Ordivis

MODUL · SERVICE DESK

Monitoring-Anbindung

Alarme einer Überwachung als Vorgänge – Entwarnung, Wartungsfenster, Sturmbremse

Produktversion 2026.09.20 · Handbuch-Revision 1.0
Stand 20.09.2026 · Plattform .NET 10 / Windows Server

ID-20a Monitoring-Anbindung

Monitoring-Anbindung

Aus Störungsmeldungen einer Überwachung werden Vorgänge — dedupliziert, mit Entwarnung, ohne Fehlalarm im Wartungsfenster.

INHALT

1 Was diese Anbindung leistet — und was nicht
2 Der Weg einer Meldung
3 Eine Quelle einrichten
4 Das Regelwerk
5 Entwarnung und Nachlaufzeit
6 Die Aufnahme technisch
6.1 Die Anmeldung der Überwachung
7 Vorlagen für CheckMK und PRTG
7.1 CheckMK — Benachrichtigungsskript
7.2 PRTG — Execute HTTP Action
8 Die Alarm-Übersicht
9 Aufbewahrung der Roh-Ereignisse
10 Rechte
Verwandte Themen

1 Was diese Anbindung leistet — und was nicht

Ordivis IT **überwacht nicht selbst**. Es nimmt entgegen, was CheckMK, PRTG, Zabbix, Nagios oder ein beliebiges anderes Werkzeug meldet, und macht daraus einen nachvollziehbaren Vorgang.

Der Nutzen liegt nicht im Weiterreichen, sondern im **Wegräumen des Rauschens**:

Ohne Anbindung	Mit Anbindung
Fünfzig Mails zu einer defekten Platte	Ein Vorgang mit Zähler
Nachts geöffnet, morgens von Hand geschlossen	Entwarnung löst automatisch
Wartungsfenster erzeugt Fehlalarme	Meldung wird vermerkt, kein Vorgang
Ein Ausfall des Rechenzentrums flutet das Postfach	Ein Sammelvorgang

2 Der Weg einer Meldung

Jede Meldung nimmt denselben Weg, gleich ob sie per Webhook kommt oder aus einem Postfach. Die **Reihenfolge** ist dabei nicht beliebig – jeder Schritt setzt voraus, dass der vorige bestanden ist (siehe Abbildung 1):

1. **Idempotenz** — Wurde diese Ereignis-Kennung schon verarbeitet? Dann Ende. Eine Wiederholung nach einem Zustellfehler darf keinen zweiten Vorgang erzeugen. Diese Prüfung steht **vor allem anderen**;

sonst liefe die halbe Kette umsonst.

2. **Episode** — Gibt es zu diesem Korrelationsschlüssel schon eine offene Störung? Dann wird angehängt, und alles Weitere entfällt.
3. **Gerätezuordnung** — Zu welchem Configuration Item gehört der gemeldete Host? Kein Treffer ist **kein Fehler**: Der Vorgang entsteht trotzdem, nur ohne Verknüpfung.
4. **Regelwerk** — Kategorie, Gruppe, Bearbeiter, Vorgangsart, oder „kein Vorgang".
5. **Wartungsfenster** — Läuft eines im Geltungsbereich? Dann wird vermerkt, nicht gemeldet.
6. **Sturmbremse** — Über der Schwelle hängt alles an einem Sammelvorgang.
7. **Vorgang** — Anlegen oder kommentieren.
8. **Alarmierung** — erst **danach**, und nur wenn an der Quelle ein Alarmkreis steht. Scheitert sie, steht der Vorgang trotzdem: Er ist das, was bleibt; der Alarm ist der Weckruf dazu.

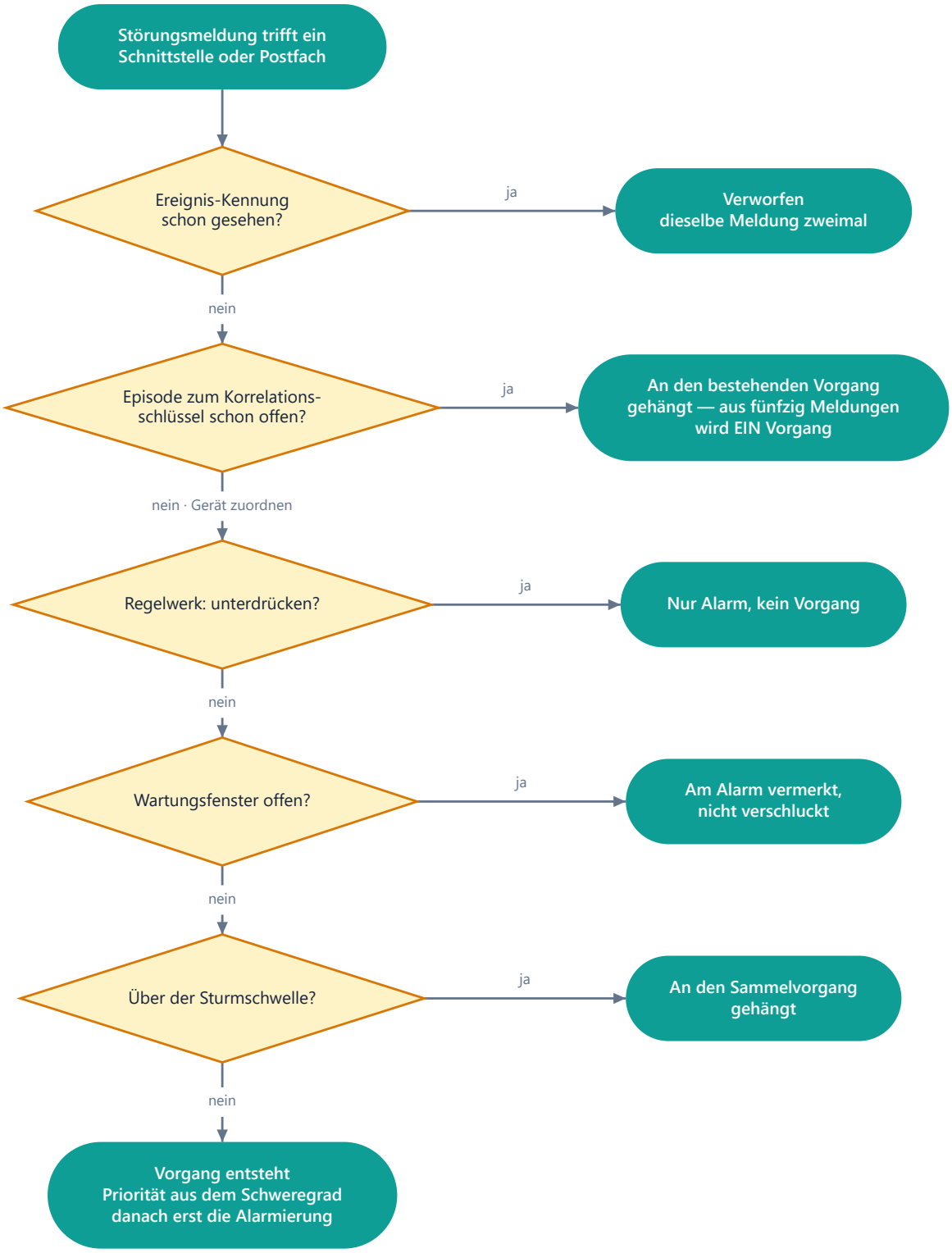

Abb. 1 – Der Weg einer Störungsmeldung. Fünf Entscheidungen liegen vor dem Vorgang, und jede hat ihren eigenen Ausgang – alle fünf enden, ohne dass ein *neuer* Vorgang entsteht.

Warum eine Episode und nicht ein Ereignis. Die Episode ist der Zeitraum von der ersten Meldung bis zur Entwarnung. Sie ist der Grund, warum aus fünfzig Meldungen ein Vorgang wird. Der **Korrelationsschlüssel** entscheidet darüber — üblicherweise `host/dienst` . Wird er falsch gebildet, entstehen entweder fünfzig Vorgänge für eine Störung oder einer für fünfzig verschiedene.

3 Eine Quelle einrichten

💻 **Nur im Client:** *Service Desk* ▸ *Monitoring-Anbindung*, Registerkarte *Quellen* · 🔑 `monitoring.source.manage` — Das Portal zeigt die eingehenden Alarme, richtet die Quellen aber nicht ein.

Sehen Sie die Registerkarte *Quellen* nicht, fehlt Ihnen das Recht. Der Menüpunkt selbst steht dem ganzen Service Desk offen (Abschnitt 10); die Quellen und der Knopf *Quelle anlegen* erscheinen nur mit `monitoring.source.manage` . Die Seite öffnet dann gleich auf *Alarme*.

🔒 **Pflichtreihenfolge** — die **Art** der Quelle belegt die Schweregrad-Abbildung vor. Wer sie zuletzt setzt, überschreibt seine eigenen Einstellungen.

1. Öffnen Sie *Service Desk* ▸ *Monitoring-Anbindung* und wechseln Sie auf die Registerkarte *Quellen*.
2. Klicken Sie auf *Quelle anlegen*.
3. Wählen Sie die **Art** des Überwachungswerkzeugs.
4. Prüfen Sie die vorbelegte **Schweregrad-Abbildung** und passen Sie sie an Ihre Skala an.
5. Stellen Sie **Nachlaufzeit**, **Sturmschwelle** und **Flatter-Erkennung** ein.
6. Hinterlegen Sie den **Tieflink** mit den Platzhaltern `{host}` und `{service}`, damit ein Klick im Vorgang zum Sensor springt.
7. Speichern Sie.

✅ **Ergebnis:** Die Quelle nimmt Meldungen entgegen. Ihr letzter Empfang steht ab jetzt in der Alarm-Übersicht (Abschnitt 8) – der Wert, an dem Sie einen stillen Ausfall erkennen.

Je Quelle einstellbar — und das ist Absicht: Wer CheckMK und PRTG nebeneinander betreibt, hat zwei Schweregrad-Skalen und zwei Meldefrequenzen.

Einstellung	Bedeutung	Vorgabe
Art	Bestimmt die vorbelegte Schweregrad-Abbildung	—
Schweregrad-Abbildung	`{"crit":2,"warn":1}` — welcher Schweregrad welche Priorität ergibt	Vorbelegung des Werkzeugs
Nachlaufzeit	Minuten zwischen Entwarnung und Lösen	10
Sturmschwelle	Meldungen je Minute, ab der gesammelt wird	20
Flatter-Erkennung	Zustandswechsel in einem Zeitfenster	5 in 30 min

Einstellung	Bedeutung	Vorgabe
Vorrang menschlicher Arbeit	Entwarnung kommentiert nur, wenn jemand am Vorgang war	an
Tieflink	Muster mit `{host}` und `{service}` — Sprung zum Sensor	—

Ein unbekannter Schweregrad fällt auf die Vorgabe, nicht durch. Eine Meldung zu verwerfen, weil ihre Stufe nicht in der Tabelle steht, ist der stille Ausfall, den niemand bemerkt: Die Überwachung meldet, Ordivis IT schweigt, und beide halten sich für zuständig.

4 Das Regelwerk

Bedingungen (alle gesetzten UND-verknüpft): Host-Muster, Dienst-Muster, Schweregrad, Label. Platzhalter ist `*` — bewusst kein regulärer Ausdruck, damit ein versehentlich gültiger Ausdruck keine unerwartete Menge trifft. Eine Regel **ohne** Bedingung trifft auf alles zu; so baut man eine Auffangregel ans Ende.

Aktionen: Kategorie, Support-Gruppe, Bearbeiter, Vorgangsart, Priorität — oder **kein Vorgang**.

Die Auswertung folgt derselben Semantik wie die Ticket-Automatisierungsregeln: Reihenfolge über die Sortierung, spätere Regeln überschreiben frühere, `Kette beenden` bricht ab.

Unterdrückung lässt sich nicht zurücknehmen. Hat eine Regel entschieden, dass kein Vorgang entsteht, kann eine spätere das nicht aufheben. Sonst hinge die Wirkung an der Sortierreihenfolge, und eine Ausnahme („dieser Testserver nie") wäre nicht verlässlich. Der Alarm wird trotzdem geführt und ist in der Alarm-Übersicht sichtbar — **unterdrückt heißt nicht verschluckt**.

5 Entwarnung und Nachlaufzeit

Meldet die Überwachung Entwarnung, wird der Vorgang **nicht sofort** gelöst: Die Nachlaufzeit läuft, und ein erneutes Problem in diesem Fenster hebt sie auf. Ohne sie öffnete und schlösse eine flatternde Prüfung im Minutentakt Vorgänge.

Vorrang menschlicher Arbeit: Wurde am Vorgang gearbeitet — der Status ist nicht mehr „Neu", oder es gibt einen Eintrag eines Menschen im Verlauf —, kommentiert die Entwarnung nur. Eine Überwachung weiß nicht, ob jemand noch prüft. Abbildung 2 zeigt, wann ein Vorgang von selbst zugeht und wann nicht.

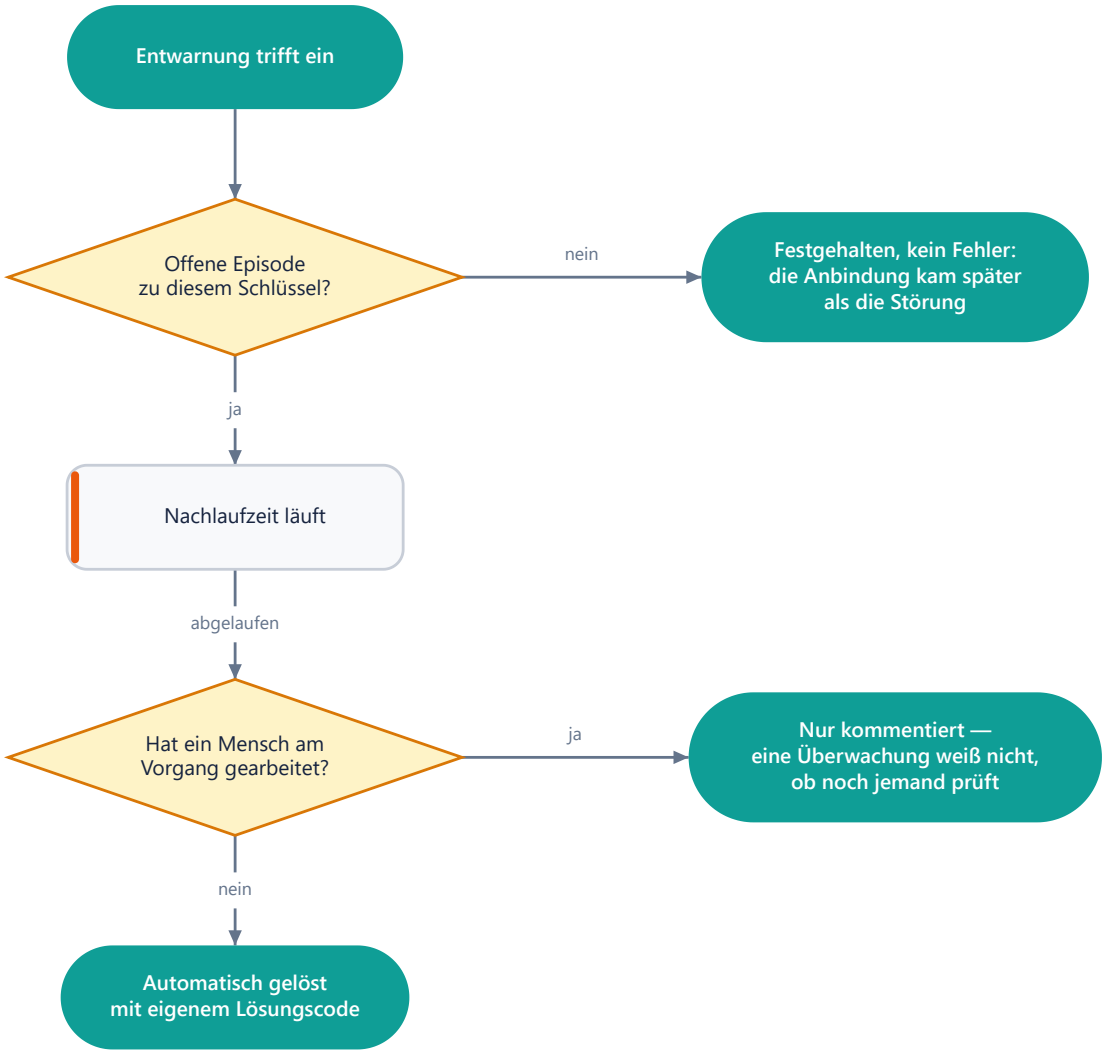

Abb. 2 – Der Rückweg. Eine Entwarnung ohne offene Episode ist kein Fehler, und eine Entwarnung auf einen bearbeiteten Vorgang löst ihn nicht.

6 Die Aufnahme technisch

`POST /api/v1/monitoring/events` · Recht `monitoring.event.ingest` · **Stapel** mit Quittung je Ereignis

```
{
  "schemaVersion": 1,
  "sourceId": "…",
  "events": [
    {
      "eventId": "checkmk-4711",
      "host": "srv-db-01",
      "service": "Speicherplatz /var",
      "severity": "crit",
      "eventType": "problem",
      "subject": "srv-db-01: /var zu 95 % belegt",
      "message": "FILESYSTEM /var 95.2% used",
      "labels": ["umgebung=produktiv"]
    }
  ]
}
```

`eventType` ist `problem` oder `recovery`. Die Antwort quittiert **je Ereignis** — ein Stapel darf nicht an einer krummen Zeile scheitern.

6.1 Die Anmeldung der Überwachung

Ein Überwachungssystem läuft unbeaufsichtigt und hat **kein Benutzerkonto**. Es meldet sich mit einem **Integrations-Token** an, das Sie in der Plattformverwaltung ausstellen:

```
Authorization: Bearer <Integrations-Token>
```

Das Token braucht ausdrücklich das Recht `monitoring.event.ingest`. Es steht bewusst **nicht** in der Vorbelegung eines neuen Tokens: Dieses Recht legt Vorgänge im Service Desk an, und ein fehlkonfigurierter Inventar-Agent, der es beiläufig mitbekäme, könnte die Vorgangsliste fluten. Ein bestehendes Token bekommt das Recht auch nicht nachträglich — stellen Sie dafür ein neues aus.

> Der Weg über ein Benutzerkonto bleibt daneben bestehen. Er ist für die Ersteinrichtung und die Parser-Probe gedacht, nicht für den Dauerbetrieb.

7 Vorlagen für CheckMK und PRTG

Beide Werkzeuge erzeugen das Format **selbst**. Das ist Absicht: Ein werkzeugspezifischer Adapter in Ordivis wäre eine zweite Stelle, an der sich Feldnamen ändern können.

7.1 CheckMK — Benachrichtigungsskript

> 🔑 `monitoring.event.ingest` — Auch diese Schritte laufen **in CheckMK**. Das Skript gehört nach `~/local/share/check_mk/notifications/ordivis` und muss ausführbar sein; danach steht es unter *Setup → Notifications* als Methode zur Wahl. CheckMK übergibt alles über Umgebungsvariablen.

```
#!/bin/bash
# Ordivis IT – Stoerungsmeldung einspeisen. Rueckgabe 0 = zugestellt, 2 = dauerhaft fehlgesc
set -u
ORDIVIS_URL="https://ordivis.example.local"
ORDIVIS_TOKEN="…"          # Integrations-Token mit Recht monitoring.event.ingest
ORDIVIS_SOURCE="…"         # sourceId der Quelle aus Ordivis

if [ "$NOTIFY_WHAT" = "SERVICE" ]; then
  SERVICE="$NOTIFY_SERVICEDESC"; STATE="$NOTIFY_SERVICESTATE"; OUTPUT="$NOTIFY_SERVICEOUTPUT
else
  SERVICE=""; STATE="$NOTIFY_HOSTSTATE"; OUTPUT="$NOTIFY_HOSTOUTPUT"
fi

# RECOVERY/UP/OK entwarnt, alles andere ist eine Stoerung.
case "$NOTIFY_NOTIFICATIONTYPE" in RECOVERY) TYPE="recovery" ;; *) TYPE="problem" ;; esac

# Die Ereignis-Kennung traegt die Idempotenz: Wiederholt CheckMK nach einem Zustellfehler,
# entsteht KEIN zweiter Vorgang. Sie muss die Meldung eindeutig benennen, nicht den Versuch.
EVENT_ID="checkmk-${NOTIFY_HOSTNAME}-${SERVICE}-${NOTIFY_MICROTIME:-$(date +%s)}"

payload=$(jq -n --arg id "$EVENT_ID" --arg h "$NOTIFY_HOSTNAME" --arg s "$SERVICE" \
                --arg sev "$STATE" --arg t "$TYPE" --arg m "$OUTPUT" --arg src "$ORDIVIS_SOU
  '{schemaVersion:1, sourceId:$src, events:[{eventId:$id, host:$h, service:$s,
     severity:$sev, eventType:$t, subject:("\($h): \($m)"), message:$m}]}')

curl -sS --max-time 20 -o /dev/null -w '%{http_code}' \
     -H "Authorization: Bearer $ORDIVIS_TOKEN" -H 'Content-Type: application/json' \
     -d "$payload" "$ORDIVIS_URL/api/v1/monitoring/events" | grep -q '^2' || exit 2
```

⚠️ **`severity` wird nicht übersetzt, sondern abgebildet.** CheckMK meldet `CRIT`, `WARN`, `OK`; die Zuordnung auf die Dringlichkeit steht in Ordivis an der Quelle (Abschnitt 3) und ist für CheckMK vorbelegt. Ein unbekannter Wert fällt auf die Vorgabe, nicht durch.

7.2 PRTG — *Execute HTTP Action*

🔑 `monitoring.event.ingest` — Diese Schritte laufen **in PRTG**, nicht in Ordivis IT. Aus Ordivis kommen nur zwei Angaben: das Integrations-Token mit diesem Recht und die `sourceId` der Quelle.

1. Öffnen Sie *Setup → Notification Templates → Add* und schalten Sie *Execute HTTP Action* ein.
2. Tragen Sie als **URL** `https://ordivis.example.local/api/v1/monitoring/events` ein und wählen Sie als **Method** `POST`.
3. Tragen Sie unter **Additional Headers** zwei Zeilen ein: `Authorization: Bearer <Integrations-Token>` und `Content-Type: application/json`.
4. Tragen Sie als **Postdata** ein:

```
json   {"schemaVersion":1,"sourceId":"…","events":[{   "eventId":"prtg-%sensorid-%datetime",
"host":"%device","service":"%name",                "severity":"%status","eventType":"problem",
```

`"subject":"%device: %name ist %status","message":"%message"}]}`

5. Speichern Sie die Vorlage und legen Sie sie ein **zweites Mal** an, diesmal mit `"eventType":"recovery"`.
6. Hängen Sie beide Vorlagen an Ihre Objekte: die erste als *Notification when Down*, die zweite als *when Up*.

✅ **Ergebnis:** Störungen und Entwarnungen laufen in dieselbe Kette.

> **Schritt 5 ist nicht optional.** PRTG trennt „when Down" von „when Up". Wer nur die erste Vorlage anlegt, bekommt Vorgänge, die nie von selbst zugehen — jede behobene Störung bliebe offen stehen, bis jemand den Vorgang von Hand schließt.
>
> ⚠️ `%datetime` **in der Ereignis-Kennung ist Absicht.** Ohne einen wechselnden Teil hielte Ordivis die zweite Meldung desselben Sensors für eine Wiederholung nach Zustellfehler und verwürfe sie — die Deduplizierung läuft über den Korrelationsschlüssel aus Gerät und Sensor, nicht über die Ereignis-Kennung.
>
> **Die Ratelimit-Richtlinie ist eine eigene** (600 Anfragen je Minute), nicht die allgemeine mit 30. Im Sturmfall — also genau dann, wenn die Meldungen zählen — liefe die Überwachung sonst in Abweisungen und verwürfe ihre Alarme. Der Sturm wird **fachlich** gebremst, nicht dadurch, dass die Leitung zumacht.

8 Die Alarm-Übersicht

> 💻 **Client:** *Service Desk* ▸ *Monitoring-Anbindung*, Registerkarten *Alarme* und *Wiederkehrend* · 🔑 `tickets.read`

Die Registerkarte *Alarme* zeigt offene, entwarnte und **unterdrückte** Störungen mit dem Grund der Unterdrückung. *Wiederkehrend* nennt die Korrelationsschlüssel, die in den letzten 30 Tagen mehrfach aufgetreten sind — die Kandidaten für das Problem-Management.

> **Warum diese Seite dem Service Desk offensteht und nicht nur der Administration.** Ein **unterdrückter** Alarm hat per Definition keinen Vorgang; in einer Vorgangsliste kann von ihm nichts stehen. Die Zusage aus Abschnitt 4 — „unterdrückt heißt nicht verschluckt" — lässt sich also nur hier nachprüfen. Wäre die Seite der Administration vorbehalten, könnte eine zu weit geratene Unterdrückungsregel außerhalb dieses Kreises niemandem auffallen.

So prüfen Sie turnusmäßig, ob die Anbindung noch liefert:

> 💻 **Client:** *Service Desk* ▸ *Monitoring-Anbindung*, Registerkarte *Quellen* · 🔑 `monitoring.source.manage`

1. Öffnen Sie *Service Desk* ▸ *Monitoring-Anbindung* und bleiben Sie auf der Registerkarte *Quellen*.
2. Lesen Sie je Quelle die Zahl der offenen Alarme und den **letzten Empfang** ab.

3. Vergleichen Sie den Wert mit der erwarteten Meldefrequenz der Quelle.

✅ **Ergebnis:** Sie erkennen eine tote Anbindung, bevor sie im Ernstfall auffällt.

> Der letzte Empfang ist der wichtigste Wert der Seite. Eine Anbindung, die seit drei Tagen nichts mehr liefert, sieht in jeder anderen Ansicht aus wie **Ruhe** — und ist in Wahrheit ein stiller Ausfall. ⚠️ Diese Kontrolle hängt an `monitoring.source.manage`: Der letzte Empfang steht bei der Quelle, und die Quellen sieht nur, wer sie verwalten darf. Wer die Überwachung der Überwachung breiter aufstellen will, muss dieses Recht vergeben — eine Ersatzansicht dafür gibt es nicht.

9 Aufbewahrung der Roh-Ereignisse

Jede eingehende Meldung wird im Wortlaut abgelegt — sie ist der Beleg, wenn der Betreiber der Überwachung und Ihr Haus sich über eine Störung uneinig sind. Diese Rohdaten wachsen aber mit der **Ereignismenge**, nicht mit der Vorgangsmenge: Eine Überwachung mit tausend Prüfungen schreibt im Sturmfall Zehntausende Zeilen am Tag.

Ein nächtlicher Lauf kürzt sie deshalb:

Was	Roh-Ereignisse der Überwachung
Aufbewahrung	**30 Tage**
Wann	täglich um 03:15 UTC, je Mandant
Auftrag	*Monitoring-Rohereignisse kürzen* (`ticketing.monitoring-event-retention`)

> **Was bleiben bleibt.** Die **Episoden** — also die Geschichte der Störungen mit Beginn, Entwarnung und Vorgangsbezug — werden **nicht** gekürzt. Und Ereignisse einer Episode, die noch **offen** ist oder in der Nachlaufzeit steht, bleiben ebenfalls stehen, auch wenn sie älter sind: Den Wortlaut ausgerechnet für einen noch laufenden Fall zu löschen, nähme Ihnen den Beleg im laufenden Streit. Eine flatternde Platte, die seit Wochen offen steht, behält damit ihre gesamte Vorgeschichte.

Der Auftrag steht wie jeder andere unter **Administration ▸ Aufträge**; Zeitpunkt und Aktivierung lassen sich dort ändern.

10 Rechte

Recht	Wer
`monitoring.event.ingest`	Das Integrations-Token der Quelle
`monitoring.source.manage`	Mandanten-Administration — die Registerkarte *Quellen*, der letzte Empfang und das Anlegen

Recht	Wer
`tickets.read`	Der Service Desk — Alarme, wiederkehrende Störungen, Kennzahlen, und damit der Menüpunkt selbst

Bewusst zwei Rechte: Wer eine Überwachung anbinden darf, muss nicht auch Störungen einspeisen dürfen — und ein Token soll nichts konfigurieren können.

Und bewusst **eine Seite für beide**: Die Alarme sind Tagesgeschäft des Service Desk, die Quellen sind Einrichtung. Sie zu trennen hieße, den unterdrückten Alarm von der Regel zu trennen, die ihn unterdrückt hat — und genau dieser Weg wird gegangen, wenn jemand fragt, warum zu einer Störung kein Vorgang entstanden ist.

Verwandte Themen

- Service Desk / Ticketing — Wartungsfenster, Automatisierungsregeln
- CMDB — die CI-Zuordnung über den Hostnamen