

# vmalert 503 錯誤處理



vmalert log出現這個，我找不到處理方法：  
VictoriaMetrics/app/vmalert/remotewrite/client.go:256  
attempt 1 to send request failed: unexpected response  
code 503



這個 `vmalert` 錯誤訊息：

```
VictoriaMetrics/app/vmalert/remotewrite/client.go:256  
attempt 1 to send request failed: unexpected response code 503
```

代表的是：**vmalert** 嘗試將 **alert** 資料透過 **Remote Write** 傳送到 **backend**（通常是 **VictoriaMetrics** 或其他 **remote\_write endpoint**）時，收到了 **HTTP 503** 錯誤碼，也就是 **服務不可用**（Service Unavailable）。

## ✅ 問題可能的原因與排查建議：

以下是最常見的幾個原因，並附上具體排查建議：

### 1 Remote Write 的目標（通常是 VictoriaMetrics）當下無法處理請求

- **原因：** VM backend server 短暫掛掉、過載、重新啟動中、或因資源（如 CPU、記憶體）爆滿無法服務。
- **建議：**

- 查看 remote write 接收端的 logs (VictoriaMetrics 或 vmagent) 是否有異常。
- 檢查它是否在重啟，或正在 OOM。
- ``kubectl get pods`` + ``kubectl logs <pod>`` 若是在 K8s 上。
- VM frontend/backend (例如 ``vminsert``) 是否正在滾動更新或讀寫負載過高。

---

## 2 網路問題

- **原因：** vmalert 到 remote\_write endpoint 中間有 network issue，例如 DNS、service name 錯誤、port 沒開、service 被 termination。
- **建議：**
  - 在 vmalert container 裡面執行 ``curl <remote_write_url>`` 看看能不能通。
  - 檢查 ``/etc/hosts`` 是否解析錯誤。
  - 確保 remote\_write URL 是正確的，例如  
``http://vminsert.default.svc.cluster.local:8480/insert/0/prometheus``

---

## 3 Remote write URL 設錯或寫法錯

- **原因：** remote write 設定不完整或拼錯。
- **建議：**
  - 檢查你的 vmalert flags (``-remoteWrite.url=...``) 有沒有設錯。
  - 確保 target 是 vminsert，而不是 vmselect。
  - 範例正確寫法：

```
-remoteWrite.url=http://vminsert.default.svc:8480/insert/0/prometheus
```

#### 4 vmalet 設定 alert 規則時，每次都送出過多 alert 導致壓力過大

- **原因：** 某些 alert expression 結果太多 series，一次送出太多資料，導致 remote write 無法處理。
- **建議：**
  - 確認每個 rule 的 expression 查詢結果是否太多。
  - 減少 rule 數量、加上 label 過濾器。

#### 5 VictoriaMetrics 集群 HA 配置錯誤或 vminsert 沒設好

- **建議：**
  - 如果你有 HA setup，確認 `vminsert` 有至少一個 pod 正常，並且有註冊到 service。
  - 可試用 `kubectl port-forward` 到 vminsert 測試 remote\_write 接口是否真的活著。

### ✓ 該怎麼做：

你可以照以下步驟依序檢查：

```
# Step 1: 驗證 remote_write 目標可以連線
kubectl exec -it <vmalet pod> -- curl -v
http://vminsert.default.svc:8480/insert/0/prometheus

# Step 2: 檢查 vmalet 啟動參數
kubectl get pod <vmalet pod> -o jsonpath='{.spec.containers[*].args}'

# Step 3: 查看 vminsert 的 log 是否報錯
kubectl logs <vminsert pod>
```

```
# Step 4: 確認 VM 是否處於高負載 (可用 Grafana / kubectl top)
kubectl top pod <vminsert pod>
```

## 若想自動重試

vmaalert 預設會自動 retry 幾次，但如果持續出現 503，建議你：

- 確保 remote write 接收方穩定。
- 或考慮在 remote write 前端加上 queue 機制（例如透過 vmagent 做緩衝）。

如果你願意貼出你的 vmaalert 啟動參數（或 helm values）、remote write 設定或架構圖，我可以幫你更具體分析。

是否要幫你寫一段 vmaalert + VictoriaMetrics 的 Remote Write 正確設定 YAML 或 flags？



```
vmaalert log出現這個，我找不到處理方法：
2025-08-01T16:56:03.749+0800
warn
VictoriaMetrics/app/ vmaalert/remotewrite/client.go:
256
attempt 1 to send
request failed: unexpected response code 503 for http://
vm-vminsert.ifond.edg.f18.tsmc
.com/insert/0/prometheus/api/v1 /write. Response body
"upstream connect error or disconnect/reset before
```

```
headers. reset reason: connection termination"
(retriable: true)2025-08-01
```



錯誤重點：

```
unexpected response code 503
upstream connect error or disconnect/reset before headers. reset reason:
connection termination
```

這代表 **vmalet** → **vminsert** 的 HTTP 連線在尚未收到回應前就被斷開，導致 vmalet remote\_write 重試。

## 可能原因與解決方向

### 1. vminsert 或上游 Service 沒有可用 Pod

- 503 常見於 Service 找不到後端 Pod。
- 檢查：

```
kubectl get pods -n <namespace> -l app=vminsert
kubectl describe svc vm-vminsert
```

若 Pod 狀態不是 `Running` 或 `Ready`，先排查 vminsert 的 logs。

### 2. Service Mesh 或 Proxy (Envoy/Istio) 重置連線

- ``reset reason: connection termination`` 典型是 Envoy (Istio/Linkerd) 或 API Gateway reset。
  - 檢查：
    - 是否有 Istio sidecar, 查看 ``istio-proxy`` logs。
    - 確認 ``DestinationRule`` 或 ``VirtualService`` 沒有錯誤 timeout。
- 

### 3. vminsert 負載過高 / OOM

- 當 vminsert 負載滿時會拒絕請求。
- 檢查：

```
kubectl top pod -n <namespace> | grep vminsert  
kubectl logs <vminsert-pod>
```

若出現 ``cannot handle request`` 或 ``too many open connections``, 需調整 ``replicas`` 或 ``-maxConcurrentInserts``。

---

### 4. URL 或路由錯誤

- 你的 URL 是：

```
http://vm-vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus/api/v1/write
```

確認 ``/insert/0/prometheus`` 是否正確。一般 remote\_write 只需：

```
http://<vminsert>/insert/0/prometheus
```

或 ``/api/v1/write``, 但不能兩者混合。

---

## 5. 網路連線不穩定

- 若 vmalet 與 vminsert 跨區域、跨 VPC，可能因防火牆或 LB reset。
- 測試：

```
kubectl exec -it <vmalet-pod> -- curl -v http://vm-  
vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus
```

---

## 建議排查步驟

### 1. 測試連線

```
kubectl exec -it <vmalet-pod> -- curl -v http://vm-  
vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus/api/v1/write
```

### 2. 檢查 vminsert 狀態

```
kubectl get pods -n <namespace> -l app=vminsert  
kubectl logs <vminsert-pod>
```

### 3. 確認 remote\_write URL

在 vmalet 參數中用：

```
-remoteWrite.url=http://vm-  
vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus
```

### 4. 監控負載

用 Grafana 或 `kubectl top`` 觀察 vminsert 的 CPU/MEM 是否達瓶頸。

是否需要我幫你產生一個 **vmaalert + vminsert** 的正確 **remote\_write** 設定範例 (Helm YAML)，並加上探針 (readiness/liveness probe) 避免 503？



這是出現在vmaalert寫入alert到vminsert時出現的log，503，可是vminsert 有把alert寫進vmstorage



非常關鍵的補充資訊是這句：

“**vminsert 有把 alert 寫進 vmstorage**”

👉 代表 **實際上資料有成功寫入**，但 **vmaalert 收到的是 503**！”

這裡的重點在於：「**vmaalert 收到 503**，但資料**實際上有進去**」，那我們就可以推論出問題比較像是：

✅ 這是「**寫入後才被中斷連線**」的情況（寫入成功但 **response 掛掉**）

錯誤訊息：

```
Response body "upstream connect error or disconnect/reset before headers. reset reason: connection termination"
```



這通常出現在下列場景：

## 1. 上游有 Proxy / Gateway（如 Envoy/Istio）提前 reset 了 response

- 資料寫入成功，但 vmalet 還沒收到 HTTP response，就被 proxy reset。
- 很常見於 Envoy / Istio 認為 timeout（即便後端完成寫入），就中斷連線。

### 你可以檢查：

- vmalet 與 vminsert 間是否有 API Gateway / Service Mesh
- 是否有設定 `timeout: 1s`、`idle_timeout`、`max_connection_duration` 等參數過短
- 檢查 Envoy 或 istio-proxy 的 log 是否有 `reset reason`

---

## 2. vminsert 在回應時，連線被中斷（例如 CPU 爆或寫回壓）

- 寫入已完成，但 response 無法送出（TCP write buffer 滿、或 pod 快被 evict）。
- 若此情況偶發且 vmalet 最後資料有進 VM storage，其實不致命，但會導致：
  - vmalet 重試（產生 duplicate）
  - log 報 warning（非 error）

### 確認重點：

- 若 vmalet 重試後資料有進 VM storage → 屬於 可容忍的 **retriable warning**
- 建議：
  - 加大 **timeout**（參考下一段）
  - 降低每次寫入資料量
  - 增加 vminsert 的 pod 數量或資源限制

## ✅ 建議調整（如果你想減少這種 warning）

### 🔧 vmaalert 加長 remote\_write timeout

vmaalert 沒有明確 `timeout` flag，但你可以在 proxy (Envoy/Istio) 層調整：

```
timeout: 10s
idle_timeout: 15s
per_try_timeout: 5s
```

### ⚙️ vmaalert 調整批次大小（避免寫入壓力過大）

可考慮升級版本或拆分 alert 規則，減少一次觸發多組 metrics。

### 🔧 Envoy / Load Balancer 側調整 timeout 設定

- 若有使用 GKE ingress、F5、Nginx LB、Istio 等，請調大 timeout。
- 否則寫入完成但 response 遲了幾百 ms 就會被 drop。

## ✅ 總結

現象	解釋	建議
vmaalert log 出現 503	RemoteWrite 寫入後，response 被提前 reset	✅ 檢查 proxy / timeout 設定
alert 有成功進 VM storage	表示資料已寫入，只是 response 掛了	✅ 可容忍，建議 fine-tune
發生頻率高	代表 vminsert 負載或中間 proxy 設定需調整	✅ 增加資源、調整網路層 timeout
想降低 warning	調整 mesh timeout、寫入粒度、加重試邏輯	✅ 配合 tuning

---

若你能提供：

- vmaalert 與 vminsert 間是否有 service mesh / proxy
- 目前的 `vmaalert` flags 和 `remote\_write.url`
- 使用的是 K8s 還是 VM 部署

我可以更具體給你修改設定或 YAML 範例。是否需要我幫你 draft 一份？



使用的是 K8s



了解你使用的是 **Kubernetes** 部署的 `vmaalert` → `vminsert` → `vmstorage` 架構，並遇到：

```
vmaalert remote_write 到 vminsert 時出現 503  
但 alert 最後有寫進 vmstorage
```

這代表「寫入成功但 response 被中斷」，很大機率是 K8s 中的 **Service Mesh**、**Ingress**、**Sidecar Proxy**、**LB timeout** 問題。

---

✅ 為什麼會這樣？

在 Kubernetes 中常見這些情況會導致：

類型	說明
<b>Istio/Linkerd proxy reset</b>	Sidecar proxy 若 timeout 太短，會中斷寫入後還沒完成 response 的連線
<b>K8s Service Endpoint 更新中</b>	Pod scaling / readiness probe 變動時 Service 短暫無可用 backend
<b>Ingress Controller reset</b>	Nginx/Traefik Ingress 設定 timeout 太短
<b>vminsert 負載高</b>	Response 遲緩被前面的 LB/reset 給 kill 掉

## 你可以先這樣排查：

### 1. vmalert ↔ vminsert 之間是否有 Istio/Envoy/Ingress？

```
kubectl get pods -n <namespace> | grep vmalert
kubectl get pods -n <namespace> | grep vminsert
kubectl get svc -n <namespace>
kubectl describe svc vm-vminsert
```

#### 確認是否：

- 有 Istio sidecar (pod 內有 `istio-proxy` container)
- 有 Ingress Controller 介入 (通常 vmalert 走 external DNS)
- 使用 ClusterIP 還是 LoadBalancer 或 ExternalName？

## 2. 檢查是否 response 被 timeout / reset

觀察 vminsert 的 logs：

```
kubect1 logs <vminsert-pod>
```

搜尋關鍵字：`timeout`、`too many open connections`、`broken pipe`、`connection reset`、`rate limit`

### ✅ 建議調整方式（避免 warning）

#### 🔧 A. 改成直接用 ClusterIP 與內部 DNS 通訊

避免透過 Ingress 或外部 URL，例如：

```
http://vm-vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus/api/v1/write
```

⚠️ 這很可能經過外部 LB、DNS 解析甚至 F5、防火牆 → 延遲高、容易 reset

✅ 改成內部 K8s ClusterIP:

```
-remoteWrite.url=http://vm-  
vminsert.monitoring.svc.cluster.local:8480/insert/0/prometheus
```

這樣會大幅減少 reset 機率。

#### 🔧 B. 若有 Istio，設定 DestinationRule + timeout

```
apiVersion: networking.istio.io/v1beta1  
kind: DestinationRule  
metadata:
```

```

name: vminsert-dr
spec:
  host: vm-vminsert.monitoring.svc.cluster.local
  trafficPolicy:
    connectionPool:
      http:
        http1MaxPendingRequests: 1000
        maxRequestsPerConnection: 100
    outlierDetection:
      consecutive5xxErrors: 10
    tls:
      mode: DISABLE

apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: vminsert-vs
spec:
  hosts:
    - vm-vminsert.monitoring.svc.cluster.local
  http:
    - route:
        - destination:
            host: vm-vminsert.monitoring.svc.cluster.local
            port:
              number: 8480
      timeout: 15s

```

## C. 若用 NGINX Ingress, 設定 proxy timeout

```

nginx.ingress.kubernetes.io/proxy-read-timeout: "20"
nginx.ingress.kubernetes.io/proxy-send-timeout: "20"
nginx.ingress.kubernetes.io/proxy-connect-timeout: "5"

```

## ✅ 範例：vmaalert 正確 remote\_write 設定

你可以參考以下部署參數，改用內部 svc URL：

```
args:
  - '-remoteWrite.url=http://vm-
vminsert.monitoring.svc.cluster.local:8480/insert/0/prometheus'
  - '-
datasource.url=http://vmselect.monitoring.svc.cluster.local:8481/select/0/prometheus'
```

## 🔧 是否要我幫你產出一份完整 Helm values.yaml / Deployment YAML 來替換？

只要你提供以下資訊即可幫你 draft：

1. `vmaalert`、`vminsert` 的 namespace 與 service 名稱
2. 有沒有用 Istio？或 NGINX Ingress？
3. 使用 Helm 還是直接寫 YAML？

我可以根據你現有環境，幫你產一份調整過的設定。需要的話請說一聲。



vminsert有掛VirtualService、Service，vmaalert的-remoteWrite.url是vminsert的VirtualService



感謝補充，根據你提供的資訊：

“✅ `vminsert` 掛的是 *VirtualService + Service*

✅ `vmalert -remoteWrite.url` 指向的是 ***vminsert* 的 *VirtualService***”

這就非常清楚了，你遇到的 503 錯誤原因 **幾乎可以確定** 是以下情況之一：

## 💣 主要問題：Istio 的 **VirtualService / Envoy proxy timeout、連線重置**

錯誤訊息：

```
reset reason: connection termination
```

這是 Istio / Envoy 最典型的 reset log，代表 **連線還沒收到 response 就被 proxy 重置了**，即便後端 `vminsert` 已成功寫入。

## ✅ 解法一：設定 **VirtualService** 的 **timeout**

預設情況下，Istio 會使用很短的 timeout（15s 或更短），如果 vminsert 處理稍久，response 還沒來就被斷了。

你可以在 vminsert 的 VirtualService 加上 `timeout: 30s`：

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: vminsert
  namespace: your-namespace
```



```
spec:
  hosts:
    - vm-vminsert.ifond.edg.f18.tsmc.com
  http:
    - route:
        - destination:
            host: vm-vminsert
            port:
              number: 8480
            timeout: 30s
```

這會讓 `vmalert` 有更多時間等待 response，不會因為傳輸太慢就被 reset。

## ✅ 解法二：DestinationRule 中設定 connection pool

為了減少 Envoy 覺得連線過舊或壅塞而斷線，可以設定：

```
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: vminsert
  namespace: your-namespace
spec:
  host: vm-vminsert
  trafficPolicy:
    connectionPool:
      http:
        http1MaxPendingRequests: 1000
        maxRequestsPerConnection: 100
    outlierDetection:
      consecutive5xxErrors: 10
  tls:
    mode: DISABLE
```

## ✅ 解法三：vmalet 改為用 K8s 內部 ClusterIP（跳過 Istio）

若允許，建議最穩定的方法是避開 Istio，vmalet 改用 K8s Service ClusterIP URL：

```
-remoteWrite.url=http://vm-  
vminsert.monitoring.svc.cluster.local:8480/insert/0/prometheus
```

這樣：

- 走的是 kube-proxy，不會被 Istio 的 proxy reset
- 更適合內部通信的高頻寫入（如 alert 推送）

如果你們團隊允許，可以考慮這樣做。

---

## ✅ 解法四（可選）：確認 Istio proxy 沒限制 body 太大

部分 `remote\_write` payload 若太大，也可能被拒絕，可以加上這些 annotations：

```
annotations:  
  proxy-body-size: "20m"  
  nginx.ingress.kubernetes.io/proxy-body-size: "20m"  
  nginx.ingress.kubernetes.io/proxy-read-timeout: "30"  
  nginx.ingress.kubernetes.io/proxy-send-timeout: "30"
```

---

## ✅ 總結與建議優先順序

優先順序	建議內容	備註
★ 首選	VirtualService 加上 <code>timeout: 30s</code>	大多能解決
★ 次選	改用 <code>ClusterIP</code> Service 直接寫入	避開 proxy reset
✓ 補充	DestinationRule 加大 connection pool	降低失敗率
🔧 檢查	vminsert 是否有 Envoy sidecar ? 負載如何 ?	OOM or 延遲也會觸發 reset

需要我幫你根據你目前的 `VirtualService` **YAML**，加上 **timeout** 調整嗎？

你可以貼出現有 YAML，我來幫你修改成合適版本。



根據目前的 VirtualService YAML，加上 timeout 調整



請看以下是你現有的 `VirtualService` 範本，並加上 `timeout: 30s`（你可以依實際值調整，例如 60s 也可）：

✓ 修改前（你可能原本是這樣）：

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
```

```
name: vminsert
namespace: monitoring
spec:
  hosts:
    - vm-vminsert.ifond.edg.f18.tsmc.com
  http:
    - route:
        destination:
          host: vm-vminsert
          port:
            number: 8480
```

✅ 修改後：加上 `timeout: 30s`

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: vminsert
  namespace: monitoring
spec:
  hosts:
    - vm-vminsert.ifond.edg.f18.tsmc.com
  http:
    - route:
        destination:
          host: vm-vminsert
          port:
            number: 8480
      timeout: 30s
```

🔄 可選：加上 `retries`（如仍有偶發 503）

```
- route:
  - destination:
      host: vm-vminsert
      port:
        number: 8480
    timeout: 30s
  retries:
    attempts: 3
    perTryTimeout: 10s
```

## 修改方式

1. 儲存這個 YAML 檔案，例如 ``vminsert-vs-patch.yaml``

2. 套用：

```
kubectl apply -f vminsert-vs-patch.yaml
```

3. 驗證是否更新成功：

```
kubectl get virtualservice vminsert -n monitoring -o yaml
```

如你需要我幫忙也補上 ``DestinationRule`` 調整，或是你用了 Gateway、TLS mode 想設定，我也可以協助。是否需要一併幫你補？



也補上 DestinationRule 調整，並說明



好的，以下是根據你的情境（K8s 上的 vminsert + 掛有 Istio VirtualService），搭配的  
`DestinationRule` 建議設定，並附上用途說明。

## ✅ 建議 DestinationRule 設定 (for vminsert)

```
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: vminsert
  namespace: monitoring
spec:
  host: vm-vminsert
  trafficPolicy:
    connectionPool:
      http:
        http1MaxPendingRequests: 1000      # 增加可排隊的請求數，預設小（如1024）
        maxRequestsPerConnection: 100      # 降低 keep-alive reuse 導致重複
    reset
    outlierDetection:
      consecutive5xxErrors: 10              # 如果連續錯誤則暫時避開該實例
      interval: 5s
      baseEjectionTime: 30s
      maxEjectionPercent: 100
    loadBalancer:
      simple: ROUND_ROBIN                  # 使用預設輪詢機制
    tls:
      mode: DISABLE                        # 明確指定關閉 TLS（你使用 http://）
```

## 說明

區塊	功能	解釋
<code>`connectionPool`</code>	控制 Envoy 與目標 pod 的 HTTP 連線行為	增加併發 / 避免重複使用太久的連線導致 reset
<code>`outlierDetection`</code>	自動避開異常實例	假如某個 <code>`vminsert`</code> pod 常常回錯，就暫時不送它
<code>`loadBalancer`</code>	載入平衡策略	預設 <code>`ROUND_ROBIN`</code> 即可
<code>`tls.mode: DISABLE`</code>	明確關閉 TLS	若你走 <code>`http://...`</code> ，建議設為 <code>`DISABLE`</code> 避免 TLS handshake 問題

## 套用方式

1. 儲存為 ``vminsert-destinationrule.yaml``

2. 套用：

```
kubectl apply -f vminsert-destinationrule.yaml
```

3. 驗證：

```
kubectl get destinationrule vminsert -n monitoring -o yaml
```

## 搭配 VirtualService 使用（完整建議）

你現在的 ``remoteWrite.url`` 指向：

```
http://vm-vminsert.ifond.edg.f18.tsmc.com/insert/0/prometheus
```

因此你需要有：

1. **VirtualService** (指定 host 對應 svc 和 port, 並加上 timeout)
2. **DestinationRule** (幫你調整連線穩定性)

這樣就可以讓 `vmalert` 與 `vminsert` 之間的通訊更穩定, 避免寫入成功卻回傳 503 的情況持續發生。

如果你使用 Helm 管理這些設定, 我也可以幫你轉成 `values.yaml` 中的方式設定。需要的話請告訴我部署方式 (例如 istio base 是怎麼部署的?) 我可以配合調整。



```
Response body "upstream connect error or  
disconnect/reset before headers. reset reason:  
connection termination" (retriable:  
true)
```



這個訊息：

```
upstream connect error or disconnect/reset before headers. reset reason:  
connection termination (retriable: true)
```

是 **Istio / Envoy Proxy** 的標準錯誤, 意思是：



“連線建立後，還沒收到 HTTP 回應的 headers，就被上游（upstream）或中間的 Proxy/LB 強制關閉（reset reason: connection termination）。”

## 典型發生原因

### 1. Istio VirtualService/DestinationRule timeout 太短

- 後端 `vminsert` 寫入完成需要較久時間，但 proxy 等不到就斷線。

### 2. Envoy connection pool / keep-alive 連線回收

- 連線被過早關閉，下一次重用時就被 reset。

### 3. Pod scaling / readiness probe 切換

- Istio 在切換 upstream endpoint 時中斷現有連線。

### 4. LB / Gateway / 防火牆 session reset

- 中間層對長連線或大流量有保護策略。

## 建議處理步驟

### 1. VirtualService 加大 timeout

```
http:
- route:
  - destination:
      host: vm-vminsert
      port:
        number: 8480
      timeout: 30s
```

---

## 2. DestinationRule 調整連線池

```
trafficPolicy:
  connectionPool:
    http:
      http1MaxPendingRequests: 1000
      maxRequestsPerConnection: 100
  outlierDetection:
    consecutive5xxErrors: 10
```

---

## 3. 檢查 Istio Proxy Log

```
kubectl logs <vmalert-pod> -c istio-proxy
kubectl logs <vminsert-pod> -c istio-proxy
```

看是否有 `upstream\_reset\_before\_response\_started` 或 `connection\_termination`。

---

## 4. 考慮改用 K8s ClusterIP

如果允許，`vmalert` 直接對 `vm-vminsert.monitoring.svc.cluster.local:8480` 發送，跳過 Istio Gateway，可完全避開這類 reset。

---

如果你要的話，我可以幫你整理一份 **VirtualService + DestinationRule** 的完整 YAML，把 timeout、connection pool、重試策略一次加好，讓這種 reset 幾乎不再出現。  
你要我幫你整合出來嗎？

