Заключение. В работе предложена структура аппаратно-программного комплекса мониторинга жизнедеятельности человека, которая позволяет пациентам самостоятельно отслеживать жизненные показатели состояния своего здоровья, а лечащим врачам – контролировать в реальном времени состояние здоровья своих пациентов в режиме удаленного доступа и проводить анализ результатов наблюдения с помощью специальных мобильных программ-приложений с целью своевременного и эффективного корректирования лечебных и профилактических мероприятий.

Эффективная работа системы обеспечивается реализацией предложенного алгоритма телемониторинга состояния здоровья человека, соответствие конечных устройств сложившимся в ходе исследования требованиям и использованием протоколов MQTT и HTTPS.

СПИСОК ЦИТИРОВАННЫХ ИСТОЧНИКОВ

1. Ericsson White Paper, Cellular networks for massive IoT [Электронный ресурс]. – 2016. – Режим доступа: https://www.ericsson.com/res/docs/whitepapers/wp_iot.pdf.
2. Scott, M. Telehealth Industry Trends 2015 [Электронный ресурс] – 2015. – Режим доступа: http://slideplayer.com/slide/9933303/.
3. Chamberlin, B. Healthcare Internet of Things: 18 trends to watch in 2016 [Электронный ресурс]. – 2016. – Режим доступа: https://ibmcai.com/2016/03/01/ healthcare-internet-of-things-18-trends-to-watch-in-2016/.
4. Ericsson, Ericsson Mobility Report [Электронный ресурс]. – 2015. – Режим доступа: http://www.ericsson.com/res/docs/2015/mobility-report/ericsson-mobility-report-nov-2015.pdf.
5. Kay, M. Telemedicine: opportunities and developments in Member States: report on the second global survey on eHealth / M. Kay, J. Santos, M. Takane. // WHO Press. – 2010. – №2. ISBN 978 92 4 456414 1 ISSN 2220-5462
6. WHO Group Consultation on Health Telematics (1997: Geneva, Switzerland). A health telematics policy in support of WHO's Health-for-all strategy for global health development : report of the WHO Group Consultation on Health Telematics, 11-16 December, Geneva, 1997 [Электронный ресурс] / WHO Group Consultation on Health Telematics (1997: Geneva, Switzerland) // Geneva : World Health Organization. – 1998. – Режим доступа: http://www.who.int/iris/handle/10665/63857.
7. Владзимирский, А.В. Телемедицина [монография] / А.В. Владзимирский. – Донецк : ООО "Цифровая типография", 2011. – 437 с.
8. Блажис А. К. Телемедицина / А. К. Блажис, В. А. Дюк. – Санкт-Петербург : СпецЛит, 2001. – 143 с. – ISBN 5-299-00084-7.
9. Вишневский, В.В. Телемедицинские технологии и научные исследования / В.В. Вишневский // Український журнал телемедицини та медичної телематики. – 2006. – № 1.
10. Multi-purpose HealthCare Telemedicine Systems with mobile communication link support / E Kyriacou, S Pavlopoulos, A. Berler [и др.]. – [Электронный ресурс] – 2003. – Режим доступа: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC153497/.
11. Michael, C. Batistatos. Mobile telemedicine for moving vehicle scenarios: Wireless technology options and challenges / C. Michael, G.V. Batistatos, Tsoulos, E. Georgia Athanasiadou // Journal of Network and Computer Applications. – 2012.
12. A Comprehensive Survey of Wireless Body Area Networks / Sana Ullah, Bart Braem, Ingrid Moerman [и др.]. // Journal of Medical Systems. – 2010.
13. Беззуб, І.І. Телемедицина в Україні: реалії та перспективи [Электронный ресурс] – Режим доступа: http://nbuviap.gov.ua/index.php?option=com_content&view= article&id=2466: telemeditsina-v-ukrajini&catid=8&Itemid=350.
14. Rouse, M. MQTT (MQ Telemetry Transport) [Электронный ресурс] / Margaret Rouse. – 2015. – Режим доступа: http://internetofthingsagenda.techtarget.com/definition/MQTT-MQ-Telemetry-Transport.
15. MQTT. Frequently Asked Questions [Электронный ресурс] – Режим доступа: http://mqtt.org/faq.
16. MQTT Essentials Part 2: Publish & Subscribe [Электронный ресурс] – Режим доступа: http://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe.
17. Что такое MQTT и для чего он нужен в IIoT? Описание протокола MQTT [Электронный ресурс]. – 2016. – Режим доступа: https://ipc2u.ru/articles/prostye-resheniya/chto-takoe-mqtt/.
18. Проектирование телекоммуникационной инфраструктуры «умных» территориальных общин, городов и регионов: состояние, опыт, перспективы / Н.Е. Кунанець, Г.П. Химич, А.В. Мацюк // Управление проектами: состояние и перспективы: материалы XII Международной научно-практической конференции, 13–16 сентября 2016. – Николаев. – С. 160–162.
19. Системные комплексы информационных технологий в проектах "Умный город" / А.М. Дуда, Н.Е. Кунанець, А.В. Мацюк, В.В. Пасичник // Системный анализ и информационные технологии: материалы 18-й Международной научно-технической конференции SAIT 2016 / ННК "ІПСА" НТУУ "КПІ", 30 мая – 2 июня 2016 р. – Киев: ННК "ІПСА", 2016. – С. 215–216.

Материал поступил в редакцию 08.01.2016

MARCENKO S.V., MATSIUK O.V., MYTNYK O.M., LOBUR T.B., PASICHNYK V.V. Hardware-software complex for telemonitoring of vital signs

The architecture of the hardware-software complex of vital signs monitoring ensuring QoS of telemetry data in the network is designed. The relevance of the research area and use cases of the developed hardware-software complex are analyzed. The complex based on multilevel scalable personalized system including the combination of two architectures: client-server and publisher-subscriber, are proposed. The operation algorithm of the remote health care system is proposed. The requirements, which should be met by the vital signs monitoring devices, are set. The types of end devices and the way of their interacting with the system of the hardware-software complex are determined. Options of using MQTT and HTTPS protocols in telemedicine area are analyzed. The modelling of the prototype of hardware-software complex in Cisco Packet Tracer 7 is carried out.

УДК 004.91.2

Савенко О.С.

РАСПРЕДЕЛЕННАЯ МНОГОУРОВНЕВАЯ СЕТЕВАЯ СИСТЕМА ОБНАРУЖЕНИЯ МЕТАМОРФНЫХ ВИРУСОВ В ЛОКАЛЬНЫХ КОМПЬЮТЕРНЫХ СЕТЯХ

Введение. Распространение информационных систем тесно связано с проблемой безопасности таких систем и использованием программных средств для обнаружения вредоносного программного обеспечения. Однако известные антивирусные средства не всегда подтверждают задекларированный уровень обнаружения, что проявляется в постоянных вспышках вирусной активности. Убытки, причиненный вредоносным программным обеспечением, достигают миллиардов долларов. Так, например, согласно отчету компании Cybersecurity ventures убытки, нанесенные вирусом WannaCry в 2017 году, составили около 15 млрд долларов, что в 15 раз выше, чем в 2015 году [1].

Савенко Олег Станиславович, *к.т.н., декан факультета программирования, компьютерных и телекоммуникационных систем, доцент Хмельницкого национального университета.*
Украина, ХНУ, 29016, г. Хмельницкий, ул. Институтская, 11, 4-214.

Среди всего множества вирусных программ одно из центральных мест занимают метаморфные вирусы. Сложность выявления такого типа вредоносного программного обеспечения обусловлена использованием ими технологий видоизменения собственного кода при каждом новом инфицировании. Две копии одного и того же метаморфного вируса будут синтаксически различными, но иметь при этом одинаковую семантику. Такая особенность метаморфных вирусов не позволяет построить сигнатуру (подпись) вирусного кода, оставляя проблему выявления метаморфных вирусов открытой.

Данная работа посвящена решению актуальной научной проблемы повышения достоверности выявления метаморфных вирусов путем разработки распределенной многоуровневой сетевой системы обнаружения с привлечением методов машинного обучения.

Обзор известных работ и постановка задачи. В работе будем рассматривать такой тип вредоносного программного обеспечения, как метаморфные вирусные программы. Сегодня значительное внимание уделяется проблеме выявления метаморфных вирусов. Для их выявления применяется ряд техник, в частности: выявление на основе опкодов (opcode-based analysis), анализ потока информации (information flow analysis) и потока управления (control flow analysis), а также на основе анализа поведенческих полиморфных преобразований [2–5]. В работе [2] для выявления метаморфных вирусов применяются скрытые марковские модели (СММ). В своей работе авторы осуществили многократное обучение СММ для различных метаморфных генераторов, опираясь на n-граммный анализ операционных кодов, полученых в процессе дизассемблирования. Предложенный подход продемонстрировал эффективность выявления новых метаморфных вирусов на уровне 90% для трех типов вирусных классов: zbot, winwebsec и zeroaccess.

Метод выявления метаморфных вирусов на основе анализа информационных потоков представлен в работе [3]. В своей работе авторы используют метод обнаружения, который основывается на анализе значений программы в процессе ее выполнения (value set analysis). Подход предусматривает, что каждая программа в момент выполнения может быть представлена набором знаний ячеек памяти и регистров. Неизвестная программа размещается в защищенной среде, после чего для каждого API вызова осуществляется анализ регистров до и после API вызова. На основе изменения значений в регистрах формируется вектор признаков для каждого регистра. Предложенный подход показал высокую эффективность с процентом ложных срабатываний на уровне 2,9%. Однако представленный метод не учитывает техник уклонения от эмуляции, использование которых может значительно снизить процент обнаружения.

В работе [4] представлен подход к выявлению метаморфных вирусов с использованием модифицированных эмуляторов, которые располагаются на хостах корпоративной сети. Метод предусматривает классификацию метаморфных вирусов с привлечением нечеткой логики. Технология позволяет отнести исследуемое программное обеспечение к одному из классов метаморфных вирусов, в которых применена техника обфускации (запутывания) кода. Результаты экспериментальных исследований продемонстрировали эффективность обнаружения метаморфных вирусов на уровне 85%. Однако предложенный метод характеризуется значительным количеством ложных срабатываний. Кроме того, предложенный метод характеризовался большой избыточностью обмена информацией внутри сети – рассылка подозрительной программы осуществлялась на все хосты сети без предварительной проверки на сходство с метаморфными вирусами.

В работе [5] представлен метод обнаружения метаморфных вирусов на основе поиска сходства графов, представленных операционными кодами. Для построения ориентированного графа операционных кодов используются множество дизассемблированных инструкций, где вершинами выступают операционные коды, а ребрами – переходы между ними. Каждое ребро графа содержит вероятность перехода из инструкции A к инструкции B с учетом количества подобных переходов. Классификация осуществляется на основе использования метода опорных векторов, где прямо осуществляется сравнение графов опкодов. Однако увеличение размера файла приводит к увеличению количества опкодов и соответственно увеличивает размер графа, что приводит к решению NP полной задачи изоморфизма графов.

Анализ известных методов показал, что в процессе выявления метаморфных вирусов применяется процедура эмуляции выполнения программного кода, которая заключается в переноске вируса в виртуальную среду с целью исследования его поведения. Значительная часть метаморфных вирусов может отслеживать выполнение в виртуальной защищенной среде и не активироваться. Поскольку известны методы являются хост-ориентированными, данный факт значительно снижает эффективность выявления таких вирусов.

Другой проблемой выступает то, что не разработаны информационные системы, которые бы вобрали в себя разработанные для определенных типов вредоносного программного обеспечения различные методы обнаружения. И эти информационные антивирусные системы, будучи многофункциональными, могли бы позволить наполняться новым функционалом и стали бы основой для развития конкурентной среды на рынке антивирусных информационных систем. Эффективность таких систем выросла бы при использовании их в корпоративных компьютерных сетях, где дополнительным преимуществом при обнаружении вредоносного программного обеспечения выступают сведения о программно-аппаратном обеспечении компьютерных систем сети и учета информации о поведении программного обеспечения из разных компьютеров сети.

Распределенная многоуровневая сетевая система (РМСС) обнаружения вредоносного програмного обеспечения (ВПО) в локальных компьютерных сетях. Рассмотрим вариант возможного использования системы обнаружения вредоносного программного обеспечения с метаморфным функционалом в корпоративной компьютерной сети.

Обнаружения вредоносного программного обеспечения осуществляется с помощью разнообразных средств. Эффективность и достоверность обнаружения существенно зависят в том числе и от архитектуры таких средств, а также их позиционирования и места размещения в компьютерных системах локальных сетей с учетом того, что процесс выявления ВПО проводится в локальных сетях, выбор модели функционирования РМСС предусматривал привлечение информации из всех компьютерных систем (КС) локальной сети, то есть размещения на всех КС этой системы. Это необходимо для повышения эффективности и достоверности обнаружения за счет учета информации о состоянии из других КС для принятия решения в конкретной КС. Эти основные требования, что система должна быть размещена в сети на каждой КС, влияли на выбор модели ее архитектуры. Также важным для таких систем является то, чтобы центр принятия решений системы не представлялся и идентифицировался однозначно, потому что его обнаружение приведет к атаке на него, чтобы вывести всю систему из рабочего состояния. Разработанная система построена так, что размещенно на КС в сети ее части эффективно обмениваются между собой информацией о состоянии каждой КС для принятия решения. Система обнаружения ВПО соответствующим образом структурировалась, чтобы иметь возможность наращиваться и ее увеличение не должно замедлять процесс обнаружения. Таким образом, система обнаружения ВПО в локальных сетях обладает следующими характеристиками:

1) распределенность в локальной сети на всех КС;
2) формирование единой системы всеми частями, расположенным в КС;
3) децентрализованность структуры во избежание вывода системы из строя при обнаружении единого центра принятия решений или управления системой;
4) возможность принятия решения модулем, расположенным в определенной КС, при поддержке других частей системы;
5) многоуровневость для четкого отделения различных по назначению функциональных частей и возможности наращивания;
6) способность к самоорганизованности при нарушении целостности системы, при атаках на информационно-коммуникационные ресурсы сети или поражении системы ВПО;
7) динамическое формирование архитектуры из имеющихся частей рабочих КС сети;

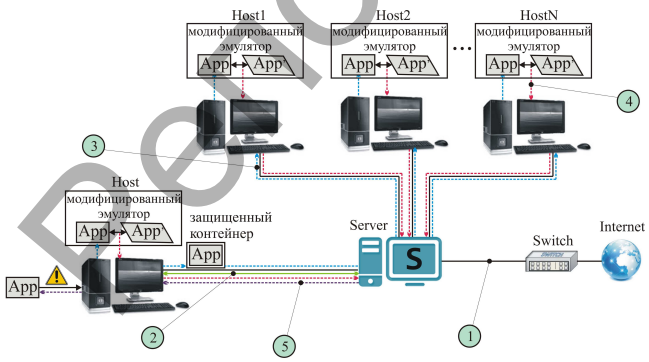
8) адаптивность к среде КС сети и специфики задач, решаемых конкретными КС;
9) возможность автоматизированной поддержки администратором сети для настройки и отчетности;
10) автономность системы в принятии решений;
11) соответствие политике безопасности предприятия (организации).

Соответствие системы заданным характеристикам повлияло на ее модель, которая стала основой архитектуры системы и, согласно характеристическим требованиям к системе, включает комбинацию моделей коллективного интеллекта многоагентных систем, распределенных систем, децентрализированных систем, самоорганизующихся и адаптивных систем. Учет такой модели повысит надежность функционирования и живучесть системы в локальной сети.

Распределенная многоуровневая сетевая система в локальной компьютерной сети представлена одинаковыми программными модулями, которые размещены в каждой компьютерной системе. То есть независимо от того сколько компьютерных систем в локальной сети, каждая из них содержит программный модуль РМСС. Если в процессе пользования компьютерными системами окажется, что не все они в сети включены, тогда РМСС состоит из тех, которые активны. Программные модули активных КС в локальной сети формируют непосредственно РМСС. Это позволит даже при наличии работы двух компьютерных систем поддерживать выполнение задач системы. Каждый программный модуль содержит на соответствующих уровнях функционал для выявления определенного типа вредоносного программного обеспечения атаки. РМСС содержит в каждом программном модуле уровень, который отвечает за коммуникацию между программными модулями всей системы, то есть обеспечивает работу каналов связи. С помощью этого уровня устанавливается связь между программными модулями системы в целом. Остальные уровни системы наполнены антивирусными функционалами. В частности, для выявления метаморфных вирусов создана подсистема, в которой реализована возможность их идентификации и выявления с привлечением всех хостов сети.

Подсистема обнаружения метаморфных вирусов в локальных компьютерных сетях. Использование сети продиктовано наличием у метаморфных вирусов, кроме обфускационных техник, антиэмуляционных средств, препятствующих осуществлению процесса эмуляции выполнения [6], что в свою очередь приводит к низкой эффективности обнаружения. Поэтому осуществление выявления метаморфных вирусов с высокой эффективностью, которые применяют антиэмуляционные технологии, средствами одной КС невозможно, в связи с чем предлагается привлечение возможностей локальной компьютерной сети.

Итак, рассмотрим локальную сеть, которая имеет топологию "звезда", управляемую одной организацией и предназначенную для удовлетворения ее производственных нужд, схема информационных потоков которой представлена на рис. 1.



Информационные потоки в сети:

1) физическое соединение;
2) запрос к серверу на наличие поведенческой сигнатуры;
3) рассылка и эмуляция подозрительной программы на всех хостах сети;
4) передача вектора признаков сходства подозрительной программы на сервер;
5) получения результата с сервера (блокировка / разрешение);

Рисунок 1 – Схема информационных потоков в локальной сети

Рассмотрим подробнее структуру и принцип функционирования подсистемы обнаружения метаморфных вирусов в локальной компьютерной сети.

Анализатор подозрительности программы. С целью выявления подозрительных действий на каждом хосте применяется анализатор подозрительности программы. Основной его задачей является отслеживание потока API вызовов, которые осуществляются в процессе выполнения неизвестной программой. В случае применения к программе техник обфускации программного кода, ее API вызовы остаются неизменными, видоизменяются только параметры и возвращаемого значения соответствующей функции [7].

Каждое отдельное подозрительное действие, представленное вызовом API функции, при выполнении неизвестной программы не опасно. Однако выполнение определенной последовательности таких действий может свидетельствовать о возможной опасности инфицирования вредоносным ПО, в частности метаморфным вирусом. Например, последовательность вызовов API функций Socket, connect, Get-SystemDirectory может свидетельствовать о потенциальной опасности, которая проявляется в создании нового соединения, в результате которого происходит доступ к системной директории.

С этой целью в основе анализатора подозрительности программы для определения программы как подозрительной заложена база эвристических сценариев, сгруппированных по уровням подозрительности – Surface, Average и Deep. Каждый сценарий представляется последовательностью API вызовов, соединенных с помощью логической операции И. Уровень подозрительности определяет условие, при котором происходит дальнейшее исследование программы и устанавливается при начальных настройках подсистемы обнаружения метаморфных вирусов (ПСОМВ). С целью анализа и динамического отслеживания API вызовов выполняемой программы с помощью программы API Monitor. Примеры сценария вредоносной активности, разделенные по уровням подозрительности, представлены в таблице 1.

Таблица 1 – Пример сценария подозрительного поведения программы

Уровень подозрительности	Сценарий подозрительного поведения
Запись в адресное пространство процесса (DLL Injection)	
Surface	OpenProcess ∧ VirtualAllocEx ∧ WriteProcessMemory ∧ GetModuleHandle
Average	OpenProcess ∧ VirtualAllocEx ∧ WriteProcessMemory ∧ GetModuleHandle ∧ ∧ GetProcAddress
Deep	OpenProcess ∧ VirtualAllocEx ∧ WriteProcessMemory ∧ GetModuleHandle ∧ ∧ GetProcAddress ∧ CreateRemoteThread

В случае активизации программой одного из сценариев происходит прекращение выполнения программы, маркировка ее как подозрительной программы и осуществление запроса к серверу на предмет наличия поведенческой сигнатуры для данной подозрительной программы.

На основе имеющейся на сервере базы потенциально опасных поведений осуществляется поиск соответствующего поведения для подозрительной программы.

В результате хосту, который получил подозрительную программу, может быть направлен один из трех результатов:
1) блокировать подозрительную программу, если соответствующее поведение присутствует в черном списке;
2) разрешить выполнение, если соответствующее поведение присутствует в белом списке;
3) выполнение дальнейшего исследования подозрительной программы.

В случае отсутствия подозрительной программы в черном и белом списке поведений производится дальнейшее исследование подозрительной программы на предмет наличия метаморфного вируса.

Модифицированный эмулятор. С целью генерации данных и во избежании распознавания вирусной программой выполнения в виртуальной среде создаются модифицированные эмуляторы [6], размещаемые на каждом хосте в сети.

Определим понятие "модифицированный эмулятор" как программное обеспечение для эмуляции аппаратного обеспечения рабочей станции, параметры и настройки которого изменяются при каждом новом запуске во избежание распознавания вирусной программой собственного выполнения в виртуальной среде.

Модифицированные эмуляторы являются составной частью подсистемы, основные функции которой запуск на выполнение подозрительной программы, ее дизассемблирование и дизассемблирование сформировавшейся копии подозрительной программы, формирования поведения подозрительной программы и внесение ее в базу поведений.

Для формирования переменной среды выполнения для каждого модифицированного эмулятора на каждом хосте сети предусматривается установление различных возможных параметров эмулятора: тип операционной системы, ее разрядность, MAC адрес гостевой ОС, скрытие процесса выполнения модифицированного эмулятора и защита процесса от чтения / записи в хостовой ОС, отключение модуля обмена данными между виртуальной машиной и хостовой ОС (Virtual Machine Communication Interface), изменение параметра ключа реестра в хостовой ОС HKEY_LOCAL_MACHINE\SYSTEM\ ControlSet001\Services\Disk\Enum и др.

Кроме того, с целью успешного распознавания и завершения процесса эмуляции инструкции, которые не представлены во множестве ассемблерных команд, заменяются на инструкцию NOP. Такая модификация позволит осуществить эмуляцию программы, даже если встретится неизвестная команда и, соответственно, не будет осуществлено аварийное завершение программы эмуляции. После завершения процесса эмуляции в сформированном дизассемблированном файле на месте NOP будет содержаться соответствующая команда, эмуляцию которой виртуальный процессор не смог выполнить.

В результате запуска подозрительной программы в модифицированном эмуляторе на сервер направляются следующие данные:

- образец дизассемблированного кода подозрительной программы в эмуляции (листинг операционных кодов);
- образец дизассемблированного кода подозрительной программы после эмуляции выполнения (листинг операционных кодов);
- поведение подозрительной программы (листинг API вызовов);
- непосредственно подозрительная программа.

Формирование вектора признаков сходства образца кода к метаморфному вирусу на основе определения и сравнения эквивалентных функциональных блоков. С целью оценки сходства подозрительной программы с метаморфным вирусом необходимо выделить характерные признаки, позволяющие идентифицировать метаморфный вирус. В качестве характеристических признаков определено расстояние Дамерау-Левенштейна, которое определяется как минимальное количество операций вставки, удаления, замены и транспозиции операционных кодов, необходимых для преобразования образца кода до эмуляции в образец кода после эмуляции и представляется в виде вектора признаков сходства образца кода к метаморфному вирусу.

Для получения указанных характеристических признаков осуществляется разбиение листингов операционных кодов до и после эмуляции (полученные в среде модифицированного эмулятора) на функциональные блоки (ФБ) и поиск между ними функциональных блоков, которые являются эквивалентными (ЭФБ), то есть имеют похожую семантику, но имеют разный синтаксис. Учитывая большие размеры современных исполняемых файлов, поиск эквивалентных функциональных блоков происходит не во всем выполняемом файле, а только в одной секции кода. Рассмотрим подробнее процесс определения эквивалентных функциональных блоков и формирования вектора признаков сходства образца кода с метаморфным вирусом.

Процесс определения ЭФБ состоит из двух этапов. Первый этап определяет ЭФБ на основные вычисления статистической оценки появления инструкций в блоке. На втором этапе осуществляется уточнение выбора ЭФБ и выбор наиболее подходящего блока, который будет использован для формирования вектора признаков сходства образца кода с метаморфным вирусом.

Подробное описание шагов метода представлено в [4, 6, 8].

С целью определения степени опасности поведения программы происходит сравнение поведения исследуемой программы с набором вредных поведенческих паттернов. Если между поведением подозрительной программы и одним из вредных паттернов есть сходство происходит совпадение, тогда признак вектора сходства копий метаморфного вируса приобретает значение подозрительности (high, medium и low), что соответствует группе вредных паттернов, с паттерном которого состоялось совпадение. Для поиска сходства поведенческих паттернов использован метод Бойера-Мура.

Таким образом, сформированный вектор признаков сходства образца кода с метаморфным вирусом состоит из количественных показателей, определяющих минимальное количество операций вставки, удаления, замены и транспозиции операционных кодов и логического признака – степени опасности подозрительной программы.

Привлечение локальной компьютерной сети и классификации вектора признаков сходства. Следующий этап предусматривает формирование результата степени сходства подозрительной программы и метаморфного вируса на основе анализа обфускации кода и поведения с использованием нечеткого классификатора. Если степень сходства с метаморфным вирусом получит значение High, то осуществляется блокировка подозрительной программы на хосте и добавление подозрительного поведения в черный список базы поведений. Если степень сходства с метаморфным вирусом подозрительной программы получит значение Low или Medium, то осуществляется рассылка подозрительной программы в защищенном контейнере и листинга операционных кодов программы к эмуляции на другие хосты в сети с целью их запуска в модифицированных эмуляторах. После выполнения эмуляции в модифицированных эмуляторах последний шаг предусматривает сбор сервером информации с хостов о поведении, предварительно разосланной подозрительной программы: образцов кода до и после эмуляции и поведения подозрительной программы. Далее происходит осуществление нечетким классификатором заключения о сходстве подозрительной программы с метаморфным вирусом. Если хотя бы на одном из хостов уровень сходства подозрительной программы с метаморфным вирусом High, то осуществляется блокировка подозрительной программы. Если уровень сходства – Low или Medium, то обеспечение разрешения на выполнение для этой программы и занесения ее в белый список программ.

Для формирования вывода об инфицировании системы метаморфным вирусом осуществляется классификация вектора признаков сходства образца кода с метаморфным вирусом средствами нечеткого логического вывода.

Система нечеткого логического вывода оперирует входными и выходными лингвистическими переменными. В качестве входных лингвистических переменных примем «степень сходства подозрительной программы с ее копией по дистанции Левенштейна» (L), «степень сходства подозрительной программы с ее копией по количеству операций вставки» (I), «степень сходства подозрительной программы с ее копией по количеству операций удаления» (D), «степень сходства подозрительной программы с ее копией по количеству операций замены» (R), «степень сходства подозрительной программы с ее копией по количеству операций перестановки» (T) и «степень сходства подозрительной программы с ее копией по количеству операций совпадения» (M). Исходной лингвистической переменной примем степень сходства с классом метаморфных вирусов (DSMV). Для каждой линвистической переменной зададим терм-множество Low, Medium и High. Для определения принадлежности подозрительной программы к метаморфному вирусу в системе задействовано 38 правил. Например, одно из правил имеет вид:

$$\begin{aligned} &\textit{if } (L \textit{ is Medium}) \textit{ and } (X \textit{ is High}) \textit{ and } (D \textit{ is Medium}) \text{ and} \\ &\textit{and } (I \textit{ is High}) \textit{ and } (R \textit{ is Low}) \textit{ and } (M \textit{ is Medium}) \textit{ and} \\ &\textit{and } (B_F \textit{ is High}) \textit{ then DSMV is High} \end{aligned} \quad (1)$$

В качестве функции принадлежности для входов была избрана трапециевидная, для выхода – треугольная. Результат работы системы нечеткого логического вывода представлен на рис. 2.

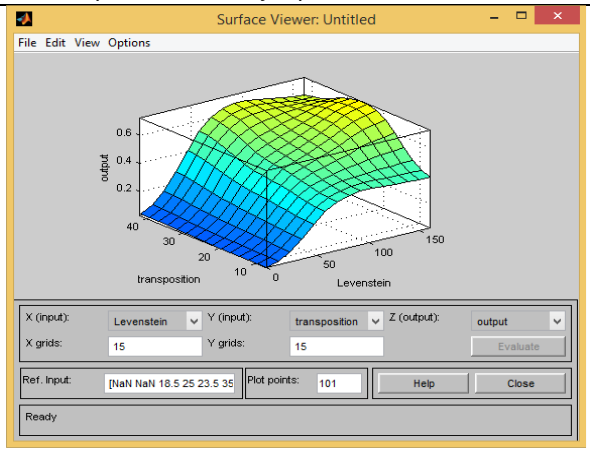


Рисунок 2 – Результат нечеткого логического вывода

Таким образом, для выявления метаморфных вирусов в локальной сети подсистема РМСС должна осуществить выполнение следующих этапов: 1) анализ подозрительности неизвестной программы; 2) запуск в модифицированном эмуляторе подозрительной программы; 3) дизассемблирование; 4) формирование поведения подозрительной программы; 5) отправки на сервер; 6) определение эквивалентных функциональных блоков; 7) определение степени опасности поведения подозрительной программы; 8) формирование вектора признаков сходства подозрительной программы с метаморфным вирусом; 9) осуществление нечеткой классификации; 10) анализ результатов; 11) рассылка на все хосты в локальной сети; 12) формирование заключения и отправки результата на хост, который получил подозрительную программу. Диаграмма последовательности выполнения этапов подсистемы обнаружения метаморфных вирусов в локальной сети с использованием модифицированных эмуляторов представлена на рис. 3.

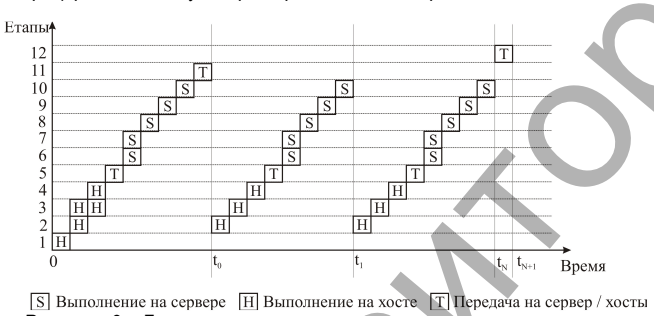


Рисунок 3 – Диаграмма последовательности выполнения этапов подсистемы обнаружения метаморфных вирусов в локальной сети с использованием модифицированных эмуляторов

Заключение. В работе представлена подсистема обнаружения метаморфных вирусов распределенной многоуровневой сетевой системы обнаружения вредоносного программного обеспечения в локальных компьютерных сетях.

Выявление осуществляется на основе отслеживания API вызовов, описывающих потенциально опасное поведение метаморфного вируса, и сравнению дизассемблированного кода функциональных блоков метаморфного вируса с кодом функциональных блоков его измененной версии. Для создания измененной версии метаморфного вируса на хостах сети устанавливаются модифицированные эмуляторы, которые обеспечивают переменную среду выполнения. С целью повышения общей эффективности выявления метаморфных вирусов система предусматривает поиск соответствия между функциональными блоками метаморфного вируса и его измененной версии. Для формирования вывода о сходстве подозрительной программы с метаморфным вирусом используется система нечеткого логического вывода. В случае недостаточного проявления вредоносного поведения и уровня достоверности для выявления метаморфного вируса привлекаются другие хосты сети.

Разрабонная подсистема РМС обнаружения метаморфных вирусов позволила повысить достоверность обнаружения новых и существующих метаморфных вирусов в сравнении с известными антивирусными информационными системами на 8-15%.

СПИСОК ЦИТИРОВАННЫХ ИСТОЧНИКОВ

1. Ransomware Damage Report [Electronic resource]. – Mode of access: https://cybersecurityventures.com/ ransomware-damage-report-2017-5-billion. – Title from the screen.
2. Thunga, S.P. Identifying metamorphic virus using n-grams and hidden markov model / S.P. Thunga, R. K. Neelisetti // Proceedings of Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference, 10–13 August, 2015: Kochi, India, 2015 – P. 2016–2022.
3. Ghiasi, M. Dynamic malware detection using registers values set analysis / M. Ghiasi, A. Sami, Z. Salehi // Proceedings of the 9th IEEE Information Security and Cryptology, 12–14 September, 2012: Tabriz, Iran, 2012 – P. 54–59.
4. Savenko, O. Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search / O. Savenko, S. Lysenko, A.Nicheporuk, B. Savenko // CEUR-WS. – 2017. – Vol. 1844. – P. 555–569.
5. Runwal, N. Opcode Graph Similarity and Metamorphic Detection / N. Runwal, R.M.Low, M. Stamp // Journal in Computer Virology. – 2012. – Vol. 8. – P. 37–52.
6. Komar, M. High performance adaptive system for cyber attacks detection / M. Komar, V. Kochan, L. Dubchak, A. Sachenko, V. Golovko, S. Bezobrazov, I. Romanets // Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 21-23 September, 2017: Bucharest, Romania, 2017. – Vol. 2. – P. 853–858.
7. Komar, M. Intelligent Cyber Defense System Using Artificial Neural Network and Immune System Techniques / M. Komar, A. Sachenko, S. Bezobrazov, V. Golovko // Communications in Computer and Information Science. – 2017. – Vol. 783. – P. 36–55.
8. Golovko, V. Evolution of Immune Detectors in Intelligent Security System for Malware Detection / V. Golovko, S. Bezobrazov, V. Melianchuk, M. Komar // Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2011), Prague (Czech Republic), 2011. – Vol. 2. – P. 722–726.
9. Branitskiy, A. Hybridization of computational intelligence methods for attack detection in computer networks / A. Branitskiy, I. Kotenko // Journal of Computational Science, Elsevier. – 2017. – № 23. – P. 145–156.
10. Котенко, И.В. Система сбора, хранения и обработки информации и событий безопасности на основе средств Elastic Stack / И.В. Котенко, А.А. Кулешов, И.А. Ушаков // Труды СПИИРАН. – 2017. – № 5(54). – С. 5–34.
11. Savenko, O. Approach for the Unknown Metamorphic Virus Detection / O. Savenko, S. Lysenko, A. Nicheporuk, B. Savenko // Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 21-23 September, 2017: Bucharest, Romania, 2017. – Vol. 1. – P. 453–769.
12. Patanaik, C.K. Obfuscated malware detection using API call dependency / C.K. Patanaik, F.A. Barbhuiya, S. Nandi // In Proc. the First International Conference on Security of Internet of Things, 2012: India. – P. 185–193.
13. Пат. на полезную модель 118456 Украина, МПК G06F 21/55 Способ выявления метаморфных вирусов на основе статистических метрик для определения эквивалентных функциональных программных блоков / О.С. Савенко, С.М. Лысенко, К.Ю. Бобровникова, А.О. Ничепорук, Б.О. Савенко; владелец Хмельницкий национальный университет. – № u201701743; заявл. 23.02.2017; опубл. 10.08.2017, Бюл. № 15/2017 (на украинском языке).

Материал поступил в редакцию 16.01.2018