

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

گزارشکار پروژه شبکه

هلیاسادات هاشمی پور

۹۸۳۱۱۰۶

خرداد ۰۱

در بخش اول باید یک برنامه agent بنویسیم که متريک های مهم را برای بررسی وضعیت سیستم را از سیستمی که بر روی آن اجرا شده است، دریافت و در بازه های زمانی مختلف برای سرور واسطه به کمک سوکت ارسال کند. کد نوشته شده به شکل زیر است.

```
import socket
from time import sleep
import psutil
import json

host = '127.0.0.1'
port = 8080
FORMAT = 'UTF-8'

def main():
    CONNECTED = True
    while CONNECTED:
        sleep(2)
        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((host, port))
            print(f'Connected to Server {host}:{port}')
            counter = 1
            CONNECTED = True
            while CONNECTED:
                information = {
                    'RAM': ': psutil.virtual_memory()[2],
                    'CPU': ': psutil.cpu_percent(interval=4),
                    'disk': ': psutil.disk_usage('.').free,
                    'CPU counts': ': psutil.cpu_count()'}
                message = json.dumps(information)
                s.send(message.encode(FORMAT))
                print(f"Message Number {counter} Sent to Server")
                counter = counter + 1
                sleep(4)
        except Exception as err:
            print(err)
            inp = input("We Lost the Connection to Server, Try again? N/Y")
            if inp == "Y":
                continue
            else:
                break
    if __name__ == "__main__":
        main()
```

برای پیاده‌سازی این برنامه برای تهیه متريک‌ها از کتابخانه `psutil` و همچنین برای ارسال آن‌ها به سرور از قابب JSON استفاده شده است. حال متريک‌هایی که من در برنامه استفاده کرده‌ام به صورت زیر است:

- 1 مقدار درصد مصرفی رم
- 2 مقدار مصرفی پردازنده (در طول ۴ ثانیه)
- 3 آمار استفاده از دیسک در مورد پارتیشنی که شامل مسیر داده شده به عنوان یک تاپل نامگذاری شده است.
- 4 تعداد پردازنده‌های منطقی سیستم

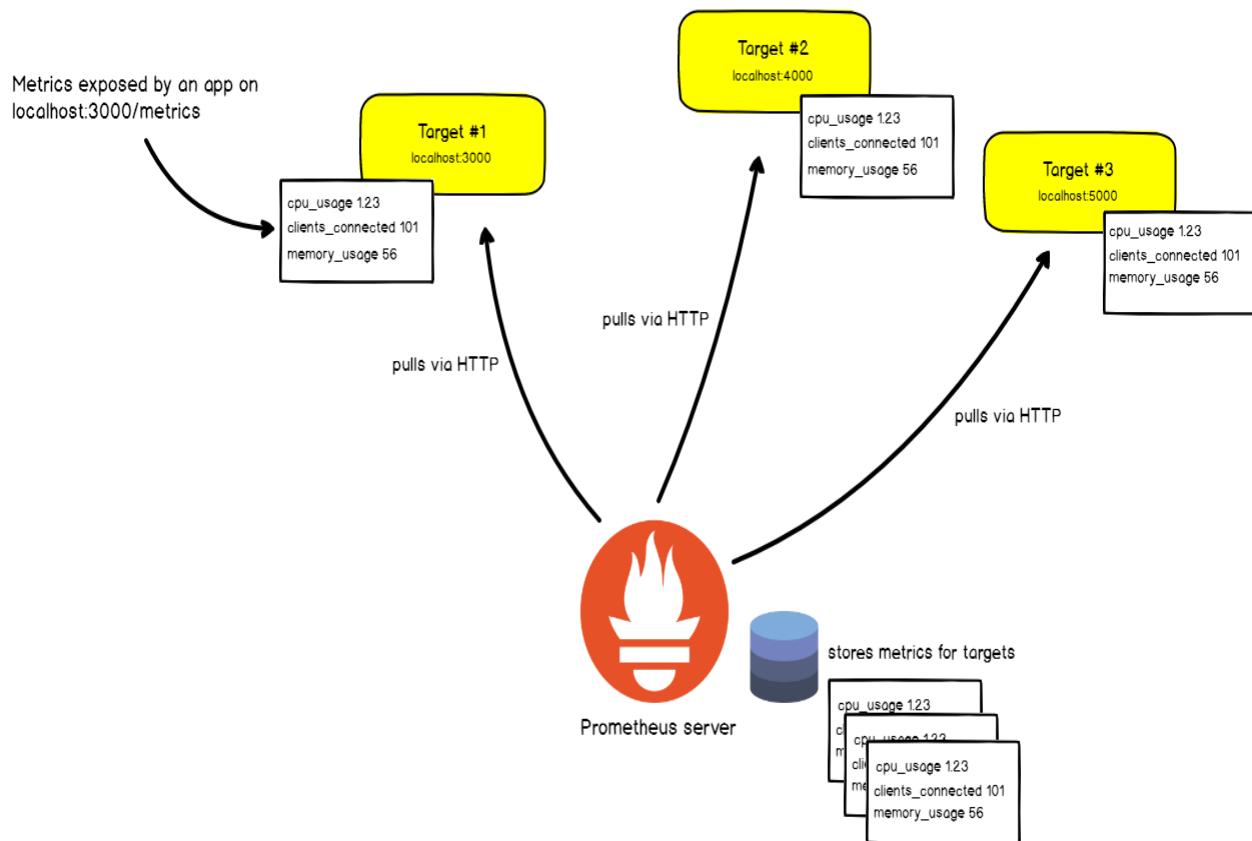
در بخش دوم هم باید یک کلاس سرور بنویسیم. آدرس IP را ۱۲۷.۰.۰.۱ که آدرس لوکال ما هست و شماره پورت را هم ۸۰۸۰ گذاشتم. کلاینت ما با استفاده از این مقادیر ذکر شده به سرور متصل می‌شود. برای آن متريک‌های ذکر شده را می‌فرستد. سرور ما به صورت `multithread` کار می‌کند. (در کد هم ساپورت شده) در اصل به این صوت است که هر کلاینت یک ترد برای آن ساخته تا چند تا کلاینت همزمان متصل به سرور داشته باشم.

```
def received_data(connection, addr, client_number):
    print(f"Client Number {client_number} Connected, {addr}")
    try:
        CONNECTED = True
        while CONNECTED:
            msg = connection.recv(1024).decode(FORMAT)
            datas = json.loads(msg)
            print(f"[{addr}] {msg}")
            metric_1.labels(f'client_{client_number}').set(datas['RAM: '])
            metric_2.labels(f'client_{client_number}').set(datas['CPU: '])
            metric_3.labels(f'client_{client_number}').set(datas['disk: '])
            metric_4.labels(f'client_{client_number}').set(datas['CPU counts: '])
    except Exception as err:
        print(err)
        print(f'Client{client_number} disconnected!!!!')

def main():
    print("Server is starting...")
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((host, port_num))
    server.listen()
    print(f"Server {host}:{port_num} is connected...")
    client_number = 1
    start_http_server(8001)
    CONNECTED = True
    while CONNECTED:
        connection, addr = server.accept()
        thread = threading.Thread(target=received_data, args=(connection, addr, client_number))
        thread.start()
        print(f"[Active Clients] {client_number}")
        client_number = client_number + 1
```

در بخش سوم هم با پرومئوس سر و کار داریم.

What does Prometheus do?



یک برنامه نرم افزاری که برای مانیتورینگ و هشدار استفاده می‌شود.

حال در این بخش باید متريک‌های کلاسی که در بخش نخست پیاده‌سازی کردیم را که به سرور فرستاده‌ایم را استخراج کنیم و با استفاده از متريک **gauge** به اين برنامه مى‌دهيم تا بتوانيم تغييرات آن را مشاهده کنیم.
متريک‌های پرومئوس به صورت زير است:

counter - 1

همانند اسمش يك شمارنده متريک تجمعی می‌باشد که در اصل نشان دهنده يك شمارنده يکنواخت در حال افزایش است.

```
from prometheus_client import Counter  
c = Counter('my_failures', 'Description of counter')  
c.inc()    # Increment by 1  
c.inc(1.6) # Increment by given value
```

gauge - 2

متريکی است که به ما يک مقدار عددی واحد را نشان می‌دهد. در اصل اين معیار می‌تواند به شکل خودسرانه بالا و پایین رود. در پروژه من از اين متريک استفاده كرده‌ام زيرا ما در اينجا به کاهش و افزایش متغيرمان برای متريک‌های ارسالی به کلاینت را داريم.

```
from prometheus_client import Gauge
g = Gauge('my_inprogress_requests', 'Description of gauge')
g.inc()    # Increment by 1
g.dec(10)  # Decrement by given value
g.set(4.2) # Set to a given value
```

در کلاس سرور هم برای هر متريک ما يک لیبل ست کردیم. همچنین برای اينکه متريک‌ها را روی سرور http بر روی پورت ۱۸۰۰۱ بالا آوردم.

histogram - 3

از مقادير متعدد که مجموع تمام مقادير ذخیره شده و تعداد رويدا هايي که ثبت شده را نمونه‌برداری می‌کند.

```
from prometheus_client import Histogram
h = Histogram('request_latency_seconds', 'Description of histogram')
(h.observe(4.7) # Observe 4.7 (seconds in this case
```

summary - 4

يك خلاصه مشاهدات را نمونه‌برداری می‌کند. همچنین تعداد کل مشاهدات و مجموع تمام مقادير مشاهده شده را به ما نشان می‌دهد.

```
from prometheus_client import Summary
s = Summary('request_latency_seconds', 'Description of summary')
(s.observe(4.7) # Observe 4.7 (seconds in this case
```

حال ابتدا فایل با پسوند `.yml` را به شکل زیر عوض می کنیم برنامه پرومتهوس را روی سیستم خود نصب کرده و یک بار به شکل زیر ریاستارت می کنیم:

```
usr > local > etc > . prometheus.yml
1 # my global config
2 global:
3   scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
4   evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
5   # scrape_timeout is set to the global default (10s).
6
7 # Alertmanager configuration
8 alerting:
9   alertmanagers:
10     - static_configs:
11       - targets:
12         # - alertmanager:9093
13
14 # Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
15 rule_files:
16   # - "first_rules.yml"
17   # - "second_rules.yml"
18
19 # A scrape configuration containing exactly one endpoint to scrape:
20 # Here it's Prometheus itself.
21 scrape_configs:
22   # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
23   - job_name: "prometheus"
24
25     # metrics_path defaults to '/metrics'
26     # scheme defaults to 'http'.
27
28     static_configs:
29       - targets: ["localhost:9090"]
30
31     - job_name: "my_exporter"
32       static_configs:
33         - targets: ["localhost:8001"]
```

```
Last login: Thu Jun  9 15:53:45 on ttys002
heliaa@MacBook-Pro-4 ~ % brew services restart prometheus
Stopping `prometheus`... (might take a while)
==> Successfully stopped `prometheus` (label: homebrew.mxcl.prometheus)
==> Successfully started `prometheus` (label: homebrew.mxcl.prometheus)
heliaa@MacBook-Pro-4 ~ %
```

برنامه را اجرا می کنیم:

طرف سرور •

```
Last login: Thu Jun  9 17:21:39 on ttys003
[heliaa@MacBook-Pro-4 server % python3 main.py
  File "/Users/heliaa/Desktop/server/main.py", line 38
    start_http_server(800)
               ^
SyntaxError: invalid decimal literal
[heliaa@MacBook-Pro-4 server % python3 main.py
Server is starting...
Server 127.0.0.1:8080 is connected
Client Number 1 Connected, ('127.0.0.1', 50997)
Client1 disconnected!
[active clients] 1
Client Number 2 Connected, ('127.0.0.1', 50999)
[active clients] 2
[('127.0.0.1', 50999)] {"RAM": "67.7", "CPU": "12.5", "disk": "22651076608", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "67.4", "CPU": "13.0", "disk": "22649675776", "CPU counts": "8"}
Client Number 3 Connected, ('127.0.0.1', 51011)
[active clients] 3
Client3 disconnected!
[('127.0.0.1', 50999)] {"RAM": "67.1", "CPU": "19.1", "disk": "22649180160", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "66.0", "CPU": "19.1", "disk": "22649167872", "CPU counts": "8"}
Client Number 4 Connected, ('127.0.0.1', 51025)
[active clients] 4
Client4 disconnected!
[('127.0.0.1', 50999)] {"RAM": "67.1", "CPU": "28.6", "disk": "22648786944", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "66.5", "CPU": "30.9", "disk": "22648250368", "CPU counts": "8"}
Client Number 5 Connected, ('127.0.0.1', 51029)
[active clients] 5
Client5 disconnected!
[('127.0.0.1', 50999)] {"RAM": "67.5", "CPU": "9.7", "disk": "22648160256", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "67.1", "CPU": "7.9", "disk": "22648119296", "CPU counts": "8"}
Client Number 6 Connected, ('127.0.0.1', 51031)
[active clients] 6
Client6 disconnected!
[('127.0.0.1', 50999)] {"RAM": "67.1", "CPU": "8.8", "disk": "22648049664", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "66.1", "CPU": "17.2", "disk": "22648037376", "CPU counts": "8"}
[('127.0.0.1', 50999)] {"RAM": "66.7", "CPU": "14.6", "disk": "22648037376", "CPU counts": "8"}
Client Number 7 Connected, ('127.0.0.1', 51042)
[active clients] 7
Client7 disconnected!
```

طرف کلاینت •

```
Last login: Thu Jun  9 17:35:59 on ttys001
[heliaa@MacBook-Pro-4 client % python3 main.py
Connected to Server 127.0.0.1:8080
Message Number 0 Sent to Server
Message Number 1 Sent to Server
Message Number 2 Sent to Server
Message Number 3 Sent to Server
Message Number 4 Sent to Server
Message Number 5 Sent to Server
Message Number 6 Sent to Server
Message Number 7 Sent to Server
Message Number 8 Sent to Server
Message Number 9 Sent to Server
Message Number 10 Sent to Server
Message Number 11 Sent to Server
Message Number 12 Sent to Server
Message Number 13 Sent to Server
Message Number 14 Sent to Server
Message Number 15 Sent to Server
Message Number 16 Sent to Server
```

حال برنامه را روی لوکال هاست با پورت ۹۰۹۰ مشاهده می‌کنیم. به بخش تارگت رفته و آنچه که در فایل با پسوند yml نوشتیم را مشاهده می‌کنیم. متريک‌ها به شکل زیر است:

The screenshot shows the Prometheus Targets page. At the top, there are navigation links: Prometheus, Alerts, Graph, Status, Help, and a settings icon. Below the navigation is a search bar labeled "Filter by endpoint or labels". There are two sections of tables:

- my_exporter (1/1 up)**: This section shows one target: `http://localhost:8001/metrics`. The status is UP, and it has labels `instances="localhost:8001"` and `job="my_exporter"`. The last scrape was 6.898s ago, and the scrape duration was 2.191ms.
- prometheus (1/1 up)**: This section shows one target: `http://localhost:9090/metrics`. The status is UP, and it has labels `instances="localhost:9090"` and `job="prometheus"`. The last scrape was 13.910s ago, and the scrape duration was 4.002ms.

```
# HELP python_gc_objects_collected_total Objects collected during gc
# TYPE python_gc_objects_collected_total counter
python_gc_objects_collected_total{generation="0"} 70.0
python_gc_objects_collected_total{generation="1"} 322.0
python_gc_objects_collected_total{generation="2"} 0.0
# HELP python_gc_objects_uncollectable_total Uncollectable object found during GC
# TYPE python_gc_objects_uncollectable_total counter
python_gc_objects_uncollectable_total{generation="0"} 0.0
python_gc_objects_uncollectable_total{generation="1"} 0.0
python_gc_objects_uncollectable_total{generation="2"} 0.0
# HELP python_gc_collections_total Number of times this generation was collected
# TYPE python_gc_collections_total counter
python_gc_collections_total{generation="0"} 41.0
python_gc_collections_total{generation="1"} 3.0
python_gc_collections_total{generation="2"} 0.0
# HELP python_info Python platform information
# TYPE python_info gauge
python_info{implementation="CPython",major="3",minor="10",patchlevel="2",version="3.10.2"} 1.0
# HELP memory_usage Usage of RAM
# TYPE memory_usage gauge
memory_usage{client_number="client_2"} 68.1
# HELP cpu_usage Usage of CPU
# TYPE cpu_usage gauge
cpu_usage{client_number="client_2"} 14.4
# HELP disk_usage usage of disk
# TYPE disk_usage gauge
disk_usage{client_number="client_2"} 2.190352384e+010
# HELP cpu_counts usage of cpu count
# TYPE cpu_counts gauge
cpu_counts{client_number="client_2"} 8.0
```

گرافهایی که داریم هم به صورت زیر می‌باشند:



