**up{}** au lieu de **probe\_success ?**

Tu peux ajouter une variante pour **up{}** au lieu de **probe\_success** si tu mesures les services internes. Surprenant car les PKI sont là pour la performance mais aussi détecter des anomalies indices pour des tentatives d'intrusions ou de **DDNS** donc oui **up{**} et **probe\_sucess** ensemble. D'ailleurs tu pourrais me les expliquer précisément.

Excellente remarque. Tu soulignes une réalité opérationnelle : **les KPI ne servent pas uniquement à surveiller la performance**, mais aussi à **détecter des comportements anormaux**, précurseurs d’un incident ou d’une attaque (**DoS**, dégradation lente, faille en escalade silencieuse…).

Voici une explication rigoureuse et comparative des deux **métriques fondamentales Prometheus** : up{} et probe\_success{}.

**1. up{} — Présence d’une cible déclarée dans Prometheus**

**Définition :** C’est une **métrique automatique générée par Prometheus** pour chaque target (cible) déclarée dans le scrape\_configs.

up{job="node\_exporter", instance="localhost:9100"}

**Valeurs possibles :**

| **Valeur** | **Signification** |
| --- | --- |
| **1** | **Cible joignable (Prometheus a réussi le scrape)** |
| **0** | **Échec de récupération → cible absente, plantée ou port fermé** |
| **null** | **Cible non déclarée, ou arrêtée avant la dernière collecte** |

**Usage typique :**

* Surveillance des **services internes** (Node Exporter, Telegraf, Prometheus lui-même, etc.)
* Vérifie que Prometheus **peut “parler” à la cible** déclarée.

**Limite :**

* up ne teste **que la connexion entre Prometheus et le service**.
* Ne teste pas un vrai **comportement applicatif ou réseau externe** (pas d’ICMP, pas d’analyse de contenu).

**2. probe\_success{} — Résultat d’un test réseau (Blackbox Exporter)**

**Définition :** Métrique générée par **Blackbox Exporter**, qui effectue un **test réel de type réseau** :

probe\_success{job="blackbox\_http", instance="https://www.google.com"}

**Modules disponibles :**

* http\_2xx : test HTTP(S)
* icmp : ping
* tcp\_connect : port ouvert
* ssh\_banner : bannières SSH
* tls\_connect : vérification SSL

**Valeurs possibles :**

| **Valeur** | **Signification** |
| --- | --- |
| **1** | **Le test défini (ping, HTTP, TCP...) a réussi** |
| **0** | **Le test a échoué (timeout, refus, 500, etc.)** |

**Avantages :**

* Permet de vérifier un **comportement réseau complet**
* Idéal pour la **disponibilité perçue** (vue client/public)
* Compatible avec la **détection d’un DDOS** (latence élevée, timeout, reset)

**Comparatif synthétique**

| **Critère** | **up{}** | **probe\_success{}** |
| --- | --- | --- |
| **Généré par** | **Prometheus** | **Blackbox Exporter** |
| **Cible** | **Interne (exporters, services)** | **Interne ou externe** |
| **Porté sur** | **Port TCP déclaré** | **Protocole ICMP, HTTP, TCP, SSH, TLS...** |
| **Résultat** | **1 = joignable, 0 = non joignable** | **1 = succès complet, 0 = échec complet** |
| **Utilisation typique** | **Vérifier que le service répond** | **Vérifier que le service fonctionne** |
| **Anomalie détectée** | **Service éteint ou refusé** | **Dégradation, latence, blocage actif** |
| **Détection de DDOS possible** | **⚠️ Non (si TCP répond)** | ✅ Oui (timeout, réponse lente, RST) |

**Corrélation possible dans un alert.rules.yml**

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

**En cybersécurité (IDS/SIEM)**

Ces deux métriques, lorsqu’elles changent :

* trop souvent (flapping)
* en décalage (ex : up = 1, probe\_success = 0)  
  => peuvent **indiquer une attaque ciblée, un filtrage réseau, un faille L7 ou un sabotage interne**.

**Script Bash de corrélation up{} vs probe\_success{} :**

**verif\_corrup\_probe.sh**

**Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.**