



Monitoring

на примере Zabbix

Tinkoff.ru

Роман Николаев

О чем пойдет речь?



1. Что такое мониторинг?
2. Существующие системы мониторинга
3. Архитектура Zabbix
4. Zabbix Items
5. User Parameters
6. Zabbix Triggers
7. Zabbix Actions
8. Low-level Discovery
9. Zabbix + Grafana

1

Что такое мониторинг?



Основные задачи:

- Оперативный мониторинг (обнаружение и диагностика проблем)
- Анализ исторических данных («разбор полетов», планирование ресурсов, ...)

Типы метрик:

- Системные метрики:
CPU, RAM, Disk Space, I/O operations, ...
- Метрики приложения/сервиса:
наличие HTTP ошибок, доступность БД, скорость выполнения метода...
- Бизнес-метрики:
кол-во активных клиентов, суммарные показатели выручки, частота покупки товара, ...

2

Существующие системы мониторинга



Существующие системы мониторинга

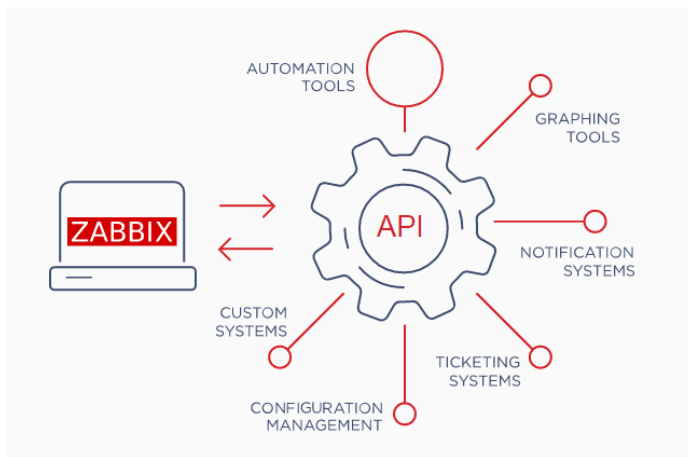


Nagios

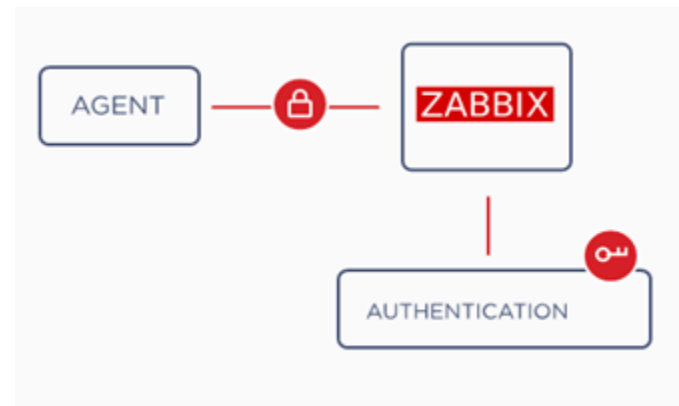
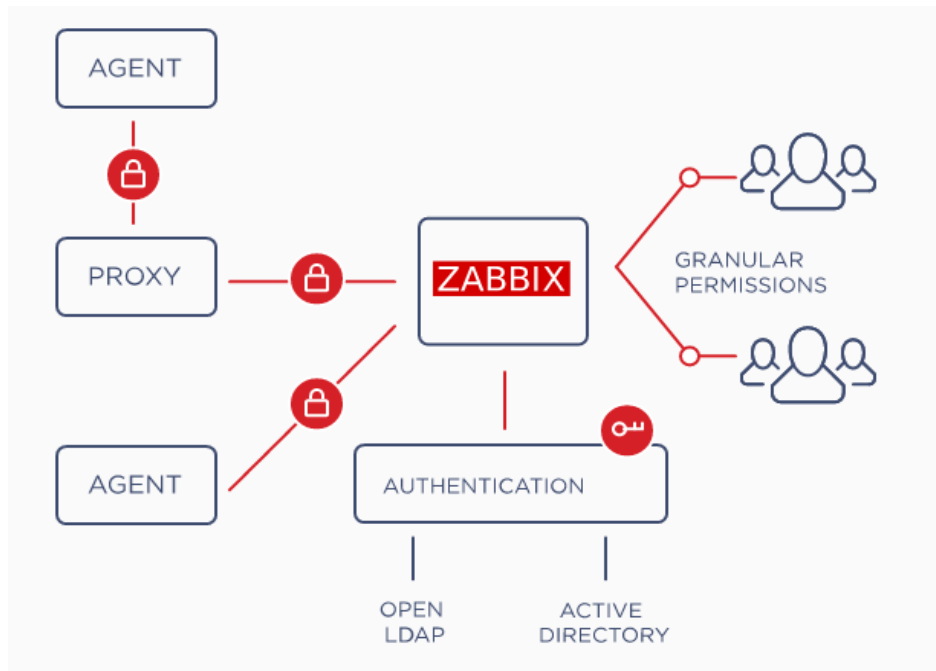


3

Архитектура Zabbix



Архитектура Zabbix



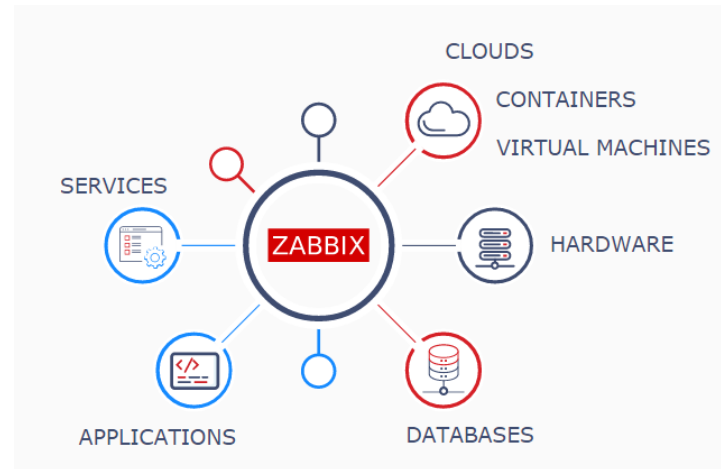
Ключевые особенности



- Frontend: веб-интерфейс на php
- Backend: реляционная БД (MySQL / PostgreSQL)
- Сбор метрик по моделям pull/push
- Добавление новых метрик через веб-интерфейс или API
- Возможность построить распределенную систему мониторинга
- Масштабирование системы возможно с помощью Zabbix Proxy
- Отказоустойчивость системы строится за счет возможностей соответствующей подсистемы (Linux HA-Cluster, Virtual IP, DB Replication)

4

Zabbix Items



Zabbix Items (элементы данных)



Сущность, которая собирает данные (метрика).

Типы:

- Zabbix агент
- SNMP агент
- SNMP трапы
- IPMI проверки
- Простые проверки
- Журнал (лог)
- Вычисляемые элементы данных
- Внутренние проверки Zabbix
- SSH проверки
- Telnet проверки
- Внешние проверки (скрипты)
- Агрегированные проверки
- Траппер
- JMX
- ODBC
- Зависимые элементы данных

Элементы данных

Все шаблоны / Template OS Linux Группы элементов данных 10 Элементы данных 32 Триггеры 17 Графики 5 Комплексные экраны 1 Правила обнаружения 2

Элемент данных Предобработка

Имя CPU \$2 time

Тип Zabbix агент

Ключ system.cpu.util[,iowait] [Выбрать](#)

Тип информации Числовой (с плавающей точкой)

Единица измерения %

Интервал обновления 1m

Пользовательские интервалы

Тип	Интервал	Период	Действие
Переменный	По расписанию	50s	1-7,00:00-24:00

[Добавить](#) [Удалить](#)

Период хранения истории 1w

Период хранения динамики изменений 365d

Отображение значения Как есть [показать преобразования значений](#)

Новая группа элементов данных

Группы элементов данных

- Her-
- CPU
- Filesystems
- General
- Memory
- Network interfaces
- OS
- Performance
- Processes
- Security

Заполнение поля инвентаря узла сети -Her-

Описание Amount of time the CPU has been waiting for I/O to complete.

Zabbix Items (элементы данных)



Zabbix агент — для сбора данных используется Zabbix агент.

SNMP агент — опрос хостов по SNMP (протокол UDP, порт 161): сетевуха, сервера, принтеры...

SNMP трапы — «ловушка» для сообщений от SNMP-устройств (протокол UDP, порт 162).

IPMI проверки — Intelligent Platform Management Interface, позволяет наблюдать за состоянием аппаратного обеспечения напрямую с так называемых карт управления “out-of-band”, независимо от ОС или же от наличия питания на хосте (HP iLO, DELL DRAC, IBM RSA...).

Простые проверки — не требуется Zabbix агент, удаленные проверки выполняются с Zabbix сервера/прокси (доступность порта, ping).

Журнал (лог) — мониторинг логов а предмет появления ключевой фразы (точная строка / регулярка). Требуется Zabbix агент.

Вычисляемые элементы данных — подсчеты на основании других элементов данных (виртуальный источник данных).

Внутренние проверки Zabbix — наблюдение за внутренним процессами Zabbix (queue, items, hosts, requiredperformance, ...)

SSH проверки — результат выполнения произвольной команды/скрипта по ssh (логин + пароль / файл ключа), Zabbix агент **не** нужен.

Telnet проверки — результат выполнения произвольной команды/скрипта по telnet (логин + пароль), Zabbix агент **не** нужен.

Внешние проверки (скрипты) — выполнение произвольного скрипта (с параметрами) с Zabbix сервера/прокси, Zabbix агент **не** нужен.

Агрегированные проверки — совокупная инфа по элементам данных на основе прямых запросов в БД («температура по больнице»).

Траппер — отправка данных (метод “push”) с узла вместо запроса данных (метод “pull”). Например, с помощью [zabbix_sender](#).

JMX — получение данных со счетчиков JMX ([Java Management Extensions](#)) из Java-приложений.

ODBC — запросы в БД напрямую с Zabbix сервера с помощью [ODBC](#)-драйвера (прослойка для работы с СУБД).

Зависимые элементы данных — основной элемент данных инициирует сбор зависимого элемента (когда нужна синхронность сбора).

5

User Parameters

{code}

User Parameters (пользовательские параметры)



Пользовательские параметры – возможность выполнять любые команды непосредственно на целевом узле, получая тем самым необходимые данные по метрикам.

Синтаксис:

UserParameter=<ключ>, <команда>

Пример:

UserParameter=mysql.ping,mysqladmin -uroot ping | grep -c alive

Агент будет возвращать '1', если MySQL сервер доступен, '0' - в противном случае.

Гибкие пользовательские параметры – допускают параметры с указанным ключом. В этом случае гибкие пользовательские параметры могут быть основой для создания нескольких элементов данных.

Синтаксис:

UserParameter=ключ[*], команда

Пример:

UserParameter=wc[*],grep -c "\$2" \$1

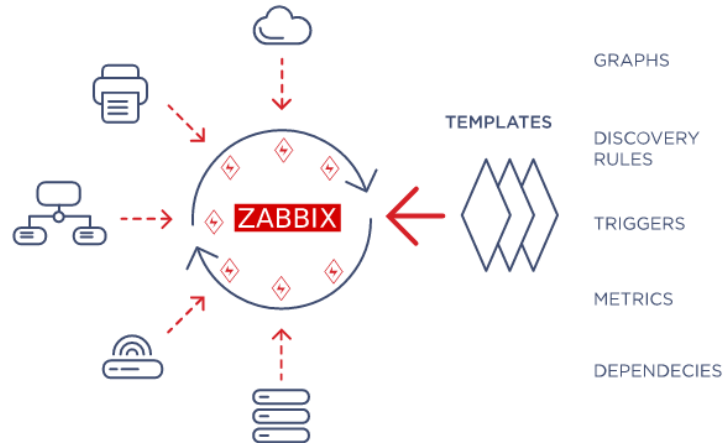
Этот параметр можно использовать для подсчета количества строк в файле.

wc [/etc/passwd, root]

wc [/etc/services, zabbix]

6

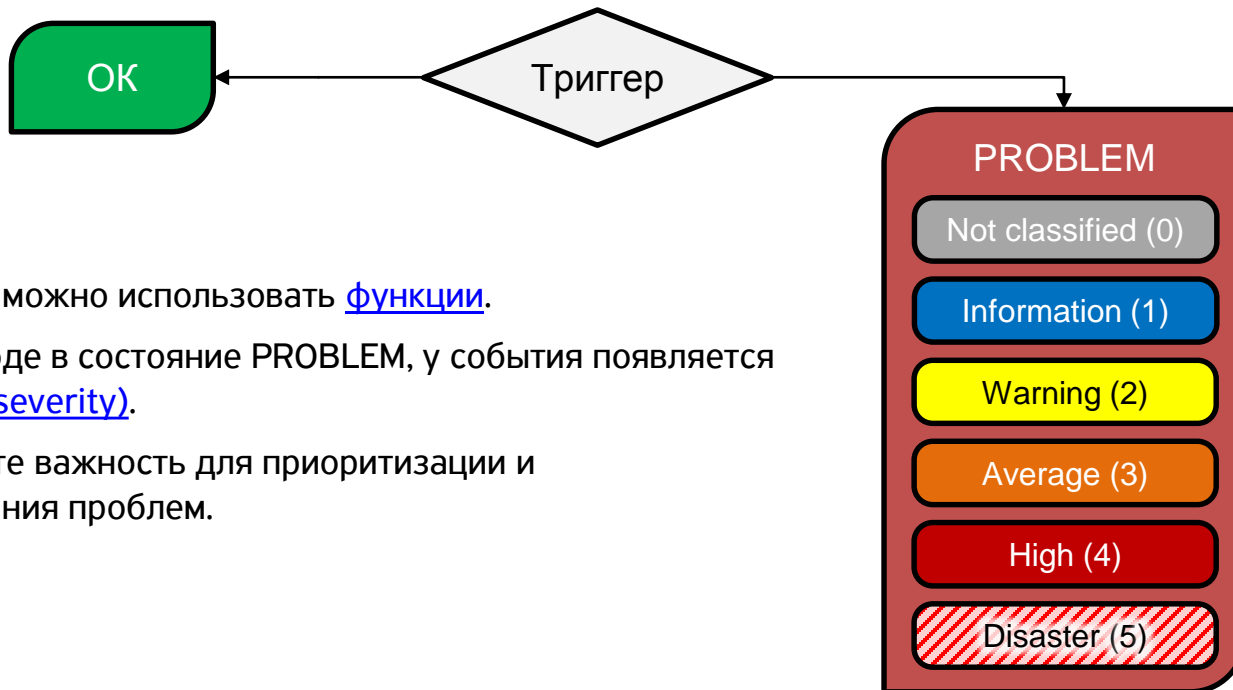
Zabbix Triggers



Zabbix Triggers (триггеры)



Триггер – логическое выражение, которое определяет какую-либо ситуацию/состояние.



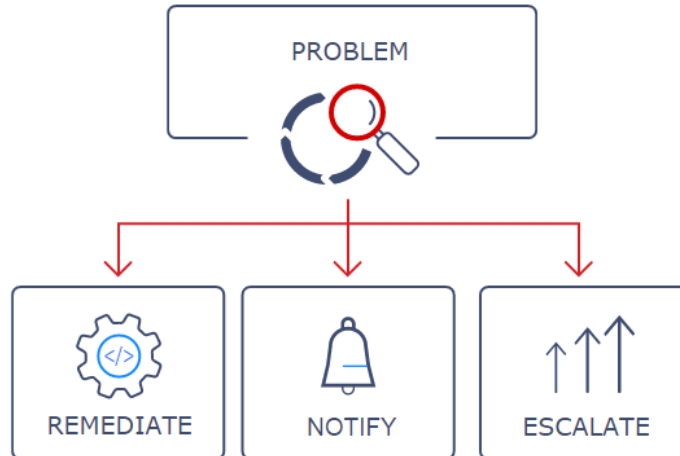
В триггере можно использовать функции.

При переходе в состояние PROBLEM, у события появляется важность (severity).

Используйте важность для приоритизации и эскалирования проблем.

7

Zabbix Actions



Zabbix Actions (действия)



Action (действия) – это реакция на какое-либо событие.

Triggers (Триггеры)

В основном используется для оповещения по сработавшим триггерам (alerts).

Auto Registration (Авторегистрация)

Удобная возможность автоматически добавлять узлы в мониторинг.

Network Discovery (Обнаружение)

Сканирование сети в указанном диапазоне и добавление найденных узлов в мониторинг.

Internal events (Внутренние события)

Возникают, когда:

- элемент данных меняет свое состояние с “нормального” на “не поддерживается” или обратно
- правило низкоуровневого обнаружения меняет свое состояние с “нормального” на “не поддерживается” или обратно
- триггер меняет свое состояние с “нормального” на “неизвестное” и обратно

Действия

Действие Операции Операции восстановления Операции подтверждения

Имя PostgreSQL Servers (Production): оповестить о любых проблемах

Тип вычисления И A and (B and C)

Условия	Подпись	Имя	Действие
A	Важность триггеров >= Warning (1)		Удалить
B	Группа узлов сети = Production hosts		Удалить
C	Группа узлов сети = PostgreSQL		Удалить

Новое условие

Имя триггера содержит

[Добавить](#)

Активировано ☒

[Обновить](#) [Клонировать](#) [Удалить](#) [Отмена](#)

События в Zabbix генерируются несколькими источниками:

- события на триггеры – всякий раз, когда триггер меняет свое состояние (ОК → ПРОБЛЕМА → ОК)
- события на обнаружение – при обнаружении узлов сети или сервисов
- события на авторегистрацию – когда активные агенты автоматически регистрируются сервером
- внутренние события – когда элементы данных/правила низкоуровневого обнаружения становятся не поддерживаемыми или триггер переходит в состояние неизвестно.

Zabbix Actions (действия)



Действия

Действие **Операции** Операции восстановления Операции подтверждения

Длительность шага операции по умолчанию:

Тема по умолчанию:

Сообщение по умолчанию:

Начало проблемы: {EVENT.DATE} — {EVENT.TIME}
Название триггера: {TRIGGER.NAME}
Хост: {HOST.NAME}
Критичность: {TRIGGER.SEVERITY}

ID исходной проблемы: {EVENT.ID} ({ZABBIX_HTTP}/tr_events.php?triggerid={TRIGGER.ID}&eventid={EVENT.ID})

Приостановить операции в режиме обслуживания: ☒

Действия

Действие Операции **Операции восстановления** Операции подтверждения

Тема по умолчанию:

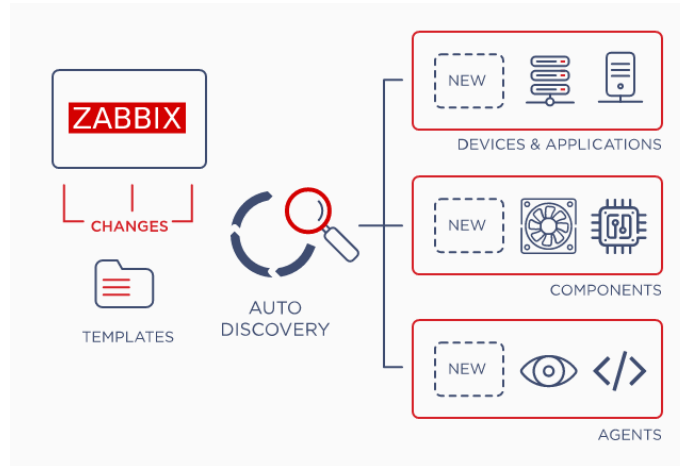
Сообщение по умолчанию:

Восстановление по проблеме: {EVENT.RECOVERY.DATE} — {EVENT.RECOVERY.TIME}

Название триггера: {TRIGGER.NAME}
Хост: {HOST.NAME}
Критичность: {TRIGGER.SEVERITY}

8

LLD



Low Level Discovery (низкоуровневое обнаружение)



Низкоуровневое обнаружение – это автоматическое создание элементов данных, триггеров и графиков на основе правил обнаружения.

1. Создаем правило обнаружения
2. Создаем прототипы элементов данных
3. Создаем прототипы триггеров
4. Создаем прототипы графиков
5. Профит!

The screenshot shows the Zabbix web interface for the 'Graphs' section. At the top, there's a 'Group' dropdown set to 'all'. Below it, a navigation bar shows 'All hosts / Zabbix server 1' followed by status indicators for 'Enabled', 'ZBX', 'SNMP', 'JMX', and 'IPMI'. On the right, it shows 'Applications 12', 'Items 74', and 'Triggers'. The main list displays several templates, each with a checkbox and a 'Name' column. The templates include 'Template OS Linux_b: CPU jumps', 'Template OS Linux_b: CPU load', 'Template OS Linux_b: CPU utilization', and three entries under 'Mounted filesystem discovery: Disk space usage /'. The last entry is expanded, showing a detailed view of a graph prototype. This view includes a list of two items: '1: Template OS Linux: Total disk space on {#FSNAME}' and '2: Template OS Linux: Free disk space on {#FSNAME}', each with a 'Graph' or 'Simple' button. Below this, there are links for 'Add' and 'Add prototype', and a table with 'tag' and 'value' columns.

checkbox	Name ▲
<input type="checkbox"/>	Template OS Linux_b: CPU jumps
<input type="checkbox"/>	Template OS Linux_b: CPU load
<input type="checkbox"/>	Template OS Linux_b: CPU utilization
<input type="checkbox"/>	Mounted filesystem discovery: Disk space usage /
<input type="checkbox"/>	Mounted filesystem discovery: Total disk space on / vfs.fs.size
<input type="checkbox"/>	Mounted filesystem discovery: Used disk space on / vfs.fs.size

1: Template OS Linux: Total disk space on {#FSNAME} Graph

2: Template OS Linux: Free disk space on {#FSNAME} Simple

Add Add prototype

tag value

Add

Low Level Discovery (низкоуровневое обнаружение)



Пример вывода кастомного правила низкоуровневого обнаружения (JSON):

```
{
  "data": [

    { "{#FSNAME}":"/",           "{#FSTYPE}":"rootfs"  },
    { "{#FSNAME}":"/sys",       "{#FSTYPE}":"sysfs"   },
    { "{#FSNAME}":"/proc",      "{#FSTYPE}":"proc"    },
    { "{#FSNAME}":"/dev",       "{#FSTYPE}":"devtmpfs"},
    { "{#FSNAME}":"/dev/pts",    "{#FSTYPE}":"devpts"  },
    { "{#FSNAME}":"/lib/init/rw", "{#FSTYPE}":"tmpfs"   },
    { "{#FSNAME}":"/dev/shm",    "{#FSTYPE}":"tmpfs"   },
    { "{#FSNAME}":"/home",      "{#FSTYPE}":"ext3"     },
    { "{#FSNAME}":"/tmp",       "{#FSTYPE}":"ext3"     },
    { "{#FSNAME}":"/usr",       "{#FSTYPE}":"ext3"     },
    { "{#FSNAME}":"/var",       "{#FSTYPE}":"ext3"     },
    { "{#FSNAME}":"/sys/fs/fuse/connections", "{#FSTYPE}":"fusectl"  }

  ]
}
```

9

ZABBIX

+



Grafana

Grafana: Zabbix Data Source



1. Переходим на <http://grafana.fintech-admin.m1.tinkoff.cloud>
2. Переключаемся на организацию: Main Org → Switch
3. Добавляем ваш Zabbix для сбора данных: Configuration -> Data Sources -> Add data source
4. Name: Zabbix, Type: Zabbix, URL: you-instance.fintech-admin.m1.tinkoff.cloud:8888
5. Save & Test

Домашнее задание



1. Установить Zabbix Server локально у себя на хосте
2. Установить Zabbix Agent локально у себя на хосте + зарегистрировать агента на Zabbix Server
3. Создать правило обнаружения (LLD): элементы данных + триггеры
4. Добавить кастомную метрику на агента + новый элемент данных + триггер
5. Настроить оповещение на триггеры (почта + Telegram)
6. Настроить дашборд в Grafana под системные метрики + добавить туда панель под кастомную метрику

Полезные ссылки / советы



- ❖ Официальная документация:
<https://www.zabbix.com/documentation/>
- ❖ Best practices:
https://www.zabbix.com/documentation/3.4/manual/installation/requirements/best_practices
- ❖ Планируйте вашу инсталляцию, особенно размер БД:
https://www.zabbix.com/documentation/3.4/manual/installation/requirements#database_size
- ❖ Используйте прокси для [масштабирования инсталляции](#).
- ❖ Не собирайте данные слишком часто (нагрузка на Zabbix сервер и БД), если этого не требуется.
- ❖ Разносите метрики и триггеры для различных систем/сервисов по шаблонам.
- ❖ Собирайте хосты в группы по системам и окружениям (MySQL servers + Production servers + MyServiceName + ...).
- ❖ Не храните исторические (неагрегированные) данные слишком долго, храните их в трендах (max/min/avg/count, агрегированные за час).
- ❖ Используйте [теги событий](#).
- ❖ Дублируйте каналы доставки алертов (почта + СМС/Telegram), не допускайте большого количества «шума».



Спасибо за внимание!

Роман Николаев

Email: r.nikolaev@tinkoff.ru

Telegram: @Nikolaev_RD