7.13 Datenträger oder Dateisystem ist voll

7.13.1 Ziel dieser Seite

Diese Seite beschreibt die systematische Diagnose, wenn ein System, Datenträger, Volume, Dateisystem, Speicherpool, Container oder eine Anwendung keinen weiteren Speicherplatz verwenden kann.

Ziele der Diagnose:

- den tatsächlich betroffenen Pfad bestimmen;
- logische und physische Speicherebenen unterscheiden;
- belegte Datenblöcke, Inodes und Quotas getrennt prüfen;
- plötzliches und kontinuierliches Wachstum unterscheiden;
- versteckte Speicherverbraucher erkennen;
- eine kurzfristige Entlastung kontrolliert durchführen;
- die eigentliche Ursache nachweisen;
- einen erneuten Kapazitätsengpass verhindern.

Ein gemeldetes „Dateisystem voll“ bedeutet nicht zwangsläufig, dass die physische Festplatte vollständig belegt ist.

7.13.2 Sicherheitskennzeichnungen

Kennzeichnung	Bedeutung
LESEND	erfasst ausschließlich Zustände und Messwerte
LASTERZEUGEND	kann CPU, Arbeitsspeicher oder Datenträger-I/O deutlich belasten
ÄNDERND	verändert Konfigurationen oder Systemzustände
LÖSCHEND	entfernt Daten und kann Informationen unwiederbringlich zerstören
AUSFALLRISIKO	kann Dienste, Dateisysteme oder ganze Systeme unterbrechen

Grundregeln:

- Zuerst den exakten Zustand und den Zeitpunkt dokumentieren.
- Keine Dateien aufgrund ihres Namens löschen.
- Aktive Datenbanken, Container-Volumes und Anwendungsdateien nicht manuell bereinigen.
- Protokolle vor dem Löschen auswerten und bei Bedarf sichern.
- Snapshots nicht ohne Kenntnis ihrer Abhängigkeiten entfernen.

- Dateisystemprüfungen nicht ungeprüft auf produktiv eingebundenen Dateisystemen ausführen.
- Speicherpools und Thin-Provisioning-Metadaten besonders vorsichtig behandeln.
- Vor jeder Änderung müssen Risiko, Freigabe, Rückweg und Erfolgskriterium feststehen.

7.13.3 Mögliche Fehlerklassen

Fehlerklasse	Beschreibung	Typischer Nachweis
Datenblöcke voll	nutzbare Speicherkapazität ist erschöpft	`df`, `Get-Volume`, `diskutil info`
Inodes erschöpft	zu viele Dateien, obwohl noch Datenblöcke frei sind	`df -i`
Benutzer- oder Gruppenquota erreicht	einzelner Benutzer oder Dienst darf nichts mehr speichern	`quota`, FSRM, Dateisystemquota
Projekt- oder Verzeichnisquota erreicht	bestimmter Pfad besitzt ein eigenes Limit	XFS-Projektquota, FSRM
Anwendungsquota erreicht	Anwendung begrenzt Speicher unabhängig vom Dateisystem	Anwendungskonfiguration und Protokolle
Snapshot-Speicher voll	Snapshots oder Shadow Copies belegen den freien Bereich	VSS-, APFS-, Btrfs- oder ZFS-Auswertung
Thin Pool voll	zugrunde liegender Thin-Provisioning-Pool besitzt keinen freien Platz	LVM-, SAN- oder Hypervisoranzeige
Thin-Pool-Metadaten voll	Verwaltungsbereich des Thin Pools ist erschöpft	`lvs` mit `metadata_percent`
Container-Layer voll	beschreibbare Container-Schicht oder Containerhost ist voll	Docker- oder Runtime-Auswertung
Kubernetes-Ephemeralspeicher voll	Node oder Pod überschreitet Ephemeral-Storage-Grenzen	`DiskPressure`, Eviction-Ereignisse
Dateisystem schreibgeschützt	Dateisystem wurde wegen Fehlern nur lesbar eingebunden	Mountoptionen, Kernel- und Systemprotokolle
Physischer Datenträgerfehler	Schreibvorgänge scheitern durch Hardware- oder I/O-Fehler	SMART, Ereignisse, Kernelmeldungen
Gelöschte Datei noch geöffnet	Verzeichniseintrag ist gelöscht, Prozess hält Datei weiterhin geöffnet	`lsof +L1`
Reservierter Speicher	freier Bereich steht normalen Benutzern nicht zur Verfügung	Dateisystem- und Reservierungsdaten
Versteckte Daten unter Mountpoint	Dateien liegen unterhalb eines später darüber eingebundenen Dateisystems	Mountstruktur und Wartungsprüfung
Kapazität des Backends voll	lokaler Client zeigt nur einen vorgelagerten NAS-, SAN- oder Cloudspeicher	Backend-, Pool- und Quotaauswertung

---

7.13.4 Typische Symptome und Fehlermeldungen

Mögliche Symptome:

- Dateien können nicht mehr gespeichert werden.
- Anwendungen starten nicht oder stürzen ab.
- Datenbanken wechseln in einen Fehlerzustand.
- Updates und Paketinstallationen schlagen fehl.
- Benutzerprofile können nicht geladen werden.
- Protokolle werden nicht mehr geschrieben.
- Backupjobs brechen ab.
- Temporäre Dateien können nicht angelegt werden.
- Container starten wiederholt neu.
- Kubernetes-Pods werden wegen `DiskPressure` beendet.
- Ein Dateisystem wird plötzlich schreibgeschützt.
- Ein Volume wird mit `100 %` Belegung angezeigt.
- Ein Benutzer kann nicht schreiben, andere Benutzer jedoch schon.
- Eine Freigabe meldet „voll“, obwohl der Server noch freien Gesamtspeicher besitzt.

Typische Meldungen:

```
No space left on device
ENOSPC
Disk full
There is not enough space on the disk
Not enough disk space
Quota exceeded
EDQUOT
Read-only file system
EROFS
I/O error
DiskPressure
Evicted
ephemeral-storage
Thin pool is full
Metadata space exhausted
```

`No space left on device` kann sowohl auf erschöpfte Datenblöcke als auch auf erschöpfte Inodes oder interne Dateisystemreserven hinweisen.

`Quota exceeded` betrifft dagegen gewöhnlich ein festgelegtes Speicherlimit und nicht zwingend das gesamte Dateisystem.

7.13.5 Sofortmaßnahmen ohne unkontrolliertes Löschen

1. Exakte Fehlermeldung und Uhrzeit erfassen.
2. Betroffenen Host, Dienst, Benutzer, Container und Pfad bestimmen.
3. Kritikalität und Auswirkung bewerten.
4. Aktuelle Kapazität, Inodes, Quotas und Speicherpools dokumentieren.
5. Schreibintensive Prozesse und aktuelle Wachstumsrate identifizieren.
6. Bei akutem Wachstum eine kontrollierte Drosselung oder Beendigung des verursachenden Dienstes prüfen.
7. Keine unnötigen Neustarts durchführen.
8. Keine Protokolle, Datenbanken oder Container-Volumes spontan löschen.
9. Nur eindeutig entbehrliche und freigegebene Daten kontrolliert entfernen.
10. Nach jeder Maßnahme denselben Speichertest wiederholen.

Ein Neustart kann die Situation verschärfen, wenn ein Dienst beim Start temporären Speicher benötigt oder eine Datenbank wegen fehlenden Speicherplatzes keine Wiederherstellung durchführen kann.

7.13.6 Speicherpfad vollständig abbilden

Der sichtbare Pfad kann mehrere Speicherebenen durchlaufen:

```
Anwendung
→ Anwendungspfad
→ Container oder virtuelle Maschine
→ Mountpoint oder Laufwerksbuchstabe
→ Dateisystem
→ Partition oder logisches Volume
→ Thin Pool, RAID oder Storage Pool
→ physischer Datenträger, NAS oder SAN
```

Jede Ebene kann ein eigenes Limit besitzen.

Beispiel:

```
Anwendung schreibt nach /var/lib/app
→ Docker-Bind-Mount
→ /srv/containers/app
→ ext4-Dateisystem
→ LVM Logical Volume
→ LVM Thin Pool
```

```
→ RAID-Verbund
→ physische SSDs
```

Freier Speicher auf einer Ebene beweist nicht, dass alle anderen Ebenen ebenfalls freien Speicher besitzen.

7.13.7 Grundfragen der Diagnose

Zu klären sind:

- Welcher genaue Pfad kann nicht beschrieben werden?
- Welches Dateisystem enthält diesen Pfad?
- Ist das gesamte Dateisystem oder nur ein Benutzer betroffen?
- Sind Datenblöcke, Inodes oder eine Quota erschöpft?
- Ist das Dateisystem beschreibbar eingebunden?
- Welche Daten sind zuletzt gewachsen?
- Trat das Wachstum plötzlich oder kontinuierlich auf?
- Existieren Snapshots, Shadow Copies oder Container-Layer?
- Gibt es einen Thin-Provisioning-Pool?
- Liegt das eigentliche Speichersystem auf einem NAS oder SAN?
- Sind gelöschte Dateien weiterhin geöffnet?
- Sind Protokolle, Caches, Dumps, Backups oder temporäre Dateien beteiligt?
- Liegt zusätzlich ein Hardware- oder Dateisystemfehler vor?
- Welche Mindestkapazität benötigt der betroffene Dienst zur Wiederaufnahme?

7.13.8 Windows – Volumeauslastung erfassen

LESEND

```
Get-Volume |
  Select-Object `
    DriveLetter,
    FileSystemLabel,
    FileSystem,
    HealthStatus,
    OperationalStatus,
    @{
      Name = "SizeGiB"
      Expression = {
        [math]::Round($_.Size / 1GB, 2)
      }
```

```
  },
  @{
    Name = "FreeGiB"
    Expression = {
      [math]::Round($_.SizeRemaining / 1GB, 2)
    }
  },
  @{
    Name = "FreePercent"
    Expression = {
      if ($_.Size -gt 0) {
        [math]::Round(
          100 * $_.SizeRemaining / $_.Size,
          1
        )
      }
    }
  }
```

Dateisystemlaufwerke der aktuellen PowerShell-Sitzung:

```
Get-PSDrive `
  -PSProvider FileSystem
```

Lokale logische Laufwerke:

```
Get-CimInstance `
  -ClassName Win32_LogicalDisk `
  -Filter "DriveType=3" |
  Select-Object `
    DeviceID,
    VolumeName,
    FileSystem,
    Size,
    FreeSpace
```

`Get-PSDrive` kann auch eingebundene Dateisystemlaufwerke anzeigen. Bei Netzwerkfreigaben können Quotas oder serverseitige Einschränkungen dazu führen, dass die gemeldete Verfügbarkeit nicht der gesamten Backendkapazität entspricht.

7.13.9 Windows – Datenträger, Partition und Volume zuordnen

LESEND

```
Get-Disk |
  Select-Object `
    Number,
    FriendlyName,
    PartitionStyle,
    OperationalStatus,
    HealthStatus,
    Size
```

```
Get-Partition |
  Select-Object `
    DiskNumber,
    PartitionNumber,
    DriveLetter,
    Type,
    Size
```

```
Get-Partition |
  Get-Volume |
  Select-Object `
    DriveLetter,
    FileSystemLabel,
    FileSystem,
    Size,
    SizeRemaining
```

Von Speicherverwaltungsanbietern gemeldete physische Datenträger:

```
Get-PhysicalDisk |
  Select-Object `
    FriendlyName,
    MediaType,
    OperationalStatus,
    HealthStatus,
```

```
  Size
```

Zu beachten:

- `Get-Disk` bildet nicht jede mögliche Storage-Architektur vollständig ab.
- Hardware-RAID-Controller können nur ein virtuelles Laufwerk anzeigen.
- SAN-LUNs erscheinen als Datenträger, obwohl die physische Kapazität extern verwaltet wird.
- Storage Spaces, Cluster Shared Volumes und herstellerspezifische Pools müssen zusätzlich in ihrer jeweiligen Verwaltung geprüft werden.

7.13.10 Windows – große Verzeichnisse und Dateien eingrenzen

Eine rekursive Suche kann auf großen Dateisystemen erhebliche I/O-Last verursachen. Der Suchpfad muss deshalb möglichst eng begrenzt werden.

LESEND, LASTERZEUGEND

Größe der direkten Unterverzeichnisse eines bekannten Pfades:

```
$ScanRoot = "C:\Data"

Get-ChildItem `
  -LiteralPath $ScanRoot `
  -Directory `
  -Force `
  -ErrorAction SilentlyContinue |
ForEach-Object {
  $Measurement = Get-ChildItem `
    -LiteralPath $_.FullName `
    -File `
    -Force `
    -Recurse `
    -ErrorAction SilentlyContinue |
  Measure-Object `
    -Property Length `
    -Sum

  [pscustomobject]@{
    Path = $_.FullName
    SizeGiB = [math]::Round(
```

```
    $Measurement.Sum / 1GB,
    2
   )
  }
} |
Sort-Object `
  -Property SizeGiB `
  -Descending
```

Größte Dateien unter einem begrenzten Pfad:

```
Get-ChildItem `
  -LiteralPath "C:\Data" `
  -File `
  -Force `
  -Recurse `
  -ErrorAction SilentlyContinue |
Sort-Object `
  -Property Length `
  -Descending |
Select-Object `
  -First 20 `
  FullName,
  Length,
  LastWriteTime
```

Einschränkungen:

- fehlende Berechtigungen können Ergebnisse unvollständig machen;
- rekursive Suchen können produktive Datenträger belasten;
- Dateisystemmetadaten, Snapshots und geöffnete gelöschte Dateien erscheinen nicht zwingend in der Summe;
- Hardlinks und Deduplizierung können die Auswertung beeinflussen;
- ein vollständiger Lauf über ein großes Servervolume darf nicht unkontrolliert gestartet werden.

7.13.11 Windows – versteckte und reservierte Speicherverbraucher prüfen

VSS und Shadow Copies

LESEND

```
vssadmin list shadowstorage
```

```
vssadmin list shadows
```

Zu prüfen sind:

- verwendeter Shadow-Copy-Speicher;
- maximal erlaubter Speicher;
- betroffene Quell- und Speicher-Volumes;
- ungewöhnlich viele Snapshots;
- fehlgeschlagene Bereinigungen;
- Backupsoftware mit VSS-Nutzung.

Shadow Copies dürfen nicht spontan gelöscht oder verkleinert werden. Sie können für Wiederherstellungen, offene Backups oder andere Dienste benötigt werden.

Windows-Komponentenspeicher analysieren

LESEND, LASTERZEUGEND

```
Dism.exe /Online /Cleanup-Image /AnalyzeComponentStore
```

Die im Explorer angezeigte Größe von `WinSxS` darf nicht durch manuelles Löschen von Dateien korrigiert werden. Hardlinks können die scheinbare Verzeichnisgröße beeinflussen.

Systemdateien und weitere Bereiche

Zu prüfen sind:

- Papierkorb;
- temporäre Benutzer- und Systemdateien;
- Windows-Update-Dateien;
- Speicherabbilder;
- Ruhezustandsdatei;
- Auslagerungsdatei;
- VSS-Speicher;
- Anwendungsprotokolle;
- Installationspakete;
- Datenbank- und Transaktionsprotokolle;
- Benutzerprofile;
- Synchronisationsordner;
- lokale Cloudkopien;
- Backup- und Exportverzeichnisse.

Diese Dateien dürfen nicht allein aufgrund ihrer Größe entfernt werden.

7.13.12 Windows – Quotas prüfen

NTFS-Quota

LESEND, administrative Berechtigung erforderlich

```
fsutil quota query C:
```

File Server Resource Manager

```
Get-FsrmQuota |
  Select-Object `
    Path,
    Size,
    Usage,
    SoftLimit,
    Status,
    Template
```

Das FSRM-Modul ist nur vorhanden, wenn die entsprechende Windows-Server-Rolle beziehungsweise Verwaltungsfunktion installiert ist.

Zu unterscheiden sind:

- physisch freier Speicher des Volumes;
- NTFS-Benutzerquota;
- FSRM-Verzeichnisquota;
- Freigabe- oder Anwendungsquota;
- Cloud- oder Storage-Backendlimit.

Wenn nur ein Benutzer oder ein bestimmtes Verzeichnis betroffen ist, muss zuerst die wirksame Quota geprüft werden.

7.13.13 Windows – Storageereignisse auswerten

LESEND

```
Get-WinEvent `
  -FilterHashtable @{
```

```
    LogName = "System"
    StartTime = (Get-Date).AddHours(-6)
  } |
Where-Object {
  $_.ProviderName -in @(
    "disk",
    "Ntfs",
    "ReFS",
    "volsnap",
    "storport"
  )
} |
Select-Object `
  TimeCreated,
  ProviderName,
  Id,
  LevelDisplayName,
  Message
```

Zu korrelieren sind:

- Zeitpunkt des ersten Speicherfehlers;
- Volume oder Gerätename;
- Dateisystemfehler;
- I/O-Timeouts;
- zurückgesetzte Storageverbindungen;
- Shadow-Copy-Fehler;
- unerwartet entfernte Datenträger;
- RAID-, SAN- oder Multipath-Ereignisse;
- Anwendungs- und Datenbankfehler zum selben Zeitpunkt.

Ein Volume kann gleichzeitig nahezu voll und technisch fehlerhaft sein. Die Freigabe von Speicher behebt dann nicht zwangsläufig die eigentliche Ursache.

7.13.14 Linux – Dateisystembelegung erfassen

LESEND

```
df -hT
```

Wichtige Felder:

Feld	Bedeutung
Filesystem	zugrunde liegendes Gerät oder logischer Speicher
Type	Dateisystemtyp
Size	Gesamtkapazität
Used	belegter Speicher
Avail	verfügbarer Speicher
Use%	prozentuale Auslastung
Mounted on	Mountpoint

Bestimmten Pfad prüfen:

```
df -hT /var/lib/app
```

Exakte Blockwerte:

```
df -B1 /var/lib/app
```

`df` wertet das Dateisystem aus, in dem der angegebene Pfad liegt. Das ist zuverlässiger, als den Mountpoint nur anhand einer angenommenen Verzeichnisstruktur zu erraten.

7.13.15 Linux – Inodes prüfen

LESEND

```
df -i
```

Bestimmten Pfad prüfen:

```
df -i /var/lib/app
```

Eine mögliche Einordnung:

Blockbelegung	Inodebelegung	Einordnung
hoch	normal	große Dateien oder große Datenmengen
normal	hoch	sehr viele kleine Dateien

Blockbelegung	Inodebelegung	Einordnung
hoch	hoch	Kombination aus Datenmenge und Dateianzahl
normal	normal	Quota, Reservierung, anderer Mountpoint oder anderer Fehler

Bei erschöpften Inodes können keine neuen Dateien angelegt werden, obwohl noch Datenblöcke frei sind.

Typische Verursacher:

- Cachedateien;
- Sitzungsdateien;
- Mailqueues;
- kleine temporäre Dateien;
- extrahierte Paketbestände;
- Container-Layer;
- Build-Artefakte;
- nicht rotierte Protokolle mit sehr vielen Einzeldateien;
- Monitoring- oder Metrikdateien;
- Anwendungen mit fehlerhafter Dateibereinigung.

7.13.16 Linux – Mounts und Speichergeräte zuordnen

LESEND

```
findmnt
```

Gezielte Ausgabe:

```
findmnt \
  -o TARGET,SOURCE,FSTYPE,OPTIONS
```

Bestimmten Pfad zuordnen:

```
findmnt \
  --target /var/lib/app
```

Blockgeräte:

```
lsblk \
  -o NAME,TYPE,FSTYPE,SIZE,FSAVAIL,FSUSE%,MOUNTPOINTS
```

Zu prüfen sind:

- tatsächlicher Mountpoint;
- zugrunde liegendes Gerät;
- Dateisystemtyp;
- Read-only-Option `ro`;
- Bind-Mount;
- Overlay-Dateisystem;
- LVM;
- verschlüsseltes Volume;
- Netzwerkdateisystem;
- Container-Mount;
- unerwartet nicht eingebundenes Ziel.

Wenn ein vorgesehenes Dateisystem nicht eingebunden ist, kann eine Anwendung unbemerkt in das darunterliegende Root-Dateisystem schreiben.

7.13.17 Linux – große Verzeichnisse und Dateien finden

Die Suche sollte auf das betroffene Dateisystem begrenzt werden.

LESEND, LASTERZEUGEND

Direkte Unterverzeichnisse:

```
du \
  -x \
  -h \
  --max-depth=1 \
  /var |
sort -h
```

Zusammenfassung eines bestimmten Pfades:

```
du \
  -x \
  -s \
  -h \
```

```
  /var/lib/app
```

Größte Dateien mit GNU `find`:

```
find \
  /var/lib/app \
  -xdev \
  -type f \
  -printf '%s\t%p\n' |
sort -nr |
head -n 20
```

Wichtige Optionen:

Option	Bedeutung
`-x` bei `du`	bleibt im selben Dateisystem
`-xdev` bei `find`	überschreitet keine Dateisystemgrenze
`--max-depth=1`	wertet nur die direkte Verzeichnisebene aus
`-type f`	beschränkt die Suche auf reguläre Dateien

Die Option `-printf` ist eine GNU-`find`-Funktion und steht nicht auf jedem Unix-System zur Verfügung.

7.13.18 Unterschied zwischen `df` und `du` untersuchen

`df` und `du` messen unterschiedliche Dinge:

- `df` fragt die Belegung des gesamten Dateisystems ab.
- `du` summiert erreichbare Dateien und Verzeichnisse.

Wenn `df` eine hohe Belegung zeigt, `du` aber deutlich weniger Daten findet, sind insbesondere zu prüfen:

- gelöschte, weiterhin geöffnete Dateien;
- fehlende Berechtigungen bei der `du`-Auswertung;
- Snapshots;
- Dateisystemmetadaten;
- reservierte Blöcke;
- Daten unterhalb eines Mountpoints;
- Copy-on-Write- und Reflink-Daten;
- Deduplizierung;
- Container-Overlay-Layer;

- unterschiedliche Dateisystemgrenzen;
- Sparse Files;
- Quotadaten;
- beschädigte Dateisystemmetadaten.

Sparse Files vergleichen:

```
du -h /pfad/datei
```

```
du \
  -h \
  --apparent-size \
  /pfad/datei
```

Die scheinbare Dateigröße kann größer als der tatsächlich belegte Speicher sein.

7.13.19 Gelöschte, aber weiterhin geöffnete Dateien prüfen

Unter Unix-ähnlichen Systemen wird der belegte Speicher einer gelöschten Datei erst freigegeben, wenn kein Prozess mehr einen offenen Dateideskriptor auf sie hält.

LESEND

```
sudo lsof +L1
```

Zu prüfen sind:

- Prozessname;
- Prozess-ID;
- Dateisystem;
- Dateigröße;
- gelöschter Dateipfad;
- Dienstabhängigkeiten;
- Möglichkeit eines kontrollierten Reloads oder Neustarts.

Ein Neustart des gesamten Systems ist nicht die erste Maßnahme. Wenn die Ursache bestätigt ist, sollte der betroffene Dienst kontrolliert neu geladen oder neu gestartet werden. Dabei sind Auswirkung und Wiederanlaufbedingungen zu prüfen.

7.13.20 Linux – Journald und Protokollwachstum prüfen

Aktuelle Journalbelegung:

LESEND

```
journalctl --disk-usage
```

Journaldateien und Protokollverzeichnisse:

```
du \
  -x \
  -h \
  --max-depth=1 \
  /var/log |
sort -h
```

Logrotate-Konfiguration im Debugmodus prüfen:

```
sudo logrotate \
  -d \
  /etc/logrotate.conf
```

`logrotate -d` führt keine Rotation aus, zeigt aber die geplante Verarbeitung.

Zu prüfen sind:

- ungewöhnlich hohe Fehlerrate;
- wiederholte identische Meldungen;
- deaktivierte oder fehlerhafte Rotation;
- Anwendungen mit eigener Logverwaltung;
- Debug- oder Trace-Level;
- fehlende Aufbewahrungsgrenze;
- gelöschte, noch geöffnete Protokolldateien;
- Komprimierungsfehler;
- falsche Dateiberechtigungen;
- voller Zielpfad der Rotation.

Das Löschen eines Protokolls behebt nicht den Prozess, der es unkontrolliert erzeugt.

7.13.21 Linux – Benutzer-, Gruppen- und Projektquotas prüfen

Quota des aktuellen Benutzers:

LESEND

```
quota -s
```

Quotaübersicht vorhandener quota-fähiger Dateisysteme:

```
sudo repquota -a
```

XFS-Quotaauswertung:

```
sudo xfs_quota \
  -x \
  -c 'report -h' \
  /mountpoint
```

Zu unterscheiden sind:

- Soft Limit;
- Hard Limit;
- Grace Period;
- Blockquota;
- Inodequota;
- Benutzerquota;
- Gruppenquota;
- XFS-Projektquota.

Eine noch laufende Grace Period kann erklären, warum Schreiben zunächst möglich war und später ohne sichtbare Änderung der Gesamtkapazität fehlschlägt.

7.13.22 LVM und Thin Provisioning prüfen

LESEND

```
sudo pvs
```

```
sudo vgs
```

```
sudo lvs \
  -a \
  -o lv_name,vg_name,lv_size,pool_lv,origin,data_percent,metadata_percent
```

Besonders kritisch sind:

- `data_percent` nahe der Poolkapazität;
- `metadata_percent` nahe der Metadatenkapazität;
- erschöpfter freier Bereich der Volume Group;
- viele oder stark gewachsene Snapshots;
- ein Logical Volume, das kleiner als der zugrunde liegende Pool ist;
- ein vergrößertes Blockgerät, dessen Partition, Logical Volume oder Dateisystem nicht mitgewachsen ist.

Ein Thin Pool kann voll sein, obwohl ein Gastbetriebssystem oder Logical Volume scheinbar noch freien logischen Speicher anzeigt.

Das Erweitern eines Speichers muss in der richtigen Reihenfolge erfolgen:

```
Backend oder physischer Datenträger
→ Partition oder PV
→ Volume Group oder Pool
→ Logical Volume
→ Dateisystem
→ Anwendung
```

Nicht jede Ebene ist in jeder Architektur vorhanden.

7.13.23 Btrfs und ZFS prüfen

Btrfs

LESEND

```
sudo btrfs filesystem usage -T /mountpoint
```

```
sudo btrfs subvolume list /mountpoint
```

Zu prüfen sind:

- Datenbelegung;
- Metadatenbelegung;
- Systembereich;
- Global Reserve;
- RAID- oder Redundanzprofil;
- fehlende Geräte;
- Snapshots und Subvolumes;
- geschätzter freier Speicher.

Bei Btrfs können Daten- und Metadatenbereiche unterschiedlich ausgelastet sein. Eine einfache `df`-Ausgabe reicht deshalb nicht immer für die vollständige Einordnung.

ZFS

```
zpool list
```

```
zfs list -o space
```

Snapshots:

```
zfs list \
  -t snapshot \
  -o name,used,refer,creation
```

Zu prüfen sind:

- Poolkapazität;
- Dataset-Quota;
- Reservation und Refreservation;
- Snapshotbelegung;
- Kinder-Datasets;
- tatsächlich verfügbarer Speicher;
- Poolzustand.

Snapshots und Copy-on-Write-Daten können Speicher belegen, obwohl aktuelle Dateien bereits gelöscht wurden.

7.13.24 macOS – Speicherzustand prüfen

LESEND

```
df -h
```

Bestimmten Pfad prüfen:

```
df -h /System/Volumes/Data
```

Datenträger und Partitionen:

```
diskutil list
```

Informationen zum Startvolume:

```
diskutil info /
```

APFS-Struktur:

```
diskutil apfs list
```

APFS-Snapshots:

```
diskutil apfs listSnapshots /
```

Lokale Time-Machine-Snapshots:

```
tmutil listlocalsnapshots /
```

Große Verzeichnisse unterhalb des Datenvolumes:

LESEND, LASTERZEUGEND

```
sudo du \
  -x \
  -h \
  -d 1 \
  /System/Volumes/Data \
  2>/dev/null |
sort -h
```

Gelöschte, noch geöffnete Dateien:

```
sudo lsof +L1
```

Bei APFS teilen sich mehrere Volumes den freien Speicher eines gemeinsamen Containers. Daher müssen Volume, Container, Quotas, Reservierungen und Snapshots gemeinsam betrachtet werden.

Die macOS-Anzeige „verfügbar“ kann zusätzlich löschbaren Speicher enthalten. Dieser ist nicht mit unmittelbar freiem Speicher in jeder konkreten Betriebssituation gleichzusetzen.

7.13.25 Docker – Speicherverbrauch prüfen

Docker-Root-Verzeichnis:

LESEND

```
docker info \
  --format '{{.DockerRootDir}}'
```

Docker-Speicherübersicht:

```
docker system df
```

Detaillierte Übersicht:

```
docker system df -v
```

Containergrößen:

```
docker ps \
  -a \
  --size
```

Volumes:

```
docker volume ls
```

Logpfade der Container:

```
docker ps -aq |
xargs -r docker inspect \
  --format '{{.Name}} {{.LogPath}}'
```

Mounts eines bestimmten Containers:

```
docker inspect \
  --format '{{json .Mounts}}' \
  <Containername>
```

Zu prüfen sind:

- Docker-Root-Dateisystem;
- beschreibbare Container-Layer;
- Images;
- gestoppte Container;
- Build-Cache;
- benannte und anonyme Volumes;
- Bind-Mounts;
- Containerprotokolle;
- Anwendungscaches;
- Datenbanken in Volumes;
- fehlende Logrotation.

Nicht zulässig als spontane Diagnosemaßnahme:

```
docker system prune -a --volumes
```

Dieser Befehl kann gestoppte Container, ungenutzte Images, Netzwerke, Build-Cache und anonyme Volumes entfernen. Vor einer Bereinigung müssen alle betroffenen Objekte einzeln geprüft werden.

Das manuelle Löschen innerhalb von `/var/lib/docker` kann die Docker-Metadaten und Containerdaten beschädigen.

7.13.26 Kubernetes – Ephemeral Storage und DiskPressure prüfen

Nodezustand:

LESEND

```
kubectl get nodes
```

Bestimmten Node untersuchen:

```
kubectl describe node <Node>
```

Zu prüfen sind insbesondere:

```
DiskPressure
NodeHasDiskPressure
ephemeral-storage
imagefs.available
imagefs.inodesFree
nodefs.available
nodefs.inodesFree
```

Ereignisse:

```
kubectl get events \
  --all-namespaces \
  --sort-by=.metadata.creationTimestamp
```

Betroffenen Pod untersuchen:

```
kubectl describe pod \
  <Pod> \
  -n <Namespace>
```

Persistente Volumes:

```
kubectl get pvc \
  --all-namespaces
```

Die angezeigte PVC-Kapazität ist nicht automatisch der aktuell freie Speicher innerhalb des eingebundenen Dateisystems.

Lokaler Ephemeral Storage kann unter anderem umfassen:

- beschreibbare Container-Layer;
- `emptyDir`-Volumes;
- Node- und Containerprotokolle;

- Image-Speicher;
- temporäre Anwendungsdaten.

Bei Node-Druck kann der Kubelet Pods beenden, um Ressourcen zurückzugewinnen. Das Löschen einzelner Pods behebt die Ursache nicht, wenn Logs, Images oder Anwendungen sofort erneut denselben Speicherverbrauch erzeugen.

7.13.27 Anwendungen und Dienste als Verursacher prüfen

Häufige Speicherverbraucher:

Bereich	Mögliche Ursache
Protokolle	fehlende Rotation, Debugmodus, Fehlerschleife
Datenbank	Transaktionslog, WAL, Temp-Bereich, Replikationsverzug
Backup	fehlende Retention, doppelte Sicherungen, abgebrochene Jobs
Container	unbeschränkte Logs, alte Images, beschreibbare Layer
Monitoring	hochauflösende Metriken, zu lange Aufbewahrung
Mail	Queue, Anhänge, Quarantäne
Druck	blockierte Spooldateien
Anwendung	Cache, Sitzungen, Uploads, Exporte
Betriebssystem	Updates, Dumps, temporäre Dateien
Entwicklung	Build-Artefakte, Paketcache, Testdaten
Virtualisierung	Snapshots, virtuelle Festplatten, Replikation
Dateidienst	Benutzerdateien, Papierkorb, Versionierung
Security	EDR-Quarantäne, Scanprotokolle, Forensikdaten

Zu prüfen sind:

- Besitzer der Daten;
- Erstellungs- und Änderungszeit;
- Wachstum pro Stunde oder Tag;
- vorgesehene Retention;
- aktive Prozesse;
- Abhängigkeit zu laufenden Diensten;
- Backupstatus;
- Replikationsstatus;
- Wiederherstellungsbedarf;
- Datenschutz und Aufbewahrungspflichten.

7.13.28 Netzwerkfreigaben, NAS und SAN prüfen

Bei SMB, NFS, iSCSI, Fibre Channel oder Cloudspeicher müssen Client und Backend getrennt geprüft werden.

Zu erfassen sind:

- eingebundener Pfad;
- verwendetes Protokoll;
- Server oder Storageziel;
- Share- oder Exportname;
- LUN;
- Dateisystem;
- Storage Pool;
- Volume;
- Snapshotbestand;
- Benutzer- oder Verzeichnisquota;
- Thin-Provisioning-Kapazität;
- Replikations- und Reservebereiche;
- tatsächlich freier physischer Speicher.

Mögliche Sonderfälle:

- nur eine Freigabequota ist erreicht;
- ein einzelner Benutzer hat sein Limit erreicht;
- das NAS-Volume ist voll, der Storage Pool aber nicht;
- der Pool ist voll, obwohl das Volume noch logischen Speicher meldet;
- Snapshots verhindern die Freigabe gelöschter Daten;
- eine Thin-LUN ist logisch größer als der physisch verfügbare Pool;
- das Dateisystem wurde wegen Backendfehlern schreibgeschützt;
- ein nicht eingebundenes Netzwerkziel führte zu lokalen Schreibvorgängen.

Herstellerspezifische Lösch-, Snapshot- und Poolbefehle dürfen erst nach Prüfung der jeweiligen Dokumentation verwendet werden.

7.13.29 Schreibgeschütztes Dateisystem und Hardwarefehler abgrenzen

Linux-Mountstatus:

LESEND

```
findmnt \
  -o TARGET,SOURCE,FSTYPE,OPTIONS
```

Aktuelle Kernelmeldungen:

```
sudo dmesg \
  --ctime |
grep -Ei \
  'error|i/o|filesystem|read-only|corrupt|nvme|ata|scsi|ext4|xfs|btrfs'
```

Systemprotokolle:

```
journalctl \
  -k \
  --since "-2 hours"
```

Falls `smartctl` installiert und für das Gerät geeignet ist:

```
sudo smartctl \
  -a \
  /dev/<Gerät>
```

macOS-Verifikation:

```
diskutil verifyVolume /
```

Ein Dateisystem kann wegen erkannter Fehler auf `read-only` wechseln. In diesem Fall ist das Löschen von Dateien weder möglich noch die richtige Erstmaßnahme.

Offline-Dateisystemprüfungen wie `fsck` dürfen nicht ungeprüft auf einem produktiv eingebundenen Dateisystem ausgeführt werden.

Wenn ein Datenträger einen bevorstehenden Ausfall meldet, haben Datensicherung, kontrollierte Außerbetriebnahme und Austausch Vorrang vor einer reinen Speicherbereinigung.

7.13.30 Hypothese und Gegenbeweis formulieren

Beispiel:

```
Hypothese:
Das Root-Dateisystem ist voll, weil ein Dienst ein
Protokoll ohne funktionierende Rotation erzeugt.
```

Erwarteter Befund:
df zeigt eine hohe Blockbelegung.
Das Protokollverzeichnis enthält eine stark gewachsene Datei.
Die Änderungszeit und die Dienstmeldungen passen zum Fehlerzeitpunkt.

Gegenbeweis:
Das Protokoll ist klein oder wächst nicht.
Die Belegung bleibt auch nach kontrolliertem Stoppen
des Dienstes unverändert.

Testmethode:
Dateisystembelegung, Verzeichnisgröße, Änderungszeit,
offene Dateien und Dienstprotokolle vergleichen.

Erfolgskriterium:
Der verursachende Schreibpfad ist eindeutig nachgewiesen.

Eine belastbare Hypothese enthält:

- vermutete Ursache;
- betroffene Speicherebene;
- erwarteten Messwert;
- möglichen Gegenbeweis;
- sichere Testmethode;
- Risiko;
- Erfolgskriterium.

7.13.31 Kontrollierte Maßnahmen

Maßnahme	Voraussetzung	Risiko
unnötige temporäre Daten entfernen	Eigentümer und Entbehrlichkeit bestätigt	benötigte Sitzungs- oder Arbeitsdaten können verloren gehen
Protokollrotation korrigieren	Logwachstum nachgewiesen	laufender Dienst kann Logdatei weiter offen halten
Journald begrenzen	Journal als Ursache bestätigt	ältere Diagnoseinformationen gehen verloren
Anwendungscache bereinigen	Cache ist dokumentiert wiederherstellbar	Lastspitze beim Neuaufbau
Backupretention korrigieren	abgelaufene Sicherungen eindeutig bestimmt	Wiederherstellungspunkte gehen verloren

Maßnahme	Voraussetzung	Risiko
Containerobjekte bereinigen	jedes Objekt auf Nutzung geprüft	Daten oder Rollbackimages können fehlen
Quota anpassen	Quota als Ursache und Kapazität vorhanden	unkontrolliertes weiteres Wachstum
Volume erweitern	Backendkapazität und korrekte Ebenen bestätigt	Partitions- oder Dateisystemschaden bei Fehlern
Thin Pool erweitern	Poolgrenze nachgewiesen	falsche Erweiterungsebene bleibt wirkungslos
Snapshot entfernen	Abhängigkeiten und Retention geprüft	Wiederherstellungspunkt geht verloren
Dienst kontrolliert neu starten	gelöschte offene Datei nachgewiesen	Dienstunterbrechung
fehlerhaften Schreibprozess stoppen	akutes Wachstum bestätigt	Anwendungs- oder Datenverlust
Storage austauschen	Hardwarefehler bestätigt	Betriebsunterbrechung und Migrationsrisiko

Vor der Maßnahme sind zu dokumentieren:

- Ausgangszustand;
- betroffener Pfad;
- aktuelle Belegung;
- erwartete Freigabe;
- Datenverantwortlicher;
- Backupstatus;
- Freigabe;
- Risiko;
- Rückweg;
- Erfolgskriterium.

7.13.32 Beispiele für verändernde Bereinigungen

Die folgenden Befehle sind keine Erstdiagnose und dürfen nur nach Prüfung und Freigabe verwendet werden.

Systemd-Journal kontrolliert begrenzen

LÖSCHEND

```
sudo journalctl \
  --rotate \
  --vacuum-size=<Zielgröße>
```

Dabei werden archivierte Journaldateien bis zur angegebenen Zielgröße bereinigt. Relevante Protokolle müssen vorher ausgewertet oder gesichert werden.

Windows-Komponentenspeicher bereinigen

ÄNDERND

```
Dism.exe /Online /Cleanup-Image /StartComponentCleanup
```

Dieser Befehl darf nur nach der vorherigen Analyse mit `/AnalyzeComponentStore` und unter Berücksichtigung laufender Wartungs- oder Updatevorgänge verwendet werden.

Docker-Bereinigung

Vor jedem Prune-Befehl müssen mindestens folgende Ausgaben geprüft werden:

```
docker system df -v
```

```
docker ps -a
```

```
docker image ls
```

```
docker volume ls
```

Prune-Befehle sind löschend. Insbesondere `--volumes` darf nicht verwendet werden, ohne jedes betroffene Volume und dessen Datenverantwortung geprüft zu haben.

7.13.33 Maßnahmen, die nicht spontan ausgeführt werden dürfen

```
rm -rf auf unbekannten Verzeichnissen
Löschen unter /var/lib/docker
docker system prune -a --volumes
Löschen aktiver Datenbankdateien
Löschen von WAL- oder Transaktionslogs
Löschen aller Snapshots
vssadmin delete shadows
unkontrolliertes Verkleinern des VSS-Speichers
manuelles Löschen aus WinSxS
```

Leeren produktiver Protokolle ohne Sicherung

fsck auf einem schreibend eingebundenen Dateisystem

Ändern reservierter ext4-Blöcke ohne Kapazitätsplanung

Vergrößern einer Partition ohne Backup und Ebenenprüfung

Entfernen von Kubernetes-PVCs

Löschen unbekannter Container-Volumes

Deaktivieren von Quotas ohne Ursachenanalyse

Neustart des gesamten Hosts als erste Maßnahme

Eine Datei mit der Endung `.log`, `.tmp`, `.bak` oder `.old` ist nicht automatisch entbehrlich.

7.13.34 Vollständiger Diagnoseablauf

1. **Fehlermeldung vollständig aufnehmen**
   Wortlaut, Anwendung, Benutzer und Zeitpunkt dokumentieren.
2. **Betroffenen Pfad bestimmen**
   Exakten Datei-, Volume-, Mount- oder Containerpfad erfassen.
3. **Auswirkung bestimmen**
   Einzelne Anwendung, Benutzer, Host oder Standort unterscheiden.
4. **Schreibtest nicht unkontrolliert wiederholen**
   Zusätzliche Schreibvorgänge können die Lage verschärfen.
5. **Speicherebene bestimmen**
   Anwendung, Container, Dateisystem, Volume, Pool und Backend zuordnen.
6. **Gesamtkapazität und freien Speicher erfassen**
   Werte in Prozent und absoluten Größen dokumentieren.
7. **Inodes prüfen**
   Besonders bei vielen kleinen Dateien.
8. **Quotas prüfen**
   Benutzer-, Gruppen-, Projekt-, Verzeichnis- und Anwendungsquotas unterscheiden.
9. **Schreibschutz prüfen**
   Read-only-Mount oder Dateisystemfehler ausschließen.
10. **Speicherpool prüfen**
    RAID, LVM, Thin Pool, Storage Spaces, NAS oder SAN berücksichtigen.
11. **Snapshots prüfen**
    VSS, APFS, Btrfs, ZFS, Hypervisor und Storage-Snapshots erfassen.
12. **Wachstumsrate bestimmen**
    Aktuelle Werte mit Monitoring oder früheren Messungen vergleichen.
13. **Große Verzeichnisse eingrenzen**
    Suche auf das betroffene Dateisystem beschränken.
14. **Große Dateien bestimmen**
    Besitzer, Zweck und Änderungszeit dokumentieren.
15. **Viele kleine Dateien berücksichtigen**
    Inodeverbrauch und Dateianzahl untersuchen.

16. **Gelöschte offene Dateien prüfen**
    Besonders nach manueller Loglöschung.
17. **Protokolle und Rotation prüfen**
    Fehlerschleifen, Debugmodus und Retention untersuchen.
18. **Datenbanken prüfen**
    Daten-, Transaktions-, WAL- und Temp-Bereiche unterscheiden.
19. **Backupdaten prüfen**
    Retention, abgebrochene Jobs und doppelte Sicherungen untersuchen.
20. **Containerdaten prüfen**
    Images, Layer, Volumes, Bind-Mounts und Logs unterscheiden.
21. **Kubernetes-Ephemeralspeicher prüfen**
    Nodefs, Imagefs, Inodes, Limits und Evictions auswerten.
22. **Betriebssystembereiche prüfen**
    Updates, Dumps, Cache, temporäre Dateien und Papierkorb untersuchen.
23. **Hardware- und I/O-Fehler prüfen**
    Ereignisse, Kernelmeldungen und Storagezustand korrelieren.
24. **Hypothese und Gegenbeweis formulieren**
    Ursache vor jeder Änderung messbar beschreiben.
25. **Akutes Wachstum kontrollieren**
    Verursachenden Schreibprozess nur mit Freigabe drosseln oder stoppen.
26. **Minimale sichere Entlastung planen**
    Nur bestätigte, entbehrliche Daten auswählen.
27. **Eine Maßnahme durchführen**
    Nicht mehrere Variablen gleichzeitig verändern.
28. **Freien Speicher erneut messen**
    Absoluten Wert und Prozentwert dokumentieren.
29. **Ursprünglichen Schreibvorgang testen**
    Anwendung und nicht nur Diagnosewerkzeug prüfen.
30. **Weitere Ebenen verifizieren**
    Pool, Volume, Dateisystem und Anwendung erneut kontrollieren.
31. **Wachstum weiter beobachten**
    Prüfen, ob der Speicher sofort wieder abnimmt.
32. **Dauerhafte Ursache beheben**
    Rotation, Retention, Quota, Kapazität oder Anwendung korrigieren.
33. **Monitoring verbessern**
    Prozentwert, freien Absolutwert, Inodes und Wachstumsrate überwachen.
34. **Vorgang dokumentieren**
    Ursache, Maßnahme, Risiko, Ergebnis und Prävention festhalten.

7.13.35 Befundmatrix

Befund	Mögliche Einordnung	Nächster Nachweis
`df` zeigt 100 %, `du` ebenfalls sehr hoch	sichtbare Dateien belegen das Dateisystem	größte Verzeichnisse bestimmen

Befund	Mögliche Einordnung	Nächster Nachweis
`df` zeigt 100 %, `du` deutlich weniger	offene gelöschte Dateien, Snapshots oder Metadaten	`lsof +L1` und Snapshotprüfung
Blöcke frei, Inodes 100 %	zu viele Dateien	Verzeichnisse mit vielen kleinen Dateien bestimmen
nur ein Benutzer betroffen	Benutzerquota	Quota des Benutzers prüfen
nur ein Verzeichnis betroffen	FSRM-, Projekt- oder Anwendungsquota	wirksames Verzeichnislimit prüfen
Dateisystem ist `read-only`	Dateisystem- oder Storagefehler	Kernel- und Ereignisprotokolle
Root-Dateisystem voll	Logs, Cache, Container oder fehlender Mount	`/var`, `/tmp`, Docker und Mounts prüfen
nach Loglöschung keine Freigabe	Datei wird noch geöffnet gehalten	`lsof +L1`
Dockerhost voll	Images, Layer, Volumes oder Logs	`docker system df -v`
Container meldet voll, Host nicht	Container-Layer, Volume oder Quota	Mounts und beschreibbaren Layer prüfen
Kubernetes-Node zeigt `DiskPressure`	Nodefs, Imagefs oder Inodes knapp	Nodebeschreibung und Ereignisse
PVC vorhanden, Anwendung meldet voll	Dateisystem innerhalb des PVC oder Storagequota	Volume im Pod und Backend prüfen
Logical Volume hat Platz, Thin Pool ist voll	Thin Provisioning erschöpft	`lvs` mit Daten- und Metadatenprozent
Dateien wurden gelöscht, Snapshotgröße steigt	Copy-on-Write hält alte Blöcke	Snapshotbelegung prüfen
Windows-Volume voll, Verzeichnissumme kleiner	VSS, Systemdateien oder unzugängliche Bereiche	VSS und Systembereiche prüfen
macOS zeigt widersprüchliche Werte	APFS-Container, Snapshots oder löschbarer Speicher	APFS-Container und Snapshots prüfen
NAS-Freigabe voll, NAS-Pool hat Platz	Share-, Benutzer- oder Volumequota	Backendquota prüfen
Speicher wächst sehr schnell	Fehlerschleife oder unkontrollierter Job	Änderungszeit und aktive Prozesse
nach Bereinigung sofort wieder voll	Ursache weiterhin aktiv	Wachstumsquelle erneut messen
I/O-Fehler zusätzlich zu wenig Speicher	Hardware- oder Pfadfehler	Storagezustand und Ereignisse

---

7.13.36 Typische Diagnosefehler

- Nur den prozentualen Wert betrachten.
- Den absolut freien Speicher nicht dokumentieren.
- Inodes nicht prüfen.
- Quotas nicht berücksichtigen.
- Anwendungspfad und tatsächliches Dateisystem verwechseln.

- Ein nicht eingebundenes Ziel übersehen.
- `df` und `du` als identische Messungen behandeln.
- Fehlende Berechtigungen bei rekursiven Suchen ignorieren.
- Gelöschte offene Dateien übersehen.
- Snapshots und Shadow Copies ignorieren.
- Thin Provisioning nur im Gastbetriebssystem prüfen.
- Container-Layer und Volume verwechseln.
- PVC-Größe mit freiem Speicher gleichsetzen.
- Protokolldatei löschen, ohne den schreibenden Prozess zu prüfen.
- Aktive Datenbankprotokolle manuell löschen.
- Docker-Volumes pauschal bereinigen.
- Alle Snapshots vorsorglich entfernen.
- Storagepool und Dateisystem gleichzeitig verändern.
- Mehrere Bereinigungen gleichzeitig ausführen.
- Keine Messung vor und nach der Maßnahme durchführen.
- Einen Neustart als Speicherbereinigung verwenden.
- Schreibschutz als reinen Kapazitätsfehler behandeln.
- Hardwarefehler nach einer kurzfristigen Speicherfreigabe ignorieren.
- Nur Platz freigeben, aber die Wachstumsursache nicht beheben.
- Keine zukünftige Kapazitäts- und Retentionsplanung festlegen.

7.13.37 Verifikation

Nach der Maßnahme müssen mindestens folgende Punkte geprüft werden:

- betroffener Pfad liegt auf dem erwarteten Dateisystem;
- Dateisystem ist schreibbar;
- ausreichender absoluter Speicher ist verfügbar;
- prozentuale Auslastung liegt im vorgesehenen Bereich;
- Inodes sind verfügbar;
- wirksame Quotas sind korrekt;
- Thin Pool besitzt freie Daten- und Metadatenkapazität;
- Storage Pool und Backend besitzen Reserve;
- Snapshots entsprechen der vorgesehenen Retention;
- keine gelöschten großen Dateien werden weiterhin offengehalten;
- Logrotation funktioniert;
- Anwendung erzeugt keine unkontrollierte Datenmenge mehr;
- Container startet ohne Neustartschleife;
- Kubernetes-Node zeigt keinen `DiskPressure`;
- Datenbank schreibt fehlerfrei;
- Backup- und Replikationsdienste funktionieren;
- keine neuen Dateisystem- oder I/O-Fehler erscheinen;
- ursprünglicher Schreibvorgang funktioniert;
- Wachstumsrate bleibt nach der Maßnahme kontrolliert;
- Monitoring löst korrekt aus;

- temporäre Änderungen wurden zurückgenommen;
- Ursache und Prävention wurden dokumentiert.

Ein einzelner erfolgreicher Schreibtest reicht nicht aus, wenn der Speicher weiterhin unkontrolliert wächst.

7.13.38 Prävention und Monitoring

Zu überwachen sind:

- prozentuale Dateisystembelegung;
- absolut freier Speicher;
- Inodebelegung;
- Wachstum pro Stunde und Tag;
- Quotaauslastung;
- Snapshotbelegung;
- Thin-Pool-Datenbereich;
- Thin-Pool-Metadatenbereich;
- Container- und Imagebelegung;
- Kubernetes `nodefs` und `imagefs`;
- Backuprepository;
- Datenbank- und Transaktionslogs;
- Protokollverzeichnisse;
- Storagepoolkapazität;
- Hardware- und I/O-Fehler.

Ein sinnvoller Alarm berücksichtigt nicht nur einen festen Prozentwert.

Beispiel:

```
Warnung:
Auslastung über organisationsspezifischem Grenzwert
UND weniger als definierter absoluter Freispeicher

Kritisch:
Kapazität reicht bei aktueller Wachstumsrate
nicht bis zum nächsten geplanten Eingriff
```

Zusätzliche Präventionsmaßnahmen:

- Kapazitätsprognosen erstellen;
- Retention verbindlich dokumentieren;
- Logrotation regelmäßig testen;
- Quotas mit Warnschwellen verwenden;

- Snapshotlebenszyklen überwachen;
- Thin Provisioning nicht überbuchen, ohne das Backend zu überwachen;
- Containerlogs begrenzen;
- Ephemeral-Storage-Requests und -Limits planen;
- Backuprepository und Produktionsdaten getrennt überwachen;
- Restore- und Bereinigungsvorgänge testen;
- Eigentümer für speicherintensive Pfade festlegen.

7.13.39 Dokumentationsvorlage

```
Störung:
<exakte Fehlermeldung>

Zeitpunkt:
<Datum und Uhrzeit>

Betroffenes System:
<Hostname, VM, Container oder Node>

Betroffene Anwendung:
<Dienst oder Anwendung>

Betroffener Pfad:
<Laufwerksbuchstabe, Mountpoint oder Dateipfad>

Speicherebenen:
<Anwendung, Dateisystem, Volume, Pool und Backend>

Dateisystem:
<Typ und Mountoptionen>

Gesamtkapazität:
<Wert>

Freier Speicher vor der Maßnahme:
<Wert und Prozent>

Inodebelegung:
<Wert und Prozent>
```

Wirksame Quota:

<Typ, Limit und Nutzung>

Snapshotbelegung:

<Wert und Bestand>

Thin-Pool-Auslastung:

<Daten- und Metadatenprozent>

Wachstumsrate:

<Wert pro Stunde oder Tag>

Größte Verbraucher:

<Pfade, Größen und Besitzer>

Gelöschte offene Dateien:

<Befund>

Storage- und Hardwarezustand:

<Befund>

Nachgewiesene Ursache:

<technischer Nachweis>

Gegenbeweis ausgeschlossen durch:

<Test und Ergebnis>

Durchgeführte Maßnahme:

<genau beschriebene Änderung>

Risiko und Rückweg:

<Beschreibung>

Freier Speicher nach der Maßnahme:

<Wert und Prozent>

Verifikation:

<Schreibtest, Anwendungstest und Monitoring>

Prävention:

<Rotation, Retention, Quota, Kapazität oder Monitoring>

7.13.40 Checkliste

- [ ] exakte Fehlermeldung dokumentiert
- [ ] Datum und Uhrzeit erfasst
- [ ] betroffenen Host erfasst
- [ ] betroffene Anwendung erfasst
- [ ] betroffenen Benutzer erfasst
- [ ] exakten Schreibpfad bestimmt
- [ ] Dateisystem des Pfades bestimmt
- [ ] Mountpoint oder Laufwerksbuchstabe bestätigt
- [ ] Gesamtkapazität dokumentiert
- [ ] absolut freien Speicher dokumentiert
- [ ] prozentuale Auslastung dokumentiert
- [ ] Inodes geprüft
- [ ] Benutzerquota geprüft
- [ ] Gruppenquota geprüft
- [ ] Projekt- oder Verzeichnisquota geprüft
- [ ] Anwendungsquota geprüft
- [ ] Schreibschutz geprüft
- [ ] Mountoptionen geprüft
- [ ] Partition oder Logical Volume geprüft
- [ ] Volume Group oder Speicherpool geprüft
- [ ] Thin-Pool-Datenbereich geprüft
- [ ] Thin-Pool-Metadatenbereich geprüft
- [ ] RAID-, NAS- oder SAN-Kapazität geprüft
- [ ] Snapshots geprüft
- [ ] Shadow Copies geprüft
- [ ] größte Verzeichnisse bestimmt
- [ ] größte Dateien bestimmt
- [ ] Änderungszeiten geprüft
- [ ] Datenbesitzer bestimmt
- [ ] Wachstumsrate bestimmt

- [ ] viele kleine Dateien berücksichtigt
- [ ] `df` und `du` verglichen
- [ ] gelöschte offene Dateien geprüft
- [ ] Protokollwachstum geprüft
- [ ] Logrotation geprüft
- [ ] Datenbanklogs geprüft
- [ ] Backupretention geprüft
- [ ] Container-Layer geprüft
- [ ] Container-Volumes geprüft
- [ ] Containerlogs geprüft
- [ ] Kubernetes `DiskPressure` geprüft
- [ ] Ephemeral Storage geprüft
- [ ] PVC und Backend getrennt geprüft
- [ ] System- und Kernelprotokolle geprüft
- [ ] Hardware- und I/O-Fehler geprüft
- [ ] Hypothese formuliert
- [ ] Gegenbeweis festgelegt
- [ ] Risiko und Rückweg dokumentiert
- [ ] Datenverantwortlicher einbezogen
- [ ] nur eine kontrollierte Maßnahme durchgeführt
- [ ] freien Speicher erneut gemessen
- [ ] Inodes erneut geprüft
- [ ] Quota erneut geprüft
- [ ] ursprünglichen Schreibvorgang getestet
- [ ] ursprüngliche Anwendung getestet
- [ ] Wachstumsrate nachkontrolliert
- [ ] Monitoring geprüft
- [ ] temporäre Änderungen zurückgenommen
- [ ] Ursache dokumentiert
- [ ] Präventionsmaßnahme festgelegt

7.13.41 Schnellreferenz

Aufgabe	Befehl
Windows-Volumes	`Get-Volume`

Aufgabe	Befehl
PowerShell-Dateisystemlaufwerke	`Get-PSDrive -PSProvider FileSystem`
Windows-Datenträger	`Get-Disk`
Windows-Partitionen	`Get-Partition`
physische Windows-Speicherobjekte	`Get-PhysicalDisk`
NTFS-Quota	`fsutil quota query C:`
FSRM-Quotas	`Get-FsrmQuota`
VSS-Speicher	`vssadmin list shadowstorage`
Shadow Copies	`vssadmin list shadows`
Windows-Komponentenspeicher	`Dism.exe /Online /Cleanup-Image /AnalyzeComponentStore`
Linux-Dateisystembelegung	`df -hT`
Linux-Pfad prüfen	`df -hT <Pfad>`
Linux-Inodes	`df -i`
Mounts	`findmnt`
Mount eines Pfades	`findmnt --target <Pfad>`
Blockgeräte	`lsblk -o NAME,TYPE,FSTYPE,SIZE,FSAVAIL,FSUSE%,MOUNTPOINTS`
Verzeichnisgrößen	`du -x -h --max-depth=1 <Pfad>`
gelöschte offene Dateien	`lsof +L1`
Journalbelegung	`journalctl --disk-usage`
Benutzerquota	`quota -s`
Quotaübersicht	`repquota -a`
XFS-Quota	`xfs_quota -x -c 'report -h' <Mountpoint>`
LVM Physical Volumes	`pvs`
LVM Volume Groups	`vgs`
LVM und Thin Pools	`lvs -a -o lv_name,vg_name,lv_size,pool_lv,origin,data_percent,metadata_percent`
Btrfs-Auslastung	`btrfs filesystem usage -T <Mountpoint>`
ZFS-Poolauslastung	`zpool list`
ZFS-Datasetbelegung	`zfs list -o space`
macOS-Dateisystembelegung	`df -h`
macOS-Datenträger	`diskutil list`
APFS-Struktur	`diskutil apfs list`
APFS-Snapshots	`diskutil apfs listSnapshots /`

Aufgabe	Befehl
lokale Time-Machine-Snapshots	`tmutil listlocalsnapshots /`
Docker-Speicherübersicht	`docker system df -v`
Docker-Containergrößen	`docker ps -a --size`
Kubernetes-Nodes	`kubectl get nodes`
Kubernetes-Nodezustand	`kubectl describe node <Node>`
Kubernetes-PVCs	`kubectl get pvc --all-namespaces`

7.13.42 Quellen

Offizielle Microsoft-Dokumentation

- Microsoft Learn – Get-Volume
- Microsoft Learn – Get-Disk
- Microsoft Learn – Get-PhysicalDisk
- Microsoft Learn – Get-PSDrive
- Microsoft Learn – Get-FsrmQuota
- Microsoft Learn – FileServerResourceManager PowerShell module
- Microsoft Learn – Vssadmin list shadowstorage
- Microsoft Learn – Determine the actual size of the WinSxS folder
- Microsoft Learn – Clean up the WinSxS folder
- Microsoft Learn – Configure Storage Sense

Offizielle Linux- und Projektdokumentation

- GNU Coreutils – df invocation
- GNU Coreutils – du invocation
- systemd – journalctl
- systemd – journald.conf
- Linux Kernel – ext4 documentation
- Red Hat – Disk Quotas
- Red Hat – Limiting XFS storage usage with quotas
- Btrfs – btrfs-filesystem documentation
- OpenZFS – zfs-list documentation

Offizielle Docker-Dokumentation

- Docker Docs – docker system df
- Docker Docs – Prune unused Docker objects
- Docker Docs – docker system prune
- Docker Docs – docker volume prune

Offizielle Kubernetes-Dokumentation

- Kubernetes – Local ephemeral storage
- Kubernetes – Node-pressure eviction
- Kubernetes – Ephemeral volumes

Offizielle Apple-Dokumentation

- Apple – Get detailed information about a disk
- Apple – Add, delete or erase APFS volumes
- Apple – View APFS snapshots
- Apple – Repair a storage device

Für diese Seite wurden keine Community-Berichte, Hersteller-Social-Media-Aussagen oder eigenen Laborergebnisse als Nachweis verwendet.
