МЕТОД ОБЕСПЕЧЕНИЯ УСТОЙЧИВОСТИ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА В УСЛОВИЯХ ВОЗДЕЙСТВИЯ ВРЕДОНОСНЫХ ПРОГРАММ

Мирзабаев А.Н.[1], Самонов А.В.[2]

Цель исследования: *разработка метода и средств обеспечения устойчивого функционирования программного обеспечения критически важных информационных систем в условиях воздействия на них вредоносного программного обеспечения.*

Метод исследования: *анализ и классификация вредоносного программного обеспечения и средств защиты от него, синтез и моделирование корректного поведения программ, временные автоматы, мониторинг текущего состояния процесса.*

Результат исследования: *дана характеристика методов и средств обнаружения вредоносных программ, использующих методы маскировки, руткит-механизмы и технологии аппаратной виртуализации. Разработана методика построения профилей корректного функционирования контролируемых программ в виде совокупности допустимых трасс исполнения, содержащих значения маршрутно-временных параметров вычислительного процесса в его наиболее информационных контрольных точках. Разработан метод мониторинга и контроля корректности текущего состояния вычислительного процесса посредством его сравнения с эталонными профилями. Метод позволяет в режиме реального времени обнаруживать и противодействовать вредоносным программам, использующим различные техники внедрения и маскировки, включая руткит-механизмы, гипервизоры на основе технологии аппаратной виртуализации, перехват и внедрение в системные функции в оперативной памяти.*

Ключевые слова: *гипервизоры, профиль корректного функционирования, руткит-механизмы, технологии аппаратной виртуализации.*

DOI:10.21681/2311-3456-2022-2-63-71

Введение

Современный этап развития общественных отношений и экономических укладов характеризуется активным внедрением информационных технологий во все сферы жизнедеятельности, создавая новые угрозы и высокие риски в области обеспечения информационной безопасности (ИБ). Число компьютерных преступлений за последние пять лет выросло с 65 тысяч до 510 тысяч. При этом число особо опасных атак на объекты критической информационной инфраструктуры (КИИ) РФ в 2020 году по сравнению с 2019 годом выросло в 3,5 раза. Технологии создания и применения средств реализации атак непрерывно развиваются и совершенствуются, существенно снижая эффективность существующих методов и средств обеспечения ИБ. Примерами таких технологий являются руткит-механизмы, технологии аппаратной виртуализации, техники маскировки и обмана. Для защиты от этих угроз требуются новые подходы, методы и средства, разработка которых осуществляется в рамках создания отечественной доверенной аппаратной и программной платформы (ДП) [1, 2]. Основными элементами ДП являются: аппаратные средства, операционные системы (ОС), гипервизоры, средства доверенной загрузки, антивирусы, системы обнаружения и предотвращения атак.

В статье представлено описание методов и средств создания одного из компонентов доверенной платформы, который должен обеспечить опера-

1 Мирзабаев Алишер Нигматджонович, начальник отдела Военно-космической академии имени А.Ф. Можайского, Санкт-Петербург, Россия. E-mail: ali_mir73@mail.ru

2 Самонов Александр Валерьянович, кандидат техническихнаук, доцент, старший научный сотрудник Военно-космической академии имени А.Ф. Можайского, Санкт-Петербург, Россия. E-mail: a.samonov@mail.ru ORCID: 0000-0002-0390-4481,

тивный контроль корректности функционирования ИС КИИ в условиях воздействия на них вредоносного программного обеспечения (ВПО). В первом разделе дан краткий обзор методов и средств обнаружения и противодействия современным типам ВПО и сформулирована постановка задачи исследования. Во втором разделе представлена методика разработки профилей корректного функционирования контролируемых программ. В третьем разделе описан алгоритм реализации метода мониторинга и контроля корректности текущего состояния вычислительного процесса посредством его сравнения с эталонными профилями. В заключении приведены отличительные особенности и достоинства предложенного подхода, определены направления развития и совершенствования представленных в статье методов и средств.

1. Обзор методов и средств обнаружения и противодействия современным видам вредоносного программного обеспечения

Бурное развитие информационных и телекоммуникационных технологий и систем сопровождается столь же бурным развитием технологий и средств нарушения информационной безопасности, направленных на совершение киберпреступлений и ведение кибервойн. Ярким подтверждением этого утверждения является применение для создания ВПО руткит-механизмов и технологий аппаратной виртуализации.

Классификация руткит-механизмов, разработанная на основе анализа исследований [3 - 9], представлена на рис.1.

Все руткит-механизмы можно разделить на четыре группы:

1) руткит-механизмы внедрения и функционирования, работающие внутри или вне операционной системы [6, 10 - 13];
2) механизмы противодействия средствам эмуляции (anti emulation) и динамического анализа [14 - 17];
3) механизмы мутации кода (code mutation) [6];
4) механизмы таргетирования (target mechanism) [3, 6].

Руткит-механизмы, работающие внутри ОС, делятся на две группы. Механизмы первой группы изменяют пути выполнения программ, а вторые изменяют системные структуры данных. Наиболее сложным для обнаружения является ВПО, использующее технологии перехвата и внедрения в системные функции непосредственно в оперативной памяти. Механизмы, работающие вне ОС, используют технологии виртуализации (программной или аппаратной) [5, 11, 12], режим системного управления (SMM) [13] и дополнительные аппаратные средства [11, 14, 15].

Для маскировки и противодействия средствам эмуляции и динамического анализа используются методы, представленные на рисунке 2. Они делятся на две группы: методы противодействия ручному анализу (anti-debugger) и автоматическому анализу в «песочницах» (anti-sandboxes) [17].

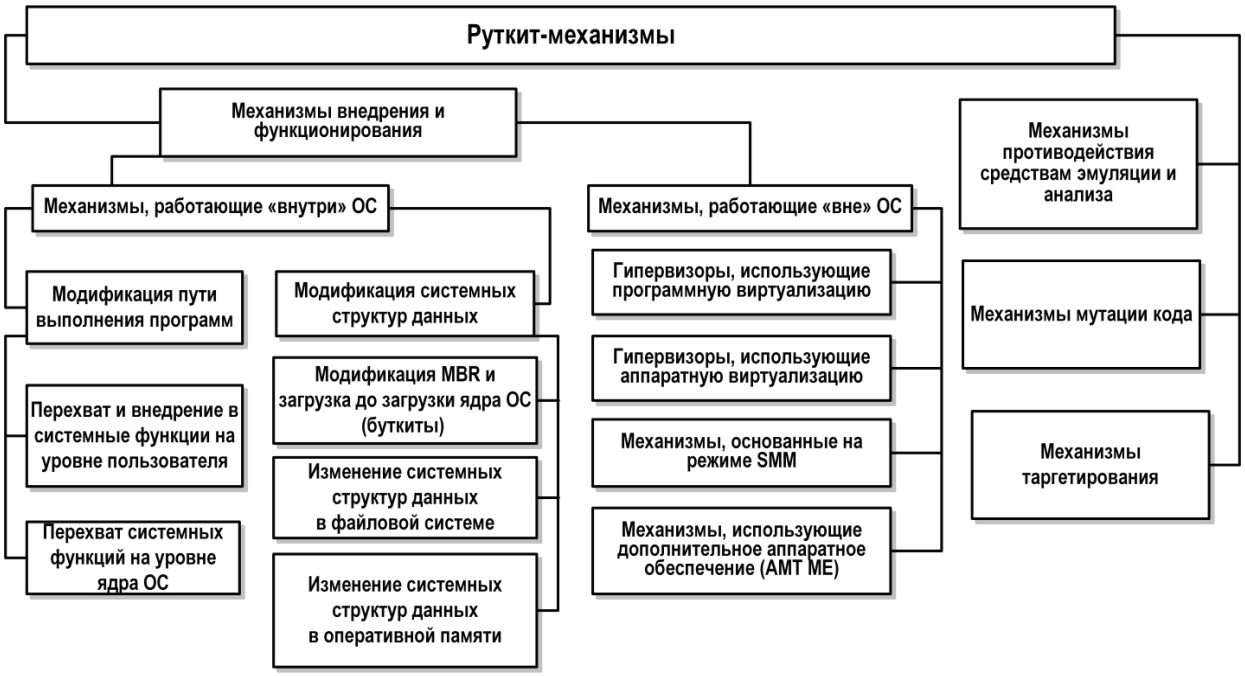

Рис. 1. Классификация руткит-механизмов

Рис. 2. Классификация методов противодействия средствам динамического анализа ВПО

Для обнаружения и противодействия ВПО создаются и применяются следующие методы и инструменты: общесистемные отладчики, антивирусы, антируткиты, гипервизоры на основе аппаратной виртуализации, счетчики производительности оборудования, инструменты оперативной криминалистики, аппаратно-программные решения. Примерами инструментов на основе привилегированных колец являются: HVM, SMM, AMT, SGX, GPU, TSX. Примерами инструментов, основанных на счетчиках производительности, являются: Branch Trace, CFIMon, KBouncer, ROPecker, Pierce, HPCHunter. Примерами аппаратных средств являются Copilot SnapMon Ki-Mon. Подробное описание данных методов и инструментов представлено в [14, 15, 17].

Основными способами обнаружения ВПО, использующего технологии аппаратной виртуализации, являются: временные, поведенческие, сигнатурные и на основе доверенного гипервизора [15, 18]. Как отмечено в [18], они имеют следующие ограничения и недостатки:

— временные способы не позволяют выявить гипервизоры в случае использования компрометации счётчика тактов или временной выгрузки гипервизора из памяти;
— поведенческие способы могут обнаруживать только известные им гипервизоры и не работоспособны на новых моделях процессоров;
— способы на основе доверенного монитора виртуальных машин уязвимы к атаке «человек-посередине» («Man-In-The-Middle»);
— аппаратные средства являются узкоспециализированными.

Анализ возможностей используемых в настоящее время методов и средств обнаружения и противодействия ВПО, использующего современные технологии внедрения и скрытного функционирования, показал их ограниченность и низкую результативность. Для преодоления этих ограничений и недостатков целесообразно:

— использовать методы обнаружения аномального поведения программ на основе сравнения реализуемого маршрута исполнения с допустимым, т.е. корректным;
— мониторинг процессов выполнения программ и контроль корректности вычислительного процесса должен быть реализован доверенным гипервизором.

Постановку задачи на разработку метода и средств обнаружения и противодействия ВПО сформулируем следующим образом.

Необходимо:

1) для обоснования контрольных точек на маршрутах исполнения программ определить типовые признаки функционирования ВПО, использующего рут-кит-средства и технологии аппаратной виртуализации, и ситуации, в которых они проявляются;

2) разработать метод и средства построения профилей корректного функционирования контролируемых программ $Ma_j(P_j) = \{Ta_i\}$, $i=1,...,I$, где Ta_i – i-я корректная трасса (маршрут) исполнения программы P_j;

3) разработать метод контроля корректности выполнения программ посредством мониторинга маршрутно-временных параметров вычислительного процесса в условиях воздействия вредоносного программного обеспечения.

Разработанные на основе этих моделей и методов средства должны обладать следующими возможностями:

— обнаруживать нелегитимные гипервизоры, использующие компрометацию счетчика тактов, временную выгрузку из памяти и другие техники маскировки;
— противостоять атаке «человек-посередине» («Man-In-The-Middle»);
— обнаруживать признаки ВПО, функционирующего на уровне ядра ОС и использующего методы маскировки и обмана средств их обнаружения.
— обеспечить низкий уровень ошибок 1 и 2 рода.

В следующих разделах статьи представлены результаты решения данной задачи.

2. Методика разработки профилей корректного функционирования программ

Для разработки профилей корректного функционирования программ целесообразно использовать математический аппарат временных автоматов вида: $TA = (CP, cp_0, E, L, clocks, guard, Id)$, где

CP – конечное множество допустимых состояний вычислительного процесса с начальным состоянием $cp_0 \ni CP$,

E – множество ребер, каждое из которых снабжено пометками src (e) и $dest$ (e), указывающими из какого состояния исходит данное ребро и в какое состояние оно входит,

$L: CP \rightarrow 2^{AP}$ – функция, которая сопоставляет каждому состоянию $cp_i \ni CP$ подмножество атомарных предложений AP,

$clocks: E \rightarrow 2^C$ – функция, которая сопоставляет каждому ребру $e_i \ni E$ интервалы времени $clocks(e)$,

$guard: E \rightarrow \Psi(C)$ – функция, которая помечает каждое ребро $e_i \ni E$ условием выполнения перехода по ребру e,

Id - уникальные идентификаторы программы в контрольных точках.

Процесс выполнения программы моделируется посредством система переходов, ассоциированной с временным автоматом: $Ta_i = (CP, cp_0 \rightarrow)$.

В отличие от представлений временного автомата, используемых в [19 - 21], в нашем случае исключена функция для инвариантов автомата, добавлен атрибут Id, позиции интерпретируются как состояния программы в контрольных точках.

Профили корректного функционирования разрабатываются для программ, включенных в состав замкнутой программной среды защищаемого ресурса (сервера, хранилища данных, клиентского рабочего места). Разработка допустимых маршрутов корректного исполнения программы включает:

— определение наиболее информационных контрольных точек;
— построение профилей корректного функционирования программы в виде набора допустимых маршрутов.

В качестве наиболее информационных контрольных точек в модели исполнения программы, которые целесообразно использовать для формирования допустимых маршрутов ее исполнения, целесообразно использовать [5, 12, 18, 22, 23]:

— системные вызовы;
— инструкции управления виртуализацией;
— управляющие структуры языков программирования (*if-then-else, do while, for while, switch case, ...*).

Примерами системных вызовов ОС Windows являются: *CallNextHookEx, CreateRemoteThread, DeviceIoControl, DllMain, FindFirstFile, FindNextFile, GetProcAddress, LoadLibrary, NtDelayExecution, NTQuerySystemInformation, OpenProcess, PsSetLoadImageNotifyRoutine, PspCreateProcessNotify, SetWindowsHookEx, SleepEx, WriteProcessMemory, ZwQuerySystemInformation*.

Примерами системных вызовов ОС Linux (Unix) являются: *fork(), exec(), open(), create(), delete(), malloc(), calloc(), realloc() и free(),socket(), close(), wait()*.

Основными инструкциями, используемыми средствами виртуализации, являются: *VMRUN, VMSAVE, VMLOAD, VMMCALL, CPUID, SKINIT*.

Множество контролируемых программ обозначим: $P = \{P_j\}$ $j=1,..., J$.

Модель функционирования программы P_j представляет собой множество допустимых маршрутов исполнения программы:

$Ma_j(P_j) = \{Ta_k\}$, $k=1,...,K$.

Каждый маршрут описывается следующим образом:

$$Ta_k = \{(cp_{ik}, dt_{ik}, id_{ik})\}, i=1 \dots I, k=1,...,K,$$

где cp_{ik} – контрольная точка под номером i на k-ом допустимом маршруте;

dt_{ik} – интервал времени между cp_{ik-1} и cp_{ik}, измеряемый в тактах центрального процессора;

```
var quick_sort = function(array){
  function partition(array, lo, hi){
   var i = lo, j = hi + 1;   while(true)
{
      while(array[++i] < array[lo]){
        if ( i == hi ) break;
       }       while (array[ - j] >
array[lo]){
        if ( j == lo ) break;
        }        if (i >= j) break;
array.swap(i,j);
    }    array.swap(lo,j);
    return j;
  }  function sort(array, lo, hi){
     if (hi <= lo) return;     var
j   =   partition(array,   lo,   hi);
sort(array, lo, j-1);
    sort(array, j+1, hi);
   }   sort(array, 0, array.length —
1);
  return array;
}
```

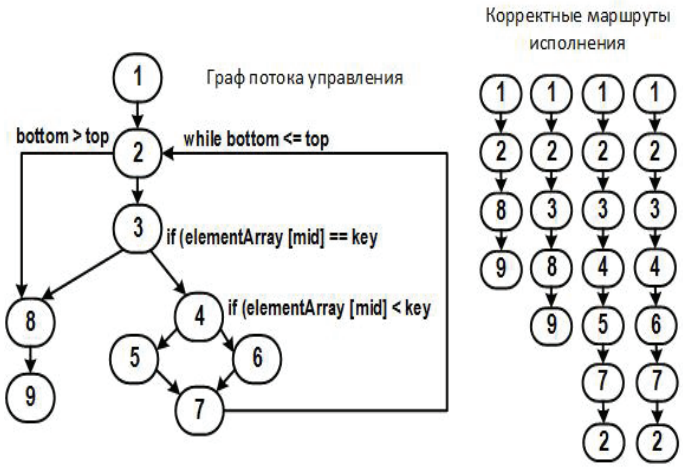

Рис. 3. Пример построение эталонного профиля для контролируемой программы

id_{ik} – уникальный идентификатор программы в контрольной точке *i*, принадлежащей *k*-му допустимому маршруту.

Начальное значение уникальному идентификатору программы рассчитывается в момент запуска программы: $id_{j0} = hash(rand())$. В процессе исполнения программы уникальный идентификатор программы в очередной контрольной точке рассчитывается по формуле $id_{jk} = hash(rand(id_{jk-1}))$. Он используется для подтверждения подлинности (аутентификации) контролируемого процесса и противодействия возможной компрометации со стороны ВПО.

При расчете временных интервалов между контрольными точками необходимо учитывать следующие характеристики аппаратно-программных платформ:

- характеристики ЦПУ: количество процессоров и ядер, тактовая частота, размер кэш-памяти;
- размеры оперативной памяти;
- механизм предварительной выборки Prefetch;
- буферы внеочередного выполнения (Out of Order Buffers);
- алгоритмы неупорядоченного исполнения (OOO execution) и предсказания ветвлений (Branch Prediction);
- механизмы управления тактовой частотой ЦПУ.

На рис. 3 представлен пример, иллюстрирующий построение эталонного профиля для контролируемой программы. Фрагмент кода контролируемой программы представлен в левой части рисунка. В средней части рисунка представлена модель программы в виде графа потока управления с заданными в нем контрольными точками (1-9). В правой части рисунка представлен профиль, состоящий из четырех допустимых маршрутов.

Разработанные профили корректного функционирования программ используются в качестве эталонов для сравнения с текущими маршрутно-временными характеристиками вычислительного процесса для обнаружения активности ВПО. Для обнаружения и предотвращения нелегитимного использования системных функций в системный диспетчер ОС устанавливается специальная программная закладка, информирующая сервис контроля корректности вычислительного процесса обо всех случаях обращения к системным функциям.

3. Метод контроля корректного выполнения программ посредством мониторинга маршрутно-временных параметров вычислительного процесса в условиях воздействия вредоносного программного обеспечения

Для реализации данного метода необходимо выполнить следующие предварительные операции.

1. Для всех контролируемых программ разрабатываются и записываются в базу данных сервиса контроля корректности *эталонные профили их корректного выполнения*.

2. В контрольных точках cp_{ik} контролируемых программ и в диспетчер системных функций устанавливаются специальные *программные закладки*, с помощью которых будет осуществляться их взаимодействие с сервисом контроля корректности. Установка

производится с помощью специальным образом модифицированного компилятора.

3. Сервис контроля корректности настраивается таким образом, чтобы он запускался в виде гипервизора до загрузки операционной системы и находился в режиме ожидания сообщений от контролируемых программ.

Реализация метода мониторинга и контроля корректности вычислительного процесса посредством сравнения маршрута реального исполнения программы с эталонными профилями их поведения включает выполнение следующих операций (рис.4):

1) При запуске процесса выполнения программы P_j осуществляется:

— расчет начального значения уникального идентификатора $id_{j0} = hash(rand())$;

— передача сервису контроля корректности сообщения о запуске программы $Pj\ send(Pj, id_{j0})$.

2) После получения от контролируемой программы сообщения сервис контроля корректности функционирования программ осуществляет выбор эталонного профиля для программы P_j – $Ma_j = \{Ta_{jk}\}$ и запускает цикл мониторинга и анализа состояний процесса выполнения программы.

3) При достижении очередной контрольной точки программа информирует об этом сервис контроля, передавая ему сообщение, содержащее информацию о текущей контрольной точке $(cp^r_{ik}, dt^r_{ik}, id^r_{ik})$.

4) Сервис контроля осуществляет проверку соответствия реального маршрута эталонному (одному из допустимых), учитывая номера контрольных точек, временные интервалы между ними и значение уникального идентификатора программы в данной контрольной точке.

5) В случае установления не соответствия – фиксируется нарушение корректного функционирования программы и выполнение процесса прекращается.

Основными признаками ВПО являются:

— попытки выполнения программы по неразрешенному маршруту, включая отсутствующие в эталонных маршрутах вызовы системных функций;

— превышение допустимого интервала времени перехода между контрольными точками;

— вызовы системных функций и инструкций управления виртуализацией из программ, не включенных в замкнутую программную среду.

В приведенном на рисунке 4 примере представлены два случая реализации данного метода. При выполнении программы по маршруту 1 обнаруживается нарушение корректного функционирования программы при переходе в контрольную точку под номером 5 (выделено красным цветом). Выполнение программы по маршруту 2 является корректным (все контрольные точки имеют зеленый цвет).

Заключение

Представленный в данной статье подход к обеспечению устойчивости функционирования ИС КИИ в условиях воздействия ВПО имеет следующие отличительные особенности и преимущества перед существующими методами и средствами.

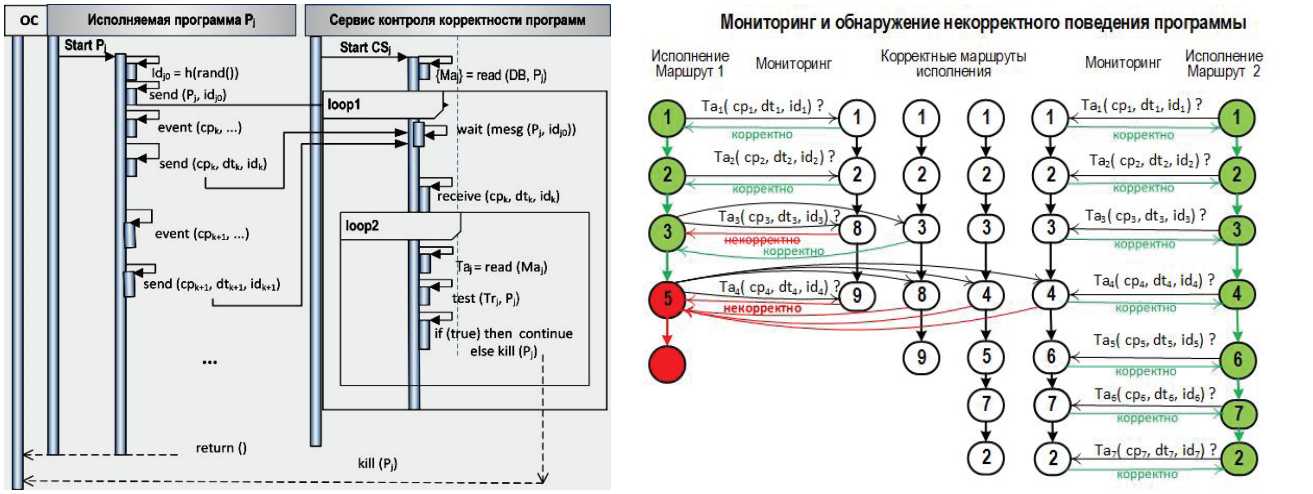

Рис. 4. Схема, иллюстрирующая применение разработанного метода для контроля корректности функционирования программы

Профили корректного функционирования программ, использующиеся в качестве эталонов для принятия решений, представляют собой допустимые маршруты выполнения программы в виде упорядоченных на относительной шкале времени состояний вычислительного процесса в контрольных точках. Контрольные точки устанавливаются в наиболее информативных с точки зрения возможности обнаружения признаков ВПО узлах контролируемых программ. Состояния программы в контрольных точках характеризуются следующими параметрами: номером состояния на маршруте, интервалом времени от предыдущего состояния и уникальным идентификатором программы в контрольной точке.

Представленный в статье метод контроля корректности функционирования программ позволяет:

— обнаруживать нелегитимные гипервизоры, использующие компрометацию счетчика тактов, временную выгрузку из памяти и другие техники маскировки;
— противостоять атаке «человек-посередине» («Man-In-The-Middle»);
— обнаруживать признаки ВПО, функционирующего на уровне ядра ОС и использующего методы маскировки и обмана средств их обнаружения.

Направлениями дальнейших исследований являются: разработка метода и средств автоматизированного формирования рационального состава контрольных точек на маршрутах исполнения программ в типовом программном окружении ОС Linux, а также реализация рассмотренного в статье метода и алгоритмов в виде гипервизора, использующего технологию аппаратной виртуализации

Литература

1. Михалевич И. Ф. Требования, принципы, практика создания отечественных аппаратно-программных платформ для автоматизированных систем в защищенном исполнении критической информационной инфраструктуры Российской Федерации. // Интеллектуальные системы. Теория и приложения. – 2018. – Т.22. Вып.4. С.11–30.
2. Борисов, А.Л. Анализ подхода к созданию доверенных программно-аппаратных платформ для органов государственного управления, силовых министерств и ведомств / А.Л. Борисов, Ю.В. Соснин, А.Л. Оружейников // Охрана, безопасность, связь. – 2016. – № 1-2. – С. 61-65.
3. Botacin, Marcus & De Geus, Paulo & Grégio, André. (2019). “VANILLA” malware: vanishing antiviruses by interleaving layers and layers of attacks. Journal of Computer Virology and Hacking Techniques. 15. 10.1007/s11416-019-00333-y (дата обращения: 30.09.2021).
4. Сахаров Д.В., Ковцур М.М., Бахтин Д.В. Модель защиты от эксплойтов и руткитов с последующим анализом и оценкой инцидентов // Наукоемкие технологии в космических исследованиях Земли. 2019. Т. 11. № 5. С. 22–31. doi: 10.24411/2409-5419-2018-10284.
5. Botacin, Marcus & De Geus, Paulo & Grégio, André. Who Watches the Watchmen: A Security-focused Review on Current State-of-the-art Techniques, Tools, and Methods for Systems and Binary Analysis on Modern Platforms. //ACM Computing SurveysVolume 51 Issue 4 September 2018. Article No.: 69 pp 1–34. URL: https://dl.acm.org/doi/abs/10.1145/3199673 (дата обращения: 30.09.2021).
6. A Survey of Stealth Malware Attacks, Mitigation Measures, and Steps Toward Autonomous Open World Solutions. Ethan M. Rudd, Andras Rozsa, Manuel Günther, and Terrance E. Boul. URL: https://arxiv.org/pdf/1603.06028.pdf. (дата обращения: 30.09.2021).
7. O.L. Fraser, N. Zincir-Heywood, M. Heywood, and J.T. Jacobs. Return-oriented programme evolution with ROPER: a proof of concept. In Proc. of the Genetic and Evolutionary Computation Conference Companion, 2017, pp. 1447–1454.
8. N.R. Weidler, D. Brown, S.A. Mitchell, J. Anderson, J.R. Williams, A. Costley, C. Kunz, C. Wilkinson, R. Wehbe, and R. Gerdes. Return-oriented programming on a resource constrained device. Sustainable Computing: Informatics and Systems, vol. 22, 2019, pp. 244-256.
9. Countering Persistent Kernel Rootkits Through Systematic Hook Discovery. /Zhi Wang, Xuxian Jiang, Weidong Cui, Xinyuan Wang. // North Carolina State University Microsoft Research George Mason University. URL: https://www.csc2.ncsu.edu/faculty/xjiang4/pubs/RAID08_HookMap.pdf (дата обращения 24.09.2021).
10. Вишняков А.В., Нурмухаметов А.Р. Обзор методов автоматизированной генерации эксплойтов повторного использования кода. Труды ИСП РАН, том 31, вып. 6, 2019 г., стр. 99–124.
11. Gabor Pek. New Methods for Detecting Malware Infections and New Attacks against Hardware Virtualization Ph.D. Dissertation. URL: https://repozitorium.omikk.bme.hu/bitstream/handle/10890/1409/ertekezes.pdf (дата обращения 23.09.2021).
12. Фурсова Н.И. Методы мониторинга объектов операционной системы, выполняющейся в виртуальной машине. Диссертация на соискание учёной степени кандидата технических наук. Великий Новгород — 2017. 120 с.
13. William Augusto Rodrigues de Souza. On Using the System Management Mode for Security Purposes. Department of Mathematics Royal Holloway, University of London. URL: https://pure.royalholloway.ac.uk/portal/files/28250097/Thesis.pdf (дата обращения 23.09.2021).
14. Zhang, Fengwei & Leach, Kevin & Stavrou, Angelos & Wang, Haining & Sun, Kun. (2015). Using Hardware Features for Increased Debugging Transparency. 2015. 55-69. 10.1109/SP.2015.11.

15. Botacin M. F., Hardware-assisted malware analysis / Marcus Felipe Botacin. – Campinas, SP: [s.n.], 2017. URL: https://www.lasca.ic.unicamp.br/paulo/teses/20170728-MSc-Marcus.Felipe.Botacin-Hardware.Assisted.Malware.Analysis.pdf (дата обращения 23.09.2021).
16. E. J. Schwartz, T. Avgerinos, and D. Brumley. All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask). In Proc. of the IEEE Symposium on Security and Privacy, 2019, pp. 317– 331.
17. Malware Dynamic Analysis Evasion Techniques: A Survey Amir Aanian, Salman Niksefat, Babak Sadeghiyan, and David Baptiste. URL:https://www.researchgate.net/publication/328758559_Malware_Dynamic_Analysis_Evasion_Techniques_A_Survey (дата обращения 24.09.2021).
18. Жуков, А. Е. Модели выполнения процессорных инструкций в условиях противодействия со стороны нарушителя для компьютерных систем с поддержкой технологии аппаратной виртуализации / А. Е. Жуков, И. Ю. Коркин, Б. М. Сухинин // Безопасность информационных технологий. – 2012. – Т. 19. – № 2. – С. 85-89.
19. Вельдер С.Э., Лукин М. А., Шалыто А. А., Яминов Б. Р. Верификация автоматных программ. СПб: Наука, 2011. 244 с.
20. Твардовский А.С., Лапутенко А.В. О возможностях автоматного описания параллельной композиции временных автоматов // Труды Института системного программирования РАН. 2018; 30(1):25-40. – URL: https://doi.org/10.15514/ISPRAS-2018-30(1)-2 (дата обращения 24.09.2021).
21. Быковский С.В. Метод встроенной динамической актуализации функциональных моделей систем на кристалле // Известия высших учебных заведений. Приборостроение. 2015. Т. 58, № 3. С. 197-202.
22. Белоус, А. Доверенная ЭКБ для доверенных аппаратно-программных платформ: проблемы и пути решения. Часть 1 // Электроника: Наука, технология, бизнес. – 2021. – № 3(204). – С. 98-104. – DOI 10.22184/1992-4178.2021.204.3.98.104.
23. Unvelling the kernel: rootkit discovery using select automated kernel memory differencing. A. Zaki, Benjamin Humphrey Sophos, UK. URL: https://www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-ZakiHumphrey.pdf (дата обращения 24.09.2021).

CONTROL METHOD OF THE CORRECT EXECUTION OF PROGRAMS BY MONITORING AND ANALYZING THE ROUTE-TIME PARAMETERS OF THE COMPUTING PROCESS

Mirzabaev A.N[3], Samonov A.V.[4]

Objective: *develop a method and means to ensure sustainable functioning of the software mission-critical information systems under impact malicious software.*

Methods: *analysis and classification of malicious software and means of protection against it, synthesis and modeling of correct behavior of programs, temporary automata.*

Study results: *the characteristic of methods and means of detecting malware, which using masking methods, rootkit mechanisms and hardware virtualization technologies is given. A methodology for constructing profiles of the correct functioning of controlled programs in the form of a set of permissible execution routes has been developed. A method for monitoring and controlling the correctness of the current state of the computing process by comparing it with reference profiles has been developed. The method allows real-time detection and counteraction of malicious programs, which using various techniques of implementation and masking, including rootkit mechanisms, hypervisors based on hardware virtualization technology, interception and introduction in system functions in RAM.*

Keywords: *malware, hypervisors, correct functioning profile, rootkit mechanisms, hardware virtualization technologies.*

3 Alisher N. Mirzabaev, Head of Department, Mozhaiskiy Military Space Academy, Saint Petersburg, Russia. E-mail: ali_mir73@mail.ru

4 Alexander V. Samonov, Ph.D. (in tech.), Associate Professor, senior research scientist Mozhaiskiy Military Space Academy, Saint Petersburg, Russia. E-mail: a.samonov@mail.ru

References

1. Mihalevich I. F. Trebovanija, principy, praktika sozdanija otechestvennyh apparatno-programmnyh platform dlja avtomatizirovannyh sistem v zashhishhennom ispolnenii kriticheskoj informacionnoj infrastruktury Rossijskoj Federacii. // Intellektual'nye sistemy. Teorija i prilozhenija. – 2018. – T.22. Vyp.4. S.11–30.
2. Borisov, A.L. Analiz podhoda k sozdaniju doverennyh programmno-apparatnyh platform dlja organov gosudarstvennogo upravlenija, silovyh ministerstv i vedomstv / A.L. Borisov, Ju.V. Sosnin, A.L. Oruzhejnikov // Ohrana, bezopasnost', svjaz'. – 2016. – № 1-2. – S. 61-65.
3. Botacin, Marcus & De Geus, Paulo & Grégio, André. (2019). "VANILLA" malware: vanishing antiviruses by interleaving layers and layers of attacks. Journal of Computer Virology and Hacking Techniques. 15. 10.1007/s11416-019-00333-y (data obrashhenija: 30.09.2021).
4. Saharov D.V., Kovcur M.M., Bahtin D.V. Model' zashhity ot jeksplojtov i rutkitov s posledujushhim analizom i ocenkoj incidentov // Naukoemkie tehnologii v kosmicheskih issledovanijah Zemli. 2019. T. 11. № 5. S. 22–31. doi: 10.24411/2409-5419-2018-10284.
5. Botacin, Marcus & De Geus, Paulo & Grégio, André. Who Watches the Watchmen: A Security-focused Review on Current State-of-the-art Techniques, Tools, and Methods for Systems and Binary Analysis on Modern Platforms. //ACM Computing SurveysVolume 51 Issue 4 September 2018. Article No.: 69 pp 1–34. URL: https://dl.acm.org/doi/abs/10.1145/3199673 (data obrashhenija: 30.09.2021).
6. A Survey of Stealth Malware Attacks, Mitigation Measures, and Steps Toward Autonomous Open World Solutions. Ethan M. Rudd, Andras Rozsa, Manuel Günther, and Terrance E. Boul. URL: https://arxiv.org/pdf/1603.06028.pdf. (data obrashhenija: 30.09.2021).
7. O.L. Fraser, N. Zincir-Heywood, M. Heywood, and J.T. Jacobs. Return-oriented programme evolution with ROPER: a proof of concept. In Proc. of the Genetic and Evolutionary Computation Conference Companion, 2017, pp. 1447–1454.
8. N.R. Weidler, D. Brown, S.A. Mitchell, J. Anderson, J.R. Williams, A. Costley, C. Kunz, C. Wilkinson, R. Wehbe, and R. Gerdes. Return-oriented programming on a resource constrained device. Sustainable Computing: Informatics and Systems, vol. 22, 2019, pp. 244-256.
9. Countering Persistent Kernel Rootkits Through Systematic Hook Discovery. /Zhi Wang, Xuxian Jiang, Weidong Cui, Xinyuan Wang. // North Carolina State University Microsoft Research George Mason University. URL: https://www.csc2.ncsu.edu/faculty/xjiang4/pubs/RAID08_HookMap.pdf (data obrashhenija 24.09.2021).
10. Vishnjakov A.V., Nurmuhametov A.R. Obzor metodov avtomatizirovannoj generacii jeksplojtov povtornogo ispol'zovanija koda. Trudy ISP RAN, tom 31, vyp. 6, 2019 g., str. 99–124.
11. Gabor Pek. New Methods for Detecting Malware Infections and New Attacks against Hardware Virtualization Ph.D. Dissertation URL: https://repozitorium.omikk.bme.hu/bitstream/handle/10890/1409/ertekezes.pdf (data obrashhenija 23.09.2021).
12. Fursova N.I. Metody monitoringa ob#ektov operacionnoj sistemy, vypolnjajushhejsja v virtual'noj mashine. Dissertacija na soiskanie uchjonoj stepeni kandidata tehnicheskih nauk. Velikij Novgorod — 2017. 120 s.
13. William Augusto Rodrigues de Souza. On Using the System Management Mode for Security Purposes. Department of Mathematics Royal Holloway, University of London. URL: https://pure.royalholloway.ac.uk/portal/files/28250097/Thesis.pdf (data obrashhenija 23.09.2021).
14. Zhang, Fengwei & Leach, Kevin & Stavrou, Angelos & Wang, Haining & Sun, Kun. (2015). Using Hardware Features for Increased Debugging Transparency. 2015. 55-69. 10.1109/SP.2015.11.
15. Botacin M. F., Hardware-assisted malware analysis / Marcus Felipe Botacin. – Campinas, SP: [s.n.], 2017. URL: https://www.lasca.ic.unicamp.br/paulo/teses/20170728-MSc-Marcus.Felipe.Botacin-Hardware.Assisted.Malware.Analysis.pdf (data obrashhenija 23.09.2021).
16. E. J. Schwartz, T. Avgerinos, and D. Brumley. All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask). In Proc. of the IEEE Symposium on Security and Privacy, 2019, pp. 317– 331.
17. Malware Dynamic Analysis Evasion Techniques: A Survey Amir Aanian, Salman Niksefat, Babak Sadeghiyan, and David Baptiste. URL:https://www.researchgate.net/publication/328758559_Malware_Dynamic_Analysis_Evasion_Techniques_A_Survey (data obrashhenija 24.09.2021).
18. Zhukov, A. E. Modeli vypolnenija processornyh instrukcij v uslovijah protivodejstvija so storony narushitelja dlja komp'juternyh sistem s podderzhkoj tehnologii apparatnoj virtualizacii / A. E. Zhukov, I. Ju. Korkin, B. M. Suhinin // Bezopasnost' informacionnyh tehnologij. – 2012. – T. 19. – № 2. – S. 85-89.
19. Vel'der S.Je., Lukin M. A., Shalyto A. A., Jaminov B. R. Verifikacija avtomatnyh programm. SPb: Nauka, 2011. 244 s.
20. Tvardovskij A.S., Laputenko A.V. O vozmozhnostjah avtomatnogo opisanija parallel'noj kompozicii vremennyh avtomatov // Trudy Instituta sistemnogo programmirovanija RAN. 2018; 30(1):25-40. – URL: https://doi.org/10.15514/ISPRAS-2018-30(1)-2 (data obrashhenija 24.09.2021).
21. Bykovskij S.V. Metod vstroennoj dinamicheskoj aktualizacii funkcional'nyh modelej sistem na kristalle // Izvestija vysshih uchebnyh zavedenij. Priborostroenie. 2015. T. 58, № 3. S. 197-202.
22. Belous, A. Doverennaja JeKB dlja doverennyh apparatno-programmnyh platform: problemy i puti reshenija. Chast' 1 // Jelektronika: Nauka, tehnologija, biznes. – 2021. – № 3(204). – S. 98-104. – DOI 10.22184/1992-4178.2021.204.3.98.104.
23. Unvelling the kernel: rootkit discovery using select automated kernel memory differencing. A. Zaki, Benjamin Humphrey Sophos, UK. URL: https://www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-ZakiHumphrey.pdf (data obrashhenija 24.09.2021).