DOI 10.37882/2223-2966.2024.01.30

РАЗРАБОТКА СИСТЕМЫ ДЛЯ ВЫЯВЛЕНИЯ ПОТЕНЦИАЛЬНЫХ ВИРУСОВ И АКТИВНЫХ ЭКСПЛОИТОВ В КОРПОРАТИВНЫХ МЕССЕНДЖЕРАХ

DEVELOPMENT OF A SYSTEM FOR DETECTING POTENTIAL VIRUSES AND ACTIVE EXPLOITS IN CORPORATE MESSENGERS

A. Rusakov
E. Amelyutin
S. Rozin
A. Samogin

Summary. The paper describes one of the possible approaches to identifying potential viruses and active exploits in corporate messengers. Timely detection of traces and descriptions of viruses and active exploits in corporate messengers helps to increase the overall level of security. The paper provides a structure and description of the development of a software package to identify potential viruses and exploits in corporate messengers. The software package is capable of working on big data in order to identify attacks that can be implemented in the form of Internet links, polymorphic malware and Trojan ransomware.in the form of Internet links, polymorphic malware and Trojan ransomware.

Keywords: text mining, information security, detection of active exploits, detection of traces and descriptions of viruses.

Русаков Алексей Михайлович
Старший преподаватель, МИРЭА — Российский технологический университет
rusal@bk.ru
Амелютин Евгений Вячеславович
Доцент, МИРЭА — Российский технологический университет
amelyutin9@yandex.ru
Розин Станислав Вадимович
МИРЭА — Российский технологический университет
stasrozin@gmail.com
Самогин Артем Сергеевич
Ассистент, МИРЭА — Российский технологический университет
samogin@mirea.ru

Аннотация. В работе приводится описание одного из возможных подходов к выявлению потенциальных вирусов и активных эксплоитов в корпоративных мессенджерах. Своевременное обнаружение следов и описаний вирусов и активных эксплойтов в корпоративных мессенджерах способствует повышению общего уровня безопасности. В работе приводится структура и описание разработки программного комплекса для выявления потенциальных вирусов и эксплойтов в корпоративных мессенджерах. Программный комплекс способен работать на больших данных с целью выявления атак, которые могут быть реализованы в виде Интернет-ссылок, полиморфных вредоносных программ (polymorphic malware) и троянских программ-шифровальщиков, занимающихся вымогательством.

Ключевые слова: интеллектуальный анализ текстов, информационная безопасность, обнаружение активных эксплоитов, обнаружение следов и описаний вирусов.

Введение

Современные организации и компании все более активно взаимодействуют с клиентами и пользователями через различные средства корпоративной коммуникации, включая электронную почту, чаты, социальные сети и другие платформы. Обеспечение безопасности и защита информации, поступающей через эти каналы, становятся важнейшими задачами [1, 2].

Объем данных, генерируемых пользователями, постоянно растет. Автоматизированный анализ и обнаружение активных угроз, эксплоитов и потенциальных вирусов в таких данных становятся неотъемлемой частью стратегий информационной безопасности.

Злоумышленники постоянно совершенствуют методы атак и маскировки. Анализ текстовых данных на предмет вирусов и активных эксплойтов требует постоянного обновления и разработки новых методов для действенной защиты.

Методы обработки естественного языка (NLP— Natural Language Processing») становятся все более мощными и доступными. Их применение для анализа текстовых данных с целью обеспечения информационной безопасности предоставляет новые возможности и перспективы [3].

Во многих странах ужесточаются законодательные требования, касающиеся обеспечения конфиденциальности и защиты данных. Работа в области обеспечения информационной безопасности и анализа текстовых данных может помочь организациям соблюдать эти нормы и требования.

Таким образом, разработка программного средства интеллектуального анализа корпоративных мессенджерах является не только актуальной, но и имеет широкий практический и теоретический интерес в контексте современных вызовов в области информационной безопасности и обработки текстовых данных.

Структура программного комплекса для выявления потенциальных вирусов и эксплойтов в корпоративных мессенджерах

Предлагается разработать облачный сервис для выявления атак на основе анализа больших данных, который включает три основных компонента: 1) Сервер, управляющий потоком входящих и исходящих данных об атаке. 2) Мультисканер с препроцессором для статического анализа. 3) Песочница (Sandbox), которая предназначена для автоматизированного запуска вредоносного кода с целью проведения динамического анализа.

Песочница — ядро системы анализа, предназначенное для динамического анализа образцов в изолированной среде.

Цель — запуск выявленных Multiscanner образцов и сбор информации об их активности.

Песочница построена на технологии VMWare для запуска нескольких экземпляров Windows 10. Каждый экземпляр включает в себя следующие модули анализа: Dumper — сохраняет дампы новых процессов в системе и дампы процессов в процессах. Для реализации сервиса была создана инфраструктура на основе гипервизора VMWare ESXi.

Сервер решает следующие задачи: загрузка входного потока в общую папку GlobalInput; поддержка базы данных образцов, загруженных в GlobalInput, и отфильтровка дубликатов; хранение образцов в GlobalOutput, загруженных после предварительной обработки; загрузка образцов из папок: GlobalOutput\win32\{dll, exe, drv, pe32}\{Suspicious, Unknown} into Input of Sandbox.

Разобрать Output Sandbox и обновить базу данных по обработанным образцам: сбор информации с Links.txt; сбор информации с File.txt; сбор MD5 отклонённых / загруженных образцов; сбор информации с wincheck.txt; запуск StringExtractor.exe и сохранение строки, извлеченной из дампа в хранилище; сохранение выражений, полученных после сканирования на VirusTotal в БД; сохранение сработанных правил Yara в БД; сохранение скриншотов, если они доступны для хранения.

Рассматриваемые образцы хранятся при необходимости.

Мультисканер решает следующие задачи:

1. Препроцесинг. Получение из папки GlobalInput на файловом сервере образцов, обработка их и размещение результата в локальном каталоге «Output».
2. Проверка локального «Output» KAV, создание папки «Detected» в одной папке и перемещение туда всех обнаруженных файлов, например: «Output\win32\exe\detected» или «Output\win32\dll\detected».
3. Перемещение в локальную папку OUTPUT в каталог файлового сервера GlobalOutput.
4. Сбор информации о сэмплах, обработка и запись в базу данных на Fileserver.

Разработка механизмов защиты

Определимся с понятиями, которые будем использовать при описании формальной модели: элемент e — значение атрибута; набор элементов q — кортеж длины q или q-кортеж; q-кортеж является распространенным или q-ассоциацией, если его появление превышает или равно предельному значению поддержки; q-ассоциация — частотный шаблон, который может иметь и другие формы; все атрибуты должны быть отличными друг от друга (неизоморфными)

Ассоциации (часто встречающиеся наборы) и обобщенные ассоциации (ассоциации в таблице с обобщениями на основе атрибутов (Attribute Oriented Generalization — AOG)) считаются часто используемыми понятиями для определения частотных шаблонов. Другими словами, ассоциация является конъюнкцией значений атрибутов, которые имеют высокую поддержку. Отметим, что некоторые дизъюнкции значений атрибутов дают новые атрибуты. Поэтому обобщенные ассоциации являются сочетанием этих новых атрибутов, которые имеют высокую поддержку [4, 5, 6, 7].

Метод основан на следующих алгоритмах анализа частотных шаблонов (Frequent Patterns — FP): FP-growth 8] и FP-trees [9]. FP-growth алгоритм используется для рекурсивного анализа и выявления условных зависимостей по частотным деревьям (FP-trees).

Частотное дерево является древовидной структурой данных, представляющей информацию о связи элементов в базе данных. FP-trees — специальный алгоритм построения таких деревьев. Оба алгоритма реализовывают идею ассоциативного анализа данных.

Сначала имеется база данных Интернет-ссылок и их атрибутов, полученная с помощью WhoIs и GeoIP онлайн-сервисов.

FP-growth алгоритм [8] является одним из самых быстрых и наиболее популярных алгоритмов для опреде-

ления часто встречающихся наборов элементов. Данный алгоритм работает с древовидным представлением базы данных транзакций (FP-tree). Данный формат представления данных позволяет сократить необходимый объем памяти для хранения транзакционных данных и подчеркнуть важность иерархических отношений между элементами наборов (атрибутами).

Основной идеей метода FP-growth считается рекурсивный поиск часто встречающихся шаблонов и разделение базы данных. Алгоритм состоит из следующих шагов: 1. Для каждого часто встречающегося элемента строится база импликативных шаблонов, а затем его частотное дерево; 2. Данный процесс повторяется для всех вновь создаваемых частотных деревьев.

До тех пор, пока конечное FP-дерево не будет пустым, либо содержать только один путь — по заданному пути генерируются все возможные комбинации входящих в него путей, каждый из которых является частотным паттерном.

Целью метода считается выявление частотных шаблонов для вредных, фишинг и чистых наборов Интернет-ссылок, которые в дальнейшем будут использованы для эвристического детектирования неизвестных Интернет-ссылок. FP-growth алгоритм, который используется в данном методе для генерации частотных шаблонов, обеспечивает отличную производительность, по сравнению с существующими аналогами, такими как: Apriori и TreeProjection [10], что сказывается на общей производительности метода, учитывая большой размер наборов вредных, фишинговых и вредоносных Интернет-ссылок. Для поиска всех частотных шаблонов необходимо всего два сканирования таблицы: первое сканирование необходимо для определения частоты каждого элемента множества, второе — для построения частотного дерева (FP-tree). Далее с помощью алгоритма FP-Growth рекурсивно сканируем дерево и получаем искомые частотные паттерны. Интернет-ссылка считается вредной тогда и только тогда, когда по ней загружаются, данные, которые фиксируются сканерами как вредные.

Интернет-ссылка считается фишинг-ссылкой тогда и только тогда, когда по ней загружаются страницы, имитирующие популярные интернет-сервисы. Такие ссылки публикуются и детектируются репутационной системой PhishTank. Интернет-ссылка считается чистой тогда и только тогда, когда по ней загружаются страницы и данные, которые не детектируются как вредоносные или фишинг. Чистые Интернет-ссылки были получены с репутацийного Интернет-сервиса Alexa.com [11], где предоставляется статистика посещений веб-страниц, на основании которой веб-сайт получает рейтинговую оценку. В выборке чистых ссылок использовались только веб-сайты, которые имеют высокие рейтинговые оценки, то есть с высоким количеством посещений.

Для выявления false negatives, пропущенных вредных и фишинговых ссылок использовались IDS Suricata [12], Google Safe Browsing [13], и Virustotal [14] сервисы.

Наборы Интернет-ссылок для выявления частотных шаблонов и построения частотных деревьев представлены в таблице 1.

Таблица 1.

Наборы Интернет-ссылок, которые используются для тестирования

Набор	Количество ссылок в наборе	Источник, откуда был загружен набор
Доброкачественные	9991	Alexa.com
Фишинг	98054	PhishTank.com
Вредоностные	16294	Lavasoft MAS

Для каждой Интернет-ссылки был получен такой набор атрибутов: страна, где находится сервер со страницей или прокси-сервер (по данным GeoIP); тип содержимого страницы (content-type); название регистратора домена, если используется доменное имя сервера, а не IP адрес; время жизни домена, если используется доменное имя сервера (Lifetime — LT).

Уровень поддержки был установлен 0.005, что означает отсеивание атрибутов, если их частота зарождения в наборе меньше 0.5 % от общего количества элементов в наборе.

Полиморфные шпионские программы становятся все более распространенными в данное время как метод, применяемый чтобы победить антивирусные сканеры. В этой главе будет рассматриваться, как полиморфная мутация помогает предотвратить выявление вредоносного программного обеспечения путем изучения недавно обнаруженного полиморфного червя NgrBot / DorkBot. Затем будет рассматриваться, как найти созданный полиморфный шпион.

Создатели вредоносных программ постоянно ищут новые технологии, чтобы оставаться на шаг впереди исследователей антивирусных программ, чтобы избежать обнаружения антивирусными программами. Метод, который будет рассматриваться здесь, является часто используемым трюком, который широко используется веб-эксплуатантами и известными ботнетами — полиморфизмом на стороне сервера. Примеры данного метода: Shiz, Carperb и Ngrbot / Dorkbot. Основной целью этих бэкдоров является кража учетных данных для интернет-банкинга, торговых платформ и RBS (удаленных банковских услуг). После выпуска, очень часто оказывается, что новая копия полиморфного шпионского программного обеспечения не обнаруживается большинством сканеров AV-файлов (рис. 1).

Рис. 1. Результат сканирования VirusTotal нового образца Ngrbot почти пустого (DrWeb: BackDoor.IRC.NgrBot.146, Fortinet: W32/EncPk.CWP!tr, TrendMicro-HouseCall: TROJ_GEN.RC9H1K6)

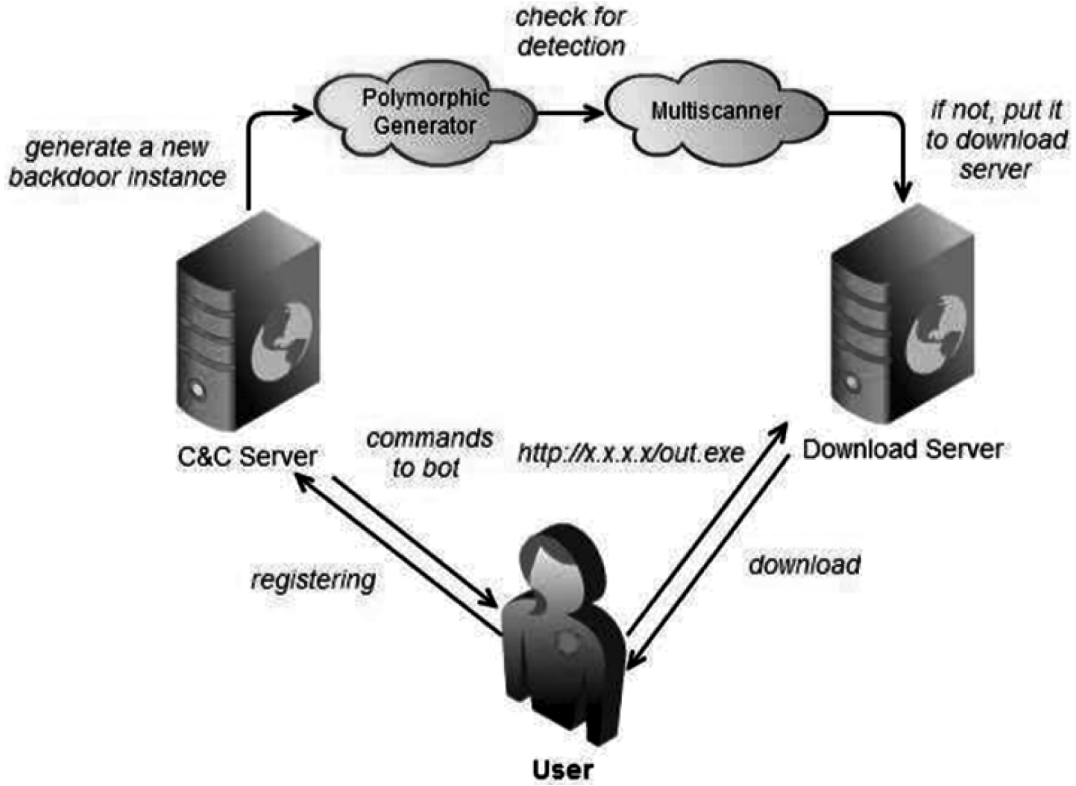

Рис. 2. Модель полиморфизма на стороне сервера

Таким образом, он делает обнаружение вредоносных программ, созданных с использованием полиморфизма на стороне сервера, более сложным для традиционного подхода на основе подписи.

Идея полиморфного шифрования не считается новой и заключается в повторном шифровании вредоносного файла на серверах злоумышленников каждый раз, когда его запрашивает инфицированная машина бота. Рассмотрим схему полиморфизма (рис. 2) [15].

После заражения компьютер пользователя отправляет регистрационную информацию на сервер C & C. Затем сервер C & C отвечает набором команд для выполнения на компьютере жертвы.

Новая часть вредоносного программного обеспечения генерируется «полиморфным генератором», который повторно пакует или повторно шифрует его с помощью случайно сгенерированного ключа. Этот метод гарантирует, что злоумышленное программное обеспечение является уникальным, придавая ему значительное преимущество — он никогда не будет обнаружен и проанализирован исследователями вредоносных программ. Это значительно увеличивает вероятность того, что она не будет обнаружена. Злоумышленник может воспользоваться популярным антивирусным программным обеспечением для сканирования вновь созданной копии, чтобы проверить, не происходит ли обнаружение. Несмотря на то, что файл может быть отсканированным с помощью онлайновых служб, таких как VirusTotal, авторы вредоносных программ, как правило, не используют этот маршрут, поскольку образец будет распределен среди AV сообщества, что приводит к анализу файла и добавлению в базы данных обнаружения. После того,

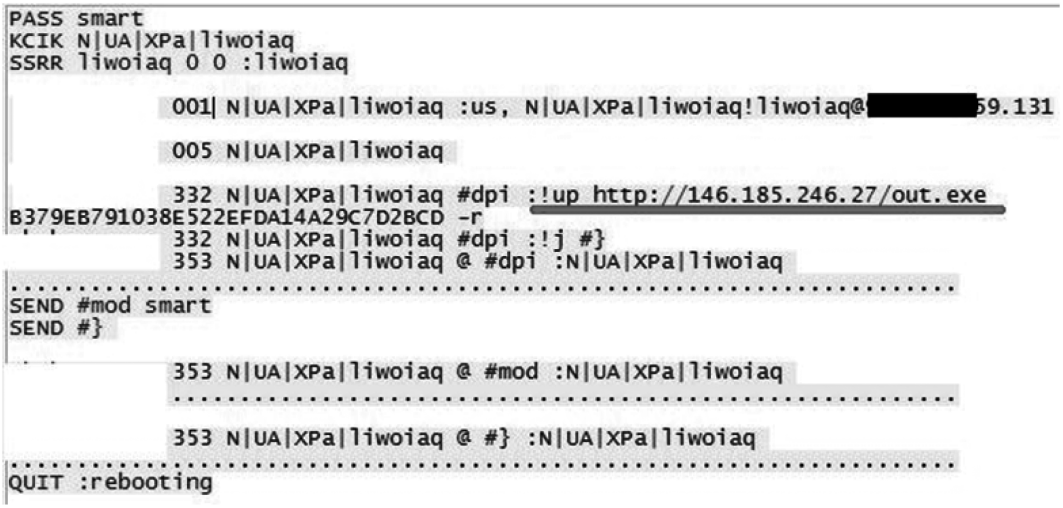

Рис. 3. Связь Ngrbot с ботом-сервером

как копия генерируется и проверяется как не обнаруженная, она хранится на сервере «Загрузки», а ссылка передается потерпевшему.

Тестирование системы защиты

Рассмотрим реальный пример. После установления Ngrbot [14] получает от C & C URL, чтобы обновить себя. После этого бот загружает новый экземпляр бэкдора. После «обновления» бэкдор становится невидимым для сканеров, созданных на основе подписей AV. Более того, такие бэкдоры часто блокируют доступ к веб-сайтам AV, останавливая приложение безопасности пользователя от загрузки новых обновлений базы данных обнаружения (рисунки 3 и 4).

Рис. 4. Обновление Ngrbot

Если провести сравнение двух полиморфных экземпляров одного и того же бэкдора, то можно увидеть следующее изображение — рисунок 5.

Видно, что код и размер файла совершенно разные. Эта разница может быть достигнута с помощью полиморфного мутатора. На рисунке показано, что структуру и размер кода можно изменять, добавляя нули и повторно шифрующие данные. Как следствие, заметны существенные различия в файловой структуре (рис. 6).

Однако, если запустить оба образца в песочнице и посмотреть на код, введенный в системные процессы, можно увидеть почти идентичные данные (рис. 7).

Несмотря на значительные различия в содержании файлов, оба образца имеют одинаковую функциональность и полезную нагрузку, что отражается во вредных инъекциях (рисунок 8). Если антивирусные сканеры были способны запускать образец в песочнице или эмуляторе во время сканирования, они не будут обмануты полиморфным шифрованием и немедленно поймают вновь созданные копии с точным семейным вердиктом.

Опишем правила Yara, какие помогут исследователям вредного программного обеспечения идентифицировать образцы вредоносного программного обеспечения Ngrbot / Dorkbot на инфицированной машине.

Чтобы найти уникальные строки, которые используются для идентификации инфекции, нужно сделать дамп кода Ngrbot. Дамп вводится в адресное пространство всех запущенных процессов, за исключением системы, smss.exe и lsass.exe.

Ниже приведен пример поиска инъекции, анализируя дескрипторы виртуальных адресов (VAD) Explorer.exe [16].

Кроме того, сброс злонамеренного кода, вложенного в процесс Explorer.exe, может быть осуществлен с помощью PETools.

Пример дампа вредоносного кода показан на рис. 10.

Дамп сканируется бесплатным онлайн-сканером VirusTotal.

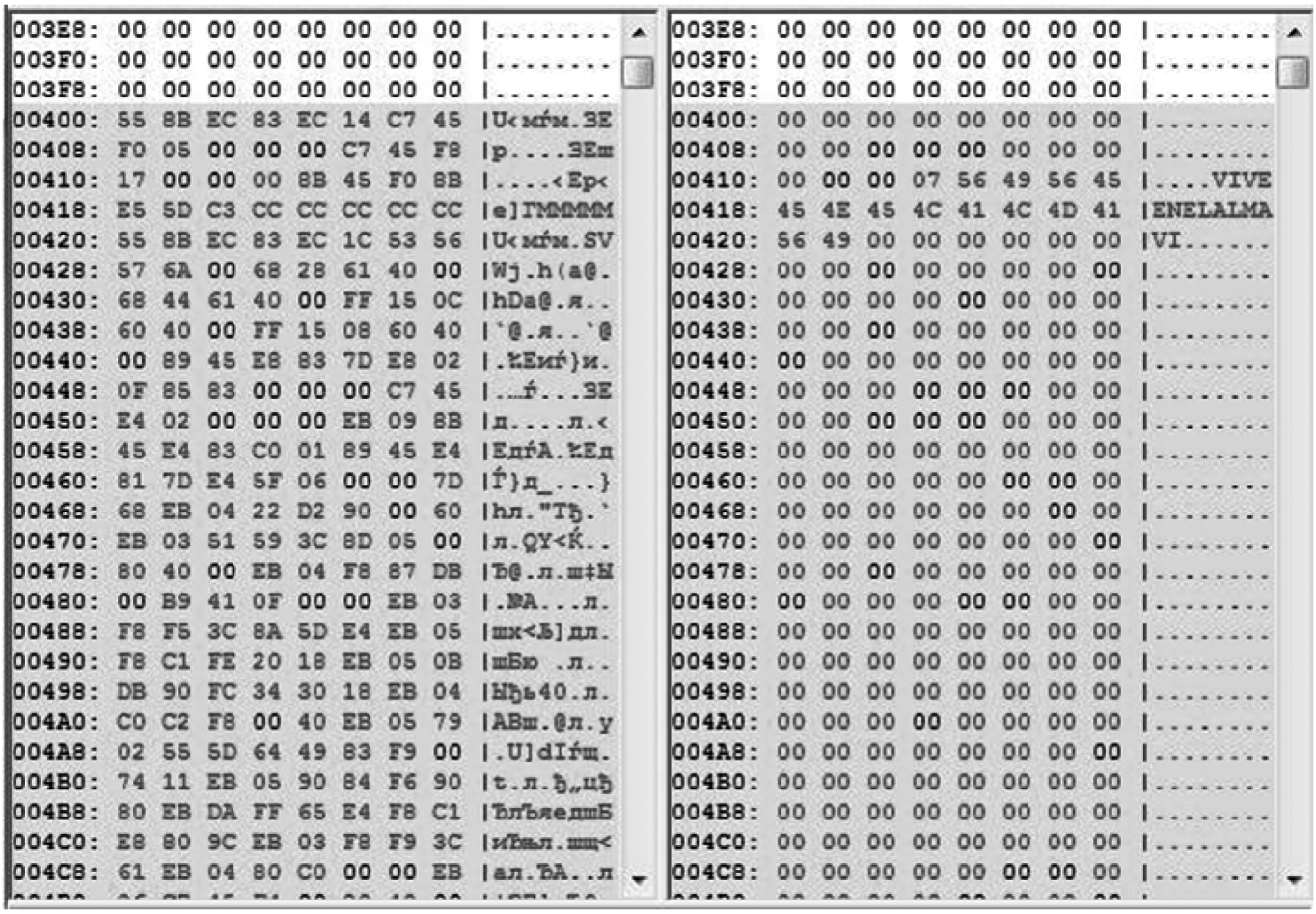

Рис. 5. Сравнение копий NgrBot

PE Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	Sectioin MD5
.text	4096	59776	61440	4,08314	5027fb97a60db04070ddc607ab6141f5
.data	65536	9772	4096	0,0	620f0b67a91f7f74151bc5be745b7110
.rsrc	77824	100660	102400	5,07721	5ae8dc0c72763d83c1c2d7cf75422a40

PE Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	Sectioin MD5
.text	4096	17659	17920	4,47736	18402d3b1eff468b3ff381ba732df8c7
.rdata	24576	8130	8192	3,28484	7407710f75d3232683ba8fe33de827ae
.data	32768	10240	7168	2,55455	59c6173eec2b21c5e6064f7160c12524
.rsrc	45056	57952	58368	5,54193	1757eaa1c0ad64e602ca1d659ecac60b
.reloc	106496	94208	20992	5,48767	63bbba5a7ca3e2a8d356c86b322baa3e

Рис. 6. PE структуры двух копий NgrBot
(MD5: ee66a7139bce6a4f9cab1e8d368cd287, MD5: fe6364de90e740b2db420940866204f8)

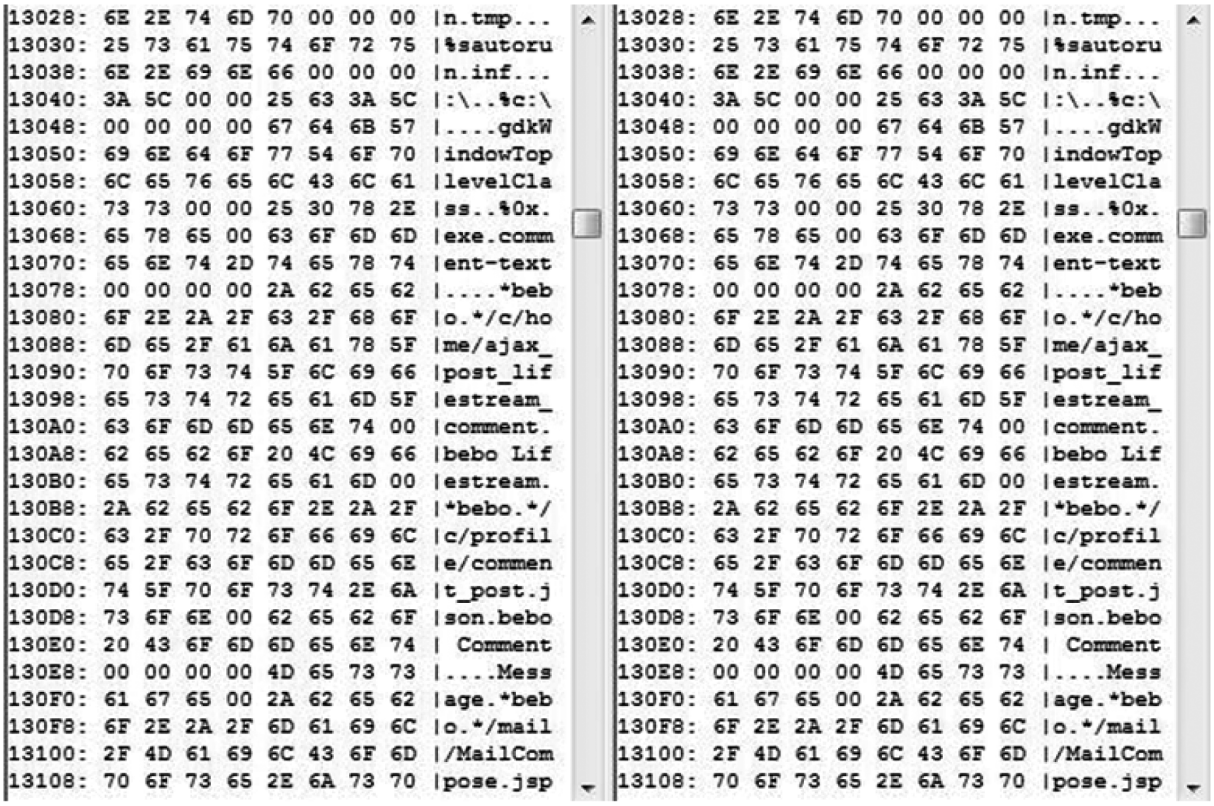

Рис. 7. Сравнение дампов двух разных инъекций Ngrbot: alg.exe_248_rwx_00A90000_0004E000.dmp и alg.exe_640_rwx_00A90000_0004E000.dmp (319488 байт)

Результаты сканирования показывают, что большинство антивирусных программ не могут обнаружить Ngrbot в памяти. Для упрощения процесса поиска уникальных строк дампа, которые будут использоваться для создания правил Yara, будем использовать утилиту String. В приложении В приведен фрагмент строк «Dump_013E0000_0004E000.dmp».

Фрагмент представляет названия интернет-ресурсов, заблокированных антивирусной программой, а также уникальный маркер «ngrBot». Используя строки «Dump_013E0000_0004E000.dmp», создадим правило Yara (рис. 12).

В вышеприведенном правиле необходимо проверить все строки в $ a1— $ a9 или только «facebook», «twitter», «symantec», «threatexpert» со стандартным маркером «ngrBot». Видно, что строка «ngrBot» исключается из первой части состояния. Это связано с образцами без подписи «ngrBot» в дампе. Используя созданное правило, сканируем процесс Explore.exe PID. Команда для сканирования процесса Explore.exe: Yara.exe Yara.txt 1544> YaraResult

Результаты тестирований и противодействие уязвимостям

В результате работы алгоритмов FP-Tree и FP-Growth были получены частотные дерева и вытекающие из них частотные шаблоны.

В результате работы реализованного метода Frequent Pattern Analysis (FPA) в рамках среды Lavasoft MAS для детектирования вредоносных и фишинг-ссылок «in-the-wild» (с «дикой» среды сети Интернет) были получены следующие результаты.

В таблицах 2–4 приведены результаты по следующим категориям соответственно: количество заблокированных ссылок, которые были определены как вредоносные данным методом (Malicious); количество ложных срабатываний (FP — False Positives), то есть чистая ссылки была определена как вредоносная; количество незадетектированных вредоносных ссылок данным методом (FN — False Negatives), но задетектированные GSB и IDS. Были использованы сторонние (бесплатные) сканеры IDS Suricata (далее IDS) [17] и Google Safe Browsing (далее GSB) [18]. Для верификации ложных срабатываний использовался онлайн-мультисканер VirusTotal [19].

Использование трехсторонних URL сканеров, позволило сравнить результаты детектирования с использованием разработанного метода с другими сканерами, а также провести оценку каждого из сканеров по сравнению с двумя другими (Табл. 2).

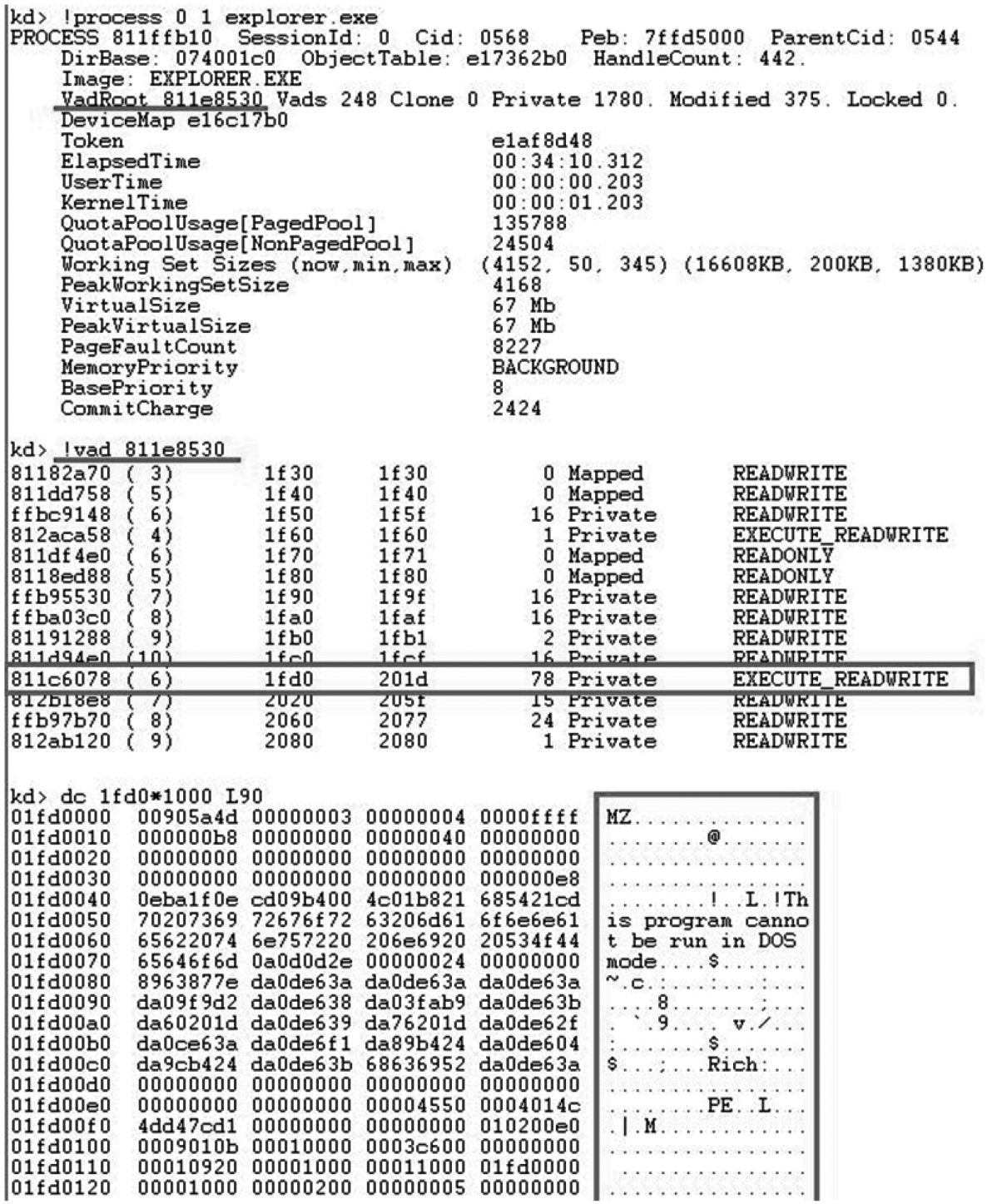

Рис. 8. Поиск инъекций Ngrbot в Windbg

Таблица 2.

Статистика детектирования вредоносных ссылок

Detection rates	05.2023	06.2023	07.2023	08.2023	Av.,%
FPA method	1112	520	1145	+1247	4,93
IDS Suricata	31	40	34	32	0.16
GSB	89	29	13	202	0,43
Total URLs	19263	22407	22266	19215	

Таблица 3.

Статистика ложных срабатываний

FN rates	05.2023	06.2023	07.2023	08.2023	Av.,%
FPA method	3	5	15	12	0.04
IDS Suricata	0	0	0	0	0.00
GSB	0	0	0	0	0.00
Total URLs	19263	22407	22266	19215	

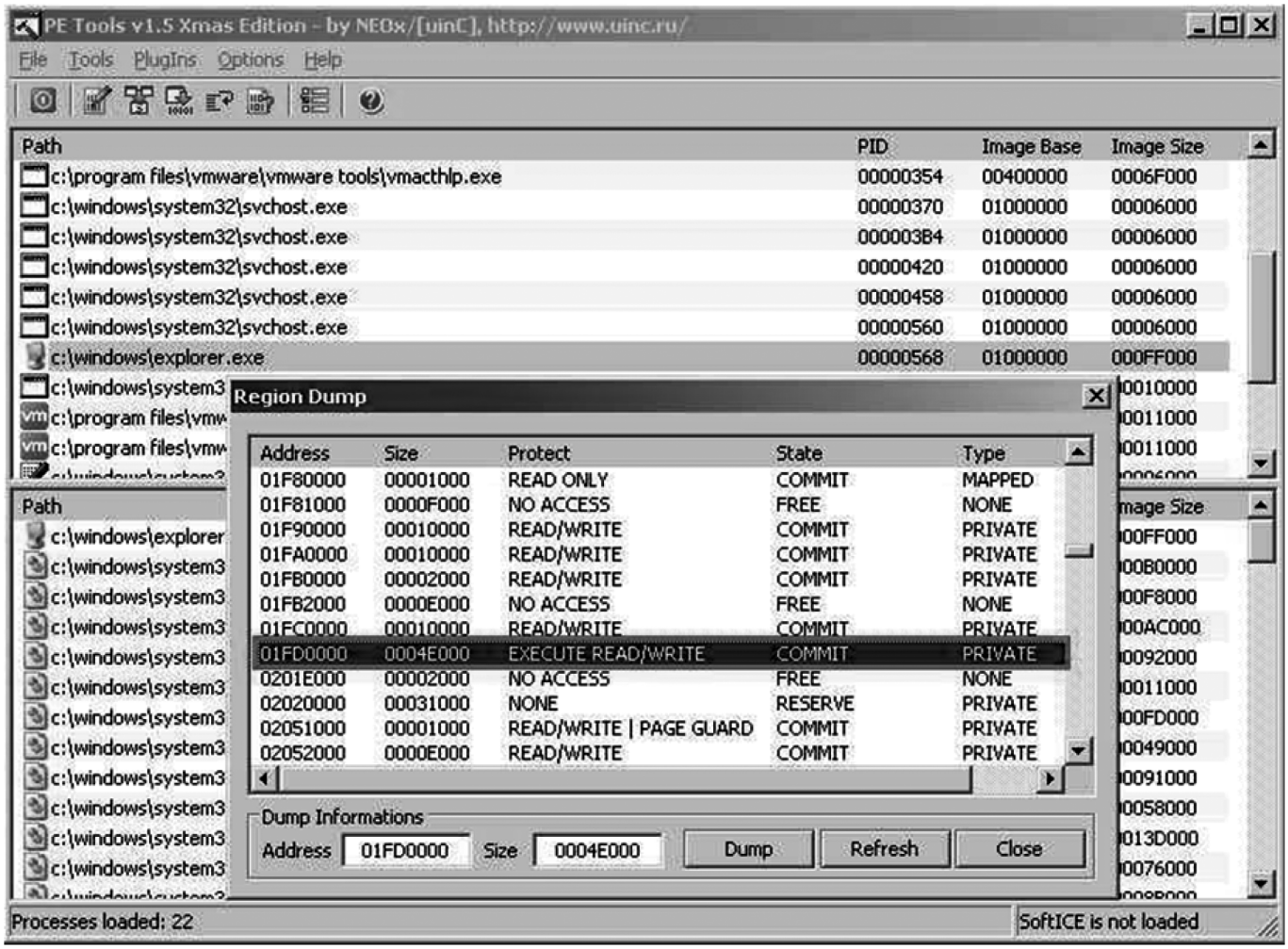

Рис. 9. Создание дампа Ngrbot с инструментами PE

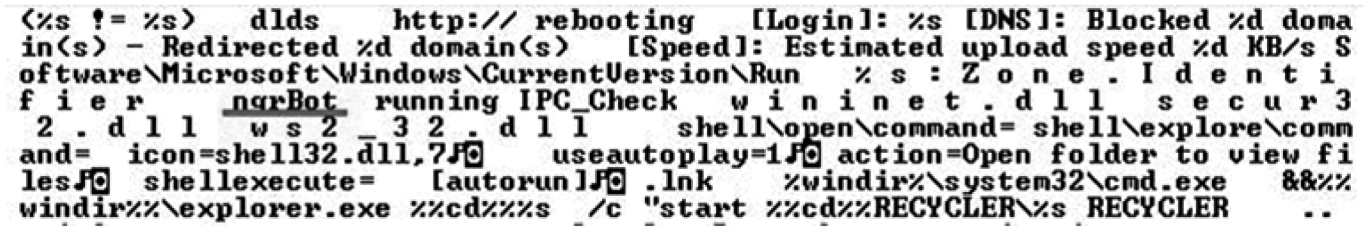

Рис. 10. Dump_01FD0000_0004E000.dmp фрагмент инъекции

Рис. 11. Результаты сканирования Ngrbot (Microsoft: Worm: Win32 / Dorkbot.A, Norman: W32 / Dorkbot.U, Sophos: W32 / Dorkbot-L)

Таблица 4.

Статистика пропущенных вредоносных ссылок

FN rates	05.2023	06.2023	07.2023	08.2023	Av.,%
FPA method	120	69	47	234	0.59
IDS Suricata	1198	544	1143	1437	5.32
GSB	1140	555	1164	1267	5.06
Total URLs	19263	22407	22266	19215	

```
rule WormDorkbot
{
    strings:
        $a1 = "facebook" nocase
        $a2 = "twitter" nocase
        $a3 = "symantec" nocase
        $a4 = "threatexpert" nocase
        $a5 = "vkontakte" nocase
        $a6 = "youtube" nocase
        $a7 = "admin" nocase
        $a8 = "letitbit" nocase
        $a9 = "lavasoft" nocase

        $b = "ngrBot"

    condition:
        (all of ($a*) ) or ($a1 and $a2 and $a3 and $a4 and $b)
}
```

Рис. 12. Yara правило для Ngrbot

Используя созданное правило, можно просканировать процесс Explore.exe по PID. Команда для сканирования процесса Explore.exe выглядит следующим образом: Yara.exe Yara.txt 1544> YaraResult Результаты представлены на рис. 15.

Yara успешно обнаружила вредоносную программу Ngrbot. Программу можно удалить вручную, руководствуясь описанием. Аналогичное правило можно создать для семейства вредоносных программ Shiz (рис. 16).

Подписи Yara, применяемые к уникальным строкам или байтовым последовательностям, взятым из дампов или инъекций вредоносных программ, позволяют идентифицировать полиморфные шпионские программы в системе.

Экспериментальная часть

Несмотря на то, что наблюдался переход от криптовымогательства к криптодобыче — новой перспективной области, чтобы заработать миллионы долларов США [20] — обгоняя, например, требования WannaCry по количеству инфекций [21], еще рано говорить о вымирании шифровальщиков. CyberSecurity Ventures предполагали, что к 2025 году глобальные потери от шифровальщиков превысят $11,5 млрд. [22]. Напротив, криптовалютный ажиотаж помогает преступникам еще больше обогащаться [23].

В январе 2018 года новое семейство шифровальщиков, которые работают по модели Шифровальщик-как-Сервис (Ransomware-as-a-Service — RaaS) GandCrab (декриптор для первой версии доступный [24]) показал неожиданный подъем в начале 2018 года [25] и затмил известных игроков RaaS с 2016 и 2017: Cerber, Locky и Spora [18]. Вторую [26] и третью [27] версию шифровальщиков GandCrab уже было выпущено в мае 2018 г. для продолжения игры в кошки мышки с исследователями безопасности.

Общепринятый сигнатурный подход не может обнаружить жестко закодированные шифры, которые не используют Crypto API или криптоконстанты. Например, RC4 не использует (крипто) константы и, следовательно, широко используется авторами вредоносных программ.

В эксперименте были использованы Krypto ANALyzer (KANAL) для инструмента PEiD [28] и общедоступные правила Yara [29, 30, 31] для выявления криптопримитивов.

Таблица 5.

Выявление крипто-примитивов в файлах PE и дампов памяти наследников с использованием правил PEiD и Yara.

Шифровальщик	Симметрический шифр	Источник данных	Сигнатурный детект (Yara, KANAL PEiD)
GlobeImposter	AES-256-CBC; RC4, 16-byte key	PE file	List of primes, Big numbers, CryptGenKey import
		Memory dump	List of primes, Big numbers, CryptGenKey import, Rijndael_AES_CHAR, Rijndael_AES_LONG
TeslaCrypt	AES-256-CBC	PE file	N/A
		Memory dump	CryptGenKey import, Big numbers
MoneroPay	Salsa20, 32-byte key	PE file	N/A
		Memory dump	N/A

Таблица 5 содержит результаты выявления криптопримитивов в PE файлах шифровальщика и дампа памяти. Полученные детективы показывают, что только в одном случае KANAL правильно обнаружил AES в шифровальщике GlobeImposter. Другими словами, выявление на основе подписи не может рассматриваться как надежный способ идентификации жестко закодированных шифров в шифровальщиках.

Предложенный способ предусматривает создание модели классификации, которая потребляет примеры кода шифра, а затем пытается найти известные фрагменты кода в шифровальщиках.

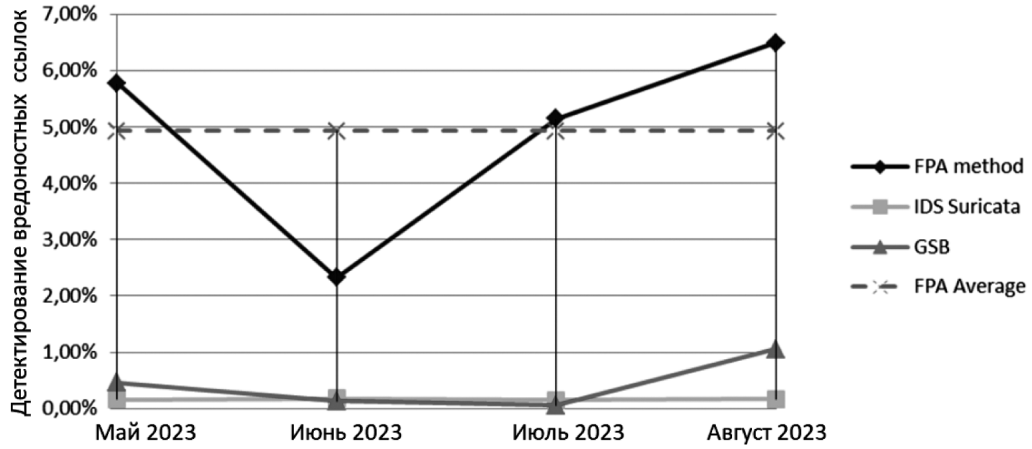

Рис. 13. Статистика детектирования вредоносных ссылок с использованием разработанного метода FPA, IDS системы и Google Safe Browsing онлайн сервиса

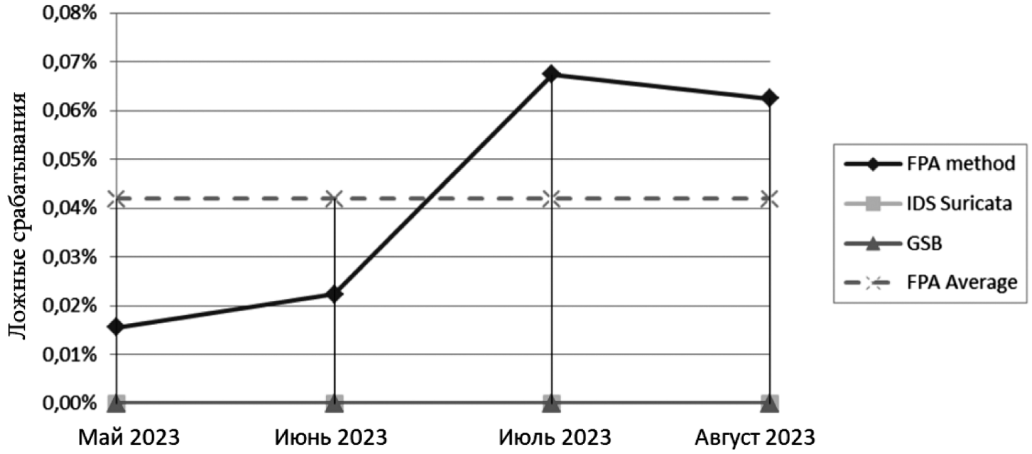

Рис. 14. Статистика ложных срабатываний с использованием разработанного метода FPA, IDS системы и Google Safe Browsing онлайн сервиса

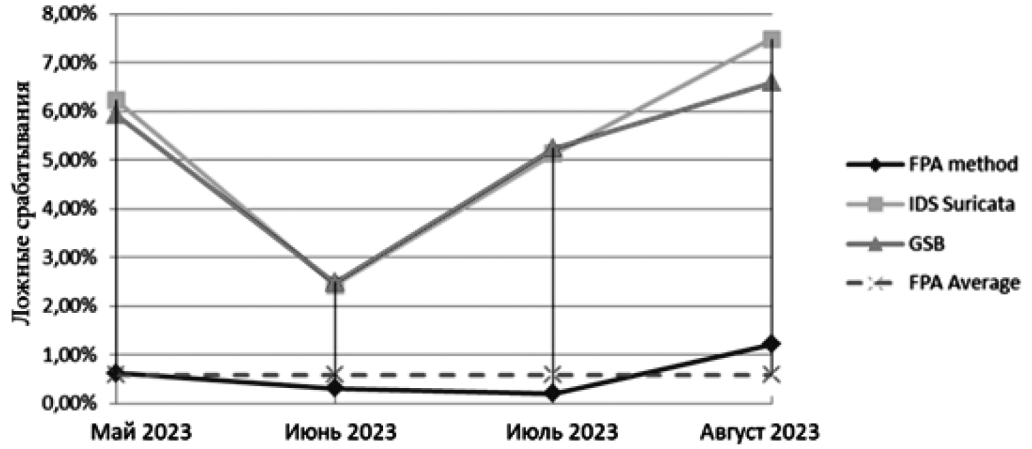

Рис. 15. Статистика пропущенных вредоносных ссылок с использованием разработанного метода FPA, IDS системы и Google Safe Browsing онлайн сервиса

Метод выявления криптопримитивов включает следующее:

1. Подготовка кодовых шаблонов шифров, представленных в Assembler — библиотеке образцов.
2. Исходный код C / C ++ загружается и компилируется с использованием различных параметров компиляции, таких как условие вызова: cdecl, stdcall, fastcall; и оптимизация генерации кода: по размеру, производительности, или без них, чтобы получить большее разнообразие шаблонов кода Assembler для распознавания данного шифра. В приложении параметр Security Check (/ GS)

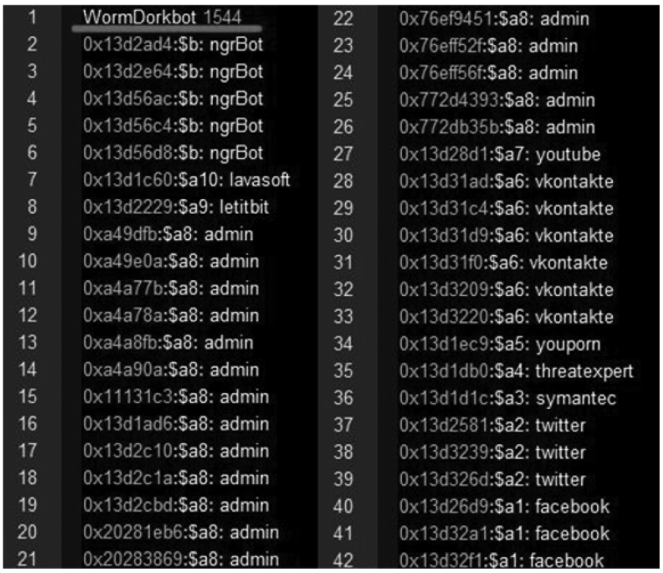

Рис. 16. Результаты сканирования процесса Explorer.exe с использованием Yara

```
rule Shiz
{
    strings:
    $a1 = "FAKTURA"
    $a2 = "IBANK"
    $a3 = "INIST"
    $a4 = "INTER"
    $a5 = "RAIFF"
    $a6 = "abc123"
    $a7 = "jordan23"
    $a8 = "xakep"

    condition:
        all of ($a*)
}
```

Рис. 17. Правило Yara для Shiz

выключен для устранения избыточного кода, добавленного компилятором VC.

3. Нормализация кода.
4. Замена имен регистров процессора и адресов памяти, которые зависят от конкретного компилятора и структуры PE-файла шаблонов для унификации кода Assembler. Например, тот же алгоритм, составленный двумя разными компиляторами с одинаковыми параметрами генерации кода, может использовать различные регистры ЦП и адреса для вызовов функций и сохранение переменных. Поэтому, прежде чем сравнивать два ассемблерных списка, необходимо заменить все имена реестра и адреса, а также удалить комментарии и соответствующие разделители. В противном случае значение расстояния Левенштейна будет больше.
5. Локализация шаблона криптокода в нормализованном коде, который соответствует использованию библиотеки Google Match.
6. Получение векторов diffs с использованием алгоритма разложения Myer [32] между кодом шаблона и фрагментами соответствующего кода в требованиях к испытанию.
7. Расчет расстояния Левенштейна [32] для векторов diffs, отражающего минимальное количество вставленных, удаленных или замененных символов, чтобы изменить шаблон в соответствующий код в требованиях.

Формально для двух строк S1 и S2 длины n и m, соответственно расстояние Левенштейна можно рассчитать, используя следующую формулу повторения:

D (n, m) = 0, когда i = 0, j = 0;
D (n, m) = i, когда i> 0, j = 0;
D (n, m) = j, когда i = 0, j> 0;
D (n, m) = min (D (i, j-1), D (i-1, j), D (i-1, j-1) + 1S1[i] ≠ 1S2[i], когда i > 0, j > 0, где i∈{0,1, ..}, j∈{0,1, ...}.

Если значение расстояния Левенштейна ниже заданного порога, два фрагмента кода можно рассматривать как связанные с тем же криптопримитивом. В этом случае можно предположить, что данный шифровальщик использует шифр, к которому принадлежит шаблон. Порог может быть получен экспериментальным путем или указанным аналитиком вредоносных программ.

8. Распознанный криптопримитив добавляется в библиотеку образцов, которая будет использована позже для идентификации того же шифра.

Эксперименты проводились с помощью библиотек Google Diff, Match Patch [33]. Библиотеки обеспечивают интеллектуальное сопоставление фрагментов кода, получение разностных векторов с использованием алгоритма различий Майера и вычисления расстояния Левенштейна.

Во время экспериментов использовался порог соответствия по умолчанию, равный 0,5.

Таблица 6.

Образцы вымогателей, использованных в экспериментах

Ransomware	SHA256
TeslaCrypt	9e3827dffc24d1da72cb3d423bddf4cd535fa636062e4ea63421ef327fec56ad
GlobeImposter	a0e5bced56025f875721043df981c400fc28e4efc68ffe42ac665633de085ab1
MoneroPay	ababb37a65af7c8bde0167df101812ca96275c8bc367ee194c61ef3715228ddc

Результаты экспериментов показывают соответствие кода расширения ключа AES у вымогателей TeslaCrypt и GlobeImposter, кода PRGA RC4 у вымогателей GlobeImposter и кода кватерфунда Salsa20 у вымогателей MoneroPay.

Таблица 7.

Распознавание AES (расширение ключа) у вымогателей TeslaCrypt

Iteration No	1	2	3	4	5
Expected location	0	1500	3000	10000	20000
Matched location	115	1473	2986	10006	19953
Levenshtein distance	95	60	93	76	75
Correct match in ransomware	FALSE	TRUE	FALSE	FALSE	FALSE

Таблица 8.

Распознавание AES (расширение ключа) у вымогателей GlobeImposter

Iteration No	1	2	3	4	5
Expected location	100	1000	4400	10000	20000
Matched location	399	999	4425	9968	19991
Levenshtein distance	61	113	50	132	91
Correct match in ransomware	FALSE	FALSE	TRUE	FALSE	FALSE

Таблица 9.

Распознавание RC4 (PRGA) у вымогателей GlobeImposter

Iteration No	1	2	3	4	5
Expected location	0	500	800	1000	1500
Matched location	340	340	828	1063	1553
Levenshtein distance	20	20	76	75	83
Correct match in ransomware	TRUE	TRUE	FALSE	FALSE	FALSE

Таблица 10.

Распознавание RC4 (PRGA) у вымогателей GlobeImposter

Iteration No	1	2	3	4	5
Expected location	0	100	1000	1500	3000
Matched location	2	100	1000	1500	3094
Levenshtein distance	118	146	177	619	389
Correct match in ransomware	TRUE	FALSE	FALSE	FALSE	FALSE

Примеры шаблонов шифрования, сценария Python, используемого для нормализации кода ASM, и полученные векторы Diffs можно найти в общедоступном репозитории [34]. Чтобы повысить производительность и точность описанного метода распознавания образов, рекомендуется указать приблизительное местоположение, где криптопримитив может появиться в целевом файле. Выявление на основе сигнатур константами может помочь методу найти криптопримитивы в вымогателях. Например, для поиска местоположения кода, который принадлежит симметричным шифрам AES, Salsa20 и TEA, которые имеют эти константы.

Заключение

Как следует из проанализированных данных, метод детектирования Интернет-ссылок на основе анализа атрибутов с использованием частотных паттернов в формах обратной связи позволяет определить 4,93 % (таблица 3) ссылок как вредные на основании анализа их атрибутов из общего числа ссылок «in-the-wild». IDS система и GSB сервис смогли определить 0,16 % и 0,43 % ссылок как вредоносные, соответственно в том же наборе.

Среднее число ложных срабатываний у метода FPA выше (таблица 4), чем у IDS и GSB сервисов, и составляет 0,04 % (9 ссылок) от общего числа ссылок. В контексте детектирования подозрительных Интернет-ссылок данный показатель не является критическим, как, например, для приложений, поскольку пользователь может зайти на заблокированную страницу, проигнорировав предупреждение системы безопасности.

При этом среднее количество пропущенных вредных и фишинг-ссылок выше у систем IDS и GSB и составляет 5,32 % и 5,06 % соответственно при том, что FPA метод пропустил всего 0,59 % опасных ссылок (таблица 5). Данный параметр является наиболее существенным в системах блокировки вредоносных фишинг-сайтов, так как позволяет предупредить пользователя о потенциальной опасности при посещении зараженного веб-ресурса.

В результате сравнительного анализа метода сигнатурного детектирования IDS системы Suricata и метода детектирования с использованием черных списков Google Safe Browsing с магическим методом FPA на основе анализа атрибутов Интернет-ссылок было отмечено значительное преимущество метода по количеству детектируемых вредоносных и фишинг-ссылок. При этом разработанный метод FPA имеет приемлемый процент ложных срабатываний (0,04 %), что можно считать незначительным недостатком метода на данный момент.

В результате проведенного анализа, технология полиморфизма может обеспечить надежную защиту новых частей программ-шпионов от обнаружения антивирусом нулевого дня, делая себя практически невидимым в компьютерной системе.

Более того, после установки полиморфные бэкдоры могут запускать процедуру обновления для загрузки новой версии шпионского ПО, что увеличивает срок его службы на зараженном компьютере. Представлен также способ обнаружения полиморфных шпионских программ и то, как этот подход главным образом основан на динамическом анализе образцов. После запуска полиморфная программа-шпион обнаруживает свою вредную полезную нагрузку непосредственно в памяти

процесса. Этот метод может быть успешно обнаружен с помощью правил Yara, специально созданных для семей Ngrbot и Shiz.

На основе проведенных экспериментов определено, что одним из возможных вариантов применения метода являются корпоративные сети. Используя описанные методы, администратор или инженер по безопасности может создать правило Yara для определенного семейства шпионских программ и начать выявлять активное заражение в сети. Как только вирус обнаружен, руководство по удалению может помочь вылечить систему.

Заключение по результатам поиска криптопримитивов в формах обратной связи. Результаты экспериментов подтвердили первоначальную гипотезу о возможности идентификации криптопримитивов в упакованном и деобфускированном коде вымогателей. Случаи, представленные в таблицах 6–10, имеют правильно распознанный фрагмент кода с минимальным расстоянием Левенштейна.

Таким образом, показана возможность эффективного распознавания криптопримитивов в коде вымогателей на основе кода ассемблера ethalon с симметричных шифров, которые рассматриваются как шаблоны поиска. Некоторая часть данного подхода также описана в [35].

Предложенный метод успешно обнаружил криптографический код симметричных шифров AES, RC4, Salsa20 в вымогателях TeslaCrypt, GlobeImposter и MoneroPay. Однако, основанный на сигнатурах подход, с использованием правил Yara и открытым исходным кодом и KANAL в PEiD, показал низкие возможности обнаружения.

Описанный способ может быть дополнительно улучшен, чтобы автоматически генерировать правило Yara для выявления фрагмента кода, который совпал с вымогателем, ответственным за симметричное шифрование данных.

ЛИТЕРАТУРА

1. Introduction to Natural Language Processing (NLP) [Электронный ресурс]. Режим доступа: https://towardsdatascience.com/introduction-to-natural-language-processing-nlp-323cc007df3d
2. Щипина Л.Ю. Информационные технологии в лингвистике. Учебное пособие — 2013. С.43–52.
3. Зубова И.И. Информационные технологии в лингвистике: Учебное пособие. — МГЛУ. — Мн., 2001.
4. T.T. Lee. Algebraic Theory of Relational Databases. -The Bell System Technical Journal Vol 62, No 10.-1983.-pp. 3159-3204.
5. E. Louie, T.Y. Lin. Finding Association Rules using Fast Bit Computation: Machine-Oriented Modeling // Foundations of Intelligent Systems, Z. Ras and S. Ohsuga.-Springer-Verlag, 2000. — pp. 486–494.
6. Y.D. Cai, N. Cercone, and J. Han. Attribute-oriented induction in relational databases, “in Knowledge Discovery in Databases, pp. 213-228. AAAI / MIT Press, Cambridge. — MA. — 1991.
7. T.Y. Lin, S. Oshuga, CJ Liau, X. Hu, S. Tsumoto. Foundations of Data Mining and Knowledge Discovery. -Springer.-2005-pp.34-53.
8. Borgelt C., An Implementation of the FP-growth Algorithm. — OSDM’05, 2005, 135 p.
9. Mao R., Adaptive-FP: An Efficient and Effective Method for Milti-Level Multi-Dimensional Frequent Pattern Mining. — Simon Fraser University. — 2001. — 56 p.
10. PhishTank, онлайн сервис. Доступен: http://www.phishtank.com/, январь 2014.
11. Alexa.com, онлайн сервис. доступен: http://www.alexa.com /, январь 2014.
12. IDS Suricata, открытый IDS / IPS / NSM движок. Доступен: http://suricata-ids.org/, январь 2014.
13. Google Safe Browsing, онлайн сервис Google для проверки URL. Дос-пен: https://developers.google.com/safe-browsing/, январь 2014.
14. VirusTotal, онлайн мультисканнер. Доступен: https://www.virustotal.com/, январь 2014.
15. The Yara rules to detect crypto primitives, https://github.com/polymorf/findcrypt-yara
16. Russinovich ME, Solomon DA. Microsoft Windows internals, Microsoft Windows Server (TM) 2003, Windows XP, and Windows 2000 (Pro-Developer). 4th ed. Redmond, WA, USA: Microsoft Press, ISBN 0735619174; 2004.
17. Система обнаружения вторжений IDS Suricata [https://suricata-ids.org/]
18. Google Safe Browsing [https://safebrowsing.google.com/]
19. Веб-ресурс мультисканнера VirusTotal [https://www.virustotal.com/]
20. Smominru Monero mining botnet making millions for operators, Proofpoint, January 31, 2018, available at https://www.proofpoint.com/us/threat-insight/post/smominru-monero-mining-botnet-making-millions-operators
21. 2017 Annual Security Roundup: The Paradox of Cyberthreats, TrendMicro, 2017, available at https://documents.trendmicro.com/assets/rpt/rpt-2017-Annual-Security-Roundup-The-Paradox-of-Cyberthreats.pdf
22. Global Ransomware Damage Costs Predicted To Hit $ 11.5 Billion By 2019, Cybersecurity Ventures, 2018, available at https://cybersecurityventures.com/ransomware-damage-report-2017-part-2/
23. SpriteCoin: Another New CryptoCurrency ... or NOT!, Fortinet, January 22, 2018, available at https://www.fortinet.com/blog/threat-research/spritecoin-another-new-cryptocurrency-or-not.html
24. No More Project» website, https://www.nomoreransom.org/en/index.html
25. https://twitter.com/WDSecurity/status/968270740549193730

26. Lawrence Abrams, GandCrab Ransomware Version 2 Released With New .Crab Extension & Other Changes, BleepingComputer.com, March 6, 2018.
27. Lawrence Abrams, GandCrab Version 3 Released With Autorun Feature and Desktop Background, BleepingComputer.com, May 4, 2018, available at https://www.bleepingcomputer.com/news/security/gandcrab-version-3-released-with-autorun -feature-and-desktop-background/
28. PEiD Tool website, http://peid.has.it
29. The Yara rules to detect crypto primitives, https://github.com/Yara-Rules/rules/tree/ae82fb6e1e3145a85f52c4856985f7743796aae6/Crypto
30. The Yara rules to detect crypto primitives, https://github.com/x64dbg/yarasigs
31. The Yara rules to detect crypto primitives, https://github.com/polymorf/findcrypt-yara
32. E.W. Myers, An O (ND) Difference Algorithm and Its Variations, Department of Computer Science, University of Arizona, Tucson, US., Available at https://neil.fraser.name/writing/diff/myers.pdf
33. Haggarty R., Discrete mathematics for computing, Pearson Education Limited, 2002 pp. 91–113
34. Google Diff, Match i Patch [https://github.com/google/diff-match-patch] Diffs векторы [https://github.com/AlexanderAda/NioGuardSecurityLab/tree/master/Ranso mwareAnalysis/DiffMatchPatterns]
35. Русаков, А.М. Программное средство обнаружения активных эксплоитов в корпоративном комьюнити / А.М. Русаков, И.В. Красноперова, Н.З. Циклаури // Наукосфера. — 2023. — № 1–2. — С. 252–261. — DOI 10.5281/zenodo.7584323. — EDN CWGGYG.