الجمهورية العربية السورية

جامعة تشرين

كلية الهندسة الميكانيكية والكهربائية

هندسة الاتصالات والالكترونيات

مشروع برمجة شبكات بعنوان:

Multi Threading Client/Server Socket – Concepts And Definitions

إعداد الطالبتين:

ريم وفيق أحمد **رهف مصطفى ريا**

بإشراف:

د.مهند عيسى

Multi threading Client/Server Socket – Concepts And Definitions

ملخص:

بمجرد توجيه البيانات عبر الشبكة وتسليمها إلى مضيف معين، يجب تسليمها إلى المستخدم أو العملية الصحيحة. أثناء تحرك البيانات لأعلى أو لأسفل طبقات TCP/IP، هناك حاجة إلى آلية لتسليمها إلى البروتوكولات الصحيحة في كل طبقة. يجب أن يكون النظام قادرا على الجمع بين البيانات من العديد من التطبيقات في عدد قليل من بروتوكولات النقل، ومن بروتوكولات النقل إلى بروتوكول الإنترنت. يسمى الجمع بين العديد من مصادر البيانات في دفق بيانات واحد تعدد الإرسال. يجب إلغاء تعدد البيانات التي تصل من الشبكة مقسمة للتسليم إلى عمليات متعددة. لإنجاز هذه المهمة، يستخدم IP أرقام البروتوكولات لتحديد بروتوكولات النقل، وتستخدم بروتوكولات النقل أرقام المنافذ لتحديد التطبيقات. يتم حجز بعض أرقام البروتوكولات والمنافذ لتحديد الخدمات المعروفة. الخدمات المعروفة هي بروتوكولات الشبكة القياسية، مثل FTP وTelnet، والتي يشيع استخدامها في جميع أنحاء الشبكة. يتم تعيين أرقام البروتوكول وأرقام المنافذ للخدمات المعروفة من قبل هيئة الأرقام المخصصة للإنترنت (IANA). يتم توثيق الأرقام الموقعة رسميا في http://www.iana.org.

في هذه المقالة، سنتحدث عن كيفية قيام التطبيقات بتسليم البيانات بين جهازين باستخدام بروتوكول التحكم في الإرسال (TCP) أو بروتوكول مخطط بيانات المستخدم (UDP). ونركز على دور socket.

كلمات مفتاحية: socket

1.مقدمة

طبقة النقل هي جزء من نموذج شبكاتTCP/IP ، وتسمى أحيانا بنية الشبكات. يحتوي على مجموعة شاملة من الوظائف التي تصف كل ما هو مطلوب لشبكة الكمبيوتر للعمل. طبقة النقل مسؤولة عن الاتصال المنطقي بين التطبيقات التي تعمل على مضيفين مختلفين، وبالتالي توفير الخدمات لبروتوكولات طبقة التطبيق على طبقة أعلى من نموذج شبكةIP / TCP . على الرغم من وجود العديد من بروتوكولات طبقة النقل، فإن البروتوكولين الأكثر استخداما هما بروتوكول التحكم في الإرسال (TCP) وبروتوكول مخطط بيانات المستخدم(UDP) . توفر هذه البروتوكولات وظائف مختلفة لمتطلبات التطبيق المختلفة. بعض من أهم الوظائف هي:

تتبع المحادثة الفردية:

تُعرف البيانات المتدفقة من تطبيق إلى آخر بالمحادثة، حيث يمكن أن يكون للمضيف تطبيقات متعددة تتواصل مع بعضها البعض، إما داخل شبكة محلية أو شبكة بعيدة. تحتوي طبقة النقل على آلية تتيح لكل تطبيق على مضيف التواصل مع تطبيق آخر على مضيف مختلف، إما داخل شبكة محلية أو شبكة بعيدة، ووفقًا لشركة Cisco، تقوم هذه الآلية بتعيين معرف يسمى رقم المنفذ لكل تطبيق، بحيث يكون لكل عملية برمجية تحتاج للوصول إلى شبكة معينة معرف فريد.

طلب نقل البيانات:

يتم تقسيم الدفق المستمر للبايتات إلى مقاطع لإرسالها وتسليمها بواسطة خدمات طبقة النقل. وفقًا لهذه المقالة، فإن معظم الشبكات لها قيود على كمية البيانات التي يمكن أن تحتويها حزمة واحدة. ولهذا السبب، تقوم طبقة نقل جهاز الإرسال بإعداد البيانات إلى شرائح، وبالمثل، تستقبل طبقة نقل الجهاز المستقبِل هذه المقاطع وتستخدم الرأس لإعادة بنائها إلى بيانات كاملة.

تعدد المحادثات باستخدام أرقام المنافذ:

عند استخدام أحد التطبيقات، عادةً ما تظهر البيانات أو الخدمات المقدمة على شكل دفق من البيانات المستمرة، ولكن إرسال البيانات (مثل الفيديو) عبر الشبكة كدفق كامل يمكن أن يستهلك كل النطاق الترددي المتاح للشبكة. هذا يمنع الخدمات الأخرى مثل البريد الإلكتروني من استخدام الوسيط ويجعل استعادة الأخطاء وإعادة إرسال بيانات التلف أكثر صعوبة. تقسم آلية تعدد الإرسال بيانات TCP وUDP إلى أجزاء صغيرة لتمكين الاتصال من مستخدمين مختلفين للتداخل على نفس الشبكة. تعتمد هذه الآلية على مفهوم يعرف باسم المقبس socket.

2. أهمية البحث وأهدافه:

تلعب socket دورًا حيويًا في تطبيقات خادم العميل. يمكن للعميل والخادم التواصل مع بعضهما البعض عن طريق الكتابة إلى هذه socket أو القراءة منها. يقدم هذا البحث عناصر من برمجة الشبكات والمفاهيم المتضمنة في إنشاء تطبيقات الشبكة باستخدام socket.

3. المقبس socket :

بشكل عام، المقابس هي نقاط نهاية داخلية مصممة لإرسال البيانات واستقبالها. ستحتوي الشبكة الواحدة على مقبسين، واحد لكل جهاز أو برنامج متصل. هذه المقابس هي مزيج من عنوان IP ومنفذ. يمكن أن يحتوي جهاز واحد على عدد "n" من المقابس استنادا إلى رقم المنفذ المستخدم. تتوفر منافذ مختلفة لأنواع مختلفة من البروتوكولات

لقد جاء هذا المصطلح " socket" بعدة طرق.

1. يمكن تعريف المآخذ على أنها نقاط نهاية اتصال بين جهازي كمبيوتر يتم تحديدهما بواسطة عنوان IP ورقم منفذ.
2. يمكن تعريف المقابس أيضًا على أنها تجريد للبرامج يستخدم لتمثيل "محطات" اتصال بين جهازين.
3. يمكن تعريفه أيضًا على أنه فكرة مجردة يتم توفيرها لمبرمج تطبيق لإرسال البيانات أو استقبالها إلى عملية أخرى.
4. المقبس هو الباب بين عملية التطبيق و.TCP

5. المقبس هو واجهة بين التطبيق والشبكة.

1.3. عمليات المقبس socket:

يقوم المقبس بأربع عمليات أساسية:

1 .الاتصال بجهاز بعيد

2 .إرسال البيانات

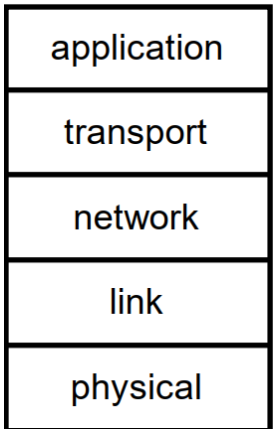
3 .تلقي البيانات

4 .إغلاق الاتصال

قد لا يتم توصيل المقبس بأكثر من مضيف واحد في نفس الوقت. ومع ذلك، قد يرسل المقبس البيانات إلى المضيف المتصل به ويستقبلها.

2.3. المنافذ:

في شبكات الكمبيوتر، يكون المنفذ عبارة عن بناء برنامج خاص بالتطبيق أو خاص بالعملية يعمل كنقطة نهاية للاتصالات في نظام التشغيل المضيف للكمبيوتر. يرتبط المنفذ بعنوان IP الخاص بالمضيف، بالإضافة إلى نوع البروتوكول المستخدم للاتصال، والغرض من المنافذ هو التعرف بشكل فريد على التطبيقات أو العمليات المختلفة التي تعمل على جهاز كمبيوتر واحد، والبروتوكولات التي تستخدم المنافذ بشكل أساسي هي النقل بروتوكولات الطبقة، مثل بروتوكول التحكم في الإرسال (TCP) وبروتوكول مخطط بيانات المستخدم (UDP) الخاص بمكدس برامج الإنترنت، والتي تسمى غالبًا TCP / IP (بروتوكول التحكم في النقل / بروتوكول الإنترنت)، كما هو موضح في الشكل (1). يستخدمون المنافذ لتعيين البيانات الواردة لعملية معينة تعمل على الكمبيوتر. لذا فإن المنفذ سيحدد المقبس على مضيف.



الشكل (1)

مرة أخرى، لا يكفي عنوان IP لتحديد خادم فريد، لأن العديد من برامج الخادم قد تكون موجودة على جهاز واحد. لذلك نحن بحاجة إلى تعريف فريد لكل خادم. يشار إلى هذا التعريف الفريد باسم المنفذ. عندما نقوم بإعداد عميل أو خادم، يجب علينا اختيار منفذ يوافق عليه كل من العميل والخادم. هذا المنفذ ليس منفذًا فعليًا، ولكنه منفذ منطقي محدد برقم صحيح 16 بت. بعض المنافذ لا. تم حجز من 0 إلى 1024 لدعم الخدمات الشائعة / المعروفة ويجب على المطور أن يكون حريصًا على عدم استخدام أحد أرقام المنافذ المعروفة والمحددة في RFCs أثناء تطوير تطبيق خاص بخادم العميل

Port	Service
21	File transfer protocol (FTP)
22	Secure shell (SSH)
23	Telenet
25	Simple mail transfer protocol (SMTP)
53	Domain name system (DNS)
80	Hypertext transfer protocol (HTTP)
110	Post office protocol (POP)
143	Internet message access protocol (IMAP)
443	HTTP secure (HTTPS)

Protocol	Port Number	Python Library	Function
HTTP	80	httplib, urllib,xmlrpclib	Web pages
FTP	20	ftplib, urllib	File transfers
NNTP	119	nntplib	Unsent news
SMTP	25	smtplib	Sending email
Telnet	23	telnetlib	Command lines
POP3	110	poplib	Fetching email
Gopher	70	gopherlib	Document transfer

edureka!

أرقام المنافذ الشائعة والبروتوكولات ذات الصلة

4. كيفية تحقيق برمجة المقبس في بايثون:

لتحقيق برمجة المقبس في بايثون، ستحتاج إلى استيراد وحدة **المقبس** أو الإطار .تتكون هذه الوحدة من طرق مضمنة مطلوبة لإنشاء مآخذ ومساعدتها على الارتباط ببعضها البعض.

بعض الطرق الهامة هي كما يلي:

أساليب	وصف
socket.socket()	تستخدم لإنشاء مآخذ التوصيل (مطلوبة على كل من الخادم وكذلك نهايات العميل لإنشاء مآخذ التوصيل)
socket.accept()	تستخدم لقبول اتصال .يقوم بإرجاع زوج من القيم(conn ، العنوان (حيث يكون conn كائن مقبس جديد لإرسال البيانات أو استقبالها والعنوان هو عنوان المقبس الموجود في الطرف الآخر من الاتصال
socket.bind()	يستخدم للربط بالعنوان المحدد كمعلمة
socket.close()	تستخدم لوضع علامة على المقبس على أنه مغلق
socket.connect()	يستخدم للاتصال بعنوان بعيد محدد كمعلمة
socket.listen()	تمكين الخادم من قبول الاتصالات

يوفر Python مستويين من الوصول إلى خدمات الشبكة. على مستوى منخفض، يمكنك الوصول إلى دعم المقبس الأساسي في نظام التشغيل الأساسي، والذي يسمح لك بنشر العملاء والخوادم لكل من البروتوكولات الموجهة إلى الاتصال وغير المتصلة.

المقابس لها مفردات خاصة بها-

الرقم.	الوصف
1	Domainالنطاق مجموعة البروتوكولات المستخدمة كآلية النقل. هذه القيم هي ثوابت مثل AF_INET و PF_INET و PF_UNIXو.PF_X25
2	typeالنوع نوع الاتصالات بين نقطتي النهاية، عادةً SOCK_STREAM للبروتوكولات الموجهة للاتصال و SOCK_DGRAMللبروتوكولات غير المتصلة.
3	protocolالبرتوكول عادةً ما تكون صفرية، يمكن استخدام هذا لتحديد متغير بروتوكول داخل مجال ونوع.
4	hostnameاسم المضيف معرف واجهة الشبكة - سلسلة، يمكن أن تكون اسمًا مضيفًا أو عنوانًا منقطًا رباعيًا أو عنوان IPV6 في علامة النقطتين (وربما نقطية) سلسلة ""، تحدد عنوان .INADDR_BROADCAST سلسلة ذات طول صفري تحدد INADDR_ANY أو عدد صحيح، يتم تفسيره كعنوان ثنائي بترتيب بايت المضيف. •
5	portالمنفذ كل قائمة الخادم للعملاء الذين يتصلون على واحد أو أكثر من المنافذ. قد يكون المنفذ هو رقم منفذ Fixnum أو سلسلة تحتوي على رقم منفذ أو اسم خدمة.

1.4. socket Module برمجة السوكيت

لإنشاء مأخذ توصيل، يجب استخدام دالة () socket.socket المتاحة في وحدة المقابس، والتي تحتوي على بناء الجملة العام

```
s = socket.socket (socket_family, socket_type, protocol=0)
```

- Here is the description of the parameters هنا وصف الوسيطات المتغيرة

- socket_family – هذا إما AF_UNIX أو AF_INET.

- socket_type – هذا إما SOCK_STREAM أو SOCK_DGRAM.

2.4. دوال سيرفر المقابس Server Socket Methods

الرقم.	الدالة والوصف
1	s.bind() هذه الطريقة تربط العنوان (اسم المضيف، زوج رقم المنفذ) بالمقبس.
2	s.listen() تقوم هذه الطريقة بإعداد وتشغيل مستمع.TCP
3	s.accept() يقبل هذا بشكل سلبي اتصال عميلTCP ، في انتظار حتى وصول الاتصال (حظر).

3.4.دوال عامة بالسوكيت General Socket Methods .

الرقم.	الدالة والوصف
1	s.recv() هذه الطريقة تستقبل رسالةTCP
2	s.send() هذه الطريقة تنقل رسالةTCP
3	s.recvfrom() هذه الطريقة تستقبل رسالةUDP
4	s.sendto() هذه الطريقة تنقل رسالةUDP
5	s.close() هذه الطريقة تغلق المقبس
6	socket.gethostname() إرجاع اسم المضيف.

مفهوم المسارات threading:

هي عبارة مكتبة برمجية تؤمن تشغيل مجموعة تعليمات برمجية موضوعة ضمن تابع على التوازي مع تابع آخر.

Simple Server : سيرفر بسيط

لكتابة خوادم الإنترنت، نستخدم وظيفة المقبس المتاحة في وحدة المقابس لإنشاء كائن مأخذ توصيل. ثم يتم استخدام كائن مأخذ التوصيل لاستدعاء وظائف أخرى لإعداد خادم مأخذ توصيل. اتصل الآن بوظيفة الربط (اسم المضيف، المنفذ) لتحديد منفذ لخدمتك على المضيف المحدد. بعد ذلك، استدعاء الأسلوب المقبول للكائن المرتجع. تنتظر هذه الطريقة حتى يتصل العميل بالمنفذ الذي حددته، ثم يُرجع كائن اتصال يمثل الاتصال بهذا العميل.

Simple Client : مثال للعميل سوكيت

نكتب برنامج عميل بسيط للغاية يفتح اتصال بمنفذ معين 6666 ومضيف معين. منفذ (اتصال TCP إلى اسم مضيف على المنفذ. بمجرد فتح مأخذ توصيل، يمكنك القراءة منه مثل أي كائن إدخال / إخراج.

5.العمل مع TCP Sockets:

برنامج Echo server:

```
import socket,threading
ss=socket.socket()
ss.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
ss.bind(('0.0.0.0',4444))
ss.listen(2)
def handel_request(i,csocket,cadd):
    print(' Accepted new client: ',i ,cadd)
    while True:
        cmsg = csocket.recv(1024).decode()
        if cmsg == 'exit':
            csocket.close()
            break
        print(" [*]msg from client{}:{}".format(cadd, cmsg))
        csocket.send(cmsg.encode())
    print("Finished Serving Current Client")

i=1
while True:
    print('[+] TCP server is waiting for new clients...')
    csocket,cadd=ss.accept()

client=threading.Thread(target=handel_request,args=(i,csocket,cadd))
    client.start()
    i+=1
```

برنامج العميل

يوجد أدناه برنامج client.py. يحاول العميل الاتصال بمنفذ الخادم، 6666(منفذ محدد جيدًا). يفتح سطر الكود، connect ((مضيف، منفذ)) اتصال TCP باسم المضيف على المنفذ 6666

```
import socket,sys
cs=socket.socket()
cs.settimeout(20)
try:
    cs.connect(("127.0.0.1",4444))
except socket.error as e:
    print(e)
    sys.exit(1)

while True:
    req=input("enter msg to send to echo server: ")
    try:
        cs.send(req.encode())
        if req == "exit":
            break
        data=cs.recv(1024).decode()
        print("msg from server",data)
    except socket.error as e:
        print(e)
```

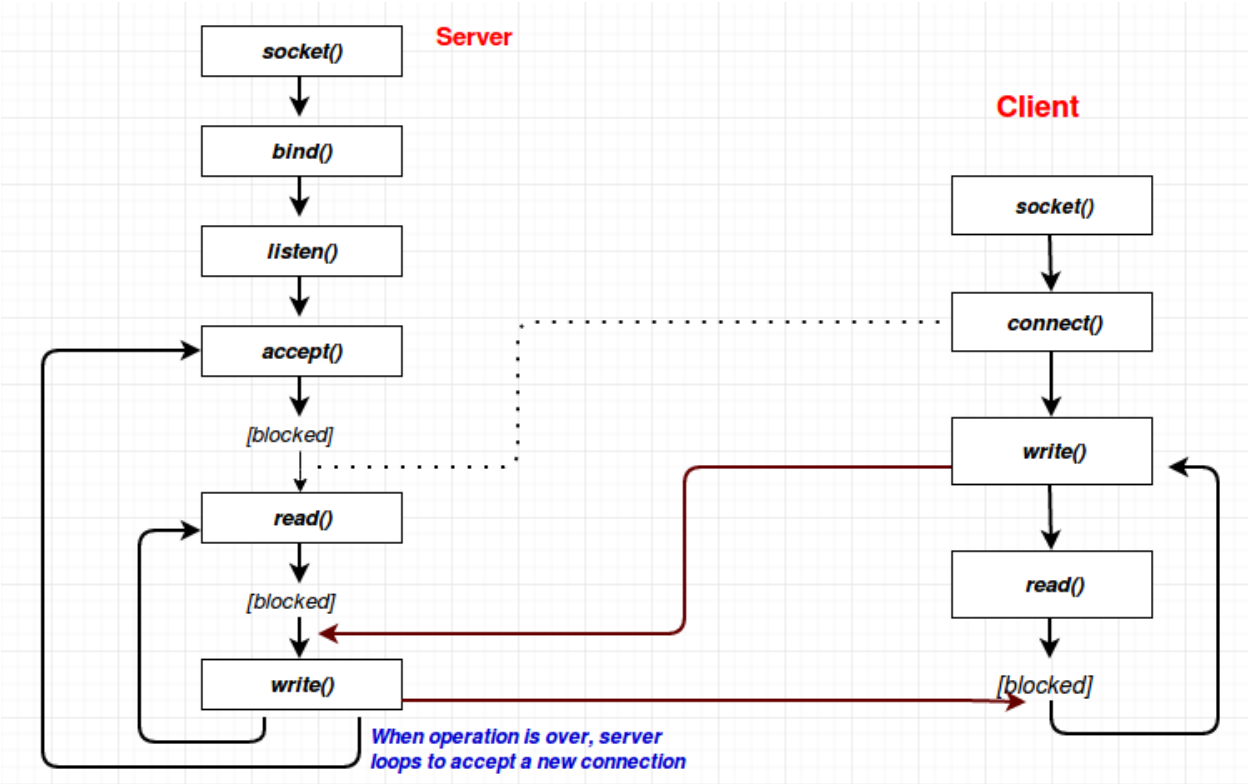
الآن ، نقوم بتشغيل البرنامج النصي server.py أولاً

```
[+] TCP server is waiting for new clients...
 Accepted new client: [+] TCP server is waiting for new clients... 1
('127.0.0.1', 55738)
 [*]msg from client('127.0.0.1', 55738):hello
 Accepted new client: [+] TCP server is waiting for new clients...
2 ('127.0.0.1', 55739)
 [*]msg from client('127.0.0.1', 55739):hi
```

ثم بتشغيل البرنامج النصي client.py.

```
enter msg to send to echo server: hello
msg from server hello
enter msg to send to echo server:
```

مخطط تدفق البرنامج:



5.المراجع:

[1] https://www.studytonight.com/network-programming-in-python/working-with-tcp-sockets.

[2] https://codingshiksha.com/python/python-3-script-to-transfer-large-images-pdf-documents-from-remote-server-using-socket-full-project-for-beginners/

[3] https://cisco.box.com/v/devnet1040.

[4] A tutorial on Networking Programming using Sockets http://beej.us/guide/bgnet/output/print/bgnet_USLetter

[5] http://en.wikipedia.org/wiki/Berkeley_sockets