МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Электронных и вычислительных машин»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«__»_____ 2023 г.

Разработка внешних скриптов для системы мониторинга Zabbix

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУРГУ-090301.2023.225 ПЗ ВКР

Консультант
к.т.н., доцент каф. СП
_____Н.Ю.Долганина
«___»_____ 2023 г.

Руководитель работы,
к.т.н., доцент каф. ЭВМ
_____ В.А.Парасич
«___»_____ 2023 г.

Автор работы,
студент группы КЭ-406
_____ Д.К.Швалёв
«___»_____2023 г.

Нормоконтролёр,
ст. преп. каф. ЭВМ
_____ С.В. Сяськов
«___»_____2023 г.

Челябинск-2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Электронных и вычислительных машин»

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«___»_____2023 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-406
Швалёву Данилу Кирилловичу,
обучающемуся по направлению
09.03.01 «Информатика и вычислительная техника»

1. **Тема работы.** «Разработка внешних скриптов для системы мониторинга Zabbix» утверждена приказом по университету от 25.04.2023г. №753-13/12
2. **Срок сдачи студентом законченной работы:** 01.06.2023 г.
3. **Исходные данные к работе:**
   - Документация к Zabbix серверу;
   - Документация к скриптам на языке Bash;
   - Сервер с виртуализацией Hyper-V;
   - Инфраструктура Суперкомпьютерного Центра ЮУрГУ.
4. **Перечень подлежащих разработке вопросов**

4.1. Выполнить анализ предметной области.

4.2. Выполнение описания системы Zabbix.

4.3. Выполнить проектирование.

4.4. Реализовать систему Zabbix.

4.5. Произвести тестирование развернутой системы.

5. **Дата выдачи задания:** 01.12.2022 г.

Научный руководитель,
доцент кафедры ЭВМ, к.т.н. _____ В.А.Парасич

Задание принял к исполнению_____ Д.К.Швалёв

КАЛЕНДАРНЫЙ ПЛАН

Этап	Срок сдачи	Подпись руководителя
Анализ предметной области	01.03.2023	
Описание системы Zabbix	01.04.2023	
Проектирование	01.05.2023	
Реализация	15.05.2023	
Тестирование	24.05.2023	
Компоновка текста работы и сдача на нормоконтроль	30.05.2023	
Подготовка презентации и доклада	30.05.2023	

Научный руководитель,
доцент кафедры ЭВМ, к.т.н. _____ В.А.Парасич

Задание принял к исполнению_____ Д.К.Швалёв

АННОТАЦИЯ

Д.К. Швалёв Разработка внешних скриптов для системы мониторинга Zabbix. – Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ВШ ЭКН; 2023, 47 с., 28 ил., библиогр. список – 11 наим.

В данной выпускной квалификационной работе рассматривается вопрос разработки внешних скриптов для системы мониторинга и реализуется решение веб-интерфейса. Процесс выполнения выпускной квалификационной работы включает в себя решение следующих вопросов: обзор аналогов, проектирование, развертывание системы, а также тестирование программного обеспечения.

В ходе обзора аналогов реализуемого веб-интерфейса выявлены и описаны основные направления в развертывании системы мониторинга. При рассмотрении программ-аналогов, связанных с мониторингом инфраструктуры, отмечались их достоинства и недостатки. Сравнивая системы мониторинга между собой, были выявлены требования к функционалу разворачиваемого веб-интерфейса.

Проведя обзор средств реализации, был составлен перечень программных продуктов и разработок, наиболее подходящих для выполнения задачи выпускной квалификационной работы.

Описывая процесс реализации системы, был разобран основной функционал разрабатываемого веб-интерфейса, рассмотрен принцип работы.

При тестировании программного обеспечения были использованы современные методики, позволяющие определить неисправности на раннем этапе разработки.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность проблемы

Проблема мониторинга инфраструктуры достаточно актуальна в современном мире, так как она затрагивает аспекты обеспечения бесперебойного функционирования различных технологических систем, на которые общество полагается в повседневной деятельности. К инфраструктуре относятся физические и виртуальные компоненты, поддерживающие различные приложения, сервисы и процессы[1].

Мониторинг инфраструктуры включает в себя постоянный контроль различных компонентов, таких как серверы, базы данных, сетевые устройства, приложения и сервисы, для выявления таких проблем, как узкие места в производительности, нарушения безопасности и простои[2]. Благодаря мониторингу этих компонентов организации могут выявлять и устранять проблемы до того, как они повлияют на конечных пользователей, что помогает поддерживать доступность, надежность и производительность критически важных систем.

В современном взаимосвязанном и оцифрованном мире мониторинг инфраструктуры приобретает все большее значение в связи с растущей зависимостью от технологий. Деятельность многих предприятий и организаций зависит от технологий, поэтому мониторинг инфраструктуры является важнейшей частью планов обеспечения непрерывности бизнеса. Без эффективного мониторинга инфраструктуры организации могут столкнуться с дорогостоящими простоями, потерей производительности и негативными последствиями для репутации.

Кроме того, мониторинг инфраструктуры имеет решающее значение для информационной безопасности. Благодаря мониторингу компонентов инфраструктуры организации могут вычислять угрозы нарушения безопасности и реагировать на них в режиме реального времени, что крайне важно для снижения потенциальных кибератак.

Суперкомпьютеры «Скиф Урал», «Нейрокомпьютер» и «Торнадо ЮУрГУ»

расположенные в Южно-Уральском государственном университете также требуют точного и тщательного наблюдения за всеми параметрами огромного множества различных компонентов необходимых для их корректной работы.

Выбор подхода для реализации системы

Когда речь идет о развертывании решения для мониторинга, можно рассмотреть несколько подходов, основанных на ваших конкретных требованиях, настройках инфраструктуры и потребностях в масштабируемости[3]. Ниже приведены некоторые распространенные подходы к развертыванию мониторинга:

1. Централизованный мониторинг: при подходе централизованного мониторинга, компания устанавливает единую систему мониторинга, такую как Zabbix или Prometheus, для мониторинга всех ваших ресурсов из центрального места. Это подходит для малых и средних сред, где все системы и устройства, подлежащие мониторингу, находятся в одной сети или центре обработки данных. Центральный сервер мониторинга собирает данные со всех контролируемых ресурсов, выполняет анализ, генерирует предупреждения и отчеты.

2. Распределенный мониторинг: при распределенном мониторинге компания развертывает несколько экземпляров мониторинга в разных местах или сетях. Каждый экземпляр, часто называемый узлом мониторинга или зондом, отвечает за мониторинг определенных ресурсов в назначенной области. Узлы мониторинга отправляют данные на центральный сервер или кластер серверов, которые агрегируют и анализируют собранные данные. Этот подход подходит для крупномасштабных сред или организаций с географически распределенной инфраструктурой.

3. Агентный мониторинг: агентный мониторинг предполагает установку агентов мониторинга на отдельные системы или устройства для сбора и передачи данных на центральный сервер мониторинга. Агенты активно собирают системные метрики, данные о производительности и информацию о конкретных приложениях с контролируемых устройств. Такой подход обеспечивает детальную видимость контролируемых ресурсов и позволяет осуществлять более детальный мониторинг и настройку. Примерами инструментов мониторинга на

основе агентов являются Zabbix Agents[3], Nagios Agents и Collectd.

4. Безагентный мониторинг: при безагентном мониторинге система мониторинга взаимодействует с целевыми ресурсами с помощью стандартных протоколов и интерфейсов, не требуя установки специальных агентов. Для сбора данных с контролируемых устройств используются такие протоколы, как SNMP (Simple Network Management Protocol), ICMP (Internet Control Message Protocol) или API (Application Programming Interfaces). Безагентный мониторинг часто используется для сетевых устройств, коммутаторов, маршрутизаторов и других компонентов инфраструктуры, которые поддерживают стандартные протоколы управления.

Структура и содержание работы

Выпускная квалификационная работа состоит из введения, пяти разделов, заключения, литературы. Объем работы составляет 47 страниц, включая 3 приложения, содержащие 4 страницы, объем библиографии – 11 наименования.

В первом разделе «Анализ предметной области» приведен анализ существующих систем для мониторинга инфраструктуры и используемых технологий.

Второй раздел «Описание системы Zabbix» содержит подробное описание данной системы.

Третий раздел «Проектирование» содержит функциональные и нефункциональные требования к системе, а также критерии, которые учитывались при проектировании системы.

Четвертый раздел «Конфигурация системы Zabbix и используемых скриптов» содержит описание средств разработки, выполненных изменений и разработанных компонентов.

Пятый раздел «Тестирование» включает результаты тестирования работы системы мониторинга Zabbix, а также работу скриптов в взаимодействи с этой системой.

Приложение А «Листинги конфигурационных файлов Zabbix» содержит листинги конфигурационных файлов различных компонентов, описывающие

функционал работы.

Приложение Б «Листинги скриптов» содержит листинги скриптов расширяющих систему мониторинга.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данной главе описаны существующие системы с открытым исходным кодом, предназначенные для мониторинга инфраструктуры.

1.1 Nagios

Программное приложение с открытым исходным кодом для мониторинга систем, сетей и инфраструктуры. Оно использует модель клиент-сервер и отправляет предупреждения при обнаружении каких-либо проблем. Nagios собирает информацию о состоянии узлов и служб и представляет ее в веб-интерфейсе.

Возможности:

− Плагины: небольшие исполняемые файлы, которые Nagios запускает для проверки состояния служб или хостов;

−Nagios Service Check Acceptor (NSCA): инструмент, позволяющий Nagios получать результаты проверки с удаленных хостов;

−Nagios Remote Plugin Executor (NRPE): инструмент, позволяющий Nagios выполнять плагины на удаленных узлах;

Кроме того, Nagios имеет ряд достоинств:

−Доступность: nagios предоставляет подробную информацию о доступности и времени работы системы, что помогает обеспечить доступность систем, когда это необходимо;

−Поддержка сообщества: nagios имеет большое и активное сообщество пользователей, предоставляющее поддержку, плагины и ресурсы, для помощи пользователям получить максимальную отдачу от программного обеспечения;

−Архитектура плагинов: nagios имеет модульную архитектуру плагинов, позволяющую пользователям добавлять новые функции мониторинга по мере необходимости. Это позволяет легко настраивать Nagios в соответствии с конкретными требованиями мониторинга;

−Простота в использовании: nagios имеет удобный веб-интерфейс, упрощающий управление и мониторинг ИТ-инфраструктуры. Он также предоставляет подробные отчеты и оповещения, чтобы помочь ИТ-администраторам оставаться в курсе состояния системы.

1.2 Prometeus+Grafana

Prometheus и Grafana - два популярных инструмента с открытым исходным кодом, используемые для мониторинга и визуализации систем, сетей и инфраструктуры. Prometheus - это база данных временных рядов и система мониторинга, а Grafana - инструмент визуализации и оповещения, который работает с Prometheus и другими базами данных.

Возможности:

−Pushgateway: служба, позволяющая приложениям передавать метрики в Prometheus;

− Node Exporter: этот экспортер собирает метрики аппаратного обеспечения и операционной системы из Linux/Unix систем, такие как использование процессора, памяти, дисков, сетевая статистика и многое другое

−Blackbox Exporter: позволяет осуществлять мониторинг конечных точек сети путем выполнения HTTP, TCP, ICMP и DNS зондов. Он может проверять доступность и время отклика конечных точек из разных мест.

−JMX Exporter: позволяет собирать метрики из Java-приложений, которые предоставляют метрики через Java Management Extensions (JMX). Он предоставляет возможности мониторинга метрик, связанных с JVM, сборкой мусора, количеством потоков и т.д.

−Nginx Exporter: этот экспортер собирает метрики с веб-серверов Nginx, включая запросы в секунду, коды состояния HTTP, соединения и многое другое.

−Диспетчер оповещений: менеджер оповещений — это компонент Prometheus, который обрабатывает и управляет оповещениями. Он получает оповещения от Prometheus на основе заданных правил.

−Оценка правил оповещения: Prometheus периодически оценивает заданные правила оповещения по собранным данным метрик. Если условие правила выполняется, Prometheus генерирует предупреждение.

−Уведомления об оповещениях: Менеджер оповещений затем заботится об отправке уведомлений на основе полученных оповещений. Он может отправлять оповещения по различным каналам, таким как электронная почта, Slack, PagerDuty

или другие интеграции.

Grafana — это мощный инструмент, который предлагает различные варианты создания визуально привлекательных и информативных графиков, диаграмм и приборных панелей. Grafana предоставляет удобный интерфейс с широкими возможностями настройки для визуализации и анализа данных из различных источников. Вот некоторые ключевые особенности и опции для графиков в Grafana:

−Типы визуализации: Grafana поддерживает широкий спектр типов визуализации, включая линейные графики, гистограммы, круговые диаграммы, таблицы, тепловые карты, датчики и многое другое. Компания может выбрать наиболее подходящий тип визуализации в зависимости от характера ваших данных и понимания, которое компания хочет передать.

Конфигурация панели: Каждая визуализация в Grafana создается в виде панели. Grafana позволяет настраивать панели с различными параметрами, такими как источники данных, временные диапазоны, свойства осей, легенды, пороговые значения и аннотации. Компания может настроить внешний вид и поведение каждой панели в соответствии с вашими конкретными требованиями.

1.3 Zabbix

Zabbix — это программное обеспечение с открытым исходным кодом, предназначенное для мониторинга производительности и доступности сетевых устройств, серверов и приложений. Впервые оно было выпущено в 2001 году, и с сейчас является одним из самых популярных инструментов мониторинга в мире. [4] Данное программное обеспечение использует архитектуру клиент-сервер, где сервер собирает и хранит данные с контролируемых устройств, а клиенты (или агенты) собирают и отправляют данные на сервер. Затем сервер может обрабатывать эти данные и генерировать предупреждения или отчеты на основе настроенных пороговых значений и триггеров.

Возможности:

− Мониторинг: Zabbix может контролировать широкий спектр устройств,

приложений, сервисов, сетей, баз данных, виртуальных машин и многого другого;

−Оповещения: Zabbix может генерировать оповещения на основе заданных пороговых значений и триггеров. Эти оповещения могут быть отправлены по электронной почте, SMS, Telegram;

−Отчетность: Zabbix предоставляет встроенные функции отчетности, позволяющие пользователям генерировать пользовательские отчеты на основе собранных данных. Отчеты могут быть созданы по различным показателям, таким как доступность сервера, производительность системы и поведение пользователей;

−Визуализация: Zabbix позволяет пользователям визуализировать собранные данные с помощью графиков, карт и экранов. Эта функция помогает пользователям быстро выявлять закономерности и тенденции в отслеживаемых данных;

−Безагентный мониторинг: Zabbix осуществляет мониторинг систем и служб без использования агентов на контролируемых узлах. Это облегчает развертывание и управление мониторингом в масштабе;

−Автоматическое обнаружение: Zabbix автоматически обнаруживает новые узлы и службы в сети и начинает их мониторинг. Эта функция помогает поддерживать мониторинг в актуальном состоянии по мере изменения ИТ-инфраструктуры;

−Планирование мощностей: Zabbix помогает спланировать будущие потребности в мощностях, отслеживая тенденции в использовании ресурсов и прогнозируя, когда ресурсы заканчиваются;

−Распределенный мониторинг: Zabbix настраивается для мониторинга распределенной ИТ-инфраструктуры, например, нескольких центров обработки данных или удаленных сайтов. Это помогает консолидировать данные мониторинга и обеспечить централизованное представление всей инфраструктуры.

1.4 Анализ рассмотренных решений

После всего вышеизложенного, можно сделать вывод, что у всех систем за исключением «Zabbix» отсутствовали необходимые функциональные аспекты, а именно:

- мониторинг инфраструктуры в одном месте;

- простота реализации сбора данных;
- быстрота опроса серверов;
- оповещение в месенджеры.

Рекомендуется многоядерный процессор с достойной вычислительной мощностью, особенно для крупномасштабных сред. Требования Zabbix к памяти зависят от количества контролируемых устройств и объема хранимых исторических данных. Рекомендуется иметь достаточный объем оперативной памяти, чтобы учесть размер контролируемой среды и ожидаемую нагрузку данных. Zabbix требуется дисковое пространство для хранения данных мониторинга, файлов конфигурации и журналов. Объем необходимого дискового пространства зависит от количества контролируемых устройств и срока хранения исторических данных. Убедитесь, что у вас достаточно дискового пространства для обработки ожидаемого объема данных.

В рамках анализа предметной области большое внимание было уделено системе Prometeus+Grafana. Была выполнена тестовая установка данной системы на виртуальную машину и попытка настройки системы для корректной работы с инфраструктурой и сетевым оборудованием. К сожалению, хоть создателями системы и заявляется возможность работы с метриками многих производителей, на практике этот функционал является недостаточно развитым, например, отсутствует возможность мониторинга чиллеров охлаждения или источников бесперебойного питания, что требует парсинга и описания всех параметров, получаемых по SNMP от конечных устройств, а затем предоставления этих данных в виде метрик для Prometheus.

2 ОПИСАНИЕ СИСТЕМЫ «Zabbix»

Данная система позволяет пользователю в режиме реального времени поддерживать следующие возможности:

- сбор информации о параметрах инфраструктуры;
- получение уведомлений об изменении параметров; инфраструктуры в мессенджеры, а также по электронной почте;
- построение графиков в режиме реального времени;
- хранение статистики за все время работы системы[4].

Система реализована с помощью следующих технологий:

- PHP 7.4.3;
- Nginx;
- MySQL 8.0.31;
- Bash 5.0.17;
- Ubuntu 20.04.

Стандартные компоненты Zabbix

Zabbix состоит из нескольких стандартных компонентов, которые работают вместе, чтобы обеспечить комплексное решение для мониторинга и управления. Вот основные компоненты Zabbix:

1. Zabbix Server — это основной компонент системы мониторинга Zabbix. Он отвечает за сбор и хранение данных мониторинга, выполнение предопределенных проверок и триггеров, обработку и анализ полученных данных, генерацию предупреждений и предоставление веб-интерфейса для настройки и визуализации данных. Zabbix Server взаимодействует с другими компонентами для сбора данных и управления инфраструктурой мониторинга[5].

2. Агенты Zabbix — это легкие программные компоненты, установленные на системах, которые компания хочет контролировать. Они собирают и сообщают данные на сервер Zabbix Server на основе настроенных проверок и интервалов. Агенты Zabbix могут собирать различные типы информации, такие как системные метрики (CPU, память, использование диска), сетевая статистика, данные,

специфичные для приложений, и многое другое. Они позволяют осуществлять активный и пассивный мониторинг и обеспечивают гибкость в мониторинге различных типов устройств и платформ.

3. Прокси Zabbix — это промежуточные компоненты, которые могут быть развернуты в распределенных средах мониторинга. Прокси получают данные мониторинга от агентов и направляют их на сервер Zabbix, снижая нагрузку на сервер и улучшая масштабируемость. Прокси также могут выполнять локальный сбор и предварительную обработку данных, что делает их подходящими для мониторинга удаленных сайтов или изолированных сетей. Они предоставляют возможности кэширования и помогают оптимизировать передачу и хранение данных.

4. Веб-интерфейс Zabbix — это удобный веб-интерфейс, который позволяет администраторам и пользователям настраивать систему мониторинга, определять параметры мониторинга, визуализировать данные и анализировать показатели производительности. Он предоставляет интерактивные информационные панели, графики, отчеты и предупреждения для целей мониторинга и устранения неполадок. Веб-интерфейс предлагает управление доступом на основе ролей, позволяя пользователям иметь различные уровни доступа и разрешений.

Zabbix требует реляционной системы управления базами данных (RDBMS) для хранения данных конфигурации, мониторинга и исторических данных. Он поддерживает множество баз данных, включая MySQL, PostgreSQL и SQLite. Сервер Zabbix Server взаимодействует с базой данных для хранения и получения данных для мониторинга и отчетности.

Агенты и прокси-серверы Zabbix настраиваются с помощью специальных конфигурационных файлов. Эти файлы определяют такие параметры, как имя хоста, IP-адрес, отслеживаемые элементы, интервалы и детали соединения для связи с сервером или прокси Zabbix. Конфигурационные файлы позволяют точно настроить параметры мониторинга в соответствии с конкретными требованиями каждого устройства или системы, за которыми ведется наблюдение

3 ПРОЕКТИРОВАНИЕ

3.1 Функциональные требования

Исходя из поставленной задачи, были сформированы следующие требования к системе:

− система должна предоставлять возможность администратору добавлять узлы сети;

− система должна собирать все метрики со всей инфраструктуры суперкомпьютерного центра Южно-Уральского государственного университета[6];

− система должна отправлять оповещения в любой месседжер, для быстрого реагирования на аварию;

− система должна содержать карту сети, содержащую все устройства с цветовым отображением работы узлов системы (зеленый - работает, красный - произошла ошибка);

− скрипт должен собирать данные с GPU кластера, данные должны быть переведены в читабельный вид и выведены на график.

3.2 Нефункциональные требования

Исходя из ряда требований руководства суперкомпьютерного центра Южно-Уральского государственного университета к системе были сформированы нефункциональные требования.

1. Система должна быть простой и понятной как для администратора, так и для пользователя.

2. Система должна корректно работать с системой windows и linux;

3. Основная программная реализация осуществляется с использованием Bash.

3.3 Подготовка оборудования к работе с системой

3.3.1 Планирование сетевого взаимодействия

Перед непосредственной установкой системы была выполнена подготовка сетевого пространства сервера для визуализации сетей на всех подуровнях.

С этой целью была создана схема сетевого взаимодействия. [7] На головном сервере была настроена сетевая конфигурация для работы со всеми вычислительными кластерами. Схема сети изображена на рисунке 1.

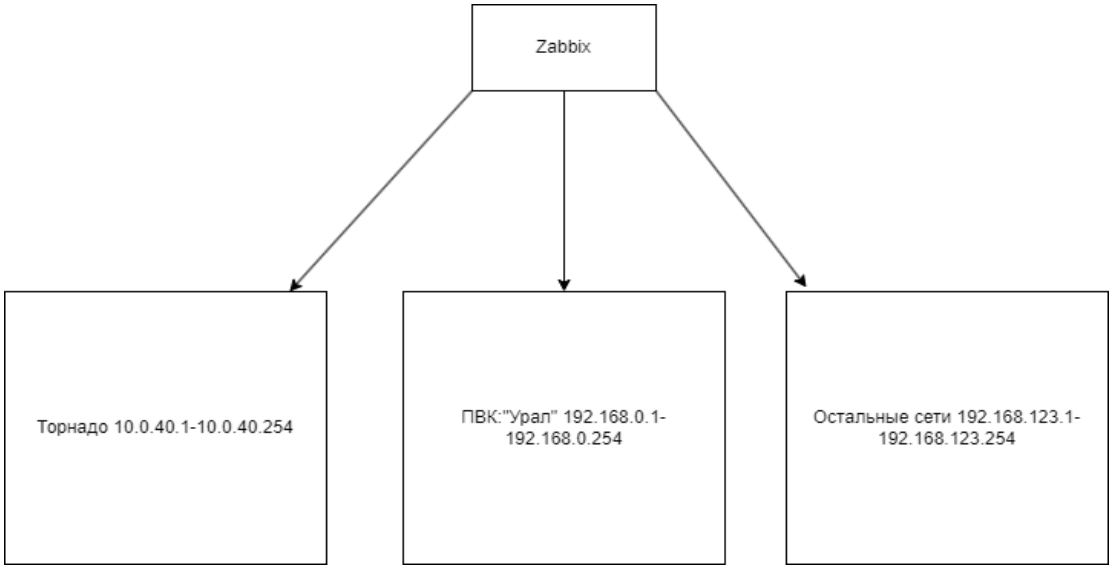


Рисунок 1 – Схема сети

Затем после настройки сети, было создано ограничение на отправку и запрос необходимые для того, чтобы не перегружать сеть и сервер Zabbix. Все метрики отправляются с разных кластеров, но при этом по требованиям безопасности Zabbix не имеет доступа к ним. [7] Код файла конфигурации приведен в листинге 1.

Листинг 1 – Файл конфигурации сети сервера Zabbix.

```
network:
  ethernets:
    eth0: название сети
      dhcp4: false отключение сетевого протокола
      dhcp6: false отключение сетевого протокола
      addresses:
      - 10.0.40.245/24 ip адрес заббикса
      nameservers:
        addresses: [10.0.40.3,192.168.10.3] ip которые он видит
      routes:
              - to: 10.0.40.0/24 ip Отправляемые на маршрутизатор
                via: 10.0.40.1

    eth1: название сети
      dhcp4: false отключение сетевого протокола
      dhcp6: false отключение сетевого протокола
      addresses:
      - 192.168.0.200/24 ip которые он видит
      nameservers:
        addresses: [10.0.40.3] ip которые он видит
      routes:
              - to: 192.168.0.0/24 ip которые он видит
                via: 192.168.0.254 ip которые он видит

    eth2: название сети
     dhcp4: false отключение сетевого протокола
     dhcp4: false отключение сетевого протокола
     addresses:
     - 192.168.123.245/24 ip адрес заббикса
     gateway4: 192.168.123.5 сетевой шлюз
     nameservers:
       addresses: [10.0.40.3,192.168.10.3,192.168.14.11,192.168.14.6] ip которые он
видит
     routes:
             - to: 192.168.123.0/24
               via: 192.168.123.5

  version: 2
```

3.3.2. Подготовка Zabbix Server

Основное требование к работе Zabbix выдвинутое руководителем суперкомпьютерного центра Южно-Уральского государственного университета, является быстрота действия сервера и отказоустойчивость, которых производится системой[8].

Перед непосредственной настройкой Zabbix на виртуальной машине была установлена операционная система, после чего была произведена установка и настройка MySQL выступающей в качестве базы данных для хранения данных Zabbix. Код фрагмента конфигурационного файла Zabbix Server изображен в листинге 2.

Листинг 2 – Фрагмент конфигурационного файла Zabbix Server

```
############ GENERAL PARAMETERS #################

### Option: ListenPort
LogFile=/var/log/zabbix/zabbix_server.log место логов
LogFileSize=0 колличество 0 бесконечный размер
PidFile=/run/zabbix/zabbix_server.pid место пидов
SocketDir=/run/zabbix где лежит ранер
DBName=zabbix название базы
DBUser=zabbix пользователь
DBPassword=12345 пароль

############ ADVANCED PARAMETERS ################
StartPollers=60
StartIPMIPollers=20
StartPollersUnreachable=3
StartPingers=500
SNMPTrapperFile=/var/log/snmptrap/snmptrap.log
HousekeepingFrequency=2
MaxHousekeeperDelete=10000
CacheSize=3092M
StartDBSyncers=20
HistoryCacheSize=1024M
HistoryIndexCacheSize=512M
ValueCacheSize=256M
Timeout=30
AlertScriptsPath=/usr/lib/zabbix/alertscripts
ExternalScripts=/usr/lib/zabbix/externalscripts
FpingLocation=/usr/bin/fping
LogSlowQueries=3000
```

Как видно из листинга, производится описание множества параметров системы, таких как параметры подключения к базе данных, параметры хранения логов, кэша и прочих параметров. В параметрах GENERAL PARAMETERS подключается база данных, задается место хранение логов размер файла; 0 имеет бесконечный размер файла. В параметрах ADVANCED PARAMETERS указывается в StartPollers минуты за которые пуллеры будут обзванивать заббикс агентов. Пуллер – обзванивается со стороны заббикс сервера, последующие 2 параметра так же указываются в секундах. StartIMPIpollers – обзванивает IPMI (удаленный доступ до сервера). StartPollersUnreachable – указывается в количестве точек пуллером. Так же указывается объем, сколько хранить кэш и таймаут, при котором опрос заканчивается.

3.4. Общее описание архитектуры системы

Администратор – пользователь, имеющий максимальные права в системе[9]. Администратор системы в сконфигурированной системе имеет следующие возможности:

- добавлять пользователей и назначать им роли;
- подключать новые узлы сети;
- создавать карту сети;
- создавать дополнительные метрики;
- удалять метрики;
- изменять ip-адреса серверов.

Компоненты изображены на рисунке 2.

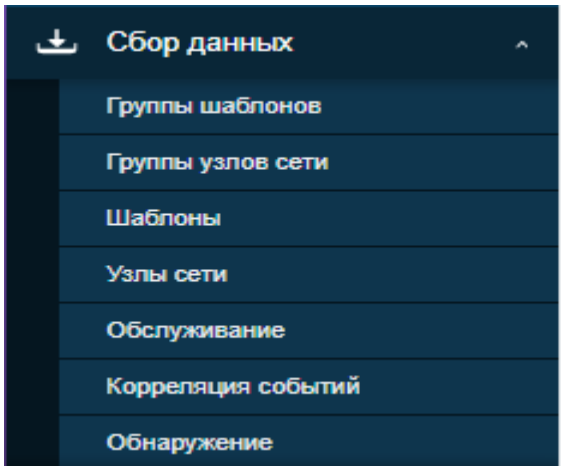


Рисунок 2 – Компоненты

Компоненты, которые были интегрированы в веб-интерфейс:

- интерфейс администратора, представленный в виде веб-интерфейса;
- интерфейс управления коммуникацией;
- интерфейс управления узлами сети;
- интерфейс графиков, с помощью которого можно выводить информацию и изменять проблемные места. Компоненты изображены на рисунке 3.

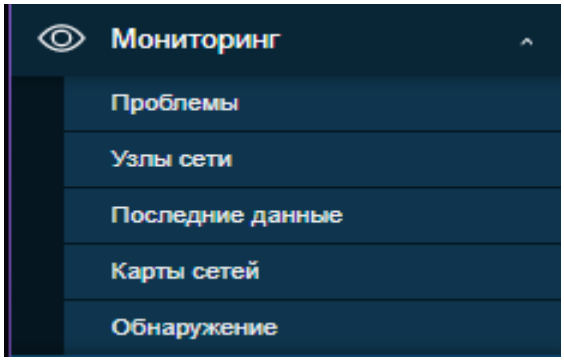


Рисунок 3 – Компоненты

Компоненты, интегрированные в серверную часть:

- функционал работы Zabbix Agent, с помощью которого можно собирать метрики с сервера;
- функционал работы с сетью, позволяющий отправлять сетевые пакеты на целевой сервер;
- функционал для работы с написанными самостоятельно скриптами, которые собирают метрики[9].

4 РЕАЛИЗАЦИЯ СИСТЕМЫ

4.1 Средства разработки

4.1.1 Используемые технологии

В данном разделе приведен анализ выбранных для реализации технологий и инструментов.

Для реализации были выбраны следующие технологии:

HTML – стандартизированный язык разметки документов во всемирной паутине;

Bash – одна из наиболее популярных современных разновидностей командной оболочки UNIX к которой можно писать различные скрипты автоматизирующие различные процессы;

Nginx– веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных операционных системах.

4.1.2 Используемое окружение

Для развертывания системы была создана виртуальная машина с операционной системой Ubuntu 20.04 на физическом сервере.

Для настройки системы «Zabbix» использовалось следующее окружение:

- PHP 7.4.3;
- Nginx 1.18.0;

- Bash 5.0.17;
- MySQL 8.0.31.

4.1.3 Инструменты разработки

Вся работа над веб-интерфейсом производилась с помощью текстового редактора vi/vim для операционных систем Linux.

4.2 Реализация работы системы

Всего было написано более 300 строк кода на языке bash для расширяющие возможности системы мониторинга.

Распишем подробнее назначение некоторых файлов, которые были созданы:

- `ssl_check.sh` – проверка цифрового сертификата, срок действия и работоспособность сертификата веб-интерфейса.
- `disc_ssl_https.sh` – вспомогательный файл для ssl_check.sh;

Данные файлы необходимы для хранения доменов и передачи на Zabbix Server.

4.2.1 Схема узлов сети на карте

В Zabbix были добавлены узлы сети (сервера) для реализации работы и наблюдением за их проблемами. Каждый узел сети был связан с сетевым оборудованием. На карте можно увидеть проблемы подписанные красным текстом под узлами сети например Replace Battery, что означает зарядить аккумуляторы, данная карта сети используется в аудитории 102 суперкомпьютерного центра Южно-Уральского государственного университета для визуального контроля состояния всего оборудования[10]. Кроме физических серверов на карту были добавлены критически важные виртуальные машины. На карте изображено: 2 хранилища Panassas,16 физических сервера для развертывания машин, 18 виртуальных машин, 8 сетевых устройств, а именно 2 маршрутизатора и 6 коммутатора. Схема узлов сети изображена на рисунке 4.

Рисунок 4 – Схема узлов сети

4.2.2 Интерфейс для работы с узлами сети

Для того, чтобы добавить узел сети, необходимо указать его имя, ip адрес, а также выполнить установку Zabbix Agent в операционной системе на этом сервере. Интерфейс работы с узлами сети изображен на Рисунке 5.

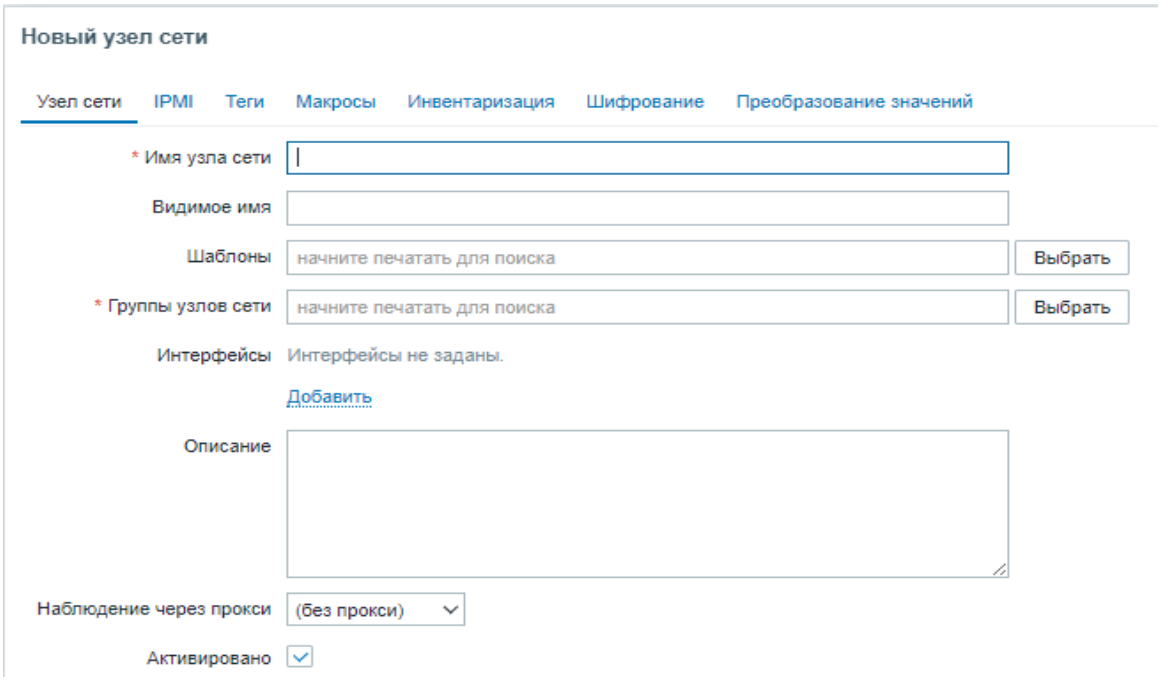


Рисунок 5– Интерфейс работы с узлами сети

Zabbix знает, на каких из узлов имеется Zabbix agent. Как только он отправляет запрос на agent, он ждет выполнения сбора данных агентом. После чего выполняет их обработку и отображение в веб-интерфейса.

4.2.3 Интерфейс для работы с графиками

Для работы с графиками был создан и настроен интерфейс внутри веб интерфейса Zabbix.

Для добавления графика необходимо настроить узел сети и убедиться в корректном получении метрик Zabbix Agent. После администратор успешно создает график. Администратор заходит во вкладку «Последние данные» вкладка изображена на рисунке 3. После чего нажимает кнопку «График» и успешно появляться график данных, которых ему нужны. На графике изображена температура чиллера и нагрузка компрессоров. Интерфейс с графиками оборудования изображен на Рисунке 6.

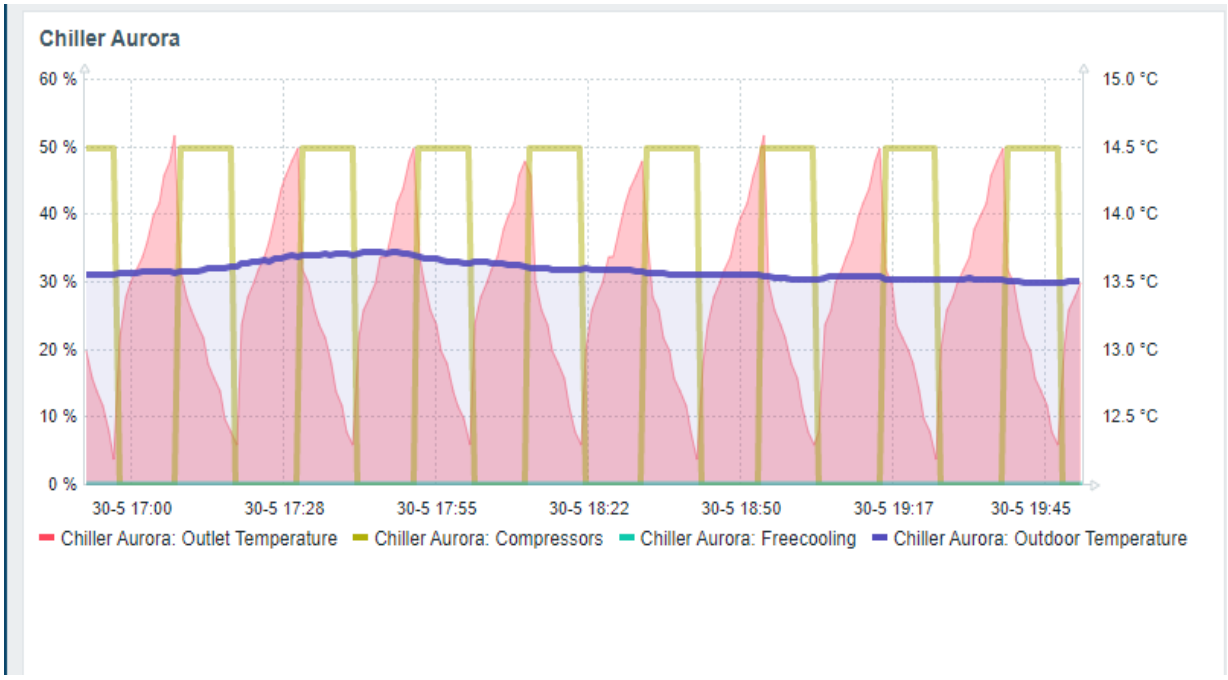


Рисунок 6 – Интерфейс структурированных графиков по оборудованию

4.2.4 Мониторинг состояния SSL сертификатов

Для получения состояния сертификатов потребовалось настроить шаблон. Шаблон будет обзванивать Zabbix agent и получать от туда метрики. Так же были добавлены ключи, собирающие метрики. Данные параметры задаются ручным образом. IPADDR-это параметр который обнаружит IP адресс. SSLPORT- это параметр посмотрит на каком порту лежат сертификаты сайта. SSLDOMAIN-это параметр, который посмотрит название доменной записи. TIMEOUT- данный параметр определяет скорость прозвонки. Если через 30 секунд сайт не отдаст сигнал то Zabbix сервер автоматически перестанет опрашивать и выдаст ошибку. Шаблон скрипта опроса состояния SSL сертификатов изображен на Рисунке 7.

```
ssl_cert_check_valid[{$IPADDR},{$SSLPORT},{$SSLDOMAIN},{$TIMEOUT}]
ssl_cert_check_expire[{$IPADDR},{$SSLPORT},{$SSLDOMAIN},{$TIMEOUT}]
```

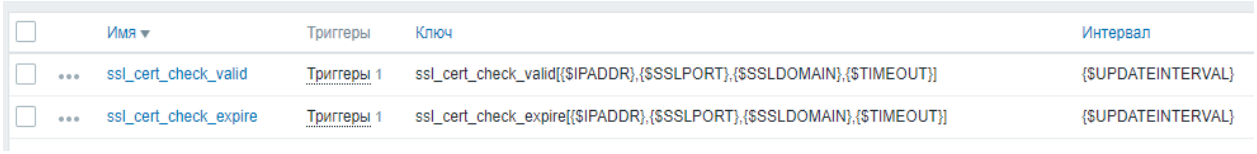


Рисунок 7 – Шаблон опроса состояния SSL сертификатов

4.2.5 Мониторинг состояния графических ускорителей

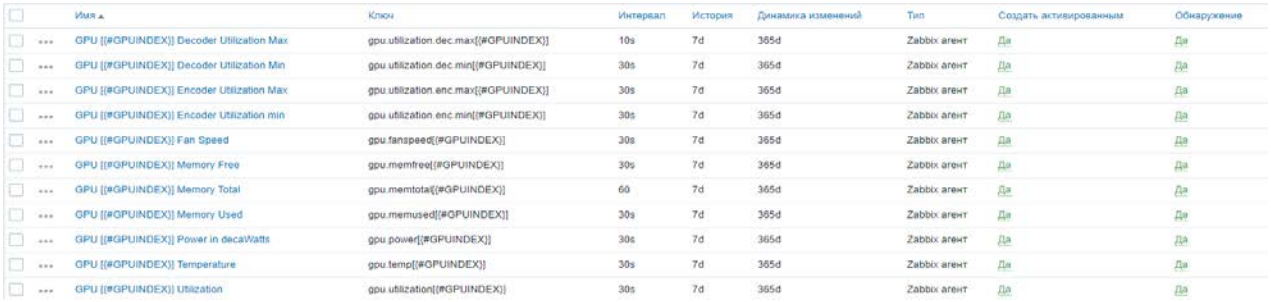
Для добавления значений утилизации графических ускорителей потребовалось настроить шаблон аналогичным способом что и с мониторингом состояния SSL сертификатов, но было необходимо указать ключи, сбор которых производиться с помощью Zabbix Agent. Шаблон мониторинга состояния графических ускорителей изображен на Рисунке 8.

```
gpu.utilization.dec.min[{#GPUINDEX}]
gpu.utilization.dec.max[{#GPUINDEX}]
gpu.utilization.enc.max[{#GPUINDEX}]
gpu.utilization.enc.min[{#GPUINDEX}]
gpu.fanspeed[{#GPUINDEX}]
gpu.memfree[{#GPUINDEX}]
gpu.memtotal[{#GPUINDEX}]
gpu.memused[{#GPUINDEX}]
```

gpu.power[{#GPUINDEX}]

gpu.temp[{#GPUINDEX}]

gpu.utilization[{#GPUINDEX}]



Имя	Ключ	Интервал	История	Динамика изменений	Тип	Создать активированным	Обнаружение
GPU [{#GPUINDEX}] Decoder Utilization Max	gpu.utilization.dec.max[{#GPUINDEX}]	10s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Decoder Utilization Min	gpu.utilization.dec.min[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Encoder Utilization Max	gpu.utilization.enc.max[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Encoder Utilization min	gpu.utilization.enc.min[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Fan Speed	gpu.fanspeed[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Memory Free	gpu.memfree[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Memory Total	gpu.memtotal[{#GPUINDEX}]	60	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Memory Used	gpu.memused[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Power in decaWatts	gpu.power[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Temperature	gpu.temp[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да
GPU [{#GPUINDEX}] Utilization	gpu.utilization[{#GPUINDEX}]	30s	7d	365d	Zabbix агент	Да	Да

Рисунок 8 – Шаблон опроса состояния графических ускорителей

Далее был настроен Zabbix Agent на серверах с графическими ускорителями. Для этого необходимо было необходимо указать пользовательские параметры в конфигурационном файле агента. Конфигурация Zabbix Agent указана на листинге 3.

Листинг 3 – Конфигурация Zabbix Agent.

```
# This is a configuration file for Zabbix agent 2 (Unix)
# To get more information about Zabbix, visit http://www.zabbix.com
############ GENERAL PARAMETERS #################
PidFile=/run/zabbix/zabbix_agent2.pid
LogFile=/var/log/zabbix/zabbix_agent2.log
LogFileSize=0
##### Passive checks related
Server=192.168.123.245,zabbix.hpc.susu.ru
############ ADVANCED PARAMETERS #################
Include=/etc/zabbix/zabbix_agent2.d/*.conf
####### USER-DEFINED MONITORED PARAMETERS #######
### Option: UserParameter
#       User-defined parameter to monitor. There can be several user-defined
parameters.
#       Format: UserParameter=<key>,<shell command>
#       See 'zabbix_agentd' directory for examples.
#
# Mandatory: no
# Default:
# UserParameter=
UserParameter=gpu.number,/usr/bin/nvidia-smi -L | /usr/bin/wc -l
UserParameter=gpu.discovery,/etc/zabbix/scripts/get_gpus_info.sh
UserParameter=gpu.fanspeed[*],nvidia-smi      --query-gpu=fan.speed      --
format=csv,noheader,nounits -i $1 | tr -d "\n"
```

```
      UserParameter=gpu.power[*],nvidia-smi       --query-gpu=power.draw       --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.temp[*],nvidia-smi     --query-gpu=temperature.gpu     --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.utilization[*],nvidia-smi  --query-gpu=utilization.gpu  --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.memfree[*],nvidia-smi     --query-gpu=memory.free     --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.memused[*],nvidia-smi     --query-gpu=memory.used     --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.memtotal[*],nvidia-smi     --query-gpu=memory.total     --
format=csv,noheader,nounits -i $1 | tr -d "\n"
      UserParameter=gpu.utilization.dec.min[*],nvidia-smi -q -d UTILIZATION -i $1
| grep -A 5  DEC | grep Min | tr -s ' ' | cut -d ' ' -f 4
      UserParameter=gpu.utilization.dec.max[*],nvidia-smi -q -d UTILIZATION -i $1
| grep -A 5  DEC | grep Max | tr -s ' ' | cut -d ' ' -f 4
      UserParameter=gpu.utilization.enc.min[*],nvidia-smi -q -d UTILIZATION -i $1
| grep -A 5  ENC | grep Min | tr -s ' ' | cut -d ' ' -f 4
      UserParameter=gpu.utilization.enc.max[*],nvidia-smi -q -d UTILIZATION -i $1
| grep -A 5  ENC | grep Max | tr -s ' ' | cut -d ' ' -f 4
      ### Option: UserParameterDir
      #       Directory to execute UserParameter commands from. Only one entry is
allowed.
      #         When executing UserParameter commands the agent will change the
working directory to the one
      #       specified in the UserParameterDir option.
      #       This way UserParameter commands can be specified using the relative
./ prefix.
      # Include configuration files for plugins
      Include=./zabbix_agent2.d/plugins.d/*.conf
```

4.2.6 Подключение Zabbix к телеграмм боту

Чтобы подключить Zabbix к боту телеграмм и получать уведомления о мониторинге через телеграмма, компания должна выполнить следующие шаги:

- zabbix использует медиатипы для определения способа отправки уведомлений. Компании потребуется установить пользовательский скрипт медиатипа для телеграмма. Этот скрипт действует как мост между Zabbix и API бота телеграмма;
- создайте бота телеграмм, обратившись к боту BotFather в

телеграмме. Следуйте инструкциям, предоставленным BotFather, чтобы создать нового бота и получить API-токен.интерфейс управления узлами сети;

- в Zabbix создайте пользователя или обновите существующего пользователя, который будет получать уведомления через телеграмм;
- определите триггеры уведомлений и действия в Zabbix, чтобы указать, когда и как должны отправляться уведомления телеграмм[11].

Точные шаги и детали конфигурации могут отличаться в зависимости от конкретного выбранного вами сценария типа медиа телеграмм и версии Zabbix, которую компания использует боту. Вывод информации в Телеграмме изображен на рисунке 9.

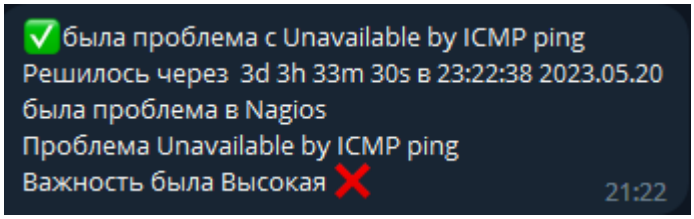


Рисунок 9 – Вывод информации

5 ТЕСТИРОВАНИЕ

Перед началом тестирования были созданы карты, добавлены узлы и сети.

Тестирование системы проводилось на 199 серверах, добавленных в веб-интерфейс Zabbix.

Тестирование веб-интерфейса осуществлялось в веб-браузерах, таких как Mozilla Firefox 92.0, Google Chrome 101.0 и Opera GX.

В ходе выполнения каждого шага тестирования создавались скриншоты экрана компьютера, на котором производились операции тестирования системы.

5.1 Тестирование работы системы

Для тестирования системы использовалось функциональное тестирование, в ходе которого проверялась работоспособность всех необходимых функций. Результаты функционального тестирования совпадают с ожидаемыми и приведены в Таблице 3 функциональное тестирование пройдено успешно.

Таблица 3 – Результаты функционального тестирования

№	Название теста	Действие	Ожидаемый результат	Тест пройден?
1	Проверка корректного созда- ния узла сети	Войти в систему с учетными данными администратора, перейти на страницу «Администрирование», нажать на вкладку «Добавить узел сети»	При незаполнении необходимых полей появляется сообщение «Это поле обязательно». В случае если ip адрес указан а остальные параметры нет, получение ошибки «Некорректное значение для поля Host». В случае корректного ввода всех полей создание узла сети происходит успешно.	Да
2	Опрос GPU	Войти в систему учетными данными администратора, перейти на страницу «Узлы сети», нажать на кнопку «Последние данные», затем «GPU» после чего осуществить проверку узла сети.	В случае если опрос завершиться неуспешно будет получена ошибка. Если опрос завершиться успешно, то будут отображены метрики	Да
3	Опрос SSL	Войти в систему учетными данными администратора, перейти на страницу «Узлы сети», нажать на кнопку «Последние данные», затем «SSL» дальше осуществить проверку узла сети.	В случае если опрос завершиться неуспешно, он выдаст ошибку. Если опрос завершиться успешно, то метрика даты завершения сертификата будет выведена на экран.	Да
4	Проверка работы Zabbix Agent на целевом сервере	Войти в систему по SSH, ввести команду «service zabbix-agent status»	В случае если имеются ошибки конфигурации, то будет получена ошибка и ее статус. В случае если ошибок нет будет выведен статус «OK».	Да

Окончание таблицы 3

№	Название теста	Действие	Ожидаемый результат	Тест пройден?
5	Проверка работы Zabbix Server на виртуальной машине	Войти в систему по SSH, ввести команду «service zabbix-server status»	В случае если имеются ошибки конфигурации, то будет получена ошибка и ее статус. В случае если ошибок нет будет выведен статус «OK».	Да
6	Проверка работы веб-интерфейса php	Войти в браузер по адресу: zabbix.hpc.susu.ru и осуществить проверку работы сайта.	В случае если имеются какие-то неисправности, Zabbix сообщит о проблеме В случае успешной работы веб-интерфейса будет получено приглашение к входу в систему.	Да
7	Отправка сообщений в мессенджере	Войти в систему веб-интерфейса под учетными данными администратора, выполнить отключение узла сети.	В случае успешного выполнения система автоматически отправит оповещение в мессенджер подключенный к системе	Да

Результаты нефункционального тестирования совпадают с ожидаемыми и приведены в Таблице 4. Нефункциональное тестирование пройдено успешно.

Таблица 4 – Результаты нефункционального тестирования

№	Название теста	Действие	Ожидаемый результат	Тест пройден?
1	Построение графика	Нажать на «Узлы сети». В списке выбрать нужный узел сети и открыть его.	Менее чем за 5 секунд происходит построение графика и вывод текущей информации.	Да
3	Проверка корректности SSL-сертификата сайта.	Открыть страницу веб-интерфейса.	Убедиться, что SSL-сертификат является доверенным.	Да
4	Проверка корректности отображения списка узлов сети.	Открыть страницу работы с узлами сети.	Список узлов сети отображается корректно.	Да

ЗАКЛЮЧЕНИЕ

Целью данной работы являлась разработка внешних скриптов для системы мониторинга Zabbix.

В ходе выполнения выпускной квалификационной работы бакалавра были решены следующие задачи:

- выполнен анализ предметной области;
- спроектированы необходимые компоненты для работы системы виртуализации Hyper-v в суперкомпьютерном центре Южно-Уральского государственного Университета;
- скомпилирован Zabbix сервер и внедрен в инфраструктуру;
- произведено тестирование развернутой системы.

В рамках данной выпускной квалификационной работы была настроена система мониторинга Zabbix и разработаны внешние скрипты расширяющие возможности системы.

Полученная система позволяет суперкомпьютерного центра Южно-Уральского государственного университета эффективно диагностировать состояние инфраструктуру и вовремя решать проблемы, возникающие в её работе.

В результате выполнения выпускной квалификационной работы были решены все поставленные задачи, таким образом, цель данной работы достигнута. Система внедрена в суперкомпьютерный центр.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Самохин Н. Мониторинг производительности ваших сетевых устройств и приложений с помощью совершенно нового. / [Электронный ресурс] // Zabbix Сетевой мониторинг систем - 2018. - URL: https://www.packtpub.com/product/zabbix-cookbook/9781784397586 / (Дата обращения: 26.02.2023г.).

2 Далле Вакче А. Эффективный мониторинг большой ИТ-среды с помощью. Zabbix/ [Электронный ресурс] // пэкт паблишинг-2021 - URL: https://www.packtpub.com/product/mastering-zabbix-second-edition/9781785289262 / (Дата обращения: 26.02.2023г.).

3 Зотов С. Использование Zabbix для мониторинга гетерогенной сети с работой по проводным и радиоканалам. // Научно-исследовательские публикации. 2017. №1, (39), С. 30-39.

4 Чистяков В., Система мониторинга Zabbix, К. Чистяков, Д. Мыктыбаев В. Жанбеков, В. Котяшев. // Science Time. 2015. Выпуск номер 12 (24), С. 836-839.

5 Лазарева Н., Системы мониторинга оборудования, К. Горбачев // E-Scio. 2020. №2 С. 89-93.

6 Суперкомпьютер «Торнадо ЮУрГУ» суперкомпьютерного центра Южно-Уральского государственного университета. [Электронный ресурс] - URL: http://supercomputer.susu.ru/computers/tornado/ (Дата обращения 10.05.2023 г.).

7 Костенко Е., Система мониторинга для контроля трафика технологических сетей передачи данных, Р. Дуйсенгалиев, Е. Барабанова // Вестник Астраханского государственного технического университета. Серия: Управление, вычислительная техника и информатика. 2015. Выпуск номер 4, С.101-109.

8 Шардаков К.С. Сравнительный анализ популярных систем мониторинга сетевого оборудования, распространяемых по лицензии // Интеллектуальные технологии на транспорте. 2018. № 1 (13), С. 44-48.

9 Олупс Р. Мониторинг сети Zabbix — второе издание. / [Электронный ресурс] // Zabbix Сетевой мониторинг систем - 2018. – URL: https://www.packtpub.com/product/zabbix-network-monitoring-second-edition/9781782161288/ (дата обращения: 26.02.2023).

10 Федорова Л.Системы мониторинга. Обзор и сравнение. // Вестник науки и образования. 2020. Выпуск номер 10-4 (88), С.16-18. (дата обращения: 26.02.2023г.).

11 Самохин Н. Информация об использовании ресурсов ЦОД при реализации брокера сообщений С. Хоружников, Р. Трубникова, В. Ахмедзянова, Д. Булыкина. // Научно-технический вестник информационных технологий, механики и оптики. 2018. №5 С. 858-862. (Дата обращения: 26.02.2023г.).

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Функционал Nginx

Листинг А. 1 – Конфигурационный файл Nginx

```
server {
       listen 80;
       server_name zabbix.local.server;
       access_log /var/log/nginx/access-zabbix.local.server.log;
       error_log /var/log/nginx/error-zabbix.local.server.log info;
       root /usr/share/zabbix;
       index index.php index.html;
       auth_basic "Nagios Restricted Access";
       auth_basic_user_file /etc/zabbix/passwd;

       location / {
              root   /usr/share/zabbix;
              index  index.php index.html;
              }

       location ~ \.php$ {
              fastcgi_pass php-fpm;
              fastcgi_index index.php;
                 fastcgi_param PHP_VALUE "
                  max_execution_time = 300
                  memory_limit = 128M
                  post_max_size = 16M
                  upload_max_filesize = 2M
                  max_input_time = 300
                  date.timezone = Europe/Kiev
                  always_populate_raw_post_data = -1
                 ";
              fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
              try_files $uri =404;
              include fastcgi_params;
       }
       location = /favicon.ico {
              log_not_found off;
              access_log off;
       }
       location ~*
".+\.(?:ogg|ogv|svg|svgz|eot|otf|woff|mp4|ttf|rss|css|swf|js|atom|jpe?g|gif|png|ico|zip|
tgz|gz|rar|bz2|doc|xls|exe|ppt|tar|mid|midi|wav|bmp|rtf)$" {
              access_log  off;
              log_not_found       off;
              expires             max;
              }

         location ~* ^/(conf|api|include)($|\/) {
                  deny all;
          }
         location ~ /\.ht {
                 deny all;
         }
         location ~ /\. {
                 deny all;
         }
}
```

Листинг А.2 – Конфигурационный файл MySQL

```
# Settings user and group are ignored when systemd is used.

# If you need to run mysqld under a different user or group,
# customize your systemd unit file for mysqld according to the
# instructions in http://fedoraproject.org/wiki/Systemd

[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
log-error=/var/log/mysql/mysqld.log
pid-file=/run/mysqld/mysqld.pid

binlog_expire_logs_seconds      = 404800
max_binlog_size   = 100M
binlog-expire-logs-auto-purge = ON
innodb_buffer_pool_size = 6G
#innodb_log_file_size = 512M
innodb_redo_log_capacity = 100M
innodb_log_buffer_size = 2M
innodb_flush_method = O_DIRECT
innodb_flush_log_at_trx_commit = 2
                return nodes def getDoubleNodes():
```

Листинг А. 3 – Конфигурационный файл Zabbix Agent GPU

```
# This is a configuration file for Zabbix agent 2 (Unix)
# To get more information about Zabbix, visit http://www.zabbix.com
############ GENERAL PARAMETERS #################
PidFile=/run/zabbix/zabbix_agent2.pid
LogFile=/var/log/zabbix/zabbix_agent2.log
LogFileSize=0
##### Passive checks related
Server=192.168.123.245,zabbix.hpc.susu.ru
############ ADVANCED PARAMETERS #################
Include=/etc/zabbix/zabbix_agent2.d/*.conf
####### USER-DEFINED MONITORED PARAMETERS #######
### Option: UserParameter
#       User-defined parameter to monitor. There can be several user-defined parameters.
#       Format: UserParameter=<key>,<shell command>
#       See 'zabbix_agentd' directory for examples.
#
# Mandatory: no
# Default:
# UserParameter=
UserParameter=gpu.number,/usr/bin/nvidia-smi -L | /usr/bin/wc -l
UserParameter=gpu.discovery,/etc/zabbix/scripts/get_gpus_info.sh
UserParameter=gpu.fanspeed[*],nvidia-smi --query-gpu=fan.speed --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.power[*],nvidia-smi --query-gpu=power.draw --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.temp[*],nvidia-smi --query-gpu=temperature.gpu --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.utilization[*],nvidia-smi --query-gpu=utilization.gpu --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memfree[*],nvidia-smi --query-gpu=memory.free --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memused[*],nvidia-smi --query-gpu=memory.used --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memtotal[*],nvidia-smi --query-gpu=memory.total --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.utilization.dec.min[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
DEC | grep Min | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.dec.max[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
DEC | grep Max | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.enc.min[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
ENC | grep Min | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.enc.max[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
ENC | grep Max | tr -s ' ' | cut -d ' ' -f 4
### Option: UserParameterDir
#       Directory to execute UserParameter commands from. Only one entry is allowed.
#       When executing UserParameter commands the agent will change the working
directory to the one
#       specified in the UserParameterDir option.
#       This way UserParameter commands can be specified using the relative ./ prefix.
# Include configuration files for plugins
```

Листинг А. 4 – Конфигурационный файл Zabbix.php

```
<?php
// Zabbix GUI configuration file.

$DB['TYPE']
     = 'MYSQL';
$DB['SERVER']                   = 'localhost';
$DB['PORT']                     = '0';
$DB['DATABASE']                 = 'zabbix';
$DB['USER']                     = 'zabbix';
$DB['PASSWORD']                 = 'zabbixbd';

// Schema name. Used for PostgreSQL.
$DB['SCHEMA']                   = '';

// Used for TLS connection.
$DB['ENCRYPTION']               = false;
$DB['KEY_FILE']                 = '';
$DB['CERT_FILE']                = '';
$DB['CA_FILE']                  = '';
$DB['VERIFY_HOST']              = false;
$DB['CIPHER_LIST']              = '';
// Vault configuration. Used if database credentials are stored in Vault secrets
manager.
$DB['VAULT']                    = '';
$DB['VAULT_URL']                = '';
$DB['VAULT_DB_PATH']            = '';
$DB['VAULT_TOKEN']              = '';
$DB['VAULT_CERT_FILE']          = '';
$DB['VAULT_KEY_FILE']           = '';
// Uncomment to bypass local caching of credentials.
// $DB['VAULT_CACHE']           = true;
// Use IEEE754 compatible value range for 64-bit Numeric (float) history values.
// This option is enabled by default for new Zabbix installations.
// For upgraded installations, please read database upgrade notes before enabling
this option.
$DB['DOUBLE_IEEE754']           = true;
// Uncomment and set to desired values to override Zabbix hostname/IP and port.
// $ZBX_SERVER                  = '';
// $ZBX_SERVER_PORT             = '';
$ZBX_SERVER_NAME                = 'Zabbix LSM';
$IMAGE_FORMAT_DEFAULT   = IMAGE_FORMAT_PNG;

// Uncomment this block only if you are using Elasticsearch.
// Elasticsearch url (can be string if same url is used for all types).
//$HISTORY['url'] = [
//      'uint' => 'http://localhost:9200',
//      'text' => 'http://localhost:9200'
//];
// Value types stored in Elasticsearch.
//$HISTORY['types'] = ['uint', 'text'];

// Used for SAML authentication.
           // Uncomment to override the default paths to SP private key, SP and IdP
X.509 certificates, and to set extra settings
//$SSO['SP_KEY']                        = 'conf/certs/sp.key';
//$SSO['SP_CERT']                       = 'conf/certs/sp.crt';
//$SSO['IDP_CERT']              = 'conf/certs/idp.crt';

//$SSO['SETTINGS']              = [];
```

ПРИЛОЖЕНИЕ Б

Функционал скриптов

Листинг Б. 5 – скрипт Bash для SSL

```
#!/bin/bash

JSON=$(for i in `cat /etc/zabbix/scripts/ssl_smtp.txt`; do printf
"{\"{#DOMAIN_SMTP}\":\"$i\"},"; done | sed 's/^\(.*\).$/\1/')
printf "{\"data\":["
printf "$JSON"
printf "]}"
```

Листинг Б. 6 – скрипт определение сколько дней осталось SSL

```
#!/bin/bash

SERVER=$1
TIMEOUT=25
RETVAL=0
TIMESTAMP=`echo | date`
EXPIRE_DATE=`echo | openssl s_client -connect $SERVER:443 -servername $SERVER -
tlsextdebug 2>/dev/null | openssl x509 -noout -dates 2>/dev/null | grep notAfter |
cut -d'=' -f2`
EXPIRE_SECS=`date -d "${EXPIRE_DATE}" +%s`
EXPIRE_TIME=$(( ${EXPIRE_SECS} - `date +%s` ))
if test $EXPIRE_TIME -lt 0
then
RETVAL=0
else
RETVAL=$(( ${EXPIRE_TIME} / 24 / 3600 ))
fi

echo ${RETVAL}
```

Листинг Б. 7 – параметры в Zabbix Agent на сервере с сайтом.

```
UserParameter=ssl_https.discovery[*],/etc/zabbix/scripts/disc_ssl_https.sh
UserParameter=ssl_https.expire[*],/etc/zabbix/scripts/check_ssl_https.sh $1
UserParameter=ssl_smtp.discovery[*],/etc/zabbix/scripts/disc_ssl_smtp.sh
UserParameter=ssl_smtp.expire[*],/etc/zabbix/scripts/check_ssl_smtp.sh $1
```

Листинг Б. 8 – скрипт Bash для работы с GPU

```
#!/bin/bash

result=$(/usr/bin/nvidia-smi -L)
first=1

echo "{"
echo "\"data\":["

while IFS= read -r line
do
  if (( "$first" != "1" ))
  then
    echo ,
  fi
  index=$(echo -n $line | cut -d ":" -f 1 | cut -d " " -f 2)
  gpuuuid=$(echo -n $line | cut -d ":" -f 3 | tr -d ")" | tr -d " ")
  echo -n {"\"{#GPUINDEX}"\":\"$index"\", \"{#GPUUUID}"\":\"$gpuuuid\"}
  if (( "$first" == "1" ))
  then
#    echo ,
    first=0
  fi
done < <(printf '%s\n' "$result")

echo
echo "]"
echo "}"
```

Листинг Б. 9 – Параметры в Zabbix Agent на сервере с GPU для Linux

```
UserParameter=gpu.number,/usr/bin/nvidia-smi -L | /usr/bin/wc -l
UserParameter=gpu.discovery,/etc/zabbix/scripts/get_gpus_info.sh
UserParameter=gpu.fanspeed[*],nvidia-smi --query-gpu=fan.speed --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.power[*],nvidia-smi --query-gpu=power.draw --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.temp[*],nvidia-smi --query-gpu=temperature.gpu --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.utilization[*],nvidia-smi --query-gpu=utilization.gpu --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memfree[*],nvidia-smi --query-gpu=memory.free --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memused[*],nvidia-smi --query-gpu=memory.used --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.memtotal[*],nvidia-smi --query-gpu=memory.total --
format=csv,noheader,nounits -i $1 | tr -d "\n"
UserParameter=gpu.utilization.dec.min[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
DEC | grep Min | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.dec.max[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
DEC | grep Max | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.enc.min[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
ENC | grep Min | tr -s ' ' | cut -d ' ' -f 4
UserParameter=gpu.utilization.enc.max[*],nvidia-smi -q -d UTILIZATION -i $1 | grep -A 5
ENC | grep Max | tr -s ' ' | cut -d ' ' -f 4
```

Листинг Б. 10 – Параметры в Zabbix Agent на сервере с GPU для Windows

```
UserParameter=gpu.number,"nvidia-smi.exe" -L | find /c /v ""
UserParameter=gpu.discovery,C:\scripts\get_gpus_info.bat
UserParameter=gpu.fanspeed[*],"nvidia-smi.exe" --query-gpu=fan.speed --
format=csv,noheader,nounits -i $1
UserParameter=gpu.power[*],"nvidia-smi.exe" --query-gpu=power.draw –
```

```
format=csv,noheader,nounits -i $1
UserParameter=gpu.temp[*],"nvidia-smi.exe" --query-gpu=temperature.gpu --
format=csv,noheader,nounits -i $1
UserParameter=gpu.utilization[*],"nvidia-smi.exe" --query-gpu=utilization.gpu --
format=csv,noheader,nounits -i $1
UserParameter=gpu.memfree[*],"nvidia-smi.exe" --query-gpu=memory.free --
format=csv,noheader,nounits -i $1
UserParameter=gpu.memused[*],"nvidia-smi.exe" --query-gpu=memory.used --
format=csv,noheader,nounits -i $1
UserParameter=gpu.memtotal[*],"nvidia-smi.exe" --query-gpu=memory.total --
format=csv,noheader,nounits -i $
```