Published in the Russian Federation
Vestnik policii
Has been issued since 1907.
ISSN: 2409-3610
Vol. 5, Is. 3, pp. 97-105, 2015

DOI: 10.13187/vesp.2015.5.97
www.ejournal21.com

UDC 004.02; 311.2

Method of Construction the Signatures of Executable Files for Identification Purposes

[1] Irina E. Krivtsova
[2] Kseniya I. Salakhutdinova
[3] Pavel A. Kuz'mich

[1] National research university of information technologies, mechanics and optics, Russian Federation
197101 Saint Petersburg, Kronverkskiyprospekt, 49
Senior lecturer
E-mail: ikr@cit.ifmo.ru
[2] National research university of information technologies, mechanics and optics, Russian Federation
197101 Saint Petersburg, Kronverkskiyprospekt, 49
Graduate Student
E-mail: kainagr@mail.ru
[3] National research university of information technologies, mechanics and optics, Russian Federation
197101 Saint Petersburg, Kronverkskiyprospekt, 49
Deputy Head of Department for Development
E-mail: kpa@cit.ifmo.ru

Abstract

The article deals with the creation of individual signatures of different versions of executable elf-files installed on various Linux distributions in order to identify them. Identifying here should be understood as the process of file recognition with its identification with a particular program. A new method of signature's creation of program is based on the frequency characteristics of the files identified with it, and the average frequency characteristics of the different versions of the program. Files identification is made on the basis of their binary codes using F-test. The *construction* of a *frequency distribution* is made for a more precise process of files identification. The article deals with 64-bit operating systems and programs.

Keywords: forensic analysis, executable elf-file; file signature; file frequency characteristics; file identification.

Введение

Одной из важнейших задач правоохранительных органов в современном обществе является борьба с киберпреступностью, а также предотвращение преступлений в сфере информационных технологий. Это новое направление работы ставит целый ряд проблем, одной из которых является аудит носителей информации на предмет выявления

несанкционированно установленных объектов (в данной статье исполняемые файлы формата elf) [1].

Существует большое количество методов анализа видео-файлов [2,3] и фотографий [4,5], на основе которых разрабатываются методы исследования исполняемых файлов. Исполняемые файлы могут являться уязвимостью ИБ (дефекты в программе, не декларированные возможности, нелегальное использование интеллектуальной собственности [6], включение в программы вредоносного кода [7] и т.п.) и привести к увеличению рисков ИБ.

Под идентификацией в данной работе следует понимать процесс распознавания некоторого файла как отождествление его с той или иной программой.

Множество программно-аппаратных продуктов [8,9,10] криминалистического анализа, обладающие функцией аудита, ориентированы, в первую очередь, на работу с Windows ОС, в то время как все более широкое распространение получают Linux системы.

Главное отличие исполняемых файлов, используемых в Windows системах, то, что они представлены пользователю уже в конечном (скомпилированном) виде. В Linux же системах, конечный вид установленной программы зависит не только от дистрибутива Linux, характеристик применяемого к ней компилятора, но также и от способа установки данной программы в системе [11].

Принимая во внимание многообразие ОС Linux и открытость их исходного кода, необходимо учитывать частоту выхода обновлений как самих ОС, так и программ для них.

Использование стандартных методов анализа и инвентаризации исполняемых файлов, в силу приведенных выше особенностей ОС Linux затруднительно, к таким методам относятся:

- ручной осмотр характерных мест установки программ;
- побайтовое сравнение;
- сравнение контрольной суммы;
- сравнение цифровой подписи;
- методы, представленные в других статьях [12, 13,14,15].

Объектом исследования в настоящей статье являются сигнатуры исполняемых файлов формата elf. Предмет исследования – индивидуальные характеристики бинарного кода файла. Цель исследования – разработка метода построения сигнатур elf-файлов для их идентификации на основе анализа сигнатур.

Предлагаемый метод включает три этапа:

- построение архива сигнатур;
- фильтрация сигнатур;
- непосредственная идентификация файла.

Предыстория вопроса. Построение архива сигнатур

Чтобы построить сигнатуру для 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, то есть, 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;

$v_i = \{f_1, f_2, ..., f_n\}$, где f_j – различные версии *i-ой* программы;

n – количество файлов в выборке.

Для построения сигнатуры *i-ой* 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 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:

$$\overline{v}_i = \frac{1}{n}\sum_{j=0}^{n} f_j\{z, o\} .(1)$$

Далее необходимо учитывать более индивидуальные характеристики 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-ой* программы. Таким образом, получаем строку с 256-ю значениями:

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

где b_j – частота *j-го* байта.

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

1. имя программы – 1 ячейка;
2. среднее количество нулей и единиц – 2 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 длину в 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* - частоты нулей и единиц бинарного кода файла;

L – побайтовая частотная характеристика.

Обсуждение. Составление алгоритма идентификации файла

Перейдем к процессу идентификации файла, который, как было указано выше, состоит из двух этапов.

Первый представляет собой отсеивание сигнатур файлов, значительно отличающихся от сигнатур программ, помещенных в архив, то есть фильтрацию сигнатур, и осуществляется с использованием многофункционального статистического критерия $\varphi*$-Фишера. Поскольку критерий Фишера основывается на альтернативной шкале «есть признак – нет признака» [16], то 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 $\varphi*$-Фишера формируем 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[17]. Эмпирическая величина угла $\varphi*$ вычисляется по формуле:

$$\varphi*_{эмп.} = (\varphi_1 - \varphi_2)\sqrt{\frac{n_1 * n_2}{n_1 + n_2}}, \quad (2)$$

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

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

Если при сравнении полученного значения $\varphi*_{эмп.}$ с критическим значением $\varphi*_{кр}$ выполняется неравенство $\varphi*_{эмп.} > \varphi*_{кр}$, 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 $\varphi*$, поэтому необходимо провести нормирование данных.

Нормирование значений количества нулей и единиц происходит по формуле (3)[18]:

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

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 величины углов φ_1 и φ_2.

Таким образом, первый этап идентификации файла – фильтрацию необходимо производить в двух случаях: при создании нового архива и при обновлении существующего.

На втором этапе 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 распределения этого же байта в архиве сигнатур менее чем на 20 % , то значение переменной x_1 - «лежит в пределах 20%» увеличивается на единицу. В противном случае на единицу увеличивается значение переменной x_2 - «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 [19]. 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 по 258 ячейки сигнатуры идентифицируемого файла;
4. значения $\overline{v}_i$ и f нормируются;
5. производится первый этап – фильтрация по критерию $\varphi*$-Фишера при этом сравниваются только первая и вторая ячейки сигнатуры идентифицируемого файла со второй и третьей соответственно ячейками сигнатуры программы из архива;
6. производится второй этап – непосредственная идентификация, при этом сравниваются частотные характеристики $\overline{L}_i$ и L, используются ячейки с 3 по 259 для сигнатуры программы из архива и используются ячейки с 2 по 258 в сигнатуре идентифицируемого файла;

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

Результаты. Описание эксперимента

Рассмотрим результаты эксперимента, проведенного на основе алгоритма, описанного выше.

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

1. Debian
2. Mint
3. Korora

И программы (разрядность 64):

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

Архив сигнатур формировался на основе указанных выше 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 таблице 1 представлены результаты фильтрации при уровне значимости $p<0,01$.

Таблица 1

Фильтрация

Сравнение одинаковых программ

Идентифицируемые файлы, версия	Программы из архива сигнатур	$\varphi^*_{эмп.}$	Результат
aircrack-ng 1.1-8	aircrack-ng	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

Идентифицируемые файлы, версия	Программы из архива сигнатур	$\varphi^*_{эмп.}$	Результат
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	-

В первом столбце таблицы 1 представлены файлы, установленные на ОС Korora. В столбце «результат» символ «+» означает, что гипотеза H_o принимается, символ «-» означает, что гипотеза H_o отвергается. Из таблицы 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 таблице 2 представлены результаты непосредственной идентификации.

Таблица 2

Проверка частотных распределений

Сравнение одинаковых программ			
Идентифицируемые файлы, версия	Программы из архива сигнатур	Угловой коэффициент	Результат
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			
Идентифицируемые файлы, версия	Программы из архива сигнатур	Угловой коэффициент	Результат
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	-

В первом столбце таблицы 2 представлены файлы, установленные на ОС Korora. В столбце «результат» символ «+» означает, что файл идентифицирован как программа из архива сигнатур (второй столбец), символ «-» означает, что файл не идентифицирован. Из таблицы 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 файла и идентификации файлов применяются математические методы обработки данных, такие как статистический критерий Фишера и угловой коэффициент φ^*.

Эксперимент показал, что на этапе фильтрации (таблица 1), значениям коэффициента Фишера $\varphi^*_{эмп} > 2{,}31$ ($p < 0{,}01$) соответствуют программы, сигнатуры которых отфильтровались; напротив, значения $\varphi^*_{эмп.} < 2{,}31$ соответствуют программам, перешедшим на этап непосредственной идентификации.

При непосредственной идентификации сигнатур программ (таблица 2), значениям $k \geq 0$ соответствуют программы, сигнатуры которых идентифицировались с сигнатурами программ в архиве.

К достоинствам разработанного метода можно отнести его способность идентифицировать файлы программ независимо от их версий и ОС 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 апробации метода подтверждают возможность его применения для криминалистического анализа программных продуктов.

Примечания:

1. McKemmish R. What is Forensic Computing? // Trends and Issues in Crime and Criminal Justice. 1999, no. 118.

2. Gloea T., Fischera A., Kirchner M. Forensic analysis of video file formats // Digital Investigation. 2014. vol.11, Supplement 1, pp. 68–76. DOI: 10.1016/j.diin.2014.03.009

3. Wang W., Farid H. Exposing digital forgeries in video by detecting double MPEG compression // ACM Multimedia and Security Workshop. Geneva, Switzerland. 2006. DOI: 10.1145/1161366.1161375

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

5. Kee E., Johnson M.K., Farid H. Digital image authentication from JPEG headers // IEEE Trans Inf Forensics Security. 2011. pp. 1066–1075. DOI: 10.1109/TIFS.2011.2128309

6. Krsul I., Spafford E.H. Authorship analysis: identifying the author of a program //J ComputSecur, no.16. 1997. pp. 233–257. DOI: 10.1016/S0167-4048(97)00005-9

7. Malware detection through mining symbol table of Linux executables / Bai I., Yang Y., Mu S., Ma Y. // Knowledge and information systems, Springer. 2013. DOI: 10.3923/itj.2013.380.384

8. Belkasoft Evidence Center. 2015. [Электронный ресурс] URL: http://ru.belkasoft.com /ru/ (дата обращения 8.05.2015).

9. En Case Forensic Guidance Software. 2015 [Электронный ресурс] URL: https://www.guidance software.com (дата обращения 8.05.2015).

10. Forensic Toolkit (FTK) AccessData. 2015. [Электронный ресурс] URL: http://accessdata.com (дата обращения 8.05.2015).

11. Хихин Р. Установка программ в Linux [Электронный ресурс] – URL:http://www.opennet.ru/docs/RUS/linux_beg_faq/Linux-FAQ-7.html (последняя дата обращения 9.04.2015)

12. Korzhuk V.M., Kuzmich P.A., Shved V.G. The method of an audit of software containing in digital drives // AICT 2014 (8th IEEE International Conference). 2014. pp. 128-132.

13. Khoo W.M., Mycroft A., Anderson R. Rendezvous: a search engine for binary code // 10th Working Conference on Mining Software Repositories, IEEE Press. 2013. pp. 329–338. DOI: 10.1109/MSR.2013.6624046

14. Moody S., ErbacherSadi R. Statistical analysis for data type identification // Systematic Approaches to Digital Forensic Engineering, SADFE '08. 2008. pp. 41–54. DOI: 10.1109/SADFE.2008.13

15. Haggerty, J., Taylor, M.: FORSIGS; Forensic Signature Analysis of the Hard Drive for Multimedia File Fingerprints // IFIP TC11 International Information Security Conference. Sandton, South Africa. 2006. DOI: 10.1007/978-0-387-72367-9_1.

16. Сидоренко Е.В. Методы математической обработки в психологии. СПб.: Речь. 2010. С.350.

17. Наследов А.Д. Математические методы психологического исследования. Анализ и интерпретация данных. Учебное пособие. СПб.: Речь, 2008. 3-е изд. С. 392.

18. Элементы теории линейных пространств. Учебное пособие / Брылевская Л.И., Лапин И.А., Ратафьева Л.С., Суслина О.Л. СПб ИТМО (ТУ). Кафедра высшей математики, 2001. С. 141.

19. Clapham C.; Nicholson, J. (2009). Oxford Concise Dictionary of Mathematics, Gradient // Addison-Wesley. p. 348.

References:

1. McKemmish R. What is Forensic Computing? // Trends and Issues in Crime and Criminal Justice. 1999, no. 118.

2. Gloea T., Fischera A., Kirchner M. Forensic analysis of video file formats // Digital Investigation. 2014. vol.11, Supplement 1, pp. S68–S76. DOI: 10.1016/j.diin.2014.03.009
3. Wang W., Farid H. Exposing digital forgeries in video by detecting double MPEG compression // ACM Multimedia and Security Workshop. Geneva, Switzerland. 2006. DOI: 10.1145/1161366.1161375
4. Kee E., Farid H. Digital image authentication from thumbnails // SPIE Symposium on Electronic Imaging. San Jose, CA. 2010.
5. Kee E., Johnson M.K., Farid H. Digital image authentication from JPEG headers // IEEE Trans Inf Forensics Security. 2011. pp. 1066–1075. DOI: 10.1109/TIFS.2011.2128309
6. Krsul I., Spafford E.H. Authorship analysis: identifying the author of a program //J ComputSecur, no.16. 1997. pp. 233–257. DOI: 10.1016/S0167-4048(97)00005-9
7. Malware detection through mining symbol table of Linux executables / Bai I., Yang Y., Mu S., Ma Y. // Knowledge and information systems, Springer. 2013. DOI: 10.3923/itj.2013.380.384
8. Belkasoft Evidence Center. 2015. [Elektronnyi resurs]. URL: http://ru.belkasoft.com/ru/ (data obrashcheniya 8.05.2015)
9. EnCase Forensic Guidance Software. 2015 [Elektronnyi resurs]. URL: https://www.guidancesoftware.com (data obrashcheniya 8.05.2015)
10. Forensic Toolkit (FTK) AccessData. 2015. [Elektronnyi resurs]. URL: http://accessdata.com (data obrashcheniya 8.05.2015)
11. Khikhin R. Ustanovkaprogramm v Linux [Elektronnyiresurs]. URL: http://www.opennet.ru/docs/RUS/linux_beg_faq/Linux-FAQ-7.html (poslednyaya data obrashcheniya 9.04.2015)
12. Korzhuk V.M., Kuzmich P.A., Shved V.G. The method of an audit of software containing in digital drives // AICT 2014 (8th IEEE International Conference). 2014. pp. 128-132.
13. Khoo W.M., Mycroft A., Anderson R. Rendezvous: a search engine for binary code // 10th Working Conference on Mining Software Repositories, IEEE Press. 2013. pp. 329–338. DOI: 10.1109/MSR.2013.6624046
14. Moody S., ErbacherSadi R. Statistical analysis for data type identification // Systematic Approaches to Digital Forensic Engineering, SADFE '08. 2008. pp. 41–54. DOI: 10.1109/SADFE.2008.13
15. Haggerty, J., Taylor, M.: FORSIGS; Forensic Signature Analysis of the Hard Drive for Multimedia File Fingerprints // IFIP TC11 International Information Security Conference. Sandton, South Africa. 2006. DOI: 10.1007/978-0-387-72367-9_1.
16. Sidorenko E.V. Metody matematicheskoi obrabotki v psikhologii. SPb.: Rech'. 2010. S. 350.
17. Nasledov A.D. Matematicheskie metody psikhologicheskogo issledovaniya. Analiz i interpretatsiya dannykh. Uchebnoe posobie. SPb.: Rech', 2008. 3-e izd. S. 392.
18. Elementy teorii lineinykh prostranstv. Uchebnoe posobie / Brylevskaya L.I., Lapin I.A., Rataf'eva L.S., Suslina O.L. SPb ITMO (TU). Kafedra vysshei matematiki, 2001. S. 141.
19. Clapham C.; Nicholson, J. (2009). Oxford Concise Dictionary of Mathematics, Gradient // Addison-Wesley. p. 348.

УДК 004.02; 311.2

Метод построения сигнатур исполняемых файлов с целью их идентификации

[1] Ирина Евгеньевна Кривцова
[2] Ксения Иркиновна Салахутдинова
[3] Павел Алексеевич Кузьмич

[1] Национальный исследовательский университет информационных технологий, механики и оптики, Российская Федерация
197101 Санкт-Петербург, Кронверкский проспект, 49
Старший преподаватель

E-mail: ikr@cit.ifmo.ru
[2] Национальный исследовательский университет информационных технологий, механики и оптики, Российская Федерация
197101 Санкт-Петербург, Кронверкский проспект, 49
E-mail: kainagr@mail.ru
Магистр
[3] Национальный исследовательский университет информационных технологий, механики и оптики, Российская Федерация
197101 Санкт-Петербург, Кронверкский проспект, 49
Заместитель заведующего кафедрой по развитию, куратор
E-mail: kpa@cit.ifmo.ru

Аннотация. В статье рассмотрено построение индивидуальных сигнатур исполняемых elf-файлов различных версий и установленных на разных дистрибутивах Linux с целью их идентификации. Под идентификацией следует понимать процесс распознавания некоторого файла как отождествление его с той или иной программой. Предложен новый метод построения сигнатуры программы на основе частотных характеристик файлов, отождествленных с ней, и усредненной частотной характеристики различных версий программы. Идентификация файлов производится на основе анализа их бинарных кодов с применением углового преобразования Фишера. Построено частотное распределение для более точного процесса идентификации файлов. В работе рассмотрены 64-разрядные ОС и программы.

Ключевые слова: криминалистический анализ, исполняемый elf-файл, сигнатура файла, частотные характеристики файла, идентификация файлов.