УДК 519.688
DOI: 10.25559/SITITO.019.202302.292-297

Оригинальная статья

Сравнительные характеристики платформ контейнеризации Docker и Singularity

А. С. Бондяков*, А. О. Кондратьев
Международная межправительственная организация Объединенный институт ядерных исследований, г. Дубна, Российская Федерация
Адрес: 141980, Российская Федерация, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6
* aleksey@jinr.ru

Аннотация

На сегодняшний день для решения задач различной сложности требуются высокие мощности и применяются повышенные требования к оборудованию. Это задачи инженерного проектирования, большой спектр задач научных вычислений, системы моделирования, анализ данных и т. д. Как правило, любая отрасль имеет задачи, для выполнения которых требуются высокопроизводительные вычисления. Для решения таких задач используются высокопроизводительные вычислительные системы, основу которых составляют технологии параллельных вычислений, технологии виртуализации и контейнеризации.

Параллельные вычисления позволяют значительно увеличить производительность вычислительных систем при выполнении одной и той же программы. Технологии виртуализации позволяют создать на одном физическом компьютере несколько виртуальных машин, каждая из которых работает под управлением отдельной операционной системы. Контейнеры — это одно из самых перспективных направлений развития современных информационных технологий. В настоящее время контейнеры используются во многих областях, включая веб-приложения (например, Apache Tomcat), мобильные приложения, системы управления базами данных, а также для развертывания серверных сред. По мнению разработчиков, контейнеры позволят упростить разработку и развернуть высокопроизводительные вычислительные системы, значительно повысить их производительность и надежность.

В данной статье представлены сравнительные характеристики платформ контейнеризации Docker и Singularity в аспекте решения таких проблем, как портируемость кода между локальной средой выполнения и промышленной средой приложения. Приведены примеры использования данных платформ для решения задач, которые являются актуальными в настоящее время. Рассмотрены наиболее важные моменты работы данных платформ применительно к высокопроизводительным вычислительным кластерам.

Ключевые слова: контейнерная платформа, виртуализация, высокопроизводительные кластеры, программные пакеты

Конфликт интересов: авторы заявляют об отсутствии конфликта интересов.

Для цитирования: Бондяков А. С., Кондратьев А. О. Сравнительные характеристики платформ контейнеризации Docker и Singularity // Современные информационные технологии и ИТ-образование. 2023. Т. 19, № 2. С. 292-297. doi: https://doi.org/10.25559/SITITO.019.202302.292-297

Original article

Comparative Characteristics of Containerization Platforms Docker and Singularity

A. S. Bondyakov*, A. O. Kondratyev
Joint Institute for Nuclear Research, Dubna, Russian Federation
Address: 6 Joliot-Curie St., Dubna 141980, Moscow region, Russian Federation
* aleksey@jinr.ru

Abstract

To date, to solve problems of varying complexity, high power and increased requirements for equipment are required. These are engineering design tasks, a wide range of scientific computing tasks, modeling systems, data analysis, etc. As a rule, any industry has tasks that require high-performance computing. To solve such problems, high-performance computing systems are used, which are based on parallel computing technologies, virtualization and containerization technologies.

Parallel computing can significantly increase the performance of computing systems when executing the same program. Virtualization technologies allow you to create several virtual machines on one physical computer, each of which runs a separate operating system. Containers are one of the most promising areas for the development of modern information technologies. Containers are currently used in many areas, including: web applications (such as Apache Tomcat), mobile applications, database management systems, and for deploying server environments. According to the developers, containers will simplify the development and deployment of high-performance computing systems, significantly improve their performance and reliability.

This article presents the comparative characteristics of the containerization platforms Docker and Singularity, in the aspect of solving such problems as code portability between the local runtime and the production environment of the application. Examples of using these platforms to solve problems that are currently relevant are given. The most important aspects of the work of these platforms in relation to high-performance computing clusters are considered.

Keywords: container platform, virtualization, high-performance clusters, software packages

Conflict of interests: The authors declare no conflict of interests.

For citation: Bondyakov A. S., Kondratyev A. O. Comparative characteristics of containerization platforms Docker and Singularity. *Modern Information Technologies and IT-Education*. 2023;19(2):292-297. doi: https://doi.org/10.25559/SITITO.019.202302.292-297

Введение

В современной науке высокопроизводительные кластеры решают сложнейшие задачи моделирования, анализа больших данных, оптимизации различных процессов и многое другое. Позволяют выполнять расчеты во многих областях науки и техники. Для выполнения таких расчетов используется специфическое программное обеспечение, такое как ABINIT, VASP, Wien2K и т. д. Некоторые кластеры могут сразу предоставить пользователю доступ к подобному ПО в качестве дополнительного сервиса. Это, как правило, ПО, распространяемое на условиях свободного лицензионного договора. Все высокопроизводительные кластеры уникальны с точки зрения своей архитектуры, настроек ПО, используемой операционной системы и прочее. Довольно часто, решая задачу на кластере, пользователь сталкивается с такими проблемами, как другая версия требуемого ПО, другая ОС, другой набор компиляторов и т. д. Одним из решений данных проблем является использование контейнерных платформ. На сегодняшний день самыми популярными и востребованными контейнерными платформами считаются Docker и Singularity.Контейнерные платформы предоставляют возможность создавать и запускать контейнеры, в которых можно упаковывать требуемые программные пакеты. Посредством Docker или Singularity, можно создать контейнер с нужным программным пакетом для его последующего запуска на вычислительном кластере. Контейнер представляет собой один файл и легко портируется между разными операционными системами. Контейнеризация во многом напоминает виртуализацию системы — создание виртуальных машин, изолированных друг от друга и от физической машины, на которой они работают.Главное отличие контейнеризации от виртуализации заключается в том, что контейнер содержит в себе только операционную систему, в то время как виртуальная машина включает в себя виртуализацию всех элементов физической машины. Контейнеры для своей работы используют физические ядра, в то время как виртуальные машины — виртуальные. Таким образом, контейнерные платформы привязаны к аппаратной архитектуре физической машины и ее ОС. Одним из основных преимуществ контейнера, по сравнению с виртуальной машиной, является минимизация времени при создании новых контейнеров на работающих узлах, что особенно важно при масштабировании кластеров.

Основная задача Docker и Singularity — простой запуск и переносимость контейнеров, содержащих приложения различной сложности, из одной операционной системы в другую с последующим их выполнением на высокопроизводительных кластерах. Docker и Singularity — как и большинство других платформ контейнеризации, основанные на ядре Linux проекты с открытым исходным кодом, активно используются как в научной среде, так и в промышленности.В настоящее время Docker и Singularity являются основными инструментами контейнеризации. Популярность, например, Docker можно объяснить востребованностью контейнерной инфраструктуры, в связи с чем данная платформа активно развивается.

Цель исследования

Проанализировать различные возможности контейнерных платформ Docker и Singularity и выбрать наиболее подходящую для выполнения вычислений.

Основная часть

Анализируя возможности контейнерных платформ Docker и Singularity, следует отметить несколько наиболее важных особенностей данных платформ, позволяющих выбрать оптимальный контейнер для выполнения высокопроизводительных вычислений.

Операционную систему Linux можно условно разделить на две основные части: область ядра и область пользователя. Ядро осуществляет взаимодействие с аппаратными средствами и обеспечивает выполнение основных системных функций. Область пользователя — это среда, в которой работают приложения, библиотеки и системные службы. Как правило, мы используем операционную систему с фиксированной комбинацией ядра и областью пользователя. При использовании, например, Oracle Linux OS, довольно сложно установить программное обеспечение, созданное для Ubuntu, потому что область пользователей этих дистрибутивов несовместима. Также очень сложно установить несколько версий одного и того же программного обеспечения [1-7].

Контейнеры Singularity делают область пользователя динамически изменяемой. Это означает, что вся область пользователя операционной системы Linux, включая программы, пользовательские конфигурации и среду, может быть независимой от ОС. Контейнер Singularity упаковывает все, что требуется, в один проверяемый файл. Благодаря этому можно создавать или разворачивать приложения на базе любой наиболее оптимальной для них операционной системы.

В работе контейнера Singularity важно выделить следующие особенности:

- Однофайловый формат контейнера SIF мобилен, его легко перемещать и совместно использовать.
- Контейнер защищен криптографическими подписями.
- По умолчанию используется интеграция вместо изоляции, т. е. приложениям контейнера доступны графические процессоры, сети, параллельные файловые системы.
- Простая, но в то же время эффективная модель безопасности. Пользователь является тем же пользователем внутри контейнера, что и снаружи, и по умолчанию не имеет прав администратора в основной (хост) системе.

Docker-контейнер так же, как и Singularity, можно запускать на локальных машинах, виртуальных машинах или разворачивать в облаке. Docker, в отличие от Singularity, кроссплатформенный (кроме Linux, поддерживает работу с Windows и MacOS). Docker использует изолированную файловую систему, которая предоставляется образом контейнера. Так как образ содержит файловую систему контейнера, он также содержит все зависимости — конфигурации, сценарии, двоичные файлы и т. д., необходимые для запуска приложения. Кроме того, образ также содержит такую конфигурацию, как переменные среды, команду по умолчанию для запуска и другие метаданные.В конфигурационных файлах образа Dockerfiles, как правило, вначале указывается используемый дистрибутив Linux. Подобные образы публикуются в интернете такими организациями, как Canonical, Debian, Oracle и многими другими посредством Docker Hub [8-12].

Далее в процессе формирования контейнера можно загрузить, настроить и скомпилировать дополнительные источники. Docker, так же как и Singularity, предоставляют сервис, посредством которого можно осуществлять хранение, а также выгружать образы контейнеров. Для создания нескольких контейнеров может быть использован один и тот же образ.Рассматривая архитектуру Docker, нужно отметить использование cgroups, важной функции ядра Linux, посредством которой осуществляется управление ресурсами и «пространством имен» для изоляции процессов, запущенных внутри контейнеров. Основой Docker на уровне операционной системы является служебный сервис dockerd, работающий с правами суперпользователя. Эта особенность ограничивает применение контейнеризации Docker на платформах высокопроизводительных кластеров. Основная задача данного сервиса — осуществлять управление контейнерами. Доступ к данному сервису осуществляется через REST API клиентом Docker, посредством командной строки терминала Linux.Singularity, в отличие от Docker, целенаправленно разрабатывалась для инфраструктуры высокопроизводительных кластеров с основной задачей — портирование пакетов, программных кодов, приложений между различными платформами. Архитектура Singularity предоставляет возможность пользователям безопасно запускать свои контейнеры на платформах высокопроизводительных кластеров, не требуя прав суперпользователя, а также использует программные решения, которые не требуют наличия служебного сервиса.Для выполнения образы Singularity используют права пользователя посредством системной организации доступа UNIX SUID. Изоляция процессов достигается благодаря функции «пространства имен» ядра Linux. Singularity, в отличие от Docker, не управляет системными ресурсами и не пользуется cgroups. Singularity поддерживает формат образов Docker, тем не менее для Singularity создан, по образу и подобию, персональный реестр Docker Hub [13-20].

В Singularity реализованы специальные механизмы, позволяющие запускать приложения без права доступа к привилегиям root, что позволяет безопасно использовать данную платформу в различных инфраструктурах для высокопроизводительных вычислений [21-22]. В таблице 1 представлен обзор [23-25] сравнения некоторых важных критериев контейнерных платформ Docker и Singularity.

Таблица 1. **Сравнение контейнеров**

Table 1. **Container comparison**

	Docker	Singularity
Нет привилегированных или доверенных сервисов	есть	нет
Требуются дополнительные настройки сети	да	нет
Доступ к файловой системе хостовой машины	есть	есть
Встроенная поддержка графического процессора (GPU)	нет	есть
Встроенная поддержка InfiniBand	есть	есть
Встроенная поддержка MPI	есть	есть
Работает со всеми планировщиками	нет	да
Среда контейнера имеет правильные разрешения	да	да
Контейнеры портируемые, не модифицируются при использовании	нет	да
Администраторы могут контролировать и ограничивать возможности	нет	да

Источник: здесь и далее в статье все таблицы и рисунок составлены авторами.
Source: Hereinafter in this article all tables and figure were made by the authors.

Для сравнения производительности описываемых контейнерных платформ в облачной инфраструктуре ОИЯИ были созданы виртуальные машины со следующими параметрами: 4 CPU, 16 RAM, 20 GB. На каждой из машин были установлены контейнеры Docker и Singularity. В качестве образа ОС для данных контейнеров использовался образ Oracle Linux 8x, установленный с Docker Hub. В контейнерах были развернуты пакеты ABINIT, VASP, Wien2K, выполняющие роль бенчмарков. Результаты выполнения контрольных задач посредством указанных пакетов представлены на рис. 1. В моменты пиковой загрузки CPU проводилось измерение мгновенного значения Load average (таблица 2).

Таблица 2. **Мгновенные значения Load average (среднее значение загрузки системы за 1, 5 и 15 минут) в момент пиковой загрузки CPU**

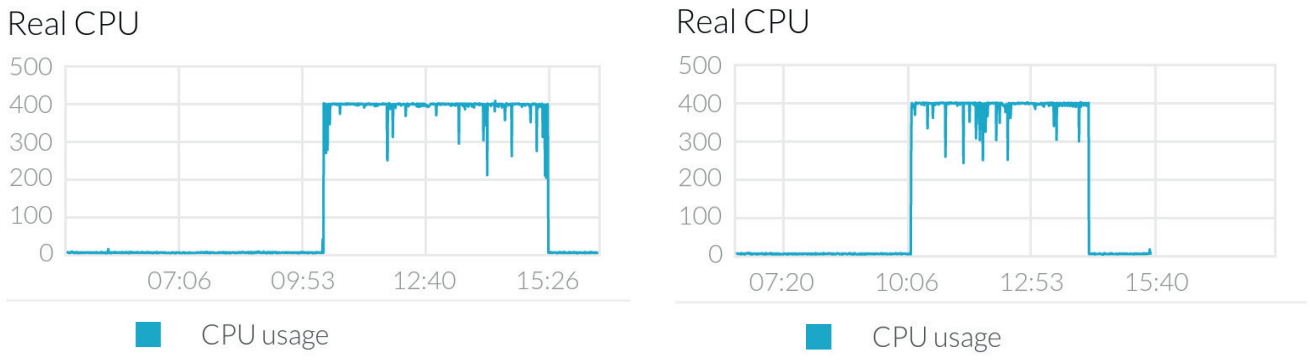
Table 2. **Instantaneous Load Average Values (average system load for 1, 5 and 15 minutes) at the time of peak CPU load**

Docker	Singularity
5.41 2.79 1.16 5/157 27158	5.35 2.95 1.25 6/157 27175

Анализируя полученные данные производительности Docker и Singularity на рис. 1 и в таблице 1, мы видим практически одинаковую загрузку CPU при выполнении тестовых расчетов. Таким образом, с точки зрения производительности Docker и Singularity практически идентичны. Но так как архитектура Singularity предоставляет возможность стандартным пользователям безопасно запускать свои контейнеры, не требуя прав суперпользователя, а также использует программные решения, которые не требуют наличия служебного сервиса, то использование Singularity представляется наиболее оптимальным решением.

Полученные результаты

На виртуальных машинах облачной инфраструктуры ОИЯИ сформированы две независимые друг от друга системы контейнеризации для вычислений, позволяющие легко запускать и портировать вычислительные контейнеры. Приведены сравнительные характеристики производительности контейнерных платформ. Проанализированы возможные сценарии использования контейнерной технологии для проведения расчетов. Рассмотрены преимущества и недостатки Docker и Singularity. В частности, уделено внимание вопросу безопасности контейнерных платформ.



а) б)

Р и с. 1.
Мониторинг производительности контейнерных платформ средствами облачной инфраструктуры ОИЯИ: а) Docker, б) Singularity
F i g. 1. Monitoring the performance of container platforms using the JINR cloud infrastructure: a) Docker, b) Singularity

Заключение

Благодаря контейнеризации можно эффективно использовать вычислительные ресурсы, что позволяет снизить общую стоимость владения вычислительной инфраструктурой. Контейнеры позволяют упростить перенос, установку и обслуживание программ и тем самым обеспечить высокий уровень безопасности и надежности. Кроме того, контейнеры позволяют обеспечить гибкость и масштабируемость, которые необходимы для работы сложных программных систем.

References

[1] Vogel M., Borodin M., Forti A., Heinrich L. Standalone containers with ATLAS offline software. *EPJ Web of Conferences*. 2020;245:07010. https://doi.org/10.1051/epjconf/202024507010
[2] Benjamin D., Childers T., Lesny D., Oleynik D., Panitkin S., Tsulaia V., Yang W., Zhao X. Building and using containers at HPC centres for the ATLAS experiment. *EPJ Web of Conferences*. 2019;214:07005. https://doi.org/10.1051/epjconf/201921407005
[3] Ozturk N., Undrus A., Vogel M., Forti A. Containerization in ATLAS Software Development and Data Production. *EPJ Web of Conferences*. 2021;251:02017. https://doi.org/10.1051/epjconf/202125102017
[4] Bocchi E., Blomer J., Mosciatti S., Valenzuela A. CernVM-FS powered container hub. *EPJ Web of Conferences*. 2021;251:02033. https://doi.org/10.1051/epjconf/202125102033
[5] Valassi A., Alef M., Barbet J.-M., Datskova O., De Maria R., Medeiros M.F., Giordano D., Grigoras C., Hollowell C., Javurkova M., Khristenko V., Lange D., Michelotto M., Rinaldi L., Sciabà A., van der Laan C. Using HEP experiment workflows for the benchmarking and accounting of WLCG computing resources. *EPJ Web of Conferences*. 2020;245:07035. https://doi.org/10.1051/epjconf/202024507035
[6] Godlove D. Singularity: Simple, secure containers for compute-driven workloads. In: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning) (PEARC '19). New York, NY, USA: Association for Computing Machinery; 2019. Article number: 24. p. 1-4. https://doi.org/10.1145/3332186.3332192
[7] Shahidullah K., Md Sadun H., Ali T., Turgay K. Container Technologies For ARM Architecture: A Comprehensive Survey Of The State-of-the-art. *IEEE Access*. 2022;10:84853-84881. http://dx.doi.org/10.1109/ACCESS.2022.3197151
[8] Bhardwaj A., Krishna C.R. Virtualization in Cloud Computing: Moving from Hypervisor to Containerization – A Survey. *Arabian Journal for Science and Engineering*. 2021;46(9):8585-8601. https://doi.org/10.1007/s13369-021-05553-3
[9] Liu P., Ji S., Fu L., Lu K., Zhang X., Lee W.-H., Lu T., Chen W., Beyah R. Understanding the Security Risks of Docker Hub. In: Chen L., Li N., Liang K., Schneider S. (eds.) Computer Security – ESORICS 2020. ESORICS 2020. *Lecture Notes in Computer Science*. Vol. 12308. Cham: Springer; 2020. p. 257-276. https://doi.org/10.1007/978-3-030-58951-6_13
[10] Martin A., Raponi S., Combe T., Pietro R. Docker ecosystem – Vulnerability Analysis. *Computer Communications*. 2018;122:30-43. https://doi.org/10.1016/j.comcom.2018.03.011Zhu H., Gehrmann C. Lic-Sec: An enhanced AppArmor Docker security profile generator. *Journal of Information Security and Applications.* 2021;61:102924. https://doi.org/10.1016/j.jisa.2021.102924
[11] De Benedictis M., Lioy A. Integrity verification of Docker containers for a lightweight cloud environment. *Future Generation Computer Systems.* 2019;97:236-246. https://doi.org/10.1016/j.future.2019.02.026
[12] Baresi L., Quattrocchi G., Rasi N. A Qualitative and Quantitative Analysis of Container Engines. *Journal of Systems and Software*. 2024;210:111965. https://doi.org/10.1016/j.jss.2024.111965

[13] Mitra-Behura S., Fiolka R.P., Daetwyler S. Singularity Containers Improve Reproducibility and Ease of Use in Computational Image Analysis Workflows. *Frontiers in Bioinformatics*. 2022;1:757291. https://doi.org/10.3389/fbinf.2021.757291

[14] Garofoli A., Paradiso V., Montazeri H., Jermann P., Roma G., Tornillo L., Terracciano L., Piscuoglio S., Ng C. PipeIT: Singularity Container for Molecular Diagnostic Somatic Variant Calling on Ion Torrent NGS Platform. *The Journal of Molecular Diagnostics.* 2019;21(5):884-894. https://doi.org/10.1016/j.jmoldx.2019.05.001

[15] Zhou N., Georgiou Y., Pospieszny M., Zhong L., Zhou H., Niethammer C., Pejak B., Marko O., Hoppe D. Container orchestration on HPC systems through Kubernetes. *Journal of Cloud Computing: Advances, Systems and Applications.* 2021;10(1). https://doi.org/10.1186/s13677-021-00231-z

[16] Kurtzer G.M., Sochat V., Bauer M.W. Singularity: Scientific containers for mobility of compute. *PLoS ONE*. 2017;12(5):e0177459. https://doi.org/10.1371/journal.pone.0177459

[17] Lee H., Kwon S., Lee J.-H. Experimental Analysis of Security Attacks for Docker Container Communications. *Electronics*. 2023;12(4):940. https://doi.org/10.3390/electronics12040940

[18] Alyas T., Ali S., Khan H., Samad A., Alissa K., Saleem M.A. Container Performance and Vulnerability Management for Container Security Using Docker Engine. *Security and Communication Networks*. 2022;2022. https://doi.org/10.1155/2022/6819002

[19] Reis D., Piedade B., Correia F., Dias J., Aguiar A. Developing Docker and Docker-Compose Specifications: A Developers' Survey. *IEEE Access*. 2022;10:2318-2329. https://doi.org/10.1109/ACCESS.2021.3137671

[20] Maruszczak A., Walkowski M., Sujecki S. Base Systems for Docker Containers – Security Analysis. In: 2022 International Conference on Software, Telecommunications and Computer Networks (SoftCOM). Split, Croatia: IEEE Computer Society; 2022. p. 1-5. https://doi.org/10.23919/SoftCOM55329.2022.9911523

[21] Combe T., Martin A., Pietro R. To Docker or Not to Docker: A Security Perspective. *IEEE Cloud Computing*. 2016;3(5):54-62. https://doi.org/10.1109/MCC.2016.100

[22] Priedhorsky R., Randles T. Charliecloud: unprivileged containers for user-defined software stacks in HPC. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '17). New York, NY, USA: Association for Computing Machinery; 2017. Article number: 36. https://doi.org/10.1145/3126908.3126925

[23] Haji L.M., Zeebaree S.R.M., Ahmed O.M., Sallow A.B., Jacksi K., Zebari R.R. Dynamic Resource Allocation for Distributed Systems and Cloud Computing. *Test Engineering and Management*. 2020;83:22417-22426. Available at: http://www.testmagzine.biz/index.php/testmagzine/article/view/11297 (accessed 04.05.2023).

[24] Hu Y., Zhou H., de Laat C., Zhao Z. Concurrent container scheduling on heterogeneous clusters with multi-resource constraints. *Future Generation Computer Systems*. 2020;102;562-573. https://doi.org/10.1016/j.future.2019.08.025

Поступила 04.05.2023; одобрена после рецензирования 14.06.2023; принята к публикации 23.06.2023.
Submitted 04.05.2023; approved after reviewing 14.06.2023; accepted for publication 23.06.2023.

Об авторах:

Бондяков Алексей Сергеевич, инженер-программист Лаборатории информационных технологий имени М.Г. Мещерякова, Международная межправительственная организация Объединенный институт ядерных исследований (141980, Российская Федерация, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6), кандидат технических наук, **ORCID: https://orcid.org/0000-0003-0429-3931,** aleksey@jinr.ru

Кондратьев Андрей Олегович, инженер-программист Лаборатории информационных технологий имени М.Г. Мещерякова, Международная межправительственная организация Объединенный институт ядерных исследований (141980, Российская Федерация, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6), **ORCID: https://orcid.org/0000-0001-6203-9160,** kondratyev@jinr.ru

Все авторы прочитали и одобрили окончательный вариант рукописи.

About the authors:

Aleksey S. Bondyakov, Software Engineer of the Mescheryakov Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russian Federation), Cand. Sci (Eng.), **ORCID: https://orcid.org/0000-0003-0429-3931,** aleksey@jinr.ru

Andrey O. Kondratyev, Software Engineer of the Mescheryakov Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russian Federation), **ORCID: https://orcid.org/0000-0001-6203-9160,** kondratyev@jinr.ru

All authors have read and approved the final manuscript.