên, việc chọn một mô hình đơn tốt nhất là không dễ, vì AUC của các *n*-gram classifiers dao động trong một khoảng tương đối lớn, trong khi không có giá trị tối ưu nào của *n* được tìm thấy cho cả 3 loại *n*-gram classifiers (DTree, NB và SVM). Hay nói cách khác, SVM cho kết quả tốt với 5-grams, trong khi đó NB và DTree cho kết quả tốt với 4-grams và 6-grams, một cách tương ứng. Kết quả thực nghiệm cũng cho thấy rằng, ngoài việc nhạy cảm đối với tham số *n*, *n*-gram classifiers cũng nhạy cảm đối với các giải thuật máy học. Do đó, việc sử dụng tập mô hình (ensemble of *n*-gram classifiers) chẳng những có thể làm nhẹ đi vấn đề nhạy cảm vừa nêu, mà còn cho kết quả tốt hơn. Thực vậy, như được thấy Bảng 2, AUC của *n*-gram ensembles cao hơn (trung bình khoảng 1.6%) so với *n*-gram classifiers.

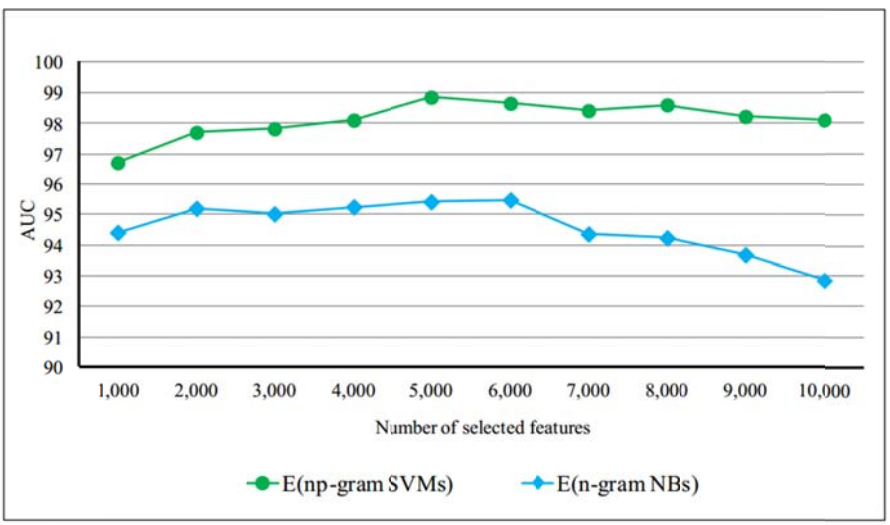
Kết quả từ Bảng 2 cũng cho thấy rằng, với số lượng mô hình thành viên bằng nhau (5 mô hình thành viên), nhưng n_p-gram ensembles cho kết quả tốt hơn đáng kể so với *n*-gram ensembles. Thậm chí, tập mô hình gồm 5 thành viên 3_p-gram SVM classifier (dòng 5, Bảng 2) có AUC cao hơn tập mô hình gồm 15 thành viên *n*-gram classifiers (dòng 9, Bảng 2). Tuy nhiên, nhằm tránh vấn đề học vẹt (overfitting) cũng như giảm độ nhạy cảm đối với việc chọn một giá trị *n* cố định, chúng tôi đề xuất sử dụng một tập mô hình gồm các thành viên n_p-gram SVM classifiers với cả *n* và *p* thay đổi. Thực vậy, AUC trung bình của tập mô hình gồm 25 thành viên n_p-gram SVM classifiers (dòng 10, Bảng 2) cao hơn 4.8% so với AUC trung bình của tất cả *n*-gram classifiers.



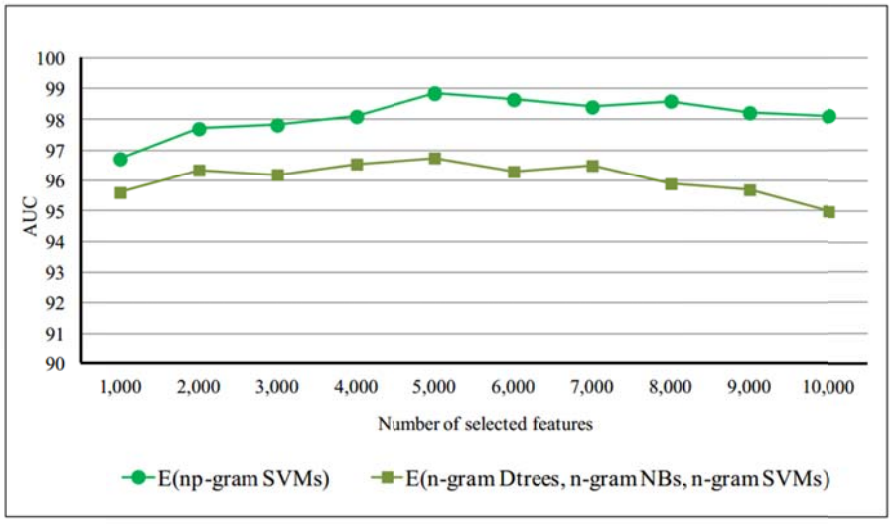
Hình 10. AUC of *E*(*np*-gram SVMs) versus *E*(*n*-gram DTrees)



Hình 11. AUC of *E*(*np*-gram SVMs) versus *E*(*n*-gram NBs)



Hình 12. AUC of *E*(*np*-gram SVMs) versus *E*(*n*-gram SVMs)



Hình 13. AUC of $E(n_p$-gram SVMs) versus $E(n$-gram DTrees, n-gramNBs, n-gram SVMs)

Để đánh giá tác động của việc chọn đặc trưng (feature selection), chúng tôi mở rộng thực nghiệm với số lượng khác nhau của các đặc trưng được chọn. Hay nói cách khác, chúng tôi chọn k = {1000, 2000, …, 10000}. Hơn nữa, vì tập mô hình cho kết quả tốt hơn mô hình đơn, nên chúng tôi tập trung so sánh n_p-gram ensembles với n-gram ensembles. Kết quả thực nghiệm mở rộng này được trình bày trong các Hình 10 – 13.

Như được thấy từ các Hình 10 – 13, khi k nhỏ, mức độ chênh lệch AUC cũng thấp, bởi vì có lẽ thiếu những đặc trưng phù hợp (relevant features). Khi k đủ lớn, mức độ chênh lệch AUC tăng đáng kể. Đáng chú ý là khi k >=7000, AUC của tất cả n-gram ensembles giảm đáng kể, trong khi AUC của các n_p-gram ensembles vẫn giữ tương đối ổn định. Điều này phần nào nói lên rằng n_p-gram ensembles ít nhạy cảm hơn n-gram ensembles đối với việc chọn đặc trưng. Tóm lại, thực nghiệm cho thấy phương pháp chúng tôi đề xuất cho kết quả tốt hơn các phương pháp được so sánh.

V. KẾT LUẬN

Nhận dạng payload độc là vấn đề khó, đòi hỏi phân tích sâu vào nội dung của gói tin để tìm ra bằng chứng mã độc. Chúng tôi đã trình bày trong bài báo này một phương pháp mới và hiệu quả để xây dựng tập mô hình nhận dạng payload độc. Theo đó, vấn đề nhận dạng payload được diễn đạt trong khuôn khổ của quy trình khai khoáng dữ liệu văn bản (text mining). Trong khuôn khổ này, chúng tôi đề xuất một kỹ thuật biểu diễn dữ liệu, gọi là n_p-grams, nhằm giành được các tập đặc trưng có chất lượng. Những vấn đề về bùng nổ không gian chiều của tập dữ liệu cũng được thảo luận và xử lý. Dựa vào kỹ thuật n_p-grams, chúng tôi đã đưa ra phương pháp tạo tập mô hình hiệu quả.

Chúng tôi đã kiểm chứng và so sánh thực nghiệm phương pháp đề xuất với các phương pháp thông dụng khác, sử dụng tập dữ liệu thực. Việc thực nghiệm cũng được mở rộng trên nhiều phương diện khác nhau, và được đánh giá bằng các độ đo chuẩn. Kết quả thực nghiệm cho thấy, phương pháp chúng tôi đề xuất cho kết quả tốt hơn các phương pháp được so sánh. Chúng tôi tin rằng nỗ lực của chúng tôi trong bài báo này có thể đóng góp những khía cạnh khoa học và thực nghiệm đến cả hai đối tượng: cộng đồng nghiên cứu máy học và cộng đồng nghiên cứu an ninh mạng.

VI. TÀI LIỆU THAM KHẢO

1. T.A. Longstaff et al., “Security of the internet”, The Kent Encyclopedia of Telecommunications, v.5, pp. 231–255.
2. K. Wang, G. F. Cretu, and S. J. Stolfo, “Anomalous payload-based worm detection and signature generation”, in RAID, ser. Lecture Notes in Computer Science, A. Valdes and D. Zamboni, Eds., vol. 3858. Springer, 2005, pp. 227–246.
3. D. Ariu, R. Tronci, and G. Giacinto, “HMMPayl: An intrusion detection system based on Hidden Markov Models”, Computers & Security, vol. 30, no. 4, pp. 221–241, 2011.
4. R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, “McPAD: A multiple classifier system for accurate payload-based anomaly detection”, Computer Networks, vol. 53, no. 6, pp. 864–881, 2009.
5. J. R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann, 1993.
6. K. Wang and S. J. Stolfo, “Anomalous payload-based network intrusion detection”, in RAID, ser. Lecture Notes in Computer Science, E. Jonsson, A. Valdes, and M. Almgren, Eds., vol. 3224. Springer, 2004, pp. 203–222.
7. D. Bolzoni, S. Etalle, P. H. Hartel, and E. Zambon, “POSEIDON: a 2-tier anomaly-based network intrusion detection system”, in IWIA. IEEE Computer Society, 2006, pp. 144–156.
8. R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification. Wiley, 2001.

9. K. Rieck and P. Laskov, “Language models for detection of unknown attacks in network traffic”, Journal in Computer Virology, vol. 2, no. 4, pp. 243–256, 2007.

10. L. I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms (Chapter 4). New Jersey: Wiley, 2004.

11. M. V. Gundy, D. Balzarotti, and G. Vigna, “Catch me, if you can: Evading network signatures with web-based polymorphic worms”, in Proceedings of the First WOOT, Boston, MA, August 2007.

    a. Ikinci, “Introduction to malicious web sites,” OWASP Turkey, Tech. Rep., 2012.

12. G. Fung and O. L. Mangasarian, “Proximal support vector machine classifiers,” in KDD, 2001, pp. 77–86.

13. G. H. John and P. Langley, “Estimating continuous distributions in Bayesian classifiers”, in Proceedings of the 11th International Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann, 1995, pp. 338–345.

14. Security Focus, http://www.securityfocus.com/archive.

15. University of Georgia, http://www.cs.uga.edu/~perdisci.

16. I-PI, http://www.i-pi.com/HTTP-attacks-JoCN-2006.