# 4.1 Störung und betroffenen Dienst eingrenzen

Bevor Prozesse, Ports oder Protokolldateien untersucht werden, muss eindeutig feststehen, **welche Funktion tatsächlich gestört ist**. Eine Meldung wie „Der Server funktioniert nicht“ reicht für eine gezielte Fehleranalyse nicht aus.

> **Grundsatz:**
> Zuerst das Fehlerbild und den Störungsumfang bestimmen. Erst danach mit technischen Prüfungen beginnen.

---

**Ziele dieser Seite**

Nach dieser Seite sollst du:

- eine ungenaue Störungsmeldung in ein überprüfbares Fehlerbild überführen können,
- den betroffenen Server und Dienst identifizieren können,
- den Störungsumfang systematisch eingrenzen können,
- zwischen Client-, Benutzer-, Netzwerk-, Dienst- und Anwendungsfehlern unterscheiden können,
- relevante Zeitangaben und Änderungen erfassen können,
- einen geeigneten Vergleichstest auswählen können,
- den Ausgangszustand vor Änderungen dokumentieren können.

---

## 1. Ruhig und hypothesenoffen beginnen

Bei einer Störung entsteht schnell der Wunsch, sofort einen Dienst neu zu starten oder eine bekannte Lösung auszuprobieren. Dadurch können jedoch wichtige Diagnoseinformationen verloren gehen.

Eine sinnvolle Haltung lautet:

1. Störungsmeldung nicht ungeprüft als Ursache übernehmen.
2. Beobachtungen von Vermutungen trennen.
3. Noch keine technische Ursache festlegen.
4. Zuerst den Umfang der Störung bestimmen.
5. Mit einfachen und möglichst lesenden Prüfungen beginnen.
6. Pro Arbeitsschritt nur eine wesentliche Veränderung durchführen.
7. Vor und nach jeder Maßnahme vergleichen.
8. Ergebnisse und Zeitpunkte dokumentieren.

| Aussage | Einordnung |
|---|---|
| „Das Intranet ist nicht erreichbar.“ | Beobachtung beziehungsweise Fehlerbild |
| „Der Webserver ist abgestürzt.“ | Noch unbestätigte Hypothese |
| „TCP-Port 443 antwortet nicht.“ | Technisches Prüfergebnis |
| „Der Webdienst befindet sich im Zustand `Stopped`.“ | Technisches Prüfergebnis |
| „Der Dienst wurde durch fehlenden Speicher beendet.“ | Mögliche Ursache, die durch Protokolle belegt werden muss |

> **Wichtig:** Eine Fehlermeldung beschreibt häufig nur die Stelle, an der ein Fehler sichtbar wird – nicht die eigentliche Ursache.

---

## 2. Aus einer Störungsmeldung ein konkretes Fehlerbild erstellen

Eine gute Fehlerbeschreibung beantwortet mindestens folgende Fragen:

| Frage | Beispiel |
|---|---|
| Was funktioniert nicht? | Anmeldung an der internen Webanwendung |
| Welche konkrete Aktion schlägt fehl? | Absenden des Anmeldeformulars |
| Welche Fehlermeldung erscheint? | HTTP-Statuscode 503 |
| Seit wann besteht das Problem? | Seit 31.07.2026 um 09:15 Uhr |
| Tritt der Fehler dauerhaft oder zeitweise auf? | Bei jedem Anmeldeversuch |
| Wer ist betroffen? | Alle Benutzer am Standort Berlin |
| Welche Clients sind betroffen? | Windows-, Linux- und macOS-Clients |
| Welche Funktion arbeitet weiterhin? | Startseite der Anwendung ist erreichbar |
| Wurde eine alternative Verbindung getestet? | Zugriff über einen zweiten Client ebenfalls fehlerhaft |
| Was wurde zuletzt verändert? | Aktualisierung der Anwendung am Vorabend |

**Unzureichende Beschreibung:**

> „Der Server geht nicht.“

**Verwertbare Beschreibung:**

>

> „Seit 09:15 Uhr erhalten alle geprüften Benutzer beim Anmelden an `https://intranet.example.local` den HTTP-Statuscode 503. Die Namensauflösung funktioniert und die Startseite ist erreichbar. Der Fehler tritt von mehreren Clients und Benutzerkonten aus auf.“

---

## 3. Störungsumfang mit einer Eingrenzungsmatrix bestimmen

| Prüfdimension | Mögliche Eingrenzung |
|---|---|
| Benutzer | Ein Benutzer, Benutzergruppe oder alle Benutzer |
| Endgerät | Ein Client, Gerätetyp oder alle Clients |
| Betriebssystem | Nur Windows, Linux, macOS oder alle Systeme |
| Standort | Ein Raum, Gebäude, Standort oder alle Standorte |
| Netzwerk | LAN, WLAN, VPN, Gastnetz oder alle Zugangswege |
| Anwendung | Eine Funktion, ein Modul oder vollständige Anwendung |
| Dienst | Ein Dienst, mehrere Dienste oder gesamter Server |
| Zeitpunkt | Dauerhaft, sporadisch, zu bestimmten Uhrzeiten |
| Daten | Einzelne Datei, Datensatz, Freigabe oder alle Daten |
| Berechtigung | Einzelnes Konto, Rolle, Gruppe oder alle Konten |
| Protokoll | HTTP, HTTPS, SMB, SSH, RDP oder Datenbankverbindung |
| Backend | Datenbank, Verzeichnisdienst, Speicher oder externe API |

### Beispiele für Schlussfolgerungen:

| Beobachtung | Wahrscheinlicher Untersuchungsbereich |
|---|---|
| Nur ein Benutzer ist betroffen | Konto, Profil, Berechtigung oder benutzerspezifische Daten |
| Nur ein Client ist betroffen | Clientkonfiguration, Cache, Zertifikat oder lokale Firewall |
| Alle Benutzer eines Standortes sind betroffen | Standortnetz, WAN, DNS, Routing oder Firewall |
| Alle Benutzer sind betroffen | Zentraler Dienst, Server, Backend oder globale Änderung |
| Nur ein Anwendungsmodul ist betroffen | Modulkonfiguration, Backend oder Berechtigung |
| Zugriff per IP-Adresse funktioniert, per Name nicht | DNS oder Namenskonfiguration |

| Beobachtung | Wahrscheinlicher Untersuchungsbereich |
| --- | --- |
| TCP-Port ist offen, Anwendung antwortet fehlerhaft | Anwendung, Backend, TLS oder Berechtigung |
| Fehler tritt nur unter hoher Last auf | Ressourcen, Grenzwerte, Verbindungspool oder Zeitüberschreitung |

Diese Zuordnungen sind **Arbeitshypothesen** und noch kein endgültiger Ursachenbeweis.

---

## 4. Betroffenen Server und Dienst eindeutig identifizieren

Vor der technischen Prüfung müssen folgende Angaben geklärt werden:

| Information | Beispiel |
| --- | --- |
| Angezeigter Dienstname | Interne Webanwendung |
| Technischer Dienstname | `intranet-api` |
| Prozessname | `intranet-api.exe` beziehungsweise `intranet-api` |
| Servername | `srv-app01.example.local` |
| IP-Adresse | `192.0.2.20` |
| Protokoll | HTTPS |
| Zielport | TCP 443 |
| Pfad oder URL | `https://intranet.example.local/login` |
| Backend | `srv-db01.example.local` |
| Verantwortliches System | Webserver, Anwendungsdienst oder Datenbank |
| Produktivstatus | Produktiv-, Test- oder Entwicklungssystem |

> Der sichtbare Name einer Anwendung, der Betriebssystem-Dienstname und der Prozessname können unterschiedlich sein.

**Beispiel einer Dienstkette:**

```
Client
  → DNS
  → Firewall oder Loadbalancer
  → Webserver
```

→ Anwendungsdienst

→ Datenbank

→ Verzeichnisdienst oder externe API

Die Störung kann an jeder Übergabestelle dieser Kette entstehen.

---

## 5. Basisinformationen des untersuchten Servers erfassen

Die folgenden Befehle verändern den Systemzustand normalerweise nicht.

| Aufgabe | Windows PowerShell | Linux | macOS |
|---|---|---|---|
| Rechnername | `[RO] hostname` | `[RO] hostname` | `[RO] hostname` |
| Vollständiger DNS-Name | `[RO] [System.Net.Dns]::GetHostEntry($env:COMPUTERNAME).HostName` | `[RO] hostname --fqdn` | `[RO] hostname -f` |
| Betriebssysteminformationen | `[RO] Get-ComputerInfo \| Select-Object WindowsProductName,WindowsVersion,OsBuildNumber` | `[RO][FILE] cat /etc/os-release` | `[RO] sw_vers` |
| Systemarchitektur | `[RO] $env:PROCESSOR_ARCHITECTURE` | `[RO] uname -m` | `[RO] uname -m` |
| Kernel beziehungsweise Systemversion | `[RO] systeminfo` | `[RO] uname -a` | `[RO] uname -a` |
| Letzter Systemstart | `[RO] (Get-CimInstance Win32_OperatingSystem).LastBootUpTime` | `[RO] uptime -s` | `[RO] sysctl -n kern.boottime` |
| Aktuelle Laufzeit | `[RO] (Get-Date) - (Get-CimInstance Win32_OperatingSystem).LastBootUpTime` | `[RO] uptime` | `[RO] uptime` |
| Aktuelle Uhrzeit | `[RO] Get-Date -Format o` | `[RO] date --iso-8601=seconds` | `[RO] date "+%Y-%m-%dT%H:%M:%S%z"` |
| Zeitzone | `[RO] Get-TimeZone` | `[RO] timedatectl status` | `[RO][PRIV] sudo systemsetup -gettimezone` |
| Angemeldete Benutzer | `[RO] quser` | `[RO] who` | `[RO] who` |

**Hinweise:**

- `hostname --fqdn` liefert nur dann ein sinnvolles Ergebnis, wenn Hostname und Namensauflösung korrekt konfiguriert sind.

- `hostname -f` kann unter macOS abhängig von der lokalen Namenskonfiguration fehlschlagen oder einen unerwarteten Namen liefern.
- `systeminfo` erzeugt eine umfangreiche Ausgabe. Für eine Dokumentation können relevante Zeilen gezielt ausgewählt werden.
- Zeitstempel müssen zusammen mit der Zeitzone dokumentiert werden, damit Client-, Server- und Netzwerkprotokolle richtig verglichen werden können.

---

## 6. Dienstnamen und Prozesse zunächst nur lesend suchen

Die Platzhalter `<SUCHBEGRIFF>`, `<DIENST>` und `<PROZESS>` müssen ersetzt werden.

| Aufgabe | Windows PowerShell | Linux | macOS |
|---|---|---|---|
| Dienst nach Namen suchen | `[RO] Get-Service -Name "*<SUCHBEGRIFF>*" -ErrorAction SilentlyContinue` | `[RO] systemctl list-unit-files --type=service \| grep -i --color=never "<SUCHBEGRIFF>"` | `[RO] launchctl list \| grep -i --color=never "<SUCHBEGRIFF>"` |
| Dienst nach Anzeigenamen suchen | `[RO] Get-Service -DisplayName "*<SUCHBEGRIFF>*" -ErrorAction SilentlyContinue` | Nicht getrennt vorhanden | Nicht getrennt vorhanden |
| Status eines Dienstes | `[RO] Get-Service -Name "<DIENST>"` | `[RO] systemctl status "<DIENST>" --no-pager` | `[RO] launchctl print "system/<DIENST>"` |
| Prozess nach Namen suchen | `[RO] Get-Process -Name "<PROZESS>" -ErrorAction SilentlyContinue` | `[RO] pgrep -a "<PROZESS>"` | `[RO] pgrep -alf "<PROZESS>"` |
| Prozesse ausführlich anzeigen | `[RO] Get-CimInstance Win32_Process \| Select-Object ProcessId,ParentProcessId,Name,CommandLine` | `[RO] ps -eo user,pid,ppid,lstart,stat,cmd` | `[RO] ps -axo user,pid,ppid,lstart,state,command` |

**Besonderheiten:**

- Windows unterscheidet zwischen internem Dienstnamen und sichtbarem Anzeigenamen.
- Bei systemd gehört die Endung `.service` zum vollständigen Unit-Namen, kann bei eindeutigen Namen aber häufig weggelassen werden.
- `launchctl list` zeigt nicht automatisch alle Konfigurationsdetails eines Dienstes.
- `launchctl print system/<DIENST>` erwartet das tatsächliche launchd-Label.
- Ein laufender Prozess beweist noch nicht, dass der Dienst auf dem erwarteten Port lauscht oder fachlich korrekt arbeitet.

> Auf dieser frühen Diagnosestufe sollte ein Dienst noch nicht neu gestartet werden. Zuerst müssen Status, Prozess, Port und relevante Protokolle erfasst

> werden.

## 7. Geeignete Vergleichstests durchführen

Ein Vergleichstest verändert jeweils nur **eine Prüfdimension**. Dadurch lässt sich erkennen, welche Eigenschaft mit dem Fehler zusammenhängt.

| Ausgangssituation | Sinnvoller Vergleich |
|---|---|
| Ein Benutzer meldet einen Fehler | Anderes Benutzerkonto am gleichen Client |
| Ein Client ist betroffen | Gleicher Benutzer an einem anderen Client |
| WLAN-Zugriff scheitert | Gleicher Client über LAN oder VPN |
| Zugriff per Servername scheitert | DNS-Ergebnis prüfen und mit dokumentierter Zieladresse vergleichen |
| Eine URL funktioniert nicht | Andere Funktion desselben Dienstes testen |
| Ein Standort ist betroffen | Zugriff von einem anderen Standort testen |
| Zugriff über Loadbalancer scheitert | Backend nur über den vorgesehenen administrativen Testweg prüfen |
| Anwendung meldet einen Fehler | Porttest und protokollspezifischen Funktionstest getrennt durchführen |
| Produktivsystem ist betroffen | Verhalten im freigegebenen Testsystem vergleichen |
| Fehler tritt zeitabhängig auf | Ergebnisse mit Last, Sicherung oder geplanten Aufgaben vergleichen |

**Regel für einen aussagekräftigen Vergleich:**

```
Gleicher Benutzer + anderer Client
oder
anderer Benutzer + gleicher Client
oder
gleicher Client + anderer Netzwerkweg
```

Werden mehrere Eigenschaften gleichzeitig verändert, ist das Ergebnis häufig nicht eindeutig.

> Der direkte Zugriff auf ein Backend kann Sicherheitsregeln, Authentifizierung, TLS-Prüfungen oder den Loadbalancer umgehen. Er darf nur über einen freigegebenen administrativen Testweg erfolgen.

---

## 8. Ersten Client-zu-Dienst-Test dokumentieren

Bevor auf dem Server weitergesucht wird, sollte der Fehler möglichst reproduziert und genau protokolliert werden.

| Prüfpunkte | Zu dokumentierende Information |
|---|---|
| Ausgangsgerät | Hostname, Betriebssystem und Netzwerkverbindung |
| Benutzerkontext | Betroffenes Konto oder Rolle, ohne Kennwort |
| Ziel | Servername, URL, Freigabe oder Anwendung |
| Zeitpunkt | Datum, Uhrzeit und Zeitzone |
| Aktion | Exakter Arbeitsschritt, der den Fehler auslöst |
| Ergebnis | Vollständige Fehlermeldung oder Statuscode |
| Dauer | Sofortiger Fehler oder Zeitüberschreitung |
| Wiederholbarkeit | Immer, sporadisch oder nur einmal |
| Vergleichstest | Zweiter Client, Benutzer oder Netzwerkweg |
| Beweismittel | Screenshot, Ereignis-ID, Logauszug oder Befehlsausgabe |

**Beispiel:**

```
Zeitpunkt:      2026-07-31T09:15:42+02:00
Client:         client-017
Betriebssystem: Windows 11
Benutzerrolle:  Standardbenutzer
Ziel:           https://intranet.example.local/login
Aktion:         Anmeldeformular abgesendet
Ergebnis:       HTTP 503
Dauer:          Antwort nach ungefähr 2 Sekunden
Wiederholbar:   Ja
Vergleich:      Gleicher Fehler auf einem zweiten Client
```

Sensible Inhalte wie Kennwörter, Sitzungscookies, Zugriffstoken, private Schlüssel oder vollständige personenbezogene Daten dürfen nicht in die Dokumentation übernommen werden.

---

## 9. Letzte Änderungen und zeitliche Zusammenhänge erfassen

Viele Dienstfehler entstehen nach einer technischen oder organisatorischen Änderung. Ein zeitlicher Zusammenhang ist jedoch noch kein Beweis für die Ursache.

Zu prüfen sind insbesondere:

- Betriebssystem- und Anwendungsupdates,
- Änderungen an Konfigurationsdateien,
- Zertifikatswechsel,
- Kennwortänderungen von Dienstkonten,
- Änderungen an Gruppen oder Berechtigungen,
- Firewall- und Proxyänderungen,
- DNS- oder Loadbalanceränderungen,
- Datenbankmigrationen,
- Speicher- und Volumeänderungen,
- Container- oder Image-Updates,
- Neustarts und Stromausfälle,
- Sicherungs- und Wartungsaufgaben,
- Änderungen an externen APIs,
- abgelaufene Lizenzen oder Zertifikate,
- neu eingerichtete Überwachungs- oder Sicherheitssoftware.

| Frage | Zweck |
|---|---|
| Was wurde geändert? | Technische Änderung bestimmen |
| Wann wurde es geändert? | Zeitliche Übereinstimmung prüfen |
| Auf welchem System? | Betroffene Komponente bestimmen |
| Wer oder welcher Prozess führte die Änderung aus? | Rückfragen und Audit ermöglichen |
| Wurde die Änderung getestet? | Fehlende Prüfungen erkennen |
| Gibt es einen Rückfallplan? | Wiederherstellung bewerten |
| Besteht der Fehler auch ohne diese Änderung? | Kausalität weiter untersuchen |

> „ Die Aussage „Seit dem Update besteht der Fehler“ ist eine wichtige Spur. Erst ein reproduzierbarer Zusammenhang oder ein passender

> Protokolleintrag macht daraus einen belastbaren Ursachenhinweis.

---

## 10. Wann die Eingrenzung abgeschlossen ist

Die erste Eingrenzung ist ausreichend, wenn folgende Punkte beantwortet sind:

- Welche konkrete Funktion ist gestört?
- Welcher Server oder Dienst stellt diese Funktion bereit?
- Welche Benutzer, Geräte und Standorte sind betroffen?
- Welche Bereiche sind ausdrücklich nicht betroffen?
- Seit wann besteht die Störung?
- Ist der Fehler reproduzierbar?
- Welche Fehlermeldung oder welcher Statuscode erscheint?
- Welche Vergleichstests wurden durchgeführt?
- Welche Änderungen gingen der Störung voraus?
- Welche technische Ebene wird als Nächstes geprüft?
- Wurde der Ausgangszustand dokumentiert?

**Beispiel einer abgeschlossenen Eingrenzung:**

> „Alle geprüften Benutzer können die Startseite des Intranets über HTTPS aufrufen. Beim Absenden des Anmeldeformulars antwortet die Anwendung jedoch mit HTTP 503. Der Fehler tritt von Windows-, Linux- und macOS-Clients sowie an zwei Standorten auf. DNS-Auflösung und Netzwerkzugriff funktionieren. Als Nächstes werden deshalb Dienststatus, Prozess und Backend-Abhängigkeiten auf dem Anwendungsserver geprüft.

---

## 11. Typische Fehler bei der ersten Eingrenzung

| Fehler | Folge | Bessere Vorgehensweise |
|---|---|---|
| Sofortiger Neustart | Flüchtige Diagnoseinformationen gehen verloren | Zustand und Protokolle zuerst sichern |
| Fehlermeldung als Ursache behandeln | Untersuchung beginnt an der falschen Stelle | Beobachtung und Ursache trennen |
| Nur einen Benutzer befragen | Störungsumfang bleibt unbekannt | Vergleich mit weiteren Benutzern durchführen |

| Fehler | Folge | Bessere Vorgehensweise |
|---|---|---|
| Nur `ping` verwenden | Dienstfunktion bleibt ungeprüft | Port- und Anwendungstest ergänzen |
| Mehrere Änderungen gleichzeitig | Wirkung einzelner Maßnahmen bleibt unklar | Änderungen einzeln durchführen |
| Servername nicht verifizieren | Falsches System wird untersucht | Ziel, IP-Adresse und Umgebung abgleichen |
| Test- und Produktivsystem verwechseln | Falsche Ergebnisse oder Betriebsrisiko | Umgebung eindeutig dokumentieren |
| Keine Zeitstempel erfassen | Protokolle lassen sich schlecht zuordnen | Datum, Uhrzeit und Zeitzone notieren |
| Nur Erfolg oder Fehler notieren | Diagnose ist nicht nachvollziehbar | Befehl, Ausgabe und Kontext dokumentieren |
| Geheimnisse in Tickets kopieren | Sicherheitsrisiko | Kennwörter, Token und Cookies entfernen |

## 12. Kompakte Checkliste für den Einsatz

- [ ] Konkrete gestörte Funktion erfasst
- [ ] Vollständige Fehlermeldung erfasst
- [ ] Datum, Uhrzeit und Zeitzone notiert
- [ ] Betroffene Benutzer bestimmt
- [ ] Betroffene Clients und Betriebssysteme bestimmt
- [ ] Betroffene Standorte und Netzwerkwege bestimmt
- [ ] Funktionierende Bereiche dokumentiert
- [ ] Fehler reproduziert
- [ ] Vergleichstest durchgeführt
- [ ] Servername und Umgebung verifiziert
- [ ] Dienstname, Prozess, Protokoll und Port zugeordnet
- [ ] Abhängige Systeme aufgenommen
- [ ] Letzte Änderungen erfragt
- [ ] Ausgangszustand vor Änderungen dokumentiert
- [ ] Keine Kennwörter, Token oder vertraulichen Daten gespeichert
- [ ] Nächster technischer Prüfschritt festgelegt

### Ergebnis dieser Diagnosestufe

Am Ende dieser Seite steht noch nicht zwingend die Ursache fest. Es muss jedoch klar sein:

```
Was ist gestört?
Wer oder was ist betroffen?
Seit wann besteht die Störung?
Welcher Dienst stellt die Funktion bereit?
Auf welchem System läuft er?
Wie lässt sich der Fehler reproduzieren?
Welche technische Ebene wird als Nächstes geprüft?
```

Die nächste Diagnosestufe ist die Überprüfung von **Dienststatus und Startart**.

---

**Weiterführende Quellen**

- Microsoft Learn – Get-Service
- Microsoft Learn – Get-CimInstance
- Microsoft Learn – Windows-Dienstprogramme und Befehle
- systemd – systemctl
- systemd – systemd.unit
- Apple – launchctl-Handbuchseite
- Apple – Aktivitätsanzeige – Benutzerhandbuch

---