Применение технологии виртуализации для защиты программного обеспечения и данных в системах добровольческих вычислений

Н. П. Храпов

Институт проблем передачи информации Российской академии наук, Российская Федерация, 127051, Москва, Большой Каретный пер., 19

Для цитирования: Храпов Н. П. Применение технологии виртуализации для защиты программного обеспечения и данных в системах добровольческих вычислений // Вестник Санкт-Петербургского университета. Прикладная математика. Информатика. Процессы управления. 2020. Т. 16. Вып. 1. С. 62–72. <https://doi.org/10.21638/11702/spbu10.2020.106>

При решении вычислительных задач научные коллективы часто применяют вычислительные ресурсы, которые предоставлены различными организациями или индивидуальными волонтерами. Программное обеспечение (ПО) и данные, используемые научным коллективом, могут являться предметом интеллектуальной собственности. По этой причине становится актуальным вопрос защиты ПО и данных от потенциальных злоумышленников, которые могут подключиться к проекту под видом добровольцев. Приводится обзорный анализ существующих подходов к преодолению данной проблемы в общем случае. Детально рассмотрены возможности инфраструктуры BOINC, актуальные для разработки разных сценариев защиты информации. Предложена концепция защиты данных от несанкционированного доступа, основанная на интеграции клиентской стороны BOINC с системой VirtualBox.

Ключевые слова: защита информации, грид-системы, добровольные вычисления, BOINC, виртуализация, VirtualBox.

1. Введение. Технологии грид-систем из персональных компьютеров (ГСПК) и добровольных вычислений постепенно набирают популярность начиная с конца XX в. Наблюдения показали, что в подавляющем большинстве случаев уровень загрузки центрального процессора типового домашнего или офисного компьютера не превышает 10 %. Это означает, что более чем 90 % процессорного времени работающего компьютера не задействовано для работы. Технологии ГСПК позволяют использовать незадействованное время центрального процессора (ЦП) для проведения вычислений, связанных с решением полезных научных задач. Наиболее распространенной реализацией ГСПК в настоящее время является система Berkeley Open Infrastructure for Network Computing (BOINC) [1].

Перед человечеством стоит множество глобальных задач, решение которых требует больших вычислительных мощностей. К их числу можно отнести моделирование белковых молекул для создания средств по борьбе с опасными вирусами и раковыми заболеваниями. Значительные вычислительные ресурсы также необходимы для решения задач в области науки о материалах, предсказания климата и погоды, физики элементарных частиц и других важных областях науки.

Инфраструктура BOINC позволяет проводить вычисления на персональном компьютере (ПК), который был предоставлен для этого добровольцем. Они могут производиться на домашнем или офисном ПК параллельно выполнению основных задач или во время отсутствия пользовательской активности. Также система BOINC дает

возможность использовать для вычислений выделенные ПК и серверы, в отдельных случаях кластеры. Запуск вычислительных заданий и обмен данными осуществляются в автоматическом режиме посредством сети Интернет. Для того чтобы предоставить свой ПК для научных вычислений, его владелец устанавливает специальную программу BOINC-клиент. В настройках BOINC-клиента указывается проект добровольных вычислений, задачи которого волонтер хочет считать на своем ПК. После необходимых настроек программа BOINC-клиент автоматически скачивает с сервера проекта вычислительное задание, запускает его и результат вычислений пересылает обратно на сервер проекта.

2. Анализ рисков в сетях добровольных вычислений. Рассмотрим основные типы угроз информационной безопасности, применительно к проектам добровольных вычислений.

1. *Угроза конфиденциальности*, т. е. несанкционированного доступа к данным. В вычислительных сетях, основанных на BOINC, злоумышленник под видом волонтера способен подключиться к проекту добровольных вычислений и получить полный доступ к программному обеспечению (ПО) и данным задания. В ряде случаев такая информация может иметь лицензионные ограничения. В настоящее время система BOINC не предусматривает какого-либо механизма защиты информации от такого типа угроз. Предложенная в рамках данной статьи концепция защиты информации ориентирована в первую очередь на преодоление этого типа угроз.

2. *Угроза целостности*, т. е. несанкционированной модификации, дополнения или уничтожения данных. Применимо к вычислительным сетям, основанным на BOINC, такой вид угрозы состоит в том, что злоумышленник под видом волонтера способен скачать вычислительное задание с сервера проекта и осуществить подмену результата для причинения намеренного вреда. Для противодействия этому виду угроз система BOINC предусматривает механизм проверки (валидации) полученных от вычислительного узла результатов. Один из подходов к валидации в BOINC состоит в проблемно-ориентированном анализе содержимого полученного результата на соответствие специальным критериям. Другой подход — в механизме репликации и кворума. При нем n копий (реплик) одного вычислительного задания рассылаются на различные вычислительные узлы случайным образом. После этого сравниваются результаты, полученные с разных вычислительных узлов. Если m результатов совпадают (достижение кворума), то общий результат считается истинным (каноническим). Концепция, предложенная в рамках настоящей статьи, допускает шифрование промежуточных данных, тем самым способна существенно повысить уровень защиты результатов вычислений от потенциальной подмены.

3. *Угроза доступности (Denial of Service — DoS-атака)*, т. е. ограничение или блокирование доступа к данным, целенаправленный вывод системы из работоспособного состояния. Применимо к технологии BOINC такая угроза состоит в том, что злоумышленник способен имитировать большое число клиентских запросов заданий с сервера с целью вывода его из строя. Для противодействия данному типу угроз система BOINC предусматривает ряд способов защиты, таких как проверка электронного адреса пользователя, ввод цифр с изображения, блокировка ненадежных узлов и пользователей. Защита от DoS-атак является предметом отдельных исследований и здесь не рассматривается.

В этой статье разработаны методы изоляции ПО и данных от потенциальных злоумышленников в рамках публичных BOINC-проектов. Использование разработанных методов в каждом конкретном случае может потребовать согласование как

с владельцем прав на интеллектуальную собственность, так и с пользователями инфраструктуры BOINC, предоставляющими ресурс для вычислений.

3. Обзор существующих подходов. Проблема изоляции ПО и данных от потенциальных злоумышленников на стороне вычислительного узла применительно к системе BOINC ранее не рассматривалась. Но к настоящему времени опубликован ряд научных работ, посвященных данной тематике, относительно других вычислительных систем.

В [2] предложен подход к защите информации, основанный на шифровании разделов виртуальной машины (ВМ). Рассматриваемый подход совместим с системами виртуализации Xen и VMware. Однако вычислительная платформа BOINC не поддерживает интеграцию с данными гипервизорами. По этой причине реализация описанного подхода невозможна для вычислительных инфраструктур, основанных на BOINC. Но идея защиты информации посредством шифрования разделов ВМ заслуживает внимания применительно и к технологии BOINC.

Стандартный сценарий использования платформы BOINC предполагает возможность создания глобальных вычислительных проектов, к которым можно подключить любой компьютер посредством сети Интернет. Но для ряда вычислительных задач достаточно применять суммарные вычислительные мощности ПК и серверов организации, занимающейся решением прикладных задач. Важным преимуществом такого подхода является возможность максимально эффективного использования имеющихся в наличии вычислительных ресурсов. В [3] рассматриваются возможности применения системы BOINC на внутрикорпоративных ресурсах в рамках одной или нескольких организаций.

Для многих научных организаций совокупная мощность представляющих доверие ПК и имеющихся в распоряжении кластерных систем способна предоставить необходимый для решения поставленных задач ресурс. В [4, 5] предлагается подход к созданию гибридной вычислительной инфраструктуры, основанной на интеграции кластеров и систем добровольных вычислений, полученных с помощью BOINC.

4. Способы запуска приложений, предоставляемые инфраструктурой BOINC. Стандартный способ запуска приложений состоит в том, что BOINC-клиент скачивает исполняемый файл приложения вместе с файлами данных с сервера проекта. Затем приложение запускается на вычислительном узле. После окончания расчетов файлы результатов отправляются на сервер для дальнейшей обработки. Процесс работы приложения на стороне вычислительного узла контролируется программой BOINC-клиент. Данный контроль подразумевает взаимодействие между приложением и запустившей его средой. Для осуществления этого необходимо, чтобы приложение было собрано и структурировано специальным образом для запуска в BOINC-среде. Взаимодействие приложения и среды запуска осуществляется со стороны приложения вызовом специальных функций, предоставляемых библиотеками BOINC-API или DC-API [6].

На практике в ряде случаев отсутствует возможность специальной разработки приложений для запуска в среде BOINC. Во-первых, может использоваться приложение с закрытым исходным кодом. Во-вторых, изменение структуры программы может быть чрезмерно трудоемким. Для адаптации уже существующих приложений разработчиками системы BOINC было создано специальное приложение wtapper [7]. Оно запускается на стороне вычислительного узла BOINC-клиентом и осуществляет запуск неадаптированного приложения и вызовы необходимых функций для взаимодействия с программой BOINC-клиент.

Как при запуске приложения, специально разработанного для BOINC, так и при технологии wrapper запускаемый файл приложения и файлы с данными помещаются в файловую систему ПК волонтера явным образом. Потому единственным способом защиты информации в рамках описанных выше подходов является интеграция методов шифрования в исходный код приложения, которая в ряде случаев требует больших трудовых затрат или невозможна. В рамках этой работы предложены способы защиты ПО и данных, которые используют существующий исполняемый файл приложения без внесения изменений в исходный код.

В настоящее время платформа BOINC поддерживает возможность запуска приложений на стороне вычислительного узла внутри ВМ. Для этого программа BOINC-клиент поддерживает интеграцию с системой виртуализации VirtualBox. При своем запуске BOINC-клиент проверяет наличие программы VirtualBox на вычислительном узле. При наличии такой программы запрос заданий содержит прикрепленную информацию о версии установленного VirtualBox. Обработывая этот запрос, BOINC-сервер отправляет задание, использующее систему виртуализации (при наличии такого задания на сервере). Разработчики системы BOINC предложили типовой сценарий запуска приложений внутри виртуальной среды на стороне вычислительного узла, имеющий следующие особенности:

1. Приложение состоит из программы VirtualBox-wrapper. Данная программа написана с помощью BOINC-API. Также в состав приложения входит файл жесткого диска ВМ.

2. Стандартный сценарий применения технологии виртуализации подразумевает, что внутри ВМ установлены дополнения гостевой операционной системы (ОС). На этапе отправки на вычислительный узел виртуальная ОС находится в выключенном состоянии.

3. BOINC-клиент запускает программу VirtualBox-wrapper, которая создает ВМ, прикрепляет к ней виртуальный жесткий диск из состава приложения и запускает ее. После загрузки виртуальной ОС создается общая папка, в которую помещаются данные приложения. Затем, используя их, проблемно-ориентированное приложение запускается под управлением виртуальной ОС. После окончания вычислений ВМ выключается, а файлы результатов из общего каталога передаются на BOINC-сервер для дальнейшей обработки.

Подход, основанный на использовании VirtualBox-wrapper, изначально не предназначен для защиты ПО и данных. В частности, в рамках этого подхода предполагается наличие ПО и данных как в файловой системе вычислительного узла, так и внутри гостевой ОС. Таким образом, потенциальный злоумышленник способен получить необходимую информацию из файловой системы или из образа ВМ. Если использовать шифрование файловой системы ВМ совместно с VirtualBox-wrapper, то потребуются ввод ключей шифрования на этапе загрузки. Так как загрузка виртуальной ОС осуществляется на стороне вычислительного узла в автоматическом режиме, то система управления загрузкой должна будет извлечь ключи шифрования из некоторого места хранения. Это создает угрозу доступа к информации потенциальным злоумышленникам посредством извлечения ключей шифрования из места хранения. Таким образом, надежно защитить данные, расположенные внутри файловой системы ВМ, применяя VirtualBox-wrapper, невозможно.

Описанная в настоящей статье концепция ориентирована на расширение возможностей подхода к использованию виртуализации, предложенного ранее разработчиками системы BOINC. Предлагаемые методы позволяют осуществить защиту ПО

и данных от доступа потенциальных злоумышленников при применении новых возможностей системы виртуализации VirtualBox. Обзор этих возможностей приведен в п. 5.

5. Обзор возможностей платформы VirtualBox. Сфера применения технологий виртуализации постоянно расширяется. Большое распространение получили серверы виртуализации, предоставляющие пользователю спектр возможностей различного вида хостинга. Дальнейшим этапом расширения области использования технологий виртуализации стало внедрение кластеров (пулов) серверов виртуализации. Информационные системы пулов виртуализации легли в основу создания концепции облачных вычислений Infrastructure as a Service (IaaS). Одним из основных требований к выделенному серверу является способность непрерывного функционирования на протяжении долгого времени. Проведение периодических технических работ на серверах виртуализации приводит к необходимости периодического выключения и переноса на другой сервер запущенных ВМ. Для решения такой проблемы в рамках современных систем облачных вычислений и технологий виртуализации была разработана возможность горячей миграции ВМ. Данный способ миграции означает, что ВМ, запущенная на сервере виртуализации, может быть без выключения перенесена на другой сервер.

Для системы VirtualBox эта технология получила название teleporting и была внедрена, начиная с версии 3.1, выпущенной в 2009 г. Функция teleporting позволяет перенести по локальной сети работающую ВМ с одного сервера виртуализации на другой без выключения самой ВМ. Описываемый в настоящей статье подход к защите информации не использует teleporting явным образом, однако применяет такие же функциональные возможности и имеет ряд сходств.

Все современные системы виртуализации (включая VirtualBox) обладают возможностью сохранения состояния ВМ. Данный процесс аналогичен постановке на паузу: текущее состояние жесткого диска, оперативной памяти и процессорных регистров сохраняются в специальных файлах. Через некоторое время состояние ВМ может быть восстановлено на основе информации, полученной из таких файлов. Наиболее важной особенностью является то, что файлы состояния ВМ могут быть скопированы на другую хост-систему, и ВМ может продолжить функционирование уже на ней. Учитывая то, что возможно создание неограниченного числа копий файлов состояния ВМ, то и копии сеанса ее работы могут продолжиться на неограниченном количестве хост-систем. Аналогичным образом в рамках предлагаемого подхода вычислительная ВМ изначально запускается и настраивается на ПК администратора проекта, а затем продолжает свое функционирование на вычислительном узле волонтера.

Запуск ВМ на вычислительном узле и дальнейшее управление осуществляются в автоматическом режиме. Рассмотрим следующие основные интерфейсы управления ВМ, предоставляемые системой VirtualBox:

- *Графический интерфейс пользователя (GUI).* Данный интерфейс максимально удобен в использовании при минимальных функциональных возможностях. VirtualBox GUI показал свою практичность для индивидуального применения ВМ, однако не предоставляет возможностей автоматизированного запуска и управления ВМ. Таким образом, GUI не удовлетворяет необходимым для реализации подхода требованиям.

- *Интерфейс командной строки (CLI).* Он позволяет пользователю применять все основные возможности системы виртуализации, а также проводить автоматизи-

рованный запуск и управление ВМ. С помощью VirtualBox CLI возможна быстрая разработка платформозависимых скриптов управления.

- *VirtualBox API*. Программный интерфейс предоставляет возможность разработки ПО, способного осуществлять управление ВМ. При этом API VirtualBox не позволяет осуществить прямое управление средой виртуализации пользователем. С помощью VirtualBox API реализовано приложение VirtualBox-wrapper, разработанное создателями платформы BOINC.

Для реализации автоматизированного управления ВМ в рамках предложенного подхода можно использовать как CLI, так и API.

6. Концепция применения технологии виртуализации для защиты ПО и данных. Методы, предложенные в настоящей работе, основаны на сохранении состояния ВМ и восстановлении рабочего состояния на другой вычислительной хост-системе. Запускаемое приложение помещается внутрь файловой системы ВМ на специальный зашифрованный раздел. Расчеты производятся под управлением гостевой ОС. На стороне вычислительного узла ВМ находится в состоянии, не допускающем вмешательства и доступа к внутренней информации со стороны пользователя хост-системы. В частности, невозможен доступ к консоли ВМ без ввода идентификационной информации. Принципиальная отличительная особенность такого подхода состоит в том, что на стороне вычислительного узла осуществляется не загрузка виртуальной ОС, а восстановление состояния работающей ВМ. В сохраненном состоянии зашифрованные разделы уже примонтированы к гостевой ОС. Отсутствие необходимости монтирования шифрованных разделов на стороне вычислительного узла снижает потенциальную угрозу похищения ключей шифрования в процессе передачи данных.

Контейнером для транспортировки входных/выходных данных для расчетов выступает виртуальный жесткий VDI-диск, который подключается к ВМ в качестве внешнего накопителя после восстановления состояния. Если используемые программой данные необходимо защитить, то они помещаются в файловую систему виртуального диска внутрь зашифрованного архива. Подготовка файла с данными осуществляется на стороне BOINC-сервера.

За управление вычислениями отвечают следующие программы:

1. На вычислительном узле BOINC-клиентом запускается *программа управления ВМ (ПУВМ)*, которая отвечает за восстановление состояния, подключение виртуальных носителей с данными и завершение расчетов. ПУВМ является BOINC-приложением, которое может быть реализовано как при помощи технологии wrapper, так и нативных BOINC-API или DC-API.

2. Внутри ВМ процесс вычислений контролируется *сценарием управления расчетами (СУР)*. Данная программа запускается администратором проекта на этапе подготовки ВМ. СУР осуществляет монтирование файловой системы виртуального носителя данных, копирование входных данных приложения на зашифрованный раздел (при необходимости их дешифровку) и запуск основного решателя. После окончания вычислений СУР копирует их результаты на виртуальный носитель и выключает ВМ. Для обеспечения целостности данных и отказоустойчивости СУР также способен производить сбор логов и подписывание передаваемых данных.

Основные этапы жизненного цикла ВМ отображены на рис. 1. Приведенная диаграмма подчеркивает, что после этапа подготовки создается множество копий ВМ для работы с разными заданиями. UML-диаграмму времени совокупной работы ВМ, СУР, ПУВМ и BOINC-клиента для отдельного задания отображает рис. 2.

Рис. 1. Этапы жизненного цикла VM

Рис. 2. UML-диаграмма времени для VM, СУР, ПУВМ, BOINC-клиента

Рассмотрим основные этапы жизненного цикла VM.

Этап подготовки осуществляется на VM администратора проекта. На этапе подготовки создается новая VM. На жестком диске VM создается зашифрованный раздел. ОС VM может быть помещена как на зашифрованный раздел, так и на отдельный. Установка ОС на зашифрованный раздел обеспечивает большую степень защиты информации, однако при данном подходе минимальный объем передаваемых данных составит 1.5 Гб. Если ОС установлена на обычный раздел, а зашифрованный содержит только лишь файлы и данные, являющиеся предметом интеллектуальной собственности, то минимальный объем приложения будет составлять 300 МБ. Различия в объеме передаваемых данных обусловлены тем, что зашифрованные данные в меньшей степени поддаются сжатию. Также на этапе подготовки производится загрузка виртуальной ОС, монтирование шифрованного раздела, содержащего файлы приложения и прочую информацию, имеющую лицензионные ограничения. После этого администратор проекта запускает внутри VM СУР в фоновом режиме и осуществляет пользовательский выход (logout) из ОС VM. Окончанием этапа подготовки является сохранение состояния VM.

На **этапе транспортировки** копии файлов ВМ в сохраненном состоянии передаются по сети на вычислительные узлы. Для системы VirtualBox ВМ в сохраненном состоянии состоит из двух основных файлов: жесткого диска и файла состояния, BOINC-приложения из файлов ВМ и ПУВМ. Для экономии сетевого трафика BOINC-инфраструктура использует промежуточное сжатие. Файлы приложения передаются на вычислительный узел один раз и могут быть применены для всех заданий, запускаемых на этом узле. Предлагаемый подход не накладывает ограничений на количество узлов, на которых ВМ способна продолжить функционирование. Ограничения могут быть связаны только с количеством доступных узлов и с физическими возможностями сервера и каналов связи.

После транспортировки ВМ на вычислительном узле запускается ПУВМ.

На **этапе вычислений** ПО объединяется с конкретными данными вычислительного задания. Для этого ПУВМ создает рабочую копию сохраненного состояния ВМ и осуществляет восстановление состояния. При этом снимок оперативной памяти ВМ загружается в оперативную память компьютера, и ВМ продолжает свою работу, начиная с момента остановки. Данные для расчетов вычислительного задания содержатся внутри отдельного VDI-образа диска. Виртуальный носитель информации пересылается на вычислительный узел в качестве данных BOINC-задания. После восстановления состояния ПУВМ подключает VDI-образ с данными к запущенной ВМ в качестве внешнего диска. Работа ПУВМ на вычислительном узле синхронизирована с работой СУР внутри ВМ таким образом, что основные действия СУР выполняет после подключения носителя с данными.

Теперь рассмотрим более подробно последовательность действий, выполняемых СУР с момента запуска.

1. Пауза. Сохранение состояния, транспортировка ВМ и восстановление состояния осуществляются в тот момент, когда СУР внутри ВМ находится в состоянии паузы. Она начинается на этапе подготовки ВМ на компьютере администратора проекта, а заканчивается на вычислительном узле. После восстановления состояния ВМ внутренний скрипт некоторое время будет пребывать в состоянии бездействия. За это время к работающей ВМ будет подключен внешний виртуальный диск с данными. Важно отметить, что время на часах ВМ, которая некоторое время пребывала в сохраненном состоянии, может сильно отличаться от общемирового времени. Период ожидания в системе времени ВМ выбирается администратором и может быть от нескольких секунд до 1 мин.

2. После выхода из состояния паузы осуществляются монтирование внешнего виртуального носителя ВМ и перенос данных с него на зашифрованный раздел. При необходимости передаваемые данные могут быть дешифрованы.

3. После подготовки данных сценарий управления вычислениями выполняет запуск приложения.

4. После окончания расчетов результаты вычислений копируются на внешний виртуальный носитель (при необходимости данные шифруются). После чего сценарий управления вычислениями осуществляет выключение ВМ.

Пункты 2–4 на UML-диаграмме времени (рис. 2) обозначены как **подготовка и проведение расчетов**.

Рассмотрим, каким образом происходит защита информации на разных этапах жизненного цикла данных.

Защита данных в процессе транспортировки (data at transport) и защита данных на носителе информации (data at rest). В рамках предлагаемой концепции ПО

передается внутри сохраненного состояния ВМ на зашифрованном разделе. Для получения доступа к ПО следует осуществить дешифрование раздела. В сохраненном состоянии зашифрованный раздел примонтирован в гостевой ОС. Для доступа к ПО злоумышленнику необходимо ввести ключ шифрования или получить доступ к управлению гостевой ОС. Возможности современных ОС способны сделать этот доступ невозможным или чрезвычайно затруднительным. Данные для вычислений передаются внутри виртуального носителя информации. При необходимости они могут быть помещены внутрь зашифрованного архива. Для получения доступа к ним злоумышленнику будет нужно расшифровать архив. Современные способы создания зашифрованных архивов делают такую задачу практически нереализуемой.

Защита данных в ходе вычислительного процесса (data at memory). Программный код и данные в процессе вычислений помещаются в оперативную память ВМ. Теоретически посредством анализа снимка оперативной памяти ВМ злоумышленник может получить некоторые части исполняемого кода программы и данных для вычислений. На практике такой анализ является чрезвычайно трудоемкой задачей, требующей высокой квалификации. В большинстве случаев стоимость работы по анализу снимка оперативной памяти для получения несанкционированного доступа к данным превышает стоимость лицензионного приобретения ПО и данных, что делает подобный взлом нецелесообразным.

7. Заключение. В статье представлена концепция защиты информации от несанкционированного доступа в проектах добровольных вычислений, основанных на технологии BOINC. Рассмотрены работы, описывающие решение проблемы изоляции ПО и данных в отличных от BOINC вычислительных сетях. Проведен обзор возможностей платформы BOINC по интеграции с системой виртуализации VirtualBox. Также охарактеризованы нетривиальные возможности VirtualBox по автоматизации управления и миграции ВМ в сохраненном состоянии. Описание концепции защиты информации носит общий характер. Приведены основные идеи, на которых построены предлагаемые подходы. Составлено описание необходимых для реализации программных компонентов, рассматриваются их жизненный цикл и принцип совокупной работы.

Предложенная концепция защиты информации является универсальной и может быть использована для адаптации широкого спектра вычислительных задач к проектам добровольных вычислений. ПО для реализации подходов и алгоритмы шифрования могут быть различным образом модифицированы для соответствия специфике решаемой задачи и требованиям разработчика используемого ПО. Также представленная концепция совместима с разными методами обеспечения информационной безопасности, на основе которых может быть создана система защиты данных, устойчивая к разнообразным типам информационных атак.

В настоящее время происходит активное внедрение предложенной концепции в проект добровольных вычислений USPEX@HOME [8]. Сейчас проект находится в стадии бета-тестирования. Тематика проекта имеет важное прикладное значение в области моделирования структуры вещества и создания новых материалов. В разное время к проекту подключалось более 1.5 тыс. вычислительных узлов. В рамках проекта USPEX@HOME на ВМ волонтеров запускается программа VASP, исполняемый файл и данные которой являются предметом интеллектуальной собственности. Использование программы-решателя VASP и сопутствующих данных было согласовано с разработчиком.

Литература

1. Berkeley Open Infrastructure for Network Computing // BOINC: офиц. сайт. <https://boinc.berkeley.edu> (дата обращения: 20.12.2019 г.)
2. Li Ch., Raghunathan A., Jha N. K. Secure virtual machine execution under an untrusted management OS // IEEE 3rd International Conference on Cloud Computing. Miami, 2010. P. 172–179.
3. Ivashko E. E. Enterprise desktop grids // CEUR Workshop Proceedings. 2015. Vol. 1502. P. 16–21.
4. Afanasiev A. P., Bychkov I. V., Manzyuk M. O., Posypkin M. A., Semenov A. A., Zaikin O. S.. Technology for integrating idle computing cluster resources into volunteer computing projects // 5th International Workshop on Computer Science and Engineering: Information Processing and Control Engineering, WCSE 2015-IPCE. Moscow, 2015. P. 109–114.
5. Afanasiev A. P., Bychkov I. V., Zaikin O. S., Manzyuk M. O., Posypkin M. A., Semenov A. A. Concept of a multitask grid system with a flexible allocation of idle computational resources of supercomputers // Journal of Computer and Systems Sciences International. 2017. Vol. 56. N 4. P. 701–707.
6. Marosi A. C., Kovacs J., Kacsuk P. K. Towards a volunteer cloud system // Future Generation Computer Systems. 2013. Vol. 29(6). P. 1442–1451.
7. WrapperApp — BOINC // BOINC: офиц. сайт. <https://boinc.berkeley.edu/trac/wiki/WrapperApp> (дата обращения: 18.03.2019 г.)
8. Khrapov N. P., Rozen V. V., Samtsevich A. I., Posypkin M. A., Sukhomlin V. A., Oganov A. R. Using virtualization to protect the proprietary material science applications in volunteer computing // Open Engineering. 2018. Vol. 8. P. 57–60.

Статья поступила в редакцию 16 апреля 2019 г.

Статья принята к печати 13 февраля 2020 г.

Контактная информация:

Храпов Николай Павлович — и.о. науч. сотр., аспирант; nkhrapov@gmail.com

Using virtualization technologies to protect software and data in volunteer computing systems

N. P. Khrapov

Institute for Information Transmission Problems of the Russian Academy of Sciences,
19, Bolshoy Karetny per., Moscow, 127051, Russian Federation

For citation: Khrapov N. P. Using virtualization technologies to protect software and data in volunteer computing systems. *Vestnik of Saint Petersburg University. Applied Mathematics. Computer Science. Control Processes*, 2020, vol. 16, iss. 1, pp. 62–72.
<https://doi.org/10.21638/11702/spbu10.2020.106> (In Russian)

When solving computational problems, research teams often use computational resources that are provided for this by various organizations or individual volunteers. Software and data used by the research team may be a subject of intellectual property. For this reason, it becomes relevant to protect software and data from potential intruders who can join the project under the guise of volunteers. The article provides an overview and an analysis of existing approaches to overcome this problem in the general case. The possibilities of the BOINC infrastructure, relevant for the development of various information security scenarios, are considered in detail. The concept of protecting data from unauthorized access, based on the integration of the client side of BOINC with the VirtualBox system, is proposed.

Keywords: protection of information, grid-systems, volunteer computing, BOINC, virtualization, VirtualBox.

References

1. Berkeley Open Infrastructure for Network Computing. Available at: <https://boinc.berkeley.edu> (accessed: 20.12.2019).

2. Li Ch., Raghunathan A., Jha N. K. Secure virtual machine execution under an untrusted management OS. *IEEE 3rd International Conference on Cloud Computing*. Miami, 2010, pp. 172–179.
3. Ivashko E. E. Enterprise desktop drids. *CEUR Workshop Proceeding*, 2015, vol. 1502, pp. 16–21.
4. Afanasiev A. P., Bychkov I. V., Manzyuk M. O., Posypkin M. A., Semenov A. A., Zaikin O. S. Technology for integrating idle computing cluster resources into volunteer computing projects. *5th International Workshop on Computer Science and Engineering: Information Processing and Control Engineering, WCSE 2015-IPCE*. Moscow, 2015, pp. 109–114.
5. Afanasiev A. P., Bychkov I. V., Zaikin O. S., Manzyuk M. O., Posypkin M. A., Semenov A. A. Concept of a multitask grid system with a flexible allocation of idle computational resources of supercomputers. *Journal of Computer and Systems Sciences International*, 2017, vol. 56, no. 4, pp. 701–707.
6. Marosi A. C., Kovacs J., Kacsuk P. K. Towards a volunteer cloud system. *Future Generation Computer Systems*, 2013, vol. 29(6), pp. 1442–1451.
7. *WrapperApp BOINC*. Available at: <https://boinc.berkeley.edu/trac/wiki/WrapperApp> (accessed: 18.03.2019).
8. Khrapov N. P., Rozen V. V., Samtsevich A. I., Posypkin M. A., Sukhomlin V. A., Oganov A. R. Using virtualization to protect the proprietary material science applications in volunteer computing. *Open Engineering*, 2018, vol. 8, pp. 57–60.

Received: April 16, 2019.

Accepted: February 13, 2020.

Authors' information:

Nikolay P. Khrapov — Researcher, Graduate Student; nkhrapov@gmail.com