ВЕСТНИК
ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
МОРСКОГО И РЕЧНОГО ФЛОТА ИМЕНИ АДМИРАЛА С. О. МАКАРОВА

УДК 004.02; 311.2

И. Е. Кривцова,
К. И. Салахутдинова,
И. В. Юрин

МЕТОД ИДЕНТИФИКАЦИИ ИСПОЛНЯЕМЫХ ФАЙЛОВ ПО ИХ СИГНАТУРАМ

В статье рассматривается метод защиты программных средств автоматизированных информационных систем на речном и морском транспорте на основе идентификации исполняемых elf-файлов программ, установленных на различных Linux ОС, по их сигнатурам специальной структуры. Под идентификацией следует понимать процесс распознавания некоторого файла как его отождествление с той или иной программой. Многообразие ОС Linux и открытость их исходного кода затрудняют использование стандартных методов анализа и инвентаризации исполняемых файлов. Предлагается новый метод решения этой задачи, который включает три этапа: построение архива сигнатур; фильтрация сигнатур; непосредственная идентификация файла. Сигнатуры как включаемых в архив программ, так и идентифицируемых файлов, имеют специальную структуру, основанную на частотных характеристиках. Фильтрация сигнатур осуществляется с использованием критерия Фишера (φ^*-критерия) на уровне значимости $p= 0,01$. Для более точной идентификации выполняется анализ частотных характеристик сигнатур, хранящихся в архиве, и сигнатур, полученных в результате фильтрации, по значению углового коэффициента при уровне различия менее 20 %.

Ключевые слова: информационная безопасность; исполняемый elf-файл; сигнатура файла; статистический критерий Фишера (φ^*-критерий); идентификация файлов.

Введение

В соответствии с Федеральным законом от 09.02.2007 г. № 16-ФЗ «О транспортной безопасности», целями обеспечения транспортной безопасности являются устойчивое и безопасное функционирование транспортного комплекса, а также защита интересов личности, общества и государства в сфере транспортного комплекса от актов незаконного вмешательства. Защита программно-технических средств информатизации на морском и речном транспорте от преднамеренных или непреднамеренных воздействий осуществляется путем разработки и реализации мер по обеспечению транспортной безопасности, контроля и надзора в области обеспечения транспортной безопасности, а также с помощью использования посредством информационного, материально-технического и научно-технического обеспечения транспортной безопасности.

Для обеспечения информационной безопасности (ИБ) на морском и речном транспорте применяются организационные, правовые и технические меры, создаются системы обеспечения безопасности мореплавания, которые, как и суда, имеют автоматизированные информационные системы, нуждающиеся в защите. Такие системы могут подвергнуться угрозам, эксплуатирующим уязвимость программного обеспечения (например, дефекты в программе, не декларированные возможности, нелегальное использование интеллектуальной собственности [1], [2], включение в программы вредоносного кода [3] и т. д.), что приводит к увеличению рисков ИБ. Таким образом, одной из основных задач в обеспечении информационной безопасности автоматизированных информационных систем на морском и речном транспорте является инвентаризация содержимого компьютерных систем с целью выявления несанкционированно установленных программных продуктов [4].

В настоящее время существует большое количество программных [5] – [8] и аппаратных [5], [6] продуктов анализа компьютерных систем с функцией аудита, а также методов анализа видеофайлов [9] – [11] и фотографий [12] – [14], на основе которых разрабатываются методы исследования исполняемых файлов. Но большинство из них рассчитаны на операционную систему Windows, в то время как все более широкое распространение получают Linux-системы.

Исполняемые файлы для Windows систем представлены уже в конечном (скомпилированном) виде. Для Linux систем известны несколько способов по установке исполняемых файлов [15].

Принимая во внимание многообразие операционных систем Linux и открытость их исходного кода, необходимо учитывать частоту выхода обновлений как самих операционных систем, так и программ для них. Такие особенности операционных систем Linux затрудняют использование следующих стандартных методов анализа и инвентаризации исполняемых файлов:

– ручной осмотр характерных мест установки программ;

– побайтовое сравнение;

– сравнение контрольной суммы;

– сравнение цифровой подписи;

– методы, представленные в других статьях: аудит программного обеспечения при помощи кусочного хеширования [16], определение кражи программного обеспечения при помощи идентификации программы через последовательность команд [17], осуществление поиска по двоичному коду [18], идентификация типов данных и применение статистического анализа [19], идентификация мультимедийных файлов [20].

Следовательно, актуальной является задача разработки такого метода идентификации исполняемых elf-файлов [21], при котором будет происходить их распознавание вне зависимости от версии или операционной системы Linux, на которой они расположены. Под идентификацией в данной работе следует понимать процесс распознавания некоторого файла как его отождествление с той или иной программой. Предлагается новый подход к решению этой задачи.

Объектом исследования в настоящей работе являются исполняемые файлы формата elf и их сигнатуры, предметом исследования — способы сравнения сигнатур исполняемых файлов. Целью исследования является разработка метода идентификации elf-файлов на основе анализа их сигнатур. Предлагаемый метод включает три этапа:

– построение архива сигнатур;

– фильтрация сигнатур;

– непосредственная идентификация файла.

Построение архива сигнатур. Для того, чтобы построить сигнатуру программы, необходимо проанализировать определенный объем исполняемых файлов, отождествленных с этой программой. На основании такого анализа для различных имеющихся программ создаются сигнатуры, объединяемые в архив. Каждая программа представлена бинарным кодом, т. е. в терминах математической статистики измерение признака происходит в номинативно-дихотомической шкале, а выборкой является представление исполняемого файла в бинарном виде.

Для построения архива сигнатур формируется обучающая выборка (*TS*), состоящая из elf-файлов, отождествленных с определенной имеющейся программой, сигнатура которой впоследствии будет включена в архив. Обучающую выборку можно представить в следующем виде:

$$TS = \{v_1, v_2, ..., v_m\},$$

где v_i — выборка различных программ; m — количество различных программ; $v_i = \{f_i, f_2, ..., f_n\}$; f_j — различные версии i-й программы; n — количество файлов в выборке.

Для построения сигнатуры i-й программы подсчитываются частоты нулей и единиц для каждого файла в выборке, а именно: $f_j = \{z, o\}$, где z — количество нулей, o — количество единиц.

На основании этих данных для каждой из программ используется формула

$$v_i = \frac{1}{n}\sum_{j=0}^{n} f_j\{z, o\}. \quad (1)$$

По формуле (1) вычисляются средние частоты нулей и единиц $\overline{v}_i = \{\overline{z}, \overline{o}\}$, где $\overline{z}$ — средние значения количества нулей; $\overline{o}$ — средние значения количества единиц, n — количество файлов в выборке.

Далее необходимо учитывать более индивидуальные характеристики каждого файла. Для этого нужно разбить исполняемый файл на части по одному байту.

Для каждого файла f_j рассчитывается побайтовая частотная характеристика и вычисляется усредненная частотная характеристика для различных версий i-й программы. Таким образом, получаем строку, состоящую из 256 значений:

$$\overline{L}_i = \{b_0, b_1, ..., b_{255}\},$$

где b_j — частота j-го байта.

Составление сигнатуры для каждой i-й программы происходит по следующей схеме:

1) имя программы — одна ячейка;

2) среднее количество нулей и единиц — две ячейки;

3) частотное распределение байт — 256 ячеек.

Таким образом, сигнатура S_i программы в архиве будет иметь следующую структуру:

$$S_i = \{N_i, \overline{v}_i, \overline{L}_i\},$$

где N_i — имя программы, и будет состоять из 259 ячеек. При этом отводимое количество памяти для каждой из ячеек может быть различным.

Особо отметим, что построение сигнатуры S идентифицируемого файла происходит по аналогичной схеме:

$$S = \{f, L\},$$

где f — частота нулей и единиц бинарного кода файла;

L — побайтовая частотная характеристика.

Фильтрация сигнатур. Данный этап представляет собой отсеивание сигнатур файлов, значительно отличающихся от сигнатур программ, помещенных в архив, и осуществляется с использованием многофункционального статистического критерия Фишера (φ^*-критерия). Поскольку критерий Фишера основывается на альтернативной шкале «есть признак — нет признака» [22, с. 158], одно лишь его применение не даст нужного результата — идентификации файла, однако с его помощью можно довольно быстро провести фильтрацию файлов.

Для сравнения двух сигнатур по критерию Фишера (φ^*-критерию) сформируем гипотезы:

H_0 — доля единиц в сигнатуре программы, хранящейся в архиве, не больше, чем в сигнатуре идентифицируемого файла;

H_1 — доля единиц в сигнатуре программы, хранящейся в архиве, больше, чем в сигнатуре идентифицируемого файла.

Основная гипотеза H_0 проверяется на уровне значимости p = 0,01 [23, с. 98]. Эмпирическая величина угла φ^* вычисляется по формуле

$$\varphi^*_{\text{эмп}} = (\varphi_1 - \varphi_2)\sqrt{\frac{n_1 \cdot n_2}{n_1 + n_2}}, \qquad (2)$$

где углы $\varphi_i = 2 \cdot \arcsin(\sqrt{P})$, где P — процентная доля значений признака, выраженная в долях единицы, при $\varphi_1 > \varphi_2$;

n_1, n_2 — суммарное количество значений признака в сигнатуре программы, хранящейся в архиве, и сигнатуре идентифицируемого файла соответственно.

Если при сравнении полученного значения $\varphi^*_{\text{эмп}}$ с критическим значением $\varphi^*_{\text{кр}}$ выполняется неравенство $\varphi^*_{\text{эмп}} > \varphi^*_{\text{кр}}$, то основная гипотеза H_0 отвергается и принимается гипотеза H_1; в противном случае принимается гипотеза H_0.

Однако, как следует из формулы (2), при больших объемах выборок n_1 и n_2 происходит значительное увеличение значения φ^*, поэтому необходимо провести нормирование данных.

Нормирование значений количества нулей и единиц выполняется по формуле (3) [24]:

$$X = \frac{x_i D}{x_{\max}}, \qquad (3)$$

где $x_{\max}$ — максимальное значение частот нулей или единиц в выборке;

x_i — частоты нуля и единицы соответственно (при i = 0 — частота нулей, при i = 1 — частота единиц);

D — константа нормирования. Опытным путем было принято D = 1500. Очевидно, что нормированные значения X будут принадлежать интервалу [0, 1500].

Заметим, что в результате процедуры нормирования сохраняются процентные соотношения нулей и единиц в каждой выборке, но значительно уменьшается величина подкорневого выражения в формуле (2), однако это не влияет на величину углов φ_1 и φ_2.

Таким образом, фильтрацию необходимо производить в двух случаях: при создании нового архива и при обновлении существующего.

Непосредственная идентификация файла. На этапе непосредственной идентификации файла производится сравнение побайтовых частотных характеристик сигнатур программ, хранящихся в архиве, и сигнатур файлов, полученных в результате фильтрации.

Рассмотрим процедуру непосредственной идентификации файла. Сравнение сигнатур происходит по следующему принципу: если частота *j*-го байта из распределения идентифицируемого файла отличается от частоты этого же байта в архиве сигнатур менее чем на 20 %, то значение переменной x_1 — «различие лежит в пределах 20 %», т. е. увеличивается на единицу. В противном случае на единицу увеличивается значение переменной x_2, значит «различие не лежит в пределах 20 %». Значение уровня различия в 20 % определено опытным путем.

Далее через точки с координатами $(x_1,0)$ и $(x_2,0)$ проводится прямая и вычисляется ее угловой коэффициент k [25]. При значении $k \geq 0$ файл считается идентифицированным, в противном случае — не идентифицированным.

Таким образом, алгоритм идентификации файла заключается в следующем.

1. На вход подается идентифицируемый исполняемый файл формата elf.

2. Для файла подсчитываются частоты нулей и единиц f.

3. Файл разбивается на байты и строится побайтовая частотная характеристика L. Значения L и f формируют с 3-й по 258-ю ячейки сигнатуры идентифицируемого файла.

4. Значения $\overline{v}_i$ и f нормируются.

5. Выполняется первый этап — *фильтрация по критерию Фишера* (φ^*-*критерию*). При этом сравниваются только первая и вторая ячейки сигнатуры идентифицируемого файла со второй и третьей, соответственно, ячейками сигнатуры программы из архива.

6. Выполняется второй этап — *непосредственная идентификация.* При этом сравниваются частотные характеристики $\overline{L}_i$ и L с использованием ячеек с 3-й по 259-ю для сигнатуры программы из архива и использованием ячеек со 2-й по 258-ю в сигнатуре идентифицируемого файла.

7. В результате принимается решение о том, какая сигнатура из архива, в меньшей степени, отлична от сигнатуры идентифицируемого файла.

Постановка эксперимента по применению метода

Приведем результаты эксперимента, проведенного на основе алгоритма, описанного ранее.

Для эксперимента были использованы следующие разновидности Linux ОС (разрядность 64):

1. Debian
2. Mint
3. Korora

Для эксперимента были использованы следующие программы (разрядность 64):

1. aircrack-ng
2. gftp-gtk
3. gimp
4. htop
5. nmap

Архив сигнатур формировался на основе указанных программ в ОС Debian и Mint.

Тестовая выборка была сформирована из указанных программ на ОС Korora. Задача состояла в идентификации тестовой выборки с архивом сигнатур.

В табл. 1 приведены результаты фильтрации при уровне значимости $p < 0{,}01$.

Таблица 1

Фильтрация

Идентифицируемые файлы, версия	Программы из архива сигнатур	$\varphi^*_{эмп}$	Результат
Сравнение одинаковых программ			
aircrack-ng 1.1-8	$\varphi^*_{эмп}$	0,15	+
gftp-gtk 2.0.19-10	gftp-gtk	0,025	+
gimp 2.8	gimp	0,64	+
gimp 2.8.8-3	gimp	0,64	+
htop 1.0.2-3	htop	0,8	+
nmap 6.40-2	nmap	0,017	+
Сравнение различных программ			
aircrack-ng 1.1-8	htop	3,51	-
gftp-gtk 2.0.19-10	gimp	0,7	+
gftp-gtk 2.0.19-10	aircrack-ng	0,64	+
gimp 2.8	htop	2,66	-
gimp 2.8	gftp-gtk	1,31	+
gimp 2.8.8-3	nmap	3,73	-
htop 1.0.2-3	nmap	1,86	+
nmap 6.40-2	aircrack-ng	4,4	-

Примечания. 1. В первом столбце табл. 1 представлены файлы, установленные на операционной системе Korora. 2. В столбце «Результат» символ «+» означает, что гипотеза H_0 принимается, символ «–» означает, что гипотеза H_0 отвергается.

Из табл. 1 следует, что при сравнении сигнатур заведомо одинаковых программ отсеивания не происходит, при этом при сравнении сигнатур заведомо различных программ не все программы, отличные от программ в архиве, были отфильтрованы.

В табл. 2 приведены результаты непосредственной идентификации.

Таблица 2

Проверка частотных распределений

Идентифицируемые файлы, версия	Программы из архива сигнатур	Угловой коэффициент	Результат
Сравнение одинаковых программ			
aircrack-ng 1.1-8	aircrack-ng	8,6	+
gftp-gtk 2.0.19-10	gftp-gtk	2,6	+
gimp 2.8	gimp	6,4	+
gimp 2.8.8-3	gimp	8,4	+
htop 1.0.2-3	htop	3,6	+
nmap 6.40-2	nmap	24,8	+
Сравнение различных программ			
gftp-gtk 2.0.19-10	gimp	–25,6	-
gftp-gtk 2.0.19-10	aircrack-ng	–25,6	-
gimp 2.8	gftp-gtk	–22,2	-
htop 1.0.2-3	nmap	–25,6	-

Примечания: 1. В первом столбце табл. 2 представлены файлы, установленные на операционной системе Korora. 2. В столбце «Результат» символ «+» означает, что файл идентифицирован как программа из архива сигнатур (второй столбец), символ «-» означает, что файл не идентифицирован.

Из табл. 2 следует, что при сравнении сигнатур заведомо одинаковых программ файлы идентифицировались верно, как совпадающие с программой в архиве. Аналогично, при сравнении сигнатур различных программ все файлы идентифицированы верно, как несовпадающие с программой в архиве.

Заключение

При разработке метода построения сигнатур рассматривается elf-формат файлов и закономерности в их бинарном коде. Для построения сигнатуры файлов и идентификации файлов применяются математические методы обработки данных, такие как статистический критерий Фишера (φ^*-критерий), побайтовая частотная характеристика файлов и угловой коэффициент.

На рис. 1 представлено графическое отображение результатов фильтрации, приведенных в табл. 1. В полуплоскости, соответствующей значению коэффициента Фишера $\varphi^*_{эмп} > 2{,}31$ ($p < 0{,}01$), находятся программы, сигнатуры которых отфильтровались; напротив, в полуплоскости $\varphi^*_{эмп} < 2{,}31$ находятся программы, перешедшие на этап непосредственной идентификации.

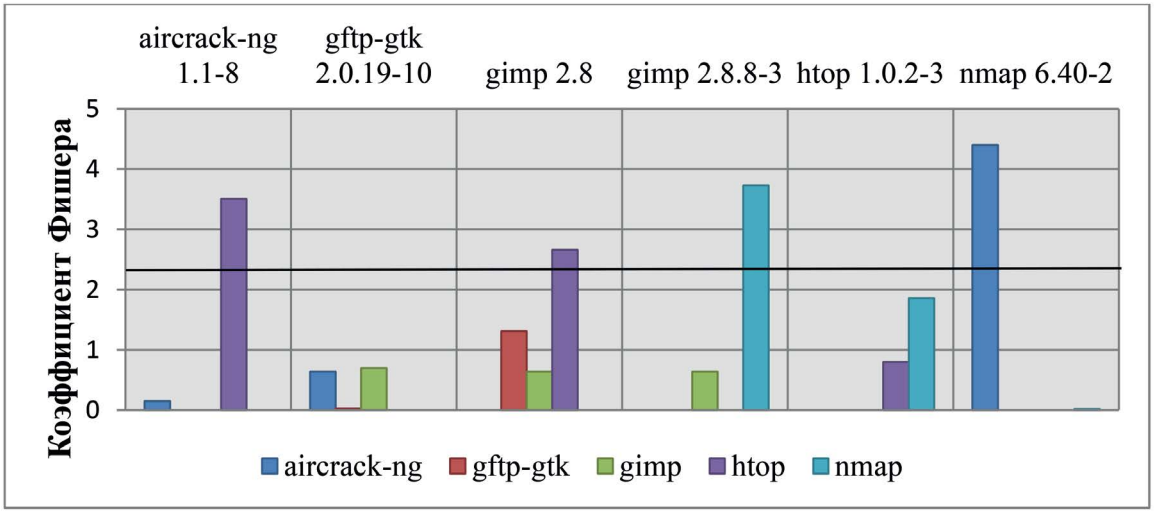

Рис. 1. Значения коэффициента Фишера при сравнении сигнатур

На рис. 2 представлено графическое отображение результатов идентификации сигнатур программ, приведенных в табл. 2. В полуплоскости $k \geq 0$ находятся программы, сигнатуры которых идентифицированы с сигнатурами программ в архиве.

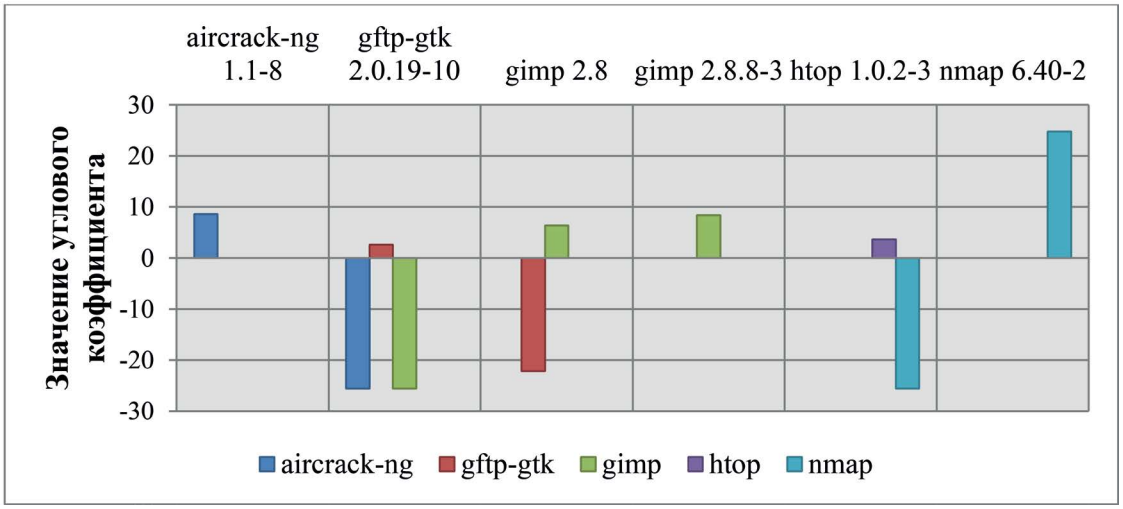

Рис. 2. Значения угловых коэффициентов при сравнении сигнатур

К достоинствам рассматриваемого метода можно отнести его способность идентифицировать файлы программ независимо от их версий и операционной системы Linux, на которой они установлены, простоту реализации и скорость выполнения задачи. Однако требуется увеличение архива сигнатур, усовершенствование самих сигнатур, включение дополнительных функций (например, такой, как способность к определению версии файла), а также анализ применения метода на больших объемах данных.

Тем не менее, результаты эксперимента подтверждают возможность применения метода для анализа содержимого компьютерных систем с целью выявления несанкционированно установленных программных продуктов в информационных системах на морском и речном транспорте, а также идентификации пользователя на основе данных, хранящихся на некотором носителе. Таким образом, представленный метод может быть использован в контрольных мероприятиях как техническое средство защиты, призванное устранить недостатки организационных мер, например, осуществлять контроль над исполнением установленной политики безопасности, а также при расследовании инцидентов ИБ.

СПИСОК ЛИТЕРАТУРЫ

1. *Krsul I.* Authorship analysis: identifying the author of a program / I. Krsul, E. H. Spafford // Computers & Security. — 1997. — Vol. 16. — № 3. — Pp. 233–257. DOI:10.1016/S0167-4048(97)00005-9.

2. *Rosenblum N.* Who wrote this code? identifying the authors of program binaries / N. Rosenblum, X Zhu, B. P. Miller // Computer Security–ESORICS 2011. — Springer Berlin Heidelberg, 2011. — Pp. 172–189.

3. *Bai J.* Malware detection through mining symbol table of Linux executables / J. Bai, Y. Yang, S. Mu, Y. Ma // Information Technology Journal. — 2013. — Vol. 12. — Is. 2. — Pp. 380–384. DOI: 10.3923/itj.2013.380.384.

4. *McKemmish R.* What is Forensic Computing? / R. McKemmish // Trends and Issues in Crime and Criminal Justice. — 1999. — No. 118. — Pp. 1–6.

5. Belkasoft Evidence Center. [Электронный ресурс]. — Режим доступа: http://ru.belkasoft.com/ru/ (дата обращения — 08.05.2015).

6. EnCase Forensic Guidance Software. [Электронный ресурс]. — Режим доступа: https://www.guidancesoftware.com (дата обращения — 08.05.2015).

7. Forensic Toolkit (FTK) AccessData. [Электронный ресурс]. — Режим доступа: http://accessdata.com (дата обращения — 08.05.2015).

8. *Brian C.* The sleuth kit and autopsy: Forensics tools for Linux and other Unixes. [Электронный ресурс]. — Режим доступа: http://www.sleuthkit.org/index.php (дата обращения — 08.05.2015).

9. *Gloe T.* Forensic analysis of video file formats / T. Gloe, A. Fischer, M. Kirchner // Digital Investigation. — 2014. — Vol. 11. — Pp. S68–S76. DOI: 10.1016/j.diin.2014.03.009.

10. *Wang W.* Exposing digital forgeries in video by detecting double MPEG compression / W. Wang, H. Farid // Proceedings of the 8th workshop on Multimedia and security. — ACM, 2006. — Pp. 37–47. DOI: 10.1145/1161366.1161375.

11. *Wang W.* Exposing digital forgeries in interlaced and deinterlaced video / W. Wang, H. Farid // IEEE Trans Inf Forensics Security. — 2007. — Vol. 2 — № 3. — Pp. 438–449. DOI:10.1109/TIFS.2007.902661.

12. *Kee E.* Digital image authentication from thumbnails / E. Kee, H. Farid // SPIE Symposium on Electronic Imaging. — San Jose, CA, 2010.

13. *Kee E.* Digital image authentication from JPEG headers/ E. Kee, M. K. Johnson, H. Farid // IEEE Trans Inf Forensics Security. — 2011. — Vol. 6 — № 3. — Pp. 1066–1075 DOI: 10.1109/TIFS.2011.2128309.

14. *Popescu A. C.* Statistical tools for digital forensics / Popescu A., Farid H. // Information Hiding. — Springer Berlin Heidelberg, 2005. — Pp. 128–147. DOI: 10.1007/978-3-540-30114-1_10.

15. *Хихин Р.* Установка программ в Linux. [Электронный ресурс]. — Режим доступа: http://www.opennet.ru/docs/RUS/linux_beg_faq/Linux-FAQ-7.html (дата обращения — 9.04.2015).

16. *Shved V.* The method of an audit of software containing in digital drives / V. Shved, P. Kuzmich, V. Korzhuk // Application of Information and Communication Technologies (AICT), 2014 IEEE 8th International Conference on. — IEEE, 2014. — Pp. 1–5. DOI: 10.1109/ICAICT.2014.7035927.

17. *Myles G.* K-gram based software birthmarks / G. Myles, C. Collberg // Proceedings of the 2005 ACM symposium on Applied computing. — ACM, 2005. — Pp. 314–318. DOI:10.1145/1066677.1066753.

18. *Khoo W. M.* Rendezvous: A search engine for binary code / W. M. Khoo, A. Mycroft, R. Anderson // Proceedings of the 10th Working Conference on Mining Software Repositories. — IEEE Press, 2013. — Pp. 329–338. DOI: 10.1109/MSR.2013.6624046.

19. *Moody S. J.* Sádi-statistical analysis for data type identification / S. J. Moody, R. F. Erbacher // Systematic Approaches to Digital Forensic Engineering, 2008. SADFE'08. Third International Workshop on. — IEEE, 2008. — Pp. 41–54. DOI: 10.1109/SADFE.2008.13.

20. *Haggerty J.* FORSIGS: Forensic Signature Analysis of the Hard Drive for Multimedia File Fingerprints / J. Haggerty, M. Taylor // IFIP TC11 International Information Security Conference. — Boston: Springer, 2006. DOI: 10.1007/978-0-387-72367-9_1.

21. Executable and linkable format. 2014.[Электронный ресурс] — Режим доступа: http://wiki.osdev.org/ELF (дата обращения — 9.04.2015).

22. *Сидоренко Е. В.* Методы математической обработки в психологии / Е. В. Сидоренко. — СПб.: Речь, 2010. — 350 с.

23. *Наследов А. Д.* Математические методы психологического исследования. Анализ и интерпретация данных: учеб. пособие; 3-е изд. / А. Д. Наследов. — СПб.: Речь, 2008. — 392 с.

24. *Брылевская Л. И.* Элементы теории линейных пространств: учеб. пособие / Л. И. Брылевская, И. А Лапин, Л. С. Ратафьева, О. Л. Суслина. — СПб.: СПбГИТМО (ТУ), 2001. — 140 с.

25. *Clapham C.* Oxford Concise Dictionary of Mathematics, Gradient / C. Clapham, J. Nicholson. — Addison-Wesley, 2009. — 876 p.

METHOD OF EXECUTABLE FILTS IDENTIFICATION BY THEIR SIGNATURES

The paper deals with the method of protecting software automated information systems at river and sea transport by identification executable elf-files installed on various Linux distributions by their specific structural signatures. Identifying here should be understood as the process of file recognition by establishing its coincidence with a particular program. The variety of Linux distributions and theirs open source codes make it difficult to use standard methods of executable files analysis and inventory. A new approach to solving this problem is given in the article.

The objects of study in this paper are elf-files and their signatures. Subject of research is methods of signature files comparing. The main aim is to develop a method of elf-files identification by their signatures analyses.

The proposed method consists of three main stages:

– acquisition of signatures' library;

– signatures' filtering;

– file identification.

Signatures, both included in the programs' archive and identificated files, has a special structure, which is based on the frequency characteristics.

Signatures' filtering is performed using Fisher's ratio test at a significance level of p=0,01.

For more accurate identification, an analysis of the signatures frequency characteristics, which are stored in the library, and the signatures, which are obtained by filtration, is held by the value of the slope at the level less than 20 % different.

Keywords: information security; executable elf-file; signature file; Fisher's ratio test; files identification.

REFERENCES

1. Krsul, Ivan, and Eugene H. Spafford. "Authorship analysis: Identifying the author of a program." *Computers & Security* 16.3 (1997): 233–257. DOI:10.1016/S0167-4048(97)00005-9.

2. Rosenblum, Nathan, Xiaojin Zhu, and Barton P. Miller. "Who wrote this code? identifying the authors of program binaries." *Computer Security–ESORICS 2011*. Springer Berlin Heidelberg, 2011: 172–189.

3. Bai, J., Y. Yang, S. Mu, and Y. Ma. "Malware detection through mining symbol table of Linux executables." *Information Technology Journal* 12.2 (2013): 380–384. DOI: 10.3923/itj.2013.380.384.

4. McKemmish, R. "What is Forensic Computing?" *Trends and Issues in Crime and Criminal Justice* 118 (1999): 1–6.

5. Belkasoft Evidence Center. Web. 8 May 2015 <http://ru.belkasoft.com/ru/>.

6. EnCase Forensic Guidance Software. Web. 8 May 2015 <https://www.guidancesoftware.com>.

7. Forensic Toolkit (FTK) AccessData. Web. 8 May 2015 <http://accessdata.com>.

8. Brian, C. The sleuth kit and autopsy: Forensics tools for Linux and other Unixes. Web. 8 May 2015 <http://www.sleuthkit.org/index.php>.

9. Gloe, T., A. Fischera, and M. Kirchner. "Forensic analysis of video file formats." *Digital Investigation* 11 (2014): 68–76. DOI: 10.1016/j.diin.2014.03.009.

10. Wang, W., and H. Farid. Exposing digital forgeries in video by detecting double MPEG compression. *Proceedings of the 8th workshop on Multimedia and security*. NY, USA: ACM, 2006. DOI: 10.1145/1161366.1161375.

11. Wang, W., and H. Farid. "Exposing digital forgeries in interlaced and deinterlaced video." *IEEE Trans Inf Forensics Security* (2007): 438–449. DOI:10.1109/TIFS.2007.902661

12. Kee, E., and H. Farid. "Digital Image Authentication from Thumbnails." *Proceedings of SPIE - The International Society for Optical Engineering* (2010).

13. Kee, E., M. K. Johnson, and H. Farid. "Digital Image Authentication from JPEG Headers." *IEEE Transactions on Information Forensics and Security* 6.3 PART 2 (2011): 1066–1075 DOI: 10.1109/TIFS.2011.2128309.

14. Popescu, Alin C., and Hany Farid. "Statistical tools for digital forensics." *Information Hiding*. Springer Berlin Heidelberg, 2005. DOI 10.1007/978-3-540-30114-1_10.

15. Khikhin R. Ustanovka programm v Linux. Web. April 9, 2015 http://www.opennet.ru/docs/RUS/linux_beg_faq/Linux-FAQ-7.html.

16. Shved, Viktor, Pavel Kuzmich, and Viktoria Korzhuk. "The method of an audit of software containing in digital drives." *Application of Information and Communication Technologies (AICT), 2014 IEEE 8th International Conference on*. IEEE, 2014: 1–5. DOI: 10.1109/ICAICT.2014.7035927.

17. Myles, Ginger, and Christian Collberg. "K-gram based software birthmarks." *Proceedings of the 2005 ACM symposium on Applied computing*. ACM, 2005. DOI:10.1145/1066677.1066753.

18. Khoo, Wei Ming, Alan Mycroft, and Ross Anderson. "Rendezvous: A search engine for binary code." *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press, 2013. DOI: 10.1109/MSR.2013.6624046.

19. Moody, Sarah J., and Robert F. Erbacher. "Sádi-statistical analysis for data type identification." *Systematic Approaches to Digital Forensic Engineering, 2008. SADFE'08. Third International Workshop on*. IEEE, 2008. DOI: 10.1109/SADFE.2008.13.

20. Haggerty, John, and Mark Taylor. "FORSIGS: Forensic Signature Analysis of the Hard Drive for Multimedia File Fingerprints." *IFIP TC11 International Information Security Conference*. 2006. DOI: 10.1007/978-0-387-72367-9_1.

21. Executable and linkable format. Web. 9 April 2015. <http://wiki.osdev.org/ELF>.

22. Sidorenko, E. V. *Metody matematicheskoy obrabotki v psikhologii*. SPb.: Rech Publ., 2010.

23. Nasledov, A. D. *Matematicheskie metody psikhologicheskogo issledovaniya. Analiz i interpretatsiya dannykh. Uchebnoe posobie*. SPb.: Rech Publ., 2008.

24. Brylevskaya, L. I., I. A. Lapin, L. S. Rataf'eva, and O. L. Suslina. *Elementy teorii lineynykh prostranstv. Uchebnoe posobie* . SPb.: ITMO, 2001.

25. Clapham, C., and J. Nicholson. "Oxford Concise Dictionary of Mathematics, Gradient." Addison-Wesley, 2009.

ИНФОРМАЦИЯ ОБ АВТОРАХ

Кривцова Ирина Евгеньевна —
старший преподаватель.
Санкт-Петербургский университет информационных технологий, механики и оптики
ikr@cit.ifmo.ru
Салахутдинова Ксения Иркиновна —
соискатель.
Санкт-Петербургский университет информационных технологий, механики и оптики
kainagr@mail.ru
Юрин Игорь Валентинович —
кандидат военных наук, доцент.
Государственный университет морского и речного флота имени адмирала С.О. Макарова.
9402015@mail.ru,kaf_koib@gumrf.ru

INFORMATION ABOUT THE AUTHORS

Krivtsova Irina Evgen'evna —
Senior Lecturer.
Saint Petersburg University of Information Technologies, Mechanics and Optics
ikr@cit.ifmo.ru
Salakhutdinova Kseniya Irkinovna —
applicant.
Saint Petersburg University of Information Technologies, Mechanics and Optics
kainagr@mail.ru
Yurin Igor Valentinovich —
PhD, associate professor.
Admiral Makarov State University of Maritime and Inland Shipping.
9402015@mail.ru, kaf_koib@gumrf.ru