Дмитрий П. Зегжда[1], Игорь Ю. Жуков[2]
[1]Санкт-Петербургский политехнический университет Петра Великого,
Политехническая ул., 29, Санкт-Петербург, 195251, Россия
[2]АО «РАМЭК-ВС»,
Волгоградский пр–т, 2, Москва, 109316, Россия
[1]e-mail: dmitry@ibks.spbstu.ru, https://orcid.org/0000-0002-0232-7248
[2]e-mail: i.zhukov@inbox.ru, https://orcid.org/0000-0002-4429-8799

ОСОБЕННОСТИ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

DOI: http://dx.doi.org/10.26583/bit.2021.1.04

Аннотация. Цель статьи – комплексный анализ вопросов технологической независимости и информационной безопасности вычислительных систем. Применение системного подхода предполагает анализ технологий защиты на всех уровнях архитектуры и их взаимодействие между собой. Рассматриваются аппаратные технологии защиты на уровне процессора и системы команд. Аппаратная поддержка виртуализации, которая обеспечивает поддержку со стороны аппаратного обеспечения такой функции как виртуализация операционных систем, становится необходимой функцией защиты и не уступает в важности уже классическим функциям: управление доступом, идентификация, аутентификация, аудит и контроль безопасности. Помимо аппаратной поддержки виртуализации рассмотрены реализации в современных процессорах группы технологий аппаратной поддержки обособленной доверенной среды. Проведен анализ возможности использования аппаратных технологий защиты злоумышленниками для вредоносных воздействий. По результатам исследований предложен подход по созданию отечественной защищенной архитектуры средств вычислительной техники (СВТ) с использованием зарубежных аппаратных технологий защиты. При этом импортозамещение не должно сводиться исключительно к репликации зарубежных решений, т.к. зарубежные СВТ содержат массу недокументированных возможностей и, как следствие, несут угрозу информационной безопасности.

Ключевые слова: архитектура вычислительных систем, аппаратные технологии защиты, виртуализация, вредоносные воздействия, технологическая независимость, импортозамещение.
Для цитирования: ЗЕГЖДА, Дмитрий П.; ЖУКОВ, Игорь Ю. ОСОБЕННОСТИ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ. Безопасность информационных технологий, [S.l.], v. 28, n. 1, p. 42–61, jan. 2021. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/1320>. Дата доступа: 01 feb. 2021. DOI: http://dx.doi.org/10.26583/bit.2021.1.04.

Dmitry P. Zegzhda[1], Igor Y. Zhukov[2]
[1]Peter the Great St. Petersburg Polytechnic University,
29 Politechnicheskaya str., Saint Petersburg, 195251, Russia
[2]JSC «RAMEC VS»,
Volgogradskiy Prospekt, 2, Moscow, 109316, Russia
[1]e-mail: i.zhukov@inbox.ru, https://orcid.org/0000-0002-4429-8799
[2]e-mail: dmitry@ibks.spbstu.ru, https://orcid.org/0000-0002-0232-7248

Features of information security of computer systems

DOI: http://dx.doi.org/10.26583/bit.2021.1.04

Abstract. Presented is a comprehensive analysis of the issues of technological independence and information security of the computing system. The systematic approach involves the analysis of protection technologies at all levels of architecture and its interaction with each other. Hardware protection technologies at the processor and command system level are considered. Hardware support for virtualization turns out to be a necessary security function and is not less important than the already traditional security functions like an access control, identification, authentication, audit and security

control. In addition to hardware support for virtualization we discuss the implementation of the group of technologies for hardware support for a separate trusted environment in modern processors. The analysis of the possibility of using hardware protection technologies by intruders for malicious influences is carried out. Based on the research results, an approach to create a domestic protected architecture of computer equipment using foreign hardware protection technologies is proposed. At the same time, import substitution should not be limited solely to the replication of foreign solutions, since foreign computer equipments contain a lot of undocumented capabilities and, as a result, pose a threat to information security.

Keywords: computer system architecture, hardware protection technologies, virtualization, malicious impacts, technological independence, import substitution.

For citation: ZEGZHDA, Dmitry P.; ZHUKOV, Igor Y. Features of information security of computer systems. IT Security (Russia), [S.l.], v. 28, n. 1, p. 42–61, jan. 2021. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/1320>. Date accessed: 01 feb. 2021. DOI: http://dx.doi.org/10.26583/bit.2021.1.04.

Введение

За последние несколько лет мощное развитие получили современные технологии аппаратной защиты [1–6]. Эти технологии являются предметом гордости любой IT-компании. В рамках статьи аппаратные технологии защиты распределим по десяти группам:

– поддержка привилегированного режима и контроль обращения к адресным пространствам;
– защита от угроз со стороны аппаратных устройств;
– аппаратная поддержка независимого контроля целостности системы в процессе загрузки;
– аппаратная поддержка виртуализации;
– аппаратная поддержка обособленной доверенной среды;
– противодействие эксплуатации уязвимостей;
– реализация криптографических примитивов и защищенное хранилище ключей;
– аппаратная идентификация и биометрическая аутентификация пользователей;
– защита от аппаратных сбоев;
– автономный чип контроля управления системой.

Каждая группа технологий содержит сразу несколько решений, однако все они стремятся решить одну из задач обеспечения безопасности системы.

Поддержка привилегированного режима и контроль обращения к адресным пространствам является базовой технологией, которой оснащены практически все современные микропроцессоры.

В первую очередь, она формирует минимально необходимые два режима процессора: разделение ПО на системное – работающее в привилегированном режиме и прикладное – работающее в «пользовательском» режиме процессора.

Вторая технология из этой группы – это реализация базового контроля обращения к изолированным адресным пространствам. Функция изоляции обеспечивает возможность локализации ошибок в рамках одного прикладного процесса, не нарушая работы всей системы. Кроме того, она создает в системе базовые сущности, которые могут быть аутентифицированы субъектами, без которых невозможно обеспечить контроль обращения.

В рамках «защищенного режима» реализуется технология виртуальной памяти, которая сразу решает несколько задач: изоляция каждого процесса в системе друг от друга в отдельном адресном пространстве и обеспечение для каждого процесса одинакового

адресного пространства вне зависимости от реального объема оперативной памяти. Эта технология реализуется в рамках модуля управления памятью (memory management unit – MMU), который является частью практически любого современного микропроцессора.

Вторая группа технологий **защита от угроз со стороны аппаратных устройств** – это технологии, обеспечивающие функции контроля обращения внешних (по отношению к центральному процессору) устройств к аппаратным ресурсам: оперативной памяти и другим устройствам.

Современные устройства научились работать друг с другом, с центральным процессором и памятью на огромных скоростях, в основном, за счет появления технологии DMA (direct memory access). Эта технология фактически разрешила доступ устройств к оперативной памяти без участия центрального процессора. В этой связи процессор стал всего лишь одним из потребителей оперативной памяти [6].

Основная угроза со стороны аппаратных устройств – внедрение вредоносного ПО в их программные прошивки.

Хорошим примером такого устройства является современный видео-адаптер, который не только обладает мощным вычислителем с сотнями процессорных ядер, собственной оперативной памятью (видео-память), но и обладает сложным программным обеспечением и даже реализует функции виртуальной памяти. Таким образом, разграничение обращения внешних устройств к оперативной памяти и их взаимная изоляция являются актуальными задачами [7].

Аппаратно-программные модули доверенной загрузки (АПМДЗ) широко применяются на отечественном рынке. Цель этих систем – обеспечение целостности конфигурации системы путем поэтапной проверки целостности состава аппаратного и программного обеспечения в течении всей процедуры загрузки. Результатом является уверенность пользователей системы в том, что в систему не был внедрен вредоносный код и не была нарушена целостность с момента предыдущего запуска. Это крайне необходимо для обеспечения безопасности работы любой информационной системы.

Суть традиционных АПМДЗ заключается в статическом контроле целостности, когда проверки осуществляются строго последовательно: каждый очередной загружаемый компонент сначала проверяется системой контроля целостности, а только затем (после успешного прохождения проверки) компонент выполняется.

Если в таком способе нарушить цепочку и пропустить хотя-бы один компонент, то это будет серьезным нарушением безопасности, которым сможет воспользоваться нарушитель. Для доверенного запуска системы контроль целостности нужно осуществлять для каждого компонента, в том числе и для тех, которые являются лишь промежуточными и не требуются для долговременной работы пользователей. При этом традиционная технология уже используется на отечественном рынке и в своем составе использует сертифицированные алгоритмы.

Однако, компаниями Intel и AMD [8, 9] была разработана интересная технология динамического контроля целостности. Суть динамического контроля целостности заключается в том, что на начальных этапах запуска системы контроль целостности не осуществляется вовсе. Затем, перед самым запуском ядра ОС (доверенным компонентом всей основной системы, с которой будут работать пользователи) выполняется контроль целостности конфигурации системы атомарным образом – за одну инструкцию (например, в технологии Intel TXT это инструкция GETSEC[ENTER]). Получается, что технология встроена в сам процессор. Однако помимо процессора она задействует возможности чипсета и внешних устройств защиты. За эту инструкцию процессор самостоятельно проверяет, что работает только одно ядро, в строго определенном режиме, что отключены

прерывания, что вся система настроена правильно (таким образом, что ни одно устройство или код не может нарушить или прервать ход дальнейшей проверки), а затем в соответствии с настроенной политикой безопасности процессор выполняет контроль целостности ПО и конфигурации системы. Все выполняется безусловно и атомарно, что обеспечивает надежный механизм контроля целостности. Кроме того, эту инструкцию можно вызвать в любой момент времени, в том числе и повторно.

К сожалению, технологии динамического контроля целостности не используют отечественные криптографические алгоритмы и не могут иметь широкого применения на отечественном рынке. Эти функции требуют реализации в составе процессоров, что конечно, соответствует требованиям нормативных документов с точки зрения цели, но не соответствует этим документам с точки зрения используемых методов алгоритма контроля. Тем не менее, ни что не мешает заменить алгоритмы и получить все преимущества этой мощной технологии.

Очень важная технология **Аппаратная поддержка виртуализации** – которая обеспечивает поддержку со стороны аппаратного обеспечения такой функции как виртуализация операционных систем. Можно утверждать, что виртуализация становится необходимой функцией защиты и не уступает в важности уже классическим функциям: управление доступом, идентификация, аутентификация, аудит и контроль безопасности. Такое значение функции виртуализации операционных систем придает то, что это единственная функция, которая способна обеспечить и гарантировать изоляцию нескольких вычислительных сред друг от друга.

Фактически виртуализация расщепляет аппаратную платформу на несколько программных платформ (виртуальных машин). Важно, что именно такие программные платформы включают в себя полноценную операционную систему со всем многообразием прикладных и системных программных средств, размещенных внутри контролируемой гипервизором виртуальной машине.

На практике такой эффект достигается тем, что в процессоре появляется две группы режимов: режимы монитора виртуальных машин (root или хост) и режимы виртуальных машин (non-root или гость). Кроме того, помимо традиционного MMU вводится дополнительный уровень контроля и виртуализации оперативной памяти.

Понятия гипервизора (его еще называют монитором виртуальных машин) и виртуальной машины были введены еще в 70-х годах прошлого века. Однако за счет возникновения аппаратной поддержки виртуализации и развития облачных систем эти понятия широко используются в IT-индустрии в настоящее время.

В настоящее время обеспечение изоляции вычислительных сред – это очень актуальная задача. Все чаще приходится использовать информационные системы, состоящие из нескольких операционных систем, где часть из них доверенные, другие не являются доверенными. Важно, что пользователям необходимо использовать возможности каждой из частей. Объединять доверенные и не доверенные части вместе в рамках одной операционной системы – невозможно! Только функция виртуализации может обеспечить их гарантированную изоляцию [8, 9].

Поскольку функция виртуализации операционных систем крайне важна для обеспечения безопасности, то ее аппаратная поддержка является краеугольным камнем многих современных средств защиты. На самом деле известно, что гипервизор можно реализовать без аппаратных технологий. Вопрос в том, насколько это будет эффективно и насколько это будет гарантированно с точки зрения надежности работы.

Помимо функции изоляции виртуализация операционных систем способна обеспечить еще целый ряд полезных функций, которые не менее важны с точки зрения безопасности:

– контроль за всеми программными средствами, в том числе ядрами операционных систем, в виртуальной машине;

– управление потоками ввода-вывода между операционной системой и внешними устройствами;

– обеспечение защищенного обмена данными между виртуальными машинами;

– контроль вложенных гипервизоров и виртуальных машин.

Фактически гипервизор может управлять поведением и даже предотвращать нежелательное поведение тех средств, которые не являются доверенными. Последняя функция крайне интересна, поскольку функция виртуализации может использоваться рекурсивно: гипервизор позволяет запускать в виртуальной машине другие гипервизоры. Такой гипервизор будет называться вложенным, как и его виртуальные машины. В этом случае, вложенный гипервизор уже не будет привилегированным, и будет действовать принцип первичности: первый, запущенный на ПЭВМ гипервизор будет привилегированным, а все остальные вложенные гипервизоры будут подчиненными.

Помимо аппаратной поддержки виртуализации в современных процессорах реализована еще одна группа технологий **аппаратной поддержки обособленной доверенной среды**.

Проблема доверия характерна не только для нашей страны. Она на самом деле имеет глобальный характер. Даже такие крупные компании, как Intel и Microsoft, все равно идут к тому, чтобы создать в рамках общего вычислительного пространства некую среду с повышенным уровнем доверия [10–12]. Такая изолированная доверенная среда необходима для обработки ключей и прочей чувствительной конфиденциальной информации. Для этого работа полноценной операционной системы не требуется, поэтому функция виртуализации также не требуется, в группу аппаратной поддержки обособленной доверенной среды входят следующие технологии:

– размещение привилегированного кода в защищенной области памяти;

– разделение вычислительной среды на доверенную и не доверенную;

– обязательная проверка подписи для доверенного кода.

Для решения указанной задачи получили широкое развитие соответствующие технологии, которые обеспечивают в первую очередь целостность и подлинность доверенной среды [13–15]. Кроме того, они ограничивают взаимодействие между доверенной средой и остальной частью системы. Фактически это традиционная защита, только реализованная прямо в рамках архитектуры аппаратного обеспечения.

В рамках этой группы следует выделить несколько любопытных технологий и описать их более подробно.

Во-первых, следует указать технологию ARM TrustZone [12], которая позволяет разделять код, данные и внешние устройства на два домена (доверенный и не доверенный). При этом на аппаратном уровне не только контролируется целостность доверенного домена, но и запрещен доступ внешним устройствам к данным доверенного домена.

Во-вторых, технология SMM (System Management Mode). Дело в том, что технология SMM известна уже давно. Изначально она решала ряд далеких от безопасности специфических задач. Как указано в официальной документации, в настоящее время SMM – это самый привилегированный режим работы процессора. Код в

этом режиме изолирован от всего остального кода и к нему запрещен доступ со стороны аппаратных устройств. Де-факто SMM формирует полноценную изолированную среду.

В-третьих, следует обратить внимание на самую «свежую» технологию – Intel SGX (Safer Guard Extension). Эта технология любопытна тем, что она, в отличие от других, обеспечивает создание изолированной вычислительной среды в рамках непривилегированного прикладного процесса операционной системы. В классической архитектуре информационных систем, чем привилегированнее компонент, тем он более защищён и тем больше к нему доверие. Технология Intel SGX поступает наоборот: она позволяет создать так называемый анклав (защищенную область кода и данных) в рамках непривилегированного прикладного процесса. При этом доступ к нему запрещен аппаратно со стороны других процессов, ядра операционной системы, гипервизора и любых внешних устройств. Такой любопытный результат достигается благодаря тому, что весь код и данные в рамках анклава хранится в общей оперативной памяти в зашифрованном виде. Такой мощный ход был реализован благодаря тому, что современные процессоры Intel [16] прямо в кристалле реализуют множество криптографических функций. В результате данные анклава в открытом виде существуют только глубоко в ядре процессора. Кроме того, технология обеспечивает аппаратный контроль целостности кода, попадающего в анклав за счет аппаратной проверки подписи перед запуском этого кода.

Противодействие эксплуатации уязвимостей. С точки зрения практики, защищенная система это та, в которой отсутствуют уязвимости. Можно определить уязвимость как специфический вид ошибки, который позволяет нарушить безопасность системы. В любой программе есть ошибки, однако не любая ошибка является уязвимостью. Не все ошибки позволяют осуществить атаку и тем более обеспечить нарушителю возможность проникнуть в систему [17].

Кроме того, есть такое понятие, как уязвимость нулевого дня. Это такая уязвимость, про которую никто в системе не знает, поэтому ни одна система защиты противостоять не может. Как следствие, эта уязвимость остается доступной для нарушителя [18].

Поэтому разработчики операционных систем и разработчики процессоров ввели меры для противодействия уязвимостям, что фактически сформировало второй эшелон защиты. Другими словами, если в первом эшелоне (в стандартных средствах контроля обращения во всех доверенных загрузчиках, гипервизорах и прочем) существует уязвимость, которую нарушитель нашел и воспользовался ею, то, благодаря второму эшелону, ему создается ряд препятствий, которые придется также преодолеть.

Типизация памяти (type enforcement): процессору указывается, где код, где стек, где данные, что выполнять можно, что модифицировать нельзя, и т. д. Кроме того, целая группа технологий препятствует передаче управления из кода на одном уровне привилегий в код на другом уровне привилегий:

- контроль выполнения, чтения, записи для виртуальных адресных пространств;
- управление типами кэширования для областей памяти;
- управление границами сегментов и типизация памяти;
- запрет обращения к непривилегированному коду из привилегированного режима.

Все перечисленное – это достаточно эффективные функции. Они реально работают на практике и обеспечивают тот самый второй эшелон защиты. Сама архитектура современных операционных систем, зачастую, препятствует применению этих технологий, поскольку некоторые системные элементы современных ОС в своем

поведении очень похожи на эксплуатацию уязвимости. Например, антивирусы или средства виртуализации часто используют такие приемы, которые облегчают разработку ПО и повышают производительность, но при этом они отходят от требований безопасности и нарушают правила, делая свое поведение похожим на уязвимость. Именно поэтому указанные технологии защиты, к сожалению, применить на все 100% не представляется возможным, хотя нарушителям создаются дополнительные препятствия.

Однако необходимость наличия технологий защиты от эксплуатации уязвимостей потребовала что-то менять. В результате необходимые функции были добавлены в модель виртуальной страничной памяти (что является частью MMU – memory management unit). Такой подход позволил разработчикам размечать код, стек, данные и иные сегменты памяти как им удобно. При этом с точностью до страницы (4 Кб) указываются атрибуты обращения: для заданной области памяти: чтение, модификация, выполнение, область системная или прикладная. Такой подход оказался простым и удобным, что способствовало его внедрению во все современные ОС.

Важной функцией защиты является **криптография**. Для современных систем необходима полноценная реализация криптографических функций, которая не только обеспечивает приемлемую производительность, но и защиту ключевой информации. Рассмотрим несколько таких технологий подробнее.

Во-первых, для решения задач ускорения криптографических функций, что крайне необходимо для современных систем, обычно используют ПЛИС системы. Однако на современных процессорах эту задачу решают путем добавления новых инструкций: например Intel AES-NI и MIPS OmniShield. Указанные технологии реализуют западные криптографические алгоритмы, которые уже встроены в процессор. Однако Intel уже давно анонсировала технологию Intel Xeon FPGA, которая должна позволить разработчикам ПО добавлять в процессор свои инструкции для реализации криптографических примитивов, которые необходимы пользователям.

Во-вторых, следует отметить популярную технологию TPM (trusted platform module), которая представляет собой изолированный чип, основная задача которого обеспечить защищенное хранилище ключей. Это достигается путем того, что сам чип хранит в себе уникальный ключ, который никогда не покидает кристалл и защищен физически от всевозможных рентгеновских сканеров и иных средств чип-реверсинга (реверс-дизайна). В самом чипе реализуется множество криптографических функций (более 20-ти) которые могут использовать тот самый внутренний ключ.

Другой важной функцией, которая важна с точки зрения архитектуры системы является **биометрическая аутентификация**. Необходимость аутентификации возникает постоянно, а пользователи не могут создать такое количество ключей и паролей, а главное их запомнить. В результате создается огромный риск утечки и компрометации этих самых ключей. Биометрическая аутентификация очень легка с точки зрения использования и минимизирует эту угрозу. Поэтому поддержка такой аутентификации тоже встраивается во многие современные вычислительные системы и тоже является элементом архитектуры.

Ошибки в программных средствах могут приводить к уязвимостям и атакам на системы, а ошибки в аппаратном обеспечении и природные воздействия на него, могут приводить к потере данных и отказу в обслуживании. Поэтому современная архитектура тоже оснащается средствами защиты от аппаратных сбоев, которые позволяют преодолеть такие проблемы.

Другими словами, помимо ошибок, допущенных разработчиками, на нарушение целостности данных могут влиять и внешние факторы. Поэтому для защиты от подобных

нарушений в современные системы был внедрен целый ряд технологий для защиты от аппаратных сбоев. Эти технологии внедряются практически на каждом аппаратном компоненте современных ПЭВМ. Для пользователей и программистов они являются прозрачными.

Что такое доверие – достаточно сложный вопрос. Понятно, что доверие имеет некую иерархическую природу: когда одни компоненты зависят от других. Например, операционная система контролирует приложения. Соответственно, приложение доверяет операционной системе. При этом гипервизор контролирует операционную систему и операционная система доверяет гипервизору. Процессор контролирует операционную систему и гипервизор. Следовательно, гипервизор и операционная система доверяют процессору.

Что контролирует процессор? Раньше внешние устройства на материнской плате не имели в своем составе функций контроля и были достаточно простыми. Однако со временем на материнской плате современных ПЭВМ появился чип, реализующий функции независимого контроля и управления системой. Это чипсет, который часто называют «южный мост». Мотивацией к такому развитию послужила необходимость реализации удаленной диагностики, управления и восстановления систем после сбоев. Эта необходимость появилась из-за того, что современным системным администраторам требуется обслуживать большое количество компьютеров одновременно. Это касается не только огромных дата-центров с облачными системами, но и обычного корпоративного сегмента, где работают стандартные конфигурации ПЭВМ. В результате, практически все современные компьютеры на рынке обладают таким независимым автономным чипом контроля (рис. 1).

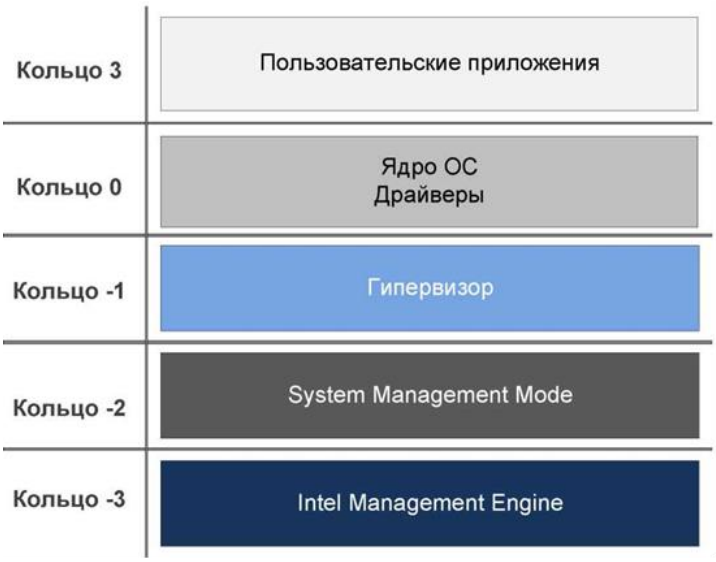

Рис 1. Новый уровень контроля и управления в системе – «чипсет»
Fig. 1. A new level of control and management in the system – «chipset»

Удаленный контроль и диагностика не единственная технология, реализуемая на этом чипе. В частности, автономный чип управления играет роль в реализации функций защиты ноутбуков от кражи, а также функции защиты авторского контента (так называемый DRM – digital rights management). При краже этот чип может самостоятельно удалить все конфиденциальные данные с компьютера, или сообщить через сайт пользователю о своем местоположении.

Фактически автономный чип управления является компьютером в компьютере. Он обладает доступом к оперативной памяти, системам управления питанием, возможностью фильтровать сетевой траффик, осуществлять активное независимое взаимодействие по сети (например, в рамках технологии Intel AMT реализуется полноценный веб-сервер), имеет доступ к видео-карте. Описание технологий защиты, которые реализуются с участием этого чипа, говорят о том, что все эти возможности чипу действительно необходимы. Однако, достаточно очевидно, что в этом чипе функций безопасности на самом деле гораздо больше. Это связано с тем, что вырабатывается определенный уровень доверия к приложению, операционной системе и процессору, но на самом деле доверять необходимо этому автономному чипу – «южному мосту», поскольку все остальное функционирование системы зависит от него. К сожалению, вопрос доверия к этому чипу становится особенно непростым из-за практически полного отсутствия документации на него.

В результате исследования составлена табл. 1, в которой представлены все рассмотренные технологии и их наличие в различных современных процессорах.

Таблица 1. Распространенность аппаратных технологий защиты среди современных микропроцессоров

	Intel x86, (Skylake)	ARM Cortex-A, Байкал-M	AMD Bulldozer	MIPS Varrior Байкал-T1	VIA Nano	PowerPC	Эльбрус	Соболь	TPM	Intel 100 series	КРИПТОН-ЗАМОК
Защита от угроз со стороны аппаратных устройств	да		да					да		да	
Аппаратная поддержка независимого контроля целостности системы в процессе загрузки	да	да	да					да	да		да
Аппаратная поддержка виртуализации	да	да	да	да	да	да					
Аппаратная поддержка обособленной доверенной среды	да	да	да	да			да				
Противодействие эксплуатации уязвимостей	да	да	да	да	да	да	да				
Реализация криптографических примитивов и защищенное хранилище ключей	да	да	да		да			да	да		да
Аппаратная идентификация и биометрическая аутентификация пользователей								да			да
Автономный чип контроля и управления системой	да		да							да	
Защита от аппаратных сбоев	да	да	да	да	да	да	да	?		да	?

Из табл. 1 следует, что очевидным лидером по количеству реализованных технологий защиты являются процессоры Intel. Этот вывод достаточно понятен, поскольку Intel является на текущий момент лидером на рынке микропроцессоров.

Эта компания производит полный спектр микропроцессоров, материнских плат, графических и сетевых адаптеров, чипсетов и других устройств. При этом, микропроцессоры Intel представлены во всех видах современных средств вычислительной техники: от контроллеров умных домов (Intel Edison) и мобильных телефонов до суперкомпьютеров.

Такая популярность процессоров Intel и архитектуры x86 неразрывно связана с двумя важными аспектами.

Во-первых, популярность провоцирует большое внимание к себе, что на практике делает проблему безопасности наиболее актуальной именно для этой платформы. Пока не было массовых компьютеров, проблема безопасности информации не стояла так остро. Но за счет широкой распространенности неквалифицированные пользователи и потребители соприкоснулись с этой техникой. При этом, возникла проблема безопасности, поскольку безопасность без человека существует. В результате такой популярности, именно от процессоров Intel стало требоваться как можно больше возможностей по защите, поскольку для них и реализуется больше всего угроз.

Во-вторых, жизнь и развитие технологий сказываются так, что при выборе той или иной архитектуры, того или иного процессора, решающий вклад имеет фактор наличия большого количества ПО для этой платформы, и, соответственно, количество разработчиков и пользователей, обученных этому программному обеспечению. За счет такого «рычага» Intel интенсивно внедряет свои технологии во все остальные сегменты. В результате все самые мощные суперкомпьютеры построены на архитектуре Intel, потому что пользователи хотят использовать популярное программное обеспечение, которое уже разработано для архитектуры x86.

1. Технологии защиты средств вычислительной техники

Широкая доступность ПК с архитектурой x86 и систем разработки ПО для них обусловило распространение этой архитектуры во все классы СВТ [3].

Современные персональные компьютеры (ПК) не только набрали огромную популярность и получили широкую распространенность, но и приобрели весь возможный спектр функциональных возможностей: они участвуют в сетевом взаимодействии, могут быть мощными вычислительными средствами, могут использовать различные датчики, управлять простыми механизмами, взаимодействовать с радиоканалами WiFi/GSM, запускать виртуальные машины, осуществлять перенаправление сетевого траффика и предоставлять различные сетевые сервисы другим пользователям, и еще многое другое.

Т.о. персональные компьютеры – это универсальное средство СВТ. С развитием современных технологий стало ясно, что уход от универсальности в сторону специализации устройств позволяет добиться лучших результатов и получить больше преимуществ. Можно выделить два основных направления развития персональных компьютеров, что привело к появлению новых видов СВТ (рис. 2).

Во-первых, развитие ПК в сторону увеличения вычислительной мощности и параллельности привело к появлению суперкомпьютеров. Кроме того, объединение множества компьютеров в вычислительную сеть привело к появлению интернет-серверов и облачных систем.

Во-вторых, уменьшение энергопотребления и облегчение функций современных процессоров привело к появлению мобильных устройств. Как развитие этого направления, появились киберфизические системы, в которых компьютерная техника выступает в роли маломощных, но «умных» датчиков. Кроме того, снижение энергопотребления и

фокусирование вычислительной техники на одной задаче привело к появлению такого вида СВТ как телекоммуникационное оборудование.

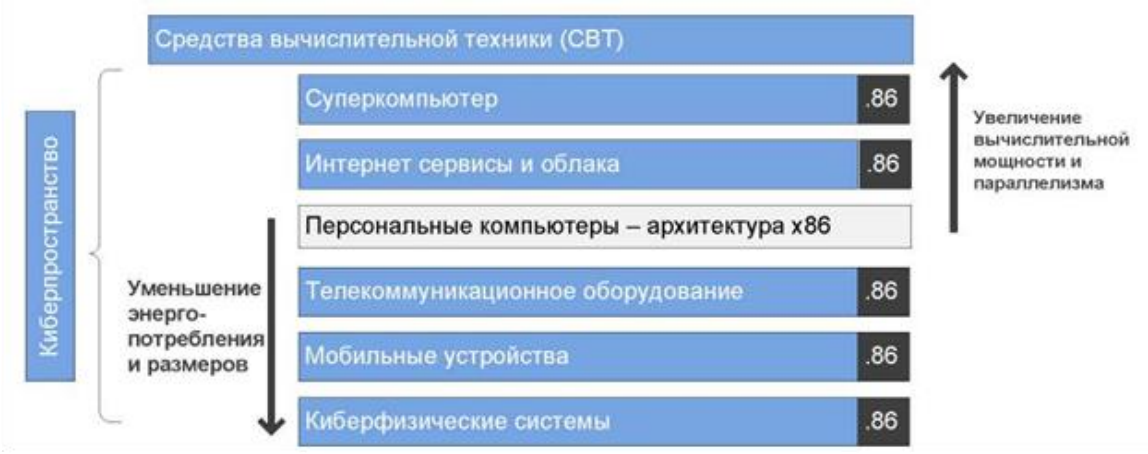

Рис. 2. Появление различных видов СВТ из ПК
Fig. 2. The emergence of various types of Computer Equipment from Personal Computers

Выделенные аппаратные технологии защиты эффективно используются при построении одной из самых распространенных инфраструктур в мире компании Apple, использующей технологии защиты на уровне процессора, СВТ (компьютера и/или мобильного устройства), операционных систем (iMAC и iOS), а также облачной системы (AppStor, iTune).

2. Угрозы аппаратных технологий защиты

Однако, некоторые технологии защиты, которые имеют аппаратную поддержку, могут быть использованы не только для решения задач защиты, но и могут быть использованы злоумышленниками для повышения качества своих средств атаки и нападения. Такими обоюдоострыми свойствами обладают две из рассмотренных аппаратных технологий защиты:

- аппаратная поддержка виртуализации;
- автономный чип контроля и управления системой.

Рассмотрим, как эти аппаратные технологии могут использоваться вредоносным программным обеспечением (ВПО). ВПО может использовать аппаратную технологию виртуализации для захвата управления системой без нарушения ее работы. Кроме того, ВПО может использовать эту технологию для скрытия своего присутствия и своей активности. Это происходит благодаря тому, что средства управления (гипервизор) фактически являются отдельным объектом атаки. При этом атака направляется туда, где нет средств защиты. Это не учитывать совершенно невозможно. Практика показывает, что уже известны такие атаки со стороны аппаратного обеспечения и со стороны программного обеспечения, которые сильно завязаны на аппаратные технологии

Наиболее известна атака, под названием Blue Pill (рис. 3), которая была продемонстрирована Иоанной Рутковской в 2006 г. в рамках конференции Black Hat. Продемонстрированная атака приводит к тому, что программное средство Blue Pill из программы Windows становится гипервизором. Гипервизор создает виртуальную машину и переносит в нее ОС, после чего начинает контролировать ее поведение, скрывая свое присутствие. Фактически нарушитель создает виртуальную ОС, реализует свои средства в

гипервизоре вне этой самой ОС. При этом злоумышленник становится совершенно прозрачен и невидим для средств защиты. Средства защиты функционируют в рамках операционной системы и играют по ее правилам, которые теперь свободно может менять нарушитель. В результате такую атаку не обнаруживают ни антивирусы, ни любые другие используемые средства защиты. Важно подчеркнуть, что это происходит не потому, что средства защиты плохо работают, а потому что они принципиально обнаружить такое воздействие на систему не могут, поскольку они живут внутри этой самой виртуальной машины, которую организовал злоумышленник. Раз такой практический эксперимент был продемонстрирован, то это реальность, а не теория.

Рис. 3. Использование виртуализации в эксперименте Blue Pill
Fig. 3. Using virtualization in the Blue Pill experiment

Другая группа аппаратных технологий защиты, которую может использовать ВПО – это автономный чип контроля и управления системой. Для наглядности эта группа технологий будет рассматриваться на примере Intel ME (Management Engine), она же Intel AMT (Active Management Technology) и Intel vPro. Эта технология фактически реализует автономную операционную систему на базе чипсета (второй по важности аппаратный чип в компьютере после процессора). В результате чипсет становится даже более привлекательной целью, чем основной процессор и ОС потому, что попав в операционную среду Intel ME можно выполнять все те действия, которые хочет выполнять обычно нарушитель. При этом злоумышленник остается абсолютно невидимым и недосягаемым для средств защиты, запущенных на центральном процессоре, так как реализация этих функций осуществляются вне ОС, в которой он работает.

Изначально технология Intel ME предназначена для того, чтобы управлять самим процессором и другими аппаратными средствами компьютера. Кроме того, на базе технологии Intel ME был реализован целый калейдоскоп различных технологий, который позволяют удаленно диагностировать состояние компьютера в дата-центрах, позволяют выключать и стирать удаленно конфиденциальные данные, загружаться с виртуальных DVD дисков, удаленно устанавливать операционные системы, скрывать доступ к заданным областям экрана для защиты видео материалов от копирования, осуществлять фильтрацию сетевых пакетов, скрывать активную работу с сетью, в том числе через WiFi. По мнению экспертов, в этой технологии кроется самая серьезная угроза. Вот известный список возможностей, которые получает нарушитель, в случае успешного перехвата управления над Intel ME:

- контроль СВТ на любой стадии загрузки ОС;
- манипулирование системным окружением СВТ;
- манипулирование аппаратными технологиями защиты;

– наличие собственного DMA-контроллера с обходом MMU;
– максимальная изоляция исполняемого кода ME/AMT от ОС;
– постоянное функционирование на дежурном питании;
– доступ к OnBoard-устройствам (GPU, USB/Ethernet/WiFi/NFC);
– удаленное управление СВТ (централизованные/децентрализованные схемы);
– прослушивание сетевого трафика / сбор парольно-адресной информации;
– возможность перенаправления/модификации/зеркалирования сетевого трафика;
– организация p2p-сети для выхода за периметр ЛВС;
– мониторинг активности пользователя;
– подмена изображения на экране;
– сбор информации со всех подключаемых устройствах (флэш-накопители /телефоны/токены/датчики);
– взаимодействие с API ОС для сбора пользовательской информации (адресная книга телефона/медиа-контент/данные токенов).

Как же противодействовать тому, что нарушитель может использовать такие опасные возможности архитектуры, как аппаратная технология виртуализации и автономный чип контроля и управления системой? Во-первых, возникает искушение от этих технологий просто отказаться: раз этими средствами может воспользоваться нарушитель, то это следует просто вырезать и убрать из системы.

По мнению экспертов, это неправильный подход потому, что это шаг назад. Ведь, если от них отказаться, то пользователи и разработчики, а не только злоумышленники, не смогут их использовать для полезных целей. Действительно, технология виртуализации и Intel ME предоставляют прекрасные и полезные возможности, которые повышают эффективность и позволяют решать более сложные задачи. Самый простой пример, без этих технологий современный дата-центр не построить. Без виртуализации экономические затраты будут слишком большие и ресурсы будут простаивать, а без технологии подобной Intel ME не обеспечить администрирование такого центра и не осуществить своевременное выявление аппаратных сбоев, которые в таких масштабах не редкость. На самом деле, отказываться от этих технологий нет необходимости. Эта проблема безопасности информации имеет решение.

Рассмотрим аппаратную поддержку виртуализации: чем можно нейтрализовать ее использование со стороны нарушителя? Во-первых, эффективно работает аппаратная технология независимого контроля целостности в процессе загрузки. Эта технология гарантирует, что гипервизор нельзя встроить в установленную на ПК прошивку. Во-вторых, тот же самый чип контроля и управления системой (даже Intel ME) следует использовать для того, чтобы предотвратить захват гипервизора нарушителем или установку гипервизора (такого как Blue Pill) в систему. Следовательно, на самом деле эти технологии образуют некую иерархию (рис. 4).

Иерархия характеризуется последовательностью контроля: более приоритетная с аппаратной точки зрения технология может контролировать другую технологию, в основе иерархии находится автономный чип контроля и управления системой, потому что он лежит на самом низком уровне системы – это даже не процессор с его режимами, а отдельный чип. Очевидно, что, если нарушитель хочет преодолеть защиту и, если он не может преодолеть ее «в лоб», то он должен «пролезть» под ней, на более низком уровне, что в частности было с успехом продемонстрировано Ионной Рутковской не только в 2006 г., но и в 2009 г., когда она успешно внедрила свой код в Intel ME.

Получается, чтобы обеспечить безопасность, следует обеспечить доверие к тому компоненту, который располагается на самом низком уровне иерархии. В результате, этот

компонент обладает самым высоким уровнем управления и контроля, и является, таким образом, корнем доверия.

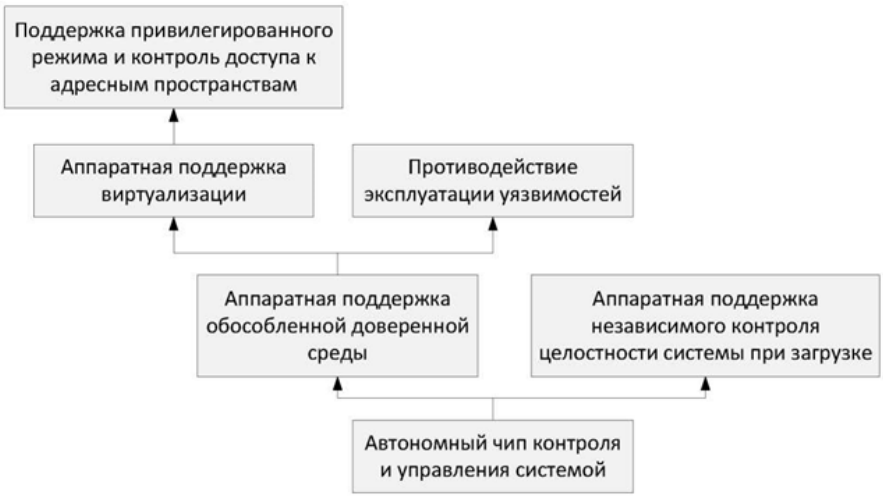

Рис. 4. Иерархическая зависимость технологий
Fig. 4. Hierarchical dependence of technologies

Раньше считалось, что таким компонентом является ядро процессора. Однако в настоящее время стало понятно, что этот компонент извлечен из процессора. По мнению экспертов, такая интерпретация понятия «доверие» и есть то, к чему должно сводиться импортозамещение. Импортозамещение не должно повторять то, ЧТО уже есть на рынке. Необходимо сосредоточиться на том компоненте системы, который будет контролировать все элементы СВТ. Вместе с этим, следует учитывать научно-технический задел, чтобы обеспечить более высокий уровень доверия ко всей системе.

Архитектура отечественного СВТ должна включать автономное средство защиты, полностью контролирующее функционирование информационной системы, обеспечивающее доверие и безопасность [19].

3. Архитектура защищенных СВТ

При рассмотрении задачи построения современных СВТ с использованием зарубежных технологий следует отметить еще несколько важных моментов.

Во-первых, импортозамещение не должно сводиться исключительно к репликации зарубежных решений. Вместо этого следует создавать и совершенствовать отечественные технологии. Это позволит обеспечить стимулирование отечественного производства и развитие научных и проектно-конструкторских исследований и разработок.

Во-вторых, на этом пути возникает указанная ранее проблема: зарубежные СВТ содержат массу недокументированных возможностей (НДВ) и как следствие несут угрозу информационной безопасности. При этом проведение достоверного анализа на НДВ для высокотехнологичных СВТ на практике невозможно. Тем не менее, также невозможно полностью отказаться от совместимости отечественных СВТ с зарубежным программным и аппаратным обеспечением. Решение проблемы заключается в том, что архитектура отечественного СВТ должна включать автономное аппаратное средство защиты, полностью контролирующее функционирование информационной системы, обеспечивающее доверие и безопасность.

В-третьих, необходимым условием внедрения этого решения будет востребованность конечными пользователями, причем не только государственными и корпоративными, но и массовыми.

В-четвертых, для того, чтобы некоторое аппаратное или программно-аппаратное средство могло быть использовано в государственном секторе, оно должно быть безопасно, что гарантируется исполнением процедуры сертификации. Эта процедура подтверждает обеспечение целого ряда требований, указанных в законодательстве. При этом существующие отечественные защищенные аппаратные средства обладают рядом недостатков, в сравнении с зарубежными аналогами:

– неполная совместимость с современной массовой аппаратурой. В результате необходима постоянная адаптация к новым устройствам, многообразие которых порождает уникальные требования к каждому из них;

– ограниченная совместимость с приложениями, вызванная необходимостью фиксации версий ПО и прошивок, и отсутствие определенной функциональности, что является следствием выполнения требований по безопасности в рамках процедуры сертификации;

– ограничения на внешнюю среду. После сертификации гарантии безопасности сохраняются только при работе в доверенном окружении. При этом отсутствуют какие-либо гарантии при взаимодействии средства с непроверенным окружением;

– сама по себе необходимость проведения процедуры сертификации фактически означает недоверие к средствам защиты, поскольку ее действенность признается только при работе с ограниченным набором сертифицированных решений.

Следовательно, необходимо разработать архитектуру современного отечественного СВТ, которая учитывает все указанные факторы и, по возможности, лишена указанных недостатков. Для защиты программного обеспечения предлагается использовать так называемую интеграционную парадигму, которая базируется на четырех постулатах:

– контроль всех без исключения взаимодействий в системе;

– инвариантность средств защиты по отношению к приложениям и ресурсам;

– контроль информационных взаимодействий на основе четко определенных правил, составляющих формальную модель;

– оценка безопасности, как текущего состояния системы, так и прогнозирование безопасности будущих состояний.

Рассмотрим использование интеграционной парадигмы для защиты программного обеспечения в системах с архитектурой x86, и затем перейдем к защищенной архитектуре СВТ. Современным СВТ присущ иерархический контроль на аппаратном уровне. Для систем на базе x86 такая иерархия порождает иерархию компонентов программного обеспечения. Однако не все режимы находятся в иерархическом отношении. Некоторые из них находятся на одном уровне. Например, режим ядра может содержать несколько подрежимов в зависимости от разрядности исполняемого кода или используемых данных, что не порождает иерархии контроля. Все эти режимы обладают одинаковыми привилегиями. Тем не менее, сразу три группы аппаратных технологий защиты все-таки порождают иерархию контроля:

– поддержка привилегированного режима и контроль обращения к адресным пространствам;

– аппаратная поддержка виртуализации;

– аппаратная поддержка обособленной доверенной среды.

Основываясь на этих технологиях, применив интеграционную парадигму для защиты программного обеспечения в такой иерархической системе, эксперты пришли к выводу, что реализуя маленький, компактный и очень простой, однако при этом и очень эффективный гипервизор, который размещен на самом-самом низком уровне, в системе возникает возможность обеспечить и контроль, и безопасность для всех вложенных в него систем (рис. 5).

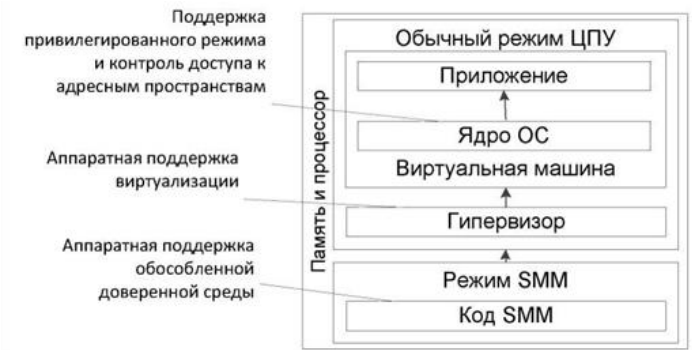

Рис. 5. Иерархия программного обеспечения в системах x86
Fig. 5. Software hierarchy in x86 systems

Такая архитектура еще называется «тонкий» гипервизор или гипервизор одной виртуальной машины. Этот гипервизор не обеспечивает планирования работы виртуальных машин – она у него одна и постоянно работает. Вместо этого он контролирует поведение всех программных компонентов в виртуальной машине, а также все взаимодействия и информационные потоки между программными компонентами в виртуальной машине и реальным аппаратным обеспечением. Предложенный подход был успешно реализован в виде опытного образца. В первых версиях разработанного опытного образца был просто гипервизор и операционная система, а в более новых редакциях за счет добавления поддержки режима SMM, сформирована более сложная конструкция.

Однако смысл от этого не изменился. Это принцип матрешки, когда одни компоненты вкладываются в другие. То, что находится снаружи, полностью контролирует и управляет теми компонентами, что находятся внутри. За счет такой архитектуры система защиты не дает нарушителям решить задачу кибератаки и захвата управления.

Основными преимуществами использования интеграционной парадигмы на примере гипервизора являются (рис. 6):

– управление безопасностью централизовано и сосредоточено в изолированном компоненте;

– возможность контролировать обмен виртуализованной системы с внешней средой с целью устранения скрытых каналов утечки информации;

– поведение виртуализованной системы тождественно поведению реальной системы и ПО работает без изменений;

– возможность имитации внешней среды для виртуализованной системы;

– объем кода гипервизора ничтожен по сравнению с ОС и приложениями;

– эксплуатация уязвимостей не приведет к компрометации изолированных средств защиты;

– сбои в виртуализованной системе могут быть оперативно нейтрализованы путем отката или сброса в исходное состояние.

По мнению экспертов, тот же самый подход применим и к аппаратному обеспечению. Стоит заметить, что Intel технологией ME делает, в какой-то степени, такой же шаг, вынося функции управления на внешний чипсет.

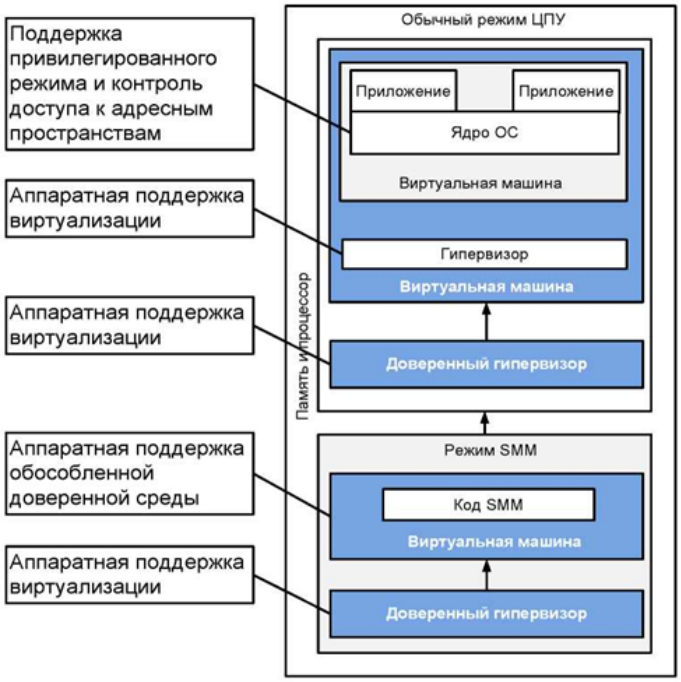

Рис. 6. Использование гипервизора как средства реализации интеграционной парадигмы
Fig. 6. Using the hypervisor as a means of implementing the integration paradigm

Интеграционную парадигму можно предложить для решения задачи построения защищенного средства СВТ (рис. 7).

По сути, предлагается разработать и создать еще один чип, контролирующий Intel ME (чипсет), контролирующий процессор, либо самим научиться контролировать поведение процессора с помощью каких-то своих чипсетов. Таким образом, будет повторяться та самая матрешка, которая и будет обеспечивать безопасность за счет того, что самый внешний слой (тот компонент системы, который контролирует все остальные) оказывается доверенным. По мнению экспертов, такой путь будет оптимальным решением рассматриваемой задачи достижения технологической независимости и цифрового суверенитета. Основными особенностями предлагаемого решения являются:

– последовательная реализация доверенных СВТ путем интеграции импортных и отечественных компонентов;

– контроль всех информационных потоков и взаимодействий со стороны отечественных доверенных средств на всех этапах интеграции;

– использование принципа матрешки – контроль вложенных компонентов и совместимость с внешними технологиями, процессорами и чипсетами;

– универсальный механизм подтверждения доверия с помощью отечественной криптографии;

– полное сохранение функциональных возможностей контролируемых систем.

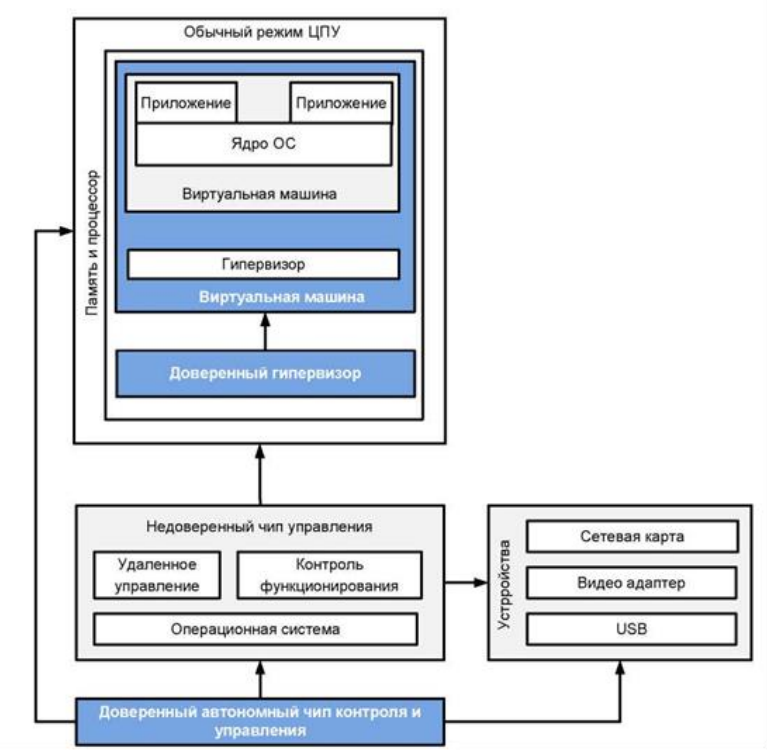

Рис. 7. Использование доверенного автономного чипа контроля и управления как средства реализации интеграционной парадигмы
Fig. 7. Using a trusted autonomous control and management chip as a means of implementing the integration paradigm

Заключение

При выборе технической и/или программной платформы, решая задачи обеспечения совместимости с имеющимся заделом разработанных и применяемых изделий, каждое ведомство планирует свое развитие на перспективу. При этом, целесообразно применять наиболее полный набор сертифицированных решений.

В условиях острого информационного противоборства отечественные разработки не должны сводиться исключительно к репликации зарубежных решений.

СПИСОК ЛИТЕРАТУРЫ:

1. Таненбаум Э., Остин Т. Архитектура компьютера. 6-е изд. – СПб.: Питер, 2019. – 816 с.
2. Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 1: Basic Architecture. URL: https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-1-manual.html (дата обращения: 01.12.2020).
3. Григорий Речистов. Десять имён для одной архитектуры. Блог компании Intel. 13 ноября 2013. URL: https://habrahabr.ru/company/intel/blog/201462/ (дата обращения: 01.12.2020).
4. Miles Murdocca, Vincent Heuring. Computer architecture and organization: an integrated approach / Miles J. Murdocca, Vincent P. Heuring. – Hoboken, N. J.: Wiley, cop. 2007. – 524 p. URL: https://www.researchandmarkets.com/reports/2239761/computer_architecture_and_organization_an (дата обращения: 01.12.2020).
5. Фаулер М. Архитектура корпоративных программных приложений. – М.: Издательский дом "Вильямс", 2006. – 544 с.
6. Rushby, John (1981). "Design and Verification of Secure Systems". 8th ACM Symposium on Operating System Principles. Pacific Grove, California, US. P. 12–21. DOI: https://doi.org/10.1145/800216.806586.

7. Alex Voica – New OmniShield platform implements multi-domain security for connected devices. URL: https://www.imgtec.com/blog/omnishield-multi-domain-security-connected-devices/ (дата обращения: 01.12.2020).
8. Intel® Virtualization Technology for Directed I/O, Architecture Specification, June 2016. URL: https://software.intel.com/sites/default/files/managed/c5/15/vt-directed-io-spec.pdf (дата обращения: 01.12.2020).
9. AMD I/O Virtualization Technology (IOMMU) Specification. URL: https://www.amd.com/system/files/TechDocs/48882_IOMMU_3.05_PUB.pdf (дата обращения: 01.12.2020).
10. Intel® Trusted Execution Technology: Software Development Guide URL: https://cs.technion.ac.il/~cs236376/readings/intel-txt-software-development-guide.pdf (дата обращения: 01.12.2020).
11. Дударев Д.А., Кравцов А.Ю., Полетаев В.М., Полтавцев А.В., Романец Ю.В., Сырчин В.К. Устройство создания доверенной среды для компьютеров специального назначения. Патент RU №2569577 C1. Заявка RU 2014132337/08, 06.08.2014. Опубликовано 27.11.2015. Бюл. №33. МПК G06F21/30, G06F21/50, G06F12/14.
12. ARM Security Technology, Building a Secure System using TrustZone® Technology. URL: https://static.docs.arm.com/genc009492/c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf (дата обращения: 01.12.2020).
13. Intel® Virtualization Technology Specification for the IA-32 Intel® Architecture URL: http://andrewl.dreamhosters.com/library/docs_intel/virtualization_Apr05.pdf (дата обращения: 01.12.2020).
14. Intel VMX technology, G. Lettieri, 28 Oct. 2015 URL: http://lettieri.iet.unipi.it/virtualization/2016/vn05.pdf (дата обращения: 01.12.2020).
15. AMD Secure Virtual Machine Architecture Reference Manual. URL: https://www.mimuw.edu.pl/~vincent/lecture6/sources/amd-pacifica-specification.pdf
16. Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3B: System Programming Guide, Part 2 URL: https://software.intel.com/content/www/us/en/develop/articles/intel-sdm.html (дата обращения: 01.12.2020).
17. Barabanov A.V., Markov A.S., Tsirlov V.L. Information Security Controls Against Cross-Site Request Forgery Attacks on Software Application of Automated Systems. Journal of Physics: Conference Series. 2018. V. 1015. P. 042034. DOI: https://doi.org/10.1088/1742-6596/1015/4/042034.
18. Barabanov A.V., Markov A.S., Tsirlov V.L. Statistics of Software Vulnerability Detection in Certification Testing. Journal of Physics: Conference Series. 2018. V. 1015. P. 042033. DOI: https://doi.org/10.1088/1742-6596/1015/4/042033.
19. Borzykh S., Markov A., Tsirlov V., Barabanov A. Detecting Code Security Breaches by Means of Dataflow Analysis. In CEUR Workshop Proceedings, 2017, Vol-2081. P. 15–20. URL: http://ceur-ws.org/Vol-2081/paper04.pdf (дата обращения: 01.12.2020).

REFERENCES:

[1] Andrew S. Tanenbaum, Todd Austin. Structured computer organization. 6 edition. – Saint Petersburg: Piter, 2019. – 816 p. (in Russian).
[2] Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 1: Basic Architecture. URL: https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-1-manual.html (accessed: 01.12.2020).
[3] Gregory Rechistov. Ten names for a single architecture. Intel's blog. November 13, 2013. URL: https://habrahabr.ru/company/intel/blog/201462/ (accessed: 01.12.2020).
[4] Miles J. Murdocca, Vincent P. Heuring. Computer architecture and organization: an integrated approach – Hoboken, N. J.: Wiley, cop. 2007. – 524 p. URL: https://www.researchandmarkets.com/reports/2239761/computer_architecture_and_organization_an (accessed: 01.12.2020)
[5] Fowler M. Architecture of corporate software applications M.: Williams Publishing House, 2006. – 544 p. (in Russian).
[6] Rushby, John (1981). "Design and Verification of Secure Systems". 8th ACM Symposium on Operating System Principles. Pacific Grove, California, US. P. 12–21. DOI: https://doi.org/10.1145/800216.806586

[7] Alex Voica – New OmniShield platform implements multi-domain security for connected devices. URL: https://www.imgtec.com/blog/omnishield-multi-domain-security-connected-devices/ (accessed: 01.12.2020).
[8] Intel® Virtualization Technology for Directed I/O, Architecture Specification, June 2016. URL: https://software.intel.com/sites/default/files/managed/c5/15/vt-directed-io-spec.pdf (accessed: 01.12.2020)
[9] AMD I/O Virtualization Technology (IOMMU) Specification. URL: https://www.amd.com/system/files/TechDocs/48882_IOMMU_3.05_PUB.pdf (accessed: 01.12.2020)
[10] Intel® Trusted Execution Technology: Software Development Guide. URL: https://cs.technion.ac.il/~cs236376/readings/intel-txt-software-development-guide.pdf (accessed: 01.12.2020).
[11] Dudarev D.A., Kravtsov A.Yu., Poletaev V.M., Poltavtsev A.V., Romanets Ju.V., Syrchin V.K. Device to create trusted execution environment for special-purpose computers. Patent RU 2569577 C1. Application RU 2014132337/08, 06.08.2014. Published 27.11.2015, bull. No. 33. IPC G06F21/30, G06F21/50, G06F12/14. (in Russian).
[12] ARM Security Technology, Building a Secure System using TrustZone® Technology. URL: https://static.docs.arm.com/genc009492/c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf (accessed: 01.12.2020).
[13] Intel® Virtualization Technology Specification for the IA-32 Intel® Architecture. URL: http://andrewl.dreamhosters.com/library/docs_intel/virtualization_Apr05.pdf (accessed: 01.12.2020).
[14] Intel VMX technology, G. Lettieri, 28 Oct. 2015 URL: http://lettieri.iet.unipi.it/virtualization/2016/vn05.pdf (accessed: 01.12.2020).
[15] AMD Secure Virtual Machine Architecture Reference Manual URL: https://www.mimuw.edu.pl/~vincent/lecture6/sources/amd-pacifica-specification.pdf (accessed: 01.12.2020).
[16] Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3B: System Programming Guide, Part 2. URL: https://software.intel.com/content/www/us/en/develop/articles/intel-sdm.html (accessed: 01.12.2020).
[17] Barabanov A.V., Markov A.S., Tsirlov V.L. Information Security Controls Against Cross-Site Request Forgery Attacks on Software Application of Automated Systems. Journal of Physics: Conference Series. 2018. V. 1015. P. 042034. DOI: https://doi.org/10.1088/1742-6596/1015/4/042034.
[18] Barabanov A.V., Markov A.S., Tsirlov V.L. Statistics of Software Vulnerability Detection in Certification Testing. Journal of Physics: Conference Series. 2018. V. 1015. P. 042033. DOI: https://doi.org/10.1088/1742-6596/1015/4/042033.
[19] Borzykh S., Markov A., Tsirlov V., Barabanov A. Detecting Code Security Breaches by Means of Dataflow Analysis. In CEUR Workshop Proceedings, 2017, Vol-2081. P. 15–20. URL: http://ceur-ws.org/Vol-2081/paper04.pdf (accessed: 01.12.2020).

Поступила в редакцию – 01 декабря 2020 г. Окончательный вариант – 14 января 2021 г.
Received – December 01, 2020. The final version – January 14, 2021.