Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра мережних технологій

Магістерська робота

СИСТЕМА ТЕЛЕМЕТРІЇ ДЛЯ ІНФОРМАЦІЙНОЇ ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА

Освітній ступінь – магістр

Виконав:

Студент 2-го року навчання

Спеціальності 121 Інженерія програмного забезпечення

Варченко Олександр Сергійович

Керівник:

Черкасов Д. І. кандидат технічних наук,

старший викладач

Київ – 2026

Вступ та Актуальність

Складність IT-інфраструктури

Сучасні підприємства мають розподілену структуру, яка налічує від десятків до тисяч компонентів, рознесених між локальними та хмарними центрами обробки даних.

Складність архітектури зумовлена наявністю фізичних серверів, віртуальних машин, контейнерів (Kubernetes) та мережевого обладнання.

Роль систем телеметрії

Обов’язковим інструментом для забезпечення надійності є система телеметрії, яка забезпечує збирання, зберігання, обробку та візуалізацію показників.

Традиційні підходи не завжди забезпечують необхідний рівень деталізації та масштабованості для хмарних технологій та мікросервісів.

Критичні задачі моніторингу

- Забезпечення високої доступності додатків (24/7) та уникнення фінансових втрат.
- Оптимізація продуктивності через відстеження ключових показників (KPI).
- Підтримка безпеки та виявлення вразливостей або підозрілої активності.

Мета та Завдання

Мета роботи

Розробка системи телеметрії, яка забезпечує моніторинг та аналіз стану інформаційної інфраструктури підприємства з метою підвищення її надійності та керованості.

1. Провести аналіз предметної області, дослідити існуючі підходи та інструменти моніторингу й телеметрії.
2. Розробити архітектуру системи телеметрії, що враховує вимоги до масштабованості та інтеграції компонентів.
3. Здійснити практичну реалізацію прототипу системи, провести його тестування та виконати оцінку ефективності запропонованого рішення.

Підходи до моніторингу

Критерій	Підхід на основі агентів	Безагентний підхід
Принцип роботи	Встановлення спеціального ПЗ (агентів) на вузлах системи	Використання існуючих протоколів (SNMP, WMI)
Розгортання	✗ Складніше через необхідність інсталяції	✓ Швидша початкова конфігурація
Обсяг даних	✓ Висока деталізація, ширша вибірка	✗ Обмежена глибина збору
Втрата зв'язку	✓ Можливе локальне збереження (буфер)	✗ Залежить від мережевої доступності
Ресурси сервера	Використовує CPU та RAM	Мінімальний додатковий вплив
Масштабованість	Складніше масштабувати	Легше завдяки відсутності агентів

Висновок: В реальних системах найефективнішим є комбінований підхід. Для критичних компонентів (де необхідний детальний збір та буферизація) використовується агентний моніторинг, а для інших – безагентний.

Моделі збору даних

Push модель

Агент сам надсилає дані на сервер

+ **Робота за NAT/Firewall:** Ініціатором є агент, легко налаштувати вихідний трафік.
+ **Реальний час:** Негайне відправлення при критичних подіях.
− **Ризик DDoS:** Після відновлення мережі агенти можуть перевантажити сервер.
− **Складність:** Кожен агент має знати адресу сервера і мати токени.

Pull модель

Сервер періодично запитує дані у агентів або сервісів

+ **Контроль навантаження:** Сервер повністю контролює частоту запитів.
+ **Миттєвий статус:** Відсутність з'єднання одразу фіксується як "Down".
− **Мережева доступність:** Сервер повинен мати прямий доступ до кожного агента (сервісу).
− **Пропуск коротких завдань:** Завдання, що працюють між інтервалами, не фіксуються.

Архітектура системи використовує **Гібридну модель**: Pull всередині кластера, Push до центрального вузла.

Архітектура системи телеметрії

Дочірній вузол (N шт.)

Збір та буферизація

- Збирання метрик (Pull-режим)
- Збирання журналів з нод
- Персистентна буферизація на диску
- Відправка даних (Push-режим)

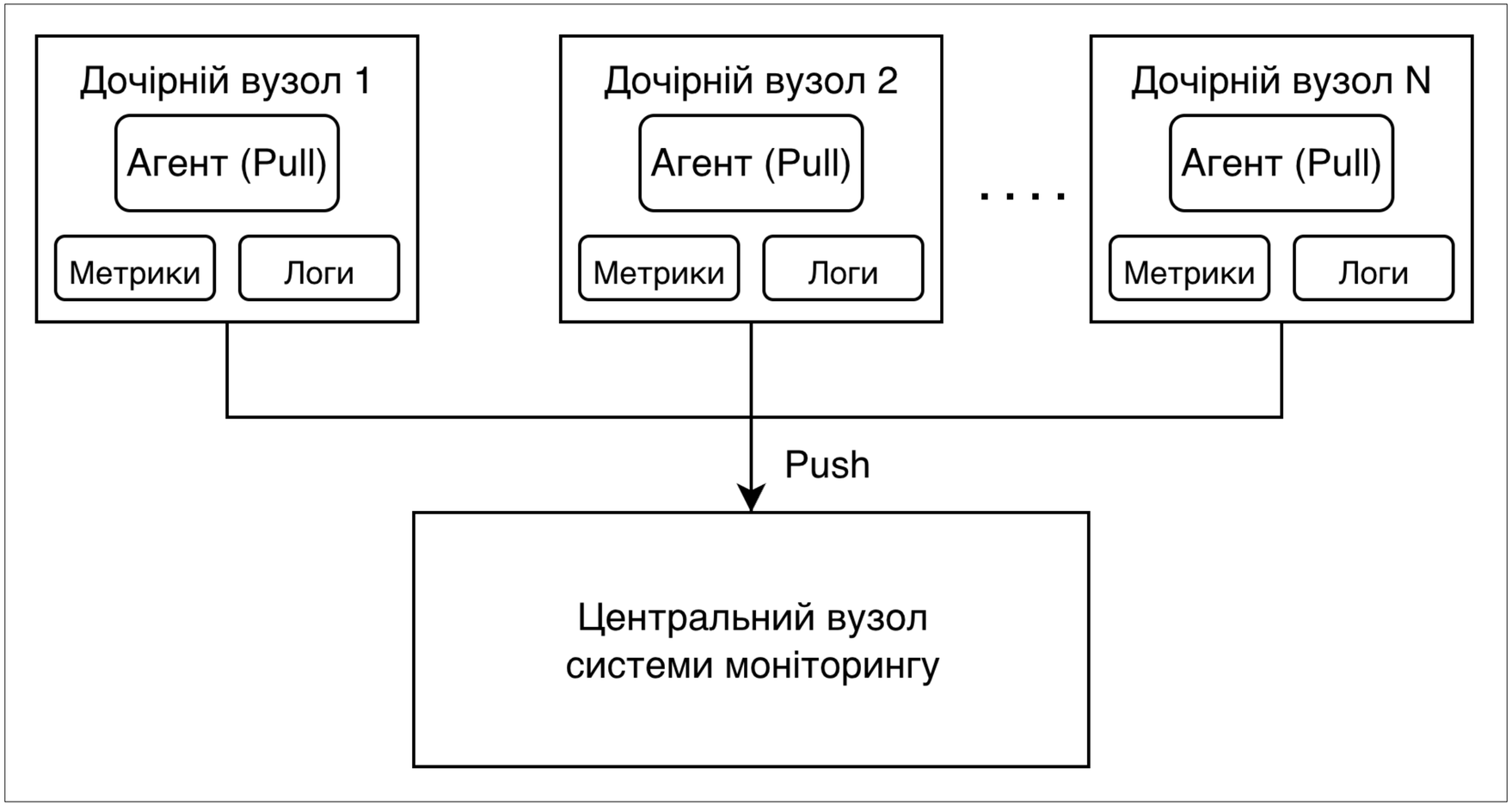
Централізований вузол

Зберігання та Аналіз

- Сховище метрик (TSDB)
- Сховище журналів
- Візуалізація (Дашборди)
- Підсистема сповіщень
- Зовнішній моніторинг ресурсів

Розділення на дочірні та центральний вузли дозволяє масштабувати систему без зміни конфігурації центрального сховища.

Архітектура системи телеметрії



Порівняльний аналіз

Prometheus	
Архітектура	**Моноліт**
Швидкість запису	**~ 240k семплів/с**
RAM (1 млн рядів)	**~ 2 - 8 ГБ**
Стиснення	**1.3 - 2.0 байт**
High Cardinality	**Низька**
Тривале зберігання	**Потребує Thanos**

InfluxDB	
Архітектура	**Моноліт / Кластер**
Швидкість запису	**~ 330k семплів/с**
RAM (1 млн рядів)	**> 16 ГБ**
Стиснення	**2.5 - 5.0 байт**
High Cardinality	**Низька**
Тривале зберігання	**Вбудоване**

Вибір для системи

VictoriaMetrics	
Архітектура	**Моноліт / Кластер**
Швидкість запису	**> 2M семплів/с**
RAM (1 млн рядів)	**~ 0.2 - 1 ГБ**
Стиснення	**0.4 - 0.8 байт**
High Cardinality	**Дуже висока**
Тривале зберігання	**Вбудоване**

Обґрунтування: VictoriaMetrics (VMSingle) споживає найменше ресурсів, ефективно обробляє високу кардинальність та підтримує тривале зберігання без сторонніх залежностей.

Порівняльний аналіз

Promtail (Loki)	
RAM (10k логів/с)	**63.00 МБ**
Процесор (CPU)	**0.412 ядра**
Пропускна здатність	**13 400 логів/с**
Буферизація на диску	**Обмежена**
High Cardinality	**Обмежена**

Vector	
RAM (10k логів/с)	**153.50 МБ**
Процесор (CPU)	**0.655 ядра**
Пропускна здатність	**25 000 логів/с**
Буферизація на диску	**Підтримується**
High Cardinality	**Повна**

Вибір для системи

VLAgent (VictoriaLogs)	
RAM (10k логів/с)	**27.91 МБ**
Процесор (CPU)	**0.062 ядра**
Пропускна здатність	**143 000 логів/с**
Буферизація на диску	**Нативна черга**
High Cardinality	**Нативна**

Обґрунтування: VLAgent потребує до 5 разів менше RAM, до 10 разів менше ядер CPU та забезпечує в 4 рази вищу пропускну здатність з надійною буферизацією.

Централізований вузол

Сховище метрик

VMSingle (VictoriaMetrics) Розгортається з персистентним сховищем. Акумулює всі часові ряди від дочірніх вузлів. Приймає дані через Remote Write протокол.

Сховище журналів

VLSingle (VictoriaLogs) Роздільне зберігання журналів дозволяє незалежно конфігурувати термін зберігання та ресурси, оптимізуючи дисковий простір.

Візуалізація

Grafana Єдина точка взаємодії. Отримує дані з обох сховищ, будує дашборди та відображає метрики й логи в єдиному інтерфейсі.

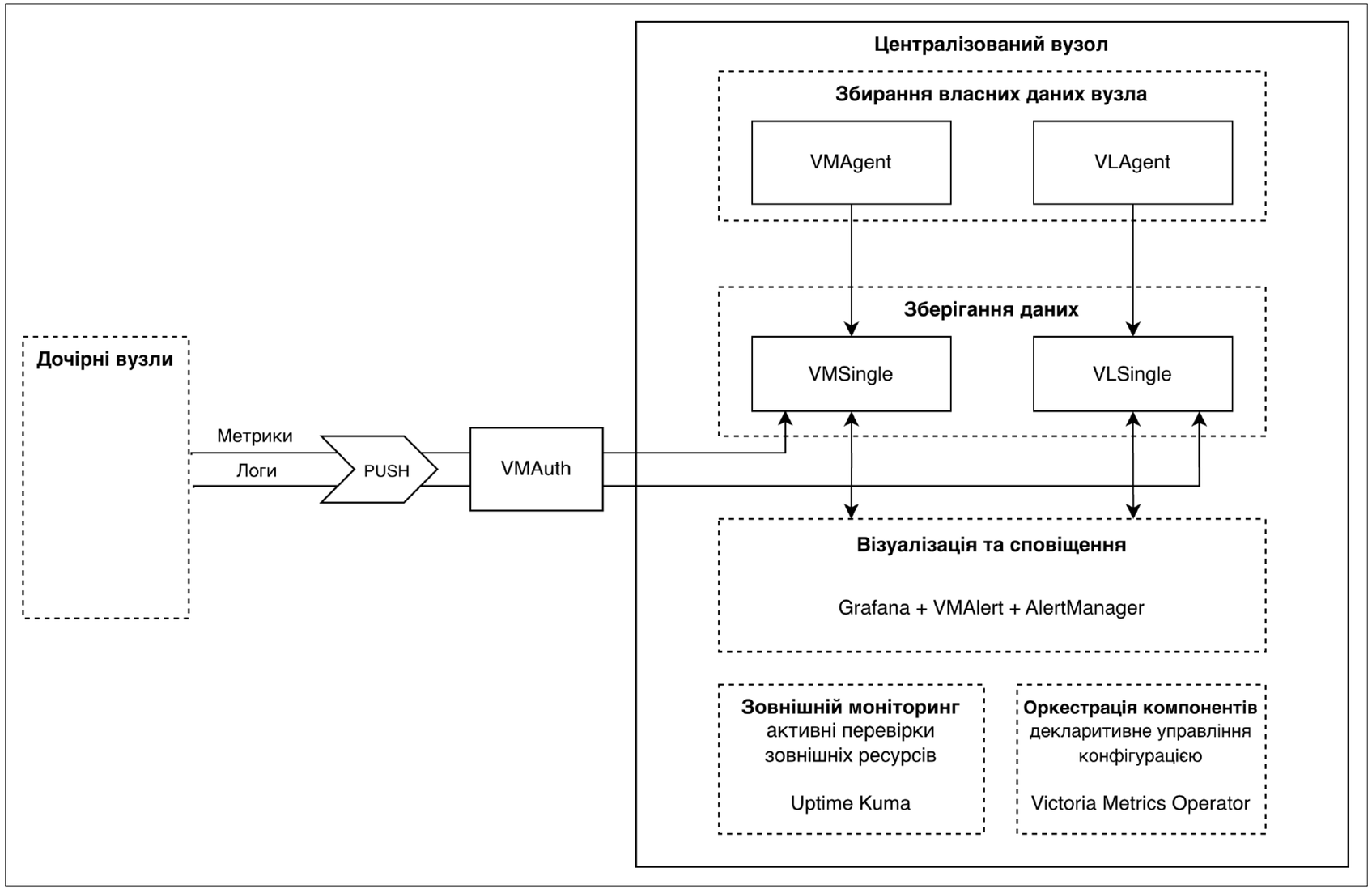
Сповіщення

VMAlert + Alertmanager Оцінка правил аномалій та надсилання агрегованих тригерів до зовнішніх каналів зв'язку (напр. Telegram).

Безпека та Доступ

VMAuth & Uptime Kuma VMAuth діє як захищений проксі (Basic Auth) для сховищ. Uptime Kuma здійснює активні зовнішні перевірки.

Централізований вузол



Дочірній вузол

Агент збирання метрик (VMAgent)

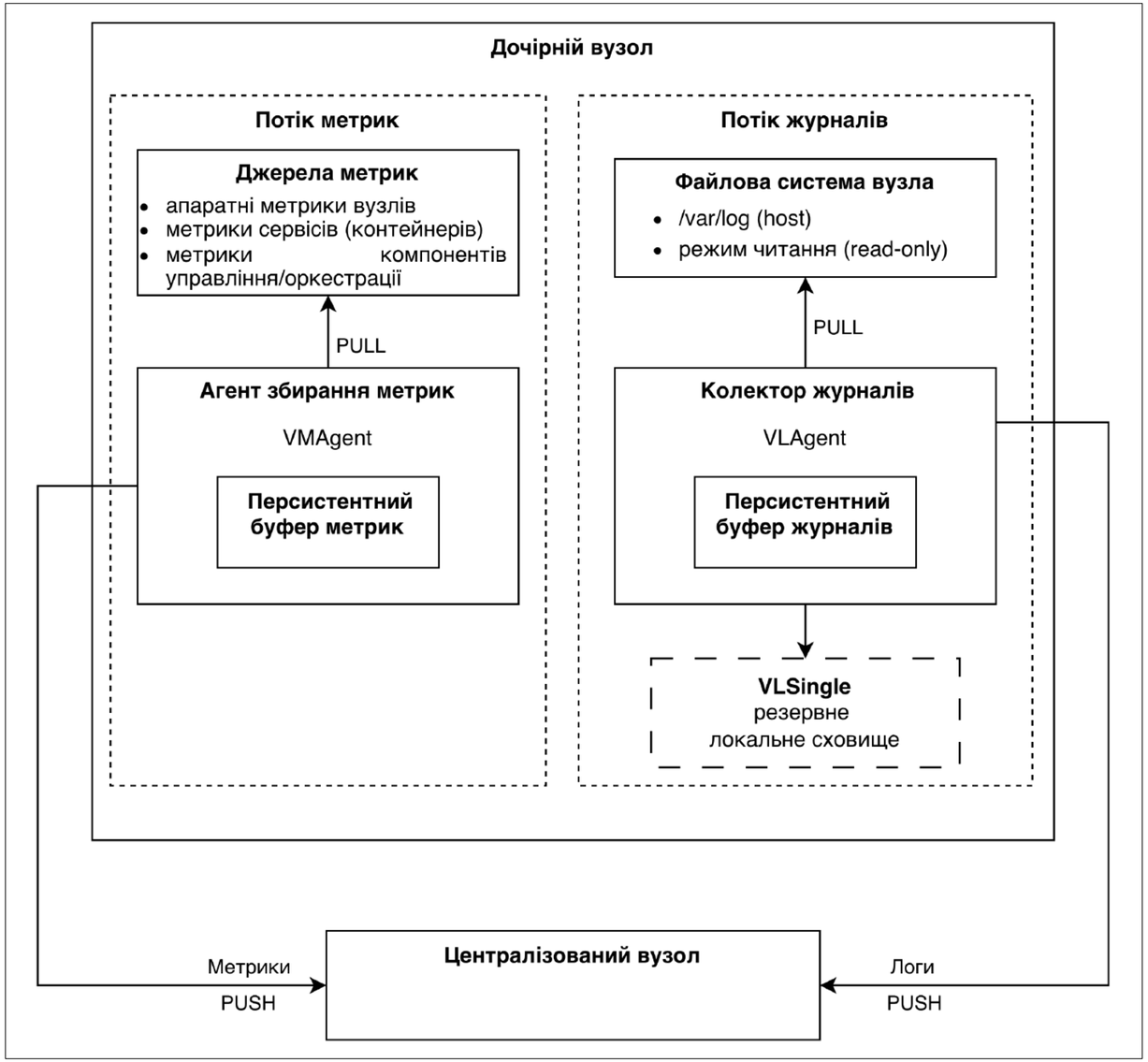
- Опитує локальні джерела (апаратні вузли, контейнери, Kubernetes API) в Pull-режимі.
- Додає унікальну мітку кластера (external label) для ідентифікації джерела на центральному вузлі.
- Передає метрики до централізованого сховища в Push-режимі через Remote Write.

Агент збирання журналів (VLAgent)

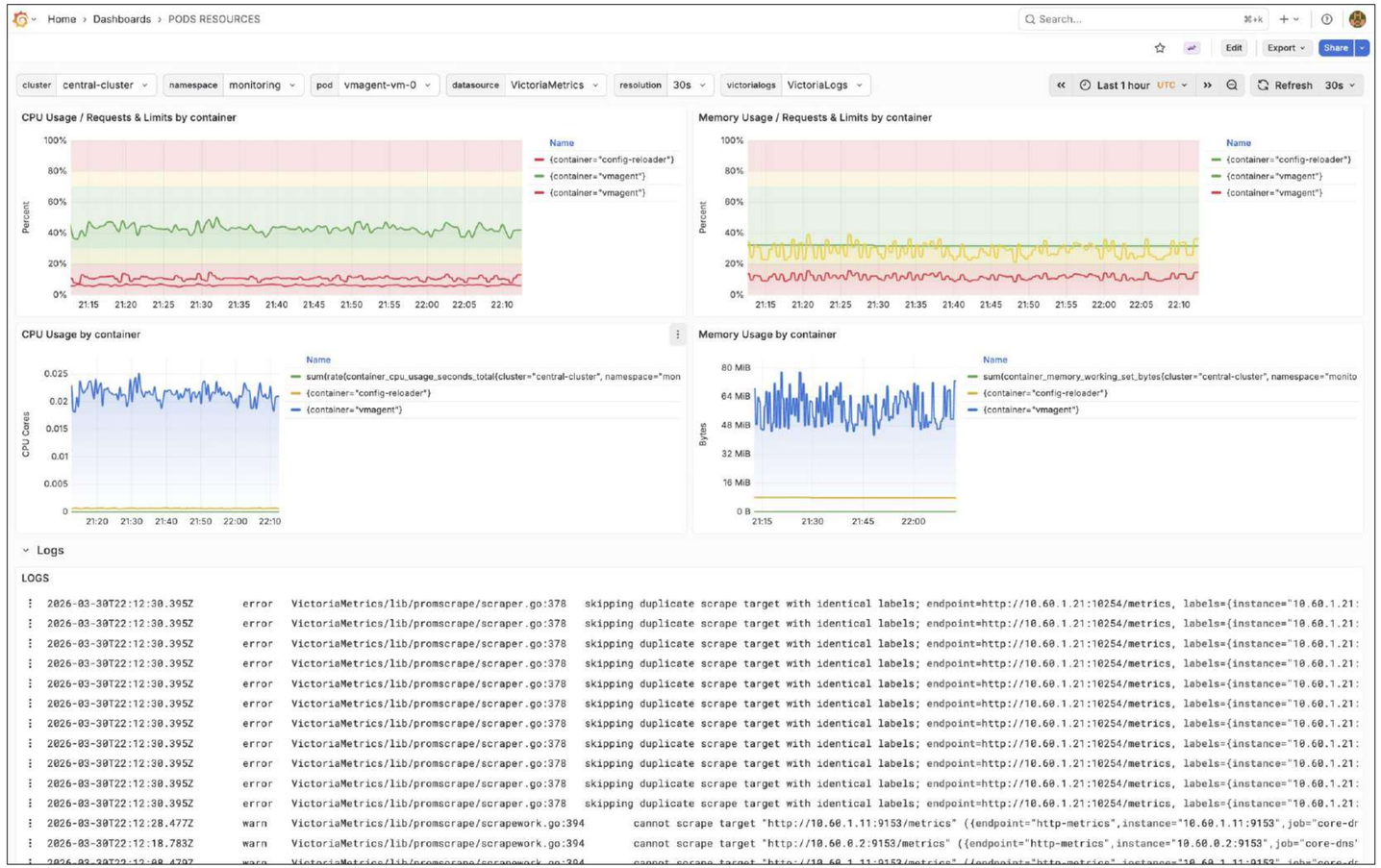
- Зчитує журнали з файлової системи вузлів у режимі read-only.
- Збагачує записи метаданими (namespace, pod) та мітками вузла.
- Підтримує дублювання даних у резервне локальне сховище для додаткового захисту.

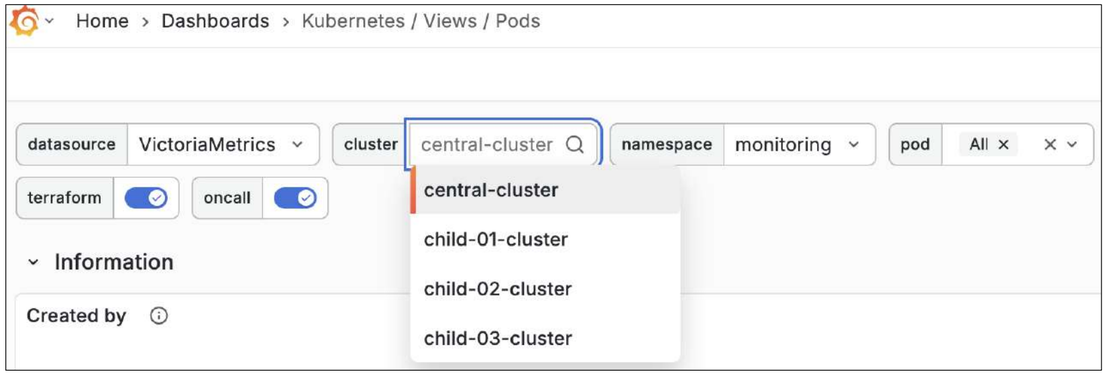
Критична вимога: **Відмовостійкість**. Обидва агенти використовують персистентні дискові буфери. При втраті зв'язку з центральним сервером дані накопичуються локально та автоматично надсилаються після відновлення з'єднання.

Дочірній вузол



Інтерактивні панелі в Grafana, Uptime Kuma

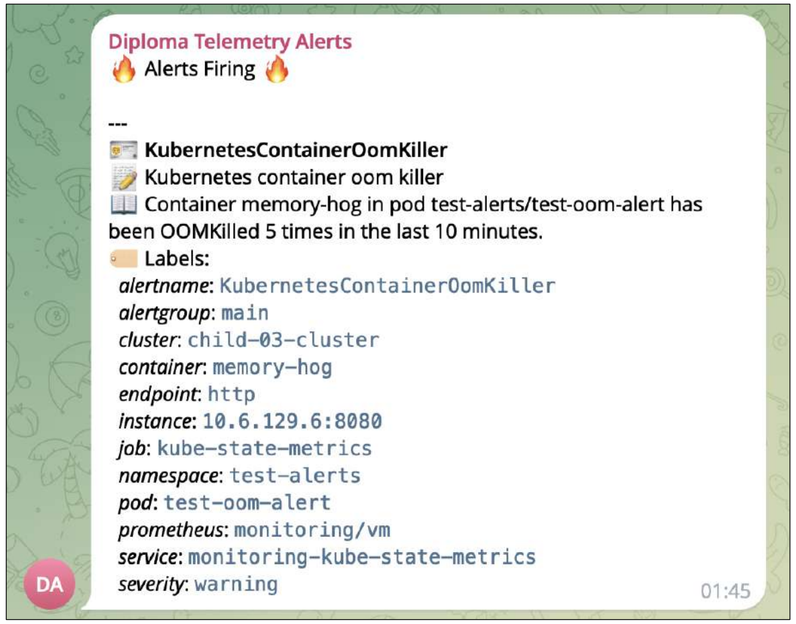




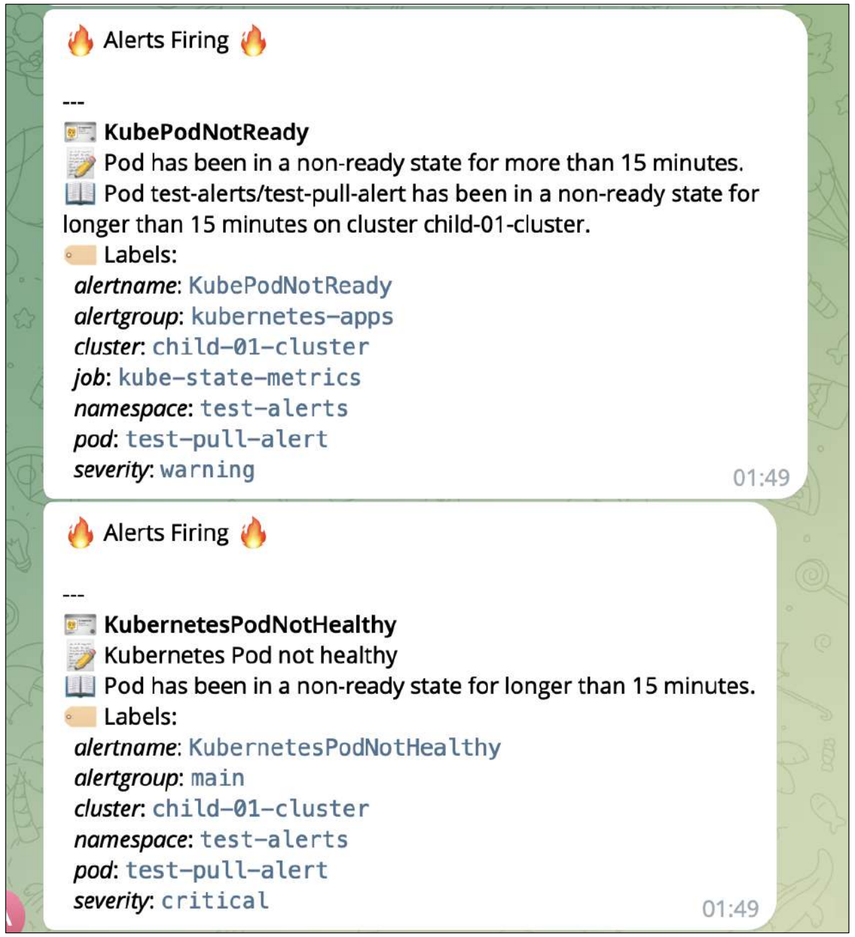
Pods with Container Issues

Container	Namespace	Pod Name	Reason
memory-hog	default	test-oom-alert	CrashLoopBackOff
non-existent-container	default	test-pull-alert	ImagePullBackOff

Сповіщення


Diploma Telemetry Alerts
🔥 Alerts Firing 🔥

KubernetesContainerOomKiller
Kubernetes container oom killer
Container memory-hog in pod test-alerts/test-oom-alert has been OOMKilled 5 times in the last 10 minutes.
Labels:
alertname: KubernetesContainerOomKiller
alertgroup: main
cluster: child-03-cluster
container: memory-hog
endpoint: http
instance: 10.6.129.6:8080
job: kube-state-metrics
namespace: test-alerts
pod: test-oom-alert
prometheus: monitoring/vm
service: monitoring-kube-state-metrics
severity: warning
01:45
DA


🔥 Alerts Firing 🔥

KubePodNotReady
Pod has been in a non-ready state for more than 15 minutes.
Pod test-alerts/test-pull-alert has been in a non-ready state for longer than 15 minutes on cluster child-01-cluster.
Labels:
alertname: KubePodNotReady
alertgroup: kubernetes-apps
cluster: child-01-cluster
job: kube-state-metrics
namespace: test-alerts
pod: test-pull-alert
severity: warning
01:49
🔥 Alerts Firing 🔥

KubernetesPodNotHealthy
Kubernetes Pod not healthy
Pod has been in a non-ready state for longer than 15 minutes.
Labels:
alertname: KubernetesPodNotHealthy
alertgroup: main
cluster: child-01-cluster
namespace: test-alerts
pod: test-pull-alert
severity: critical
01:49


✅ Alerts Resolved ✅

KubePodCrashLooping
Pod is crash looping.
Pod test-alerts/test-crash-alert (failing-container) is in waiting state (reason: "CrashLoopBackOff") on cluster child-02-cluster.
Labels:
alertname: KubePodCrashLooping
alertgroup: kubernetes-apps
cluster: child-02-cluster
container: failing-container
endpoint: http
instance: 10.116.0.6:8080
job: kube-state-metrics
namespace: test-alerts
pod: test-crash-alert
prometheus: monitoring/vm
reason: CrashLoopBackOff
service: monitoring-kube-state-metrics
severity: warning
02:00
DA

Висновки

Розроблено та реалізовано комплексну систему телеметрії

Надійність та Відмовостійкість

Гібридна архітектура з локальною буферизацією забезпечує безперервність моніторингу навіть при мережевих збоях. Додаткові локальні сховища VLSingle гарантують збереження логів при аваріях центрального вузла.

Масштабованість

Використання стеку VictoriaMetrics гарантує низьке споживання ресурсів. Декларативний підхід (Helm) дозволяє додавати нові вузли без змін у конфігурації центрального сервера.

Запропоноване рішення готове до промислового впровадження. Воно повністю відповідає вимогам сучасних хмарних систем, забезпечуючи високу доступність, безпеку (VMAuth), глибоку спостережуваність (Grafana + Uptime Kuma) та легкість розгортання і масштабування.

Перспективи розвитку

Шляхи подальшого вдосконалення системи

Кластерні Сховища

Перехід від VMSingle/VLSingle до VMCluster та VLCluster. Це усуне єдину точку відмови, розподіливши функції запису, зберігання та читання для високонавантажених середовищ (High Availability).

Географічна Реплікація

Впровадження резервних центральних вузлів. Налаштування агентів на дочірніх вузлах для паралельного надсилання даних одразу в декілька дата-центрів.

Посилена Безпека

Заміна Basic Auth на приватні мережеві тунелі, VPN або взаємну TLS-автентифікацію (mTLS). Це підвищить рівень захисту телеметричних даних у відкритих мережах.

Дякую за увагу!