Архитектура системы мониторинга центрального информационно-вычислительного комплекса ОИЯИ

В.В. Кореньков, В.В. Мицын, П.В. Дмитриенко

Аннотация. Для функционирования сложной распределенной вычислительной системы и входящих в ее состав ресурсных центров требуется качественная система мониторинга, дающая подробную картину функционирования и производительности её элементов, своевременно оповещающая о сбоях и позволяющая проводить комплексный анализ работы системы. В статье рассмотрена информационная модель взаимодействия компонент системы мониторинга и предложена её архитектура, реализующая принципы универсальности и расширяемости. Рассмотрено решение задач, возникших в процессе построения системы мониторинга ЦИВК ОИЯИ на основе открытого программного продукта Nagios.

Ключевые слова: распределенные вычислительные системы**,** мониторинг, сети, Nagios**.**

Введение

Проводимые в настоящее время Европейской организацией ядерных исследований на Большом адронном коллайдере крупномасштабные эксперименты в области физики высоких энергий, а также будущие эксперименты на ускорителе NICA в ОИЯИ и проекте FAIR (SIS100/300) в Дармштадте требуют объединения мощности множества географически распределенных вычислительных систем (РВС), предоставляющих доступ к разного рода службам (законченным функциональным программным компонентам, доступным посредством интернет-протоколов [1, 2]) для обработки данных. К наиболее динамично развивающимся системам такого типа относятся грид (компьютерная инфраструктура, обеспечивающая глобальную интеграцию информационных и вычислительных ресурсов) и облачные вычисления (концепция предоставления пользователю компьютерных ресурсов и мощностей в виде интернет-сервисов).

Ключевым элементом в обеспечении бесперебойной работы этих сложных систем является качественная система мониторинга, своевременно оповещающая о сбоях, позволяющая проводить комплексный анализ работы системы, дающая подробную картину функционирования и производительности её элементов – объектов мониторинга. *Объект мониторинга* – это устройство либо служба, за которым осуществляется регулярное наблюдение с целью контроля над его состоянием, анализа протекающих с его участием процессов, выявления и прогнозирования нештатных состояний. *Нештатное (критическое) состояние* объекта мониторинга – состояние, препятствующее корректной работе системы, над которой осуществляется контроль; *нештатная ситуация* – ситуация нахождения одного или более объектов мониторинга в нештатном состоянии. *Событие* в системе мониторинга – более широкий термин, обозначающий любое значимое для наблюдателя изменение состояния объекта мониторинга.

На примере грид-систем можно выделить несколько «точек зрения» на объекты мониторинга РВС. Наиболее широкий взгляд свойственен гло-

бальному мониторингу с обновляемым в реальном времени отображением состояний, активности и взаимодействия ресурсных центров и пользователей грид-служб например, на географической карте [3]. Следующий уровень детализации – это мониторинг отдельной виртуальной организации (динамичного целевого объединения пользователей, ресурсов и служб) – здесь важна информация об исполнении текущих задач, их распределении между отдельными сайтами (стабильными уникально идентифицируемыми наборами служб, поставщиков и ресурсов), взаимоотношениях их отправителей и исполнителей (например, для проекта Alice: http://alimonitor.cern.ch/reports/). Наконец, локальный мониторинг предоставляет данные о состоянии служб, их поставщиков и ресурсов, входящих в состав отдельного сайта.

В соответствии со спецификой контроля и обслуживания объекты локального мониторинга можно распределить по трём уровням.

1) На нижнем, аппаратном уровне осуществляется сбор и отображение данных об отдельных узлах сети, их аппаратном обеспечении и операционных системах; проверяется доступность их по сети, загруженность процессоров, оперативной и дисковой памяти, состояние источников питания, температурный режим.

2) На сетевом уровне рассматриваются устройства и службы, обеспечивающие работу локальной сети: состояние памяти и загрузки процессоров коммутаторов, характеристики их портов; состояние внешнего канала и доступность необходимых для работы сетей.

3) На верхнем уровне – уровне служб – осуществляется контроль работы служб, предоставляемых конечным пользователям (людям, использующим подконтрольные системе мониторинга ресурсы для решения стоящих перед ними задач).

Между объектами этих уровней могут существовать зависимости, например, корректная работа службы доступа к файлам (уровень служб) зависит от состояния компьютеров, предоставляющих для неё дисковое пространство (аппаратный уровень) и коммутатора, обслуживающего сегмент сети, в котором они располагаются (сетевой уровень).

Существует ряд решений задачи мониторинга для отдельных классов систем. Эти решения различаются:

- по виду охватываемых ресурсов: сетевой мониторинг, анализ протоколов, диагностика и управление конкретными устройствами, мониторинг грид-служб и др.;
- по используемым технологиям, например, с использованием протокола SNMP [4] или на основе активных автономных элементов – агентов;
- по условиям использования: коммерческие, условно-бесплатные, свободные (GPL).

Важными недостатками развитых коммерческих продуктов в качестве решений для крупных научных, национальных, а также и учебных проектов видятся их дороговизна и закрытость программного кода. Поэтому актуальной представляется задача разработки архитектуры, методики построения и рабочего прототипа инструмента мониторинга вычислительных ресурсов, обладающего следующими свойствами:

а) отсутствием ограничения по виду охватываемых ресурсов или сетевым параметрам (универсальность);

б) реализацией всех трёх уровней локального мониторинга и поддержка зависимостей между их объектами;

в) базированием на свободном программном обеспечении с открытым кодом.

Такой программный комплекс должен осуществлять наблюдение за интересующими объектами различной природы, извещать о сбоях и предоставлять средства для их анализа, в удобной форме представлять данные о функционировании системы. Изложенные в статье положения были использованы при разработке прототипа, а затем и рабочей версии системы локального мониторинга Центрального информационно-вычислительного комплекса (ЦИВК) ОИЯИ.

1. Структура системы локального мониторинга

Применение метода функциональной декомпозиции [5] позволяет выделить составные части любой системы мониторинга (СМ).

1) *Подсистема сбора данных* – осуществляет опрос объектов мониторинга с заданными временными интервалами для получения значений исследуемых параметров этих объектов. Может также включать в себя первичный анализ полученных данных с целью, например, квалификации полученных значений как нормальных, требующих вмешательства оператора либо критических.

2) *Подсистема хранения* – отвечает за накопление, хранение, архивацию данных о результатах проверок. Включает компоненты для работы с базами данных (БД) или иными репозиториями, программные средства сжатия данных для уменьшения объема хранимой информации и т.п.

3) *Подсистема анализа данных* – включает компоненты, производящие исследования данных, накопленных системой, их статистический анализ, нахождение корреляционного отношения величин и тому подобные операции.

Эти три подсистемы решают задачи, относящиеся к *фоновому мониторингу* – то есть систематическому долговременному накоплению, классификации и анализу данных о работе объектов мониторинга, не подразумевающему какую-либо реакцию на получаемые данные.

4) *Подсистема оповещения* – отвечает за уведомление лиц, ответственных за функционирование проверяемых объектов и самой системы мониторинга о нештатных ситуациях и других значимых изменениях состояний объектов.

5) *Подсистема вывода* – отвечает за представление информации о работе системы и результатов проверок в виде, удобном для восприятия пользователем, причём независимо от его местонахождения и используемой операционной системы. Данное требование обуславливает реализацию подсистемы вывода в виде веб-интерфейса. Поскольку количество объектов мониторинга и объемы собранных данных могут быть весьма большими, необходимо предусмотреть генерацию различных типов отчетов и сводок (таблицы, графики, секторные и иные диаграммы), а сам веб-интерфейс должен предоставлять средства удобной навигации и поиска необходимых данных.

6) *Подсистема коррекции* – предоставляет возможность выполнения системой действий по устранению возникших нештатных ситуаций. Включает компоненты для выбора и осуществления подходящих действий в соответствии с типом проблемы и другими параметрами.

Эти подсистемы ориентированы на *оперативный мониторинг* – то есть направленный на оценку текущей работоспособности и эффективности исследуемых объектов (как с помощью пользователя СМ, так и автоматически), а также на немедленную реакцию на обнаруженные нештатные ситуации.

Выделим внешние по отношению к системе мониторинга объекты и субъекты, с которыми она должна взаимодействовать:

а) объекты мониторинга;

б) одна или более систем хранения данных, например системы управления базами данных (СУБД);

в) пользователи (лица, ответственные за корректную работу объектов мониторинга; обслуживающий персонал самой СМ; пользователи-гости, которым предоставляется информация о текущем состоянии объектов мониторинга).

Функции (отдельные подпрограммы) и программные модули (функционально законченные поименованные фрагменты) подсистем можно разделить на две категории:

- реализующие внутренние процессы самой СМ (информационные потоки между подсистемами; механизмы передачи данных, запуска процедур сбора, оповещения, сохранения и архивации данных; функции обработки данных);

- осуществляющие взаимодействие с внешними объектами – их реализация зависит от вида этих объектов и предоставляемых ими протоколов взаимодействия или интерфейсов.

Функции и модули первой категории должны быть реализованы максимально просто и универсально. Назовём их совокупность *ядром системы мониторинга*. Ядро должно предоставлять сотруднику, сопровождающему систему, гибкий инструментарий для построения оптимального решения стоящих перед ним задач, связанных с конкретным набором внешних объектов. Система должна позволять легко менять конкретную реализацию функций и модулей второй категории (и добавлять новые) в соответствии со стоящей задачей. Сопровождающий систему сотрудник должен иметь возможность реализовать

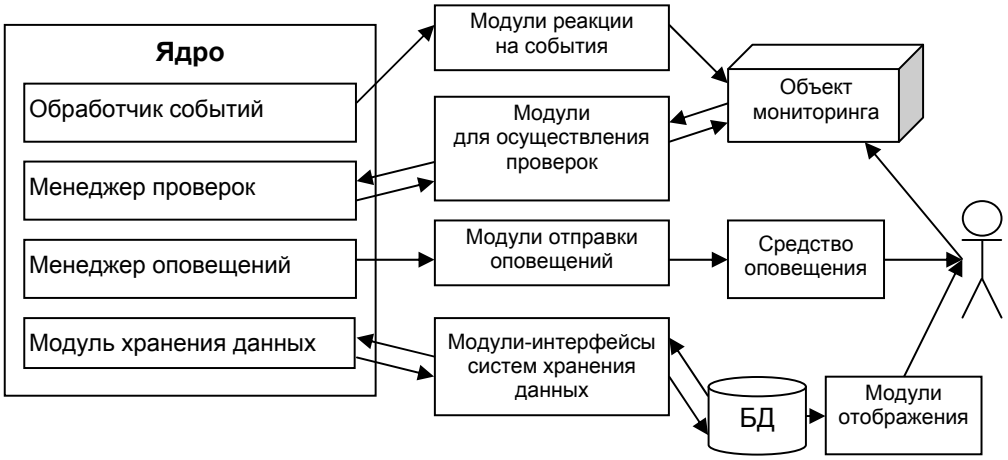

Рис. 1. Архитектура системы мониторинга

и использовать в системе собственные модули оповещения, вывода, сбора и анализа данных, работающие в рамках поддерживаемой и регламентируемой ядром схемы.

Указанным требованиям к архитектуре системы соответствует вариант «главная программа (core program) + подключаемые модули/агенты (plugins)», характерный, например, для Nagios или Munin [6, 7]. Подобные системы позволяют осуществлять мониторинг объектов любой природы – при условии разработки программного модуля, поставляющего интересующие данные, и выполнять в критической ситуации любые действия – в случае разработки модуля, реализующего при выполнении некоторого условия требуемую последовательность операций. На Рис. 1 приведена архитектурная схема такой системы мониторинга. Ядро системы составляют компоненты, реализующие её основные внутренние механизмы (получение и накопление данных об объектах мониторинга произвольной природы, диагностика нештатных ситуаций и оповещение о них, принятие решения о надлежащей реакции на обнаруженную нештатную ситуацию). Компоненты ядра осуществляют вызов необходимых подключаемых модулей, реализующих решения частных задач, соответствующих области применения системы (мониторинг конкретных типов объектов и служб, взаимодействие с системами хранения данных, отображение состояния системы в различных форматах и т.п.). Отметим, что функции подсистемы вывода целиком вынесены из ядра как не требующие взаимодействия с самими процессами мониторинга, а взаимодействующее только с поставляемыми системой хранения накопленными данными (как об объектах мониторинга, так и о работе самой системы).

2. Характеристики системы Nagios

Сравнительный анализ существующего программного обеспечения [8, 9], отвечающего перечисленным выше требованиям к ядру, позволил при разработке рабочего прототипа остановить выбор на системе Nagios, которая осуществляет вызов модулей проверки, оповещения и реакции в соответствии с конфигурационными файлами, описывающими объекты мониторинга и правила взаимодействия с ответственными за их функционирование лицами. Модули Nagios являются исполняемыми файлами, возвращающими текстовое значение, реализуются на любом удобном языке (Perl, PHP, bash) и могут вызываться независимо от самой системы, что упрощает их написание и отладку. Работа ядра Nagios заключается в запуске указанных для каждого контролируемого параметра проверочных команд с заданными интервалами. В случае возвращения командой статуса Warning либо Critical, производится оповещение указанных лиц и запуск модулей реакции на наступившие нештатные ситуации в соответствии с их типом. Можно определить иерархию объектов мониторинга и организовывать совместную работу нескольких систем мониторинга для повышения надёжности и распределения нагрузки между несколькими серверами [6].

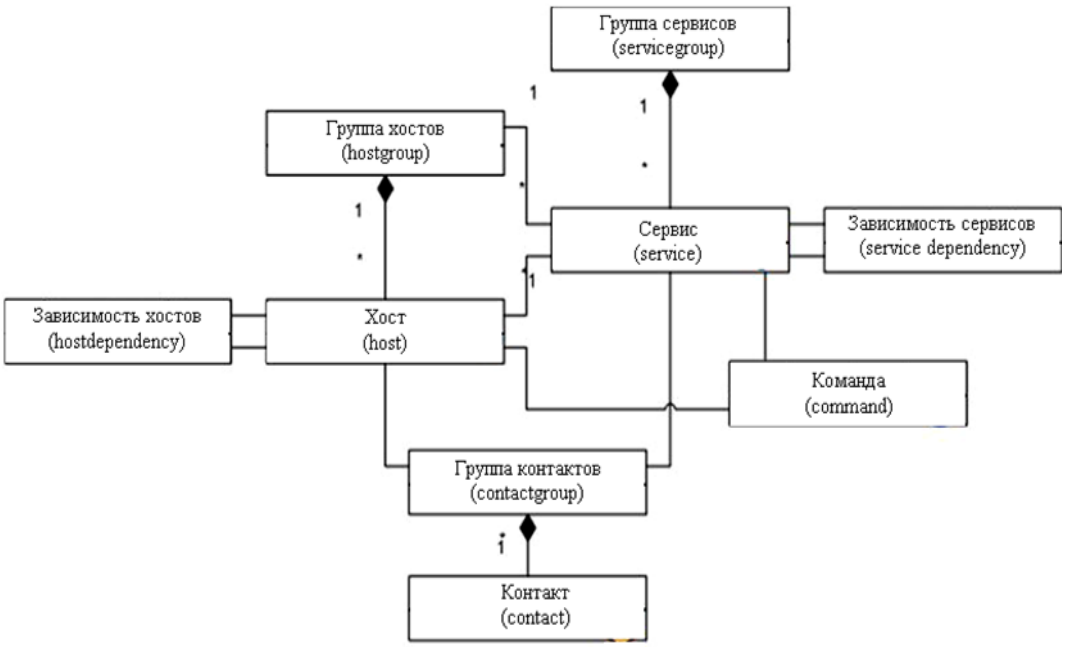

Рис. 2. Модель данных Nagios

Взаимосвязи между базовыми объектами модели данных Nagios представлены на Рис. 2. Модель содержит следующие объекты.

Хост (host) – одно из устройств, за которыми осуществляется наблюдение. Для каждого хоста определяется IP-адрес и команда, проверяющая работоспособность устройства. Для упрощения описания конфигурации подконтрольной системы хосты могут быть объединены в **Группы хостов (hostgroups).**

Сервис (service) – некоторый параметр хоста, подлежащий проверке. Определяются триггеры для следующих состояний сервиса:

OK – данные получены, величина параметра находится в пределах оптимального интервала;

Warning – данные получены, величина параметра выходит за пределы оптимального интервала;

Critical – данные получены, величина параметра приняла критическое значение, что требует оперативного вмешательства;

Unknown – данные не могут быть получены.

Сервисы могут быть объединены в **Группы сервисов (servicegroups).**

Контакт (contact) – специалист, информируемый системой о событиях (изменениях состояния объектов мониторинга). Необходимо указать адрес электронной почты, номер телефона. Может входить в состав Группы контактов (contact group).

Команда (command) – описание вызова внешней программы с указанием необходимых параметров.

Зависимости (servicedependency, hostdependency) – используются для описания иерархических зависимостей между хостами или сервисами.

Для построения на основе Nagios системы, отвечающей сформулированным требованиям, необходимо в каждом конкретном случае решить следующие задачи:

1) организовывать получение данных от разнородных объектов мониторинга (сетевые устройства, сервера, службы);

2) организовывать хранение информации о состоянии объектов и работе системы в базе данных (Nagios не работает с базами данных, фиксируя историю проверок, системные сообщения и другую информацию о своей работе в текстовых файлах);

3) создавать удобные для конечного пользователя отображения состояния различных групп устройств и сервисов; строить отчеты и графики.

Решение задачи отображения данных сильно зависит от специфики конкретной разрабатываемой системы и требований пользователей, однако для задач получения, анализа и хранения данных можно привести набор универсальных решений.

3. Методы получения данных

Системы мониторинга подразделяют на две группы: централизованные и децентрализованные. Централизованный подход подразумевает сбор и обработку данных об объектах мониторинга в одном определённом узле сети (сервере мониторинга). Децентрализованный подход предполагает распределение этих действий между несколькими узлами, например, с использованием автономных агентов (программ, работающих на объектах мониторинга независимо от сервера мониторинга, самостоятельно принимающих решения об отправке данных на сервер и реакции на изменение состояния объекта).

Наиболее универсальным методом извлечения данных об интересующих параметрах функционирования удаленных объектов является запрос по протоколу SNMP (Simple Network Management Protocol — простой протокол управления сетью на основе архитектуры UDP) [4]. Переменные (характеристики объекта), доступные через SNMP, организованы иерархически. Эти иерархии и другие метаданные (такие, как тип и описание переменной) описываются Базами Управляющей Информации (Management Information Bases (MIBs)), использующимися в качестве моделей управляемого объекта. Эти модели расширяемы, поэтому производители оборудования и программного обеспечения могут определять свои собственные элементы мониторинга для настройки удалённых объектов. Основными взаимодействующими объектами протокола являются агенты и системы управления. Агенты (устройства, для опроса состояния которых и был разработан протокол) выполняют функции сервера. Система управления осуществляет сбор информации о функционировании агентов.

Для объектов и сервисов, не поддерживающих протокол SNMP, необходимы другие средства связи с системой мониторинга – агенты или иные программные компоненты, передающие серверу мониторинга информацию о состоянии объекта. Для Nagios стандартным средством такого мониторинга считается NRPE (Nagios Remote Plugin Executor). NRPE со стороны объекта мониторинга (клиентская часть) – это процесс, который ожидает запросы от сервера мониторинга, запускает объявленный в файле конфигурации программный модуль и отсылает полученные данные серверу.

Другая концепция получения данных от объектов – это так называемые пассивные проверки (Passive Checks), при которых инициатива проверки принадлежит не ядру мониторинга, а самому программному модулю, который непрерывно работает и посылает данные на сервер мониторинга только при выполнении некоторых условий (например, при обнаружении нештатной ситуации). Для систем на базе Nagios данная возможность реализована в модуле NSCA (Nagios Service Check Acceptor), основной процесс которого работает на сервере мониторинга и принимает результаты пассивных проверок.

4. Хранение данных

Для успешного решения задач поиска закономерностей в собранных данных, их статистического анализа, выполнения отчетов о работе системы необходимо организовать хранение результатов проверок в базе данных (БД). Рассмотрим схему реализации подсистемы хранения данных, взаимодействующей с системой управления базами данных СУБД MySQL, PostgreSQL и другими с использованием библиотеки NDO (Nagios Database Output) [10]. Основные компоненты этой библиотеки:

1) событийный брокер NDOMOD для экспорта данных главного процесса Nagios в текстовый файл по протоколу TCP/IP;

2) утилита LOG2NDO для импорта информации из лог-файлов Nagios;

3) утилита NDO2DB для записи в БД.

NDOMOD и LOG2NDO выполняют функции клиентов, поставляющих данные серверному процессу NDO2DB. NDO допускает параллельную работу нескольких клиентов, собирающих данные систем Nagios на разных серверах для централизованного хранения в одной БД. В этом случае запускается по одному

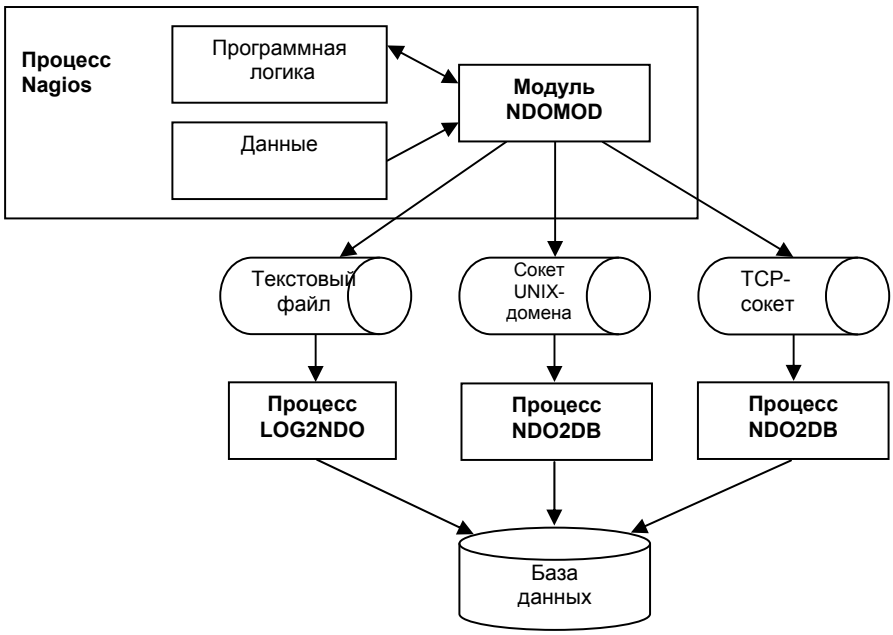

Рис. 3. Способы экспорта данных Nagios в базу данных посредством NDO2DB

процессу NDO2DB для каждого клиентского соединения. Схема информационных потоков между Nagios и БД с участием компонентов NDO2DB приведена на Рис. 3.

Стандартная структура NDO содержит 59 таблиц, агрегирующих информацию обо всех объектах модели данных Nagios. Этот набор целесообразно расширить в соответствии с требованиями к системе – добавить таблицы для хранения данных о пользователях системы, соответствующих контактам Nagios, и их правах. Также необходимо хранить ассоциированные с сервисами шаблоны для построения регулярных выражений [11], используемых при анализе данных.

Ещё одна важная задача – организация резервного копирования базы данных, архивирования неактуальных данных для предотвращения снижения эффективности работы БД. Так, для ЦИВК ОИЯИ зафиксирована генерация системой мониторинга до 1 Гб данных в день при выделенном дисковом пространстве 62 Гб. Простейший способ решения этой проблемы – периодическое (раз в неделю либо чаще) резервное копирование базы данных, сжатие (архивация) копии и последующая очистка базы. При необходимости анализа данных за некоторый период времени нужная копия может быть восстановлена и временно объединена с текущей базой, а затем осуществлен возврат БД к состоянию на начало анализа.

5. Построение системы мониторинга ЦИВК

Центральный информационно-вычислительный комплекс ОИЯИ является крупнейшим в России комплексом для моделирования, хранения, обработки и анализа данных с экспериментов на Большом адронном коллайдере (LHC). В 2011 году на вычислительном комплексе ЦИВК ОИЯИ было выполнено около 6 миллионов задач пользователей, предоставлено около 27 миллионов часов процессорного времени и около 1 Петабайта дискового хранилища. Этот комплекс активно используется для крупномасштабных вычислений многими научными коллективами ОИЯИ, России и других стран [12]. Системы хранения данных вычислительного комплекса широко используются для поддержки проектов ОИЯИ и международных коллабораций, в том числе проекта NICA/MPD. Ресурсы ЦИВК ОИЯИ по состоянию на март 2012 года включают в себя 2072 вычислительных узлов, более 80 управляющих серверов. Базовой операционной системой, под управлением которой работают серверы является Scientific Linux (SL5). К важнейшим ресурсам

комплекса относятся две системы хранения данных dCache [13], включающие 12 серверов - основных интерфейсов системы - и 32 пула (системы хранения данных), а также три системы XROOTD [14], включающие сервер обработки запросов к системе и 12 пулов. Для обеспечения этих систем предоставляется около 1000 TB (1 PB) дискового пространства, организованного в RAID-массивы. Локальная сеть ЦИВК построена на базе агрегированных GigabitEthernet-соединений (транков), коммутаторов и маршрутизаторов HP Procurve и Cisco Catalyst.

Анализ и решение задач оперативного и фонового мониторинга отдельных групп устройств и сервисов требуют рассмотрения объектов мониторинга всех трёх уровней (аппаратного, сетевого и уровня служб), определения и разработки необходимых алгоритмов и программных компонентов.

6. Аппаратный уровень

Мониторинг источников бесперебойного питания (ИБП) включает слежение за значениями таких параметров, как ёмкость аккумулятора, внешняя и внутренняя температура, сила тока и напряжение, нагрузка и состояние батареи. В случае выхода значения некоторого параметра из допустимого диапазона необходимо оповестить осуществляющего контроль администратора. Необходимо также хранить историю изменений этих значений. Все необходимые для этого данные могут быть получены по протоколу SNMP. Идентификаторы переменных, хранящих значения необходимых величин, можно найти в спецификации базы управляющей информации (MIB), предоставляемой фирмой-производителем ИБП (компания APC). Для получения значений переменных используется стандартный модуль Nagios check_snmp. Необходимо сконфигурировать этот модуль, задав необходимые параметры: IP-адрес устройства, пароль для чтения значений переменных, предельно допустимые значения параметра для состояний Nagios OK и Warning.

Другая группа устройств, для мониторинга которых достаточно запросов по протоколу SNMP – это вентиляционные блоки серверных стоек. Для них анализируются значения таких параметров, как показания температурных датчиков, флаги состояния, скорость вращения вентиляторов и объём генерируемого ими воздуха (в час). Эффективный мониторинг этих величин может предотвратить сбои в работе оборудования вследствие неполадок в работе систем охлаждения.

Для эффективного мониторинга серверов протокола SNMP недостаточно. Система мониторинга должна получать информацию о таких параметрах, как загрузка процессора оперативной памяти, свободном дисковом пространстве на доступных разделах, а для отдельных серверов – специфические данные по работающим на них процессам (сервисам). Контроллеры дисковых массивов RAID могут служить примером серверных объектов, которые не предоставляют необходимую информацию по протоколу SNMP. Ситуация осложняется тем, что не существует единообразного для всех RAID-контроллеров средства контроля и управления. Большинство фирм-поставщиков RAID-контроллеров предлагают своё программное обеспечение для мониторинга и управления массивами дисков (3ware – утилита tw_cli; HighPoint – hptraidconf; Adaptec – arcconf). Был разработан и реализован универсальный алгоритм проверки RAID-контроллеров, который состоит из следующих этапов:

- проверка наличия и доступности на данном компьютере raid-контроллеров указанного производителя;
- получение списка идентификаторов контроллеров;
- получение для каждого из полученных контроллеров общей информации о текущем состоянии, списка активных портов, информации о подключенном диске и его состоянии, используя предоставляемую производителем утилиту;
- определение на основании полученных данных результата проверки (корректное состояние; состояние, требующее предупреждения администратора; нештатное состояние),
- сформирование отчета о проверке.

Другие задачи, связанные с мониторингом серверов (состояние процессора, памяти, дисков, работающих процессов), можно решить с помощью адаптации стандартных модулей Nagios (check_load, check_disk, check_procs).

7. Сетевой уровень

Для корректной работы локальной сети необходимо постоянное наблюдение и выявление неэффективно работающих или неисправных объектов. Структурная схема сети ЦИВК ОИЯИ приведена на Рис. 4. К основным задачам мониторинга ЦИВК на данном уровне относятся наблюдение за состоянием коммутаторов HP Procurve и Cisco, агрегированных соединений (транков), а также внешнего канала. Транк – это способ распараллеливания потока информации между двумя сетевыми устройствами путём прокладки двух или более каналов связи, повышающий пропускную способность соединения. Набор соединенных портов выглядит как один логический порт, но с соответственно увеличенной пропускной способностью. Если один из портов в агрегированном канале выходит из строя, агрегированный канал работает. Он остается работоспособным до тех пор, пока активен хотя бы один порт. Важнейшей задачей является грамотная балансировка трафика, для чего необходим сравнительный анализ графиков загрузки отдельных портов. Для оптимизации работы сети и устранения возможных узких мест необходимо проводить накопление и анализ статистических данных об ошибках разных типов, динамике загрузки процессоров коммутаторов, динамике передачи данных на отдельных портах.

Мониторинг всех объектов сетевого уровня может быть осуществлен с помощью SNMP с привлечением некоторых дополнительных средств анализа на серверной стороне. Так, для эффективного наблюдения за состоянием портов и каналов требуется построение графиков загрузки, обновляющихся в реальном времени. Для этого была применен набор утилит MRTG (Multi Router Traffic Grapher). Его основной процесс должен считывать информацию о переменных, графики которых строятся из указанного в качестве параметра конфигурационного файла и в соответствии с этой информацией получать новые значения переменных.

Данная схема использована не только для отображения состояния объектов сетевого уровня, но и для визуализации информации общего вида (устройства, сервисы, другие объекты).

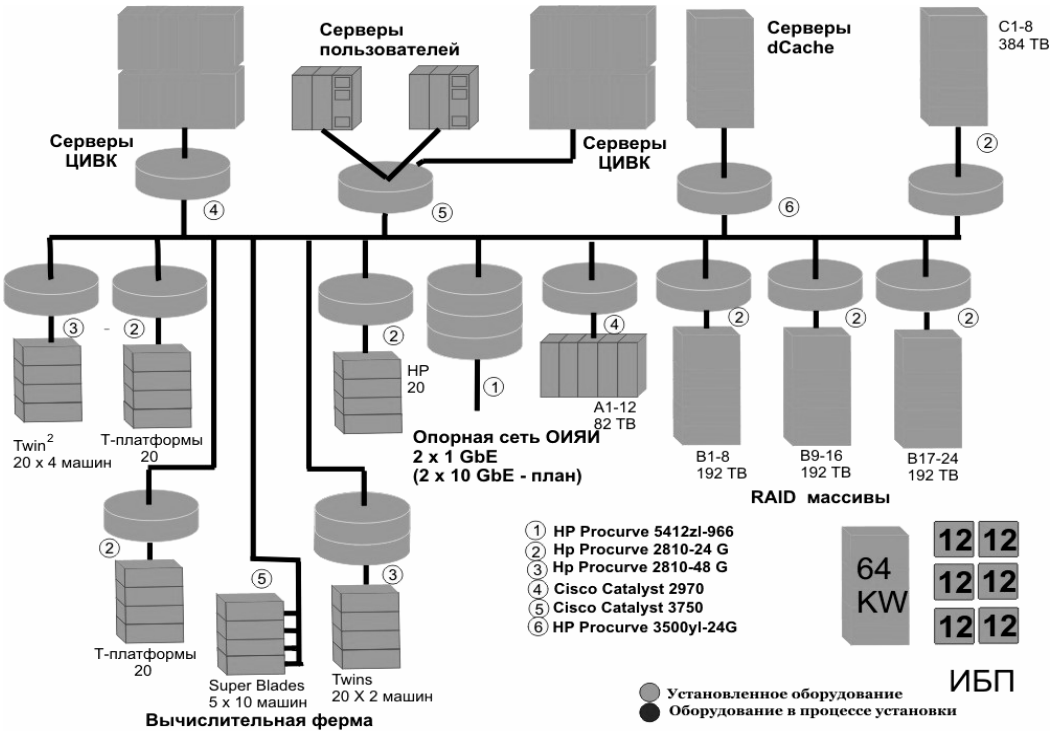

Рис. 4. Структура локальной сети ЦИВК ОИЯИ

8. Уровень служб

Для массового хранения экспериментальных данных в ЦИВК ОИЯИ используется система dCache, которая предоставляет (в том числе средствами грида) доступ к каталогам и файлам, логически объединённым в общую структуру, но расположенным на разных дисках. Наиболее актуальной является задача учета и анализа эффективности распределения ресурсов дисковой памяти. Система dCache не содержит встроенных средств анализа эффективности собственной работы, достаточных для принятия решений по её программной или аппаратной реконфигурации. Набор параметров, используемых стандартной для dCache системой анализа заданий, недостаточен для задач локального администрирования системы [15]. Для мониторинга были выбраны следующие дополнительные параметры:

- объем и эффективность использования памяти;
- количество и объём файлов, которые были однажды записаны и с тех пор ни разу не читались;
- количество одновременно выполняемых процессов передачи данных;
- уровень ошибок – статистика успешных/неуспешных обращений пользователей к системе;
- уровень трафика и уровень нагрузки на основные системные процессы.

Эти данные собираются при помощи разработанных модулей, запускаемых на сервере dCache посредством NRPE.

Помимо мониторинга dCache к уровню служб относится анализ состояния RAID-массивов, обеспечивающих системы хранения. Для каждого из серверов собирается информация о текущем статусе, числе проверенных дисков, дисковых массивов и контроллеров. Если обнаруживается проблема, то указывается ее возможный источник. Существует два основных класса таких проблем (некорректных состояний):

- проблемы, непосредственно относящиеся к raid-массивам, например: *<контроллер> is DEGRADED-VRFY* (один из дисковых приводов вышел из строя, производится его проверка и восстановление рабочего состояния);
- внешние проблемы (недоступность того или иного сервера в сети) – например: `CHECK_NRPE: Socket timeout after <N> seconds` (клиентская часть NRPE на требуемой машине не вернула ответ в установленный срок).

К прикладному уровню относится также мониторинг таких сервисов, как SMTP, DNS, HTTP и другие. Он осуществляется с помощью стандартных модулей check_http, check_dns, check_smtp.

9. Представление данных

Веб-интерфейс системы мониторинга реализован на языке программирования PHP с использованием XHTML и Javascript для разметки страниц и представления данных. Для автоматического обновления графиков в реальном времени использована технология AJAX (jQuery) [16]. В основном режиме работы пользователю доступно дерево объектов, где на верхнем уровне находятся группы хостов, а «листьями» дерева являются сами хосты и отдельные сервисы (Рис.5). Для каждого хоста динамически генерируется страница с данными по всем определенным для него сервисам – текущему состоянию, графику значений за прошедшие сутки и другим вспомогательным данным. Нажатием специальной кнопки можно вызвать всплывающее окно с отчетом по выбранному сервису. Следует отметить большой объём предоставляемой системой информации: так, по каждому из 12 коммутаторов приводятся несколько десятков графиков загрузки его портов; доступны подробные данные о состоянии 400 серверов. Помимо получения общего представления о работе и текущем состоянии вычислительного комплекса (включая актуальные проблемы) эти данные могут быть использованы для исследования закономерностей и взаимосвязей между его отдельными компонентами с целью оптимизации и повышения эффективности работы комплекса.

Создано несколько специальных отображений, позволяющих более наглядно оценить состояние той или иной группы сервисов и проследить взаимосвязь между объектами. Для этой цели используется пакет NagVis (дополнение к Nagios), который обладает широкими

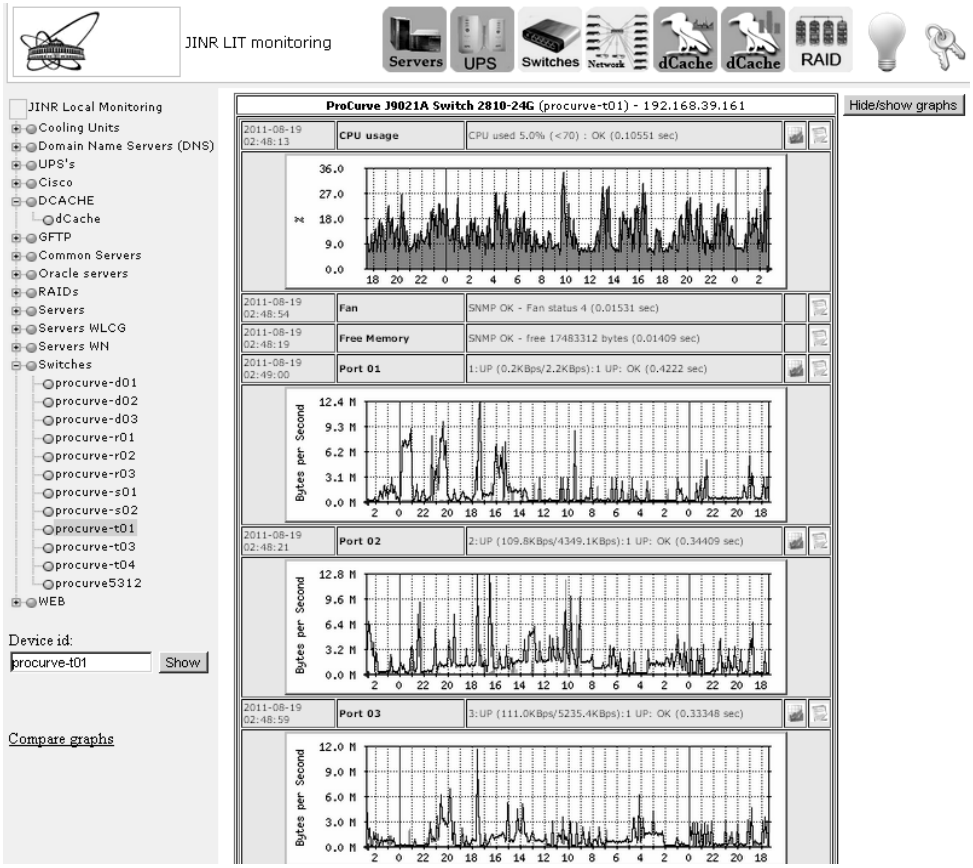

Рис. 5. Web-интерфейс системы

Слева – дерево объектов, справа – фрагмент отчёта о состоянии коммутатора HP Procurve

возможностями для визуализации состояния хостов, сервисов, объектов, IT-процессов. С помощью пакета NagVis создано отображение структурной схемы сети с маркерами, которые характеризуют состояние соответствующих объектов.

Для группы сервисов, относящихся к dCache, создано отображение, позволяющее сравнить наблюдаемые значения проверяемых параметров для экспериментов Atlas и CMS. Создана отчётная таблица по дисковым массивам с указанием состояния каждого контроллера и сводки данных по серверам (с распределением их по стойкам, ИБП, администраторским группам).

В случае возникновения нештатной ситуации (либо выхода системы из неё) система рассылает сотрудникам, ответственным за проблемные службы и устройства, соответствующие оповещения посредством электронной почты или SMS.

Заключение

В работе определены основные понятия и функциональные элементы программной системы локального мониторинга вычислительных ресурсов, требования к которой включают универсальность и расширяемость. Обоснована целесообразность построения подобных систем на базе архитектуры «ядро – подключаемые модули». Описан реализующий такую архитектуру свободно распространяемый программный продукт Nagios и разработанная на основании указанных положений система локального мониторинга ЦИВК ОИЯИ (http://litmon.jinr.ru). Предложены методы мониторинга обеспечивающих функционирование комплекса устройств и сервисов, распределённых по трём уровням – аппаратному, сетевому и уровню служб. Работа представляется важным вкладом в теорию разработки систем мониторинга, являющихся основным инструментом контроля

за функционированием сложных РВС и позволяющих не только своевременно реагировать на возникающие сбои, но и прогнозировать возможные нештатные ситуации и принимать меры по их предотвращению.

Литература

1. Фиаммант В., Боуз С., Биберштейн Н. Компас в мире сервис-ориентированной архитектуры (SAO): КУДИЦ-Пресс, 2007 г.
2. Коптелов А., Голубев В. Сервис-ориентированная архитектура: от концепции к применению // BYTE: Москва: СК Пресс, №6, 2008.С.32-35
3. Мицын С. Визуализация мониторинга грид-инфраструктуры WLCG/EGEE как географически распределённой системы // Научный отчет Лаборатории информационных технологий ОИЯИ 2009-2010: Дубна: ОИЯИ, 2010. С.37-40.
4. Фейт С. TCP/IP. Архитектура, протоколы, реализация (включая IPv6 и IP Security): Лори, 2009
5. Кинг Д. Создание эффективного программного обеспечения: Пер. с англ.- М.: Мир, 1991
6. Barth W. Nagios System and Network Monitoring: No Starch Press, 2006 г.
7. Jung P. Munin - the Raven Reports // Linux Journal №180, April 2009. С.25-30.
8. Comparison of network monitoring systems [Электронный ресурс]: URL: http://en.wikipedia.org/wiki/Comparison_of_network_monitoring_systems
9. Kamarudzzaman K.A., Rusalan N. Comparison report on network monitoring systems (Nagios and Zabbix) [Электронный ресурс]: Malaysian public sector open source software programme. URL: http://knowledge.oscc.org.my/practice-areas/rnd/benchmark-report/comparison-report-on-network-monitoring-system/at_download/file
10. Galstad E. NDOUTILS Documentation Version 1.4 [Электронный ресурс]: Etien Galstad, NDOUtils Documentation. URL: http://nagios.sourceforge.net/docs/ndoutils/NDOUtils.pdf
11. Фридл, Дж. Регулярные выражения: СПб.: «Питер», 2001. — 352 с.
12. Астахов Н.С., Долбилов А.Г., Иванов В.В., Кореньков В.В., Мицын В.В., Трофимов В.В. Развитие Центрального информационно-вычислительного комплекса ОИЯИ в 2010-2011 году и текущее состояние программно-аппаратной среды // Научный отчет 2010-2011 Лаборатории информационных технологий, ISBN 978-5-9530-0312-4., 2012, С.16-20.
13. Fuhrmann P. dCache, the Overview [Электронный ресурс]: dCache.org, Patrick Fuhrmann: dCache, the Overview. URL: [http://www.dcache.org/manuals/dcache-whitepaper-light.pdf]
14. Hanushevsky A. Xrootd Architectures [Электронный ресурс]: Xrootd Project. URL: [http://xrootd.slac.stanford.edu/presentations/OSGAHM_1103.Plenary.pptx]
15. Трофимов В., Дмитриенко П. Об одном подходе к мониторингу и последующей оптимизации элемента памяти грид-структуры, реализованного на основе системы dCache // Научный отчет Лаборатории информационных технологий ОИЯИ 2008-2009: Дубна: ОИЯИ, 2009. С.41-43.
16. Крейн Д., Бибо Б., Сонневелдьд Д. Ajax на практике: М.: «Вильямс», 2007.

Кореньков Владимир Васильевич. Заместитель директора Лаборатории информационных технологий ОИЯИ. Окончил Московский государственный университет в 1976 году. Кандидат физико-математических наук, старший научный сотрудник. Автор более 250 печатных работ. Область научных интересов: распределенные и параллельные вычисления, грид-технологии, сети, базы данных и распределенные системы хранения сверхбольших объемов информации, корпоративные информационные системы. E-mail: korenkov@cv.jinr.ru

Мицын Валерий Валентинович. Старший научный сотрудник Лаборатории информационных технологий ОИЯИ. Окончил Московский государственный университет в 1975 году. Автор 15 печатных работ. Область научных интересов: распределенные вычисления (грид), сети, распределенное хранение и доступ к сверхбольшим объемам информации, системное администрирование больших вычислительных установок и хранилищ данных. E-mail: vvm@cv.jinr.ru

Дмитриенко Павел Владимирович. Инженер-программист Лаборатории информационных технологий ОИЯИ. Аспирант. Окончил Орловский государственный технический университет в 2008 году. Автор 11 печатных работ. Область научных интересов: распределенные и параллельные вычисления, грид-технологии, сети, проектирование высоконагруженных интернет-проектов. E-mail: orelnotre@mail.ru