перезагружали микроконтроллер, но некоторые ошибки приводили к неправильным вычислениям. Далее лазер фокусировался на поверхности, положение платформы фиксировалось в точке, где была получена хоть одна ошибка, а шаг времени запаздывания выбирался как можно меньшим. Если в этой точке удавалось получить несколько ошибок, то она полагалась уязвимой и уже в дальнейшем использовалась для атак на криптографические алгоритмы. Данный метод позволил обнаружить уязвимую зону, которая составляет всего 0,125 % от площади поверхности чипа.

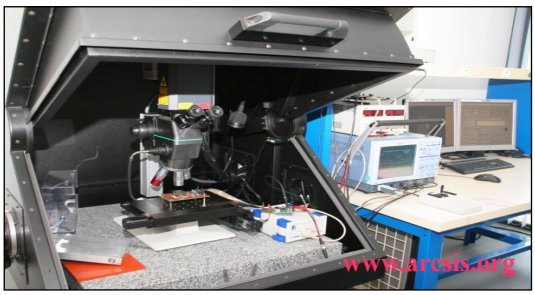

Рис. 1. Оборудование для эксперимента

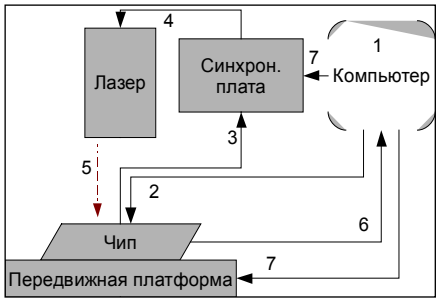

Рис. 2. Схема работы алгоритма

СПИСОК ЛИТЕРАТУРЫ:

1. *Киви Б.* Что показало вскрытие // Хакер. 2003. № 36. Р. 56–62.
2. *Skorobogatov S. P.* Semi-invansive attacks // Technical report. 2005.
3. *Agoyan M., Dutertre J.-M., Mirbaha A.-P., Naccache D., Ribotta A.-L., Tria A.* How to flip a bit? // IEEE 16th International On-Line Testing Symposium. 2010. P. 235–239.

И. Ю. Коркин

МЕТОД ВЫЯВЛЕНИЯ АППАРАТНОЙ ВИРТУАЛИЗАЦИИ В КОМПЬЮТЕРНЫХ СИСТЕМАХ НА ОСНОВЕ МЕХАНИЗМА КЭШИРОВАНИЯ

Несанкционированное присутствие монитора виртуальных машин в системах представляет особую угрозу для безопасности в ситуациях, когда виртуализация сочетается с мерами по искажению показаний процессорного счетчика тактов. Поэтому разработка средств обнаружения монитора для таких ситуаций представляет актуальную задачу.

Штатные средства обнаружения монитора в системах отсутствуют, а опубликованные программные средства в условиях противодействия неработоспособны [1–15]. Под противодействием понимается воздействие монитора виртуальных машин на процессорный счетчик тактов, в результате чего время выполнения трассы под монитором совпадает со временем выполнения в случае отсутствия монитора.

В связи с тем что в предложенном ранее методе обнаружения монитора на основе числа линий точечного графика [16] выявлены недостатки, метод позволяет лишь с некоторой вероятностью определить присутствие монитора. В результате появляется необходимость проведения повторных тестов, что затрудняет применение метода.

Предлагается новый метод, исключающий упомянутые недостатки.

Исследованиями установлено, что если трасса — набор перехватываемых монитором инструкций — выполняется в цикле, то время выполнения трассы t, измеряемое в первой итерации цикла, намного превосходит время трассы во всех последующих итерациях.

Это объясняется тем, что при первом выполнении трассы информация о ней кэшируется, на что процессору требуется больше времени, чем в последующих итерациях цикла, когда процессор с учетом записей в кэше обращается к памяти [17].

Выявлено также, что в целях сокрытия монитора в системе искажением показаний процессорного счетчика тактов время трассы t_V в ПК под виртуализацией (в V-режиме) можно уравнять со временем t_R без виртуализации (в R-режиме) во всех итерациях цикла выполнения трассы, кроме первой итерации, в которой $t_V >> t_R$.

Попытки этим же способом уравнять t_V и t_R в первой итерации цикла могут нарушить работу ОС и поэтому неприменимы на практике, в этом случае будут также отличаться средние арифметические времени выполнения трассы, что позволит выявить монитор.

Благодаря этому для обнаружения монитора в системах вместо двумерных массивов времени трассы, как, например, в методе [16] с последующим непростым статистическим анализом, достаточно получать сравнительно небольшие по объему данные, статистическая обработка которых проста.

Новизна метода заключается в использовании первых измерений с применением очистки кэш-памяти перед каждым выполнением, и используемая характеристика не поддается преднамеренному искажению.

Метод прошел успешную апробацию на следующих моделях процессоров и версиях ОС: Core 2 Duo E6300 (Windows 7, Live CD XP), Core 2 Duo E8200 (Windows 7, Live CD XP), Core 2 Duo E8600 (Windows Live CD XP), Core i7 950 (Windows XP, Live CD XP), Core i3 M330 (Windows 7, Live CD XP). Выполнением повторных опытов на разных компьютерах и в разные календарные дни выявлено, что сходимость и воспроизводимость результатов опытов достаточны.

Метод позволяет обнаруживать присутствие одного монитора в системе, использует сравнительно небольшой объем обрабатываемого материала. В настоящее время мы не располагаем сведениями о наличии способов противодействия предложенному методу.

СПИСОК ЛИТЕРАТУРЫ:

1. Fionnbharr Davies, Hypervisor Malware, Honors Thesis, 2007.
2. *Bulygin Y.* Cpu side-channels vs. virtualization malware: the good, the bad, or the ugly, 2008.
3. *Bulygin Y.* Chipset based approach to detect virtualization malware a.k.a. DeepWatch, 2008.
4. *Garfinkel T., Adams K., Warfield A., Franklin J.* Compatibility is not Transparency: VMM Detection Myths and Realities, 2007.
5. *Adams K.* BluePill detection in two easy steps, 2007.

6. *Barbosa E.* Detecting BluePill, 2007.
7. Hypersight Rootkit Detector. URL: http://northsecuritylabs.com.
8. *Nguyen A. M., Schear N., Jung H. D., Godiyal A., King S. T., Nguyen H. D.* MAVMM: Lightweight and Purpose Built VMM for Malware Analysis.
9. *Sharif M., Lee W., Cui W.* Secure In-VM Monitoring Using Hardware Virtualization, 2009.
10. *Becher M., Dornseif M., Klein C. N.* FireWire: all your memory are belong to us, 2005.
11. *Petroni N. L., Fraser T., Molina J., Arbaugh W. A.* Copilot — a Coprocessor-based Kernel Runtime Integrity Monitor, 2004.
12. *Rutkowska J.* Beyond The CPU: Defeating Hardware Based RAM Acquisition.
13. *Lawson N., Goldsmith D., Ptacek T.* Don't Tell Joanna the Virtualized Rootkit is Dead.
14. *Ramos, Jozo Carlos Carvalho dos Santos.* Security challenges with virtualization, 2009.
15. *Fritsch H.* Analysis and detection of virtualization-based rootkits, 2008.
16. *Коркин И. Ю., Петрова Т. В., Тихонов А. Ю.* Метод обнаружения аппаратной виртуализации в компьютерных системах // Бизнес и безопасность в России. 2010. № 56. С. 114–115.
17. Intel® 64 and IA-32 Architectures Application Note TLBs, Paging-Structure Caches, and Their Invalidation. URL: http://www.intel.com/products/processor/manuals.

Е. В. Котов, С. В. Ктитров

РАЗРАБОТКА ДОМЕНА ЗАЩИЩЕННОЙ ОС НА ОСНОВЕ LINUX

Основной особенностью мандатного разграничения доступа является ограничение передачи данных между субъектами доступа. При получении субъектом доступа к объекту доступа (например, процесса, выполняющегося от имени учетной записи некоторого пользователя, к объекту файловой системы) дальнейшие действия данного субъекта (в нашем случае процесса) в отношении к полученным данным от объекта (в нашем случае файла) методами дискреционного разграничения доступа никак не регламентируются. Таким образом, в рамках дискреционной модели пользователь, имеющий доступ к конфиденциальной информации, может ее дискредитировать, т. е. организовать утечку конфиденциальных данных, поместив эти данные в другой объект файловой системы, права на чтение которого имеет субъект, изначально не имеющий прав на доступ к конфиденциальной информации.

Мандатное разграничение доступа предназначено для ограничения информационных потоков между субъектами доступа.

Сетевая модель разграничения доступа защищенной модификации ОС Linux разрабатывалась с учетом локального разграничения доступа. Сокеты TCP/IP являются одним из средств передачи данных в рамках локальной системы (при взаимодействии через интерфейс обратной связи loopback). Реализация мандатной модели доступа в рамках сетевой подсистемы рассматриваемой ОС ограничивается исключительно добавлением маркеров доступа в сетевые пакеты и анализом их в ядре операционной системы.

Подсистема аутентификации также является локальной, так как задействует исключительно базы данных учетных атрибутов пользователей (в том числе и мандатных), располагающиеся в рамках локальной файловой системы. Для аутентификации в ОС МСВС используются следующие службы:
- PAM — служба загружаемых модулей аутентификации;
- nsswitch — служба доступа к учетной информации системы (кроме мандатного разграничения доступа);
- iss — служба доступа к учетной мандатной информации системы.

Всю информацию в конечном итоге службы получают из баз данных локальной файловой системы.

Мандатное управление доступом, реализованное в рассматриваемой ОС, позволяет осуществлять построение защищенных систем с повышенными требованиями к безопасности и