在地端 K8s 的排障故事分享

玉山銀行 | 核心組副主任工程師 李啓維

我是誰

玉山銀行智能金融處核心組

- 照料所有 MLaaS k8s
- 部署維運的疑難雜症
- 展覽咖啡長板

演講經歷

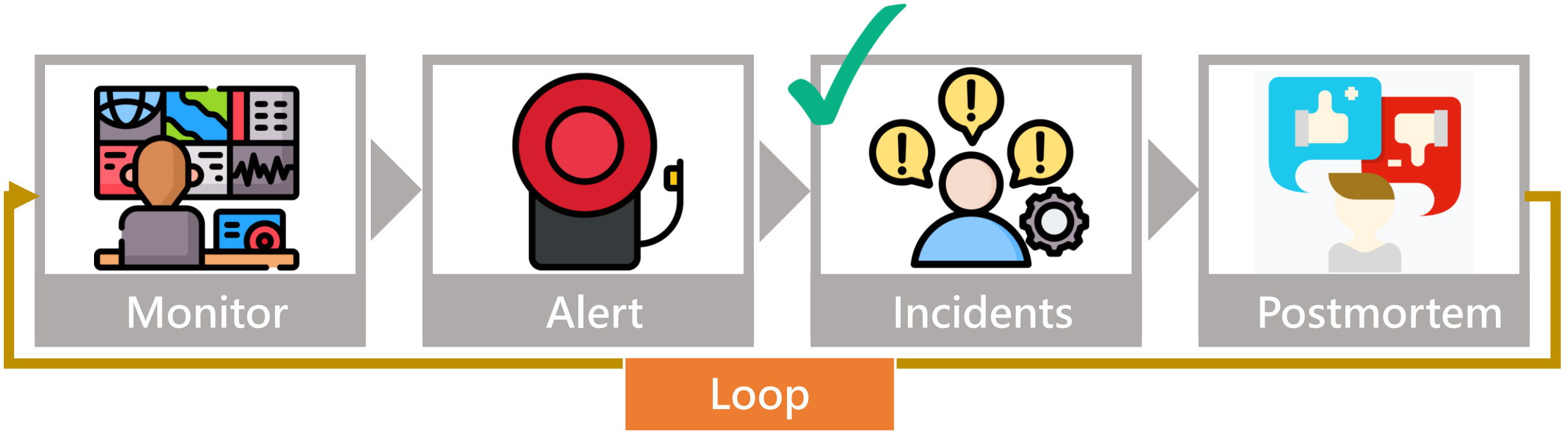
- 2023 k8s summit / CPU throttling
- 2024 hello devSecOps / 導入 OPA

sean22492249@gmail.com
https://kiwi-walk.com
M | https://sean22492249.medium.com/

今天要來跟大家分享
「當初笑不出來的故事」

要用地端的 k8s 嗎


資源
系統
服務
錯誤
這些
prod
事件
原因
無法
炸掉
導致
配置
staging
網絡
timeout
Gateway
可能
管理
分配
異常
例如
情況
服務器
過載
節點
blackboxprobefailed
rawdata
airflow
error
server
不足
失敗
顯示
表明
以下
回應
明顯
通常
不當
不可
整體
因為
CPU
node
聯通
有效
超時
具體
元件
cilium
worker
我們
上游
網站
普遍
表示
多種
存在
latency
ansible
Proxy
一系列
背後隱藏
出資源
監控
定性
系列
時間
能夠
穩定
延遲
響應
常見
造成
包括
以及
誤碼
Unavailable
request
memory
consumer
nodes
Production
現代分
流量
案例
普遍存在
feedbackAPI
並緩解
根本
這期間
連線
資源塞滿
管理策略
運營
範圍
兼容性
匣子
不斷
作業
發現
不穩
可靠性
稱之為
多服
影響
發生
不當會
高延遲
至關
當節點
多樣
例子
或耗
挑戰
可靠
啟動
Grafana
往往
remittance
總結
操作
頻繁
連接
有些
峰值
優化
明系統
增長
而言
notification
負荷
部署
不少
ArgoCD
服務無法
如何
務發生
黑匣子
connection
根本原因
環境
網絡拓
高高
然而
協助
容性
截圖
當服務
分析
服務連
出明顯
pressure
回顧
崩潰
本原
表現
測試
發送訊息
本身
潛在
變高時
盡時
相關
揭示
減少
展示
示例
兼容
策略
緩慢
實際
重要
什麼
Latency
ingress
apiserver
服務連接
圖像化
並解決
根本原因
通過
中斷
專案
deployment
提升
會導致
stack
分析
其他
下線
最常見
等待
一些
維護
網路
上述
下陷
系
暫時
一個
這類
Service
ssh
dag
k8s
Staging
大量
Timeout
狀態
function
服務過
播放
尤為
檢查
Argocd
健康
定位
Aliyu
qmech
Jaben
增加
POD
引起
不上
可以
黑匣
repo
應時
細心



不探討

- monitor 如何規劃
- alert/incidents 如何偵測
- 不會過於深入技術

要分享的

- **出事時，我自己的經驗整理**
- 配上血的故事
- 與許多的梗圖


出事了阿伯
參戰!

常見的事件處理 SOP

讓大家安心

救火

以後不會在犯了

列出事件發生

事件檢測

影響範圍

根本原因

學到的教訓

時間線

定期的回報

恢復

無責備的根本原因

事件處理時會遇到的鼓勵

又是一個美妙的工作日從修復系統開始。

什麼時候恢復

什麼時候能上板

你們有沒有在做事

那我們是不是會延誤計劃？

你們行嗎？感覺很常壞掉ㄟ

好像每個月都有固定的驚喜，是不是？

我們要怎麼跟客戶說明

系統怎麼一直壞掉

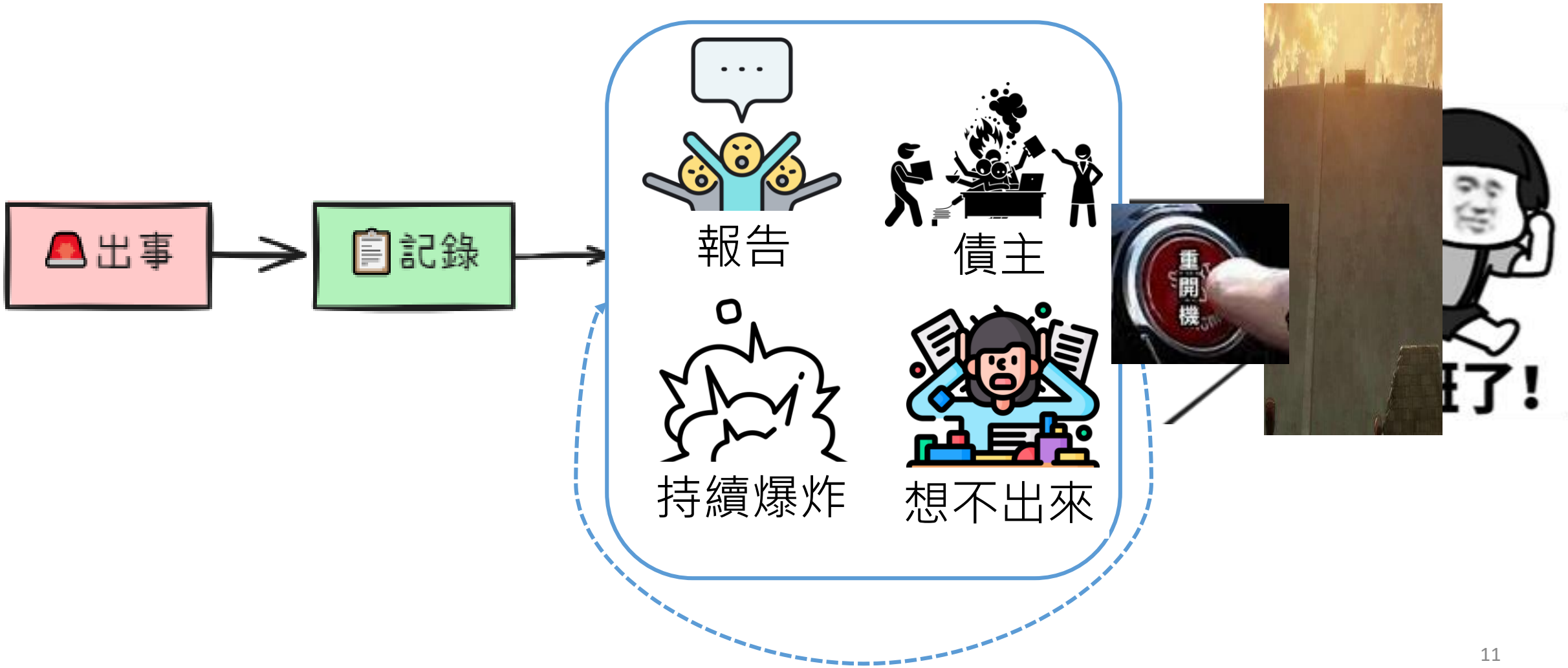
火真的好大

我們的系統是否有不太行？

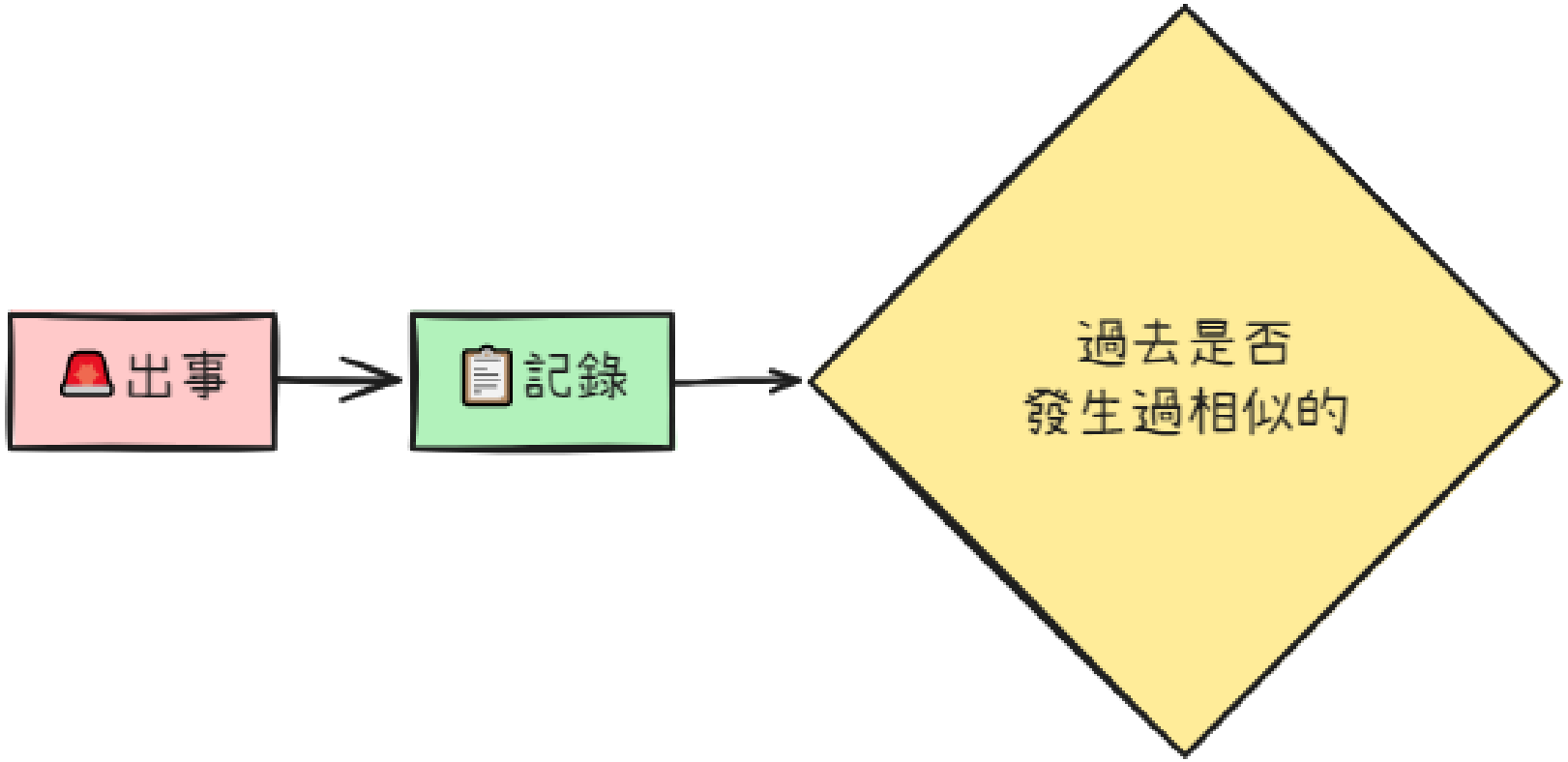
向高層報告

身為處理人的內心

實際上我們在(想)做的



第一步



事件➡記錄

優先確認資訊來源 & 記錄編號

- who, when, what

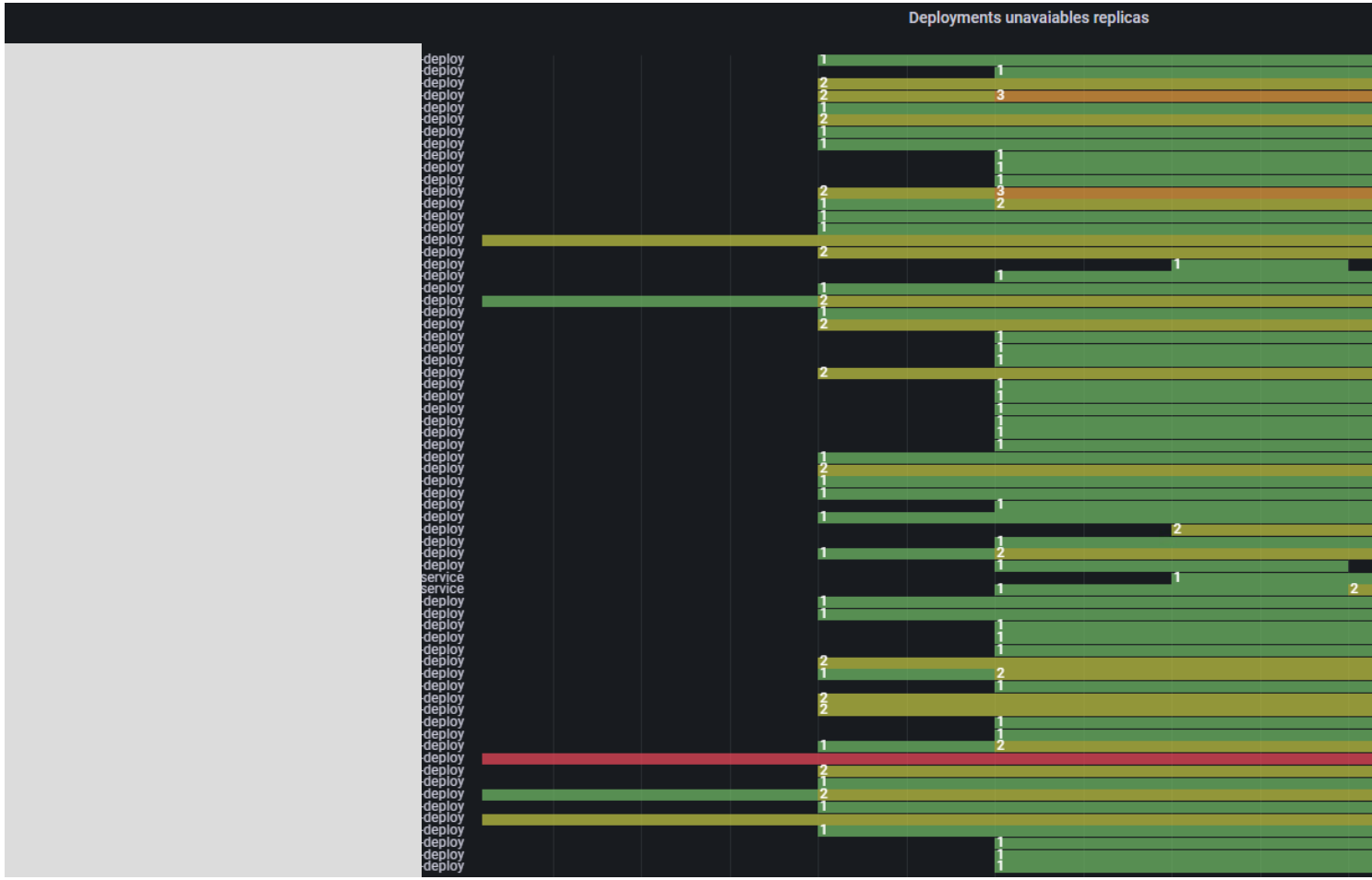
尋找過去的參考

事件關鍵字：

- CVE Patch
- container

歷史痛苦：

- 2023.12 發生過
- 同樣是 container 問題



回到那時候，來自 CVE 的更新

主旨: 通報單-(中)-GNU C函式庫存在緩衝區溢位漏洞(CVE-2023-4911)

各位好：

發佈 資安通報，請各位於回覆期限內回報處理方式，謝謝。

一、通報主旨：GNU C函式庫存在緩衝區溢位漏洞(CVE-2023-4911)

二、通報描述：

1. 風險等級：
2. 通報回覆期限： **(註：若無法完成影響範圍之評估，請務必於期**
3. 通報描述：
   - 擁有存取權限的攻擊者成功利用時可取得Root權限。
   - 細節請參考附件。

三、建議措施：請安排更新作業。

四、

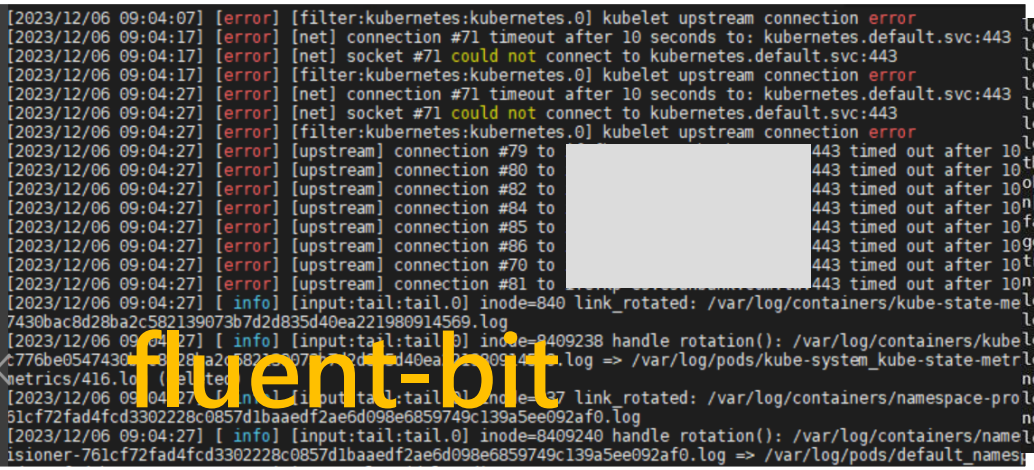
NetworkManager 有重啟

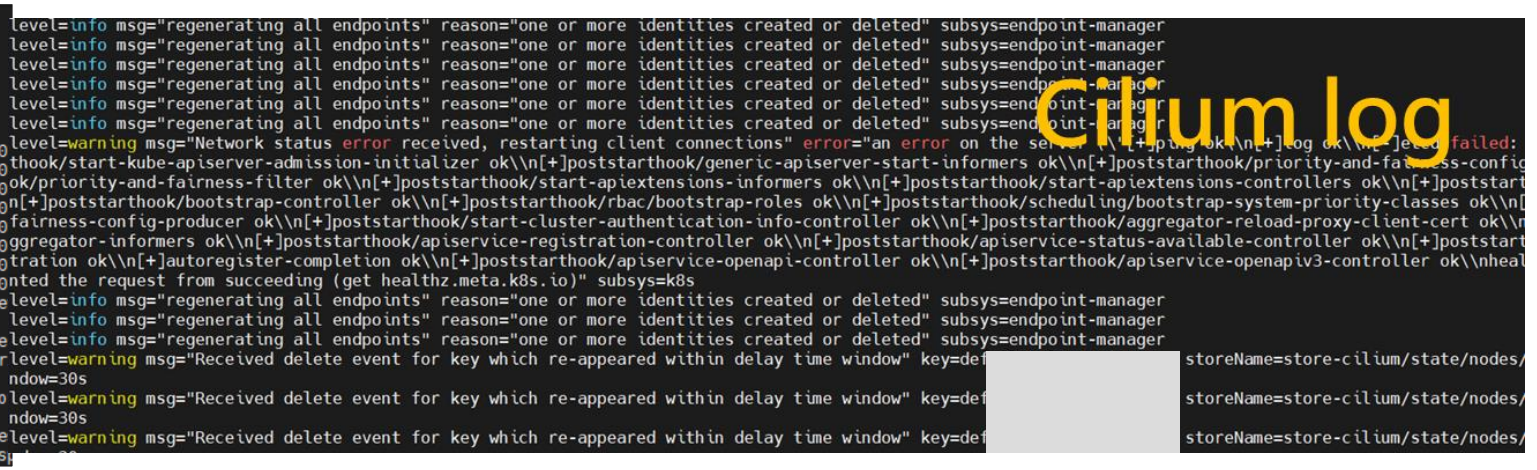
當時的出事

deployment 出現 unavailable

- 40+ | **10分鐘後**自動恢復
- 8+ | 無法恢復

部分節點發生了問題






```
[root@ ~]# kubectl -n argocd get po -owide
NAME                                                READY   STATUS             RESTARTS        AGE   IP              NODE   NOMINATED NODE   READINESS GATES
mlaas-argocd-application-controller-0               0/1     CrashLoopBackOff   9 (3m32s ago)   17m                            <none>           <none>
mlaas-argocd-applicationset-controller-7b66475c46-xm9tm   0/1     CrashLoopBackOff   7 (2m57s ago)   17m                            <none>           <none>
mlaas-argocd-notifications-controller-86c89d4d6-bnvk8     1/1     Running            0               17m                            <none>           <none>
mlaas-argocd-redis-bd44f8f56-7mxvf                  2/2     Running            0               17m                            <none>           <none>
mlaas-argocd-repo-server-f44f5598c-lmbmd            1/1     Running            0               17m                            <none>           <none>
mlaas-argocd-server-75cc89f64d-bs99j                0/1     CrashLoopBackOff   21 (3m14s ago)  54m                            <none>           <none>
[root@mlaas20webm01p ~]# kubectl -n argocd get po -owide
NAME                                                READY   STATUS    RESTARTS   AGE    IP              NODE   NOMINATED NODE   READINESS GATES
mlaas-argocd-application-controller-0               1/1     Running   0          113s                           <none>           <none>
mlaas-argocd-applicationset-controller-7b66475c46-tq867   1/1     Running   0          34s                            <none>           <none>
mlaas-argocd-notifications-controller-86c89d4d6-bnvk8     1/1     Running   0          20m                            <none>           <none>
mlaas-argocd-redis-bd44f8f56-7mxvf                  2/2     Running   0          20m                            <none>           <none>
mlaas-argocd-repo-server-f44f5598c-lmbmd            1/1     Running   0          20m                            <none>           <none>
mlaas-argocd-server-75cc89f64d-btw7h                1/1     Running   0          34s                            <none>           <none>
[root@mlaas20webm01p ~]#
[4] 0:ssh*Z                                                                                                                  "mlaasasb1p" 16:52 06-Dec-23
```

service recover when pods moved to other nodes

Cilium 只會在一開始修改 iptables

ff28c83 cilium / pkg / datapath / iptables / iptables.go

Code Blame 1858 lines (1648 loc) · 58.8 KB

```
276  type params struct {
288  func newIptablesManager(p params) *Manager {
289          iptMgr := &Manager{
290                  logger:        p.Logger,
291                  modulesMgr:    p.ModulesMgr,
292                  cfg:           p.Cfg,
293                  sharedCfg:     p.SharedCfg,
294                  haveIp6tables: true,
295          }
296
297          p.Lifecycle.Append(iptMgr)
298
299          return iptMgr
300  }
301
302  // Start initializes the iptables manager and checks for iptables kernel modules availability.
303  func (m *Manager) Start(ctx hive.HookContext) error {
304          if os.Getenv("CILIUM_PREPEND_IPTABLES_CHAIN") != "" {
305                  m.logger.Warning("CILIUM_PREPEND_IPTABLES_CHAIN env var has been deprecated. Please use
306                          "env var or '--prepend-iptables-chains' command line flag instead")
```

- To my surprise, cilium doesn't periodically synchronize those rules like kube-proxy. If you somehow remove a rule in its custom chain, you have to add it back manually or restart cilium-agent. Is this a bug or feature ?

規則：

只會在 onStart, onStop

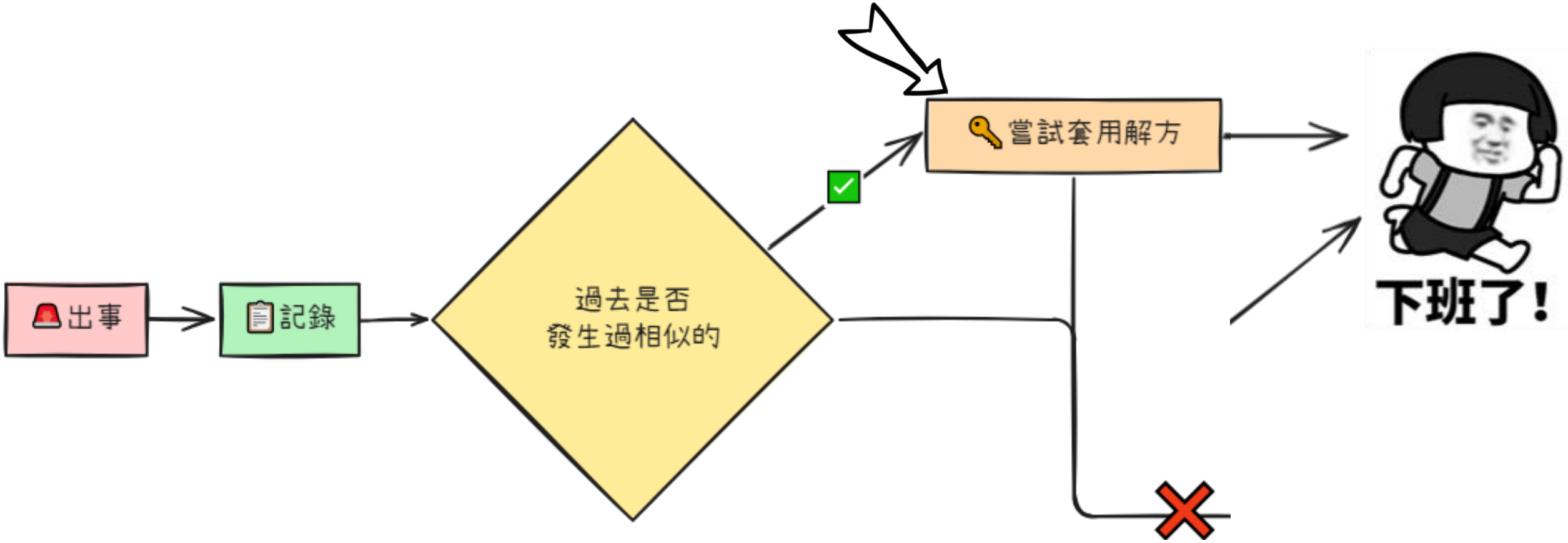
導致：

流量卡住出不去

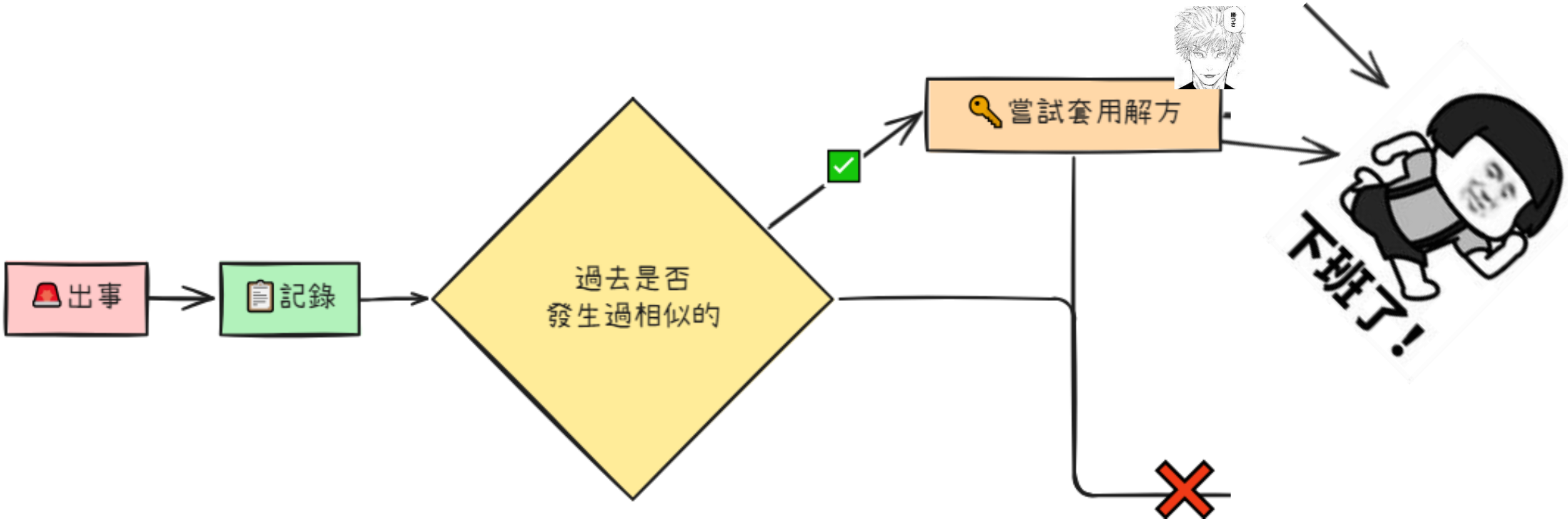
Ref: CNI Benchmark: Understanding Cilium Network Performance

結合前面事件的參考

- 知道「重啟與 container network 有關的即可」



記錄的重要性在於：不要考驗自己的記憶力



只是如果

- 解方不管用的話呢？
- 或是沒有可以參考的呢？

另一個事件：response failed

嘗試了重啟 cilium

事件影響範圍：

apiserver, kube-scheduler, cilium-operator, cilium-agent 發生服務重啟

幾乎每個 node 的 cilium pod 都有重啟，其中 更陷入無限的重啟，而 則發生無法新增 pod 的情形

導致當天 airflow 的 dag 無法順利執行。

當時擔心損害擴散，所以將 w04p cordon，拒絕新的 pod 部署

先前另一件事情 202 服務不少 deployment 下陷

在處理完成觀察後，未將 uncordon

因此在 prod cluster 有 的 node 無法部署下，使得 airflow 的 dag，因 node 限制 110 個 pods 運行，kube

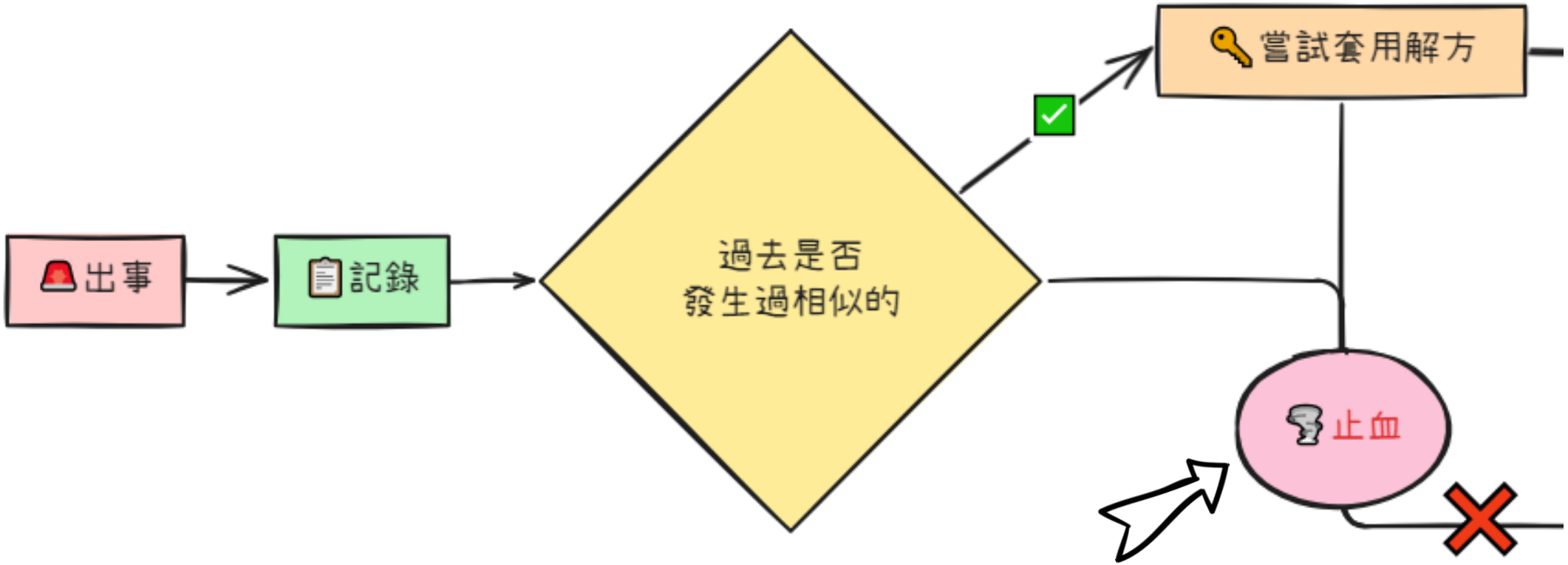
後來有將 uncordon，一切恢復正常

跟隨先前的解法

- 重啟 cilium pod
- 一時無法解除

止血

- 判斷與 node 相關
- 先 cordon 節點


出事
記錄
過去是否
發生過相似的
嘗試套用解方
止血

止血

】對外

- 發公告
- 提出預計修復時間
- 定時回報

各位主管同仁大家好：
［緊急公告］kubernetes 同步作業異常處理。
維護範圍：esunisgood.com 相關的服務
維護起始時間：2024/10/24(四) 18:45
預計結束時間：2024/10/24(四) 20:45

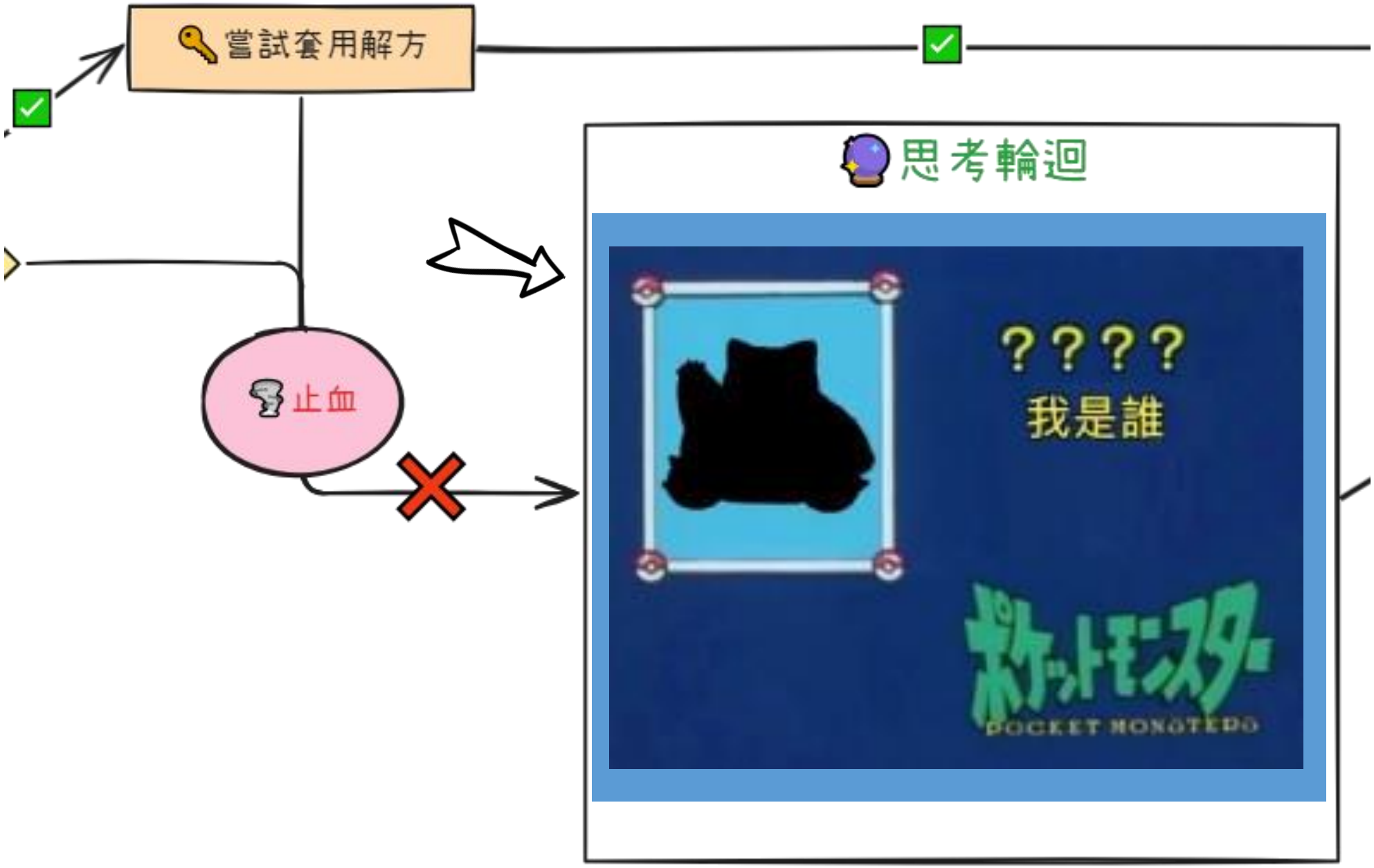
維護說明：
1. 目前 k8s hello world 這台因同步異常，需暫停與 k8s hi 同步後恢復服務
2. 作業期間若有使用 k8s hello world 的請暫時改用 esunistmp.com

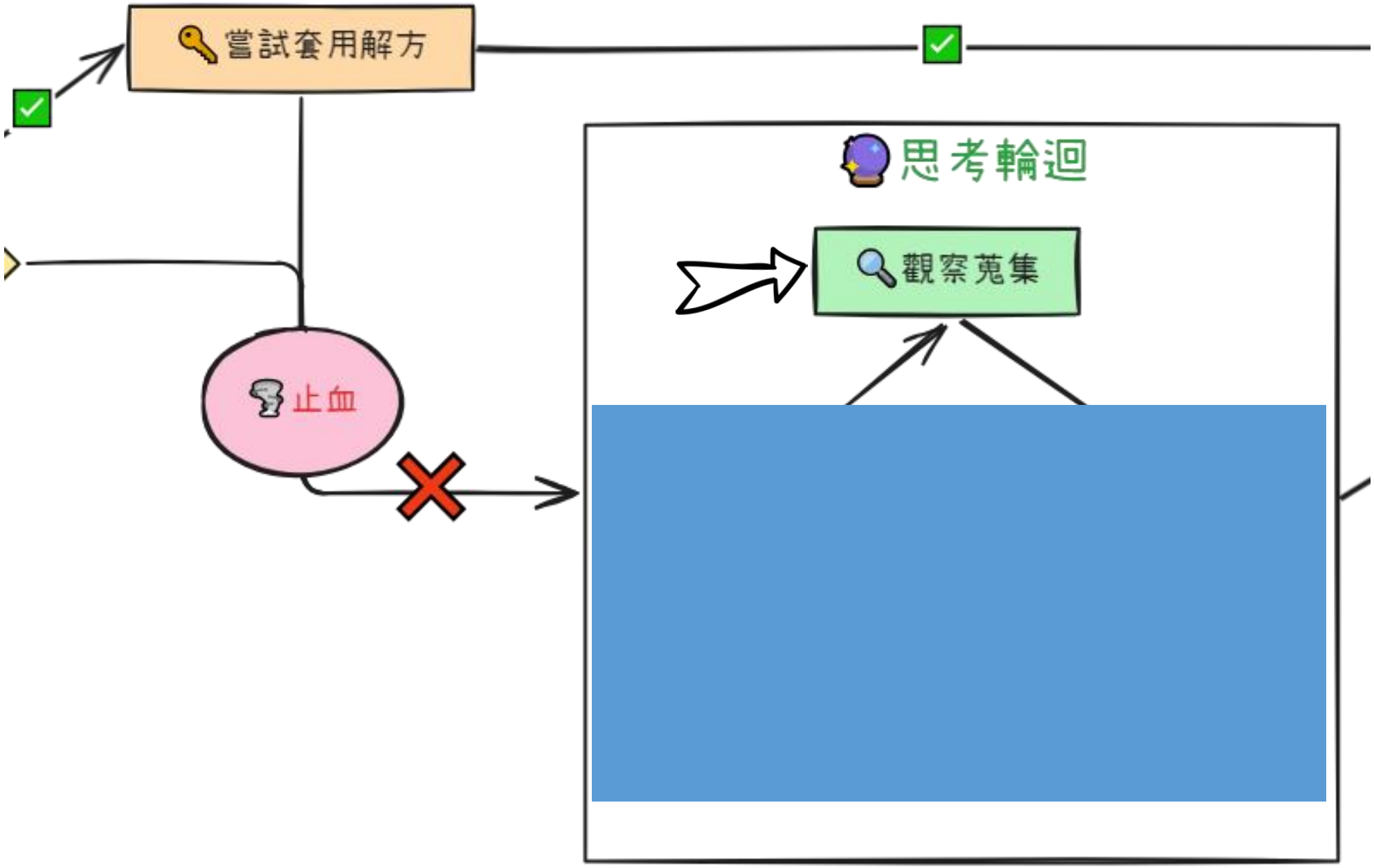
維護負責人：kiwi

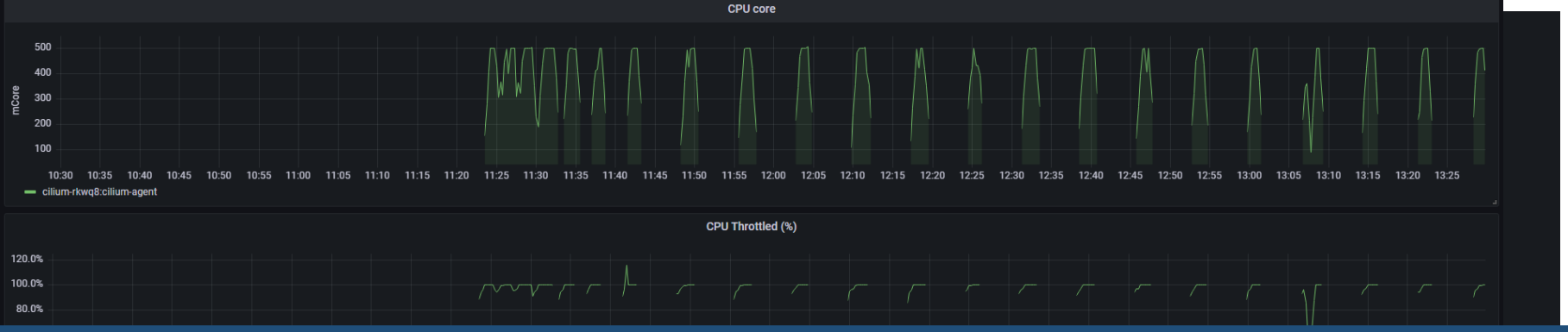
謝謝

【對內

- 看要停止哪些服務
  - pod 驅離
  - node cordon
  - load balancer 直接導轉
  - 啟動 HA 機制
- 相信自己的判斷
  - 錯了趕快修正


嘗試套用解方
思考輪迴
止血
????
我是誰
ポケットモンスター
POCKET MONSTERS


嘗試套用解方
思考輪迴
觀察蒐集
止血



node 使用量

```
level=error msg="Command execution failed" cmd="[tc filter replace dev lxc7bd94e69e05d ingress prio 1 handle 1 bpf da obj 1100_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxc3b478442030f ingress prio 1 handle 1 bpf da obj 3376_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxc8c5595eafa1d ingress prio 1 handle 1 bpf da obj 1448_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxc8e6a056eaf3f ingress prio 1 handle 1 bpf da obj 2340_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxca77270ade973 ingress prio 1 handle 1 bpf da obj 1965_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxcb4c733564269 ingress prio 1 handle 1 bpf da obj 2730_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxce9492526cc87 ingress prio 1 handle 1 bpf da obj 3885_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
level=error msg="Command execution failed" cmd="[tc filter replace dev lxcfaaf789c6fae ingress prio 1 handle 1 bpf da obj 214_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-loade
r
level=error msg="Command execution failed" cmd="[tc filter replace dev lxc0115cb12ae45 ingress prio 1 handle 1 bpf da obj 360_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-loade
r
level=error msg="Command execution failed" cmd="[tc filter replace dev lxcc112b87efe47 ingress prio 1 handle 1 bpf da obj 875_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-loade
r
level=error msg="Command execution failed" cmd="[tc filter replace dev lxcd24f883551c7 ingress prio 1 handle 1 bpf da obj 3834_next/bpf_lxc.o sec from-container]" error="signal: killed" subsys=datapath-load
er
```

各個地方的 log
(假設 monitor infra 沒建好)

```
128771: lxcb8506afda8a6@if128770: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
128773: lxcf7e870a7c084@if128772: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
128775: lxcee17e513782c@if128774: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
128777: lxc86b0d58e2908@if128776: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
99087: lxc6e04432c37ef@if99086: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
120081: lxc4c635f66abe6@if120080: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
99089: lxc15339e9e44b8@if99088: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
17: lxc3183896e4c34@if16: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
120083: lxcc29620ebf6c1@if120082: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
117523: lxcca7b34ab498e@if117522: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
99091: lxcb8f8e0bb396e@if99090: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
99093: lxcd71eae72366a@if99092: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
117527: lxcc3a64084f63f@if117526: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000
```

Node 的 state
(iptabels,iplink,netfliter,route...)

系統元件的 metrics
(cilium,apiserver,etcd,cadvisor...)

找到是關於 cilium state 的儲存機制

```
[root@mlaas20itcm01p state]# kubectl -n kube-system get po cilium-l57xx -oyaml | grep hostPath -A 10
  - hostPath:
      path: /var/run/cilium
      type: DirectoryOrCreate
    name: cilium-run
  - hostPath:
      path: /sys/fs/bpf
      type: DirectoryOrCreate
    name: bpf-maps
  - hostPath:
      path: /proc
      type: Directory
    name: hostproc
  - hostPath:
      path: /run/cilium/cgroupv2
      type: DirectoryOrCreate
    name: cilium-cgroup
  - hostPath:
      path: /opt/cni/bin
      type: DirectoryOrCreate
    name: cni-path
  - hostPath:
      path: /etc/cni/net.d
      type: DirectoryOrCreate
    name: etc-cni-netd
  - hostPath:
      path: /lib/modules
      type: ""
    name: lib-modules
  - hostPath:
      path: /run/xtables.lock
      type: FileOrCreate
    name: xtables-lock
```

hostPath 儲存

state 狀態

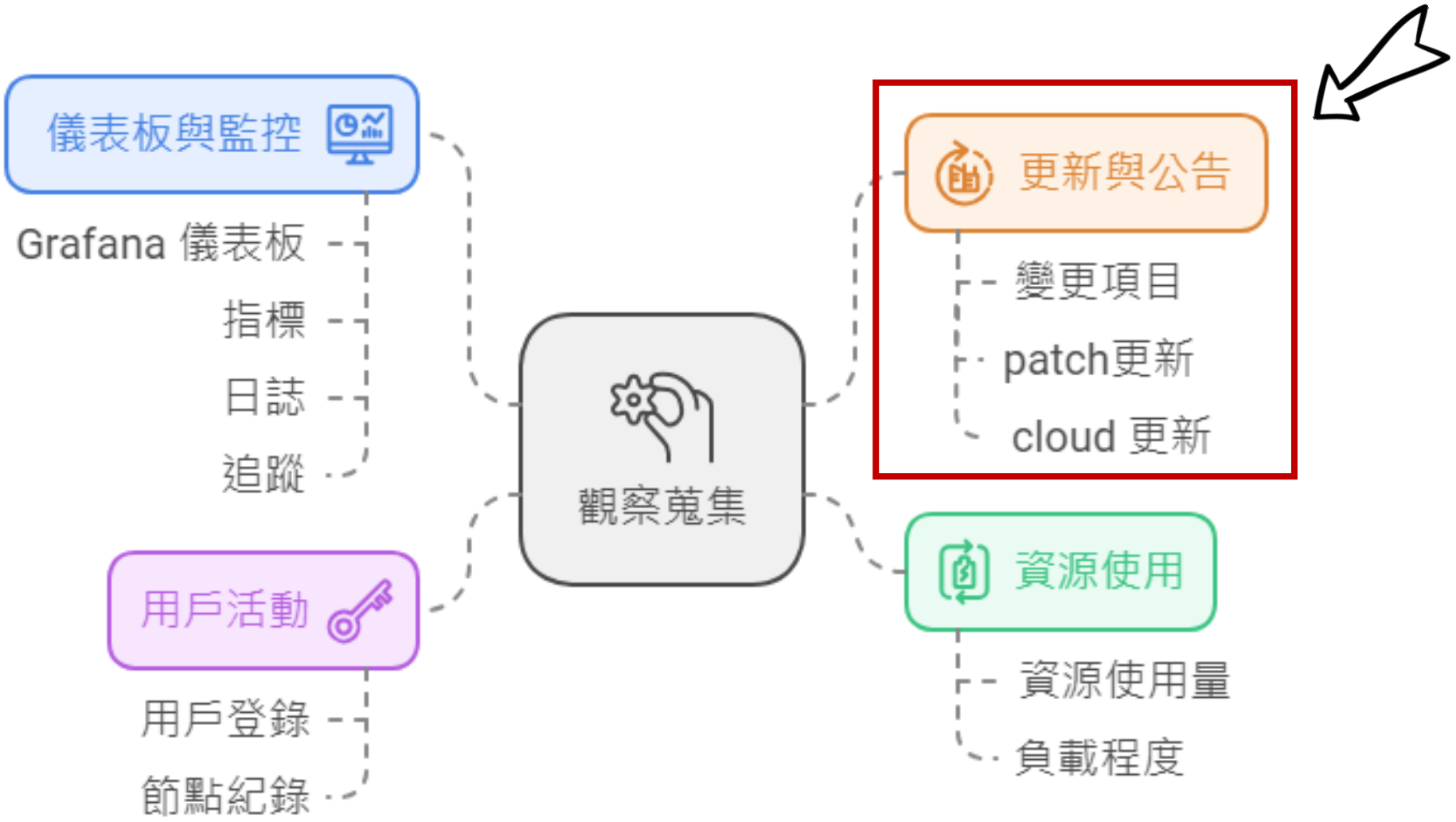
-> 重啟 pod 沒有用

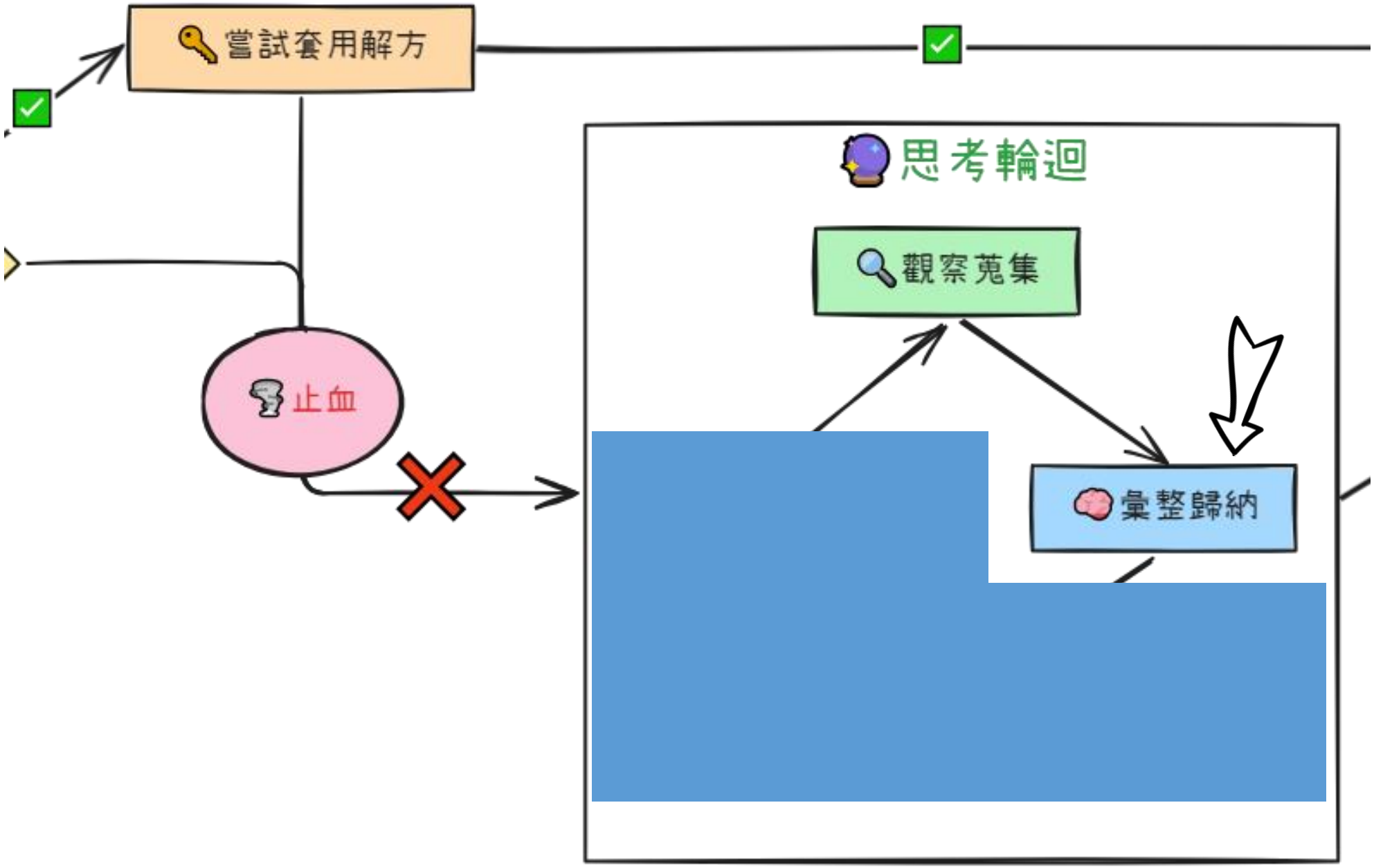
解法

1. 在 pod 重啟前清掉 state
2. 重開機

Ref: https://rexrock.github.io/post/cilium2/

觀察蒐集

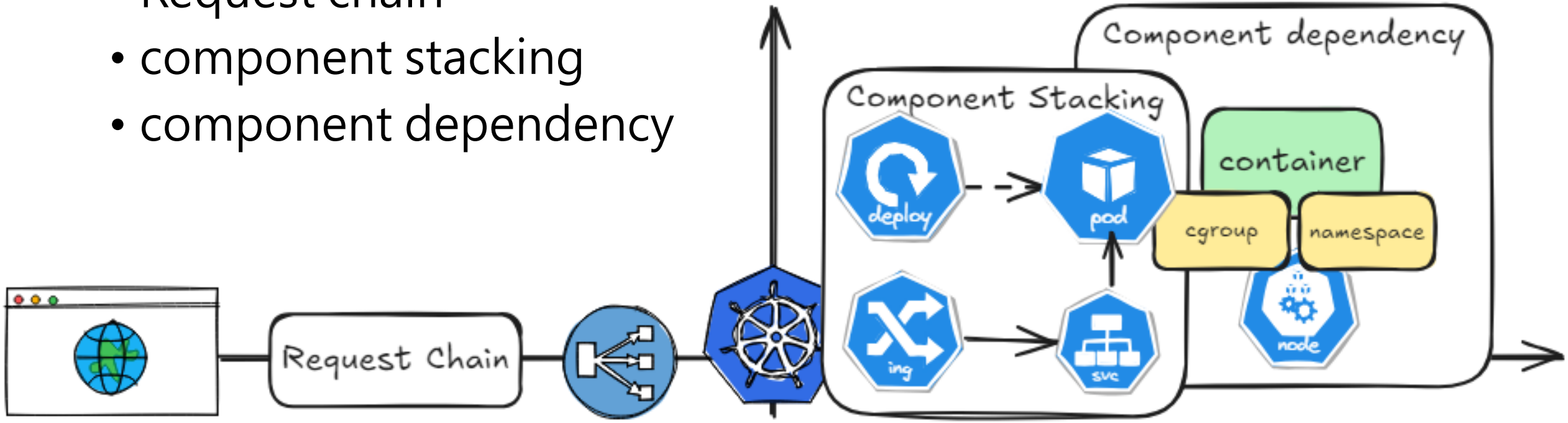

儀表板與監控
Grafana 儀表板
指標
日誌
追蹤
用戶活動
用戶登錄
節點紀錄
觀察蒐集
更新與公告
變更項目
patch更新
cloud 更新
資源使用
資源使用量
負載程度


嘗試套用解方
思考輪迴
觀察蒐集
彙整歸納
止血

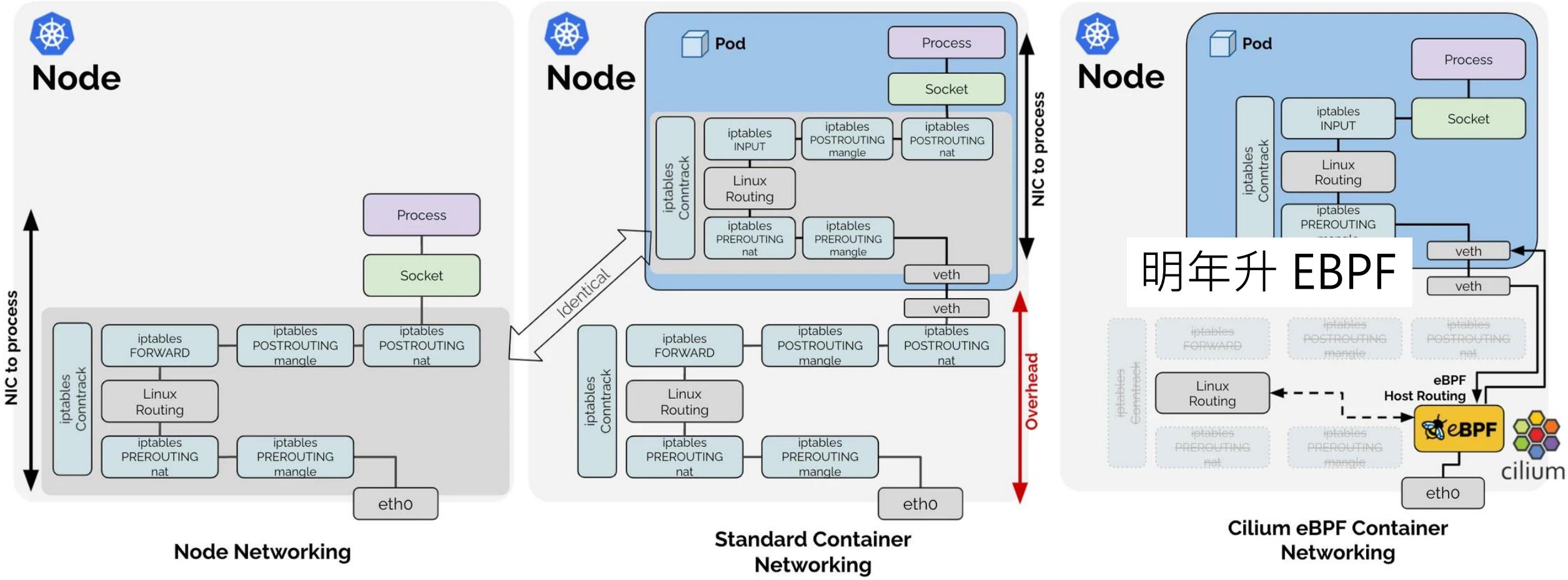
彙整歸納

釐清災難範圍

- Request chain
- component stacking
- component dependency

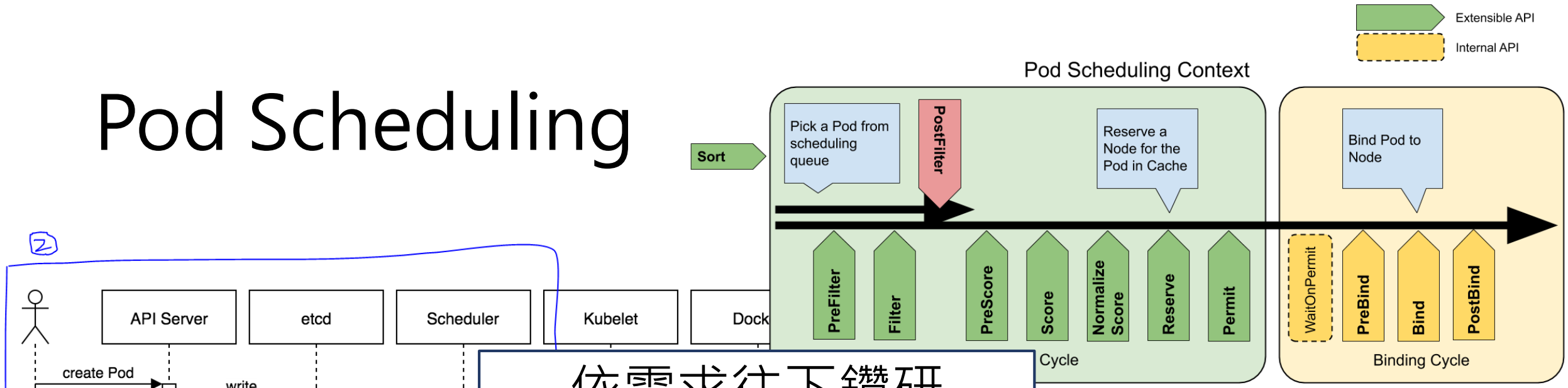


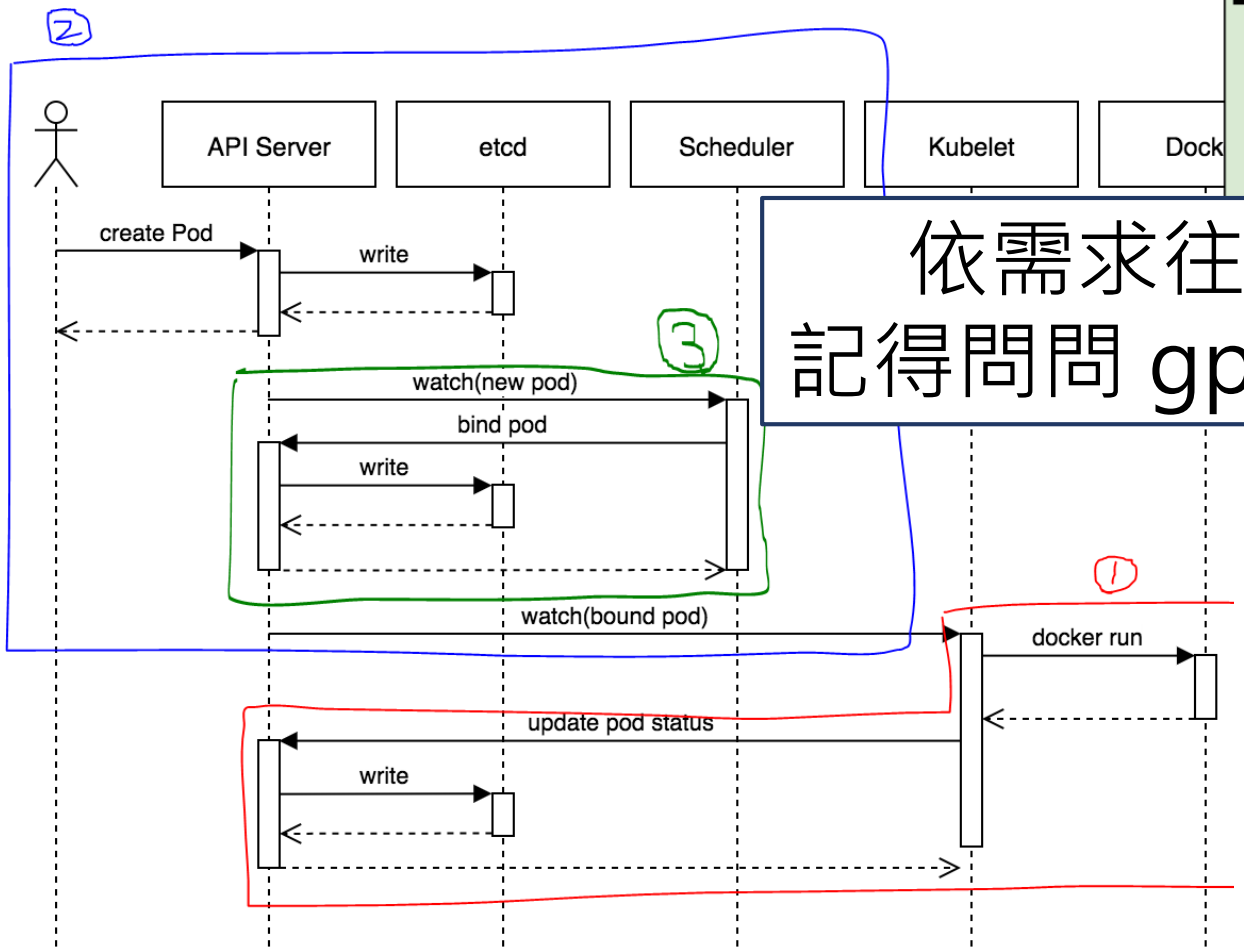
Cilium CNI 解釋



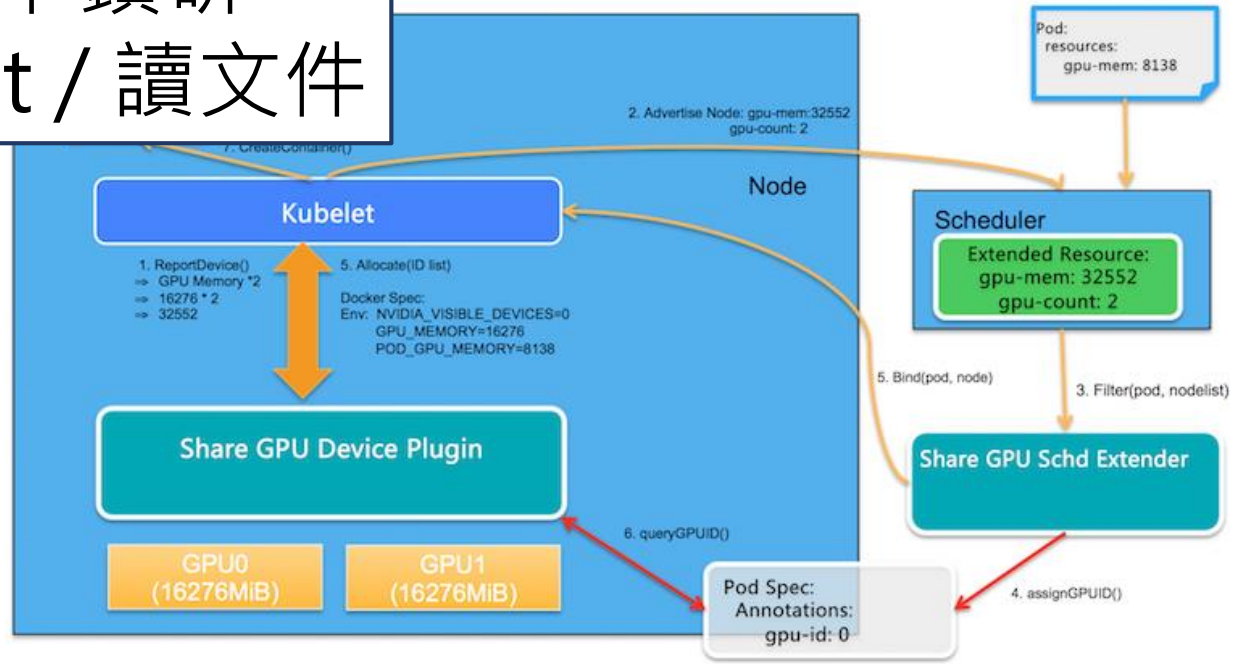
Ref: [CNI Benchmark: Understanding Cilium Network Performance]

Pod Scheduling



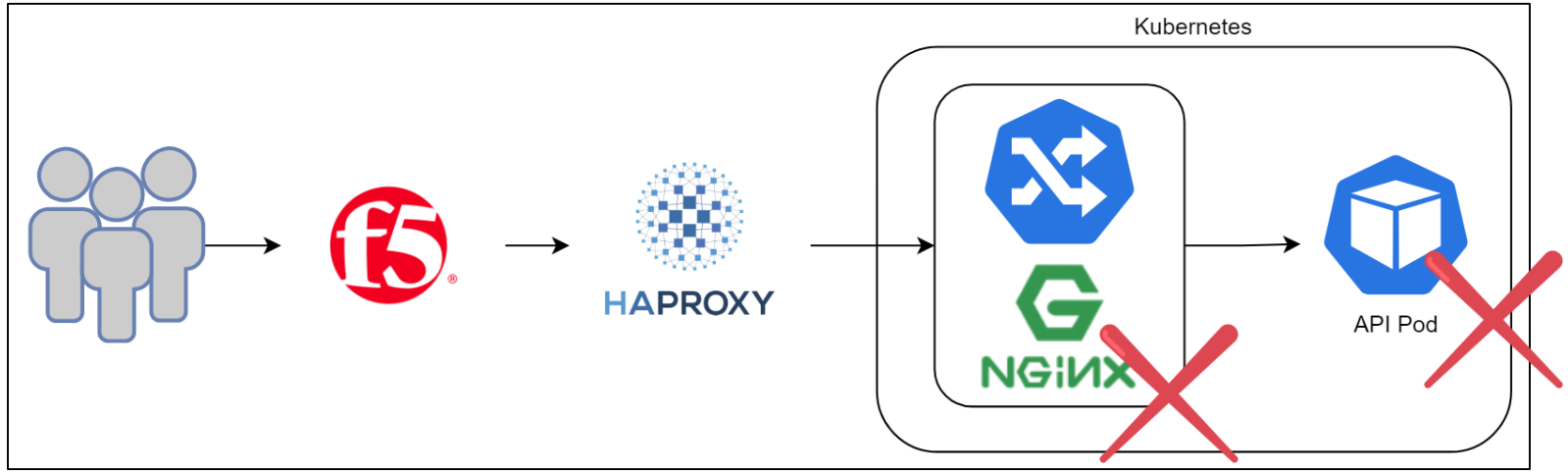


依需求往下鑽研
記得問問 gpt / 讀文件



專案團隊說：「response 400」

Request Chain 解決

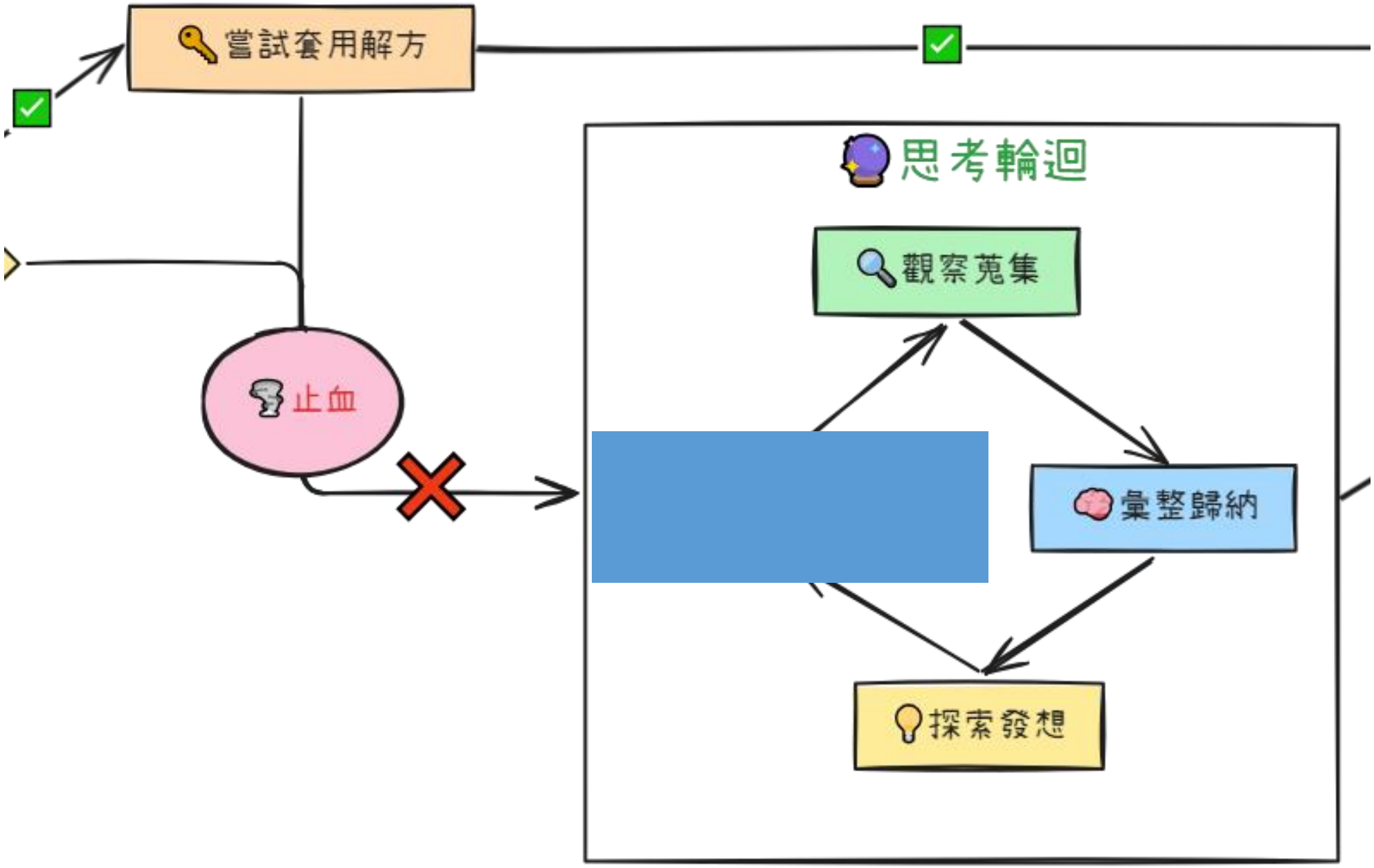



```
:11 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 42/42/0/0/0 0/0 {} "<BADREQ>"
:14 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 37/36/0/0/0 0/0 {} "<BADREQ>"
:16 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 24/23/0/0/0 0/0 {} "<BADREQ>"
:17 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 19/18/0/0/0 0/0 {} "<BADREQ>"
:18 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 17/16/0/0/0 0/0 {} "<BADREQ>"
:19 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 17/16/0/0/0 0/0 {} "<BADREQ>"
:19 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 13/12/0/0/0 0/0 {} "<BADREQ>"
:20 localhost haproxy[                mlaas mlaas/<NOSRV> -1/-1/-1/-1/0 400 187 - - PR-- 17/16/0/0/0 0/0 {} "<BADREQ>"
```

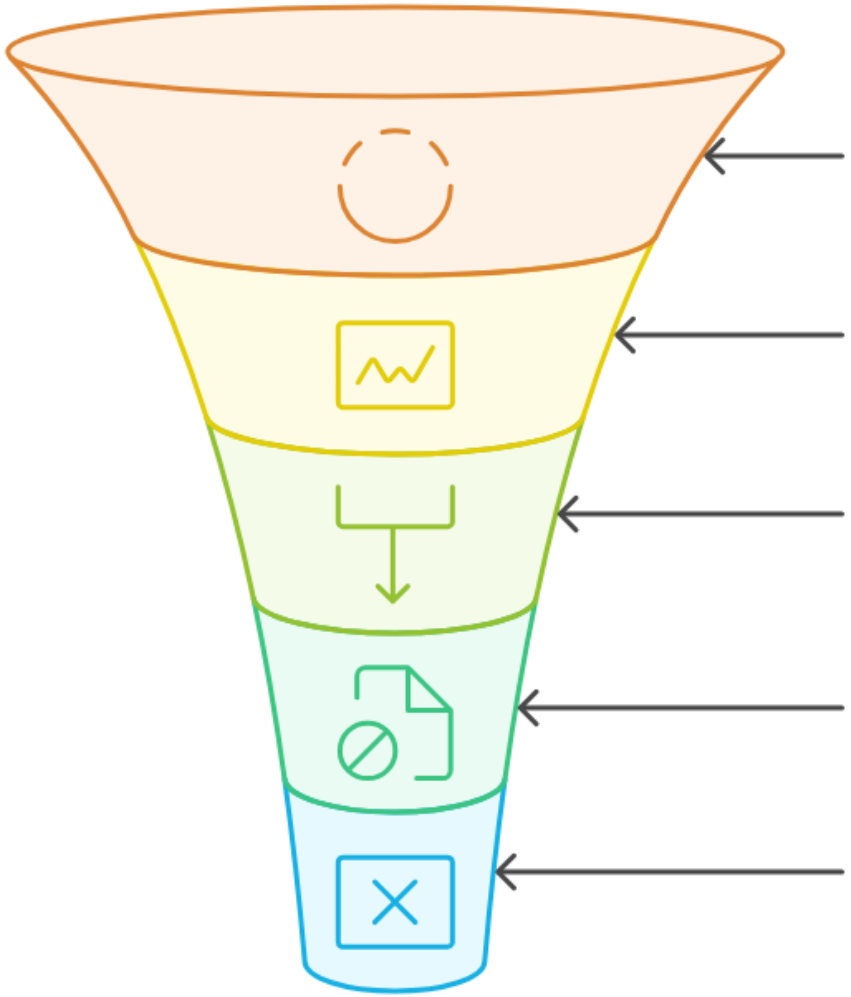
前回應 400 bad request ， MLaaS 有收到 HTTP header 但沒收到 HTTP content body

- 回應 400 bad request ， MLaaS 只有 TCP 連線成功，沒有收到 HTTP header & content body

客戶端呼叫錯誤☺


嘗試套用解方
止血
思考輪迴
觀察蒐集
彙整歸納
探索發想

探索發想



了解直接影響

拆分間接影響

識別關聯反應

排除不受影響的元件

辨識錯誤層級

問問自己

錯誤是如何擴散呢？

是否集中在某個服務/任務呼叫？
會不會是來源端就出錯了？

ex. cluster, node, pod, container

自己弄得錯誤

Actions

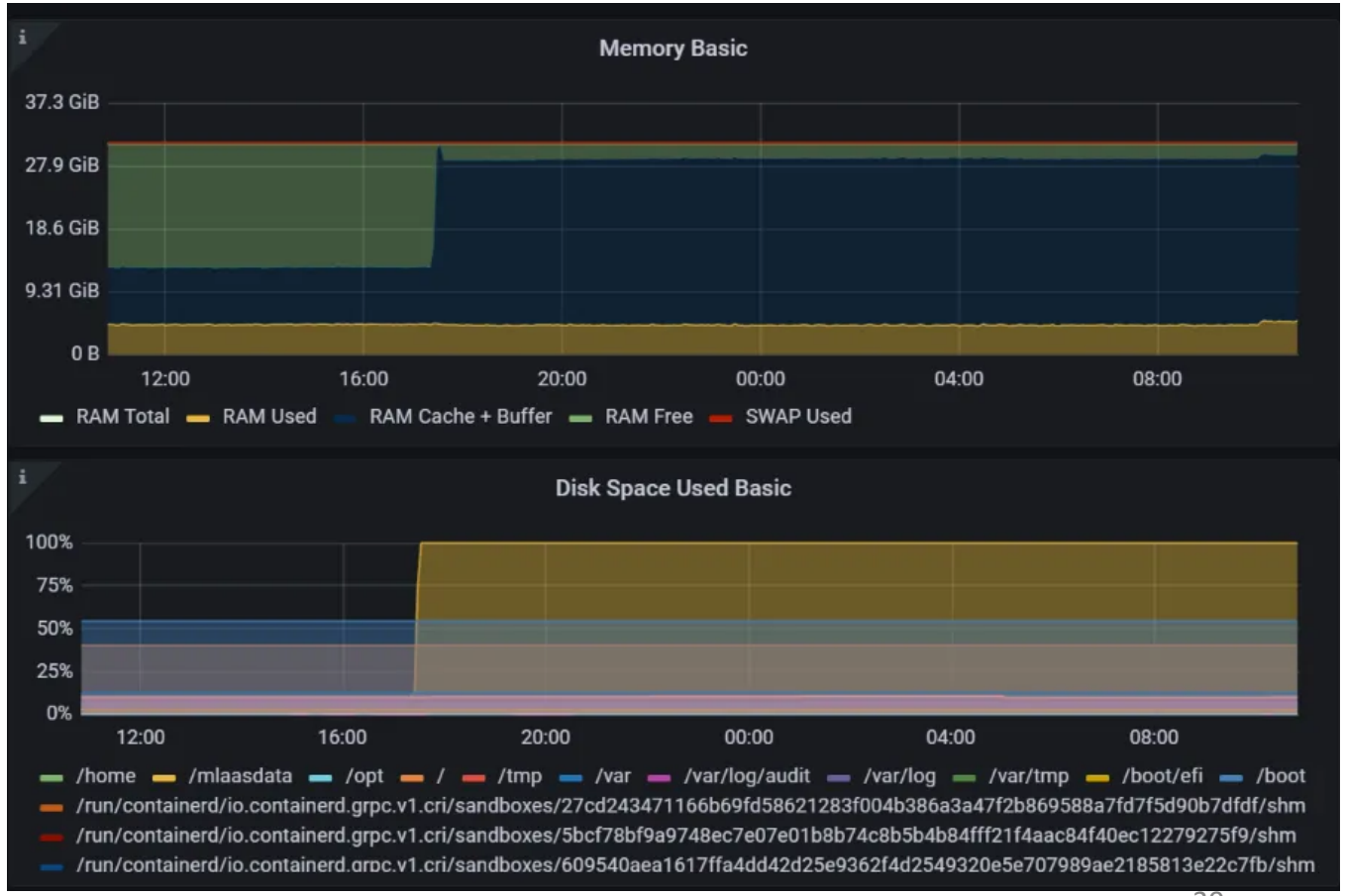
```
:22:34| nerdctl pull
:31:41| nerdctl run --entrypoint bash -it
:31:58| du -h
:32:01| df -h
:32:14| nerdctl run --entrypoint bash -it
:32:27| nerdctl images
:32:34| nerdctl image prune
:32:38| nerdctl image prune --all
:33:59| kubectl get nodes -owide
```

緊急處置後

- ctr images 沒看到
- ctr container 沒看到

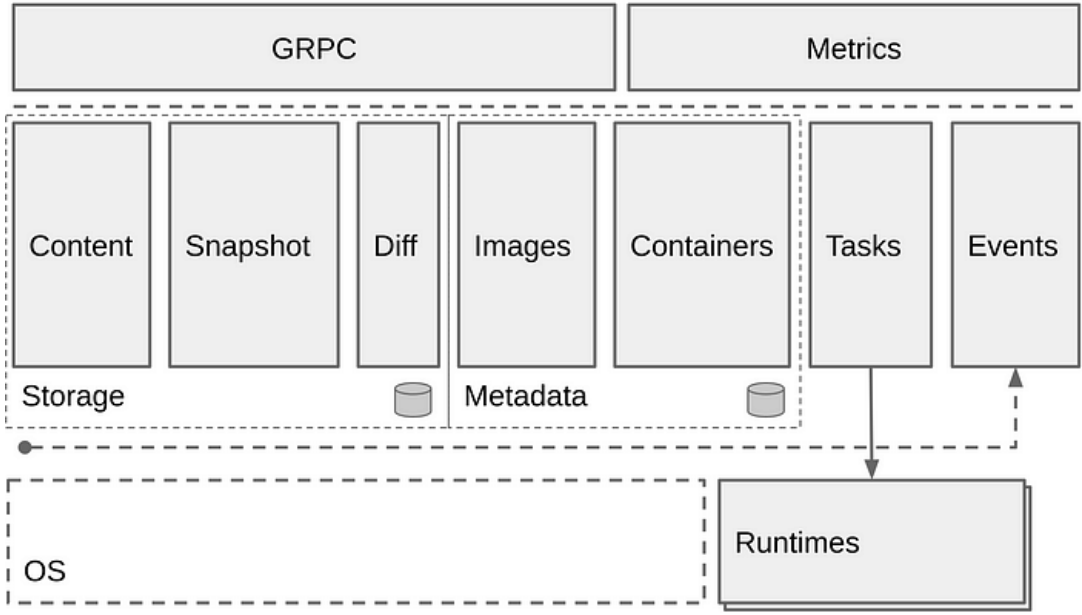
resources 仍然沒降下來

Result

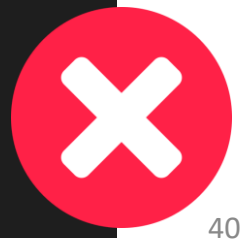


彙整歸納後，嘗試解看看

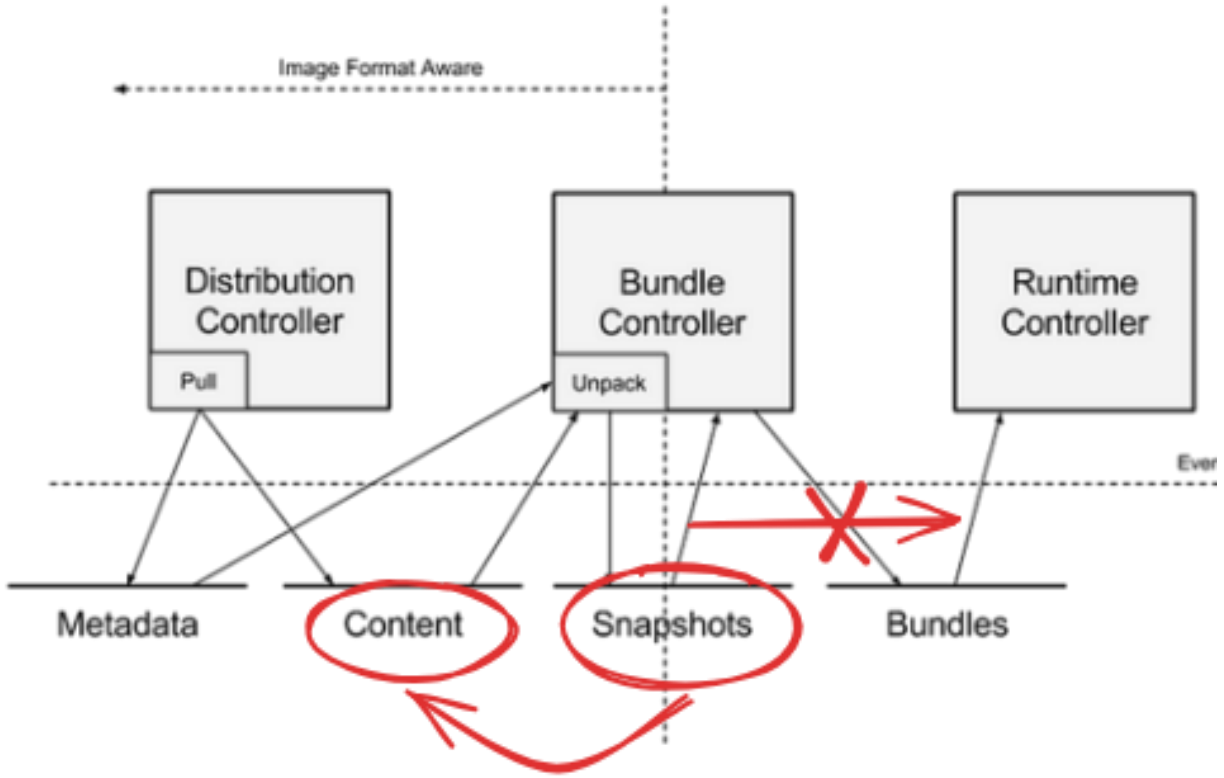
```
[root containerd]# du -h -d 1 containerd/
3.7G    containerd/io.containerd.content.v1.content
1.1G    containerd/io.containerd.snapshotter.v1.overlayfs
72K     containerd/io.containerd.runtime.v2.task
4.0K    containerd/tmpmounts
4.0K    containerd/io.containerd.runtime.v1.linux
8.0K    containerd/io.containerd.grpc.v1.introspection
268K    containerd/io.containerd.grpc.v1.cri
91G     containerd/io.containerd.snapshotter.v1.native
2.7M    containerd/io.containerd.metadata.v1.bolt
4.0K    containerd/io.containerd.snapshotter.v1.btrfs
96G     containerd/
```




```
    \_ sha256:308ef665d277cad0b42fb039b1a4d1c702228646a126ddb4da3062a2ddb7c9cb
      \_ sha256:55608c1965eb0f277f8572e0cc953d1f8a34e80965711f5fcd888deeb067ff50
        \_ sha256:717f9a2779ca8929b3100ecaa9506a1d23c5d15c4b9f58d57b3c1e665b3f4ffc
          \_ extract-536596800-fPIn sha256:3450ad4047717bef9f23b8a873b31cbeedcebe91c39b4fea143d3c72bd8ed908
          \_ extract-748197178-ByGC sha256:3450ad4047717bef9f23b8a873b31cbeedcebe91c39b4fea143d3c72bd8ed908
          \_ sha256:3450ad4047717bef9f23b8a873b31cbeedcebe91c39b4fea143d3c72bd8ed908
sha256:5b1fa8e3e100361047c8bcd5553ab6329b9c713c1d4eb87a646760329cea5b3a
 \_ sha256:f18795e2ae0e0fba2c4fad9a76f7d2432e3c4159dbf3722bf7194daf19b55b80
sha256:87b6a930c8d02125eedb40c3bc2801f3cfd99fcc8c05134a33b0cc74a2fa6b45
 \_ sha256:8cc32bcda2b1d188e9e3a931e4d39ac3bde8e2337d8cc7b700af954f7b63e71e
[root@ ~]# ctr -n k8s.io snapshot --snapshotter rm  sha256:3450ad4047717bef9f23b8a873b31cbeedcebe91c39b4fea143d3c72bd8ed908
No help topic for 'sha256:3450ad4047717bef9f23b8a873b31cbeedcebe91c39b4fea143d3c72bd8ed908'
[root@ ~]# ctr -n k8s.io snapshot --snapshotter rm  sha256:717f9a2779ca8929b3100ecaa9506a1d23c5d15c4b9f58d57b3c1e665b3f4ffc
No help topic for 'sha256:717f9a2779ca8929b3100ecaa9506a1d23c5d15c4b9f58d57b3c1e665b3f4ffc'
```



探索發想 - 從 mount 繞圈



卡了一段時間

- 在 snapshot, namespace
- 挖到 overlayfs, mount 的關聯

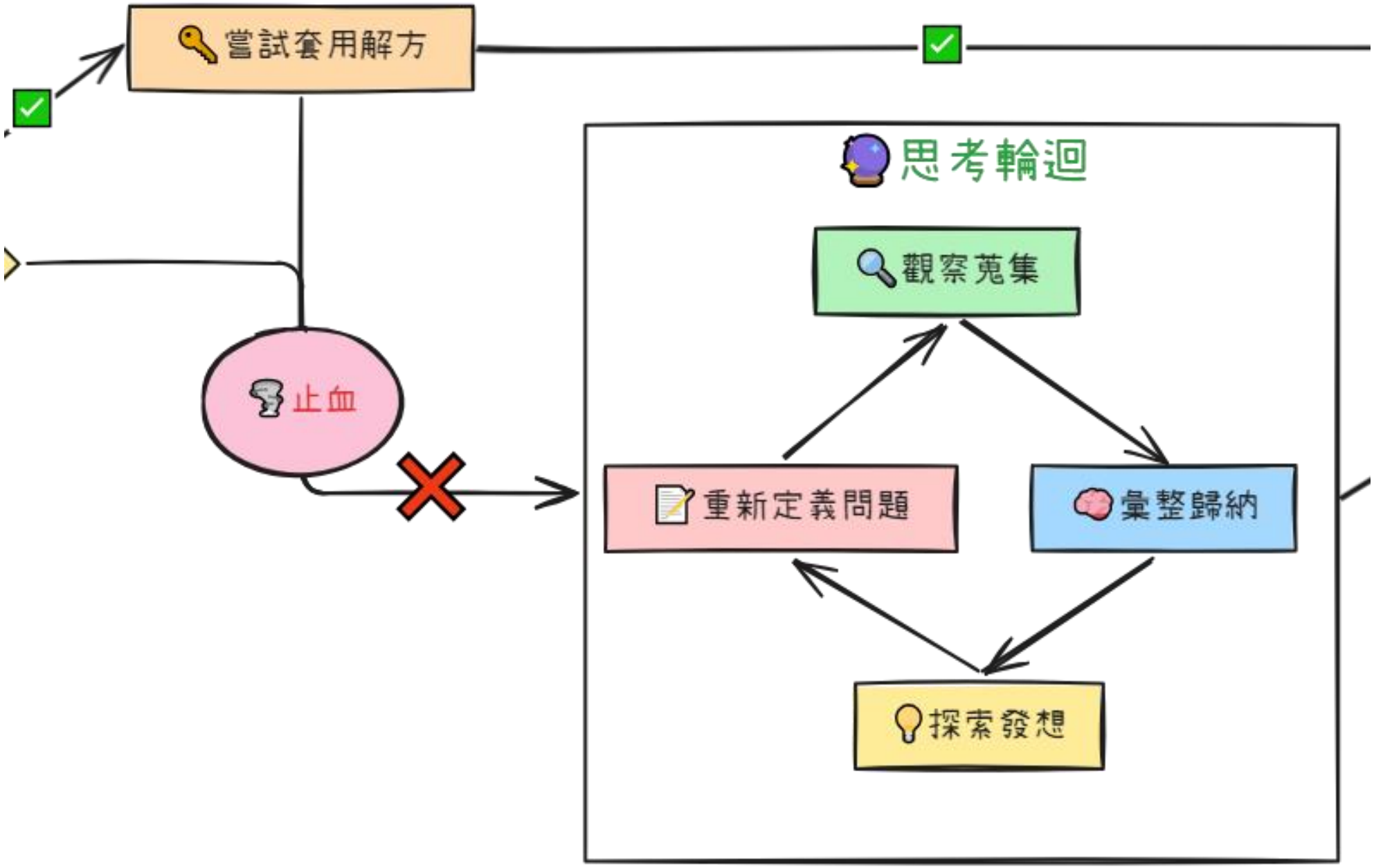
然後意外發現

- content 還沒檢查過
- 之前只有到 image

```
[root@ sha256]# ctr -n k8s.io content ls | grep  | awk '{print $1}' | xargs ctr -n k8s.io content rm
sha256:093f9cb4ed5900be5b069aa140bee4a6533cc0734efe69b68011cb00bc533761
sha256:0c177adbc1a6746a3e0a7fe09576028f3e2522a58b4fd15305d2f1ab6411c519
sha256:1841a82df382ec6f438928bdb39f5e40378de67b1ac84e1424e4b65f818d4f21
sha256:382d92eb8ea3e46c02c6c3f64e34d0564a425211b14730e99b13c4c172465019
sha256:3f87ccbacbdc7fa17cc187efd79dfc2759caafc1e4a7657714bc342add5f9d30
sha256:5cf4eab5d1dafcdd75784e4c9f4ab10d43bda0b3d4397d8a492a31e549b59bfd
sha256:64266c6e10350c49d372bb295f5845a9c2d04b2484841363ede54998d19fb193
sha256:68ef693ccc58a8aa9c387ac7658ae8f56d3f1db06f8e8fad3dca42903c9c4653
sha256:7e77df6a2c111332ec2ad5e83ae851e004544a3933cac31a2988578be4362586
sha256:806911a0f330b943ce4398bfc852cd3a93559e91f9a505cd6baa50242a6cfe33
sha256:852236cde8166ae74149b017340dfcf02d57f02b3e5adbbbaa8383ff069f935e
sha256:924a30603b3c77cd2dc59cf9996b4f294d8ebb7f22b50411d018559132223b79
sha256:9295acad8e8d5fe32390043ab325f91d11a1ddc6c76a1ef5ab854095b45160d0
sha256:9e08999d446baf672263edd667fd0b50cb1c40303530deb803deb0671ceaaf4a
sha256:a2c53e669ed826a359f466e4de5a31619d5efb411f32b5b004bffe9fac305469
sha256:a542df56279ad54cbc6d09c07b0e0549f827d3e9e06f0563292b91a9f8defd39
sha256:a775e38b5f142ba50f73121074d9208f4a7fe5f7d6b35cdf7a1e3ad484664740
sha256:a7b60a932d48762a6a1f3ac8b82f0ca9cd3f9acec09b84f2fc9fb25a4babae79
sha256:a812797e651bb55279090fc711aee33a8ce66905018212c7cf4c0e531007af40
sha256:ab353a332cea8162589830e13948955024e24763883826db65ca28242c36e272
sha256:b07590bc38a13cfcacd7d3ae30925b9b0926959ecd268bcf548dbbef59a76ae9
sha256:ba7e7656f6366696323d20f4de10f052cd938b9038574a8f0c9ce8674fb01e7e
sha256:dc9affc4ae35e5b2340249841e0ebb584a905c4fedda1fd9e195a47f333470a1
sha256:dcc2a1c0b4a6d4453c477c65a05d2a40cb226a1c4a1e34fd4fd50a12eb1fa280
sha256:ee706bdc96476b63d0ff21dbdaf77a33184e33f474000cba5a2
sha256:f775f244232d4c19f65c76d713ac7df9dc5f681e7eeff37fa27
sha256:fb668870d8a72b5d72a3b6d98ee626e00f9f7c29c6f4f7d3a63
```

DONE!!!

```
[root@ snapshots]# du -h -d 0 containerd/io.containerd.snapshotter.v1.native/snapshots/
2.9G    containerd/io.containerd.snapshotter.v1.native/snapshots/
[root@ snapshots]# du -h -d 0 containerd/io.containerd.content.v1.content/blobs/sha256/
384M    containerd/io.containerd.content.v1.content/blobs/sha256/
```

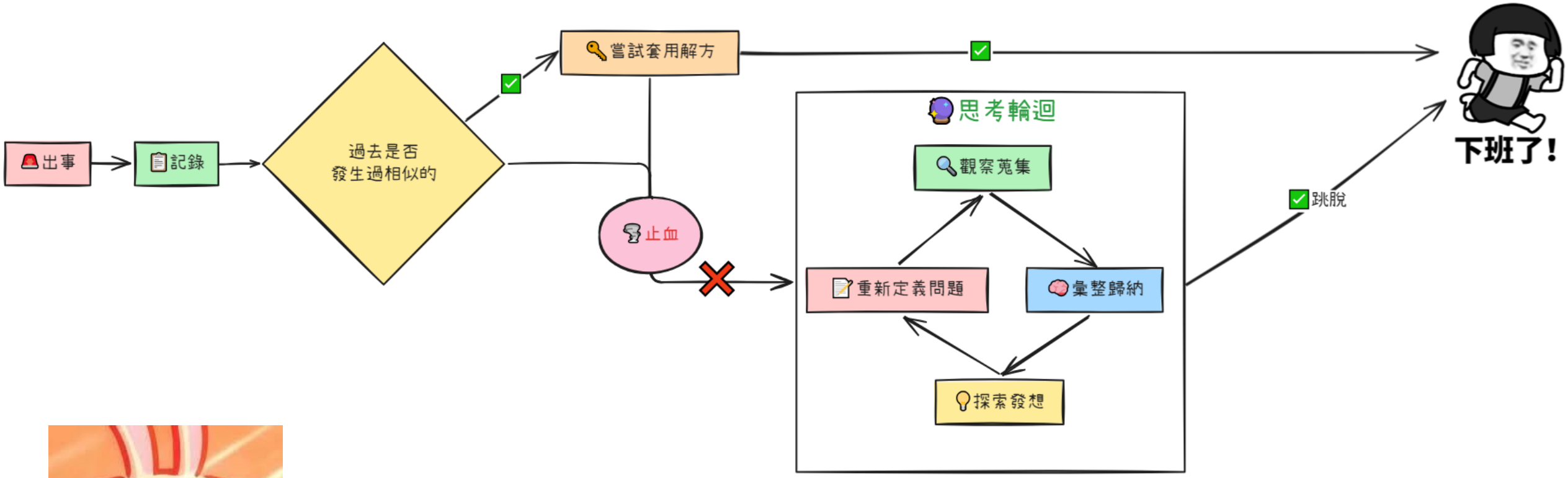

嘗試套用解方
止血
思考輪迴
觀察蒐集
重新定義問題
彙整歸納
探索發想

重新定義

- **保持冷靜 & 維持好心情**
  - 承認自己知識的不足，與他人分享可能會獲得靈感
- **重新聚焦**
  - 根據前面的資料，重新說明問題的範圍和特徵。
- **持續溝通回報**
  - 尋求外部協助，及系統上可能的取捨

撞牆時，相信直覺

- **跳脫慣性思維**
  - 每次都是網路問題啦
  - 每次都是 DNS
  - 每次都是我們系統問題 🤨
- **弄髒手**
  - 直接挖底層 (tcp, packet 追查)
  - 硬體更換 (曾經的麵包板)


出事
記錄
過去是否
發生過相似的
嘗試套用解方
止血
思考輪迴
觀察蒐集
重新定義問題
彙整歸納
探索發想
跳脫
下班了！


前途一片光明啊

事件解決後呢？

- 系統性反思 / Deep Dive
  - 回顧解謎流程，拼湊框架
  - 梳理紀錄 & 錯誤經過
  - 整理成文件，並分享

- 持續
  - 監控系統的擴充與完善（metrics, logs, traces）
  - 預防重於治療， alert 與 playbook 的定期盤點
  - 接觸更廣的技術維度

Takeaway

- 不要過度信任元件（包括開源項目）的完美運行
- 認識元件流程與工作原理
  - 使用 ChatGPT、Perplexity 等工具輔助擴展想法
  - 看文件，看之前的事件處理
- 保持冷靜與好心情

Thank you!

✉ | sean22492249@gmail.com

🌍 | https://kiwi-walk.com

M | https://sean22492249.medium.com/