



CENTQUATRE-PARIS
5 rue Curial, 75019



Mardi 3 Février
2026



CLOUD NATIVE DAYS FRANCE 2026

Du chaos à la clarté: booster votre pipeline de logs avec VictoriaLogs et LogsQL

Speaker:

Diana Todea - DevEx Engineer @VictoriaMetrics





Membre et contributrice d'OpenTelemetry

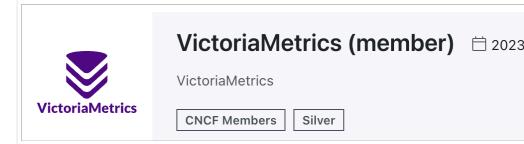
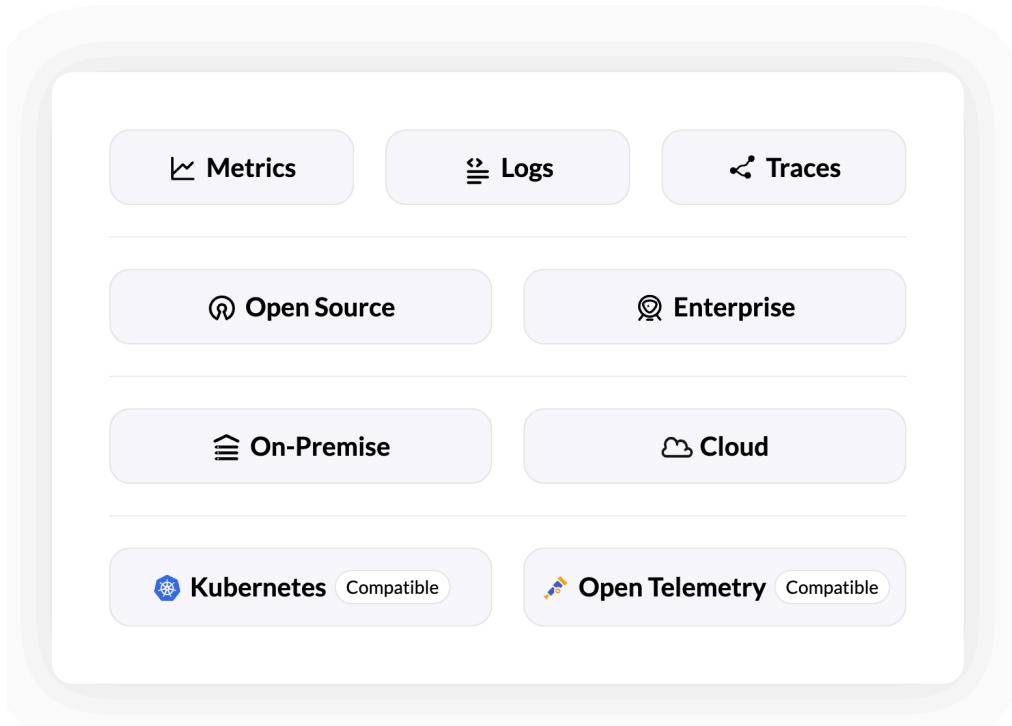
Organisatrice de Cloud Native Days Roumanie

Co-lead CNCF Merge-Forward Neurodiversité



A dark rectangular profile card featuring a circular photo of a woman with short brown hair, identified as Diana Todea. Below the photo, her name is displayed in white text, followed by her professional roles: "DX@VictoriaMetrics | Observability SME | OpenTelemetry docs localizatio...". To the right of the text is a large QR code.

VictoriaMetrics Observability Stack



VictoriaMetrics et sa stratégie open source

“En tant qu'utilisateur, j'apprécie l'open source car il permet d'apprendre et d'explorer le code source en production.

En tant que directeur technique de VictoriaMetrics, j'apprécie l'open source car il simplifie le recrutement d'excellents ingénieurs au sein de notre communauté, qui aiment travailler avec le code source des produits VictoriaMetrics.

En tant qu'actionnaire de VictoriaMetrics, j'apprécie l'open source car il contribue à accroître le nombre d'utilisateurs satisfaits.

Nous sommes satisfaits de la licence Apache2 pour les produits VictoriaMetrics et nous n'avons pas l'intention de la changer à l'avenir.”

Aliaksandr Valialkin - CTO et fondateur de VictoriaMetrics



Les contributions de VictoriaMetrics à l'Open Source et au Cloud Native



VictoriaLogs - qui suis-je?

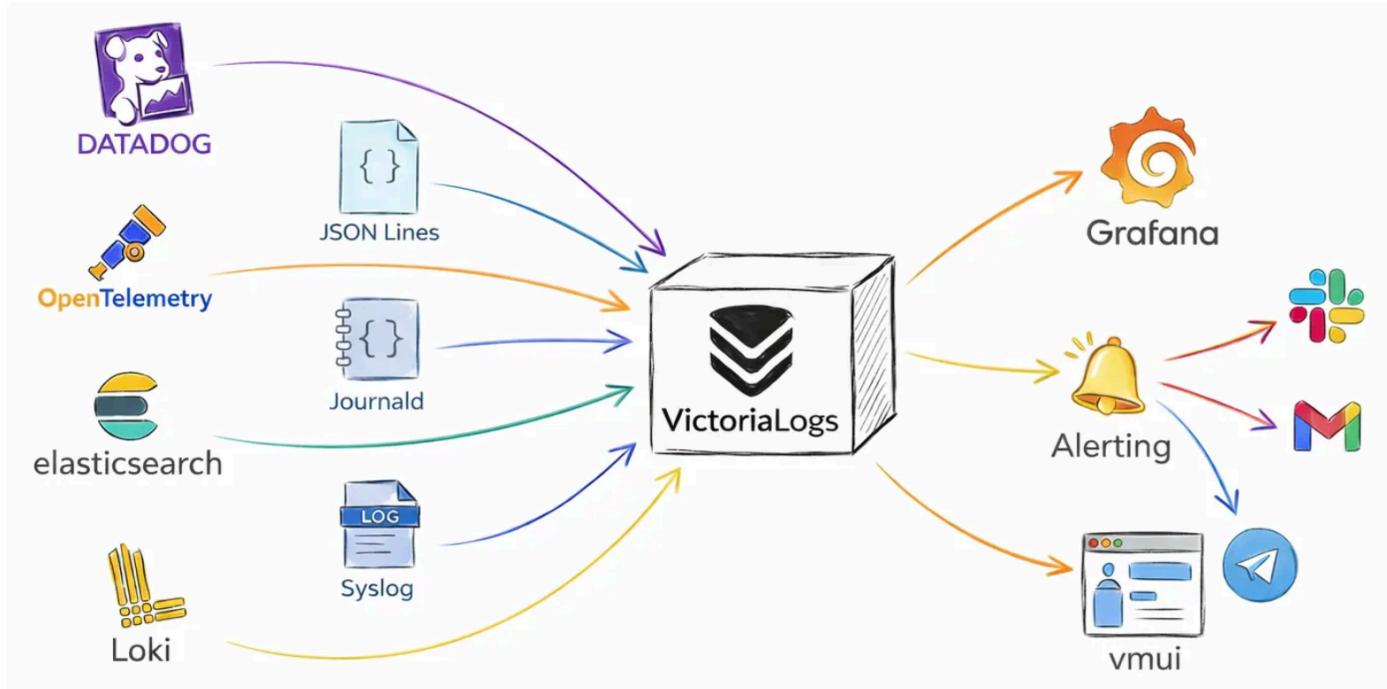
- Base de données de logs open source entièrement écrite en Go, évolutive des configurations les plus petites aux systèmes à plusieurs centaines de cœurs de processeur.
- Sans schéma, aucune installation, réglage ou configuration requise.
- Binaire léger (v1.43.1) : 20,7 Mio. Comparé à Loki (v.3.6.3) - 123,8 Mio, Clickhouse (25.12.3.21) - 727,1 Mio, Elasticsearch (8.17.0, intégré) - 1,3 Gio
- Gère des téraoctets de logs par jour sans problème et prend en charge la mise à l'échelle verticale et horizontale
- Consomme moins de CPU et de mémoire que la plupart des systèmes de journalisation, réduisant ainsi les coûts



VictoriaLogs - qui suis-je?

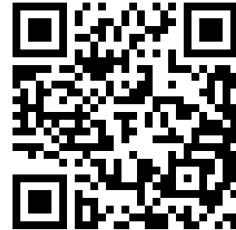
- Recherche et investigation plus rapides que la plupart des systèmes de journalisation; chaque champ est indexé
- Prend en charge les requêtes, filtres, groupes, calculs, analyses et la surveillance en direct des logs pour une visualisation en temps réel
- Compatible avec Prometheus et les tableaux de bord Grafana
- Gère les alertes basées sur vos logs et les règles d'enregistrement pour générer des métriques à partir des logs
- Créez des graphiques pour une vue d'ensemble du problème: explorez les détails des ID utilisateur, adresses IP, paiements, ID de traces, etc.





Benchmarks pour VictoriaLogs





Comparaison détaillée

	VictoriaLogs (c8g.4xlarge)	VictoriaLogs (c6a.4xlarge)	Elasticsearch (tuned) (c6a.4xlarge)	Timescale ☁ (Timescale ☁: 16 vCPU 64GB) (Timescale ☁: 8 vCPU 32GB)	Timescale ☁ (Timescale ☁: 8 vCPU 32GB)	Elasticsearch (c6a.metal)	Elasticsearch (c6a.4xlarge)
Load time:	1055s (×1.13)	1050s (×1.12)	7140s (×7.62)	3582s (×3.82)	4100s (×4.38)	9565s (×10.21)	9531s (×10.17)
Data size:	15.80 GiB (×1.00)	15.81 GiB (×1.00)	106.47 GiB (×6.75)	17.98 GiB (×1.14)	17.98 GiB (×1.14)	78.85 GiB (×5.00)	79.96 GiB (×5.07)
Q0.	0.044s (×2.39)	0.060s (×3.10)	0.017s (×1.20)	0.064s (×3.28)	0.063s (×3.27)	0.012s (×1.00)	0.024s (×1.49)
Q1.	0.026s (×2.20)	0.037s (×2.83)	0.007s (×1.05)	0.252s (×15.86)	0.373s (×23.21)	0.006s (×1.00)	0.044s (×3.24)
Q2.	0.923s (×5.07)	0.574s (×3.17)	0.896s (×4.93)	0.322s (×1.80)	0.504s (×2.79)	0.575s (×3.18)	0.634s (×3.50)
Q3.	1.018s (×5.91)	0.790s (×4.60)	1.558s (×9.01)	0.261s (×1.56)	0.411s (×2.42)	0.240s (×1.44)	0.680s (×3.97)
Q4.	1.113s (×3.77)	2.121s (×7.16)	0.503s (×1.72)	14.705s (×49.42)	22.495s (×75.59)	0.290s (×1.01)	0.297s (×1.03)
Q5.	0.929s (×1.74)	1.637s (×3.06)	1.458s (×2.72)	21.977s (×40.81)	43.578s (×80.91)	0.529s (×1.00)	0.745s (×1.40)
Q6.	0.040s (×3.72)	0.057s (×5.00)	0.439s (×33.28)	0.220s (×17.00)	0.318s (×24.33)	0.252s (×19.43)	0.306s (×23.39)
Q7.	0.025s (×1.00)	0.035s (×1.29)	0.265s (×7.90)	0.238s (×7.14)	0.325s (×9.65)	0.903s (×26.28)	0.887s (×25.80)
Q8.	1.364s (×1.00)	2.168s (×1.59)	2.391s (×1.75)	25.253s (×18.39)	80.955s (×58.94)	9.155s (×6.67)	8.898s (×6.48)
Q9.	1.808s (×1.00)	2.901s (×1.60)	3.333s (×1.84)	29.337s (×16.14)	88.163s (×48.50)	13.552s (×7.46)	12.829s (×7.06)
Q10.	0.279s (×1.07)	0.434s (×1.64)	0.305s (×1.16)	2.502s (×9.28)	9.625s (×35.59)	1.154s (×4.30)	1.258s (×4.69)
Q11.	0.336s (×1.06)	0.501s (×1.56)	0.359s (×1.13)	3.282s (×10.05)	13.538s (×41.37)	1.410s (×4.34)	1.592s (×4.89)
Q12.	0.769s (×2.39)	1.290s (×3.99)	0.316s (×1.00)	6.921s (×21.28)	7.551s (×23.21)	4.271s (×13.14)	4.254s (×13.09)
Q13.	1.594s (×4.58)	2.621s (×7.51)	0.340s (×1.00)	10.307s (×29.44)	32.204s (×91.91)	4.405s (×12.60)	4.219s (×12.07)
Q14.	0.856s (×2.26)	1.415s (×3.72)	0.373s (×1.00)	8.150s (×21.30)	8.493s (×22.20)	4.513s (×11.81)	4.484s (×11.73)
Q15.	1.527s (×2.49)	2.447s (×3.98)	0.950s (×1.55)	16.341s (×26.47)	17.416s (×28.21)	0.608s (×1.00)	0.815s (×1.34)



Benchmarks 2026 publiés par notre concurrent Arc

Ingestion: 4.58 Million Logs Per Second (With WAL Enabled)

Here's the headline number. Arc ingested 4,587,906 logs per second sustained over 60 seconds with WAL (Write-Ahead Log) enabled for durability. That's not a burst. That's sustained throughput with zero data loss guarantees.

SYSTEM	LOGS/SEC	TOTAL LOGS	VS ARC	P50	P95	P99	P999
Arc (WAL on)	4,587,906	275.3M	—	1.00ms	3.04ms	6.27ms	15.42ms
VictoriaLogs	2,262,593	135.8M	2x slower	0.86ms	16.34ms	22.90ms	37.10ms
Loki	1,135,568	68.1M	4x slower	4.38ms	8.91ms	25.36ms	31.43ms
ClickHouse	400,851	24.1M	11.4x slower	9.15ms	28.85ms	58.96ms	1,084ms
Quickwit	251,933	15.1M	18.2x slower	4.06ms	93.49ms	122.77ms	265.55ms
Elasticsearch	101,087	6.1M	45.4x slower	37.93ms	78.88ms	807.50ms	2,288ms

All systems ran for 60 seconds with 12 workers, 500 logs per batch, and 100% success rate (zero errors across the board).

Arc maintained a consistent 4.3–4.8M RPS throughout the entire test with WAL enabled, zero errors, and minimal latency variance. ClickHouse and Elasticsearch both showed throughput degradation over time—ClickHouse due to background merge operations, and Elasticsearch from segment management overhead.

VictoriaLogs deserves a shoutout here. At 2.2M logs/sec, it held strong and was the closest competitor. Solid system.



<https://basekick.net/blog/arc-log-benchmark-2026>



Fonctionnalités durables de VictoriaMetrics

1.7x

less memory

90%

energy cost
reduction

2.5x

less disk space

10x

infrastructure
cost Savings

16x

faster query
latency

4x

network cost
Savings



Benchmarks

Energy Savings

Hardware reduction



VictoriaMetrics

Simple, Reliable, Efficient Monitoring

victoriametrics.com

Fonctionnalités durables de VictoriaLogs

1000x

boosts haystack
search speed

30x

less RAM

15x

less disk
space



Benchmarks

Improved Query Performance

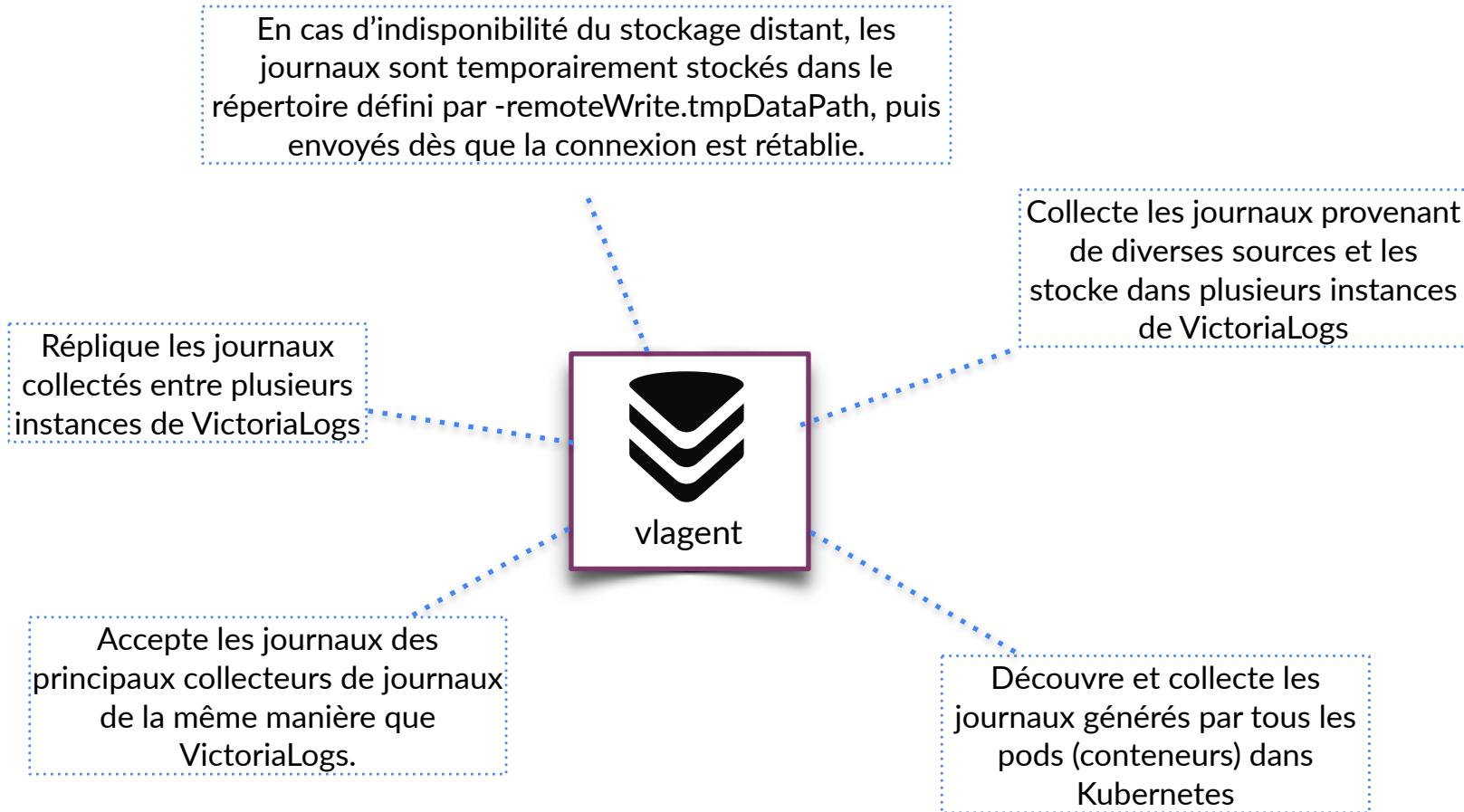
Resource Efficiency



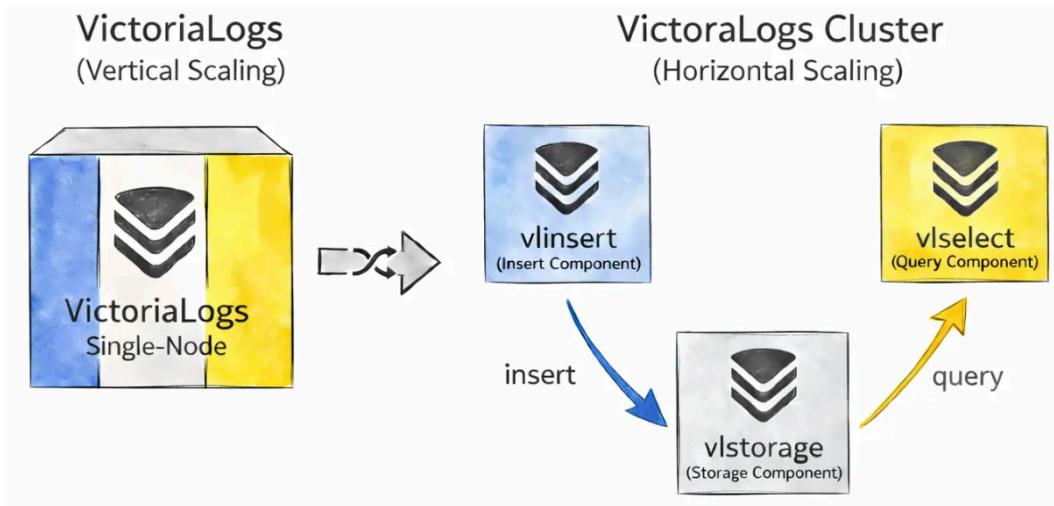
VictoriaMetrics

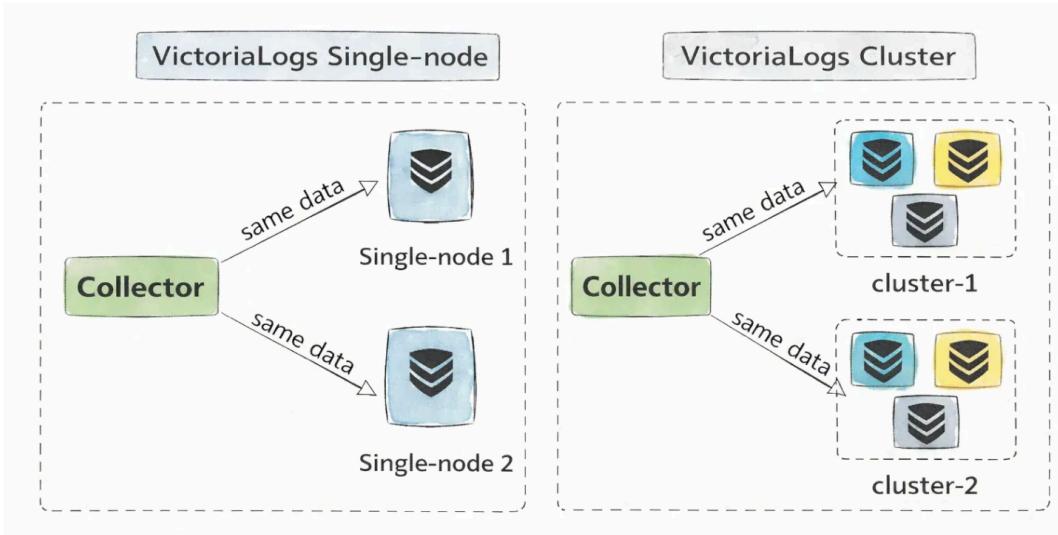
Simple, Reliable, Efficient Monitoring

victoriametrics.com



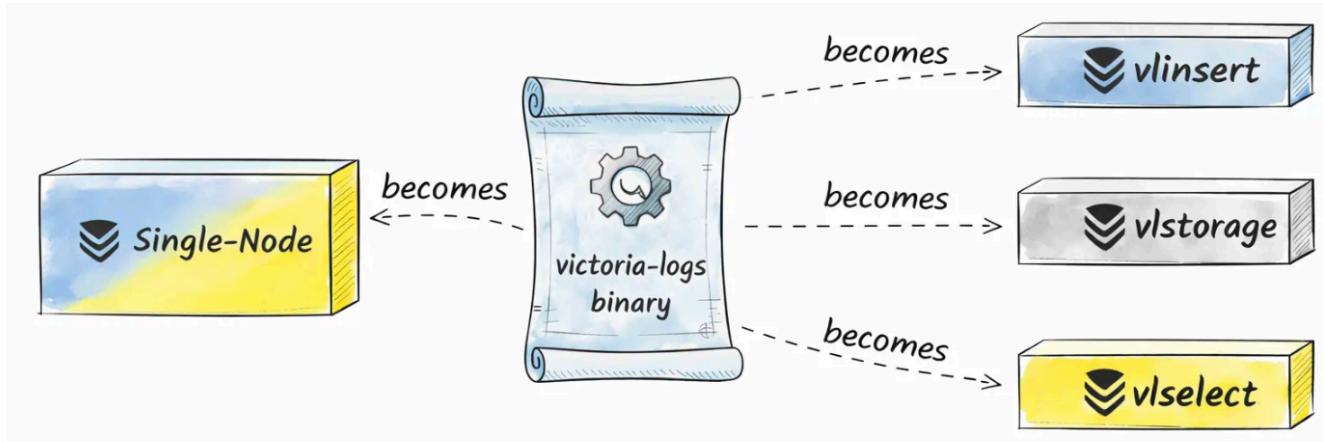
Nœud unique vs. cluster





- Avez-vous besoin du mode cluster pour la réPLICATION et la haute disponibilité?
- VictoriaLogs assure la haute disponibilité en répliquant les données au niveau de la couche d'ingestion (par exemple, vagent) et en envoyant les mêmes journaux à plusieurs cibles VictoriaLogs indépendantes: plusieurs instances mono-nœud ou plusieurs clusters distincts.

Les fonctionnalités de base sont également partagées !



Langage de requête : LogsQL



Tableau de bord Grafana mono-nœud de VictoriaLogs

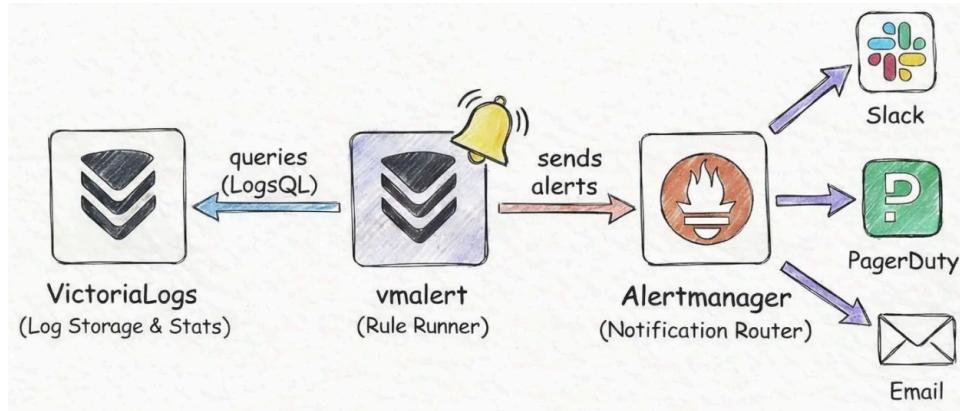


VictoriaMetrics

Simple, Reliable, Efficient Monitoring

victoriametrics.com

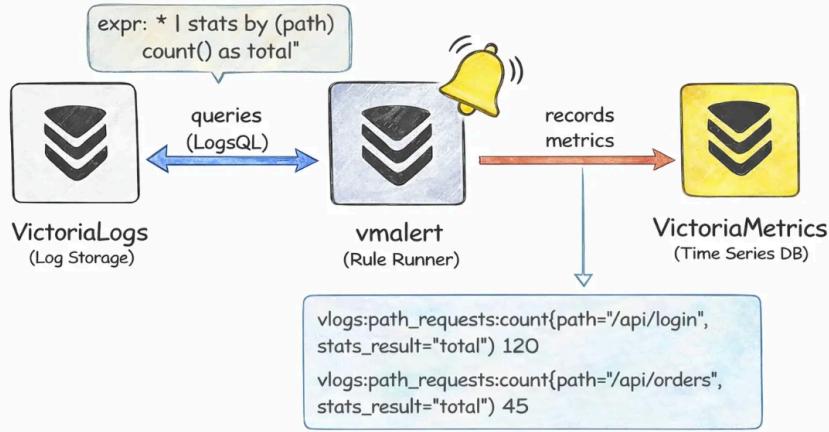
Règles d'alerte



- VictoriaLogs est la source de données pour les alertes, tandis que vmalert est l'exécuteur de règles.

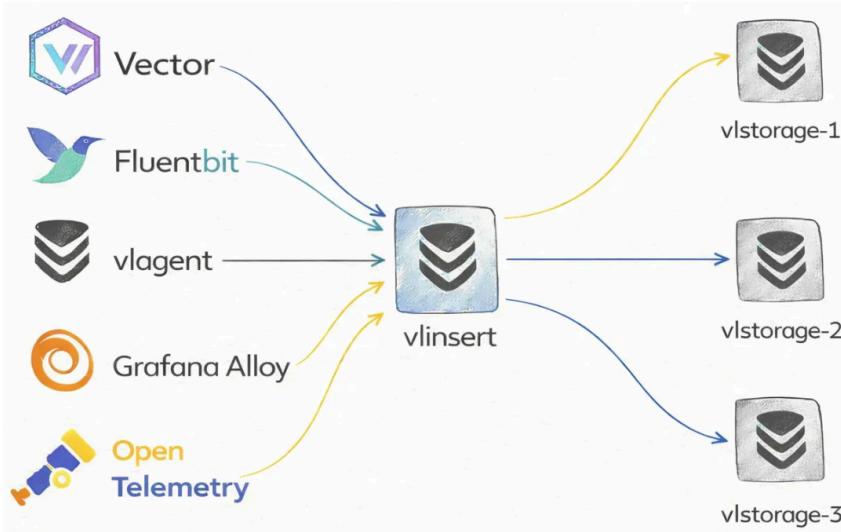
- VictoriaLogs stocke les journaux et calcule des statistiques à partir de ces journaux à l'aide de LogsQL.
- vmalert lit les fichiers YAML des règles, exécute périodiquement les requêtes de ces règles, détermine si une alerte est déclenchée et choisit l'emplacement d'enregistrement des résultats des règles.
- Alertmanager reçoit les notifications de vmalert et les achemine vers Slack, par e-mail et PagerDuty.

Règles d'enregistrement

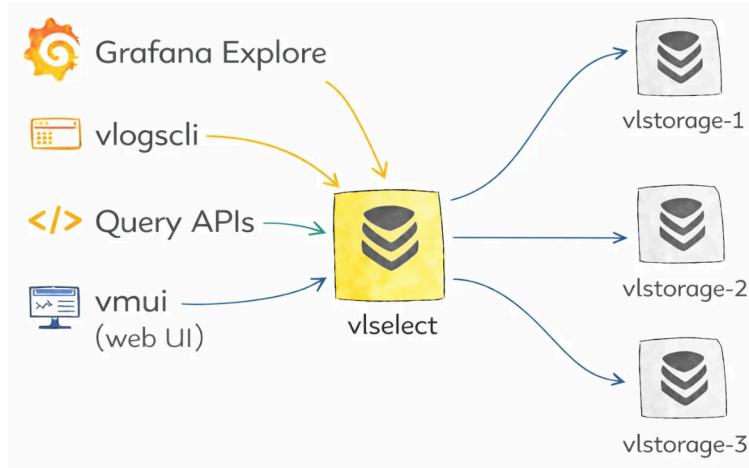


- Une règle d'enregistrement est une règle que vmalert exécute de manière planifiée. Elle transforme le résultat numérique de votre requête en une série temporelle, puis envoie ces échantillons à un système de métriques (généralement VictoriaMetrics via une écriture distante).
- Elle ne produit que des données de métriques réutilisables dans des tableaux de bord ou comme entrée pour d'autres alertes.

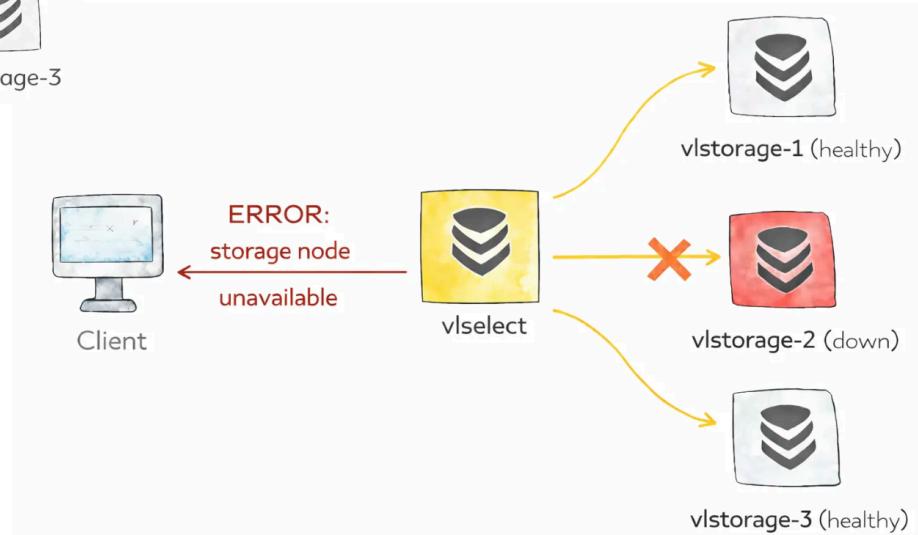
Composants du cluster VictoriaLogs



- **vlinsert** est la passerelle d'écriture. Les expéditeurs de journaux envoient des données à **vlinsert**.
- Ce dernier partitionne les journaux entrants et les transmet aux nœuds **vlstorage** (via le point de terminaison **interne /internal/insert**).

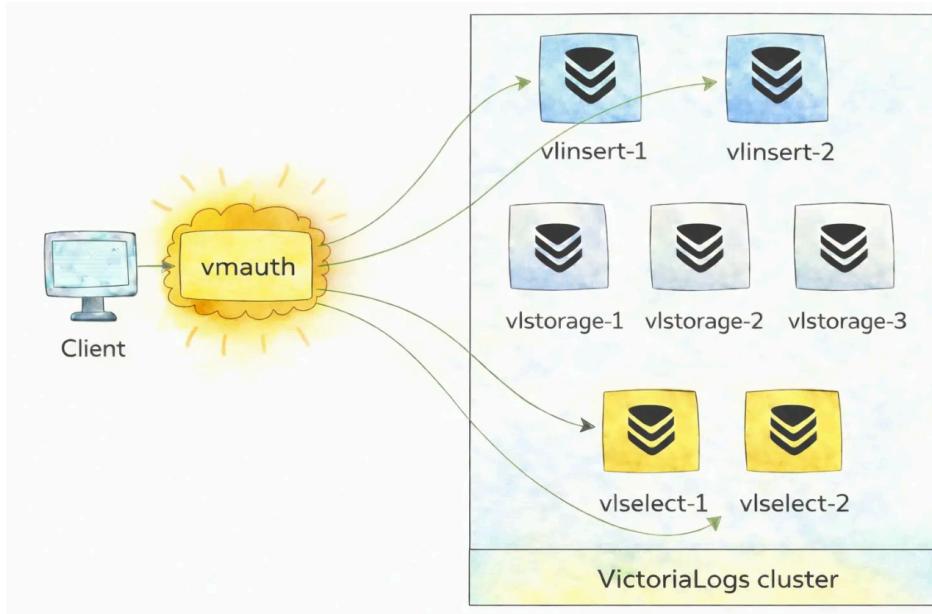


► lorsqu'un nœud devient indisponible, **vlselect** cesse temporairement de lui envoyer des données et redirige les écritures vers les nœuds de stockage restants.



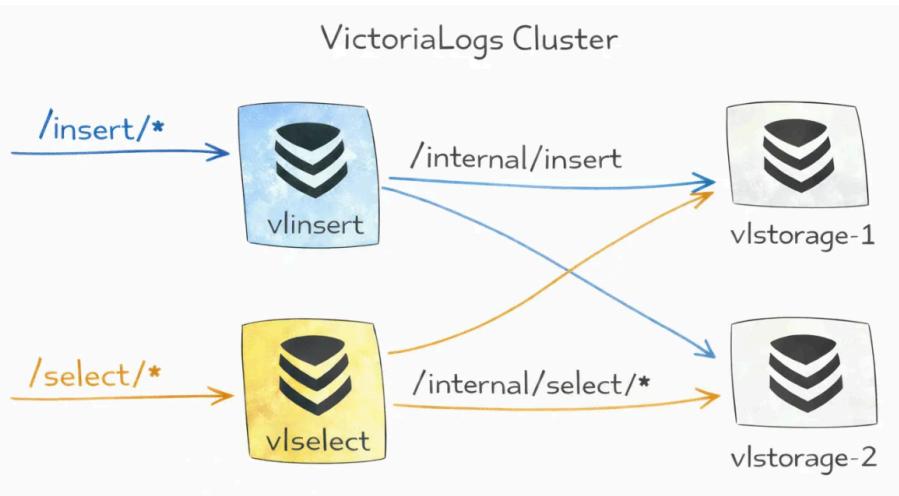
► par défaut, si un nœud de stockage est indisponible, la requête échoue et une erreur se produit.

Routage et sécurité



- Pour éviter toute utilisation inattendue de ces composants, il est courant d'utiliser un équilibrEUR de charge prenant en charge le routage strict, tel que **vmauth**.

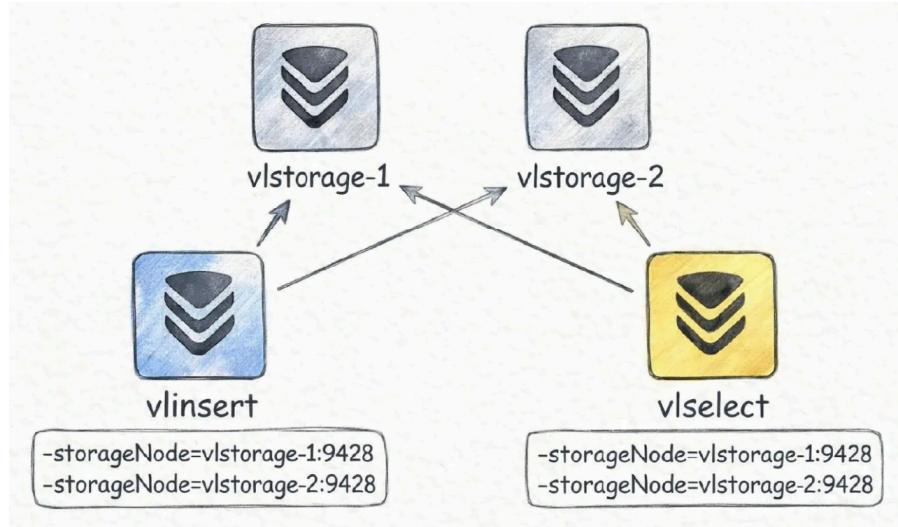
Comprendre le modèle de point final



- Les points de terminaison de l'API publique sont destinés aux clients : `/insert/*` pour l'ingestion dans **vlinsert** et `/select/*` pour les requêtes.
- Les points de terminaison internes au cluster sont réservés à la communication entre composants : `/internal/insert` et `/internal/select/*`.
- Ces chemins d'accès doivent être accessibles uniquement au sein du réseau privé et ne jamais être exposés publiquement.
- **vlstorage** est un service privé accessible uniquement depuis l'intérieur du cluster.

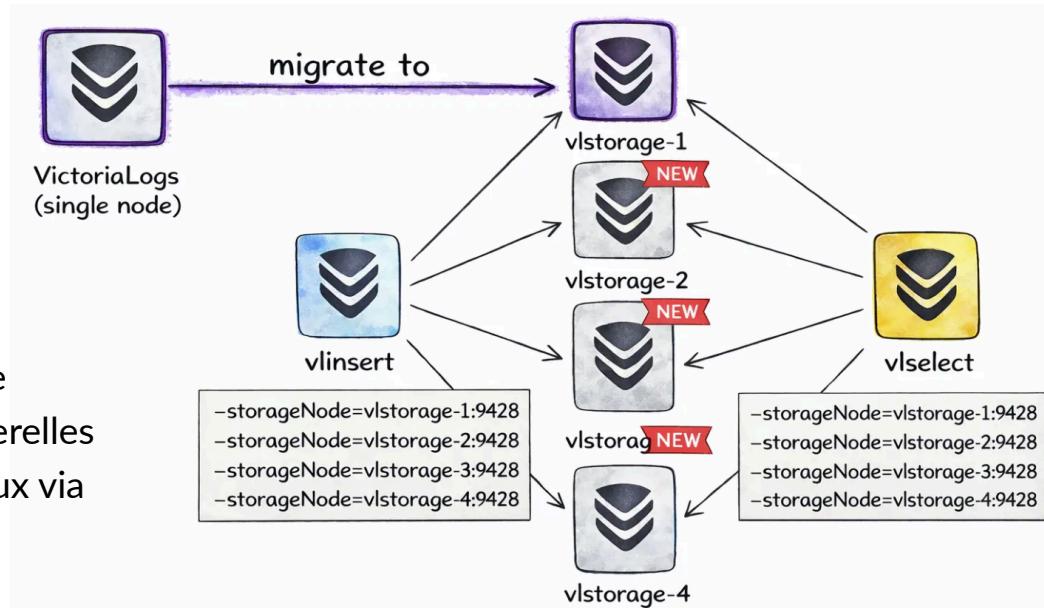
VictoriaLogs Cluster

- Tous ces rôles restent identiques au système binaire victoria-logs.
- La différence réside dans la manière de le démarrer.



Mise à l'échelle vers un cluster VictoriaLogs

- Ajoutez davantage de nœuds de stockage, puis pointez vos passerelles d'insertion et de requête vers eux via -storageNode.



Autres fonctionnalités utiles

- L'API **Delete** permet de supprimer les journaux déjà ingérés lorsque vous devez annuler une erreur (par exemple, des secrets/informations personnelles enregistrés accidentellement). Doit être activée avec l'option `-delete.enable`.
- **Le suivi en direct** permet de recevoir les nouveaux journaux en temps réel (comme `tail -f`, mais filtré par LogSQL) via `/select/logsql/tail`. Utilisé pour le débogage.
- **La multilocation (isolation des locataires)** vous permet d'utiliser un seul déploiement VictoriaLogs (mono-nœud ou cluster) pour plusieurs équipes ou environnements tout en gardant leurs journaux séparés.
- **Protection contre le manque d'espace disque:** le point de terminaison `/metrics` expose `vl_free_disk_space_bytes{path="..."}` et `vl_storage_is_read_only{path="..."}`. Ces métriques suffisent pour une alerte simple, permettant ainsi de gérer les situations de faible espace disque avant le blocage de l'ingestion.



Environnement de test de conversion de requêtes LogQL en LogsQL

VictoriaLogs endpoint
You can query data from VictoriaLogs instance or just translate LogQL to LogsQL without querying

URL: Bearer token:

LogQL: `{collector="otel-collector"} |= "POST"` Example: Line filter (contains) Last 1 hour

LogsQL query results

_time	_stream_id	_stream	_msg
2026-01-13T13:30:56.179Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:56.179Z] "POST"
2026-01-13T13:30:56.152Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:56.152Z] "POST"
2026-01-13T13:30:35.596Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:35.596Z] "POST"
2026-01-13T13:30:26.215Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:26.215Z] "POST"
2026-01-13T13:30:26.202Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:26.202Z] "POST"
2026-01-13T13:30:26.182Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:26.182Z] "POST"
2026-01-13T13:30:26.167Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:26.167Z] "POST"
2026-01-13T13:30:03.540Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:03.540Z] "POST"
2026-01-13T13:30:00.326Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:30:00.326Z] "POST"
2026-01-13T13:29:45.147Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:29:45.147Z] "POST"
2026-01-13T13:29:32.856Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:29:32.856Z] "POST"
2026-01-13T13:29:17.653Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:29:17.653Z] "POST"
2026-01-13T13:28:41.606Z	00000000000000061aa9bfe28dc9e7d8fb8daa972f1c99d	{collector="otel-collector",k8s.namespace.name="play-otel",k8s.pod.name="frontend-proxy-6f796d6c45-q84pd",service.name="frontend-proxy"}	[2026-01-13T13:28:41.606Z] "POST"

Information about LogQL to LogsQL local

Service that helps to query VictoriaLogs with Loki LogQL

Source code and documentation

Supported query types

- Log queries
- Stats queries

Supported pipeline stages

Supported metric functions (subset)

La feuille de route de VictoriaLogs

- **Stockage objet:** permet de stocker et de lire les données de journalisation depuis des backends compatibles S3, tels qu'AWS S3 et MinIO. La hiérarchisation des données (chaud/froid) est ainsi activée: les données récentes restent sur des disques locaux rapides tandis que les données plus anciennes restent consultables à moindre coût.
- **Transformation des journaux (vlagent):** intègre un pipeline de transformation au sein de l'agent. Les journaux peuvent être analysés, normalisés, enrichis, filtrés et leurs champs renommés ou supprimés avant ingestion, ce qui réduit la complexité des requêtes en aval et améliore la cohérence des données.



La feuille de route de VictoriaLogs

- **Outils de migration:** fournit des utilitaires de migration pour le transfert de données vers VictoriaLogs (par exemple, depuis Elasticsearch) et entre instances VictoriaLogs, en privilégiant l'exactitude, la réutilisabilité et une interruption de service minimale.
- **Outils de sauvegarde et de restauration:** intègre des outils officiels de type `vmbackup/` `vmrestore/vmbackupmanager` pour VictoriaLogs. Les opérateurs peuvent ainsi effectuer des sauvegardes cohérentes, restaurer les données sur de nouveaux clusters, automatiser la conservation des sauvegardes et des instantanés, et prendre en charge les procédures de reprise après sinistre.



Ressources

<https://play.victoriametrics.com/>

<https://play-vmlogs.victoriametrics.com/>

<https://play-logql.victoriametrics.com/>

<https://play-vtraces.victoriametrics.com/>

<https://play-grafana.victoriametrics.com/dashboards>

<https://docs.victoriametrics.com/victorialogs/>

<https://docs.victoriametrics.com/victorialogs/logql-to-logsql/>



Merci!

Bsky: @didiviking.bsky.social

X: @dianavtodea

Github: @didiViking/Conferences_Talks

LinkedIn: @diana-todea-b2a79968



VictoriaMetrics Community



Apprendre encore plus



community.cncf.io/merge-forward

Créer des communautés **diversifiées** et **solidaires** ainsi que **des réseaux d'alliés** pour l'apprentissage partagé, le mentorat, l'amitié et l'échange collaboratif d'idées.

#merge-forward sur le Slack de
CNCF!

Avis :

