

# Разработка системы мониторинга нагрузки slurm задач на вычислительные ресурсы гетерогенной платформы HybriLIT

Г. А. Карпов<sup>1,\*</sup>, М. А. Сказкин<sup>1</sup>, Д. В. Беляков<sup>2</sup> and М. И. Зуев<sup>2</sup>

<sup>1</sup>Департамент Математического и компьютерного моделирования, Институт Математики и Компьютерных Технологий, Дальневосточный Федеральный Университет, Владивосток, Россия, 690922

<sup>2</sup>Объединённый институт ядерных исследований, ул. Жолио-Кюри 6, Дубна, 141980, Российская Федерация

## Аннотация

Гетерогенная платформа «HybriLIT» используется в ОИЯИ для решения различных вычислительных задач, связанных с обработкой данных научных экспериментов. Основным компонентом платформы является суперкомпьютер «Говорун», состоящий из более чем ста серверов, постоянно выполняющих программы пользователей, и сообщающихся между собой по высокоскоростному интерконнекту. Поддержание работоспособности платформы является одной из задач системного администрирования. Для контроля за состоянием платформы применяется система мониторинга, которая позволяет в режиме реального времени следить за состоянием вычислительных узлов, включая загруженность центрального и графических процессоров, использование оперативной памяти и систем хранения данных, объём сетевого трафика и т.п. Однако существующие решения (системы мониторинга) не позволяют связать выполнение задач пользователей с оказываемыми ими нагрузками на компоненты вычислительных узлов, а только предоставляют интегральные значения этих нагрузок. В работе представлена разработанная система мониторинга, которая позволяет отслеживать нагрузку, создаваемую задачами пользователей, предоставляя информацию, как по интегральным значениям, так и детальную информацию по использованию ресурсов вычислительного узла задачами пользователя. При разработке решения были учтены преимущества и недостатки ранее используемых систем. Результатом работы стал программный продукт, состоящий из серверной, клиентской части и ряда сенсоров, позволяющий в режиме реального времени следить за состоянием платформы, оценивать текущую нагрузку на компоненты вычислительных узлов, проводить анализ данных за краткосрочный период (до 24 часов) по создаваемой нагрузке задачами пользователей через удобный для администратора графический интерфейс.

## Ключевые слова

мониторинг, пользовательские задачи, режим реального времени, гетерогенная платформа, Slurm

## 1. Введение

Гетерогенная платформа HybriLIT [1] состоит из суперкомпьютера «Говорун» и учебно-тестового полигона. Платформа активно используется пользователями для выполнения сложных научных вычислений, которые относятся к задачам ядерной физики и различным научным исследованиям. Основным компонентом платформы является суперкомпьютер «Говорун», который был построен на более чем ста серверах. Эти серверы подключены к высокоскоростному интерконнекту, который работает со скоростью 100 Гбит/с. Пользователи постоянно запускают вычислительные задачи, которые выполняются в «SLURM resource manager» [2].

*Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems 2025 (ITTMM 2025), Moscow, April 07–11, 2025*

\* Автор, отвечающий за публикацию.

✉ asgardkarpov@yandex.ru (Г. А. Карпов); makcymal@yandex.ru (М. А. Сказкин); dmitry@jinr.ru (Д. В. Беляков); zuevmax@jinr.ru (М. И. Зуев)

🆔 0000-0001-6516-1479 (Д. В. Беляков)



© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Работа платформы требует постоянного контроля со стороны службы поддержки. Существует два метода контроля: мониторинг аппаратных компонентов и анализ статистики использования ресурсов. Первый используется для контроля состояния аппаратных компонентов всех вычислительных узлов в режиме реального времени. Он включает в себя нагрузку на центральный и графический процессоры, использование оперативной памяти и локальной системы хранения данных, сетевой трафик и температуру физических компонентов серверов. Второй метод используется для контроля состояния всей вычислительной системы на основе анализа комплексных параметров, накопленных за определенный период времени работы. Данные параметры связывают состояния физических частей вычислительных узлов с активностью пользователя в процессе выполнения заданий. Такой подход можно рассматривать как добавление высокого уровня детализации в систему мониторинга и анализа статистики использования ресурсов. Что должно давать более подробные ответы на более сложные вопросы о функционировании, производительности и эффективности использования ресурсов гетерогенной платформы HybriLIT. Например, какая группа пользователей создала наибольшую нагрузку на сетевую систему хранения данных в прошлом месяце и каков личный вклад каждого пользователя в эту нагрузку? Или как распределение оперативной памяти, выделенной пользователем для выполнения задания SLURM, соотносилось с фактическим потреблением оперативной памяти, распределенным по времени выполнения задания? Таким образом, существует потребность в новых системах, которые могли бы обеспечить такие функциональные возможности.

## 2. Архитектура системы мониторинга вычислительных ресурсов кластера

Система состоит из трех основных компонентов: сенсоров, клиентов и сервера. Сенсоры разделены на группы в соответствии с их техническими характеристиками и назначением, и каждый сенсор имеет свое собственное название. Все компоненты взаимодействуют друг с другом посредством запросов и независимы друг от друга. Общую архитектуру системы можно увидеть на рис. 1.

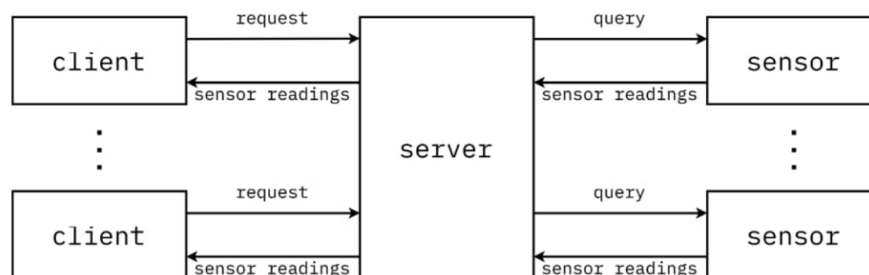


Рис. 1: Общая архитектура системы.

Сенсор – это программа, запущенная на вычислительном узле, которая постоянно собирает данные о состоянии и загрузке компонентов через определенные промежутки времени. Список необходимой информации определяется запросами с сервера. Поточковая передача данных между сервером и сенсорами осуществляется через TCP-сокеты внутри локальной сети, по которым данные передаются в формате JSON в соответствии с разработанным протоколом связи. Пример протокола показан на рис. 2.

Протокол для обмена данными между сервером и клиентом имеет структуру, аналогичную протоколу сенсор-сервер, и осуществляется через WebSockets [3]. Сервер возвращает JSON с ключом заголовка в структуре, аналогичной запросу от клиента. Остальные поля соответствуют запросу от клиента. Структура протокола показана на рис. 3.

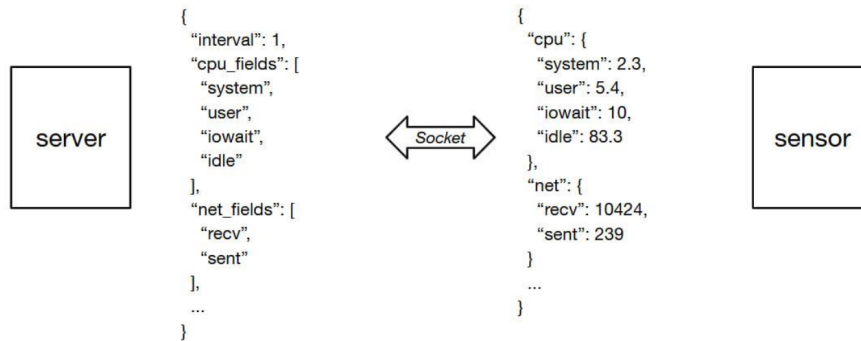


Рис. 2: Пример протокола сенсор-сервер.

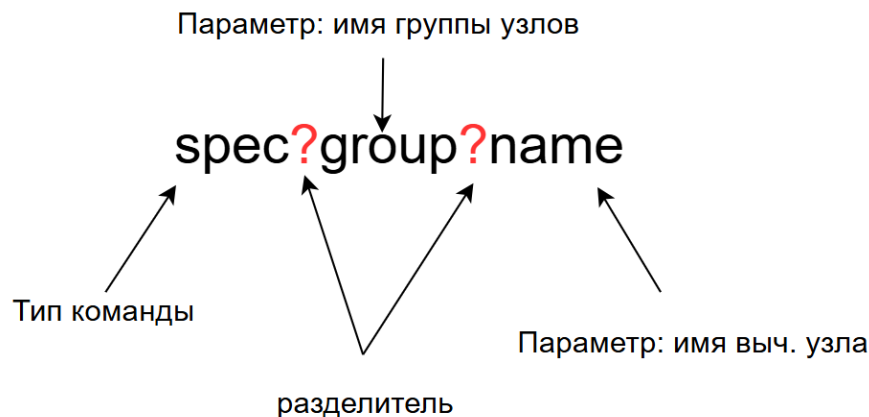


Рис. 3: Пример протокола сенсор-сервер.

Чтобы иметь возможность настраивать различные уровни доступа в рамках системы, авторизация пользователя была реализована с помощью сервиса FreeIPA [4, 5]. Авторизация осуществляется с помощью токена JWT [6], который генерируется при входе в систему с использованием учетной записи, зарегистрированной в сервисе FreeIPA, и содержит информацию о текущем пользователе.

Новая версия системы включает в себя мониторинг использования ресурсов по отдельным Slurm задачам и просмотр истории данных мониторинга за последние 24 часа.

При запуске задачи на вычислительном узле SLURM использует механизм `setuid` для установки владельца процессов - пользователя, запустившего данную задачу. Сенсоры на вычислительных узлах используют стандартные Linux-утилиты (например, `ps`, `grep`, `pgrep`, `top` и т.п.) для того, чтобы извлечь все процессы, принадлежащие владельцу задачи, а по уже по ним – использование CPU и памяти.

Реализация суточного мониторинга происходит в серверной части системы путем сохранения данных сенсоров в базу MongoDB [7]. Сервер добавляет новые данные сенсоров в хранилище, удаляя при этом данные, которые старше 24 часов. Такой функционал позволяет отслеживать детализированные изменения показаний сенсоров во времени за последние сутки.

### 3. Архитектура системы мониторинга вычислительных ресурсов кластера

В настоящее время разработаны следующие страницы и компоненты, обеспечивающие функциональность веб-приложения: страница авторизации, страница профиля пользователя, главная страница, страница мониторинга узлов группы, страница расширенного мониторинга узлов. Сенсоры и сервер были реализованы с использованием языка программирования Python 3 [8] и веб-фреймворка FastAPI [9]. Клиентская часть была реализована с использованием JavaScript [10], CSS [11], HTML [12] языков и фреймворка Vue 3 [13]. Диаграммы на странице мониторинга узлов группы показывают нагрузку на различные компоненты (процессор, сетевой адаптер, оперативная память, система хранения) для выбранной группы вычислительных узлов. Отображаемые данные представляет собой среднее значение по нагрузке на каждый компонент узлов группы. Диаграммы представлены в виде круговых диаграмм и столбчатых диаграмм. Компонент круговых диаграмм показаны на рис. 4.

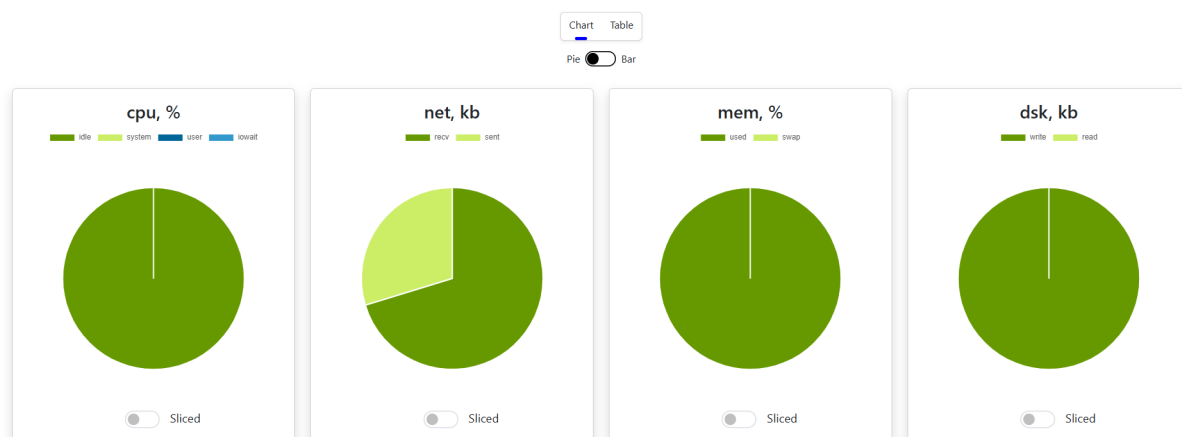


Рис. 4: Представление данных в виде круговых диаграмм.

Представление данных в виде таблицы показывает нагрузку на компоненты каждого вычислительного узла выбранной группы (рис. 5). Там, где это уместно, также отображается средняя и общая нагрузка по различным параметрам компонентов вычислительного узла. Особенностью данного представления и отличием от дизайна предыдущей версии системы является то, что таблица с данными всегда находится на экране пользователя с отображением названий параметров и средней и общей нагрузки. В случае большого количества вычислительных узлов появляется возможность прокручивать строки таблицы. Также имеется цветовая индикация большого значения нагрузки любого отображаемого параметра.

Страница расширенного мониторинга содержит детальную информацию о состоянии вычислительного узла, представленную в виде отдельных таблиц для каждого компонента (рис. 6). Также имеется представление, содержащее спецификацию вычислительного узла.

Суточный мониторинг представлен в виде графика (рис. 7), на котором можно изменять масштаб и перемещать видимую область в рамках последних 24 часов.

### 4. Заключение

Разработанная система обеспечивает решение задачи мониторинга вычислительных ресурсов суперкомпьютера Govogun как в реальном времени, так и за краткосрочный период времени (до 24 часов). В настоящее время данная система успешно развёрнута на ресурсах

ChartTable

body part of a table is scrollable

INFO		CPU					NET		MEM		DSK	
id	name	system	user	iowait	idle	freq	recv	sent	used	swap	read	write
1	rtma105.jinr.ru57	0.1 %	0.1 %	0 %	99.6 %	2000 hz	1130 kb	762 kb	17.6 %	0 %	0 kb	0 kb
2	rtma105.jinr.ru444	0.1 %	0.1 %	0 %	99.6 %	2000 hz	1176 kb	761 kb	17.6 %	0 %	0 kb	0 kb
3	rtma105.jinr.ru27	0 %	0 %	0 %	99.9 %	2000 hz	1176 kb	760 kb	17.6 %	0 %	0 kb	0 kb
4	rtma105.jinr.ru417	0 %	0 %	0 %	99.9 %	2000 hz	1176 kb	760 kb	17.6 %	0 %	0 kb	0 kb
5	rtma104.jinr.ru96	0.1 %	0 %	0.2 %	99.6 %	2000 hz	2730 kb	990 kb	18.6 %	0.2 %	0 kb	16 kb
AVERAGE		0 %	0 %	0 %	100 %	-	1478 kb	807 kb	18 %	0 %	0 kb	3 kb
TOTAL		-	-	-	-	-	7388 kb	4033 kb	-	-	0 kb	16 kb

Рис. 5: Представление данных в виде таблицы.

cpu net mem dsk

body part of a table is scrollable

INFO		CPU										
id	name	system	user	nice	iowait	idle	irq	softirq	steal	guest	guest_nice	freq
1	thread1	0 %	0 %	0 %	0 %	100 %	0 %	0 %	0 %	0 %	0 %	2000 hz
2	thread2	0 %	0 %	0 %	0 %	100 %	0 %	0 %	0 %	0 %	0 %	2000 hz
3	thread3	0 %	0 %	0 %	0 %	100 %	0 %	0 %	0 %	0 %	0 %	2000 hz
4	thread4	0 %	0 %	0 %	0 %	100 %	0 %	0 %	0 %	0 %	0 %	2000 hz
AVERAGE		0 %	0 %	0 %	0 %	100 %	0 %	0 %	0 %	0 %	0 %	-
TOTAL		-	-	-	-	-	-	-	-	-	-	-

Рис. 6: Таблица расширенного мониторинга

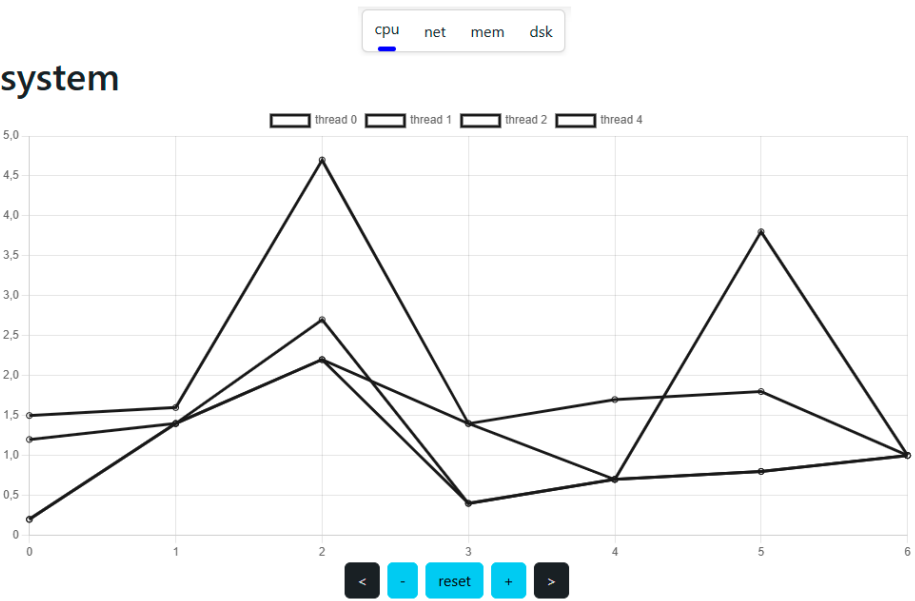


Рис. 7: Представление данных суточного мониторинга в виде графика.

гетерогенной платформы HybriLIT. Разработанный дизайн сохранил концепцию предыдущей версии системы, но получил важные улучшения в реализации функций мониторинга. Данная система построена на основе современных IT-технологий и имеет возможность расширения функционала в будущем.

## Список литературы

1. Anikina, A. *и др.* Structure and Features of the Software and Information Environment of the HybriLIT Heterogeneous Platform, 444—457. doi:10.1007/978-3-031-80853-1\_33 (2025).
2. *SLURM* <https://slurm.schedmd.com/> (2025).
3. *WebSockets docs*. [https://developer.mozilla.org/ru/docs/Web/API/WebSockets\\_API](https://developer.mozilla.org/ru/docs/Web/API/WebSockets_API) (2025).
4. *FreeIPA* <https://www.freeipa.org/> (2025).
5. *Python FreeIPA client docs*. <https://python-freeipa.readthedocs.io/en/latest/> (2025).
6. *JWT-token* <https://jwt.io/> (2025).
7. *MongoDB* <https://www.mongodb.com/> (2025).
8. *Python 3* <https://docs.python.org/3/index.html> (2025).
9. *FastAPI* <https://fastapi.tiangolo.com/> (2025).
10. *JavaScript docs* <https://developer.mozilla.org/en%20US/docs/Web/JavaScript> (2025).
11. *CSS docs* <https://developer.mozilla.org/en%20US/docs/Web/CSS> (2025).
12. *HTML docs* <https://developer.mozilla.org/en%20US/docs/Web/HTML> (2025).
13. *Vue.js 3* <https://vuejs.org/> (2025).