Zərərli Proqramların Aşkarlanması Üçün Maşın Təlimi Metodlarının Tətbiqi

Yadigar İmamverdiyev[1], Elşad Kərimov[2]
[1,2]AMEA İnformasiya Texnologiyaları İnstitutu, Bakı, Azərbaycan
[1]*yadigar@lan.ab.az*, [2]*elsad.k.1994@gmail.com*

***Xülasə*— Məlum zərərli proqamları (malware) müəyyən etmək üçün siqnatura və evristika əsaslı aşkarlama metodlarından geniş istifadə olunur. Lakin qarşıya ilk dəfə çıxan zərərli proqramları bu metodlarla aşkarlamaq mümkün deyil. Bu problemi maşın təlimi metodlarından istifadə etməklə həll etmək olar. Baxılan işdə PE (portable executable) formatlı icra edilə bilən zərərli və zərərsiz fayllardan istifadə edilir. Bu fayllardan PE başlığı, DLL(Dinamic Link Library – Dinamik əlaqə kitabxanası) adları və DLL daxilindəki müraciət olunan funksiyaların adları götürülərək baza yaradılır. Bu bazadan istifadə edərək maşın təlimi metodları ilə ilk dəfə rast gəlinən zərərli proqramların aşkarlanmasına baxılır.**

Açar sözlər*— *zərərli proqram, maşın təlimi, siqnatura, sıfırıncı gün, portable executable, təsnifat

I. GİRİŞ

İlk dəfə Con Fon Neyman tərəfindən öz-özünü kopyalaya bilən komputer proqramı fikri irəli sürülmüşdür [1]. 1984-cü ildə Fred Kohen tərəfindən hazırlanan məqalədə "virus" termini işlədilmişdir [2]. Zərərli proqramlar 1980-ci illərdə fərdi kompüterlərin sürətli inkişafı ilə əlaqədar olaraq geniş yayılmağa başlamışdır. Zərərli proqramlar müasir informasiya cəmiyyətinin əsas problemlərindən biridir. Zərərli proqramların trojan, worm, virus, spam və s. kimi müxtəlif növləri vardır. Hər bir zərərli proqramın məqsədi və fəaliyyəti də müxtəlifdir. Zərərli proqramları aşkarlamaq üçün müxtəlif üsul və metodlar istifadə edən proqramlar (antiviruslar və s.) mövcuddur. Lakin bu metodların bir çoxu siqnatura əsaslı olub, məlum olan zərərli proqramların aşkarlanmasını təmin edir. Yeni yaradılmış zərərli proqramların siqnaturası məlum olmadığından ilk aşkarlandığı anda bu metodlar vasitəsilə qarşısının alınması mümkün olmur. Çünki əvvəlcə zərərli proqram analiz edilməli, ondan lazımı məlumatlar alınmalıdır. Sonda isə mövcud proqramlar əldə olunan yeni məlumatlara əsasən yenilənib istifadəçilərə çatdırılmalıdır. Göründüyü kimi burada müəyyən vaxt tələb olunur və bəzən yeni yaradılmış zərərli proqramı analiz etmək mümkün olmur. Yaranmış bu problemi maşın təlimi metodları ilə aradan qaldırmaq mümkündür. Maşın təlimi metodları ilə ilk dəfə rast gəlinən zərərli proqramlar aşkarlana bilər. Bunun üçün maşın təliminin SVM (Support vector machine – Dayaq vektor maşını), Naive Bayes üsulu, neyron şəbəkələr və s. kimi metodlarından geniş istifadə olunur. Maşın təlimi metodları əvvəlcədən məlum olan zərərli proqramlardan alınmış məlumatlara əsasən ilk dəfə müəyyən edilən zərərli proqramları yüksək dəqiqliklə aşkarlaya bilir.

Baxılan işdə PE (portable execute) formatlı icra edilə bilən zərərli və zərərsiz fayllardan istifadə olunur. Bu fayllardan alınmış əlamətlər vektoru əsasında baza yaradılır. Sonda isə bu bazadan istifadə edərək maşın təlimi metodları ilə ilk dəfə rast gəlinən zərərli proqramların aşkarlanmasına baxılır.

II. ƏLAQƏDAR TƏDQİQATLAR

Zərərli proqramlar daim informasiya cəmiyyəti üçün böyük problemlərə səbəb olmuşdur. Bu səbəbdən zərərli proqramların aşkarlanması üçün çoxlu sayda tədqiqatlar aparılmışdır. Tədqiqatlarda əsasən statik və dinamik analiz üsullarından istifadə olunur. Statik analizdə zərərli proqramı işə salmadan müəyyən etməyə çalışılır. Dinamik analizdə isə zərərli proqram işə salınır və proqramın davranışına, hara müraciət etdiyinə, hansı funksiyalardan, kitabxanalardan istifadə etdiyinə və s. görə aşkarlanmasına baxılır.

Son illərdə zərərli proqramların sürətli inkişafı onların aşkarlanmasını da çətinləşdirir. Nəticədə bir çox tədqiqatçı ilk dəfə rast gəlinən (ing. zero day – sıfırıncı gün) zərərli proqramları aşkarlamaq üçün maşın təlimi metodlarının istifadəsinə yönəlməyə başladı. Tədqiqatçıların çoxu zərərli proqram nümunələrini təsvir etmək üçün ilkin əlamətlər olaraq *n-gram* və ya API (*Application Programming Interface* – Tətbiqi proqramlaşdırma interfeysi) müraciətlərini istifadə edirdi.

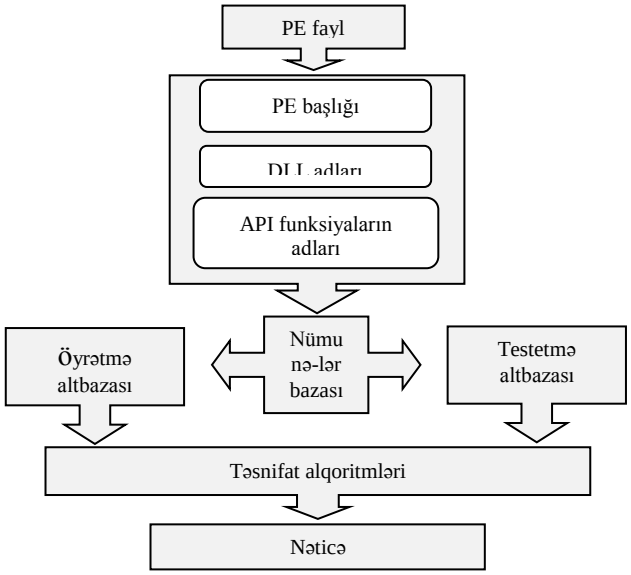
Shultz və b. yeni zərərli proqramları aşkarlamaq üçün maşın təlimindən istifadə edərək yeni metod təklif etdi [3]. İcra oluna bilən fayllardan DLL-lərin siyahısı, DLL-dəki funksiyalara olan müraciətlərin siyahısı və hər DLL-də istifadə olunan müxtəlif sistem müraciətlərinin sayı olmaqla üç əlamət götürülür. Baza 3265 zərərli, 1001 zərərsiz olan 4266 fayldan ibarətdir. Öyrənmə alqoritmi olaraq Naive Bayes (NB) üsulundan istifadə olunmuşdur. Baza öyrətmə və test etmək üçün iki yerə bölünmüşdür. Nəticədə Naive Bayes algoritmi 97.11% dəqiqliklə ən yüksək təsnifat nəticəsi göstərmişdir. Tədqiqatçılar nəticələrini siqnatura əsaslı metodlarla müqayisə edirlər və maşın təlimi ilə yeni zərərli proqramların aşkarlanmasının siqnatura əsaslı alqoritmlərdən 2 dəfə daha effektiv olduğunu iddia edirlər.

Kolter və Maloof icra edilə bilən zərərli proqramları aşkarlamaq üçün n-gram analizindən və maşın təlimi metodlarından istifadə etmişdir [4]. Tədqiqatçılar icra edilə bilən proqramları ASCII (American Standard Code for Information Interchange – İnformasiya mübadiləsi üçün Amerika Standart Kodu) formatında 16-lıq formata çevirib, hər dörd baytlıq ardıcıllığı birləşdirərək n-gram əlamətlərini müəyyən ediblər. Baza 1971 zərərli və 1651 zərərsiz proqramdan ibarətdir. Təsnifat üçün SVM, Qərar ağacı (*Decision Tree*), *Naive Bayes* və s. kimi müxtəlif metodlardan istifadə olunmuşdur.

DOI: 10.25045/NCSoftEng.2017.44

III. ZƏRƏRLİ PROQRAMLARIN AŞKARLANMASI SİSTEMİNİN ARXİTEKTURASI

Zərərli proqramları aşkar etmək üçün istifadə olunan arxitektura üç moduldan ibarətdir. Birinci modulda icra edilə bilən fayldan əlamətlər götürülür [5]. Bunun üçün Microsoft Visual Studio 2015 proqramından istifadə edilir [6]. İkinci modulda SQLite kitabxanasının köməkliyi ilə məlumatlar bazaya yazılır [7]. Yaradılmış baza təsadüfi olaraq testetmə və öyrətmə altbazalarına bölünür. Son olaraq isə WEKA proqramından istifadə edilir [8]. Öyrətmə altbazasından istifadə edərək maşın təlimi metodları ilə təsnifat prosesi reallaşdırılır. Sonda isə testetmə altbazası ilə sistemin effektliyi sınaqdan keçirilir. İstifadə edilən arxitektura şəkil 1-də göstərilmişdir.



Şəkil 1. Zərərli proqramların aşkarlanması sisteminin arxitekturası

İcra edilə bilən fayllar bir çox informasiyanı özündə saxlayır [5,9]. Bu məlumatlar zərərli faylları aşkarlamaq üçün çox əhəmiyyətlidir, amma bəzi əlamətlər də var ki, zərərli faylları təyin etmək üçün onların əhəmiyyətli bir rolu olmur. Baxılan işdə zərərli faylları aşkarlamaq üçün aşağıdakı cədvəldə verilmiş əlamətlərdən istifadə edilir:

CƏDVƏL 1. İCRA EDILƏ BILƏN FAYLDAN GÖTÜRÜLƏN ƏLAMƏTLƏR

	əlamətlər	**tipi**	**sayı**
1.	PE DOS header	integer	31
2.	PE file header	integer	7
3.	PE optinal header	integer	30
4.	PE data directories	integer	16
5.	PE section headers	integer	30
6.	PE delay import	integer	8
7.	PE TLS table	integer	6
8.	DLLs	binary	63
9.	DLL functions	binary	1802
Cəmi			1993

IV. BAZANIN YARADILMASI

Microsoft Visual Studio 2015 və SQLite kitabxanasından istifadə edərək icra edilə bilən fayllardan götürülmüş məlumatlar əsasında baza yaradılır [6,7]. Baza ümumilikdə 3086 sayda icra edilə bilən fayldan ibarətdir. Bu fayllardan 2156 zərərli, 930 isə zərərsiz fayldır. Zərərli faylar VX Heaven Virus Collection bazasından əldə edilmişdir [10]. Zərərsiz fayllar olaraq Windows sistem faylları götürülmüşdür. Təsnifat üçün Weka proqramından istifadə edilmişdir [8]. SQLite kitabxanasından istifadə edilərək zərərli və zərərsiz fayllardan ibarət yaradılmış baza SQLite Studio proqramı vasitəsi ilə Weka proqramının dəstəklədiyi CSV (Comma Separated Values) formatına çevrilmişdir [7,8].

V. TƏSNİFAT METODLARININ SEÇİLMƏSİ

Baxılan işdə təsnifat üçün maşın təliminin *Naive Bayes*, *J48*, Təsadüfi Meşə (*Random Forest*), IBk (*İnstance Based Learner*) və SVM (*Support Vector Machine*) metodlarından istifadə edilir.

Naive Bayes metodunda fərz edilir ki, $x \in X$ obyektləri statistik asılı olmayan *n* əlamətlə təsvir olunur:

$$X = (\xi_1,...,\xi_n) = (f(x_1),..., f(x_n))$$

Asılı olmama fərziyyəsi o deməkdir ki, siniflərin həqiqətə oxşarlıq funksiyalarını $p_y(x) = p_{y1}(\xi_1)..p_{yn}(\xi_n)$ şəklində göstərmək olar, burada $p_{yj}(\xi_j)$ – y sinfində j-cu əlamətin qiymətlərinin paylanma sıxlığıdır.

Asılı olmama fərziyyəsi məsələni əhəmiyyətli dərəcədə sadələşdirir, çünki birölçülü sıxlıqları qiymətləndirmək, *n*-ölçülü paylanma sıxlığını qiymətləndirməkdən daha asandır. Təəssüf ki, bu fərziyyə praktikada nadir hallarda yerinə yetirilir, buradan da metodun adı yaranmışdır.

Naive Bayes metodunun qərar qaydası

$$h(x) = \arg\max_{y \in Y} \prod_{i=1}^{n} p(x_i \mid y)\, p(y)$$

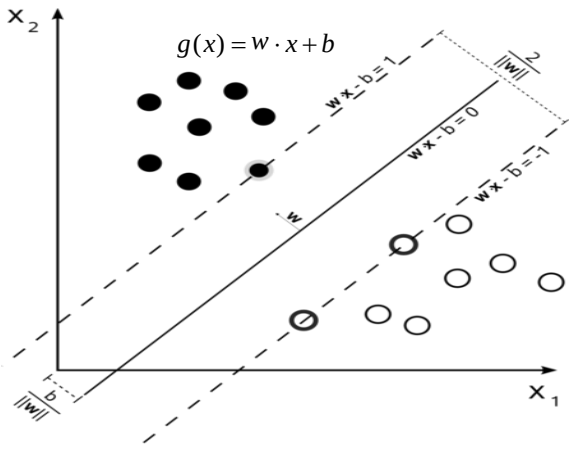
şəklindədir. Beləliklə, Naive Bayes təsnifat alqoritminin öyrədilməsi üçün siniflərin $p(y)$ aprior ehtimallarını və $p(x_i|y)$ şərti ehtimallarını qiymətləndirmək lazımdır.

J48 təsnifat alqoritmi C4.5 Qərar ağacı (*Decision Tree*) alqoritminin WEKA proqramına uyğunlaşdırılmış formasıdır [12,13]. Qərar ağacı alqoritmləri vəziyyətlər və nümunələr qrupları ilə başlayaraq, yeni vəziyyətləri təsnifat edə bilmək üçün ağac strukturu yaradırlar. Ağacın düyünlərində növbəti addımda hansı budaq üzrə hərəkət ediləcəyi barədə informasiya saxlanılır. Yarpalar sinifləri əks etdirir və yarpaq-lara çatdıqda alqoritm burada olan informasiya əsasında faylın hansı sinfə aid olduğuna qərar verir.

Təsadüfi Meşə (*Random Forest*) təsnifat alqoritmi də Qərar ağacı alqoritmlərinə aiddir [14]. Təsadüfi Meşə metodu bir ağac strukturu yaratmaq əvəzinə çox sayda və çox dəyişənli ağac strukturlarının yaradılmasını təklif edir. Ağac strukturlarının hamısı bazadan təsadüfi olaraq seçilmiş müxtəlif öyrətmə altbazaları ilə öyrədilir. Sonda bütün ağac struktur-

DOI: 10.25045/NCSoftEng.2017.44

larından alınmış nəticələr birləşdirilərək təsnifat prosesi yerinə yetirilir. Yəni, Təsadüfi Meşə metodu bir təsnifat alqoritmi əvəzinə birdən çox təsnifat alqoritmi yaradan və bu təsnifat alqoritmlərindən alınmış nəticələri birləşdirərək yeni faylın hansı sinfə aid olduğunu müəyyən edən alqoritmdir.



Şəkil 2. Hipermüstəvi ilə iki sinfin xətti ayrılması halı

b – sürüşmə; w – normal vektor.

SVM (*Support Vector Machine*) metodu – xətti ayrılan iki sinif üçün təlim (nümunələr) çoxluğunun ən yaxın nöqtələrindən maksimum məsafədə keçən ayrıcı hipermüstəvi tapmaqla obrazların tanınmasını yerinə yetirir. Bu nöqtələr dayaq vektorları adlanır (support vectors) [11,12]. Nümunələr çoxluğu $\{(x_i, y_i), i = 1,2,...,n, x \in R^m, y \in \{+1,-1\}\}$ olan xətti ayrılan iki sinif üçün xətti təsnifat funksiyası aşağıdakı kimidir:

İki sinfin bütün nümunələri $|g(x)| \geq 1$ şərtini, hipermüstəviyə digərlərindən ən yaxın yerləşən nümunələr isə $|(x)|=1$ şərtini ödəyir. Aralığın eni $\frac{2}{\|w\|}$ -dir. Optimal hipermüstəvinin qurulması məsələsi aşağıdakı kimi ifadə olunur:

$$\phi(w) = \frac{1}{2}\|w\|^2 = \frac{1}{2}(w \cdot w) \to \min$$

$$y_i[(w \cdot x_i) + b] - 1 \geq 0, \forall i$$

IBk (Instance Based Learner) təsnifat alqoritmi kNN (k-Nearest Neighbors) metodunun WEKA-da istifadə olunan formasıdır [15]. kNN alqoritmində öyrətmə üçün istifadə olunmuş sinifləri məlum olan elementlərdən alınmış əlamətlərə əsasən yeni elementin hansı sinfə aid olduğunu müəyyən etmək üçün k sayda qonşu elementə baxılır. Yəni, təsnif edilmiş siniflərdən yeni daxil edilən elementə ən yaxın olan k sayda element götürülür və yeni element götürülmüş qonşu elementlərdən hansılarına daha yaxındırsa, yeni elementin sinfi həmin elementlərin sinfi olaraq təyin edilir.

Qeyd etdiyimiz bu təsnifat alqoritmlərinin hər birində siniflər və öyrətmə üçün istifadə olunacaq nümunələrin hansı sinfə aid olduğu əvvəlcədən məlumdur. Təsnifat alqoritmləri məlum olan informasiyaya əsasən yeni faylın hansı sinfə aid olduğunu müəyyən edir.

VI. TƏSNİFAT NƏTİCƏLƏRİNİN KEYFİYYƏT XARAKTERİSTİKALARI

Təsnifat üçün yaratdığımız baza iki hissəyə bölünərək, bir hissə öyrətmə, digər hissə isə testetmə üçün istifadə edilir. Təsnifat alqoritmlərinin effektivliyini müəyyən etmək və nəticələri müqayisə edərək hansı təsnifat alqoritminin daha effektiv olduğunu aşkarlamaq üçün aşağıda verilən keyfiyyət xarakteristikalarından istifadə olunur [16,17]:

True positive (TP) – zərərsiz fayl kimi təsnif olunmuş zərərsiz faylların sayı;

False positive (FP) – zərərsiz fayl kimi təsnif olunmuş zərərli faylların sayı;

True negative (TN) – zərərli fayl kimi təsnif olunmuş zərərli faylların sayı;

False negative (FN) – zərərli fayl kimi təsnif olunmuş zərərsiz faylların sayı.

Təsnifat metodlarının effektivliyi Dəqiqlik (Precision), Tamlıq (*Recall*) və *F*-ölçü (*F-measure*) kəmiyyətlərinin qiymətləri ilə müəyyən edilir. Bu kəmiyyətlər yuxarıda qeyd edilən keyfiyyət xarakteristikalarına əsasən aşağıda göstərilmiş ifa-dələrlə hesablanır:

Dəqiqlik:
Zərərli və zərərsiz fayllar üçün dəqiqlik qiymətləri Mənfi proqnostik qiymət (*Negative predictive value - NPV*) və Müsbət proqnostik qiymət (*Positive predictive value - PPV*) adlanan iki kəmiyyətlə aşağıdakı kimi ifadə olunur:

$$NPV = \frac{TN}{TN + FN} \qquad PPV = \frac{TP}{TP + FP}$$

Tamlıq:
Uyğun olaraq tamlıq qiymətləri də Yalan pozitiv əmsal (*False positive rate - FPR*) və Doğru pozitiv əmsal (*True positive rate - TPR*) adlanan iki kəmiyyətlə aşağıdakı kimi hesablanır:

$$FPR = \frac{TN}{TN + FP} \qquad TPR = \frac{TP}{TP + FN}$$

F-ölçü:
Zərərli və zərərsiz fayllar üçün F-ölçü qiymətləri aşağıdakı iki düsturla müəyyən edilir:

$$F_{TN} = 2 \times \frac{NPV \times FPR}{NPV + FPR} \qquad F_{TP} = 2 \times \frac{PPV \times TPR}{PPV + TPR}$$

VII. EKSPERİMENTLƏR

SQLite kitabxanasından istifadə edərək yaratdığımız bazada ümumilikdə 3086 fayldan istifadə edilmişdir. Təsnifat aparmaq üçün maşın təliminin Naive Bayes, J48, Təsadüfi Meşə, IBk və SVM metodlarından istifadə edilir [18, 21]. Maşın təlimi metodları ilə təsnifat Weka proqramı ilə həyata keçirilib. Baza iki hissəyə bölünərək, bir hissə öyrətmə üçün,

DOI: 10.25045/NCSoftEng.2017.44

digər hissə isə test etmə üçün istifadə edilir. Eksperimentlər İntel Core 2 Duo 2.93GHz prosessor, 2GB RAM və Microsoft Windows 7 Ultimate əməliyyat sistemi yüklü olan komputerdə aparılmış-dır.

3086 fayldan ibarət bazanı təsadüfi olaraq iki hissəyə bölürük. Bazanın bir hissəsi, yəni 2055 fayl öyrətmə üçün istifadə olunur. Qalan 1031 fayl isə testetmə üçün istifadə edilir. Aşağıdakı cədvəllərdə maşın təliminin müxtəlif təsnifat metodlarından istifadə edilərək əldə olunmuş nəticələr göstərilmişdir [19, 20]:

CƏDVƏL 2. *NAIVE BAYES* METODU İLƏ ƏLDƏ OLUNMUŞ NƏTİCƏLƏR

	Fayllar	TP	FP	Precision	Recall	F-measure
Naive Bayes	Zərərsiz	304	13	95.89 %	92.96 %	94.40 %
	Zərərli	23	691	96.77 %	98.15 %	97.45 %

CƏDVƏL 3. *J48* METODU İLƏ ƏLDƏ OLUNMUŞ NƏTİCƏLƏR

	Fayllar	TP	FP	Precision	Recall	F-measure
J48	Zərərsiz	311	6	98.10 %	96.88 %	97.48 %
	Zərərli	10	704	98.60 %	99.15 %	98.87 %

CƏDVƏL 4. *SVM* METODU İLƏ ƏLDƏ OLUNMUŞ NƏTİCƏLƏR

	Fayllar	TP	FP	Precision	Recall	F-measure
SVM	Zərərsiz	306	11	96.52 %	95.32 %	95.91 %
	Zərərli	15	699	97.89 %	98.45 %	98.16 %

CƏDVƏL 5. *TƏSADÜFİ MEŞƏ* METODU İLƏ ƏLDƏ OLUNMUŞ NƏTİCƏLƏR

	Fayllar	TP	FP	Precision	Recall	F-measure
Random Forest	Zərərsiz	309	8	97.48 %	97.78 %	97.63 %
	Zərərli	7	707	99.01 %	98.88 %	98.94 %

CƏDVƏL 6. *IBK* METODU İLƏ ƏLDƏ OLUNMUŞ NƏTİCƏLƏR

	Fayllar	TP	FP	Precision	Recall	F-measure
IBk	Zərərsiz	298	19	94.01 %	92.55 %	93.27 %
	Zərərli	24	690	96.63 %	97.32 %	96.97 %

NƏTİCƏ

Bu məqalədə maşın təlimi metodlarından istifadə edərək zərərli proqramların aşkarlanması məsələsinə baxdıq. 2156 zərərli, 930 zərərsiz olmaqla, ümumilikdə 3086 icra edilə bilən fayldan istifadə edilmişdir. İcra edilə bilən fayllardan götürülmüş PE başlığı, DLL adları və DLL-lərdə müraciət olunan API funksiyalarından ibarət 1993 əlamət əsasında baza yaradılmışdır [22, 23]. Yaradılmış bazadan istifadə edərək təsnifat üçün Naive Bayes, J48, Təsadüfi Meşə, IBk və SVM metodları ilə eksperimentlər aparılmışdır. Aparılmış eksperimentlər nəticəsində təsnifat üçün istifadə olunan maşın təliminin beş metodundan ən yüksək dəqiqlik Təsadüfi Meşə və J48 metodlarında, ən aşağı dəqiqlik isə IBk metodunda müəyyən olunmuşdur.

Beləliklə, aparılmış tədqiqatda ilk dəfə rast gəlinən zərərli proqramları aşkarlamaq üçün maşın təlimi metodlarının effektivliyi göstərilmişdir. Siqnatura əsaslı metodların əksinə, ilk dəfə rast gəlinən zərərli proqram maşın təlimi metodları ilə yüksək dəqiqliklə həmin anda aşkarlamaq mümkündür.

ƏDƏBİYYAT

[1] J.V. Neumann, "Theory of Self-reproducing Automata," IEEE Trans. Neural Networks, Vol. 5, No. 1, pp. 3–14, 1994

[2] Cohen, F.: Computer viruses. PhD thesis. University of Southern California. 1985

[3] M. G. Schultz, E. Eskin, E. Z., and S. J. Stolfo, "Data mining methods for detection of new malicious executables," Proceedings of the IEEE Symp. on Security and Privacy, pp. 38-49, 2001

[4] J. Z. Kolter and M. A. Maloof, "Learning to Detect Malicious Executables in the wild," Proceedings of the ACM Symp. on Knowledge Discovery and Data Mining (KDD), pp. 470-478, August 2004

[5] X86 Disassembly/Windows Executable files, https://en.wikibooks.org/wiki/X86_Disassembly/Windows_Executable_Files

[6] Microsoft Visual Studio 2015, https://msdn.microsoft.com/en-us/library/dd831853.aspx

[7] Sqlite Studio, https://www.sqlite.org/

[8] WEKA, http://www.cs.waikato.ac.nz/ml/weka/

[9] Peering inside the PE: A tour of the Win32 Portable Executable file format, https://msdn.microsoft.com/en-us/library/ms809762.aspx

[10] VX Heaven, http://vxheaven.org/vl.php

[11] C.M. Bishop, "Pattern Recognition and Machine Learning", 2006.

[12] S. Theodoridis, K. Koutroumbas, "Pattern Recognitation", fourth edition, 2009

[13] Quinlan, J.R., Induction of Decision Trees, Machine Learning, 1986, pp. 81-106.

[14] A. Cutler, "Random Forests for Regression and Classification", 2010.

[15] Aha, D., Kibler, D. Instance-based learning algorithms, Machine Learning, 1991, pp. 37-66.

[16] Precision and recall, https://en.wikipedia.org/wiki/Precision_and_recall

[17] J. Davis, M.Goadrich, "The Relationship Between Precision-Recall and ROC Curves", 2005.

[18] U. Baldangombo, N. Jambaljav, S. Horng, "A static malware detection system using Data Mining methods," arXiv preprint arXiv:1308.2831, 2013.

[19] M. Sikorski, A. Honig, "Practical Malware Analysis", 2012.

[20] E.Eilam, "Reversing: Secrets of reverse engineering", chapter 8, 2005.

[21] T. M. Mitchell, "Machine Learning", 1997.

[22] M.H. Ligh, S. Adair, B. Hartstein, M. Richard, "Malware Analyst`s Cookbook and DVD: Tools and techniques for fighting malicious code", 2011.

[23] CFF Explorer, http://www.ntcore.com/exsuite.php

DOI: 10.25045/NCSoftEng.2017.44