10/1/2005

פרוטוקול SMTP ויישומו בתוכניות בשפת C

מסמך זה הורד מהאתר http://www.underwar.co.il.

המסמך נכתב על ידי **KaMiKaZe aka K4xZ**
המסמך נערך על ידי **ניר אדר** (nir@underwar.co.il).

פרוטוקול SMTP:

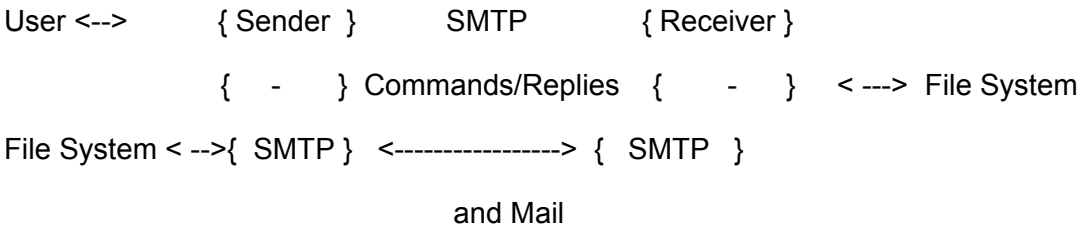
במהלך מאמר זה אני אסביר בקצרה על פרוטוקול SMTP, כיצד הוא עובד ובעיקר יישומו בתוכנית שנכתבה בשפת C. המאמר מלווה בקוד בשפת C שמייד אחריו הסברים על הקוד.

מהו פרוטוקול SMTP?

פרוטוקול SMTP הוא קיצור ל-Simple Mail Transfer Protocol, פרוטוקול זה טוב יותר מפרוטוקולים אחרים משום שהוא אמין. הפרוטוקול הינו עצמאי למערכת משנה ודורש זרימה של מידע אמין ועובד על פורט 25.
התכונה החשובה ביותר של פרוטוקול זה היא היכולת לשלוח מייל לעבר שירות העברה סביבתי.

כיצד עובד הפרוטוקול?

המודל הבא מדגים את הדרך שבה עובד הפרוטוקול:



פקודות

לאחר התחברות לשרת מקבלים מהשרת תגובה 220 וכמה מילים על השרת.
בשלב הראשון שולחים לשרת את ההודעה:

HELO <domain>

זוהי בעצם הודעת הזדהות בפני השרת.

השרת מחזיר תגובה 250 <name_of_the_host>

כאשר name_of_the_host יהיה שם השרת שדרכו שולחים.

בשלב השני צריכים לשלוח את כתובת האימייל של השולח, לכן הפקודה הבאה תהיה

MAIL FROM:<reverse-path><CRLF>

כאשר reverse_path היא כתובת השולח ו-CLRF מתחלק למעשה לשניים, CR מסמן את סוף השורה ו-LF מסמן תחילת שורה חדשה.

על פקודה זאת מקבלים תגובת אישור מהשרת 250 OK.

בשלב השלישי נשלח לשרת את הפקודה:

RCPT TO:<forward-path><CRLF>

כאשר forward-path זוהי כתובת האימייל של הנמען.

השרת מחזיר תגובת 250 OK.

הפקודה הבאה מודיעה לשרת שעכשיו אנחנו הולכים לשלוח את המידע שנמצא בתוך האימייל.

DATA<CRLF>

השרת שולח בתגובה 354 Start mail input.

את הפקודות לאחר מכן שולחים באופן הבא:

FROM: bla@bla.bla<CRLF>

ולא מקבלים שום תגובה.

לאחר מכן שולחים:

Subject: Hello<CRLF>

גם כעת עדיין לא מקבלים שום תגובה.

כעת מכניסים את תוכן ההודעה:

blalblalblalblalblal<CRLF>.<CRLF>

הנקודה מציינת שכתיבת האימייל הסתיימה ואנחנו רוצים לשלוח אותו.

כעת נקבל את התגובה 250 OK.

כעת נותרו שלוש פקודות אחרונות:

הפקודה הראשונה:

RSET<CRLF>

הפקודה אומרת לשרת שיודיע למקבל שמסיימים והוא יכול לנקות את החוצצים.

התגובה על פקודה זאת היא 250 OK.

הפקודה השנייה:

NOOP<CRLF>

הפקודה רק בודקת שהשרת עדיין נמצא.

התגובה על פקודה זאת היא 250 OK.

פקודה שלישית:

QUIT<CRLF>

הפקודה מציינת שרוצים לנתק את התקשורת עם השרת.

השרת מחזיר בתגובה 221 לאישור.

תגובות מהשרת

S(Success),F(Failure),E(Error)

CONNECTION ESTABLISHMENT

S: 220

F: 421

HELO

S: 250

E: 500, 501, 504, 421

MAIL S: 250

F: 552, 451, 452

E: 500, 501, 421

RCPT

S: 250, 251

F: 550, 551, 552, 553, 450, 451, 452

E: 500, 501, 503, 421

DATA

I: 354 -> data -> S: 250

F: 552, 554, 451, 452

F: 451, 554

E: 500, 501, 503, 421

RSET

S: 250

E: 500, 501, 504, 421

NOOP

S: 250

E: 500, 421

QUIT

S: 221

E: 500

הקוד

התחברות לשרת:

בכתובת הבאה תוכלו למצוא את הכתובות אליהם מתחברים:

http://www.netguru.co.il/content/view/86/2/

במידה ומשהו לא יודע כיצד להפוך ל IP את השרתים, הוא יכול להשתמש בפונקציה gethostbyname(). או לעשות ping mail gw.netvision.net.il (במידה ואתם לקוחות של netvision כמובן).

כעת אנחנו צריכים 3 חוצצים בגודל 256 בתים ומשתנה err על מנת שנוכל לדעת מתי יש שגיאה. הקצאת החוצצים והמשתנה תיעשה באופן הבא:

```
char *recvbuf = (char *)malloc(sizeof(char)*256);
char *recvbuf_tmp = (char *)malloc(sizeof(char)*256);
char sendbuf[256] = "";
int err = 0;
```

כעת לאחר שנשתמש בפונקציה connect, נשתמש בפונקציה recv() באופן הבא:

```
recv(msock,recvbuf_tmp,256,0);
Sleep(500);
if(*(recvbuf_tmp)=='4' || *(recvbuf_tmp)=='5')
{
        err = 1;
}
if(*(recvbuf_tmp)=='2' && *(recvbuf_tmp+1)=='2' && *(recvbuf_tmp+2) =='0' && err==0)
```

```
{
        puts(recvbuf_tmp);
}
```

הסבר:

- קלטנו את המידע לתוך החוצץ והוספנו את התו ('\0') המסמן סוף מחרוזת.
- הפונקציה ()sleep גורמת להפסקה של מספר שניות.
- כעת בודקים אם יש שגיאה, כאשר בודקים אם התו הראשון בחוצץ recvbuf_tmp שווה לתו '4' או לתו '5', משום שתווים אילו מייצגים שגיאה, במידה וכן אז err=1, התנאי הבא לא יוכל להתקיים והתוכנית תפסיק.
- כאשר אין שגיאה עוברים לתנאי הבא שבודק אם התגובה המוחזרת היא 220 באמצעות בדיקה. אם התו הראשון הוא התו '2', התו השני הוא גם כן '2' והתו השלישי הוא '0'.

הקוד הבא הוא פקודת ה-HELO:

```
strcpy(sendbuf, "HELO ");
strcat(sendbuf,"bezeqint.net\r\n");
send(msock,sendbuf,strlen(sendbuf),0);
Sleep(500);
recv(msock,recvbuf,256,0);
```

הסבר:

- Sendbuf יכיל את **HELO… helo.bezeqint.net\r\n**.
- התווים \r\n הם הפקודה CRLF.
- שולחים את החוצץ ומחכים לאישור.

הקוד לפקודת ה-MAIL FROM:

```
if (*(recvbuf)=='2' && *(recvbuf+1)=='5' && *(recvbuf+2) == '0' && err==0)
{
        strcpy(sendbuf, "MAIL FROM:<bla@gmail.com>\r\n");
```

```
        send(msock,sendbuf,strlen(sendbuf),0);
        Sleep(500);
        recv(msock,recvbuf_tmp,256,0);
}
if (*recvbuf_tmp =='4' || *recvbuf_tmp=='5')
{
        err=1;
}
if (*recvbuf_tmp=='2' && *(recvbuf_tmp+1)=='5' && *(recvbuf_tmp+2)=='0' && err == 0)
{
        puts(recvbuf);
        putchar('\n');
        strcpy(sendbuf, "RCPT TO:<arafat@bezeqint.net>\r\n");
        send(msock,sendbuf,strlen(sendbuf),0);
        Sleep(500);
        recv(msock,recvbuf_tmp,256,0);
}
```

<u>הסבר:</u>

- התגובה לפקודת MAIL FROM מתקבלת ב recvbuf_tmp.
- אם ישנה שגיאה אז התו הראשון יהיה '4' או '5', במקרה זה err=1 והתוכנית מפסיקה.
- אם אין שגיאה ממשיכים ועוברים לפקודת RCPT TO.
- כעת נשלח את החוצץ שלו, לאחר שהוספנו את **\r\n** שהם בעצם <CRLF>.
- את התגובה קיבלנו ל-recvbuf_tmp.

```
if (*recvbuf_tmp=='4' || *recvbuf_tmp=='5')
{
        err=1;
        puts(recvbuf_tmp);
}
```

```
if (*(recvbuf_tmp)=='2' && *(recvbuf_tmp+1)=='5' && *(recvbuf_tmp+2)=='0' && err==0)
{
	strcpy(sendbuf, "DATA\r\n\r\n");
	send(msock,sendbuf,strlen(sendbuf),0);
	Sleep(500);
	recv(msock,recvbuf_tmp,256,0);
}
if (*(recvbuf_tmp)=='4' || *(recvbuf_tmp)=='5')
{
	err=1;
}
if (*(recvbuf_tmp)=='3' && *(recvbuf_tmp+1)=='5' && *(recvbuf_tmp+2)=='4' && err==0)
{
	puts(recvbuf_tmp);
	printf("\n\nstarting with sending data: \n");
	putchar('\n');
	strcpy(sendbuf, "From: <");
	strcat(sendbuf, "blabla"); // (: פה אתם יכולים לרשום מה שבא לכם
	strcat(sendbuf, ">\x0d\x0a");
	send(msock,sendbuf,strlen(sendbuf),0);
	Sleep(500);
	strcpy(sendbuf, "Subject: We want peace\x0d\x0a"); //הנושא יושב פה בסטלה
	send(msock,sendbuf,strlen(sendbuf),0);
	Sleep(500);
	strcpy(sendbuf, "This is a virus! and we don't want peace\x0d\x0a.\x0d\x0a"); //
פה ההודעה
	send(msock,sendbuf,strlen(sendbuf),0);
	Sleep(500);
	recv(msock,recvbuf_tmp,256,0);
}
```

הסבר:

- בדקנו אם יש שגיאה.
- במידה ואין שגיאה מתחילים לשלוח את המידע.
- הפקודה **DATA\r\n** אומרת שעכשיו שולחים את תוכן האימייל.
- בודקים אם יש שגיאה.
- במידה ואין ממשיכים עד שמקבלים **354 start input**.

```
if(*recvbuf_tmp=='5' || *recvbuf_tmp=='4')
{
        err = 1;
        printf("\ncan't send data");
}
if(*(recvbuf_tmp) == '2' && *(recvbuf_tmp+1)=='5' && *(recvbuf_tmp+2)=='0' && err==0)
{
        printf("\n\n\n\nSENDED...\n\n");
}
strcpy(sendbuf, "QUIT\x0d\x0a");
send(msock,sendbuf,strlen(sendbuf),0);
printf("Closing connection ...finished with sex :) \n\n");
getchar();
}
```

<u>הסבר:</u>

- בקוד זה בדקנו אם המידע נשלח.
- השתמשנו בפקודה QUIT, אך לא היינו חייבים.