Московский Государственный Технический Университет им. Н.Э.Баумана

Отчет по мониторингу состояния оборудования

Выполнили студенты:

Бойнов М.А.

Корнуков Н.

Проверил:

Тарнакин А.О.

Москва, 2017

# Предметная область

Мониторинг состояния оборудования

# Назначение, цели, задачи системы

Структура СОИУ:

Клиент-Посредник-Сервер-Веб-приложение

Агент-Logstash-ElasticSearch-Веб-страница

Этапы работы системы:

1. Сбор данных агентом
2. Передача данных посреднику
3. Обработка данных
4. Передача данных серверу обработки
5. Построение статистики
6. Передача статистики на веб-интерфейс
7. Вывод статистики конечному пользователю

# Требования к функциональным характеристикам

Мониторинг заключается в следующем: программа-агент находится на клиентской машине и собирает следующую информацию:

* Имя ПК
* IP-адрес
* Загрузка ЦП
* Загрузка ОЗУ

Формат данных: любой текстовый

Сбор данных производится любым способом на усмотрение разработчиков

Процесс передачи и обработки информации:

1. Клиент передает собранную информацию по сетевому протоколу (TCP или UDP) к посреднику.
2. Посредник производит преобразование полученной информации и отправляет готовый json-документ на сервер обработки данных
3. Полученные данные поступают на сервер обработки данных и индексируются
4. На основе проиндексированных данных создается статистика. Статистика создается отдельно для каждой клиентской машины.
5. Затем происходит передача статистики в веб-приложение с использованием REST API.
6. Статистика отображается конечному пользователю на веб-странице

# Требования к составу и параметрам технических средств

Реализовать СОИУ с помощью следующих технологий:

Агент – Java

Посредник – Logstash

Сервер обработки данных – ElasticSearch

Веб-интерфейс – Kibana или что угодно

Платформа – Linux или Windows

Хостинг – на усмотрение разработчиков (web или локальный)

# Требования к программной документации

К защите подготавливается отчет по лабораторной работе в свободной форме

# Порядок контроля и приемки

Сроки: начало мая

Связь с преподавателем: [alexei.tarnakin@gmail.com](mailto:alexei.tarnakin@gmail.com)

# Агент

Агент реализован на языке java. Он состоит из двух классов.

SystemInfo.java - простой класс который собирает нужные данные: Имя ПК, IP-адрес, загрузка ЦП, Загрузка ОЗУ.

|  |
| --- |
| **import** **com.sun.management.OperatingSystemMXBean**;  **import** **javax.json.\***;  **import** **java.lang.management.ManagementFactory**;  **import** **java.net.InetAddress**;  **import** **java.net.UnknownHostException**;  **public** **class** **SystemInfo** {  OperatingSystemMXBean osBean;  **public** **SystemInfo**() {  osBean = ManagementFactory.getPlatformMXBean(OperatingSystemMXBean.class);  }  **public** **double** **cpu**() {  **return** osBean.getSystemCpuLoad();  }  **public** **double** **memory**() {  **return** (**double**) osBean.getFreePhysicalMemorySize() /  osBean.getTotalPhysicalMemorySize();  }  **public** String **computerName**() {  **try** {  **return** InetAddress.getLocalHost().getHostName();  } **catch** (UnknownHostException ex) {  **return** "unknown";  }  }  **public** String **ip**() {  **try** {  **return** InetAddress.getLocalHost().getHostAddress();  } **catch** (UnknownHostException ex) {  **return** "unknown";  }  }  **public** String **toString**() {  **return** Json.createObjectBuilder()  .add("cpu", cpu())  .add("memory", memory())  .add("computer", computerName())  .add("ip", ip())  .build()  .toString();  }  } |

TCPServer.java - простой сервер. При соединении клиента, он начинает отправлять ему каждую секунду имя ПК, IP-адрес, загрузку ЦП и загрузку ОЗУ, в формате JSON.

|  |
| --- |
| **import** **java.io.DataOutputStream**;  **import** **java.net.ServerSocket**;  **import** **java.net.Socket**;  **import** **java.net.SocketException**;  **class** **TCPServer**  {  **static** **int** PORT = **6879**;  **public** **static** **void** **main**(String argv[]) **throws** Exception  {  ServerSocket socket = **new** ServerSocket(PORT);  **while** (**true**) {  System.err.println("Waiting for connections on port " + PORT);  Socket conn = socket.accept();  System.err.println("New connection");  DataOutputStream out = **new** DataOutputStream(conn.getOutputStream());  **boolean** isOpen = **true**;  SystemInfo info = **new** SystemInfo();  String response;  **while** (isOpen) {  **try** {  out.writeBytes(info.toString() + "\n");  } **catch**(SocketException e) {  System.err.println(e);  isOpen = **false**;  }  Thread.sleep(**1000**);  }  System.err.println("Connection closed");  }  }  } |

В процессе компиляции получается файл с расширением jar. Для запуска агента нужно выполнить следующие действия:

# Посредник

Посредник - logstash. Перед запуском нужно настроить свою конфигурацию в файле tcp2elastic.conf. При запуске, он соединяется с агентом и со сервером. Потом, каждую секунду, он получает данные от агента, добавляет в них timestamp и другую информацию, и отправляет их на сервер. Файл конфигурации:

|  |
| --- |
| input {  tcp {  host => "192.168.1.19"  port => **6879**  mode => "client"  codec => "json"  }  }  output {  stdout { codec => rubydebug }  elasticsearch {  hosts => ["localhost:9200"]  user => "elastic"  password => "changeme"  index => "monitor"  }  } |

Для запуска нужно перейти в папку bin и ввести команду:

logstash -f tcp2elastic.conf

# Сервер хранения и обработки данных

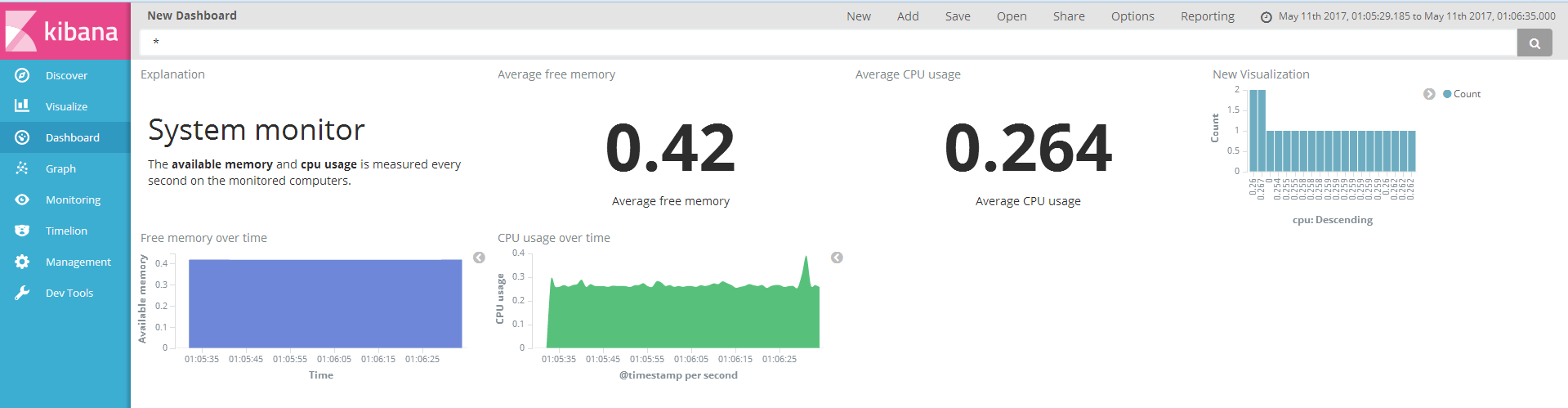
Сервер - ElasticSearch. Он сохраняет данные, создает индексы, и отвечает на запросы интерфейса через протокол HTTP на формате JSON.

Для запуска нужно запустить elasticsearch.bat

# Интерфейс

Интерфейс - Kibana. В нём построено несколько полезных визуализации наших данных.

Для запуска нужно запустить kibana.bat



# Архитектура

