1.Write a Linux shell script that determines if any of the mounted file system has less than 20% disk free

```
#!/bin/bash

THRESHOLD=80 #If a filesystem is using 80% or more of its capacity, an alert will be sent.

mounts=$(df -h | awk '{print $6}' | grep -v "^Mounted$")

for mount in $mounts; do

  usage=$(df -h $mount | awk 'NR==2 {print $5}' | grep -o '[0-9]*')

  if [ $usage -ge $THRESHOLD ]; then

    echo "Usage of $mount is at $usage %." | mail -s "High disk usage alert: $mount" mail@example.com

  fi

done
```

Explaination:

#!/bin/bash This line indicates that the script should be executed using the Bash shell.

THRESHOLD=80 This sets a threshold value of 80%. If any filesystem usage exceeds this value, an alert will be triggered.

- mounts=$(df -h | awk '{print $6}' | grep -v "^Mounted$")
- `df -h`: This command displays disk space usage in a human-readable format.
- `awk '{print $6}'`: This extracts the 6th column from the `df -h` output, which corresponds to the mount points.
- `grep -v "^Mounted$"`: This removes the header line from the output, which contains the word "Mounted".

for mount in $mounts; do This starts a loop that iterates over each mount point retrieved in the previous step.

usage=$(df -h $mount | awk '{print $5}' | grep -o '[0-9]*')

- `df -h $mount`: This command displays disk usage for the specific mount point.
- `awk '{print $5}'`: This extracts the 5th column from the `df -h` output, which corresponds to the percentage usage.

- `grep -o '[0-9]*'`: This extracts only the numeric part of the percentage (removing the '%' character).

if [$usage -ge $THRESHOLD]; then echo "Usage of $mount is at $usage %." | mail -s "High disk usage alert: $mount" mail@example.com fi done

- `if [ $usage -ge $THRESHHOLD ]; then`: This checks if the usage percentage is greater than or equal to the threshold.
- `echo "Usage of $mount is at $usage %."`: This constructs the message to be sent in the email.
- `mail -s "High disk usage alert: $mount" mail@example.com`: This sends an email with the subject "High disk usage alert: $mount" and the message about the current usage to `mail@example.com`.

2. Write a python equivalent version of a script that determines if any of the mounted file system has less than 20% disk free

```
import subprocess

import smtplib

from email.mime.text import MIMEText


threshold = 80

partition = "/"


def report_via_email():

    msg = MIMEText("Server running out of disk space")

    msg["Subject"] = "Low disk space warning"

    msg["From"] = "admin@example.com"

    msg["To"] = "test@gmail.com"

    with smtplib.SMTP("smtp.gmail.com", 587) as server:

        server.ehlo()
```

```
        server.starttls()
        server.login("gmail_user", "gmail_password")
        server.sendmail("admin@example.com", "test@gmail.com", msg.as_string())

def check_once():
    df_output = subprocess.check_output(["df", "-h"]).decode("utf-8")
    lines = df_output.splitlines()
    for line in lines[1:]:  # Skip the header line
        splitline = line.split()
        if splitline[5] == partition:
            if int(splitline[4][:-1]) > threshold:
                report_via_email()

check_once()
```

Explaination:

```
import subprocess
import smtplib
from email.mime.text import MIMEText
```

- `subprocess`: This module allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes. It is used here to execute the `df -h` command to check disk usage.
- `smtplib`: This module defines an SMTP client session object that can be used to send email to any Internet machine with an SMTP or ESMTP listener daemon.
- `email.mime.text.MIMEText`: This class is used to create MIME objects of text type. It is used here to create the body of the email.

Configuration

```
python
Copy code
```

```
threshold = 80
partition = "/"
```

- `threshold`: This sets the disk usage percentage threshold to 80%. If any partition's usage exceeds this value, an alert will be sent.
- `partition`: This specifies the partition to monitor, which is the root partition (`/`) in this case.

Function to Send Email

```
def report_via_email():
    msg = MIMEText("Server running out of disk space")
    msg["Subject"] = "Low disk space warning"
    msg["From"] = "admin@example.com"
    msg["To"] = "test@gmail.com"

    with smtplib.SMTP("smtp.gmail.com", 587) as server:
        server.ehlo()
        server.starttls()
        server.login("gmail_user", "gmail_password")
        server.sendmail("admin@example.com", "test@gmail.com",
msg.as_string())
```

- `msg = MIMEText("Server running out of disk space")`: This creates the email body with the message "Server running out of disk space".
- `msg["Subject"] = "Low disk space warning"`: This sets the email subject.
- `msg["From"] = "admin@example.com"`: This sets the sender's email address.
- `msg["To"] = "test@gmail.com"`: This sets the recipient's email address.
- `with smtplib.SMTP("smtp.gmail.com", 587) as server`: This establishes an SMTP connection to the Gmail server on port 587.
    - `server.ehlo()`: This sends an EHLO (Extended HELO) command to the server, which identifies the client to the server.
    - `server.starttls()`: This puts the SMTP connection in TLS (Transport Layer Security) mode.
    - `server.login("gmail_user", "gmail_password")`: This logs into the Gmail account using the provided username and password.

    - `server.sendmail("admin@example.com", "test@gmail.com", msg.as_string())`: This sends the email from admin@example.com to test@gmail.com with the specified message.

Function to Check Disk Usage

```
def check_once():
    df_output = subprocess.check_output(["df", "-h"]).decode("utf-8")
    lines = df_output.splitlines()

    for line in lines[1:]:  # Skip the header line
        splitline = line.split()
        if splitline[5] == partition:
            if int(splitline[4][:-1]) > threshold:
                report_via_email()
```

- `df_output = subprocess.check_output(["df", "-h"]).decode("utf-8")`: This executes the `df -h` command and captures its output. The `decode("utf-8")` part converts the output from bytes to a string.
- `lines = df_output.splitlines()`: This splits the output into a list of lines.
- `for line in lines[1:]:`: This iterates over each line, starting from the second line to skip the header.
- `splitline = line.split()`: This splits each line into a list of words.
- `if splitline[5] == partition`: This checks if the 6th column (mount point) matches the specified partition.
- `if int(splitline[4][:-1]) > threshold`: This checks if the usage percentage (stripping the '%' character) exceeds the threshold.
    - `report_via_email()`: If the usage exceeds the threshold, it calls the `report_via_email()` function to send an alert email.

check_once()

This calls the `check_once()` function to check the disk usage and potentially send an alert email.

Summary

This script checks the disk usage of the specified partition and sends an email alert if the usage exceeds 80%. It leverages the `subprocess` module to execute system commands,

the `smtplib` module to send emails, and the `email.mime.text` module to format the email content.

3. zabbix python script

```
#!/usr/bin/env python3
 import psutil


def check_disk_space(threshold=20):
   partitions = psutil.disk_partitions()
   for partition in partitions:
      usage = psutil.disk_usage(partition.mountpoint)
      free_percentage = usage.free / usage.total * 100
      if free_percentage < threshold:
         return 1  # Return 1 if any filesystem is using over 20% of its free disk space
   return 0  # Return 0 if all filesystems have more than 20% free disk space


if __name__ == "__main__":
   result = check_disk_space(20)
   print(result)
```

Explaination:

Shebang Line

```
#!/usr/bin/env python3
```

This line specifies that the script should be run using the Python 3 interpreter found in the user's environment.

Importing Required Module

```
import psutil
```

psutil is a Python module that provides an interface for retrieving information on running processes and system utilization (CPU, memory, disks, network, sensors). It is used here to get disk usage statistics.

Function to Check Disk Space

def check_disk_space(threshold=20):

This defines a function named check_disk_space which accepts an optional parameter threshold. The default value for threshold is 20, meaning it will check if any filesystem has less than 20% free disk space.

```
    partitions = psutil.disk_partitions()
```

psutil.disk_partitions(): This function returns a list of all mounted disk partitions as sdiskpart objects, which contain information about each partition such as the device, mount point, filesystem type, etc.

```
    for partition in partitions:
        usage = psutil.disk_usage(partition.mountpoint)
        free_percentage = usage.free / usage.total * 100
        if free_percentage < threshold:
            return 1  # Return 1 if any filesystem is using over 20% of its free disk space
```

This loop iterates over each partition retrieved from psutil.disk_partitions().

psutil.disk_usage(partition.mountpoint): For each partition, it retrieves the disk usage statistics. This returns an sdiskusage object which contains information about the total, used, and free space as well as the percentage used.

free_percentage = usage.free / usage.total * 100: This calculates the percentage of free disk space by dividing the free space by the total space and multiplying by 100.

if free_percentage < threshold: This checks if the free percentage is less than the specified threshold.

return 1: If any partition has less than the threshold of free space, the function returns 1 immediately.

```
    return 0  # Return 0 if all filesystems have more than 20% free disk space
```

If the loop completes without finding any partitions with less free space than the threshold, the function returns 0, indicating that all partitions have more than the specified threshold of free space.

Main Execution Block

```
if __name__ == "__main__":
    result = check_disk_space(20)
    print(result)
```

if __name__ == "__main__":: This checks if the script is being run directly (not imported as a module).

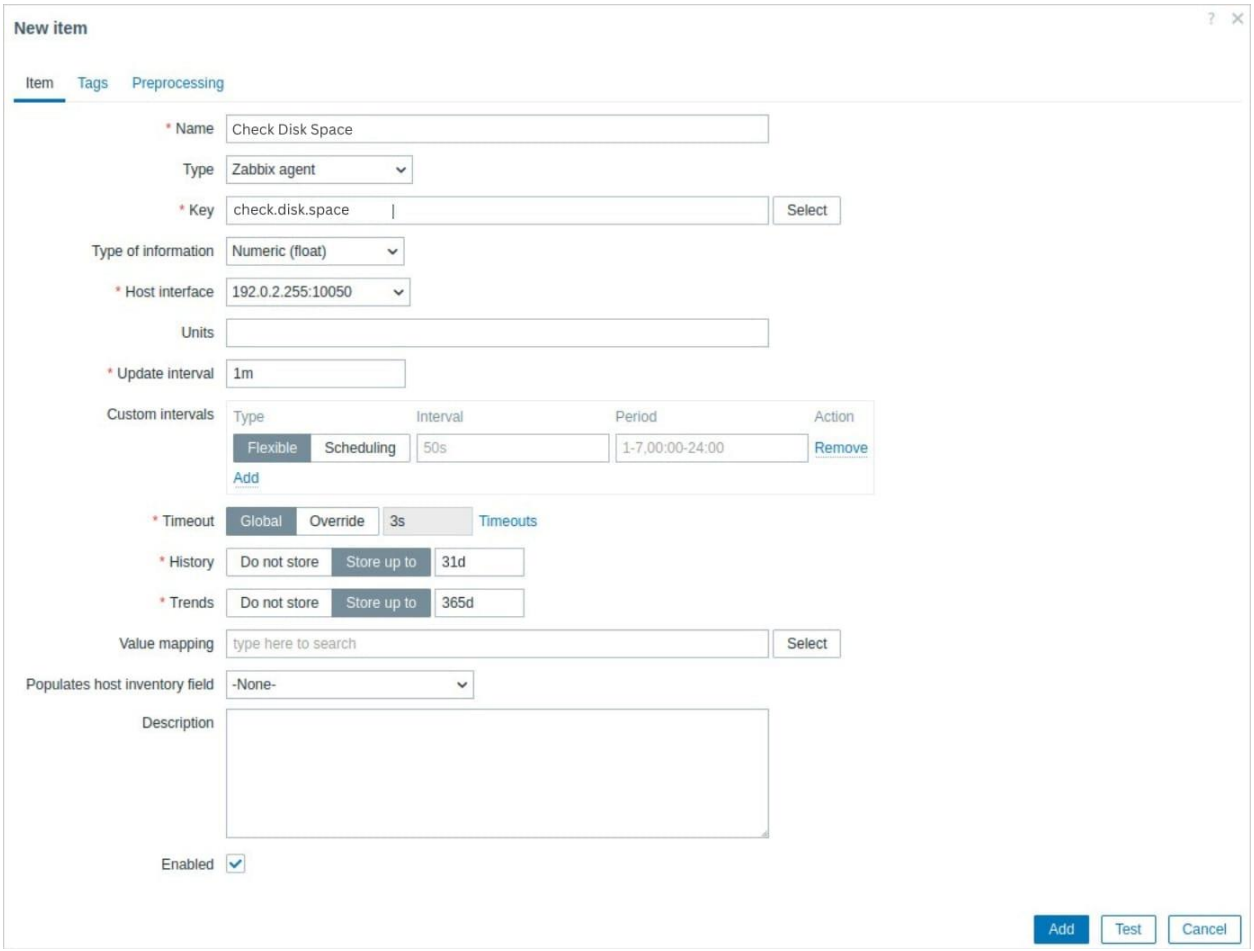
result = check_disk_space(20): This calls the check_disk_space function with 20 as the threshold value and stores the result in the result variable.

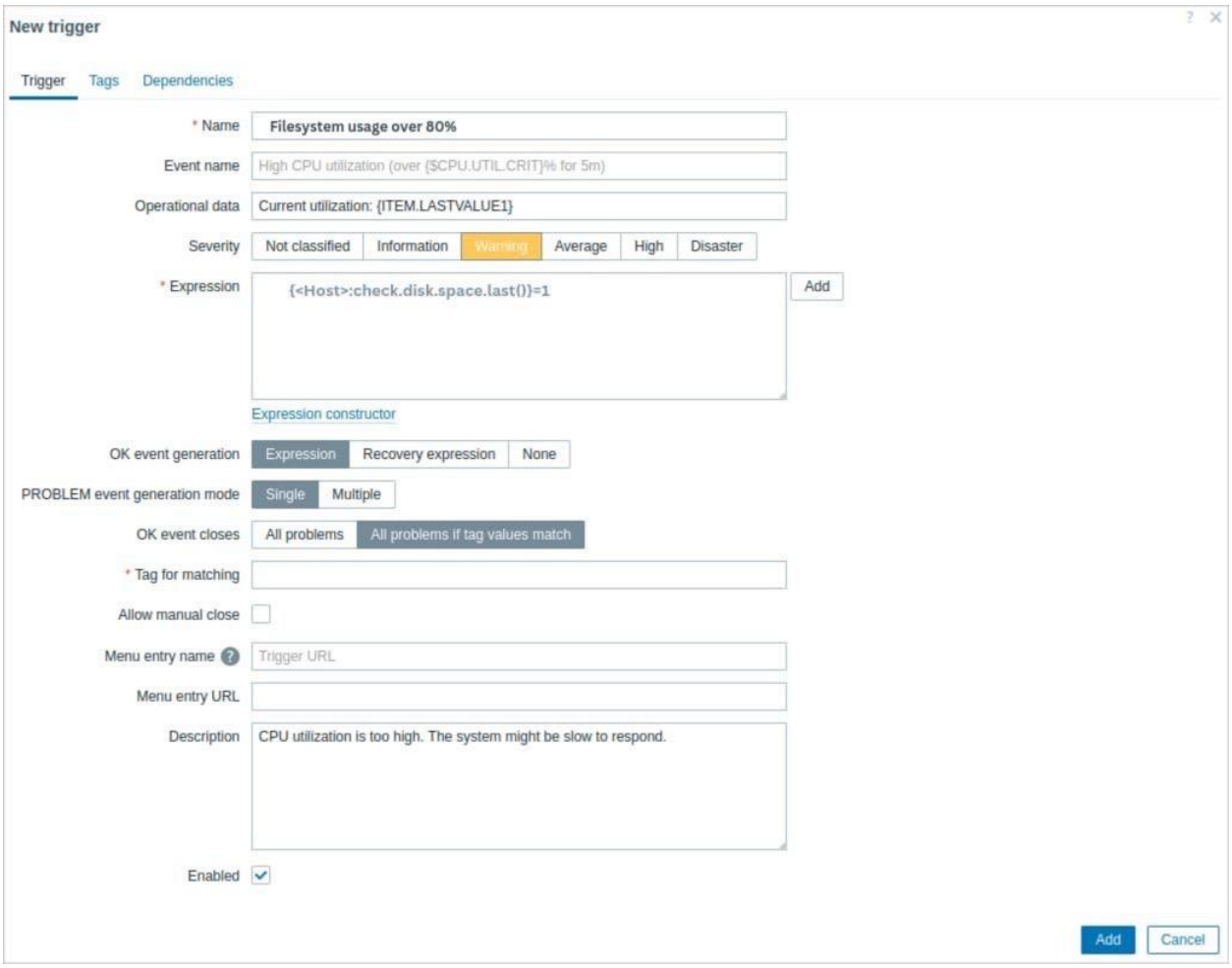
print(result): This prints the result. It will print 1 if any partition has less than 20% free space, and 0 if all partitions have more than 20% free space.

Summary

This script uses the psutil library to check the disk usage of all mounted partitions on the system. It calculates the percentage of free space for each partition and checks if it is below a specified threshold (default 20%). If any partition is below the threshold, it returns 1; otherwise, it returns 0. This script can be used in automation or monitoring systems to alert administrators about low disk space.

Screenshots of configuration:




New trigger
Trigger
Tags
Dependencies
* Name
Filesystem usage over 80%
Event name
High CPU utilization (over {$CPU.UTIL.CRIT}% for 5m)
Operational data
Current utilization: {ITEM.LASTVALUE1}
Severity
Not classified
Information
Warning
Average
High
Disaster
* Expression
{<Host>:check.disk.space.last()}=1
Add
Expression constructor
OK event generation
Expression
Recovery expression
None
PROBLEM event generation mode
Single
Multiple
OK event closes
All problems
All problems if tag values match
* Tag for matching
Allow manual close
Menu entry name
Trigger URL
Menu entry URL
Description
CPU utilization is too high. The system might be slow to respond.
Enabled
Add
Cancel

Graph | Preview

* Name: Free Disk Space

* Width: 900

* Height: 200

Graph type: Normal

Show legend: ☑

Show working time: ☑

Show triggers: ☑

Percentile line (left): ☐

Percentile line (right): ☐

Y axis MIN value: Fixed | 0

Y axis MAX value: Calculated

* Items

	Name	Function	Draw style	Y axis side	Color	Action
1:	My host: Outgoing network traffic on eth0	avg	Filled region	Left	00C800	Remove
2:	My host: Incoming network traffic on eth0	avg	Bold line	Left	C80000	Remove

Add

Add | Cancel