vmagent 數據處理與優化

How vmagent Collects and Shi...
PDF

參考"How vmagent Collects and Ships Metrics Fast with Aggregation, Deduplication, and More.pdf"，以繁體中文詳細說明

vmagent 如何透過聚合、去重與其他技術高效收集與傳輸指標數據

1. vmagent 簡介

VictoriaMetrics Agent (vmagent) 是一款輕量級的數據收集工具，設計用來從多個來源收集監控指標數據。收集後，vmagent 可以透過 **重標籤（Relabeling）**、**數據篩選（Reducing Cardinality）**、**流式聚合（Stream Aggregation）** 和 **去重（Deduplication）** 來處理數據，並將其傳輸到儲存系統，如 **VictoriaMetrics**，或其他支援 Prometheus Remote Write 協議的存儲服務。

---

2. 數據接收與處理

vmagent 透過 **API 接收數據** 或 **定期從目標抓取（Scraping）數據**。

2.1 API 接收數據

- 支援 HTTP API，允許使用者透過 **查詢參數** 或 **Pushgateway 格式 URL** 添加額外標籤。
- 採用 **並發限制（Concurrency Limiter）**，以防止過多請求佔用資源：
  - 預設允許的並發寫入數量為 **CPU 核心數的 2 倍**。
  - 若請求排隊時間超過 **1 分鐘**，vmagent 會回應 **503 錯誤**。

2.2 抓取指標數據（Scraping）

- vmagent 可從 Prometheus 兼容的端點 **定期抓取數據**。
- 預設抓取間隔 **1 分鐘**，超時 **10 秒**。
- 若回應超過 **16 MB**，vmagent 會丟棄超過該大小的數據。

數據處理模式

- **單次處理模式（One-shot mode）**：適用於小型請求，處理整個抓取回應。
- **流式處理模式（Stream mode）**：適用於大型請求，按 **64 KB** 塊逐步處理。

---

3. 數據標籤處理與降維（Relabeling & Cardinality Reduction）

- **重標籤（Relabeling）**：
  - vmagent 可透過 **metric_relabel_configs** 根據條件篩選、修改或刪除標籤，以降低數據量。
  - 例如，將 `{job=~"my-app-.*", env!="dev"}` 的 `foo` 標籤改為 `bar`。
- **降維（Reducing Cardinality）**：
  - 避免高標籤基數（High Cardinality）問題，例如 `user_id` 或 `IP` 產生過多時序數據。
  - 設定 `-remoteWrite.maxHourlySeries` 和 `-remoteWrite.maxDailySeries` 限制小時或每天允許的唯一時序數量，超過則丟棄。

---

4. 去重（Deduplication）與流式聚合（Stream Aggregation）

- **去重（Deduplication）**
  - vmagent 透過 `-streamAggr.dedupInterval` 設定 **去重時間窗口**，僅保留時間窗口內最新或最大值的數據。
  - 例如，設定 `30s`，則只保留該 30 秒內最新的指標數據。

- **流式聚合（Stream Aggregation）**
  - vmagent 可按 **固定間隔** 聚合數據，以減少存儲佔用。
  - 例如，設定 `5m`，則所有 `_total` 結尾的指標將被聚合為 `5 分鐘總計`。
  - 透過 `-streamAggr.keepInput` 和 `-streamAggr.dropInput` 來控制原始數據是否保留或丟棄。

去重與聚合的順序

1. vmagent **先執行去重**，確保每個時間窗口內只保留最重要的數據。
2. **剩餘數據再進行流式聚合**，壓縮數據量。

---

5. 分片（Sharding）與複製（Replication）

- **數據複製（Replication）**
  - 若配置了多個遠端存儲（`-remoteWrite.url`），則 vmagent 會向每個存儲系統 **發送相同的數據**。
- **數據分片（Sharding）**
  - 透過 `-remoteWrite.shardByURL`，vmagent 可根據時序數據的標籤計算哈希值，將數據 **均勻分配** 到不同存儲節點，避免單點過載。
  - 也可透過 `-remoteWrite.shardByURLReplicas` **增加副本數量**，確保存儲系統容錯能力。

---

6. 遠端存儲優化與樣本值處理

- vmagent 支援：
  - **重新標籤遠端數據**（`-remoteWrite.urlRelabelConfig`）。
  - **遠端存儲級別的流式聚合**（`-remoteWrite.streamAggr.config`）。

- **數據標籤統一化**（`-remoteWrite.label`）。
- **樣本值四捨五入**：
  - `-remoteWrite.significantFigures` 控制**有效數字**（如四捨五入至 2 位數）。
  - `-remoteWrite.roundDigits` 控制 **小數點位數**（如保留 2 位小數）。

這些處理有助於提高 **數據壓縮率**，減少存儲佔用。

---

7. 數據快取與寫入機制

vmagent 透過 **內存快取（In-Memory Queue）** 和 **磁碟快取（File-based Queue）** 來避免數據丟失。

- **內存快取**
  - 使用 Go **FIFO 佇列**，若 5 秒內未能寫入遠端存儲，則數據轉存磁碟快取。
  - 默認最多允許 60% 記憶體儲存數據 (`-memory.allowedPercent`)。
- **磁碟快取**
  - 預設存儲路徑：`/vmagent-remotewrite-data`，可透過 `-remoteWrite.tmpDataPath` 調整。
  - 若存儲空間達上限，vmagent 會 **刪除最舊的數據塊** 來騰出空間 (`-remoteWrite.maxDiskUsagePerURL`)。

---

8. 最終數據發送（Flush & Remote Write）

- vmagent 會根據 `-remoteWrite.queues` 配置的 **寫入佇列數量**，並行發送數據到遠端存儲。
- 透過 **速率限制（Rate Limiting）**（`-remoteWrite.rateLimit`）防止存儲過載。
- 若遠端存儲回應 **409 或 400 錯誤**（如請求衝突或格式錯誤），vmagent 會 **跳過該數據塊**。

- 其他錯誤則採用 **指數退避機制（Exponential Backoff）** 重試。

總結

vmagent 透過 **並發控制、數據標籤處理、去重、聚合、分片、複製與快取**，確保監控數據的 **高效傳輸與存儲**。這些機制大幅降低網路流量、存儲需求，同時保持數據的完整性與可用性。

這使得 vmagent 成為 **高效能、低資源佔用** 的監控指標收集代理，適用於高並發、大規模監控環境。

vmagent如何快速收集和轉發Metri...
PDF

參考"How vmagent Collects and Ships Metrics Fast with Aggregation, Deduplication, and More.pdf"、"vmagent如何快速收集和轉發Metrics - charlieroro - 博客園.pdf"，以繁體中文詳細說明

vmagent 快速收集與傳輸 Metrics 的運作原理

參考資料來源：

1. **《How vmagent Collects and Ships Metrics Fast with Aggregation, Deduplication, and More》**
2. **《vmagent如何快速收集和轉發Metrics - charlieroro - 博客園》**

---

1. vmagent 簡介

VictoriaMetrics Agent（**vmagent**）是一款輕量級的指標數據收集代理，可從不同來源擷取數據，並在發送前進行**過濾、降維、聚合、去重、重新標籤（Relabeling）** 等處理，以減少數據量，提高傳輸效率。其主要特點包括：

- **支持 HTTP API 及 Scrape（拉取）方式接收數據**
- **數據處理（Relabeling、去重、降維、聚合）**
- **高效存儲數據，並發送到遠端存儲**
- **支持 Prometheus Remote Write 及 VictoriaMetrics Remote Write 協議**
- **內建高效快取與佇列（Memory Queue & Persistent Queue）**，確保高吞吐量及可靠性

---

2. vmagent 如何收集數據

vmagent 可以透過 **HTTP API 推送數據** 或 **Scrape 指標數據** 來獲取指標。

2.1 HTTP API 方式

- vmagent 提供 **HTTP API** 以接受 Prometheus 兼容的指標數據。
- 支援 **附加標籤（Extra Labels）**，例如：

```
http://vmagent-host:8429/api/v1/write?
extra_label=foo=bar&extra_label=baz=aaa
```

- **並發限制（Concurrency Limiter）**：
  - 預設最多允許 **`2 × CPU 核心數`** 的請求 (**`-maxConcurrentInserts`**)。
  - 若請求超過 **`1 分鐘（-insert.maxQueueDuration）`**，vmagent 會回傳 **503 錯誤**。

2.2 Scrape 指標

- **類似 Prometheus 的 Pull 模式**，可定期抓取目標指標。
- **配置示例**：

```yaml
global:
  scrape_interval: 10s
scrape_configs:
  - job_name: "example"
    scrape_interval: 5m
    scrape_timeout: 30s
```

- **數據大小限制**：
  - 預設最大抓取回應大小為 `16MB`（`-promscrape.maxScrapeSize`）。
  - 若 `scrape_interval` 小於 `scrape_timeout`，則 `scrape_timeout` 優先。

---

3. 數據預處理：Relabeling & 降維

3.1 Relabeling（重新標籤）

- 用於修改、篩選或刪除標籤，以降低指標數據量。
- **示例**：將 `job=my-app-*` 但 `env!=dev` 的 `foo` 標籤改為 `bar`：

```yaml
metric_relabel_configs:
  - source_labels: [foo]
    target_label: bar
    regex: "my-app-.*"
    action: replace
```

3.2 降維（Reducing Cardinality）

- **限制高標籤基數（High Cardinality）**，減少存儲壓力。
- 設定 `-remoteWrite.maxHourlySeries` 與 `-remoteWrite.maxDailySeries`，超過則丟棄。

---

4. 去重（Deduplication）與流聚合（Stream Aggregation）

4.1 去重（Deduplication）

- vmagent 會 **在一定時間窗口內刪除重複數據**，只保留最新或最大值。
- **配置方式**：

```
-streamAggr.dedupInterval=30s
```

  - 30s 內的重複數據僅保留最新的數據點。

4.2 流聚合（Stream Aggregation）

- **聚合類似指標**，以減少存儲佔用量，提高查詢效率。
- **示例**：將 `_total` 結尾的指標數據每 `5 分鐘` 聚合一次：

```
- match: '{__name__=~".+_total"}'
  interval: 5m
  outputs: [total]
```

- **去重 + 聚合時的規則**：
  - **流聚合間隔須為去重間隔的倍數**（如 `dedup_interval = 30s`，則 `aggregation_interval = 1m`）。

---

5. 分片（Sharding）與複製（Replication）

5.1 複製（Replication）

- 若有多個遠端存儲，vmagent **會將相同數據發送至每個存儲**：

```
-remoteWrite.url=http://storage1,http://storage2
```

5.2 分片（Sharding）

- vmagent **將數據均勻分配到不同存儲**，避免負載不均：

```
-remoteWrite.shardByURL=true
```

- **透過 Hash 計算分片**，可指定：

```
-remoteWrite.shardByURL.labels=instance,job
```

---

6. vmagent 內建快取與數據佇列

6.1 內存快取（Memory Queue）

- **快取數據，避免磁碟 I/O 開銷**
- **若 5s 內無法處理，則寫入磁碟快取**
- **記憶體佔用限制**：
  - `-memory.allowedPercent=60%`
  - `-memory.allowedBytes=2GB`

6.2 磁碟快取（File-based Queue）

- **數據持久化，避免數據丟失**
- **預設最大佔用空間：無限制**（可調整 `-remoteWrite.maxDiskUsagePerURL`）。
- **快取數據存放在 `/vmagent-remotewrite-data`**。

---

7. 最終數據發送

7.1 傳輸與速率限制

- vmagent **從快取佇列讀取數據並傳輸至遠端存儲**。
- **控制發送速率**（預設無限制）：

```
-remoteWrite.rateLimit=10MB/s
```

7.2 錯誤處理

- **若發送失敗**：
  - `409/400` 狀態碼：直接丟棄數據。
  - 其他錯誤：**使用指數退避機制（Exponential Backoff）進行重試**。

---

8. 總結

功能	描述
數據收集	HTTP API / Scrape
預處理	Relabeling、降維
去重	移除重複數據
流聚合	降低數據佔用
分片 & 複製	提高存儲可用性
快取機制	內存 & 磁碟快取
最終發送	限速控制 & 錯誤重試

vmagent 透過 **數據壓縮、流聚合、去重、分片與快取機制**，大幅減少網路流量與存儲開銷，是 **高效能、低資源消耗** 的指標收集代理，適用於大規模監控環境。