Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Grafana + Telegram: алерты, которые будят дежурного

Unified Alerting в Grafana 12.3: contact points, дерево политик, mute-окна и шаблоны под Git

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Метрики в офисе обычно уже собираются (Zabbix/Prometheus), но авария в пятницу вечером обнаруживается в понедельник: письма-алерты читают дважды в день. Мы доставляем critical в дежурный Telegram-чат за десятки секунд встроенным Unified Alerting Grafana — без отдельного Alertmanager. Бездействие — это простой всей компании и восстановление AD или 1С из бэкапа в рабочие часы.

Почему это важно бизнесу

- Час простоя сервера 1С или контроллера домена останавливает всех: 25–50 сотрудников не работают, а ФОТ и аренда капают
- Авария вечером пятницы без алерта — потерянное утро понедельника: восстановление начинается, когда все уже пришли
- Ранний сигнал (диск 85%, degraded RAID) превращает будущий простой в плановую замену без остановки бизнеса
- Шумный мониторинг хуже отсутствующего: дежурный свайпает уведомления, и реальная авария тонет в потоке

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

12.3

версия Grafana в наших внедрениях: Unified Alerting встроен, отдельный Alertmanager не нужен

Grafana 12.3.8, 06.2026

10 с

group_wait в critical-ветке вместо дефолтных 30 с — первое сообщение группы уходит почти сразу

наш стандарт

30 мин

repeat_interval для critical — напоминание дежурному; дефолт 4 ч для аварии слишком редкий

наш стандарт

4096

лимит символов одного сообщения Telegram — шаблон держим компактным, иначе доставка сорвётся

по докам Telegram Bot API

20/мин

лимит Bot API на сообщения в одну группу — без group_by шторм алертов упрётся в ошибку 429

по докам Telegram Bot API

10.4

с этой версии Grafana шлёт алерты в топики супергрупп через message_thread_id

по докам Grafana

РЕАЛИЗАЦИЯ 1 ИЗ 3

Бот и contact points: от BotFather до provisioning

Что настраиваем

Выделенная VM мониторинга клиента: Grafana 12.3 + источник данных Prometheus или Zabbix

Как мы это делаем

1. /newbot в @BotFather → токен в менеджер паролей; privacy mode отключаем командой /setprivacy, иначе getUpdates не увидит сообщения группы
2. Создаём две группы (дежурная и архив), добавляем бота; chat_id берём из api.telegram.org/bot<токен>/getUpdates — у групп он отрицательный
3. Contact point типа telegram: settings bottoken, chatid, parse_mode: HTML, disable_web_page_preview: true, message — наш шаблон tg.itfresh
4. YAML кладём в /etc/grafana/provisioning/alerting/ под Git; токен подставляем как ${TG_BOT_TOKEN} из EnvironmentFile systemd с правами 600
5. Проверяем кнопкой Test у contact point — тестовое сообщение должно прийти в оба чата

РЕЗУЛЬТАТ

Конфигурация воспроизводима: новый клиент подключается за часы, изменения проходят код-ревью, секреты не попадают в репозиторий, доставка подтверждена тестом до сдачи работ.

КЛЮЧЕВОЙ НЮАНС

Provisioning-YAML умеет только env-подстановку $VAR/${VAR}; синтаксис $__file{} работает лишь в grafana.ini — секрет передаём службе через окружение, а не пишем в файл.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Дерево notification policies и mute-окна

Что настраиваем

Маршрутизация трёх уровней severity в типовом офисе на 25–50 рабочих мест

Как мы это делаем

1. Корень: receiver tg-warning, group_by [alertname, instance]; дефолты по докам — group_wait 30s, group_interval 5m, repeat_interval 4h
2. Ветка severity=critical → tg-critical: group_wait 10s, repeat_interval 30m, continue: true — копия идёт дальше в email-архив для постмортемов
3. Ветка severity=warning: mute_time_intervals [nighttime] — две time_range 23:00–24:00 и 00:00–07:00 (диапазон не пересекает полночь), location Europe/Moscow
4. muteTimes backup-window (сб 23:00–24:00 и вс 00:00–03:00) глушит только алерты с label component=backup — critical по железу проходит всегда
5. Critical и warning разносим по топикам одной супергруппы через message_thread_id (поддерживается с Grafana 10.4)

РЕЗУЛЬТАТ

Critical будит дежурного за десятки секунд и напоминает каждые 30 минут, warning ждёт утра, субботний бэкап не спамит; в почтовом архиве — полная лента для разборов.

КЛЮЧЕВОЙ НЮАНС

Без continue: true алерт останавливается на первой совпавшей ветке — дубль в архив не уйдёт; порядок routes в дереве значим, проверяем предпросмотром маршрутизации.

Правила, пороги и шаблон сообщения

Что настраиваем

15–25 alert rules: хосты, диски, AD, rphost 1C, бэкапы, UPS, SSL-сертификаты

Как мы это делаем

1. Пороги нашего стандарта: диск >85% warning / >95% critical, RAM <10% 10 мин, CPU >90% 15 мин, бэкап не выполнялся >26 ч — critical
2. Pending period (for): 1m для critical, 5m для warning — секундные пики CPU не будят; от мигания — recovery threshold в threshold-выражении правила
3. Шаблон в templates.yaml: define "tg.itfresh" — статус, alertname, хост, severity, время; держим короче лимита Telegram в 4096 символов
4. В annotations каждого правила — summary и runbook-ссылка на карточку базы знаний: дежурный идёт по шагам, а не вспоминает ночью

РЕЗУЛЬТАТ

Дежурный за 5 секунд видит, что упало, где и что делать; 14 базовых правил закрывают ~95% офисных аварий, дальше — специфика: SIP-транки, кассы, SCADA.

КЛЮЧЕВОЙ НЮАНС

Каждый алерт обязан требовать действия и иметь runbook; метрике «для наблюдения» место на дашборде, а не в дежурном чате — иначе alert fatigue съест реакцию на реальную аварию.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **Бот не отдаёт chat_id группы**
  Privacy mode включён по умолчанию — getUpdates возвращает пустой массив. Отключаем /setprivacy в BotFather или даём боту админ-права и добавляем в группу заново.

- ✕ **$__file{} в alerting-YAML**
  В provisioning работает только $VAR/${VAR} из окружения; $__file{} — синтаксис grafana.ini. Токен храним в EnvironmentFile с chmod 600 на пользователя grafana.

- ✕ **parse_mode: HTML роняет доставку**
  Неэкранированный < или & в значении label даёт 400 от Bot API — уведомление теряется молча. Экранируем в шаблоне и тестируем на живых алертах, не только кнопкой Test.

- ✕ **Дефолтный repeat_interval 4 ч**
  Для critical это одно напоминание за смену — инцидент забывается. В critical-ветке ставим 30m, для warning оставляем дефолтные 4h.

- ✕ **Правки в UI поверх provisioning**
  Ресурсы из файлов получают признак provenance и в UI не редактируются; параллельные ручные правки создают расхождение. Меняем только YAML в Git и перечитываем конфиг.

- ✕ **Мониторинг молчит вместе с сетью**
  Если лёг интернет площадки или сама VM мониторинга, слать алерты некому. Держим heartbeat-правило always-firing и внешнюю проверку его регулярной доставки.

- ✕ **Все алерты в один чат**
  Info хоронит critical в ленте — дежурный перестаёт читать. Три severity — три канала: дежурный чат, чат смены, архив; или топики через message_thread_id.

- ✕ **Шторм при массовом падении**
  Bot API пропускает ~20 сообщений/мин в группу; без group_by и group_wait каскад алертов упрётся в 429. Группируем по alertname+instance, критику — group_wait 10s.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Grafana 12.x со встроенным Unified Alerting, без отдельного Alertmanager
- Бот через BotFather, два чата: дежурный (critical) и архивный (warning/info)
- 14 базовых правил: ping, диски, RAM/CPU, AD, rphost 1С, бэкап, UPS, SSL

НОРМАЛЬНО

- Весь алертинг в provisioning-YAML под Git: contactPoints, policies, muteTimes
- Ветки по severity: critical 10s/30m + continue, warning с mute 23:00–07:00
- Pending period for: 1m critical / 5m warning + recovery threshold от мигания
- HTML-шаблон tg.itfresh с runbook-ссылкой в каждом critical

ХОРОШО

- Топики супергруппы через message_thread_id (Grafana 10.4+)
- Heartbeat-правило + внешняя проверка живости самого мониторинга
- Еженедельный 15-минутный review сработавших и ложных алертов
- Mute-окна под регламентные работы по label component=backup

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Critical-алерт доходит до дежурного Telegram-чата быстрее чем за минуту — проверяли тестом?
- [ ] Токен бота лежит вне Git — в EnvironmentFile с правами 600, а не в YAML открытым текстом?
- [ ] В critical-ветке заданы свои group_wait и repeat_interval, а не дефолтные 30s/4h?
- [ ] Warning ночью замьючен через mute timing, а critical проходит круглосуточно?
- [ ] В каждом critical-сообщении есть ссылка на runbook — что делать по шагам?
- [ ] Плановые бэкапы и обновления закрыты mute-окном с label, а не терпением дежурного?
- [ ] Есть проверка «упал сам мониторинг» — heartbeat-правило и контроль его доставки извне?
- [ ] Ложных срабатываний не больше 1–2 в неделю и по пятницам они разбираются на review?
- [ ] Вся конфигурация алертинга поднимется из Git на чистой VM без ручных кликов в UI?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Разворачиваем VM мониторинга: Prometheus/Zabbix + Grafana 12.x с provisioning под Git за 2–3 рабочих дня
- Создаём бота, чаты и топики, строим дерево политик и mute-окна под график вашей смены
- Пишем 15–25 правил под вашу инфраструктуру: 1С, AD, бэкапы, UPS, SSL, телефония
- Готовим шаблоны сообщений с runbook-ссылками на базу знаний и обучаем дежурных
- Сопровождаем: разбор ложных срабатываний, новые правила, обновления Grafana

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 Configure Telegram for Alerting (contact points) (grafana.com — 12.3)

02 Import alerting resources with file provisioning (grafana.com — 12.3)

03 Configure notification policies (grafana.com — 12.3)

04 Provisioning — environment variable interpolation (grafana.com — 12.3)

05 Telegram Bot API: sendMessage, getUpdates (core.telegram.org — 2026)

06 Bots FAQ: broadcast limits (core.telegram.org — 2026)

07 Шаблон tg-itfresh + базовый набор из 14 правил (itfresh.ru — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.