

# *Subnetting and TCP/IP Addressing Basics by Daniel Petri*

## **זכויות יוצרים:**

אני מאפשר שימוש במאמר זה למטרת חיזוק החומר הנלמד בכיתה וחזרה עליו בזמן העבודה בבית.

**הערה למרצים:** אין המאמר מתיימר ללמד את כל החומר. אנא הפנו את התלמידים לחלקים החסרים, במיוחד לנושא פעולת ה-TCP/IP על שכבותיו ומרכיביו השונים.

**הפצת המאמר מותרת כל עוד לא נמחק או שונה או שובש שם הכותב – דניאל פטרי, וכל עוד לא נעשו שינויים בגוף המאמר עצמו.**

**שימו לב!** נתקלתי במספר מקרים בהם נעשה שימוש במאמר זה (ואחרים שנכתבו על-ידי) תוך כדי הפרה בוטה של חוק זכויות היוצרים ותוך כדי הסרה מכוונת של שמי מן המאמר, והפצתו כאילו הוא נכתב ע"י אותו אדם שגנב את המאמר ממני. מה שמזעזע מכל, הוא העובדה שבשני מקרים עליהם אני יודע, מדובר היה באנשי מקצוע מעולים בתחומם, אבל עובדה זו לא מנעה מהם לגנוב מאמר שלא נכתב על-ידם ולהפיץ אותו בכיתות ובמוסדות בהם הם עבדו, תוך כדי מחיקה מכוונת של שמי מן המאמר והטעיית הקוראים כאילו הם עצמם כתבו אותו.

קוראים יקרים! אל תהיו ישראלים. קיראו, תהנו, העבירו הלאה, אבל אל תגנבו.

כל מקרה שכזה טופל בהתאם ונידון בימים אלה בערכאות שיפוטיות מתאימות.

טענות, הצעות לשיפורים, תיקון טעויות, בעיות, קושיות ושאלות אפשר לשלוח לדואר אלקטרוני:  
[mct@petri.co.il](mailto:mct@petri.co.il)

תמיד תוכלו למצוא את החומר העדכני ביותר באתר הבא <http://www.petri.co.il>

כרגע ניתן להוריד מאמר זה מהלינק הבא: [http://www.petri.co.il/subnetting\\_tutorial\\_he.htm](http://www.petri.co.il/subnetting_tutorial_he.htm)

**דניאל פטרי**

גירסה נוכחית: 2.25  
עודכן לאחרונה: 4 לספטמבר 2005

## **תוכן העניינים**

חלקו העיקרי של המאמר שלפניכם עוסק בהסבר מפורט של פעולת ה-Subnetting.

למאמר 3 חלקים עיקריים:

- **החלק הראשון** דן במרכיבים הטכניים של כתובת ה-IP, ומסביר קצת על אופן ההגדרה של הפרוטוקול. נדבר על מושגים כמו כתובת Host ID, Subnet Mask, Default Gateway, Net ID. בנוסף נדבר על שיטות להקצאת כתובות IP למחשבים ונציין גם את מנגנון ה-APIPA ונראה גם כיצד לבטל אותו. בחלק זה נדון גם בחוקי היסוד של הקצאת כתובות ה-IP - מה מותר ומה אסור לעשות כשמחלקים למחשבים ורשתות כתובות IP. בחלק זה נדבר גם על כתובות ציבוריות וכתובות פרטיות וההבדלים ביניהן. בנוסף, נזכיר את ההבדלים בין Unicast, Broadcast ו-Multicast. חלק זה מכיל פרטים טכניים שעשויים להיות לרובכם מוכרים.
- **החלק השני** נדבר על המרה ממספרים בינאריים לעשרוניים ומעשרוני לבינארי, חלוקה ל-Classes השונים, אבחנה בין ה-Classes השונים (מה שמכונה Classful IP Addressing), נדבר על כמות ה-Host ID וה-Network ID בכל רשת, ועל האבחנה בחלוקת טווחי הכתובות לפי 3 ה-Classes הרגילים (לא נדבר על D ו-E במאמר זה).
- **בחלק השלישי** נחזור על פעולת ה-AND, משמעות ה-Subnetting, על Subnetting של אוקטטה שלמה, של חלק מהאוקטטה, של יותר מאוקטטה אחת, נלמד אותכם כמה חוקים שיעזרו לכם לפתור שאלות בנושא, וניתן הרבה דוגמאות.

יש לציין כי מאמר זה אינו מיועד לתלמידים שאין להם שום מושג בחומר. כתבתי את המאמר כתזכורת בלבד, והוא לא מתיימר ללמד את כל החומר מהתחלה. מאמר זה יכול לעזור למי ששכח או לא בדיוק הבין את יסודות ה-TCP/IP ואת פעולת ה-Subnetting, ולמי שמעוניין לקבל דגשים על נקודות ספציפיות. אם את/ה לא יודע על מה אני מדבר, ואת/ה מנסה לעשות קיצורי דרך, זה לא המקום בשבילך.

**מרצים שימו לב!** אין המאמר מתיימר ללמד את כל החומר, ואנא, הפנו את התלמידים לחלקים החסרים, במיוחד לנושא פעולת ה-TCP/IP על שכבותיו ומרכיביו השונים.

## חלק ראשון

בחלק הראשון נעסוק ב-3 המרכיבים החשובים של ה-IP Addressing והם:

- IP Address
- Subnet Mask
- Default Gateway

לקראת סוף החלק נזכיר בקצרה כיצד ניגשים להגדרות הפרוטוקול במערכות ההפעלה השונות של מיקרוסופט.

## IP Address

כל מכשיר המשתמש ב-TCP/IP צריך כתובת IP. זה יכול להיות מחשב (כתובת אחת עבור כל כרטיס רשת), נתב (או Router - כתובת אחת עבור כל "רגל" או Port של הנתב), מדפסות רשת (Network Interface Printer Device), או כל מכשיר (Device) אחר שמתקשר באמצעות פרוטוקול ה-TCP/IP.

כתובת IP בנוייה מ-32 ביטים (או 4 בייטים) היכולים להיכתב בצורה עשרונית או בצורה בינארית. הכתובות הללו משמשות לזהות את המחשב בתוך רשת המבוססת על פרוטוקול ה-TCP/IP. הפורמט העשרוני של כתובת IP נקרא Dotted-Decimal Notation והוא נועד להקל על משתמשים אנושיים כיוון שהוא הרבה יותר פשוט ונוח לעבודה ולזיכרון מאשר הפורמט הבינארי.

בצורה עשרונית: **132.37.68.92**

אותה כתובת בצורה בינארית: **10000100.00100101.01000100.01011100**

כתובת IP מחולקת ל-4 חלקים פיזיים שונים. כל חלק כזה נקרא "אוקטטה" (בדוגמה הקודמת השתמשתי בצבעים כדי לציין כל אוקטטה) – על שם העובדה שהוא מורכב מ-8 ביטים (או בייט אחד). כל ביט כזה יכול לקבל את הערך 0 או את הערך 1. דוגמה של אוקטטה כזו יכול להיות המספר 00110100, או 52 בעשרוני. למען הדיוק יש לציין שההגייה הנכונה של המילה "אוקטטה" צריכה להיות לפי המילה האנגלית Octet ולא לפי ההגייה העברית שלה.

אוקטטה היא מונח שמציין אוסף של 8 דברים. במקרה שלנו, ה"דברים" הללו הם ביטים של מחשב, כלומר צירוף של 8 ביטים שונים. מכיוון שכל ביט יכול לקבל 2 ערכים שונים (או 0 או 1), ניתן "לשחק" בטווח האפשרויות השונות החל מנקודת המוצא – 8 ביטים שכולם בעלי ערך 0, דרך כל הקומבינציות האפשריות, עד לנקודת הסיום – 8 ביטים בהם כל הביטים בעלי ערך של 1.

00000000  
00000001  
00000010  
00000011  
00000100

וכו' וכו' עד שנגיע ל-

11111111

אם נתבונן שוב במסיפור הבינארי נוכל להבחין שהערכים הללו משמשים כקצוות הטווח. כלומר, מצד הנמוך יש לנו 00000000 (או 0 בעשרוני), ואז מתחילים לגדול לפי סדר הקומבינציות (00000001, 00000010, ..., 00000011 וכו') עד שמגיעים לקצה השני, שהם 11111111 (או 255 בעשרוני).

אם נמיר את המספרים הבינאריים הללו לעשרוני (מיד נלמד איך לעשות זאת בדרך נוחה וקלה), הרי שהטווח של כל אוקטטה כזו יכול לנוע מ-0 ועד 255, או בעצם 256 אפשרויות. דרך נוספת לחשב את מספר האפשרויות מבלי לכתוב את כולן היא זו: יש לנו 8 ביטים שונים כשלכל אחד מהם 2 אפשרויות, לכן, יש לנו 2 בחזקת 8 אפשרויות, או בעצם 256 אפשרויות.

0 = 00000000  
 1 = 00000001  
 2 = 00000010  
 3 = 00000011  
 4 = 00000100

וכו' וכו' עד שנגיע ל-

255 = 11111111

זיכרו שיש לנו 4 אוקטטות כאלה. 4 אוקטטות שכל אחת מהן יש לה 256 אפשרויות. 4 אוקטטות עם 8 ביטים יוצרות למעשה מספר בן 32 ביטים (Bit), או 4 בייטים (Byte).

כמה אפשרויות שונות קיימות בכתובת בת 32 ביטים? 2 בחזקת 32 שווה ל-4,294,967,296, כלומר קצת פחות מארבע וחצי מיליארד כתובות IP שונות. העובדה שבתיכון תצורת כתובות ה-IP המקורית נלקחו בחשבון "רק" 32 ביטים גורמת לבעיה חמורה של הידלדלות מאגר כתובות ה-IP הזמין בעולם. מה שהיה טוב לתחילת שנות השמונים, (לצורך הדיוק ההיסטורי, IPv4, שהיא הגרסה הנוכחית של TCP/IP, הוכנסה לשימוש רשמי בתאריך ה-1 לינואר 1983) בזמן שהאינטרנט כלל לא היה בתיכון, כבר מזמן לא מתאים לרשתות העולמיות בימינו. ולכן נעשו מאמצים רבים כדי לאפשר "קומבינות" או דרכים להתחמק מהמחסור התמידי בכתובות IP. על כך בהמשך.

## מרכיבי כתובת ה-IP

כתובת IP מחולקת ל-2 חלקים לוגיים: כתובת רשת וכתובת מחשב, או באינגליזית **Network ID** ו-**Host ID**.

- ה-**Host ID** משמש לזיהוי **המחשב הבודד** בתוך הרשת, או לפי דוגמה מציאותית יותר, לזיהוי הבית ברחוב ע"י מתן מספר בית, או למשל את מספר הטלפון הייחודי בתור אותו אזור חיוג.
- ה-**Network ID** מזהה את **כתובת הרשת**, כלומר (לפי הדוגמה שלנו) את מספר הרחוב או אזור החיוג ברשת הטלפון הציבורית.

אם לשני מחשבים (Hosts) יש את אותה כתובת רשת (Network ID) אז הם נמצאים על אותה רשת, או בעצם הם נמצאים באותו רחוב. אם יש להם כתובות רשת שונות, אז מן הסתם הם נמצאים על רשתות שונות, או ברחובות שונים.

לשני מחשבים (Hosts) באותה רשת (Network ID) אסור שיהיה את אותו Host ID. לדוגמה, אי אפשר לבנות 2 בתים בעלי המספר 100 ברחוב הרצל. אפשר בית מספר 100 ברחוב הרצל, ואפשר לחזור על המספר 100 גם ברחובות אחרים, העיקר שלא באותו רחוב – או בעצם באותה רשת. אסור שבאזור חיוג 03 יהיו 2 מניי טלפון עם אותו מספר טלפון. אפשר להעניק מספר כמו 6165678 באזור חיוג 03, ואת אותו מספר גם באזור חיוג 04, אבל בטח שלא בתוך אותו אזור חיוג.

אם נגדיר בטעות לשני מחשבים הנמצאים באותו סגמנט פיזי של הרשת 2 כתובות IP עם Network ID שונה (למשל: **192.168.3.67** ו-**12.87.0.2** – בהמשך נזכיר איך ניתן בקלות לזהות איזה חלק מהכתובת הוא ה-**Network ID** ואיזה הוא ה-**Host ID**) אז כששני המחשבים ירצו לתקשר זה עם זה הם יחשבו (בצדק) שהם לא נמצאים באותה רשת ולכן הם ישאפו לצאת החוצה לנתב שמכונה לרוב בשם Default Gateway – על כך בהמשך.

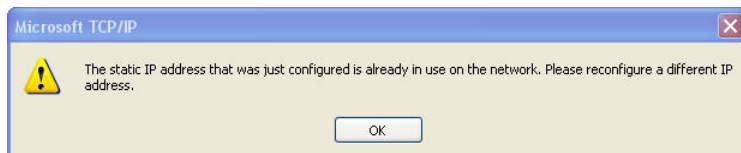
במידה ונגדיר את 2 המחשבים שנמצאים באותו סגמנט פיזי של הרשת 2 Network IDs שונים, המחשבים שירצו לתקשר זה עם זה ירצו לצאת החוצה דרך הנתב, אבל בגלל שהנתב איננו מודע לשינויים שביצענו, הוא יקבל את המידע מהמחשב אבל לא ידע מה לעשות איתו, ולכן יחזיר אותו לשולח. במקרה כזה לא יוכלו 2 המחשבים לתקשר זה עם זה.

## חוקי יסוד בכתובות IP

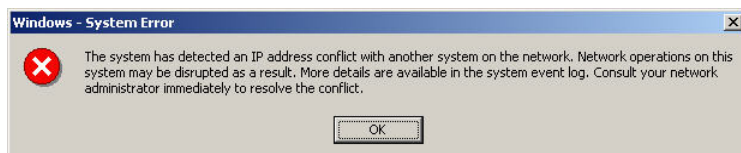
לכתובות IP בפרוטוקול ה-TCP/IP הוגדרו מספר חוקי יסוד אותם אנו חייבים לזכור. אסמן אותם באדום, רישמו אותם לפניכם:

### 1. כתובת ה-IP חייבת להיות ייחודית בתוך אותה רשת.

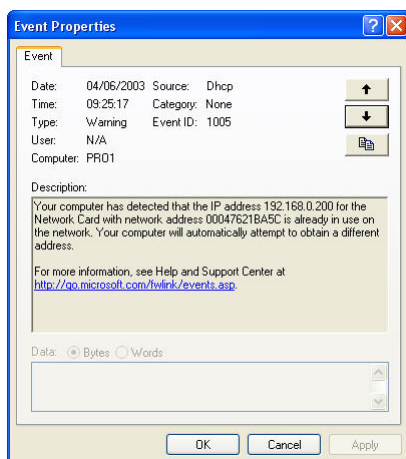
כמו שכבר אמרנו: אסור 2 מחשבים עם אותה כתובת בתוך אותה רשת. אם תרצו, תוכלו לבנות 2 רשתות נפרדות אחת מהשניה ולתת למחשבים באחת מהן כתובות זהות למחשבים ברשת השניה. אבל ברגע ששתי הרשתות יחבורו יחד, המחשבים בעלי הכתובות הזהות לא יוכלו לתקשר זה עם זה.



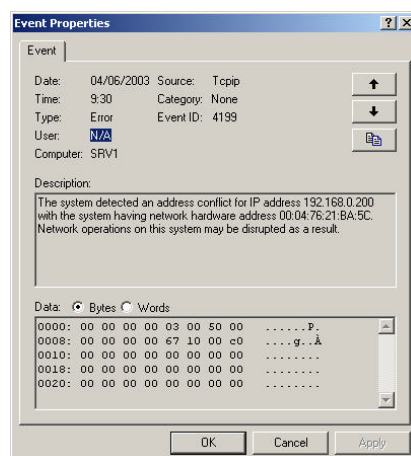
בתחנות NT ומעלה (W2K, XP, Win2003) מחשב שיגדיר לעצמו כתובת IP שתפוסה כבר ע"י מחשב אחר באותה רשת - יקבל הודעת שגיאה בסגנון כזה:



המחשב השני, זה שבו הוגדרה הכתובת המקורית, יקבל הודעת שגיאה בנוסח הזה:



במחשב שניסה לתפוס את כתובת ה-IP שהיתה שייכת למישהו אחר הודעה זו תירשם גם ב-Event Viewer כהודעת שגיאה מס' 1005:



ואילו במחשב שנפגע, זה שניסו לקחת לו את כתובת ה-IP, הודעה זו תירשם גם ב-Event Viewer כהודעת שגיאה מס' 4199:

הודעות שגיאה אלה יופיעו אך ורק במחשבי NT ומעלה, ובמחשבים בעלי מערכות הפעלה ישנות יותר יהיה קשה יותר למנהל הרשת לאתר את התקלה ולהבין שמדובר בכפילות כתובות IP. כתובת ה-IP "שייכת" רק למחשב שבו היא הוגדרה קודם, כך שכל עוד לא נבצע איתחול למחשב המקורי - כתובת ה-IP תמשיך לשרת אותו ורק אותו. כל זה כדי שהמחשב ה"פוגע" לא יצליח להשבית את תפקודו של המחשב ש"נפגע".

```

C:\WINDOWS>ipconfig /all
Windows IP Configuration

Host Name . . . . . : proi
Primary Dns Suffix . . . . . : 
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  : 
   Description . . . . . : 3Com EtherLink 10/100 PCI For Complete PC Mana
   Physical Address (MAC) . . . . . : 00-04-76-21-B0-5C
   DHCP Enabled. . . . . : No
   IP Address . . . . . : 0.0.0.0
   Subnet Mask . . . . . : 0.0.0.0
   Default Gateway . . . . . : 
   DNS Servers . . . . . : 192.168.0.200
C:\WINDOWS>_

```

אגב, ניתן בקלות לראות שיש לי קונפליקט של כתובות IP גם אם נריץ את פקודת `ipconfig /all` על המחשב ה"פוגע". בתוצאה נוכל לראות:

שימו לב שכתובת ה-IP במחשב ה"פוגע" היא 0.0.0.0 וכך גם ה-Subnet Mask. איך יודעים שלמחשב יש בעיה? הרי לא בהכרח יש כאן שגיאה.

אולי האדמיניסטרטור לא נתן לו בכוונה כתובת IP ולכן היא מופיעה כאפסים? אולי המחשב אמור לקבל כתובת משרת DHCP אבל טרם קיבל אותה ולכן היא מופיעה כאפסים? התשובה פשוטה: אם נתבונן בהגדרות ה-TCP/IP עצמן (ב-GUI) נראה שלמחשב הוגדרה כתובת IP, ואילו בשורת הפקודה אנחנו רואים אפסים, מה שמצביע בבירור על בעיה. לגבי ה-DHCP ועצם קבלת כתובת ממנו – רואים בפלט שבשדה DHCP Enabled מופיעה המילה No, מה שמצביע בבירור על כך שלמחשב יש כתובת IP מוגדרת ידנית, ושהוא איננו אמור לקבל כתובת IP משרת DHCP (זוהי המשמעות של המילה NO בשדה ה-DHCP Enabled) אבל העובדה שאנו רואים אותה כאפסים, מצביעה על תקלה.

החוק הבא:

## 2. לכל המחשבים הנמצאים ברשת אחת חייבת להיות אותה כתובת רשת או Network ID

כבר אמרנו: מחשבים שנמצאים על אותה רשת חייבים את אותה קידומת, אחרת הם יחשבו שהם לא נמצאים על אותה רשת. בהמשך נראה כיצד "יודעים" המחשבים לזהות את כתובת הרשת שלהם.

## 3. אסור Network ID שמתחיל ב-127

כתובת IP שמתחילה ב-127 משמשת לצורך בדיקה עצמית ואין לה שימוש ברשת – Loopback Address. לא משנה לי מה מופיע אחרי ה-127, כל כתובת שמתחילה ב-127 היא פסולה ולא ניתן להשתמש בה. למעשה, אם תבצעו בדיקה ע"י פקודת Ping לכל כתובת שמתחילה ב-127, מי שיענה לכם הוא המחשב שלכם עצמכם.

## 4. אסור Network ID שמתחיל ב-0

אין רשת 0. הרשת הראשונה האפשרית היא רשת 1, כלומר 1.0.0.0 (זה כמו לקבל מספר הטלפון בלי קידומת. אין כזה דבר. לכל קו טלפון יש קידומת. לחלופין, כתובת של בית ללא שם רחוב). למעשה, הסיבה האמיתית לביטול רשת ה-0 היא העובדה שכתובת 0.0.0.0 שמורה לצורך הגדרת מה שנקרא ה-Default Route. נדון בכך, אולי, במאמר אחר.

(מיד נסביר איך תוכלו לזהות איזה חלק מכתובת ה-IP הוא ה-Network ID, ואיזה חלק הוא ה-Host ID).

עוד 2 חוקים:

## 5. אסור Host ID שבו כל הביטים המהווים את ה-Host ID הם בעלי ערך של 1 בבינארי.

## 6. אסור Host ID שבו כל הביטים המהווים את ה-Host ID הם בעלי ערך של 0 בבינארי.

אם נמשיך בדוגמת מספרי הטלפון, אז אסור שבאותו אזור חיוג יהיו מספרי טלפון שבהם כל הספרות הן 0 או 9. כמה מספרים כאלה יש בכל אזור חיוג? בדיוק 2. למה רק 2? כי כמה מספרים יכולים להיות מורכבים מספרות שהן כולן 0 או 9? או המספר הראשון או המספר האחרון ברצף הקומבינציות.

כך גם בכתובות ה-IP. כאן אמנם הספרות 0 עד 9 לא מופיעות כיוון שאנחנו סופרים בבינארית, ולכן נתייחס ל-0 עד 1. מכיוון שאנחנו מחוייבים תמיד להוריד מהחשבון את שני המספרים הקיצוניים ביותר, הכתובות שבהן

כל הביטים הם 0 ואלה עם שבהן כל הביטים הם 1, הרי שבכל פעם שנרצה לחשב את מספר המחשבים ברשת נפחית 2 מהתוצאה הכוללת של כמות כתובות ה-IP הזמינות בכל רשת.

אבל רגע! בעצם, מדוע אסור Host ID שכולו 0 או כולו 1? הסיבה האמיתית לכך הוא העובדה שכתובת IP בעלת Host ID שכולו 0 מסמלת לנתבים ניתוב שנקרא "This Network", ואילו Host ID שכולו 1 מסמל לנתבים Broadcast מקומי. כאמור, נדון בנקודות אלה במאמר אחר.

**הערה חשובה לגבי חוקים 5 ו-6:** תלמידים מתחילים נוטים להתבלבל ולחשוב שרק לאוקטטה האחרונה אסור להיות 0 או 255. לא כך הוא הדבר. תמיד צריך להסתכל על כל ה-Host ID ככלל וכשלם, ולא על אוקטטות בודדות. למשל, כתובת IP של 10.1.1.0 היא כתובת חוקית וכשרה למהדרין. מצד שני, כתובת IP כמו 192.168.1.0 היא לא כתובת נכונה. חלקכם אומר בוודאי לעצמו "ברור, כי הכתובת השניה היא מ-Class C ולכן היא לא חוקית". מבלי להסביר ברגע זה מהו Class C ובכלל, מהם ה-Classes השונים (תוכלו לקרוא על כך בהמשך), הרי שלמרות שלמראית עין אותו בחור צודק, הרי שלפעמים, בתנאים מסויימים, גם כתובת כמו 192.168.1.32 היא לא נכונה ולא חוקית, ועל כך, כאמור, בהמשך.

הגדרה לא נכונה של כתובת IP עלולה לפגוע בתקשורת המחשב. לדוגמה, אם נגדיר 2 מחשבים שנמצאים מטר אחד ליד השני, על אותו LAN, כבעלי Network ID שונה, הרי הם לא יוכלו לתקשר זה עם זה כלל. איך יודעים מה נכון ומה לא נכון? איך יודעים מהי כתובת חוקית ומה לא? תוכלו לקרוא על כך בהמשך.

בדוגמה שלפניכם 2 האוקטטות הראשונות הן ה- Network ID, כלומר מספר הרשת, ואילו 2 האוקטטות האחרונות הן ה- Host ID, כלומר מספר המחשב בתוך אותה רשת:

**132.37.68.92**

נציב אותה בתוך תבנית צבעונית:

132	37	68	92
Network ID		Host ID	

זוהי רק אפשרות אחת מני רבות מכיוון שתיכף נראה שיש יותר מאפשרות אחת לחלק את 4 האוקטטות ביניהן ל- Network ID ול- Host ID.

**בעיה:** איך "ידוע" מחשב בעצמו איזה חלק מה-IP הוא ה- Network ID ואיזה חלק הוא ה- Host ID?

**תשובה:** המחשב מזהה את חלוקת ה- Network ID וה- Host ID בתוך כתובת ה-IP לפי מרכיב מספרי שנקרא **Subnet Mask**.

## Subnet Mask

ה- Subnet Mask הוא ערך מספרי נוסף המלווה כל מחשב ומציין למחשב איזה חלק מתוך כתובת ה-IP הוא ה- Network ID, ואיזה חלק הוא ה- Host ID.

למשל (לפי הדוגמה הקודמת), נחזור על הכתובת:

**132.37.68.92**

מתחתיה נכתוב את ערך ה- Subnet Mask המתאים:

**255.255.0.0**

נכניס את הנתונים לתוך טבלה צבעונית כדי להקל על ההבנה:

132	37	68	92
255	255	0	0

בדוגמה זו, כל אוקטטה שמתחתיה מופיע הערך 255 נחשבת כחלק מה- Network ID, וכל אוקטטה שמתחתיה מופיע ערך של 0 נחשבת כחלק מה- Host ID. ה- Subnet Mask מופיע בדרך-כלל בצורה קבועה עם מספרים כמו 255 ו-0, אבל בהמשך נראה איך אפשר לשנות אותו כדי שיתאים לצרכנו.

כדי להיות לגמרי כנים, ההגדרה הקודמת לא מדויקת ב- 100%. המחשב לא מסתכל על ה- 255 ועל ה- 0 בפורמט העשרוני שלהם, אלא דווקא בצורה הבינארית. למעשה כדי להיות מדויקים יותר עלינו לומר שכל ביט מכתובת ה- IP, שמתחתיו, ב- Subnet Mask, כתוב הביט 1 – שייך לצד של ה- Network ID, ואילו כל ביט בכתובת ה- IP, שמתחתיו, ב- Subnet Mask כתוב הביט 0 – שייך לחלק של ה- Host ID.

אם נתבונן בדוגמה הקודמת בצורה בינארית נראה משהו כזה:

10001000	0100101	01000100	01011100
11111111	11111111	00000000	00000000

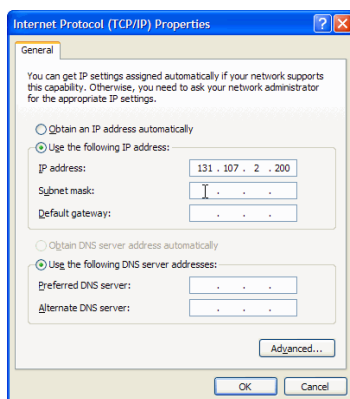
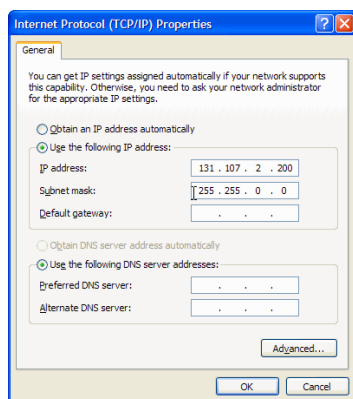
תוכלו לראות מיד למה התכונותי כשאמרתי שה- Subnet Mask עובד בעצם בצורה בינארית. מצד ימין אנחנו רואים רצף של ביטים בעלי ערך של 1. רצף הביטים הזה מתחיל בביט השמאלי ביותר, ומסתיים, בדוגמה הספציפית הזו, לאחר הביט ה- 16 במספר. אם נתבונן באוקטטות כשלמות ולא כביטים יחידים, אז קל להבחין שבמקרה הזה, ה- Network ID מסתיים מיד לאחר האוקטטה השניה, ולאחריו מתחיל רצף של ביטים בעלי ערך של 0. כאן בעצם מתחיל ה- Host ID שלנו.

במקרה שלנו הגבול בין כתובת הרשת לכתובת המחשב עובר בדיוק ברווח בין האוקטטה השניה לאוקטטה השלישית, אבל ישנם מצבים בהם רצף ה- 1 איננו נעצר ב"גבול הגזרה" של האוטטה, אלא "פולש" אל תוך האוקטטה הבאה. הציור הבא ממחיש את הדרך שבא המחשב באמת "רואה" את הכתובת ואת ה- Subnet Mask שמתחתיו:

1000100001001010100010001011100
11111111111111111000000000000000

כדי ללמוד על הדרך שבה אנו מחשבים את "גבול הגזרה" קרא בהמשך.

**בעיה:** למה אנחנו צריכים להגדיר בעצמנו עבור המחשב איזה חלק מהכתובת הוא ה- Network ID ואיזה חלק הוא ה- Host ID? האם המחשב לא "יודע" זאת בעצמו?



**תשובה:** בעיקרון, המחשב מסוגל "לנחש" איזה חלק הוא הוא ה- Network ID ואיזה חלק הוא ה- Host ID, אבל הוא לא יכול להיות "בטוח" במאה אחוז עד שלא נגדיר לו את הערך בעצמנו. למשל, בתחנת W2K תוכל להזין כתובת IP ולהתקדם עם העכבר אל השדה של ה- Subnet Mask, ובאופן אוטומטי המחשב ינסה "לעזור" לך ע"י זה שהוא יזין בעצמו את ה- Subnet Mask המצופה מכתובת ה- IP שאותה הקלדת.

בתחנת Win98 זה לא יקרה, ולמעשה המחשב יתן לך ללחוץ על OK למרות שלא הכנסת את השדה של ה- Subnet Mask כנדרש. כמובן שקונפיגורציה כזו לא תוכל לעבוד. בנוסף, אמנם למחשבים בעלי מערכות הפעלה "מודרניות" יש GUI ש"עוזר" לנו לנחש ערכים נחוצים, אבל להתקנים רבים אחרים לא קיים GUI וגם אם ישנו כזה, הרי שהוא לא ממש יוצא מגדרו כדי לעזור למנהל הרשת המתחיל. אם תצטרכו להגדיר נתב זה או אחר מתוך שורת הפקודה שלו (CLI) תוך זמן קצר תגלו שללא ידע מעולה בהגדרת כתובות IP לא תגיעו רחוק.

## שני הערכים הללו, ה- IP Address וה- Subnet Mask הם ערכים שחובה עלינו להגדיר אותם עבור כל מחשב ומחשב ברשת ה- TCP/IP.

אם לא נעשה זאת, אותו מחשב לא יוכל לתקשר בפרוטוקול זה. הגדרה לא-נכונה של Subnet Mask עלולה לגרום למחשב להתבלבל ולחשוב שהוא נמצא על רשת אחרת מזו שהוא נמצא בה באמת, או, לחלופין, לחשוב שכתובת מסויימת לא נמצאת על הרשת שלו אלא על רשת אחרת. עוד על ה- Subnet Mask בהמשך.

בואו נתבונן בחלק השלישי בפאזל הגדרת פרוטוקול ה- TCP/IP.



## Default Gateway

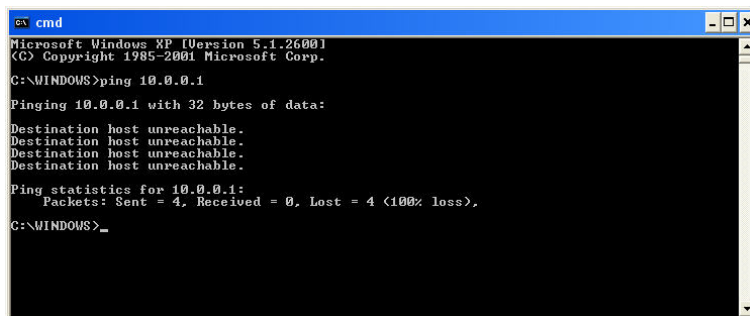
כיצד יכולים שני מחשבים לתקשר ביניהם במידה והם נמצאים על רשתות שונות? איך אפשר להתקשר בטלפון לחבר שנמצא בעיר אחרת? אם אחייג את מספר הטלפון שלו סתם כך אני עלול להגיע לאדם אחר הנמצא בעיר שלי והמחזיק באותו מספר טלפון. אני חייב לציין אזור חיוג.

במחשבים, התקשורת בין 2 הרשתות מתבצעת באמצעות מתווך בין 2 הרשתות. למתווך זה קוראים **נתב** או באנגלית **Router**. כשמגדירים למחשב את כתובת היציאה שלו החוצה מהרשת הפנימית, אנחנו קוראים להגדרה זו בשם **Default Gateway**. אנא נסו לא להתבלבל בין המונח Default Gateway בהקשר של TCP/IP ונתבים, לבין המונח Gateway בהקשר הגלובלי שלו שבא לציין רשת או תוכנה שיודעת לתרגם בין שפות או פרוטוקולים שונים (כמו למשל Gateway Services for NetWare).

מנקודת מבטו של המחשב, ה- Default Gateway הוא בעצם כתובת IP של אותו מחשב או נתב שמשמש כיציאה החוצה מן הרשת המקומית אל רשת או רשתות אחרות. זוהי כתובת ה- IP של הנתב המשמש כיציאה המומלצת והרגילה של התקשורת היוצאת מהרשת שלנו אל רשתות מרוחקות אחרות. במידה וישנן כמה יציאות כאלה, אז ה- Default Gateway יהיה הכתובת של היציאה האופטימלית מהרשת שלנו (אופטימלית מתייחס למהירות הנתב, קירבה פיזית, זמינות וכו').

אם לא נגדיר Default Gateway, הרי שמחשב לא ידע מאיפה עליו להוציא את התשדורות שמיועדות לרשתות שהן שונות מהרשת שלו.

אם לא נגדיר Default Gateway, אז מחשב יהא מוגבל בתקשורת אך ורק עם המחשבים האחרים ברשת הפנימית שהוא נמצא עליה, וכל פניה למחשב שנמצא על רשת אחרת מזו שלו, תיענה בהודעה הבאה: "Destination host unreachable".



```

C:\WINDOWS>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Destination host unreachable.
Destination host unreachable.
Destination host unreachable.
Destination host unreachable.

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\WINDOWS>_
  
```

למשל, אם יש לנו כתובת IP 192.168.0.100 ואין לנו Default Gateway אז כשנסה לעשות פינג (Ping) למחשב אחר שנמצא ברשת אחרת (זכורים? Network ID שונה משלי) אז נקבל תוצאה כזו:

זה הזמן לעוד חוק יסוד:

### 7. Default Gateway חייב להיות באותה רשת כמו המחשבים אותם הוא אמור לשרת!

כלומר, כתובת ה- IP של ה- Default Gateway חייבת להיות באותה רשת כמו זו של המחשב שרוצה לצאת דרכו. לכתובת שלי ולכתובת של ה- Default Gateway חייב להיות Network ID זהה!!!

למה הכוונה? אם למשל יש לי מחשב עם כתובת IP של **192.168.2.200**, ואנחנו רוצים להגדיר לו את כתובת ה- Default Gateway, אני יכול לתת כתובת כמו למשל **192.168.2.1**.

אבל אני לא יכול לתת כתובת כמו למשל **192.168.3.1** או **131.107.3.1** או כל כתובת IP אחרת שנמצאת על סגמנט (מקטע רשת) שאיננה הרשת שלי. חשוב מאוד להבין את זה: זה כמו לומר לבנאדם "אם יש שריפה תצא מדלת החירום שנמצאת בבנין ממול". אני חייב דלת שנמצאת אצלי בחדר, לא בבניין אחר!

אגב, נהוג (אבל לא הכרחי) לתת את הכתובת הראשונה או האחרונה במקטע של רשת (מה שנקרא Network Segment) ל- Default Gateway. כלומר, ברשת **192.168.2.0** נהוג לתת את **192.168.2.1** או את **192.168.2.254** עבור הכתובת של ה- Default Gateway, וזה בעיקר בגלל סיבות של קלות ניהול. יותר נוח לאדמיניסטרטור ויותר נוח לאלה שעובדים איתו.

## אפשרויות להגדרת ה-TCP/IP

כאמור, כשנגדירים מחשבים לתקשר באמצעות פרוטוקול ה-TCP/IP חייבים להגדיר למחשבים את שלושת הפרמטרים שהוזכרו לעיל. במערכות מבוססות מיקרוסופט קיימות 2 דרכים לגדיר אותם: הגדרה ידנית והגדרה אוטומטית. במערכות הפעלה מיקרוסופטיות רבות ישנו גם מנגנון שלישי עליו נדבר מיד. מן הסתם, גם במערכות הפעלה אחרות כדוגמת NetWare של נובל, MacOS של אפל, לינוקס ואחרות יש אפשרות להגדרה ידנית או אוטומטית של כתובות ה-IP.

### • הגדרה ידנית

במונח "הגדרה ידנית" מתכוונים לכך שהאדמיניסטרטור או אדם אחר מטעמו הולך לכל אחד מהמחשבים או הנתבים ומזין ידנית את הגדרות ה-TCP/IP. כך הוא מוודא שלכל מחשב או נתב תהיה כתובת IP קבועה מראש, כתובת שלא תשתנה לעולם כל עוד לא שינו אותה ידנית. כמובן שהגדרה ידנית היא הזמנה לצרות ולעומס אדמיניסטרטיבי. דמיין את עצמך רץ עם רשימה של 200 תחנות וכתובות ה-IP של כל אחת מהן ומנסה למצוא כתובת אחת פנויה עבור המחשב הנייד של הבן של המנכ"ל שהרגע הגיע מחו"ל וחייב להתחבר לרשת, וכל זה בשעה 6 בערב אחרי שאתה עובד מהבוקר על כל מיני קריאות של משתמשים מטומטמים! מצד שני, ברשת קטנה יהיה נוח אם תדע בדיוק איזו כתובת יש לכל מחשב ומהן הכתובות המדויקות של השרתים.

### • הגדרה אוטומטית

במונח "הגדרה אוטומטית" מתכוונים לכך שברשת יש שימוש בשרת **DHCP** שיחלק את הגדרות ה-TCP/IP באופן אוטומטי למחשבים השונים. DHCP, או **Dynamic Host Configuration Protocol**, הוא שרת שמחלק עבורנו את הכתובות ומבצע את העבודה השחורה. כל שעלינו לעשות הוא להגדיר בצורה נכונה את השרת ולתת לו את הטווחים מהם הוא יכול להתחיל לחלק (המושג נקרא Scope בעגת ה-DHCP). ה-DHCP לא יכול לטעות (אלא אם הגדרת אותו באופן לא נכון, אבל אז, שוב, אתה הוא זה שטעה ולא ה-DHCP...) ולכן לא תיתכנה התנגשויות או תקלות הנובעות כתוצאה מבילבול בכתובות. בצורה כזו תוכל גם לחסוך בכמה כתובות IP שיישמרו במאגר של ה-DHCP אם לא נמצא להן דורש, לעומת כתובות IP שמוקצות באופן ידני למחשבים שיתכן וכבר זמן רב אינם בפעולה.

עם זאת, עם המעבר לשימוש ב-DHCP יכול להיווצר מצב שבו המחשבים יקבלו בכל פעם כתובת IP אחרת. זה בסדר גמור במקרה שמדובר במחשבים המשמשים כ- Clients. אבל כשמדובר במחשבים חשופים (מחשבים הממלאים תפקיד מרכזי ברשת, כמו שרתי קבצים, שרתי הדפסה, Domain Controllers, WINS, DNS וכו') או במכשירים חשופים (כמו נתבים) זה מצב שאנחנו רוצים למנוע. לכן בד"כ עבור מחשבים פחות חשובים (כמו תחנות עבודה וניידים) נשתמש בהגדרות אוטומטיות, אבל עבור המחשבים החשובים, השרתים, הנתבים והמדפסות נגדיר את הכתובות באופן ידני.

### • APIPA

אפשרות שלישית להגדרת כתובות ה-IP היא ע"י שימוש במנגנון חכם שמסוגל להקצות לעצמו כתובות IP. הצורך בהגדרה שכזו עלול להתקיים בשני מקרים: במידה והאדמיניסטרטור לא יודע כיצד לעשות זאת או במקרה ששרת ה-DHCP לא זמין מסיבה זו או אחרת. למנגנון זה קוראים **APIPA** (או **Automatic Private IP Addressing**) והוא מופעל ברירת מחדל בזמן התקנת המערכת.

כל עוד מערכת ההפעלה (Win9X, W2K, XP, Win2003) מותקנת באופציות ברירת המחדל שלה, ה-APIPA יופעל במידה והמערכת מחפשת שרת DHCP ולא מסוגלת למצוא אותו בתום כחצי דקה. במידה והמערכת מוצאת שרת DHCP, השרת יכתוב למערכת את כתובות ה-IP ושאר ההגדרות שלה. אך אם שרת ה-DHCP לא זמין ברגע ההדלקה, המערכת תגדיר את עצמה באמצעות APIPA.

APIPA מגדיר למחשב כתובת רנדומלית בטווח הכתובות 169.254.0.1 ועד 169.254.255.254 ורק בטווח זה. לא ניתן להגדיר למחשב לקבל כתובת APIPA מטווח אחר.

כאשר המערכת מפעילה את עצמה עם APIPA היא תבחר לעצמה כתובת IP רנדומלית מתוך הטווח הנ"ל, וכדי לוודא שאף אחד אחר לא הקצה לעצמו במקרה את אותה כתובת (יש שם כ- 65 אלף אפשרויות...) היא תבצע בדיקה ברשת לראות האם מישהו עונה לאותה כתובת. היה ואף אחד לא עונה, הכתובת תוגדר בהצלחה. היה וענה מישהו לאותה קריאה, המערכת תנסה להקצות לעצמה מספר רנדומלי אחר, וחוזר חלילה.

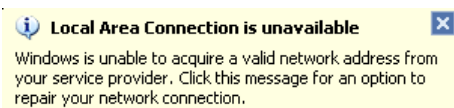
בלי שום קשר, כל 5 דקות תנסה המערכת למצוא שרת DHCP בשנית, והיה הוא נמצא, הוא יגדיר את המערכת וה-APIPA יפסיק לפעול. עוד על APIPA במאמר נפרד.

## • הערה לגבי Windows XP ו- Windows Server 2003

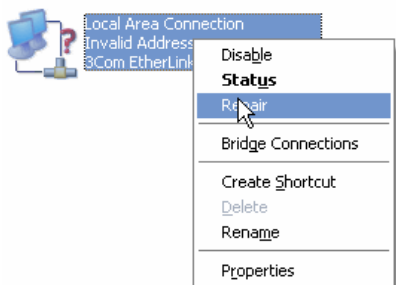
ב- Windows XP ו- Windows Server 2003 יש מנגנון שמודיע למשתמש שיש בעיה בהשגת כתובת IP משרת DHCP.



אייקון כרטיס הרשת בתוך חלון ה- Network Connections יקבל סימן שאלה אדום:



וליד שעון הזמן מצד ימין למטה יופיע הבלון הבא:



לחיצה על הבלון או בחירה ב- Repair מתוך תפריט הלחצן הימני על האייקון של כרטיס הרשת ינסה לתקן את הבעיה.

פעולת ה- Repair גורמת לתהליכים הבאים:

1. ביצוע Broadcast על מנת לבקש כתובת IP חדשה
2. ניקוי ה- ARP cache
3. ניקוי ה- NetBIOS name cache
4. ניקוי ה- DNS name cache
5. רישום מחדש בשרת ה- WINS
6. רישום מחדש בשרת ה- DNS

במידה והתהליך מצליח אייקון סימן השאלה ייעלם. אופציית ה- Repair טובה גם כדי לבצע את פעולות 2 עד 6 גם במידה והמחשב מוגדר לעבוד עם כתובת IP ידנית. כדאי לזכור אותה.

## • ביטול ה- APIPA

כדי לבטל את מנגנון ה- APIPA יש לערוך את הרגיסטרי של המחשב (כמובן בהירות תוך כדי גיבוי הערכים הקיימים, רק למקרה הצורך). פתח את עורך הרגיסטרי (Regedit.exe) ולך ל:

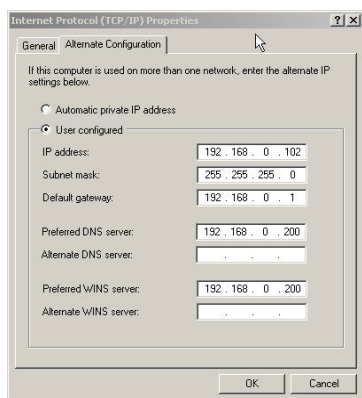
HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\adapter\_name

צור ערך חדש מסוג REG\_DWORD:

IPAutoconfigurationEnabled

ותן לו ערך של 0 (אפס). כדי להחזיר את המנגנון יש להפוך את הערך הנ"ל ל-1 (אחד).

תוכלו לקרוא עוד על כך במאמר שלי: [http://www.petri.co.il/disable\\_apipa.htm](http://www.petri.co.il/disable_apipa.htm)



## • הערה לגבי Windows XP ו- Windows Server 2003

ב- Windows XP ו- Windows Server 2003 יש מנגנון שמאפשר עבודה עם כתובות IP חלופיות בנוסף ל- APIPA. מנגנון זה נקרא **Alternate Configuration**. כלומר ניתן להגדיר למחשב לחפש כתובת מ- DHCP, אבל אם לא ימצא, שישתמש בסט הכתובות הבא (וכאן מגדירים את כתובת ה- IP החלופית, את ה- Subnet Mask והגדרות נוספות כמו Default Gateway וכו'). שימוש אחד בתכונה זו הוא במקרה ויש ברשותכם מחשב נייד המבלה חלק מהזמן ברשת החברה שלכם (בה קיים DHCP המקצה לו כתובת IP), אבל חלק אחר מן הזמן הוא מחובר לרשת הביתית שלכם, בה לא קיים (לצורך הדוגמה) מנגנון הקצאת כתובות IP, ולכן ברשת זו נוח לכם להקצות למחשב הנייד כתובת ידנית קבועה אבל אלטרנטיבית.

## בדיקת הגדרות ה-TCP/IP

### • מערכות NT/XP/W2K/2003

```

C:\cmd
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT>ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IP Address . . . . . : 192.168.237.179
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.237.254

C:\WINNT>

```

איך בודקים את הגדרות ה-TCP/IP של המחשב שלנו?

תחת מערכות NT/XP/W2K/2003 מקישים **ipconfig** בשורת הפקודה ומקבלים משהו כזה:

```

C:\cmd
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT>ipconfig /all

Windows 2000 IP Configuration

Host Name . . . . . : instructor
Primary DNS Suffix . . . . . : class.lab
Node Type . . . . . : Broadcast
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : class.lab

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    Description . . . . . : Intel(R) PRO/100+ Management Adapter
    Physical Address. . . . . : 00-80-B3-24-45-37
    DHCP Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    IP Address . . . . . : 192.168.237.179
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.237.254
    DHCP Server . . . . . : 192.168.237.250
    DNS Servers . . . . . : 192.168.237.254
    Lease Obtained. . . . . : Thursday, September 19, 2002 12:40:58 PM
    Lease Expires . . . . . : Thursday, September 19, 2002 7:40:58 PM

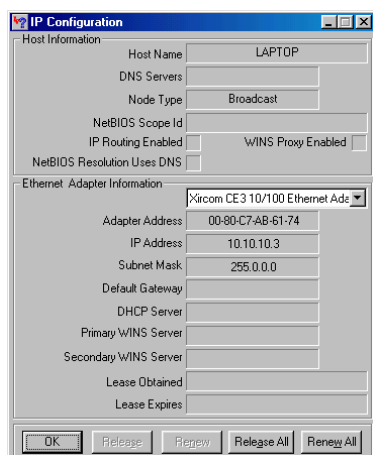
C:\WINNT>

```

לפקודת **ipconfig** יש פרמטרים רבים, כמו גם לכל שאר פקודות ה-TCP/IP. אם נקיש **ipconfig /all** נקבל הרבה יותר מסתם כתובת IP, אלא גם את כל שאר הגדרות הפרוטוקול, כמו הכתובת הפיסית של כרטיס הרשת (ה-MAC Address) ונתונים רבים אחרים.

שימו לב, ברוב רובן המכריע של פקודות שמקישים משורת הפקודה יש לשים רווח בין הפקודה עצמה ללוחסן - "/" שמגיע אחריה. כך למשל יש להקליד **ipconfig /all** ולא **ipconfig/all**, למרות שבדוגמה הספציפית הזו הפקודה דווקא כן תעבוד גם בלי רווח, הרי שכאמור ברוב הפקודות האחרות תקבלו הודעת שגיאה.

### • מערכות Win95/98



תחת Windows 95/98 יש תוכנה שנקראת **Winipcfg** שמראה את אותם נתונים אבל בצורה גראפית:

ניתן באמצעות התוכנה לבצע שיחזור ו/או בקשה לחדש את כתובת ה-IP של כל אחד מכרטיסי הרשת שהותקנו על המחשב או עבור כולם יחד. בנוסף ניתן לקבל אינפורמציה לגבי הגדרות שונות של הכרטיסים.

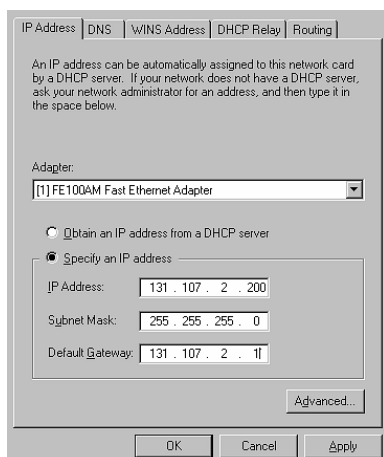
בגרסאות מאוחרות יותר של Win98 מופיעה גם תוכנת ה-**ipconfig** הרגילה, אולם היא מוגבלת ביכולות שלה לעומת הפקודה המקבילה במערכות Windows XP/2000/2003.

## הגדרת ה-TCP/IP על תחנות עבודה ושרתים

איך משנים את הגדרות ה-TCP/IP של המחשב שלנו?

### • Windows NT 4.0

ב-Windows NT 4.0 - לחצן ימני על Network Neighborhood, מאפיינים, ואז – Protocols ומאפיינים של TCP/IP. מקבלים מסך כזה:

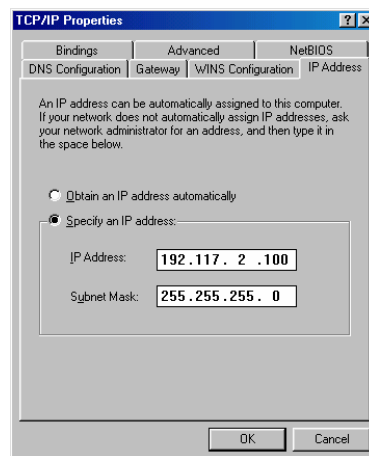
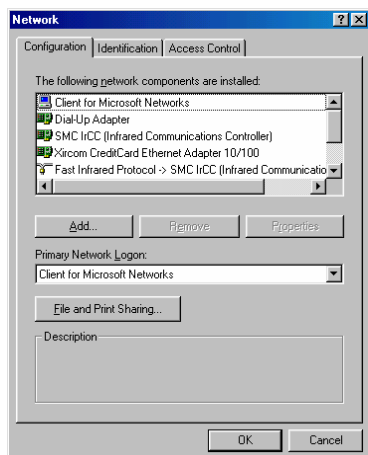


מכאן אפשר לשנות את כל הגדרות הפרוטוקול והגדרות שונות, כמו כתובות DNS, WINS, וכו'. ב-NT ניתן בלי בעיה להסיר את ה-TCP/IP ולהתקינו מחדש, אבל אז תיאלצו לבצע התקנה מחדש של ה-Service Pack האחרון שברשותכם – בד"כ זה SP6a. עוד על כך באתר שלי.

ב-NT כל שינוי ברמת ה-TCP/IP מצריך בד"כ ביצוע Restart.

### • Windows 9X

ב-Windows 95/98 ניגשים למסך ההגדרות באותו אופן כמו ב-NT, רק ששם הוא נראה קצת שונה. כמובן שאפשר לגשת לאותו מסך גם דרך אייקון ה-Network בלוח הבקרה:



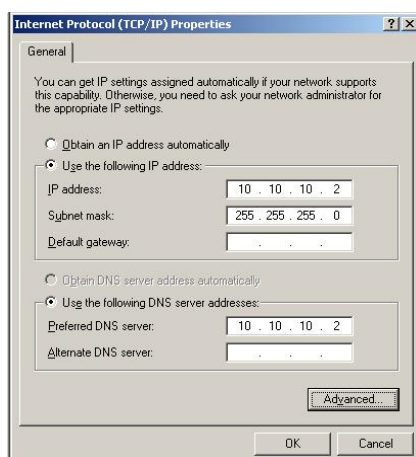
וכאשר נכנסים להגדרות הפרוטוקול מקבלים מסך כמו זה שמופיע בצד ימין של הדף.

גם ב-Win9X, כמו ב-NT, כמעט כל שינוי ברמת כתובות ה-IP או הגדרות אחרות דורש ביצוע Restart.



### • Windows 2000/XP/2003

תחת Windows 2000 ומערכות ההפעלה שבאו אחריו ניגשים להגדרות הפרוטוקולים וכרטיסי הרשת ע"י לחצן ימני על My Network Places, מאפיינים, ואז לחצן ימני על Connection ומאפיינים. מקבלים משהו כזה:



ואז אם ניגשים למאפיינים של TCP/IP מקבלים חלון הגדרות כזה:

במערכות W2K/XP/2003 אין צורך, בד"כ, לבצע איתחול לאחר שינוי בהגדרות ה-TCP/IP.

### טיפ למנהלי רשת מתחילים

תוכנת ה-Ipconfig תחת Windows 2000, Windows XP ו-Windows Server 2003 השתפרה בתחומים רבים וכוללת בתוכה פקודות חדשות רבות. כדאי להתעניין בנושא. גם תוכנות עזר אחרות השתנו ברבות הזמנים וכעת (תחת XP למשל) הן עושות דברים שלא היו ניתנים לביצוע תחת NT ואפילו לא תחת W2K. כדאי להתעכב עליהן.

מנהל רשת טוב מכיר את המערכת שלו היטב. גם אתם, אם ברצונכם להפוך לאנשי מקצוע מיומנים תהיו חייבים להכיר ולדעת מה אומרת כל שורה בתוצאות הפקודה, ובעיקר איך להסתכל על התוצאות ולהקיש מהן לגבי תקלות או בעיות אפשריות בהגדרות המחשב. הפקודות הבאות חשובות לנו כאדמיניסטרטורים. לא רק הפקודה והפרמטרים שלה, אלא מה בדיוק היא מבצעת ומה ההבדל בינה לבין פקודה אחרת:

PING  
IPCONFIG  
HOSTNAME  
ROUTE  
NETSTAT  
NBTSTAT  
ARP  
TRACERT  
PATHPING  
FINGER  
NSLOOKUP

בנוסף, רצוי מאוד להכיר תוכנות ניטור כדוגמת ה-Network Monitor שמגיעה בצורה מובנית עם Windows 2000/2003 (בגרסאות ה-Server בלבד, ב-Pro היא לא מובנית) וגם בגרסה מלאה ב-SBS 2000/2003. התוכנה מאפשרת ניטור של התעבורה ברשת, מעקב אחרי מחשבים ש"מזבלים" את הרשת, איסוף וצפייה ב-Frames (כולל בתוכן שלהם) וכן בדיקות באטחח לזיהוי תעבורה לא-מוצפנת.

תוכנת ניטור מעולה אחרת הניתנת להורדה בחינם מהאינטרנט היא Ethereal: <http://www.ethereal.com>

בנוסף, אני ממליץ על תוכנה חגיגית נוספת בשם Dice המאפשרת ניתוח יעיל יותר של תוצאות הלכידה של תוכנות הניטור. הגרסה העדכנית עומדת על 2.9.4 ונותן להוריד אותה מכאן: <http://www.ngthomas.co.uk/dice.htm>

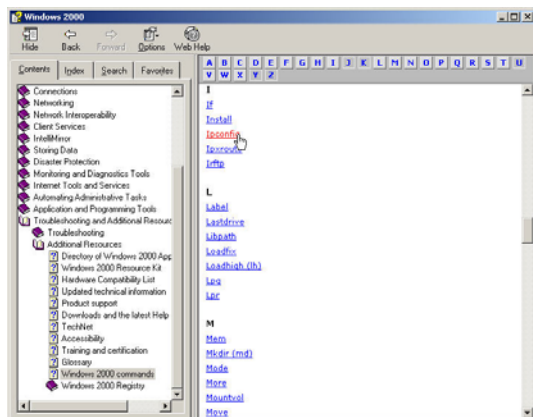
תוכנה נוספת בעלת חשיבות עליונה למנהלי רשת מנוסים היא פקודת ה-Netsh ב-Windows 2000/XP/2003. היא עושה פעולות מגוונות כגון הגדרת כתובות IP וכרטיסי רשת משורת הפקודה, הגדרת שרתי DHCP, הגדרת טבלאות ניתוב ושירותי ניתוב ועוד. לדוגמה, כדי להגדיר כתובת IP למחשב מתוך שורת הפקודה ניתן בקלות ליצור סקריפט שמכיל את הפקודה הבאה (כל הפקודה היא שורה אחת ארוכה):

```
netsh interface ip set address name="Local Area Connection" source=static  
addr=192.168.1.100 mask=255.255.255.0
```

באמצעות Netsh ניתן לבצע גם פעולות שלא ניתן לבצע מתוך ה-GUI הרגיל, למשל הסרה והתקנה של TCP/IP (ב- Windows 2003/XP זה לא ניתן לביצוע מתוך ה-GUI), וכן התקנה, הגדרה, תמיכה ועבודה עם IPv6. במערכות XP Pro עם Service Pack 2 ניתן להשתמש ב- Netsh גם כדי להגדיר את ה-Firewall מתוך שורת הפקודה.

את הפקודות (כמו את רוב הפקודות שנכתבות בשורת הפקודה) ניתן לבדוק באמצעות הקשת שם הפקודה בצירוף לוכסן וסימן שאלה, כמו למשל `ipconfig /?`

לחלק מהפקודות צירוף זה הוא בלתי אפשרי, ולכן מקישים את שם הפקודה עצמה, כמו למשל `ping`, ואז מקבלים את העזרה שלו. בחלק אחר מהפקודות (כמו `Nslookup` ו-`FTP`) ניתן להקליד את המילה `Help` בתוך הפקודה עצמה, ואז מקבלים רשימה של תת-פקודות אפשריות ולפעמים גם את ההסבר שלהן.



ניתן ומומלץ לחפש גם בקובץ העזרה של Windows 2000. יש שם קטע שמתאר את כל הפקודות הקיימות במערכת ההפעלה כשהן מסודרות לפי ה- "א.ב. חפשו בפרק שנקרא Troubleshooting and Additional Resources (הפרק האחרון) את החלק שנקרא Additional Resources ובו את **Windows 2000 Commands** שמסדר את רוב הפקודות לפי ה- a,b,c, וסביר יפה לגבי כל אחת מהן.



ב- Windows XP וגם ב- Windows Server 2003 מנגנון העזרה משופר להפליא ומכיל גם הפניה לאתרי אינטרנט שונים העולים בזמן שאתה מחפש מושג כלשהו. גם כאן יש חלק שדן בפקודות המערכת לפי "א.ב. חפשו **Command-line utilities** ותוכלו למצוא את מאמר העזרה המלא המתאר כל פקודה לפי ה- a,b,c, ומפרט את כל האפשרויות השונות בכל פקודה.

**המלצתי:** בנו לעצמכם מערכת דוגמה אותה לא תחששו להרוס ו/או להשביט. לימדו היטב את המערכת ואת מרכיביה, הכירו היטב את פקודות ה- Command Line ואת הפרמטרים שלהם.

**המלצה נוספת – Virtual Machines:** במידה ויש לכם מחשב מספיק חזק (עם למעלה מ- 750 מ"ב זיכרון פיזי) תוכלו להקים מערכות וירטואליות נוספות על מערכת ההפעלה הקיימת שלכם. המערכות הוירטואליות הללו (המכונות גם Virtual Machines) מתנהגות כ- PC נפרדים לכל דבר, ומסוגלים לקבל עליהם התקנת מערכות הפעלה לפי בחירתכם (כולל כל מע' ההפעלה של מיקרוסופט, גירסאות לינוקס וכו'). המערכות הללו נראות כמחשבים לכל דבר ברשת, וגם מסוגלים לגלוש באינטרנט או להיות מוגדרות כבעלי תפקידים מיוחדים, כדוגמת Domain Controllers, שרתי דואר, שרתי DNS וכו'. מערכות אלה יכולות גם להישמר בתצורת Snapshots כך שכל פעולה שביצעתם אבל לא מוצאת חן בעיניכם יכולה להיות מוחזרת אחורה לאותו Snapshot שיצרתם. יתרון נוסף של המערכות הוירטואליות הוא שאמנם הן תופסות מקום בדיסק (מקום כגודל הדיסק הוירטואלי שהגדרתם להן) אבל מדובר רק בקובץ אחד גדול שניתן בקלות למחוק אותו או אפילו לשכפל אותו וכך ליצור עוד מכונה וירטואלית שהיא למעשה שיכפול של הראשונה.

כרגע ישנם שני מוצרים עיקריים בתחום המכונות הוירטואליות: **Wmware** ו- **Microsoft Virtual PC**. חיפוש בגוגל יתן לכם לינק להורדה של גירסה שיתופית מוגבלת בזמן. עוד המלצות לבניית רשת ביתית:

[http://www.petri.co.il/build\\_home\\_network\\_he.htm](http://www.petri.co.il/build_home_network_he.htm)



## הקצאת כתובות IP למחשבים

כאמור, לכל מחשב שעובד באמצעות פרוטוקול ה-TCP/IP חייבים להקצות לפחות 2 פרמטרים: **IP Address** ו-**Subnet Mask**. פרמטר ה-**Default Gateway** הוא אופציונאלי, מכיוון שבמידה והוא לא יהיה מוגדר, המחשבים שלנו עדין יוכלו לתקשר בינם לבין עצמם כל עוד הם נמצאים על אותה רשת, אך במידה ונרצה לצאת ממגבלות הרשת לא נוכל לעשות זאת ללא הקצאת ה-**Default Gateway**.

מהיכן ניתן להשיג נתונים אלה?

ובכן, כל עוד הרשת שלנו היא רשת מבודדת שאינה מחוברת לאינטרנט ולא לאף רשת אחרת, נוכל בנקל "להמציא" לעצמנו סט של כתובות ולהקצות אותן למחשבים שלנו. אך כמו שהזכרנו בעבר, ברגע שנרצה להתחבר לרשת אחרת באמצעות נתבים, נהיה חייבים לוודא שברשת האחרת לא הוקצו טווחי כתובות שעלולים ליצור קונפליקט עם אלה שהוגדרו ברשת שלנו.

## כתובות ציבוריות וכתובות פרטיות

כאשר אנחנו מחלקים כתובות IP (לא משנה באיזו שיטה – ידנית או אוטומטית) אנחנו יכולים לבחור בין 2 סוגי כתובות: **כתובות ציבוריות** (Public Addresses) ו**כתובות פרטיות** (Private Addresses).

### • כתובות ציבוריות

כתובת ציבורית היא כתובת שניתנת לנו ע"י האחראים על חלוקת הכתובות באינטרנט – גוף הנקרא **IANA** או **Internet Assigned Numbers Authority** (<http://www.iana.com>). לעיתים מתייחסים אליו גם בשם **InterNIC**. ברוב המקרים אנחנו צריכים **לשלם כסף** עבור הכתובות הללו. מחשבים שמקבלים כתובות ציבוריות יכולים להשתמש בכתובות הללו להתחבר לאינטרנט, והכתובות הללו הן שלנו לנצח (או לפחות עד שנפסיק לשלם עבורם). לדוגמה – כשמחייגים לספק האינטרנט שלנו, אנחנו מקבלים ממנו כתובת ייחודית שהיא אמנם בבעלותו של הספק, אבל הוקצתה לנו באופן דינאמי עד להתנתקות שלנו, ואז היא תוקצה ללקוח אחר שמתחבר באותו רגע שבו אנחנו התנתקנו. כשלקוח מעוניין בחיבור אינטרנט קבוע ובכתובת IP קבועה הוא משלם X כסף לספק שלו, והספק שומר את אותה כתובת (או בלוק של כתובות) עבור הלקוח לשימוש.

כאשר אתה, בתור מני של ספק זה או אחר, פונה אל הספק ומבקש ממנו הקצאה של X כתובות IP ציבוריות, אתה משלם לספק על-מנת שלא יקצה את אותן כתובות למנויים אחרים.

הבעיה עם הקצאת הכתובות הציבוריות היא בעיקר העובדה שאין מספיק מהן. מתוך 4 וחצי מיליארד כתובות IP אפשריות למעלה מ-50% כבר הוקצה ונמסר, לעיתים חנים אין כסף, לאירגונים, גופים ממשלתיים ואקדמיים וכן לספקי אינטרנט גדולים. אבל מכיוון שיוצרי האינטרנט לא חזו מראש את ההתפתחות העצומה שלו בזמן שהם חילקו את אותם טווחי כתובות, נוצר מצב שבו חברות בסדר הגודל של מיקרוסופט, IBM, אינטל ואחרות קיבלו, כל אחת, כ-16 מיליון כתובות IP ציבוריות, אבל בפועל משתמשת בפחות מעשירית מכתובות אלה.

לדוגמה, הנה חלק מהכתובות הנמצאות בשימוש בידי חברות אמריקאיות גדולות, וכן תאריך ההקצאה המקורית:

8.0.0.0	Dec 92	Bolt Beranek and Newman Inc.
13.0.0.0	Sep 91	Xerox Corporation
15.0.0.0	Jul 94	Hewlett-Packard Company
16.0.0.0	Nov 94	Digital Equipment Corporation
17.0.0.0	Jul 92	Apple Computer Inc.
18.0.0.0	Jan 94	MIT
19.0.0.0	May 95	Ford Motor Company
55.0.0.0	Apr 95	Boeing Computer Services
56.0.0.0	Jun 94	U.S. Postal Service

חברות גדולות רבות ביקשו, וקיבלו, בלי כל בעיה, מיליוני כתובות IP ציבוריות. רוב החברות והגופים הללו הם אמריקאים במקורם (בגלל שהשיטה הומצאה שם) אבל ישנם גם מספר גופים בינלאומיים אחרים שמחזיקים אצלם מאגר עצום של כתובות IP ציבוריות ללא כל צורך. ספקי האינטרנט הישראליים קיבלו גם הם את נתח הכתובות המתאים, אבל בגלל שמאגרי הכתובות הציבוריות הדלדלו במהירות, הספקים הישראליים קיבלו רק "נתחים" קטנים מאוד מן העוגה.



## בפועל, למרות ש- 50% מכתובות ה- IP הציבוריות נמסר, רק כ- 69 מיליון כתובות נמצאות בפועל בשימוש...

לפני מספר שנים יצאה דרישה לקבל חזרה את אותם טווחי כתובות לא מנוצלים. בפועל כמעט אף גוף לא טרח להחזיר את הטווחים שהוקצו לו, בין היתר בגלל העובדה שמדובר בנכס השווה מאות אלפי דולרים לחברה המחזיקה בו, אם לא למעלה מזה.

כדי לזהות למי שייכת כתובת IP ציבורית יש להשתמש בשאילת Whois אצל רשם הכתובות המתאים לאזור הגיאוגרפי בו אתה חושד שהכתובת נמצאת. הלינק לביצוע שאילתות לכתובות IP השייכות לאירופה והמזרח התיכון הינו: <http://www.ripe.net/whois>

### • כתובת פרטית

כתובת פרטית היא כתובת שלא רשומה בשום מקום (הכוונה שהיא לא ניתנת לנו ע"י איזה שהוא אירגון, ואנחנו לא צריכים לשלם עבורה). כתובות פרטיות הן כתובות שאנחנו ממציאים ומשתמשים בהם לצרכינו הפרטיים.

כאשר אנחנו משתמשים בכתובות פרטיות אנחנו לא יכולים להתחבר באמצעותם לאינטרנט. אף נתב ציבורי לא יעביר אלינו תשדורות, ולא יוציא מאיתנו תשדורות וזאת משום שנתבי האינטרנט לא מוגדרים להעביר תעבורה שנושאת בתוכה ייעדים המוגדרים בטווח הכתובות הפרטיות. אם נתחבר לאינטרנט עם כתובות פרטיות לא נוכל לעבור את הנתב של ספק האינטרנט שלנו, מה שמאפשר עבודה בו-זמנית של הרבה רשתות עם טווחי כתובות זהים וזאת מבלי להפריע זו לזו.

אם בכל זאת נרצה לחבר רשת כזו אל האינטרנט נצטרך להשתמש בשירות הנקרא NAT (Network Address Translation). כך, ע"י שימוש בבלוק כתובות ציבוריות קטן מאוד (לעיתים לא יותר מ- 2 או 6), נוכל בקלות לחבר אירגון גדול לאינטרנט, מבלי לתת דין וחשבון לאף אחד ומבלי להצטרך לרכוש יותר מאשר 2 או 6 כתובות IP ציבוריות. אלמלא היה שימוש ב- NAT הופך להיות כה נפוץ, מאגר כתובות ה- IP הזמינות היה נגמר מהר מאוד. רשת בעלת 2000 משתמשים יכולה בקלות להתחבר לאינטרנט תוך "ביזבז" של 2 כתובות IP ציבוריות בלבד, מה שמאריך לנו במקצת את מועד ההידלדלות הסופית של מאגר כתובות ה- IP הציבוריות בעולם.

ה- InterNIC הקצה לנו 3 טווחים לשימוש פרטי, ודאג שטווחים אלה לא יישמשו שום רשתות "אמיתיות" בשום מקום בעולם. יתרה מזאת, כתובות אלה יזכו להתעלמות מכוונת מצד נתבי האינטרנט במידה ומישהו ינסה לצאת באמצעותם החוצה לעולם הרחב. כתובות אלה מוגדרות במסמכים RFC 1816, RFC 1918 ו- RFC 1597, והן:

- 10.0.0.0 - 10.255.255.255 (10/8 prefix)
- 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)

(המשמעות של ה- 10/8 prefix וכו' תתברר לכם בהמשך).

בנוסף, הטווח של 169.254.0.0 שמור עבור תהליך הקצאת כתובות ה- IP האוטומטי, או APIPA.

לינק ל- RFC 1918: <http://www.faqs.org/rfcs/rfc1918.html>

כאשר אנחנו רוצים לבנות רשת באמצעות שימוש בכתובות ציבוריות אנחנו צריכים לרכוש מספק האינטרנט (ISP) שלנו כתובת של רשת, כמו למשל 137.57.0.0, וברשת זו להתחיל להגדיר את כל ה- Hosts אחד אחד.

מצד שני, בזמן שימוש בכתובות פרטיות, דבר ראשון שצריך לעשות זה לקבוע לעצמנו, ועל דעת עצמנו בלבד, את כתובת הרשת - למשל 10.0.0.0. מומלץ מאוד לבדוק קודם ב- RFC 1918 כדי לראות אילו כתובות יכולות לשמש אותנו. לאחר מכן עלינו להתחיל לחלק כתובות Hosts מתוך הרשת הזו לכל אחד מהמחשבים ברשת שלנו. בכתובות פרטיות אין כל צורך לתאם את חלוקת הכתובות עם אף אחד בעולם, וממילא אם נשתמש בכתובות מסוג זה אף אחד בעולם לא יוכל לתקשר איתנו אלא אם נשתמש בשירותים מתקדמים מסוג NAT וכו'.

כמובן שגם בשימוש בטווחי כתובות פרטיות עדין חלים עלי כל חוקי ה- TCP/IP שהוגדרו עד כה, והתנהגות המחשבים זהה להתנהגותם ברשת ציבורית. ההבדל היחיד הוא שאני יכול לעשות עם הרשת שלי מה שאני רוצה מבלי לבקש רשות או לקנות כתובות אמיתיות.

## הבדלים בין Unicast, Broadcast ו-Multicast

אחד הנושאים שנוטים לבלבל תלמידים מתחילים הוא ההבדלים בין תקשורת מבוססת Unicast, לזו המבוססת Broadcast, ולבין זו העובדת באמצעות Multicast. לכאורה מדובר בתחום שלא קשור ספציפית ל-TCP/IP אלא מתאים לרוב סוגי הפרוטוקולים, אבל בכל זאת שווה להזכיר את העניין בגלל חשיבותו בנושא הבנת הסגמנטציה ברשת TCP/IP.

### Unicast •

תקשורת מסוג Unicast דומה לאדם שמשוחח עם אדם אחר בקול רם בתוך חדר מלא באנשים אחרים. תקשורת זו מבוססת על ההנחה ששני מחשבים "משוחחים" זה עם זה ישירות. מחשב א' (להלן המקור) מכין את ה-Packets למשלוח ברשת בצורה כזו שרק מחשב ב' (להלן המקבל) יבין שהוא צריך לקרוא אותם. מחשבים אחרים אמנם "שומעים" את התעבורה העוברת דרך כבל הרשת (במיוחד כשמדובר ברשת מסוג "כוכב" - או Star - שבה יש רכזת - או Hub - במרכזה) אבל הם מתעלמים לחלוטין מהתעבורה הזו. מדוע מחשבים אחרים לא מעוניינים לקבל את התעבורה הזו? מכיוון שהמחשב השולח מכין את התעבורה כך שרק כרטיס הרשת של המחשב המקבל "יקרא" את התעבורה מהרשת.

לפי איזה פרמטר ניתן לגרום אך ורק למחשב המקבל לקרוא את המידע מהרשת? איך יכול המחשב השולח "לדעת" להכין את ה-Packets כך שיגיעו אך ורק אל המחשב המקבל? לפי פרמטר הנקרא MAC Address.

כתובת MAC, או Media access control address, היא כתובת פיזית הצרובה בכרטיס הרשת. יצרן החומרה מקבל הקצאה של טווח כתובות MAC מאיגוד ה-IEEE, ולפי הקצאה זו פועל לצרוב כתובת ייחודית לכל כרטיס רשת שהוא מייצר. דוגמה לכתובת MAC אפשר לראות בפקודת ipconfig /all. כתובת MAC נראית כך:

00-11-11-12-EC-DB

המחשב השולח מברר ברשת מהי הכתובת הפיזית של המחשב המקבל, ומרגע שהוא מצא אותה הוא "זוכר" אותה בתוך Cache דינאמי בזיכרון שלו. המחשב השולח מכין את ה-Packet כך שאחד הפרמטרים הגלויים ביותר והקלים ביותר לעיבוד בו הוא כתובת ה-MAC של כרטיס הרשת של המחשב המקבל, אבל לא שוכח להוסיף גם את כתובת ה-MAC של עצמו בתור המקור לתעבורה.

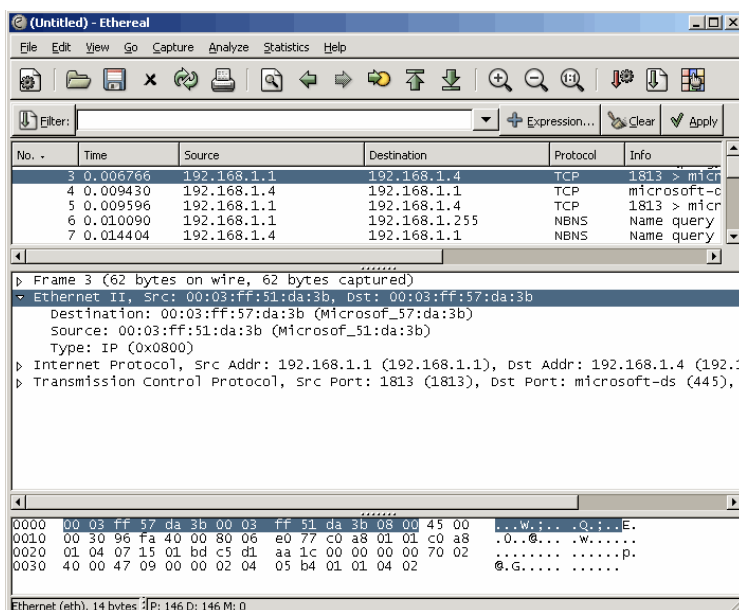
המחשב המקבל "מקשיב" לתעבורה ברשת, אבל מעוניין לקבל אך ורק תעבורה המיועדת לכתובת ה-MAC שלו עצמו. כשהוא מזהה כזה Packet הוא קורא אותו ומעביר אותו לעיבוד בשכבות גבוהות יותר, ומגיב בהתאם. במידה והוא מצופה להחזיר תשובה, הוא מכין את ה-Packet של התשובה לפי ה-MAC של המחשב השולח. גם המחשב השולח מקשיב לתעבורה, וכשהוא מזהה Packet שמכיל את הכתובת ה-MAC שלו עצמו ככתובת יעד, הוא קורא אותה, מעביר אותה לעיבוד וכך חוזר חלילה.

הרעיון מאחורי Unicast הוא שכאשר שני מחשבים מתקשרים זה עם זה המידע שעובר ביניהם ממילא "מזהם" את כל הרשת (שוב, במיוחד כשמדובר ברשת כוכב המבוססת על רכזת), אבל אף מחשב, פרט לשולח והמקבל לא מתעניינים בתעבורה הזו, למרות שאף אחד אחר לא יכול לתקשר באותו זמן. התקנה של Switch במקום

ה-Hub המרכזי תיעל עד מאוד את התעבורה כיוון שבמהלך העבודה לומד ה-Switch את מיקומן של כתובות ה-MAC ברשת, ונמנע מלשדר את התעבורה המועברת בין השולח לנמען לכל הרשת, אלא רק אל שני המחשבים הללו.

שידור מידע בין 2 מחשבים ב-Unicast הוא דרך יעילה להעברת מידע, אך מרגע שהמידע הזה אמור להגיע ליותר ממחשב אחד נוצר מצב שבו אותו מידע נשלח שוב ושוב, ושוב, מה שמובן מעמיס על תעבורת הרשת.

בצילום המסך המופיע בצד שמאל ניתן לראות תקשורת



Unicast שנוצרת כאשר מחשב אחד מנסה לפנות לשיתופים על מחשב אחר.

דוגמאות ל- Unicast ניתן למצוא כמעט בכל שידור ותקשורת מחשבים, למשל הורדת קבצים מהאינטרנט, משלוח דואר (כן, גם משלוח דואר להרבה נמענים בבת-אחת מבוצע בעצם אחד-אחד), שימוש בפקודות כמו Telnet ועוד.

## Broadcast •

תקשורת מסוג Broadcast דומה לאדם שמכריז עם רמקול בתוך חדר סגור מלא אנשים אחרים, ומחפש למשל, את כל האנשים שיש להם רכב מסוג מאזדה שחונה בחניון וחוסם את הכניסה. תקשורת זו מבוססת על ההנחה שמידע שמשודר על-ידי מחשב א' (להלן השולח) מיועד להגיע ולהיות מעובד על-ידי כלל המחשבים הנמצאים במקטע רשת מסויים. בתקשורת זו המחשב השולח מכין את ה-Packets למשלוח כך שכל המחשבים בסגמנט (מקטע הרשת) יקבלו, יקראו ויעבדו את המידע. השאלה האם הם יעשו עם המידע הזה משהו (למשל יחזירו תשובה כלשהי) תלויה בסוג המידע המועבר. בניגוד לתקשורת Unicast שבה כל המחשבים "שומעים" את התעבורה אך מתעלמים ממנה (אלא אם היא נושאת כתובת MAC שלהם עצמם) הרי שבתקשורת Broadcast כל המחשבים "שומעים" את התשדורת, קוראים אותה ומעלים אותה לעיבוד בשכבות גבוהות יותר של הפרוטוקול.

כיצד יודעים המחשבים שעליהם לעבד את המידע למרות שלא מדובר בכתובת ה-MAC שלהם עצמם? התשובה נמצאת בכתובת ה-Broadcast. כתובת זו כתובה בצורה כזו:

FF-FF-FF-FF-FF-FF

וכך כל מחשב ש"קולט" תשדורת העוברת ברשת הנושאת כתובת יעד כזו "מבין" שעליו לקרוא את התשדורת ולפעול בהתאם. למשל, בדוגמה של האדם המכריז בחדר עם רמקול ומחפש אנשים בעלי רכב מסוג מאזדה, כל הקהל שמע את ההכרזה, אבל לא רק שמע, אלא גם הקצה משאבי מערכת כדי לעבד את הנתונים. יתכן והרבה אנשים יענו לקריטריונים של השידור (הרבה בעלי מאזדות) אבל יתכן וגם אף אחד לא יענה לקריטריון (החדר מלא אנשים בעלי אופנועים). לו היה הכרוז צריך לבצע את השידור ב-Unicast הרי שהוא היה נאלץ ללכת אדם אדם, לטפוח לו על השכם, ולשאול אותו האם, במקרה, יש לו מאזדה בחניון. הכרוז היה נאלץ לבצע את אותה עבודה מספר פעמים רב.

אם כך, מתברר ששידור מידע בתצורת Broadcast הוא דרך יעילה להגיע, בזמנית, אל כמות גדולה של מחשבים, וזו מבלי להצטרך לשלוח את אותו מידע שוב ושוב ברשת כמו ב-Unicast. מצד שני, כמות גדולה של Broadcasts ברשת יגרום לכל המחשבים ברשת להתעסק בקריאה של כמות גדולה של מידע העובר ברשת, ולבזבז משאבי מערכת על עיבוד נתונים שלא בהכרח קשור אליהם. לכן נהוג לשאוף להקטנה של כמות ה-Broadcasts ברשת עד כמה שאפשר, למרות שבלתי אפשרי כמעט להגיע למצב נטל Broadcasts לחלוטין.

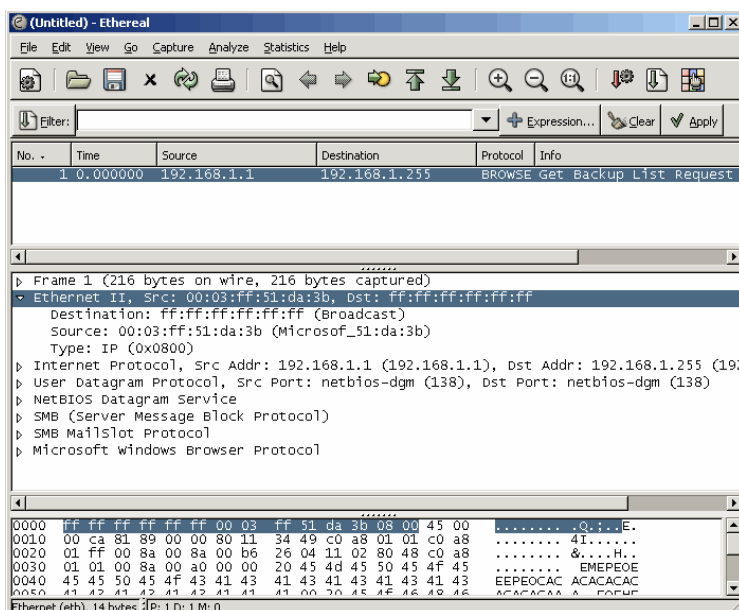
ל-Broadcasts יש חיסרון (או שמא נאמר יתרון?) נוסף, והוא חוסר היכולת של נתבים להעביר אותם לרשתות אחרות. כלומר כל תשדורת Broadcast שנובעת ממחשב ברשת A לא יכול בשום פנים ואופן להגיע לאף מחשב ברשת B או C בהנחה שמה שמחבר בין הרשתות הללו הוא נתב. אפשר לומר שנתב הוא מעין מחסום

לשידור Broadcast, ולרוב

מדובר ביתרון, למרות שלעיתים זה עלול ליצור בעיות מסוימות.

בצילום המסך המופיע בצד שמאל אפשר לראות דוגמה ל-Broadcast שנוצר כתוצאה מכך שיוזר על מחשב אחד ביצע דאבל-קליק על אייקון ה"שכנים ברשת".

דוגמאות נוספות לשידור Broadcast ניתן למצוא בפעולות רשת פנימיות כגון פעולות לוג-און של תחנות 95/98 ו-NT, ועוד. שידורים מסוג זה ניתנים להפחתה ע"י שימוש באמצעים כגון WINS.

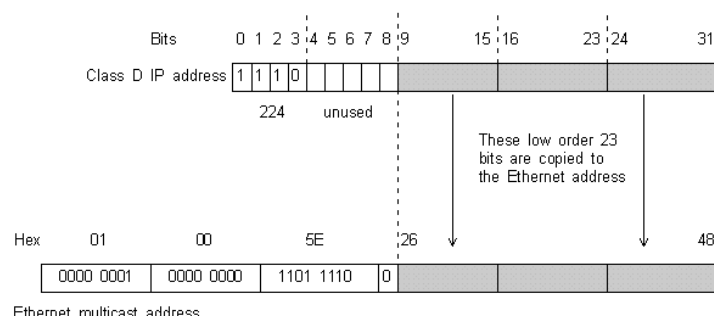


על כך במאמר אחר. מצד שני, כמעט כל תקשורת Unicast מתחילה בביצוע פעולת Broadcast אחת לפחות של המחשב השולח בכך שהוא בעצם שואל ברשת "מי זה מחשב ככה וככה?", ואז מחשב היעד מחזיר תשובה רגילה (מסוג Unicast) אל המחשב השולח ואומר לו "אני ככה וככה". מאותו רגע התקשורת עוברת לתעבורת Unicast בין שני המחשבים.

### • Multicast

תקשורת זו מבוססת על ההנחה שמידע יכול להגיע (כמו ב-Broadcasts) אל כמות גדולה (לעיתים עצומה) של מחשבים. אבל בניגוד ל-Broadcast רגיל שבו למחשבים הקולטים אין ברירה אם לקבל או לא את המידע, הרי שבתקשורת Multicast המחשב השולח מכין את המידע בצורה כזו שהוא יעבור בין כל המחשבים, אבל רק מחשבים (שימו לב – מחשבים, ברבים) שביקשו לקבל אותו אכן יקראו אותו.

תשדורת זו משלבת את היתרונות של 2 השיטות הקודמות. מכיוון שבניגוד לתשדורת Unicast שבה יש שימוש בכתובת ה-MAC של המחשב הספציפי שאמור לקבל את התשדורת, הרי שבתקשורת מבוססת Multicast השידור אמור להגיע למספר רב של מחשבים אשר לכל אחד יש MAC Address משלו. לכן המחשב השולח יוצר Packets המיועדים ל-MAC ספציפי המורכב מ-MAC אוניברסלי קבוע, פלוס תוספת ייחודית לאפליקציה המפעילה את מערכת הקליטה אצל המחשבים שאמורים לקבל את המידע. כתובת ה-MAC הזו תמיד מתחילה בתצורה קבועה כפי שמתואר בציר הבא:



כך רק מי שמפעיל את האפליקציה הייחודית באמת מקבל ומעבד את המידע, למרות שכמו ב-Unicast גם כאן המידע עובר בפועל דרך כל המחשבים.

תשדורות כאלה משתמשות בכתובות IP מ-Class D, ובטווח של החל מ-224.0.0.0 ועד ל-239.255.255.255. ה-IANA הקצה מספר כתובות Class D לשימוש מטרות ספציפיות של Multicast. למשל:

Address	Purpose
224.0.0.1	All hosts on a subnet
224.0.0.2	All routers on a subnet
224.0.0.5	All OSPF routers (DR Others)
224.0.0.6	All OSPF Designated Routers
224.0.0.9	All RIPv2 routers
224.0.0.10	All EIGRP routers
224.0.0.12	DHCP Server/Relay Agent
224.0.0.13	All PIM routers
224.0.0.14	RSVP Encapsulation
224.0.1.39	Cisco RP Announce
224.0.1.40	Cisco RP Discovery

דוגמאות של Multicast כוללות הקמת ערוצי שיחות ועידה, שידורי רדיו באינטרנט, שכפולי WINS אוטומטיים ועוד. ברגע שמחשב מסוים התחבר לאתר אינטרנט המשדר רדיו הוא למעשה "מאותת" על כוונתו להסכים לקבל את סוג המידע הזה. במקרה זה המידע היה משודר ברשת גם ללא בקשתו של המחשב, ולא משנה למשדר אם אין לו מקשיבים בכלל או אם יש לו 1000 מאזינים – הוא תמיד ישדר את אותה כמות מידע.

תשדורות מסוג Multicast מטופלות ע"י הנתבים ברשת שאחראים להפיץ את התשדורת אל הסגמנטים שאמורים לקבל אותם.

מקור נחמד של מידע אודות Multicast אפשר למצוא בלינק הבא:

<http://www.rhyshaden.com/multicas.htm>

## לסיכום –

זיכרו את חוקי היסוד:

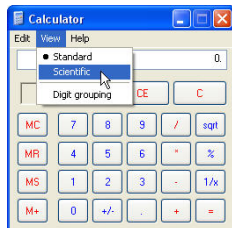
1. כתובת ה- IP חייבת להיות ייחודית בתוך אותה רשת.
2. לכל המחשבים הנמצאים ברשת אחת חייבת להיות אותו Network ID. במידה וה- Network ID שונה, המחשבים יהיו חייבים לערב נתב (או Default Gateway) בתקשורת ביניהם.
3. אסור Network ID שמתחיל ב- 127.
4. אסור Network ID שמתחיל ב- 0.
5. אסור Host ID שבו כל הביטים המהווים את ה- Host ID הם בעלי ערך של 1 בבינארי.
6. אסור Host ID שבו כל הביטים המהווים את ה- Host ID הם בעלי ערך של 0 בבינארי.
7. ל- Default Gateway חייב שיהיה אותו Network ID כמו למחשבים שאותם הוא אמור לשרת!
8. ללא Default Gateway לא יוכל המחשב לתקשר עם מחשבים אחרים בעלי Network ID שונה משלו (אלא אם הוגדרה לו טבלת ניתוב ידנית).
9. ברשתות פרטיות יש לעקוב אחר RFC 1918 על מנת לדעת באילו טווחי כתובות יש להשתמש.

## חלק שני

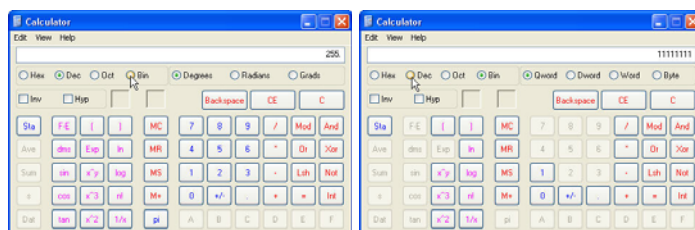
בחלק השני נדבר על המרה ממספרים בינאריים לעשרוניים ומעשרוני לבינארי, חלוקה ל- Classes השונים, אבחנה בין ה- Classes השונים (מה שמכונה Classful IP Addressing), נדבר על כמות ה- Network ID וה- Host ID בכל רשת, ועל האבחנה בחלוקת טווחי הכתובות לפי 3 ה- Classes הרגילים (לא נדבר על D ו- E במאמר זה).

### המרה מעשרוני לבינארי ולהיפך

הדרך הקלה ביותר היא ע"י שימוש במחשבון של Windows או באמצעות קיצור הדרך או ע"י הקשת calc בפקודת ה- Run. בתוך המחשבון צריך לעבור לתצוגה מדעית בתפריט View:



ואז אם נקיש מספר בעשרוני, נוכל ללכת לכפתור ה- Bin והפלא ופלא – המספר הפך לבינארי.



כן"ל, אם נהיה במצב Bin ונקיש מספר בבינארי, נוכל לעבור למצב Dec וראו איזה פלא – המספר הפך לעשרוני. לידיעת אלה מכם שמתכוונים למבחן, המחשבון קיים גם בזמן המבחן האמיתי, אבל אני ממש לא ממליץ להיעזר בו.

"למה אתה מתעקש שנמיר את המספרים בראש או ידנית ולא באמצעות מחשבון?" תשאלו.

תשובתי היא חד משמעית: מניסיוני התמליד חייב להבין את הפעולה שהוא עושה. אם נשתמש במחשבון אז אמנם הפעולה תתבצע מהר יותר, אבל התלמיד לא תמיד שם לב למה שהוא מקיש, ויתכן שבגלל טעות הבקלדה (כמו כל הטעויות שהופיעו בכוונה במשפט הקודם למשל...) המחשבון אמנם יתן תוצאה, אבל לא את התוצאה הנכונה. כדי להוכיח את טענתי נסו נא להמיר מספר כמו 57 לבינארי באמצעות המחשבון. כמה קיבלתם? 111001. זו אמנם תוצאה נכונה, אבל לצורך הבנה וביצוע נכון של תהליך ה- Subnetting אני צריך שתדעו שהתוצאה היא דווקא 00111001. "מה ההבדל?" תשאלו. ובכן ההבדל הוא במספר הביטים. שתי התשובות הן נכונות ואמיתיות, אבל אני צריך שתסתכלו על המספר בצורת 8 ביטים, ולא 6. עשו לכם מנהג של קבע, הפכו את העשרוני לבינארי ולהיפך באמצעות המוח ולא המחשבון. ככה גם תבינו את מה שאתם עושים, ולאחר תרגול מספיק תוכלו גם להבחין במגמות מסויימות בחשבונות שאתם עורכים ולזהות מראש נקודות מפתח מוגדרות מראש.

### מבינארי לעשרוני

אז איך הופכים מבינארי לעשרוני בראש? יש כמה שיטות. חלקכם זוכרים אולי שיטות שלימדו אותכם בבית-הספר. לי אין בעיה עם זה, אבל הרשו לי להציע לכם אלטרנטיבה פשוטה התפורה בדיוק לצרכים שלנו. מציירים טבלה קטנה בת 8 עמודות, בדיוק כמו זאת:


בטבלה יש 8 עמודות וכמה שורות שאתם רוצים. כל עמודה מציינת ביט אחד באוקטטה. בשורה העליונה כותבים את המספרים הבאים (לא צריך את הכל, רק את אלה הכתובים **באדום מודגש**):

$2^7=128$	$2^6=64$	$2^5=32$	$2^4=16$	$2^3=8$	$2^2=4$	$2^1=2$	$2^0=1$

(למי שלא מכיר, הסימן הקטן ^ מסמן חזקה...)

אם מתבוננים מימין לשמאל מיד רואים שעמודה אחת היא בעצם הכפולה של זו שלפניה, או בעצם לפי הנוסחה 2 בחזקת 0, 2 בחזקת 1, 2 בחזקת 2, 2 בחזקת 3 וכו' ועד ל-2 בחזקת 7 שזה שווה ל-128. אנחנו יכולים תיאורטית להמשיך עד שתכאב לנו היד, אבל במציאות אנחנו צריכים רק 8 ביטים (זוכרים? אוקטטה?), ולכן סופרים מ-2 בחזקת 0 ועד 2 בחזקת 7, סה"כ 8 ביטים.

יפוי. עכשיו בואו נניח שקיבלנו מספר בינארי מפוצץ כמו 00000101. לפני שאתם נבהלים, בואו כתוב אותו לפי הסדר בשורה השניה בטבלה שלנו:

128	64	32	16	8	4	2	1
0	0	0	0	0	1	0	1

כדי להפוך לעשרוני פשוט נסתכל איפה כתובה הספרה 1 ואיפה כתובה הספרה 0. איפה שכתוב 1, נרשום את הערך שמעליה, ואיפה שכתוב 0 נתעלם מהערך שמעליה. בדוגמה שלנו:

128	64	32	16	8	4	2	1
0	0	0	0	0	1	0	1

איזה ערכים סימנו? 1 ו-4. מה הסכום של 1+4? בדיוק 5. לכן, התוצאה של 00000101 בעשרוני זה 5. תוכלו לבדוק את התוצאה במחשבון, אבל כאמור המחשבון יתן לכם תוצאה קצת שונה: 101. אני יודע שזו באמת התשובה הנכונה, אבל אני רוצה שתכתוב אותה בצורת 8 ביטים, לכן המחשבון במקרה זה רק יעכב אותנו.

#### דוגמה נוספת:

00110111

128	64	32	16	8	4	2	1
0	0	1	1	0	1	1	1

$$32+16+4+2+1=55$$

#### דוגמה נוספת:

11101110

128	64	32	16	8	4	2	1
1	1	1	0	1	1	1	0

$$128+64+32+8+4+2=238$$

#### תרגיל:

הפוך מבינארי לעשרוני את הערכים הבאים (בדוק את עצמך באמצעות מחשבון):

11000011 .11	00001101 .1
01111111 .12	11101100 .2
11110111 .13	00001111 .3
11111111 .14	10110011 .4
10000000 .15	11110001 .5
01010101 .16	01101100 .6
11100000 .17	11111100 .7
10011111 .18	11000000 .8
11110000 .19	10110001 .9
11111000 .20	11111110 .10

**מעשרוני לבינארי**

ההמרה מעשרוני לבינארי נעשית באותה צורה, בעזרת הטבלה, אבל הפעם צריך להכניס ערכים עשרוניים ולקבל ערכים בינאריים.

נהפוך את המספר **131** לבינארי:

128	64	32	16	8	4	2	1

נבדוק: בעמודה הראשונה מצד שמאל, האם 128 נכנס ל-131?

כן.

כמה נשאר עודף?

**131** פחות 128 שווה 3.

לכן תחת 128 כותבים 1 וזוכרים שנשארו לנו עוד 3 להכניס לטבלה.

128	64	32	16	8	4	2	1
1							

נמשיך ימינה. האם 64 נכנס ל-3?

לא.

האם 32 נכנס ל-3?

לא.

האם 16 נכנס ל-3?

לא.

האם 8 נכנס ל-3?

לא.

האם 4 נכנס ל-3?

לא.

בכל העמודות האלה נרשום 0, כיוון שהערכים שלהם לא נכנסים ל-3.

128	64	32	16	8	4	2	1
1	0	0	0	0	0		

נמשיך ימינה. האם 2 נכנס ל-3?

כן.

נכתוב 1 תחת העמודה.

כמה נשאר?

3 פחות 2 שווה 1.

האם 1 נכנס ב-1?

כן.

נכתוב גם תחת העמודה אחרונה 1.

128	64	32	16	8	4	2	1
1	0	0	0	0	0	1	1

מה קיבלנו? התוצאה היא **10000011**, שזה בעשרוני בדיוק **131**.

**דוגמה נוספת:**

הפעם מקוצרת. הפוך **243** לבינארי.

128	64	32	16	8	4	2	1
1	1	1	1	0	0	1	1

משמאל לימין בטבלה: 128 נכנס ל-243 ונשאר עודף **115**. 64 נכנס ל-115 ונשאר עודף **51**. 32 נכנס ל-51 ונשאר עודף **19**. 16 נכנס ל-19 ונשאר עודף **3**. 8 לא נכנס ל-3. 4 לא נכנס ל-3. 2 נכנס ל-3 ונשאר עודף **1**. 1 נכנס ל-1.

לכן התוצאה היא **11110011**.



**תרגיל:**

הפוך מעשרוני לבינארי את הערכים הבאים (בדוק את עצמך באמצעות מחשבון):

254 .11	20 .1
127 .12	92 .2
256 .13	192 .3
255 .14	193 .4
168 .15	220 .5
248 .16	242 .6
200 .17	239 .7
224 .18	240 .8
242 .19	247 .9
128 .20	250 .10

הפוך את כתובות ה-IP הבאות מעשרוני לבינארי:

210.23.67.102	.1
66.23.148.0	.2
192.254.23.123	.3
144.207.78.1	.4
63.125.23.211	.5
192.25.128.36	.6
134.223.156.89	.7
127.0.0.1	.8
224.23.108.23	.9
223.78.27.144	.10

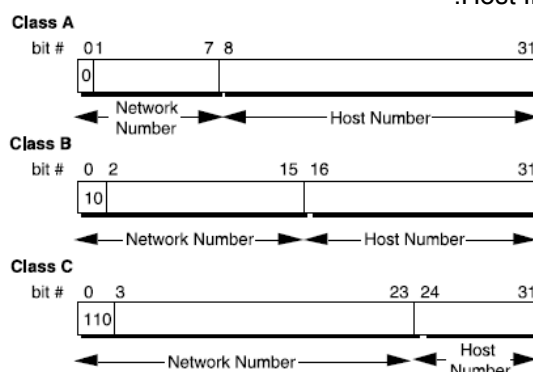
## חלוקה לפי Classes, או Classful IP Addressing

כש-IP הוגדר לראשונה ב-1981 נקבע כי כתובת ה-IP בת ה-32 ביטים תחולק לשני חלקים: החלק הראשון, השמאלי ביותר, שנקרא ה- Network ID (או Net ID בקיצור), והחלק השני, הימני ביותר, שנקרא ה- Host ID.

חלוקה זו צריכה למלא את מספר חוקים שהוגדרו מראש (ראה את 7 החוקים בחלק הראשון של המאמר), אבל החוקים שנוגעים לנו כרגע אומרים כי **כל המחשבים שנמצאים באותה רשת פיזית (אותו סגמנט) חייבים להיות בעלי Net ID זהה, אבל להיות שונים זה מזה ב- Host ID.**

כדי לתמוך בגמישות הנדרשת לחלוקה למספר רב של רשתות בעלות גדלים שונים, החליטו מתכנני האינטרנט לחלק את כתובות ה-IP בעולם ל-5 מחלקות או קבוצות הנקראות Classes. ה-Classes מציינים איזה חלק מתוך 4 האוקטטות משמש כ- Network ID, ואיזה חלק משמש כ- Host ID. רק 3 ה-Classes הראשונים נמצאים בשימוש שוטף, ואילו 2 ה-Classes האחרונים לא יכולים לשמש להקצאת כתובות IP.

כל Class מבין 3 ה-Classes הראשונים (A, B או C) מגדיר בצורה שונה את המיקום שבו כתובת ה-IP מתחלקת בין ה- Net ID ל- Host ID.



## החלוקה בין ה- Net ID ל- Host ID

- ב- **Class A** הגבול בין הביטים שמשמשים את ה- Net ID לבין הביטים שמשמשים את ה- Host ID עובר בין ביט מספר 7 לביט מספר 8. כלומר, ביט 0 ועד ביט 7 שייכים ל- Net ID, ואילו ביטים 8 ועד הסוף – ביט 31 – שייכים ל- Host ID.
- המסקנה, ברשת מסוג Class A האוקטטה הראשונה (8 הביטים הראשונים) שייכת ל- Net ID ואילו 3 האוקטטות האחרונות (24 הביטים הנותרים) שייכות ל- Host ID.
- ב- **Class B** הגבול בין הביטים שמשמשים את ה- Net ID לבין הביטים שמשמשים את ה- Host ID עובר בין ביט מספר 15 לביט מספר 16. כלומר, ביט 0 ועד ביט 15 שייכים ל- Net ID, ואילו ביטים 16 ועד הסוף – ביט 31 – שייכים ל- Host ID.
- כלומר, ברשת מסוג Class B שתי האוקטטות הראשונות (16 הביטים הראשונים) שייכות ל- Net ID ואילו 2 האוקטטות האחרונות (16 הביטים הנותרים) שייכות ל- Host ID.
- ב- **Class C** הגבול בין הביטים שמשמשים את ה- Net ID לבין הביטים שמשמשים את ה- Host ID עובר בין ביט מספר 23 לביט מספר 24. כלומר, ביט 0 ועד ביט 23 שייכים ל- Net ID, ואילו ביטים 24 ועד הסוף – ביט 31 – שייכים ל- Host ID.
- לכן, ברשת מסוג Class C שלוש האוקטטות הראשונות (24 הביטים הראשונים) שייכות ל- Net ID ואילו האוקטטה האחרונה (8 הביטים הנותרים) שייכת ל- Host ID.

## Leading Bit Pattern

בנוסף לאמור לעיל, חוקי יסוד ב- Classes מגדירים שבכל כתובת IP יופיע איזשהו קידוד בראש הכתובת שמאפשר לנתבים שדרכם עוברת התעבורה לדעת מיד בין איזה ביטים עוברת החלוקה בין ה- Net ID ל- Host ID. ע"י קריאה של 3 הביטים הראשונים בכל כתובת יודע הנתב מיד איפה לחפש את ה- Net ID, וזאת מבלי שייצטרך לקרוא את כל ה- 32 ביטים של הכתובת. לקידוד זה קוראים בשם **Leading Bit Pattern**.

הביטו בצירור שמופיע בעמוד הקודם:

- ב- **Class A** ה- Leading Bit Pattern הוא **0**. כלומר, הביט השמאלי ביותר חייב להיות תמיד 0 (אפס) ולא יכול להיות שום דבר אחר. אם הוא שונה מ- 0 (כלומר 1) הכתובת שעומדת לבוא לאחר מכן לא יכולה להיות כתובת מ- Class A, אלא אולי מ- Class B או C. לכן, נתב שקורא את הביט הראשון ורואה שהוא **0** מיד יודע לחפש את ה- Net ID אך ורק עד הביט ה- 7 (זיכרו, ב- Class A ה- Net ID הוא רק האוקטטה הראשונה).
  - ב- **Class B** ה- Leading Bit Pattern הוא **10**. כלומר, 2 הביטים השמאליים ביותר חייבים להיות תמיד 10 ולא יכולים להיות שום דבר אחר. אם הם שונים מ- 10 (כלומר 0 או 110) זו לא יכולה להיות כתובת מ- Class B, אלא אולי מ- Class A או C בהתאמה. לכן, נתב שקורא את הביטים הראשונים ורואה שהם **10** מיד יודע לחפש את ה- Net ID אך ורק עד הביט ה- 15 (זיכרו, ב- Class B ה- Net ID הוא 2 האוקטטות הראשונות).
  - ב- **Class C** ה- Leading Bit Pattern הוא **110**. כלומר, 3 הביטים השמאליים ביותר חייבים להיות תמיד 110 ולא יכולים להיות שום דבר אחר. אם הם שונים מ- 110 (כלומר 0 או 10) זו לא יכולה להיות כתובת מ- Class B, אלא אולי מ- Class A. לכן, נתב שקורא את הביטים הראשונים ורואה שהם **110** מיד יודע לחפש את ה- Net ID עד הביט ה- 23 (זיכרו, ב- Class C ה- Net ID הוא 3 האוקטטות הראשונות).
- הצורך ב- Leading Bit Pattern היה חיוני בימים הראשונים של הקמת טכנולוגיית הניתוב, כיוון שהנתבים המוקדמים לא הכילו כל מנגנון לתרגום כתובות היעד ל- Network IDs. בימינו, אין למעשה ל- Leading Bit Pattern חשיבות עקרונית, אבל הוא עדין נותר איתנו בגלל סיבות היסטוריות.
- אגב, כתובת IP מ- Class D חייבת שהביטים הראשונים באוקטטה הראשונה יהיו **1110**, וכתובת IP מ- Class E חייבת שהביטים הראשונים באוקטטה הראשונה יהיו **1111**, אבל אנחנו לא נתעסק בהם במאמר זה.

## כמות Networks וכמות Hosts בכל רשת

ברשת Class A אוקטטה אחת משמשת לשם הרשת ו- 3 אוקטטות משמשות לשמות המחשבים – Host ID. אבל באוקטטה הראשונה "לקחו" לנו את הביט הראשון (חייבו אותו להיות **0**), לכן נותרנו עם 7 ביטים "למשחק" עבור הרשת, ועוד 3 אוקטטות "למשחק" עבור ה- Host או 24 ביטים. כלומר 2 בחזקת 7 עבור הרשתות, ו- 2 בחזקת 24 עבור מספר ה- Hosts בכל רשת.

### רשת Class A:

ברשת מסוג Class A יש לנו אוקטטה אחת עבור ה- Net ID. התבוננו באוקטטה הראשונה משמאל (בסגול):

**Network ID . Host ID . Host ID . Host ID**  
**00000000.00000000.00000000.00000000**  
 7 Bits for Network ID and 8+8+8 Bits for Host ID

אך שימו לב לחוק יסוד מספר 4 (לא זוכרים? ראו עמוד 6 במאמר זה):

### 4. אזור Network ID שמתחיל ב- 0.

כלומר, אין רשת שבה כל הביטים הם 0, ולכן האפשרות הראשונה שלנו חייבת להיות לפחות ביט אחד מעל 0 (שימו לב לאוקטטה השמאלית ביותר), כלומר:

**00000001.00000000.00000000.00000000**

נהפוך לעשרוני ונראה מה יצא:

**1.0.0.0**

מהי הקומבינציה האחרונה שאפשר לעשות באוקטטה הראשונה?

**01111111.00000000.00000000.00000000**

(זיכרו שה- Leading Bit Pattern הוא 0 ולכן אני לא יכול "לגעת" בביט השמאלי ביותר)

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**127.0.0.0**

עצור! מה המשמעות של כתובת כזו? נכון מאוד! זוהי כתובת השמורה עבור ה- Loopback Address, ולכן אין לנו יכולים להשתמש בה. זיכרו את חוק יסוד מספר 3:

**3. אזור Network ID שמתחיל ב- 127.**

בלית ברירה נפחית ביט אחד מהמקסימום, כלומר מהכל 1, (שימו לב לאוקטטה השמאלית ביותר):

**01111110.00000000.00000000.00000000**

נהפוך לעשרוני ונראה מה יצא:

**126.0.0.0**

כלומר טווח כתובות הרשת השונות ב- Class A הוא בין 1 ל- 126.

אם כן, כמה אפשרויות יש לנו באוקטטה הראשונה?

כדי לדעת את התשובה נוכל לספור על האצבעות (אבל לצערי יש לנו רק 20, כולל הרגלים...), או שנלך בדרך המתמטית: "לקחו" לנו את הביט הראשון ונותרו לנו רק 7 ביטים לשחק איתם. 2 בחזקת 7 שווה 128 אפשרויות שונות. אבל מכיוון שהרשת הראשונה (הכל 0) והרשת האחרונה (127) "נופלות", יש לנו רק **126** אפשרויות שונות (מ- 1 ועד 126). כלומר הנוסחה לפיה נוכל לעבוד היא 2 בחזקת 7 מינוס 2.

כעת, מה לגבי מספר ה- Hosts האפשריים בכל אחת מ- 126 הרשתות השונות? יש לנו 126 רשתות שונות, אבל כמה מחשבים יכולים להיות בכל אחת מהרשתות האלה? ברשת מסוג Class A יש לנו 3 אוקטטות עבור ה- Host ID. נהפוך שוב לבינארי ונראה. הקצה התחתון של הקומבינציות האפשריות הוא:

**00000001.00000000.00000000.00000000**

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**1.0.0.0**

אבל זיכרו את חוק יסוד מספר 6:

**6. אזור Host ID שבו כל הביטים המהווים את ה- Host ID הם בעלי ערך של 0 בבינארי.**

בלית ברירה נחזור ונשנה את הכתובת לאחד יותר מהמינימום, כלומר נעלה ביט אחד מעל "הכל 0" (שימו לב לאוקטטה הימנית ביותר):

**00000001.00000000.00000000.00000001**

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**1.0.0.1**

נחזור לבינארי. נתחיל להעלות את הביטים אחד אחד:

```
00000001.00000000.00000000.00000010
00000001.00000000.00000000.00000011
00000001.00000000.00000000.00000100
00000001.00000000.00000000.00000101
```

וכן הלאה, ונגיע לקומבינציה הכי גבוהה:

```
00000001.11111111.11111111.11111111
```

נהפוך לעשרוני ונראה מה יצא:

**1.255.255.255**

אבל כמו שציינו בחוק יסוד מספר 5:

### 5. אסור Host ID שבו כל הביטים המהווים את ה- Host ID הם בעלי ערך של 1 בבינארי.

לכן נוריד מהאוקטטה האחרונה שהיא המקסימום האפשרי ביט אחד, ולמעשה נפחית ביט אחד כדי למנוע מצב של Host ID שהוא "הכל 1" (שימו לב לאוקטטה הימנית ביותר):

```
00000001.11111111.11111111.11111110
```

נהפוך לעשרוני ונראה מה יצא:

**1.255.255.254**

לכן, כל רשת מסוג Class A יכולה להכיל מחשבים בין הטווחים 0.0.1 עד ל- 255.255.254. בכמה אפשרויות בדיוק מדובר? אני יודע מהו הטווח הקיצוני הנמוך ומהו הטווח הקיצוני הגבוה, אבל אני רוצה לדעת בדיוק בכמה מחשבים מדובר. מכיוון שיש לנו 3 אוקטטות, ובכל אחת מהן 8 ביטים:

$$2^8 * 2^8 * 2^8 = 2^{24} = 16,777,216$$

רק שבמקרה זה נאלצנו גם להוריד את הקיצוני התחתון והקיצוני העליון (אי אפשר הכל 0 והכל 1) לכן מורידים 2 מהתוצאה הכוללת:

$$(2^8 * 2^8 * 2^8) - 2 = (2^{24}) - 2 = 16,777,216 - 2 = \mathbf{16,777,214}$$

למעשה אפשר לנסח פה מעין חוק עזר ולומר בביטחה שבכל פעם שמחשבים את מספר ה- Host IDs ברשת, יש להפחית 2 מהתוצאה הכוללת.

### רשת Class B:

ברשת מסוג Class B קיימות 2 אוקטטות עבור ה- Net ID ולא אחת, כמו ב- Class A. התבוננו ב- 2 האוקטטות הראשונות (בסגול):

```
Network ID. Network ID. Host ID. Host ID
10000000.00000000.00000000.00000000
6+8 Bits for Network ID and 8+8 Bits for Host ID
```

האפשרות הראשונה (הנמוכה ביותר) שלנו היא:

```
10000000.00000000.00000000.00000000
```

(זיכרו שה- Leading Bit Pattern הוא 10 לכן אני לא יכול "לגעת" בשני הביטים השמאליים ביותר)

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**128.0.0.0**

מה הקומבינציה האחרונה שאפשר לעשות באוקטטות של ה- Network ID?

10111111.11111111.00000000.00000000

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

191.255.0.0

כלומר טווח כתובות הרשת השונות ב-Class B הוא בין 128.0 ל-191.255.

כשבדקנו את רשתות Class A ראינו יש לנו בסה"כ 126 רשתות כאלה. כמה רשתות מסוג Class B קיימות בעולם?

החישוב הוא פשוט: כמה אפשרויות יש לנו באוקטטה הראשונה? "לקחו" לנו שני ביטים מובילים ולכן נותרו לנו רק 6 ביטים "לשחק" איתם. 2 בחזקת 6 שווה 64 אפשרויות שונות. באוקטטה השנייה יש לנו 8 ביטים מלאים, ולכן 2 בחזקת 8 שווה 256. נכפיל את 2 התוצאות: 64 כפול 256 = 16,384 אפשרויות שונות לרשתות ב-Class B:

$$2^6 * 2^8 = 2^{14} = 16,384$$

אגב, מדוע לא הורדתי 2 מהתוצאה הכוללת כמו ברשת Class A? התבוננו באוקטטה הראשונה וראו שחייבו אותנו שני הביטים הראשונים יהיו 10, ולכן בעצם לעולם לא יהיה מצב שבו כל הביטים יהיו 0 או 1, וממילא עברנו את 127, כך שאין מה להוריד כאן. לכן, ברשת מסוג Class B אין צורך להוריד 2 מהתוצאה הכוללת.

קעת נחשב את טווח ה-Hosts IDs בכל אחת מ-16,384 מהרשתות ואת מספר ה-Hosts האפשריים בכל אחת מהרשתות השונות. עבור ה-Host ID יש לנו 2 אוקטטות ולא 3 כמו ב-Class A. מן הסתם יש פחות אפשרויות, בואו נבדוק כמה בדיוק. נהפוך שוב לבינארי ונראה. הקצה התחתון של הקומבינציות האפשריות הוא:

10000000.00000000.00000000.00000000

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

128.0.0.0

כאן התרגולת אמורה להיות כבר מוכרת: אסור Host ID שבו כל הביטים הם 0, ולכן נחזור ונשנה את הכתובת לאחד יותר מהמינימום, כלומר נעלה ביט אחד מעל "הכל 0" (שימו לב לאוקטטה הימנית ביותר):

10000000.00000000.00000000.00000001

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

128.0.0.1

נתחיל להעלות את הביטים אחד אחד ונחפש את הקומבינציה הכי גבוהה:

10000000.00000000.11111111.11111111

נהפוך לעשרוני ונראה מה יצא:

128.0.255.255

גם כאן, כמו ב-Class A, אסור Host ID שבו כל הביטים הם 1, ולכן נפחית ביט אחד כדי למנוע מצב של Host ID שהוא "הכל 1" (שימו לב לאוקטטה הימנית ביותר):

10000000.00000000.11111111.11111110

נהפוך לעשרוני ונראה מה יצא:

128.0.255.254

כלומר, כל רשת מסוג Class B יכולה להכיל מחשבים בין הטווחים 0.1 עד ל- 255.254.

בכמה אפשרויות בדיוק מדובר? מדובר ב- 2 אוקטטות, כל אחת 8 ביטים. לכן:

$$2^8 * 2^8 = 2^{16} = 65,536$$

אבל שוב, אי אפשר הכל 0 והכל 1, לכן מורידים 2:

$$(2^8 * 2^8) - 2 = (2^{16}) - 2 = 65,534$$

### רשת Class C:

ברשת מסוג Class C יש לנו 3 אוקטטות עבור ה- Net ID ולא אחת או שתיים, כמו ב- Class A או B. התבוננו ב- 3 האוקטטות הראשונות (בסגול):

**Network ID. Network ID. Network ID. Host ID**

**11000000.00000000.00000000.00000000**

**5+8+8 Bits for Network ID and 8 Bits for Host ID**

האפשרות הראשונה שלנו היא:

**11000000.00000000.00000000.00000000**

(זיכרו שה- Leading Bit Pattern הוא 110 ולכן לא נוכל "לגעת" ב- 3 הביטים השמאליים ביותר)

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**192.0.0.0**

מה הקומבינציה האחרונה שאפשר לעשות באוקטטות של ה- Network ID?

**11011111.11111111.11111111.00000000**

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**223.255.255.0**

כלומר טווח כתובות הרשת השונות ב- Class C הוא בין 192.0.0 ל- 223.255.255

כמה אפשרויות יש לנו באוקטטה הראשונה? "לקחו" לנו שלושה ביטים מובילים ולכן נותרו לנו רק 5 ביטים "לשחק" איתם. 2 בחזקת 5 = 32 אפשרויות שונות. באוקטטה השנייה יש לנו 8 ביטים מלאים, ולכן 2 בחזקת 8 = 256. באוקטטה השלישית יש לנו 8 ביטים מלאים, לכן 2 בחזקת 8 שווה גם כאן ל- 256. נכפיל את התוצאות: 32 כפול 256 כפול 256 שווה **2,097,152** אפשרויות שונות לרשתות ב- Class C:

$$2^5 * 2^8 * 2^8 = 2^{21} = 2,097,152$$

כמו ב- Class B אין צורך להפחית 2 מסך כל הרשתות.

אמנם יש לנו למעלה מ- 2 מיליון רשתות, אבל מה לגבי מספר ה- Hosts האפשריים בכל אחת מ- 2,097,152 הרשתות השונות? זיכרו, ב- Class C יש לנו רק אוקטטה אחת, הימנית ביותר, שמשתת ל- Host ID. נהפוך שוב לבינארי ונראה. הקצה התחתון של הקומבינציות האפשריות הוא:

**11000000.00000000.00000000.00000000**

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

**192.0.0.0**

שוב נדקלם את החוק: אסור Host ID שבו כל הביטים הם 0, לכן נחזור ונשנה את הכתובת לאחד יותר מהמינימום, כלומר נעלה ביט אחד מעל "הכל 0" (שימו לב לאוקטטה הימנית ביותר):

11000000.00000000.00000000.00000001

נהפוך את האוקטטות לעשרוני ונראה מה יצא:

192.0.0.1

נתחיל להעלות את הביטים אחד אחד (00000010, 00000011, 00000100, 00000101 וכו') ונחפש את הקומבינציה הכי גבוהה:

11000000.00000000.00000000.11111111

נהפוך לעשרוני ונראה מה יצא:

192.0.0.255

אבל כמו שצינו בהתחלה, אסור Host ID שבו כל הביטים הם 1, ולכן נפחית ביט אחד כדי למנוע מצב של Host ID שהוא "הכל 1" (שימו לב לאוקטטה הימנית ביותר):

11000000.00000000.00000000.11111110

נהפוך לעשרוני ונראה מה יצא:

192.0.0.254

כלומר, כל רשת מסוג Class C יכולה להכיל מחשבים בין הטווחים 1 עד ל-254. בכמה אפשרויות בדיוק מדובר? כאן החישוב הוא פשוט יותר, אבל שוב, לצורך התרגול, מדובר באוקטטה אחת בת 8 ביטים. לכן:

$$2^8 = 256$$

אבל שוב, אי אפשר הכל 0 והכל 1, לכן מורידים 2:

$$256 - 2 = 254$$

כדי לסכם את כל המספרים הללו:

Class	Leading Bit Pattern (תחילית)	כמות הביטים שנותרו באוקטטה הראשונה	כמות הביטים באוקטטות האחרות השייכות ל-Net ID	סה"כ כמות הביטים המשמשים עבור Net ID ה-	כמות הרשתות הכוללות 2) בחזקת סה"כ כמות הביטים המשמשים עבור ה-Net ID (פרט ל- Class A שבו מפחיתים 2 מהתוצאה)
A	0	7		7	126
B	10	6	8	6+8=14	16,384
C	110	5	8 8	5+8+8=21	2,097,152

Class	כמות הביטים ב- Host ID			2 בחזקת כמות הביטים ב- Host ID	כמות ה- Host IDs הזמינים לכל רשת (2 בחזקת כמות הביטים ב- Host ID, פחות 2)
A	8	8	8	1,677,7216	16,777,214
B		8	8	65,536	65,534
C			8	256	254



## כמה כתובות IP אפשריות קיימות בעולם?

כעת, כשאנחנו מודעים להגבלות שהוטלו עלינו בגלל ה- Classes השונים, האם תוכלו לומר לי מהי המגבלה האמיתית של כמות כתובות ה- IP האפשריות בעולם? בהתחלת המאמר דיברנו על קצת פחות מארבע וחצי מיליארד כתובות, אבל זה היה כשחשבנו שיש לנו את כל 32 הביטים לשחק איתם.

אז מהו המספר האמיתי?

אם נחשב בזריזות, נראה שצריך להכפיל את כמות הרשתות של Class A בכמות ה- Host IDs האפשריים, כנ"ל לגבי B ו- C, ולחבר את שלושת התוצאות.

Class	כמה רשתות קיימות?	כמה כתובות יש בכל אחת מהרשתות?	סה"כ כמות הכתובות האפשריות
A	126	16777214	2,113,928,964
B	16384	65534	1,073,709,056
C	2097152	254	532,676,608
			3,720,314,628

כלומר אנחנו עומדים על **3,720,314,628** "בלבד" ולא על 4,294,967,296 כמו שחשבנו קודם... סיבה טובה להתחיל להשתמש ב- IPv6...

אגב, שימו לב שיש בסה"כ 2,113,928,964 כתובות ברשתות Class A, שהם בדיוק **50%** (או חצי) מסך כל כתובות ה- IP האפשריות בתיאוריה - 4,294,967,296;

ויש בסה"כ 1,073,709,056 כתובות ברשתות Class B, שהם בדיוק **25%** (או רבע) מסך כל כתובות ה- IP האפשריות בתיאוריה;

ויש בסה"כ 532,676,608 כתובות ברשתות Class C, שהם בדיוק **12.5%** (או שמינית) מסך כל כתובות ה- IP האפשריות בתיאוריה.

כלומר, בסה"כ יש לנו יכולת לנצל רק **87.5%** מסך כתובות ה- IP האפשריות מתוך כתובת בת 32 ביט.

### תרגיל:

זהה את ה- Class של הכתובות הבאות. חפש תקלות ובעיות פוטנציאליות:

377.123.28.167 .14	210.23.67.102 .1
191.249.222.234 .15	66.23.148.0 .2
619.23.12.25 .16	158.23.251.33 .3
188.67.76.235 .17	144.23.117.254 .4
134.255.123.22 .18	192.254.23.123 .5
143.52.213.212 .19	144.207.78.1 .6
257.22.45.219 .20	63.125.23.211 .7
117.117.117.117 .21	192.25.128.36 .8
193.23.255.77 .22	215.78.0.0 .9
199.23.255.7 .23	134.223.156.89 .10
145.2.229.252 .24	127.0.0.1 .11
242.23.177.8 .25	224.23.108.23 .12
	223.78.27.144 .13

## אבחנה בין ה- Classes השונים

איך יודעים אם 2 מחשבים נמצאים על אותה רשת או לא כשהנתון היחיד הוא כתובות ה- IP שלהם? בעבודה עם Classfull IP Addressing שתואר לעיל התשובה היא פשוטה:

במקרה של **Class A** נקח לדוגמה כתובת כמו **17.0.0.0** – זו כתובת הרשת או ה- Network ID. זוכרים את הדוגמה של שם הרחוב ומספר הבית? אז זהו שם הרחוב. לשם של רחוב אין תוספות אחריו. כלומר אי אפשר

לקרוא לרחוב בשם "הרצל 134". זה "הרצל" וזהו, בלי תוספות אחריו. זה כמו לומר שאדם שמספר הטלפון שלו הוא 04-6564565 יאמר שאזור החיוג שלו הוא 04-0000000. כך גם ה- Network ID משתמש בתצורה של "איפוס" האוקטטות של ה- Host ID.

איזו כתובת מחשב או Host ID יכולה להיות על הרשת הזו? כל כתובת שמתחילה בטווח שבין 17.0.0.1 ועד 17.255.255.254, כלומר בסה"כ כמעט 17 מיליון כתובות אפשריות לאותה רשת. מחשב עם הכתובת **17.132.12.56** נמצא על אותה רשת כמו המחשב **17.56.83.97**. למה? כי כתובת הרשת של שניהם זהה. כלומר בית מספר 98 נמצא ברחוב הרצל. בית אחר שמספרו 184 נמצא גם הוא ברחוב הרצל, ולכן שני הבתים נמצאים באותו רחוב.

במקרה של **Class B** נקח לדוגמה כתובת כמו **131.117.0.0** – זהו ה- Network ID. איזו כתובת מחשב או Host ID יכולה להיות על הרשת הזו? כל כתובת שמתחילה בטווח שבין 131.117.0.1 ועד 131.117.255.254, כלומר קצת יותר מ-16 אלף מחשבים לאותה רשת. מחשב עם הכתובת **131.117.4.9** נמצא על אותה רשת כמו המחשב **131.117.35.72**. למה? כי כתובת הרשת של שניהם זהה.

במקרה של **Class C** נקח לדוגמה כתובת כמו **192.117.1.0** – זהו ה- Network ID. איזו כתובת מחשב או Host ID יכולה להיות על הרשת הזו? כל כתובת שמתחילה בטווח שבין 192.117.1.1 ועד 192.117.1.254, כלומר 254 מחשבים לאותה רשת. מחשב עם הכתובת **192.117.1.69** נמצא על אותה רשת כמו המחשב **192.117.1.17**. למה? כי כתובת הרשת של שניהם זהה.

### תרגיל:

סמן בקו את ה- Network ID וה- Host ID בכל אחת מהכתובות. חפש בעיות פוטנציאליות (לא בכולן יש בעיות):

245.12.33.102	.26	1.102.45.177	.1
123.123.123.123	.27	196.22.177.13	.2
199.23.107.255	.28	133.156.55.102	.3
199.23.107.0	.29	221.252.77.10	.4
156.266.12.103	.30	123.12.45.77	.5
99.0.0.12	.31	126.252.77.103	.6
153.0.0.0	.32	13.1.255.102	.7
153.0.0.255	.33	171.242.177.109	.8
191.23.255.255	.34	193.156.155.192	.9
33.255.255.0	.35	21.52.177.188	.10
12.0.0.0	.36	77.77.45.77	.11
12.255.255.255	.37	191.252.77.13	.12
12.0.0.255	.38	15.155.231.91	.13
127.0.0.1	.39	221.252.117.254	.14
127.23.109.122	.40	203.10.233.1	.15
0.23.12.122	.41	191.2.227.19	.16
192.12.255.102	.42	23.156.1.92	.17
191.105.0.0	.43	121.2.199.88	.18
203.123.45.255	.44	202.27.189.177	.19
254.0.23.198	.45	177.222.177.28	.20
224.56.204.112	.46	198.215.67.233	.21
223.255.255.254	.47	128.252.17.24	.22
126.0.0.1	.48	99.19.292.12	.23
177.45.123.255	.49	159.255.17.218	.24
191.255.217.227	.50	155.25.169.133	.25

## חלק שלישי

בחלק זה נתחיל להסביר כיצד פועל ה- Subnet Mask, פעולת ה- AND, הסבר קצר על סגמנטים, הסבר ארוך על פעולת ה- Subnetting, Subnetting של אוקטטה שלמה ושל חלק מאוקטטה, וחלוקה של טווחי רשתות.

### **איך פועל ה- Subnet Mask?**

אנחנו יודעים שבמידה ויש לנו כתובת IP שבה האוקטטה הראשונה נמצאת בטווח מסויים, אפשר במבט אחד לקבוע חד משמעית לאיזה Class שייכת אותה כתובת. אבל שיוך הכתובת ל- Class מסויים לא נותנת לנו שום דבר כשמדובר ברשת שמחולקת ל- Subnets, כפי שעוד מעט נראה.

למעשה, שיוך הכתובת ל- Class מסויים הוא שיוך קוסמטי בלבד. מה שבאמת קובע הוא כתובת הרשת, או ה- Network ID של אותה כתובת. אני צריך לקבוע באותה כתובת אילו מהאוקטטות יהיו שייכות לכתובת הרשת, ואילו מהאוקטטות יהיו שייכות ל- Host ID. כאן נכנס לתמונה ה- Subnet Mask.

ה- Subnet Mask מאפשר לי לקבוע באופן שרירותי אילו מהאוקטטות יהיו שייכות ל- Network ID, וכל היתר, מן הסתם, יהא שייך ל- Host ID (הסבר אודות המושג תמצאו בעמוד 7 במאמר).

מכיוון שאנחנו כבר יודעים ש- Class A משתמש רק באוקטטה הראשונה לכתובת הרשת, ניתן לו Subnet Mask קבוע של 255.0.0.0. אם אני יודע ש- Class B משתמש ב- 2 האוקטטות הראשונות לכתובת הרשת, ניתן לו Subnet Mask קבוע של 255.255.0.0, וכו' לגבי Class C – 255.255.255.0.

	W	X	Y	Z	Default Subnet Mask
Class A	Network	Host	Host	Host	255.0.0.0
Class B	Network	Network	Host	Host	255.255.0.0
Class C	Network	Network	Network	Host	255.255.255.0

לאחר שהאדמיניסטרטור קובע את ה- Subnet Mask של הרשת ומיישם אותו על כל המחשבים באותה רשת, פרוטוקול ה- IP בוחן את כתובת ה- IP דרך המסיכה – ה- Subnet Mask כדי להחליט לגבי כתובת הרשת שלו. המילה Mask (מסיכה) מקבל את המשמעות של "עדשה" (כמו במשקפיים), כיוון שהתוכנה מתבוננת בכתובת ה- IP דרך העדשות של ה- Subnet Mask כדי לקבוע מה היא כתובת הרשת שלה.

ל- Subnet Mask יש פורמט קבוע. הוא יכול להיות משהו כמו 255.0.0.0, או משהו כמו 255.255.255.0. הפורמט הזה קובע למחשב איזו אוקטטה שייכת לכתובת הרשת באמצעות פעולה מתמטית שנקראת AND. שימו לב, בשלב זה אנחנו מתייחסים לכל יחידה בתוך ה- Subnet Mask בפורמט העשרוני שלה (כלומר למשל 255), אבל מיד בהמשך המאמר תראו שבעצם אנחנו צריכים להסתכל על כל יחידה דווקא בצורה הבינארית שלה (כלומר 11111111 ולא דווקא 255).

### **פעולת ה- AND**

כיצד "יודע" המחשב לחלץ מתוך סבך כתובות ה- IP את רכיב ה- Network ID? כיצד הוא יודע לנחש איזו אוקטטה משמשת ל- Network ID, ואיזו אוקטטה משמשת ל- Host ID? כיצד "יודע" המחשב האם היעד אליו אמור המידע להגיע נמצא יחד איתו באותה רשת, או שמא הוא נמצא ברשת אחרת, מרוחקת? הוא עושה זאת באמצעות פעולה מתמטית הנקראת AND.

פעולת ה- AND היא בעצם פעולת כפל פשוטה. להזכירכם (לכל המוגבלים מתמטית...) בפעולת כפל רגילה, כפולה של כל מספר ב- 0 = תמיד 0, ומן הסתם, כפולה של כל מספר ב- 1 = המספר עצמו. בחישובי AND של כתובות IP התהליך הוא פשוט מאוד, כיוון שיש לנו "רק" 2 ספרות אפשריות – 0 או 1. המחשב לוקח את כתובת ה- IP הנתונה (הוא הרי רואה אותה בפורמט בינארי) וכותב מתחתיה את ה- Subnet Mask שהגדרנו לו.

הטבלה הבאה תעזור להמחיש זאת ביתר קלות: נקח מספר בינארי (סתם מספר בשלב הזה – אני בחרתי ב- 11101110) ונבצע עליו את פעולת ה- AND. השורה העליונה בטבלה היא המספר עצמו. השורה האמצעית

ממחישה את פעולת הכפל (\*), והשורה התחתונה היא ה- Subnet Mask (בשלב הזה מדובר רק באוקטטה אחת, להדגמה):

1	1	1	0	1	1	1	0
*	*	*	*	*	*	*	*
1	1	1	1	1	1	1	1

פעולת ה- AND היא כאמור פעולת כפל. אז ניקח ביט אחד ונכפיל אותו בביט שמתחתיו, ובשורה שלמטה נכתוב את התוצאה:

1	1	1	0	1	1	1	0
---	---	---	---	---	---	---	---

כאשר כופלים ערך בינארי כלשהו (השורה העליונה בטבלה) בערך בינארי אחר, שכולו 1 (או בעשרוני – 255), תמיד תתקבל התוצאה הזוהי לשורה שמעליה. ואם נכפיל את הערך הבינארי (בשורה העליונה בטבלה) בערך בינארי אחר שכולו 0 (או בעשרוני – 0), אז תמיד תתקבל תוצאה הזוהי ל-0.

מטרתה של פעולת ה- AND היא, אם כך, לחלץ מתוך כתובת ה- IP המורכבת מ- 32 ביטים את אותם ביטים שמשמשים לטובת ה- Network ID, ולאפשר למחשב להחליט האם הוא ומחשב היעד נמצאים באותו Network ID. אם התשובה חיובית, המידע יועבר דרך הרשת המקומית אל היעד. במידה והתשובה שלילית או אז ישלח המחשב את המידע אל היעד דרך ה- Default Gateway (ראו הסבר על המונח בעמוד 9 במאמר זה). במידה ולא הוגדר כזה, התקשורת תכשל מראש.

### דוגמה:

בואו נראה איך יודע המחשב לחלץ Network ID מתוך כתובת IP נתונה:

בכתובת ב- Class A כמו **64.0.0.10**, אנחנו יודעים שרק האוקטטה הראשונה מייצגת את ה- Network ID. נהפוך 64.0.0.10 לבינארי ונכתוב את הכתובת בטבלה הבאה:

כלומר (כל הכתובת):

$$01000000.00000000.00000000.00001010 = \mathbf{64.0.0.10}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.00000000.00000000.00000000 = \mathbf{255.0.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$01000000.00000000.00000000.00000000 = \mathbf{64.0.0.0}$$

רואים שהתוצאה של האוקטטה הראשונה זהה לכתובת הרשת המקורית – **64.0.0.0**.

עכשיו, נניח שהמחשב בעל הכתובת הזו מנסה לעשות Ping למחשב אחר בעל כתובת **64.0.0.45**. לפני שהוא מבצע משלוח חומר כלשהו, המחשב שלי חייב לבצע החלטה: האם כתובת היעד נמצאת אצלו ברשת המקומית, או שמא הוא צריך לשלוח את המידע החוצה לרשת מרוחקת, דרך ה- Default Gateway? לכן הוא מיד מבצע פעולת AND של **כתובת היעד** עם ה- Subnet Mask המקורי שלו עצמו. נכתוב זאת שוב:

$$01000000.00000000.00000000.00101101 = \mathbf{64.0.0.45}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.00000000.00000000.00000000 = \mathbf{255.0.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$01000000.00000000.00000000.00000000 = \mathbf{64.0.0.0}$$

התוצאה זהה! כתובת הרשת המקורית של המחשב זהה לכתובת הרשת של היעד. מכאן מגיע המחשב ל"מסקנה" שהיעד נמצא ברשת זהה לזו של המחשב שלו עצמו, ולכן המחשב השולח יבצע את כל הפעולות הנדרשות כדי להגיע אליו (ARP וכו') ולא יעביר את התשדורת אל הנתב.

**דוגמה נוספת:**

נניח שאני מנסה לעשות Ping למחשב בעל הכתובת **64.32.67.88**. המחשב שלי מבצע פעולת AND עם כתובת היעד ומשווה אותה לתוצאת ה- AND של הכתובת המקורית שלו. אם התוצאה שווה, היעד נמצא יחד עם המחשב המקורי באותה רשת. אם לא – שלח אותו לכל הרוחות לכיוון ה- Default Gateway!

נשתמש בתוצאה שכבר חישובנו בדוגמה הקודמת, אבל קעת נבצע AND לכתובת **היעד**:

$$01000000.00100000.01000011.01011000 = \mathbf{64.32.67.88}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.00000000.00000000.00000000 = \mathbf{255.0.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$01000000.00000000.00000000.00000000 = \mathbf{64.0.0.0}$$

גם כאן, כמו בדוגמה הקודמת, יצא שכתובת הרשת המקורית של המחשב זהה לכתובת הרשת של היעד. היעד נמצא ברשת זהה לזו של המחשב המקורי, ולכן המחשב השולח יבצע את כל הפעולות הנדרשות כדי להגיע אליו ולא יעביר את התשדורת אל הנתב.

**דוגמה נוספת:**

מהמחשב שלי למחשב **65.32.67.88**.

$$01000001.00100000.01000011.01011000 = \mathbf{65.32.67.88}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.00000000.00000000.00000000 = \mathbf{255.0.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$01000001.00000000.00000000.00000000 = \mathbf{65.0.0.0}$$

במקרה זה התוצאה המצביעה על כך שמדובר בכתובת רשת אחרת. לכן, כשהמחשב המקורי ירצה לתקשר עם מחשב היעד, הוא יהיה חייב לשלוח את המידע אל ה- Default Gateway. אין למחשב שום ברירה במקרה הזה. ברגע שהוא מזהה שהכתובת המרוחקת נמצאת על רשת אחרת, הוא אוטומטית מפנה את התשדורת אל הנתב שאמור להציע נתיב אל אותה רשת מרוחקת, או במילים אחרות, ה- Default Gateway.

במידה ולמחשב השולח לא הוגדר Default Gateway ולא בוצעו שום שינויים בטבלת הניתוב שלו, המחשב השולח לא יהיה מסוגל לתקשר עם היעד פשוט בגלל שאין לו מושג לאן להפנות את התשדורת. הוא "מבין" שהוא צריך להוציא את התשדורת החוצה דרך הנתב, אבל אין לו מושג מיהו הנתב ומהי כתובתו, ולכן התשדורת נועדה לכישלון.

**הערה:** יש לציין שאת ההחלטה לגבי "לאן יש להפנות את המידע כדי שיגיע ליעדו" מחליט המחשב לא רק לפי ה- Default Gateway שהוגדר לו, אלא, בעצם, לפי טבלה מורכבת יותר הנקראת "טבלת ניתוב" (או Routing Table). לא זה המקום להיכנס לנושא, אבל אם תקלידו את הפקודה Route Print בשורת פקודה תוכלו לצפות בטבלה זו בעצמכם.

אגב, ניתן לבצע את פעולת ה- AND גם באמצעות המחשבון שלנו ע"י מעבר לתצוגה מדעית ושימוש בכפתור  $\Rightarrow$  AND הנמצא מצד ימין.

**תרגיל:**

כתוב את ה- Default Subnet Mask של הכתובות הבאות לפי שיוך ל- Classes. חפש גם בעיות פוטנציאליות. לבסוף, מצא זוגות של כתובות IP הנמצאות באותה רשת.

200.120.5.2	.16	210.23.67.102	.1
162.83.38.10	.17	66.23.148.0	.2
78.65.201.33	.18	158.23.251.33	.3
214.9.107.74	.19	144.23.117.254	.4
102.204.27.126	.20	192.254.23.123	.5
44.27.7.11	.21	214.9.107.1	.6
63.125.23.211	.22	63.125.23.211	.7
10.114.178.96	.23	192.25.128.36	.8
18.192.54.198	.24	121.12.254.98	.9
171.67.116.0	.25	134.223.156.89	.10
70.80.90.100	.26	127.0.0.1	.11
220.221.222.223	.27	224.23.108.23	.12
2.178.67.104	.28	223.78.27.144	.13
87.35.78.107	.29	220.221.223.167	.14
121.12.20.44	.30	191.249.222.234	.15

הערה: תוכל להתאמן בכתיבת ה- Subnet Mask גם על התרגילים הקודמים. ככל שתעשה את זה יותר פעמים כך תגדל רמת המיומנות שלך בזיהוי מידי של בעיות אפשריות.

### Classless Inter-Domain Routing – CIDR

לעיתים אנחנו עשויים לקבל מידע אודות כתובות IP וה- Subnet Masks שלהם בפורמט שהוא מעט שונה מהפורמט שבו דנו עד עתה. למשל:

IP Address: **172.16.45.2 /12**

מהו ה- **/12** הזה?

מדובר בשיטה חלופית לכתיבת ה- Subnet Mask. במקום לכתוב:

IP Address: **172.16.45.2**  
Subnet Mask: **255.240.0.0**

לוקחים את כל הביטים שהם 1 בתוך ה- Subnet Mask, כלומר בבינארי, משמאל לימין, את כל רצף הביטים שהם הסיפורה 1 (אחת) וסופרים אותם. את התוצאה כותבים לאחר הלכסן, במקרה זה **12**.

Subnet Mask: **11111111.11110000.00000000.00000000 = /12**

יש מספר תוכנות ושירותים מובנים שמשתמשים בפורמט הזה, בעיקר מוצרים צד-שלישי שמתעסקים בניתוב או בהגנה על הרשת, אבל גם במספר מרכיבים בתוך Windows 2000/2003 עצמו, כמו ב- RRAS, DHCP ואחרים.

קל לזכור שה- Classes השונים מחולקים ביניהם בצורה הבאה:

Class A: 255.0.0.0 = **/8**  
Class B: 255.255.0.0 = **/16**  
Class C: 255.255.255.0 = **/24**

וכל מה שנופל בין היחידות ה"שלמות" הללו משמעותו שיש לנו כאן מניפולציה של ה- Subnet Mask בדרך שאותה נלמד מיד.

#### תרגיל:

כתוב את ה- Default Subnet Mask של הכתובות מהתרגיל הקודם לפי CIDR.

## מהו סגמנט?

סגמנט הוא חלק של רשת. מקטע של רשת שמופרד ממקטע אחר ע"י נתבים (או Routers). אפשר לאפיין סגמנט כמקטע פיזי של רשת שבתוכו אפשר לקבל מידע ממחשב אחר באמצעות Broadcasts. כלומר, שני מחשבים שנמצאים בסגמנט אחד יכולים לחלוף ביניהם מידע באמצעות Broadcasts. להזכירכם, נתבים לא מעבירים Broadcasts, ולכן אם יש לנו רשת שמחולקת ל-2 חלקים ע"י נתב באמצע, יש לנו 2 סגמנטים. אם הרשת היתה מופרדת ע"י מכשירים אחרים כדוגמת Switch או Bridge אזי לא היו לנו 2 סגמנטים, כיוון שהמכשירים המוזכרים כאן מסוגלים לבצע העברה של Broadcasts מצד אחד לצד השני, ולכן כן נוכל לקבל מידע ממחשב אחר שנמצא מעבר ל-Bridge ע"י Broadcasts. יש לציין שישנם סוויצ'ים רבים המסוגלים לבצע גם סגמנטציה של רשת בשכבה השלישית, אבל אנחנו לא ניכנס לנושא הזה במאמר הספציפי שלנו.

חשוב מאוד לציין כי במידה ושני חלקי רשת מופרדים ביניהם באמצעות נתב, חובה עלינו להגדיר לכל אחד מהסגמנטים גם Network ID שונה. מדוע?

ובכן, כזכור לכם כשדיברנו על המשמעות של Network ID הסברנו ששני מחשבים שנמצאים באותה רשת חייבים להיות גם בעלי אותו Network ID. כאשר שני המחשבים הללו ירצו לתקשר זה עם זה הם יבצעו באמצעות פעולת AND בדיקה האם המחשב השולח והמחשב הנמען מחזיקים באותה כתובת רשת. במידה והתשובה היא חיובית, המחשב השולח מכין את המידע ומעביר אותו ישירות אל המחשב הנמען.

אבל במידה והתשובה היא שלילית, המחשב השולח כלל לא טורח לנסות את מזלו ברשת המקומית, אלא מעביר את המידע אל הנתב (ה- Default Gateway), ומבקש ממנו להעביר את המידע אל הרשת שבה נמצא הנמען.

קעת בואו נתאר לעצמנו מצב לא-תקין, שבו אדמיניסטרטור חסר נסיון הגדיר בטעות 2 סגמנטים שונים בעלי אותו Network ID, ושם ביניהם נתב. במקרה כזה, מחשב א' הנמצא בסגמנט A מעוניין לתקשר עם מחשב ב' הנמצא גם הוא באותו סגמנט. לפי בדיקת AND של עצמו אל מול היעד, מחשב א' מניח, ובצדק, שמחשב ב' נמצא באותה רשת, ולכן שולח אליו ישירות את המידע.

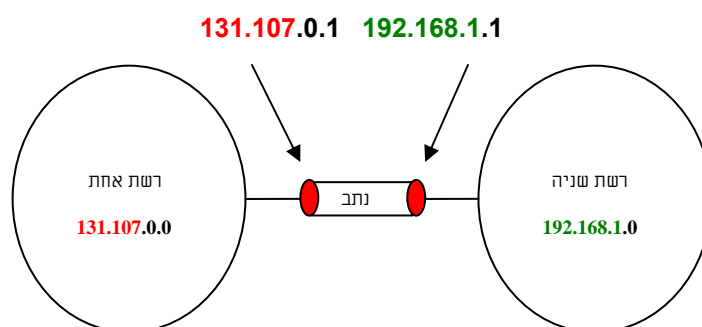
לעומת זאת, כשהוא מנסה לתקשר עם מחשב ג' הנמצא בסגמנט B, מכיוון שלשני המחשבים יש את אותו Network ID (כבר אמרנו – אדמיניסטרטור טירון?) הוא יגיע, והפעם בטעות, למסקנה הלא-נכונה ששניהם נמצאים באותה רשת, ולכן שוב, ינסה לשלוח את המידע ישירות אליו. אבל מכיוון שמחשב ג' לא נמצא מחובר לסגמנט A אלא נמצא מאחורי נתב, התקשורת לא מסוגלת להגיע אליו כלל.

לכן כאמור, חשוב מאוד לשמור על Network ID שונה לכל אחד מהסגמנטים, ולהגדיר Default Gateway לכל אחת מהרשתות.

## כמות ה- Network IDs הדרושה לרשתות

כאשר אני מתבונן בציור או בהגדרות רשת אני חייב לדעת לקרוא ולהבין כמה כתובות רשת אני צריך כדי לקיים את הרשת. זיכרו – כל סגמנט או מקטע של רשת חייב לקבל Network ID שונה משל האחרים. למשל, ברשת אחת בה יש 200 מחשבים, והיא אינה מפוצלת לתת-רשתות, אני חייב Network ID אחד, וזה יספיק לי לקיים את הרשת. איזה Class אני צריך? מספיק לי C כיוון שיש לי צורך רק ב-200 מחשבים, וידוע ש-Class C מספיק ל-254 מחשבים.

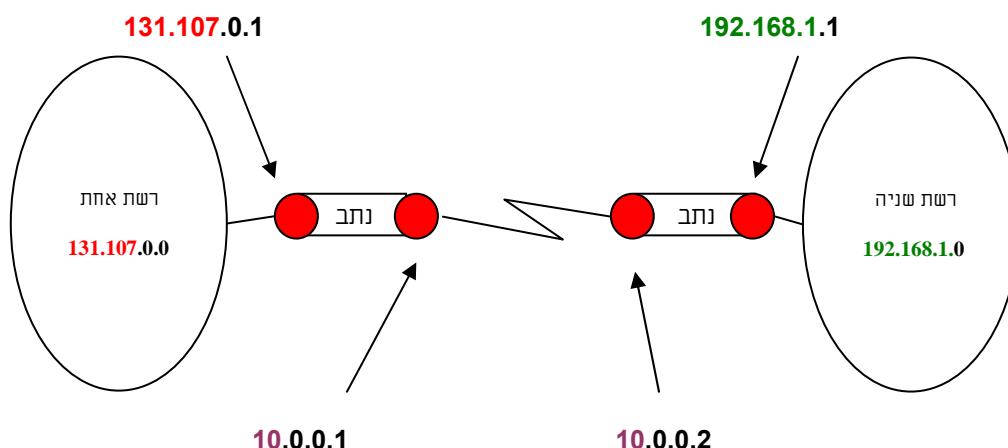
במצב שבו יש לי צורך בכמה רשתות, צריך לקרוא את הציור או הנתונים ולהחליט לגבי מספר כתובות הרשת הנחוצות. לדוגמה:



במקרה זה, הנתב "מפוצל" בין 2 רשתות, כשכל רגל (או פורט פיזי) של הנתב מקבלת כתובת IP שמתאימה לאותה רשת אליה היא מחוברת, והיא משמשת כ- Default Gateway של אותה רשת כלפי הרשת השניה.

כמה Network IDs אנחנו צריכים במקרה זה? אנחנו צריכים שני Network IDs - אחד עבור כל רשת.

מצד שני, בואו נתבונן בציר שמראה 2 רשתות מחוברות ביניהן באמצעות קו WAN:



שימו לב שבמקרה זה לכל רשת יש נתב שמחבר אותה אל ה-WAN, שבתורו מתחבר לנתב של הרשת השניה. גם כאן, לכל נתב יש 2 רגלים כמו בשרטוט הקודם, אבל כאן הרגל שניה מחוברת לרשת ה-WAN ולא ישירות אל הרשת השניה.

מכיוון שכל נתב מפוצל בין 2 רשתות, וכל רגל שלו חייבת כתובת IP, והיא איננה מחברת ישירות את הרשת השניה אלא ביניהם יש רשת נוספת, אני חייב 3 כתובות רשת או 3 Network IDs כדי להקים רשת כזו. ברור שלרוב, אנחנו כמנהלי רשת, לא מסוגלים לקבוע את הזהות של הרשת השלישית בעצמנו, ולרוב אנחנו מקבלים את ההגדרות מבזק או מהספק שממנו רכשנו את חיבור ה-WAN שלנו (למשל בחיבור Frame Relay).

נסו לשים לב לשרטוטים במבחן (וגם בחיים) כדי לא ליפול במלכודת כזו.

נקודה נוספת אליה יש לשים לב היא שלכל נתב חייבים להקצות כתובות IP כמספר הרגליים שלו. כך שבציר הקודם, פרט להקצאת הכתובות לכל רשת ל-HOSTS שלה, יש להקצות עוד 4 כתובות עבור הרגליים של 2 הנתבים.

### כמות ה- Host IDs הנחוצים ברשת

כאשר אני עומד לתכנן רשת אני צריך לדעת לכמה כתובות IP אשתמש ואצטרך, גם בהווה וגם בעתיד. אני צריך לדעת כמה מחשבים יהיו לי, כמה רכיבי רשת אחרים (כמו נתבים, מדפסות וכו') יהיו לי, ולהשאיר גם מקום לגידול עתידי. בעיקרון כלל הברזל אומר שלכל כרטיס רשת צריכה להיות כתובת IP, מכאן שלכל כרטיס רשת צריך להקצות Host ID פוטנציאלי.

לכל מחשב בעיקרון יש כרטיס רשת אחד (יש כאלה עם יותר), ולכן לכל מחשב יש כתובת IP אחת. למדפסות רשת (כאלה המחוברות ישירות ל"קיר") יש כתובת IP משלהן, וכך גם לנתבים, אבל לפי האמור למעלה, לכל נתב יש יותר מכתובת IP אחת (אחת לכל רגל או כניסה בנתב).

יש גם לקחת בחשבון את הגידול העתידי ברשת. אין טעם לתכנן מבנה רשת שמתאים להיום אבל בעוד שנה יהיה חסר כל ערך.



## Subnetting

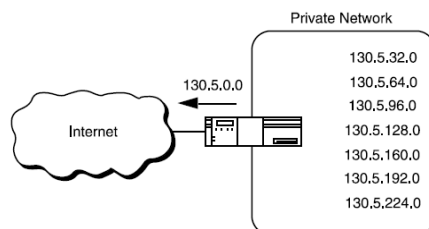
בשנת 1985 הוצע RFC 950 שהגדיר חלוקת משנה של כתובות רשת מסוג Class A, B ו-C. חלוקת המשנה יצרה מצב שבו רשת גדולה מסוג Class B למשל פוצלה להרבה תת-רשתות קטנות יותר. לפיצול זה קוראים Subnetting.

לינק ל-RFC 950: <http://www.faqs.org/rfcs/rfc950.html>

שיטת ה-Subnetting הגיעה בדיוק בזמן כדי לנסות ולעזור להתגבר על מספר בעיות שהתחילו לצוץ בחלקים מסוימים של רשת האינטרנט הצעירה:

- טבלאות הניתוב של נתבי ה-Backbone של האינטרנט הלכו ותפחו. הסטטיסטיקה מראה שמספר טבלאות הניתוב בנתבים אלה הולכות וגודלות בסידרה הנדסית. אם בשנת 1990 היו "רק" 2,190 שורות בטבלאות הניתוב, הרי שבשנת 1995 היו יותר מ-30,000 שורות, ואילו בסוף שנת 2000 עברנו את ה-100,000 שורות. גידול זה היה מעל ומעבר ליכולתם של הנתבים שהצטרפו לעבר כמות עצומה של מידע. ע"י שימוש ב-Subnetting, ה-IANA (ראשי תיבות של Internet Assigned Numbers Authority), הוא הארגון שאחראי על הקצאת מספרי פורטים, טווחי כתובות IP וכן הלאה) מקצה לאירגון Net ID אחד בלבד שמספיק לכל צרכיו, ומשאיר את זה למנהלי הרשת של אותו אירגון להחליט לגבי החלוקה הפנימית של אותו Net ID. מנהלי הרשת יגדירו נתב אחד חיצוני, זה שמקבל את כל התעבורה מן העולם אל תוך האירגון, ואותו נתב יחלק את המידע הנכנס לסגמנט (או תת-רשת) המתאים בתוך הרשת האירגונית, וזאת מבלי להעמיס במידע מיותר על נתבי ה-Backbone של האינטרנט.

בציור הבא רואים שהנתב שמחבר בין הרשת האירגונית לבין האינטרנט מחזיק עליו את רשימת תת-הרשתות הפנימיות, אבל רק עליו ולא על שום נתב אחר בעולם. מבחינת הנתבים האחרים בעולם – הם מכירים אך ורק את הרשת "הראשית" או המקורית – 130.5.0.0 – ולא את תת-הרשתות שמנהל הרשת הקים בצורה שרירותית. אגב, ציור זה ממחיש בד"כ את הגדרות הנתבים אצל ספקי האינטרנט, והם אלה שבד"כ מקצים ללקוחות הקטנים/בינוניים את מספרי תת-הרשתות מראש.



- מנהלי רשת שרצו להגדיל את מספר המחשבים ברשת שלהם (כתוצאה מהרחבת הרשת, מיזוג עם חברה אחרת וכו') נאלצו לבקש מ-IANA כתובת רשת נוספת כדי לתמוך בגידול. כך למשל מנהל רשת שטיפל ב-200 מחשבים החזיק עבורם כתובת רשת אחת מ-Class C. ברגע שהוא רצה להגדיל את הרשת שלו בעוד 100 מחשבים הוא נאלץ לעשות אחד משניים: או לבקש עוד רשת Class C ואז להחזיק ברשותו 254+254 כתובות IP עבור 300 תחנות; או לבקש רשת Class B ואז "לבזבז" מעל 65,000 כתובות IP ציבוריות רק בגלל שהוא היה צריך 300 כתובות. לו היה בוחר באופציה מספר 1, הוא היה מגיע למצב של 2 רשתות שונות (כבר אמרנו, מחשבים בעלי Net ID שונה נמצאים ברשתות שונות), וזה לא המצב שהוא היה מעוניין בו. לו היה בוחר באופציה מספר 2 הוא היה מבזבז כאמור כמות גדולה מאוד של כתובות IP לגיטימיות.

- מנהלי רשת שרצו לפצל את הרשת הקיימת שלהם או לפתוח מספר סניפים המרוחקים זה מזה גיאוגרפית נאלצו לבקש Network ID נפרד עבור כל סניף, מה שכמובן יצר ביזבז משווע של כתובות IP ציבוריות. הגיוני שאותו ארגון ישאף לחלק את הרשת הענקית שלו לכמה סגמנטים או רשתות קטנות יותר המחוברות ביניהן באמצעות נתבים. אבל אם אותו מנהל רשת ימשיך לעבוד בתצורה של Net ID אחד עבור כל המחשבים ברשת תהיה לו בעיה של ניתוב, כיוון שמחשב א' לא "יבין" שכדי לתקשר עם מחשב ב' הוא צריך לצאת ל-Default Gateway, שהרי ה-Net IDs של שניהם זהים. כאשר ארגון גדול מקבל רק כתובת רשת אחת (Network ID) מ-IANA, אז לאותו ארגון יש בעיה. תארו לעצמכם שכל ארגון שיש לו 5 סניפים עם 300 מחשבים בכל סניף, יבקש וגם יקבל 5 כתובות רשת נפרדות של Class B! מאגר הכתובות הפנוי בעולם היה נגמר תוך בערך שבוע. גם אם נסתכל על תסריט יותר נפוץ של ארגון שחייב 10 רשתות של 20 מחשבים בכל רשת – גם כאן היה יכול

להתרחש בזבז משווע של 10 רשתות Class C שלמות, עם מעל 2500 כתובות לגיטימיות, רק כדי לאפשר קיום של כ- 200 מחשבים ב- 10 רשתות נפרדות...

פרט ליתרונות שצוינו בשלושת הסעיפים הקודמים, לחלוקת רשת גדולה לתת-רשתות יש מספר יתרונות:

- הקטנה משמעותית של עומס הרשת: כולנו מעדיפים פחות תעבורה, בין אם מדובר בפקק תנועה ובין אם מדובר בעומס על ספק האינטרנט שלנו. אילמלא הרשת הייתה מחולקת באמצעות נתבים, הרי שכולנו היינו מתערבבים ברשתות של האחרים, והעומס היה הופך להיות בלתי נסבל. באמצעות הנתבים רוב התעבורה תישאר ברשת המקומית ולא תפריע לרשתות השכנות. רק ה-Packets שמיועדים לרשתות אחרות יורשו לצאת החוצה דרך הנתבים, ואילו תשדורות מקומיות כמו Broadcasts יישארו בתוך הרשת המקומית – המקום הטבעי שלהם.
- שיפור הביצועים של הרשת: הוא בעצם המשמעות של הקטנת העומס ברשת.
- ניהול פשוט יותר: קל יותר לבודד בעיות ברשת כאשר הרשתות הן גופים קטנים יחסית. כך אפשר למשל לנתק חיבור של נתב ולראות האם הבעיה ממשיכה וכו'.
- חיבור זול יותר של רשתות: כאשר אני מחלק רשת גדולה אחת לכמה תת-רשתות קטנות יותר אני יכול לבחור באמצעות איזה מדיום אני מחבר ביניהם. בנוסף, החיבור ביניהם (WAN) לא צריך לפעול כל הזמן, אלא רק כאשר יש לו דרישה.
- חיבור של כמה טופולוגיות רשת: כאשר אני משלב למשל Ethernet עם Token Ring אני צריך לעשות זאת באמצעות נתבים, והרי כדי להשתמש בנתב אני צריך בעצם לבנות 2 רשתות נפרדות, אחרת המחשבים הנמצאים משני עברי הנתב לא ידעו שהם צריכים להפנות את התקשורת לנתב ולא להמשיך לחפש את היעד ברשת המקומית.

**הערה חשובה:** ישנם שני סוגי סגמנטציה - סגמנטציה פיזית של רשת (כאמור ע"י הפרדה באמצעות נתבים) וסגמנטציה לוגית. בסגמנטציה לוגית אנחנו לוקחים מחשבים שנמצאים על אותו סגמנט פיזי (כלומר מחוברים לאותה רכזת) אבל מקצים להם כתובות IP השייכות ל-Network IDs שונים, וע"י כך גורמים להם "לחשוב" שהם נמצאים בסגמנטים שונים. מיותר אולי לציין, אבל מחשבים אלה לא יוכלו לעולם לתקשר זה עם זה כי בעצם אין ביניהם כל נתב.

## דוגמה 1

לחברה יש 100 סניפים, כל אחד מהם עם 10 מחשבים. מנהל הרשת של החברה יכול לבחור האם לקנות כתובת רשת אחת כללית, ולבנות רשת אחת גדולה שתקיף את כל 100 הסניפים שלה, או שמא לקנות כמה כתובות של רשתות ולפצל את הרשת שלה לכמה רשתות קטנות יותר. לפעמים לא מדובר ברכישה של כתובת רשת אלא בהקצאה מלמעלה, מחברת האם למשל. לקוח שלי מנהל רשת מקומית של חברה בינלאומית. מנהל הרשת שיושבים בלונדון קבעו כתובת של תת-רשת לכל אחת מחברות הבת או מהסניפים השונים, כך שנוצר מצב שבו הלקוח שלי נאלץ להסתדר עם תכתיבים מלמעלה.

בכל מקרה, אם מדובר ברשת אחת גדולה – איזה Class צריכה החברה לקנות? Class B כמובן, כיוון שמדובר ב- 100 סניפים ובכל אחד כ- 10 מחשבים, כלומר 1,000 מחשבים. אבל Class B מספיק להרבה יותר מ- 1,000 מחשבים! חזרו לטבלה בתחילת הפרק וראה שמדובר בכ- 65,000 מחשבים – מעל ומעבר למה שהחברה צריכה במציאות.

ואם מדובר בכמה רשתות קטנות יותר – איזה Class צריכה החברה לקנות? Class C כמובן, אבל נוצר מצב שהחברה תצטרך לקנות 100 רשתות שלמות של Class C, ולבזבז משהו כמו 24,000 כתובות IP חוקיות למהדרין, רק כדי שיהיו לה 100 סניפים עם סה"כ 1,000 מחשבים בכולם.

בכל מקרה, לא משנה איך תסתכלו על זה, מדובר פה בביזבז משווע של כתובות IP ציבוריות.

הפתרון הוא ב- Subnetting. החברה תרכוש מספק האינטרנט שלה רק קטע אחד של Class B, קטע שמספיק ל- 1,000 מחשבים, אבל מנהל הרשת של החברה יבצע סגמנטציה נוספת של אותם 1,000 מחשבים מבלי להודיע על כך לאף אחד ומבלי לעדכן שום טבלת ניתוב. ספק האינטרנט של הלקוח ידאג לעדכן בטבלאות הניתוב של נתבי ה- Backbone את כתובת הרשת שניתנה לאירגון, אבל ספק האינטרנט לא צריך להיות מודע לחלוקת המשנה הנוספת שעשה מנהל הרשת של אותו אירגון. מנהל הרשת יגדיר נתב אחד חיצוני, זה שמקבל את כל התעבורה מן העולם אל תוך האירגון, ואותו נתב יחלק את המידע הנכנס לסגמנט (או תת-רשת) המתאים בתוך הרשת האירגונית, וזאת מבלי להעמיס במידע מיותר על נתבי ה- Backbone של האינטרנט.

## דוגמה 2

לקוח אחר צריך להקים רשת גלובאלית הכוללת למעלה מ- 5,000 סניפים ובהם עד 1,000 מחשבים, בסה"כ משהו כמו 50,000 מחשבים. ברור שהלקוח יבקש כתובת רשת מ-Class B, אבל כאן מדובר רק ב- Net ID אחד, ואילו הלקוח זקוק ל- 5,000 NetIDs שונים – אחד עבור כל סניף.

חלוקה פיזית של הרשת למקטעים לא תעזור. למה? כיוון שאם נפריד את אותה רשת ל- 5,000 חלקים ונחבר אותם באמצעות נתבים, אם מחשב אחד ירצה לתקשר עם מחשב אחר במקטע אחר של הרשת, הוא צריך לדעת שהוא אמור לשלוח את המידע דרך Default Gateway כלשהו, אחרת הוא יחפש אותו מקומית אצלו במקטע, וכמובן שהוא לא ימצא. אני חייב למצוא דרך לומר לאותו מחשב "תראה, אתה שייך לרשת כזאת, והוא שייך לרשת אחרת, לכן אנא שלח את המידע שלך דרך הנתב החוצה ואל תנסה לחפש אותו אצלך ברשת".

הלקוח חייב למצוא דרך לגרום למחשב של שושנה הנמצא בסניף ת"א לדעת שכדי לתקשר עם השרת הנמצא בסניף המרכזי בבוסטון הוא יהיה חייב לצאת דרך ה- Default Gateway. אבל כאמור יש לנו רק Net ID אחד.

בואו ניזכר מה גורם למחשב להבין שהיעד נמצא ברשת מרוחקת (Remote Subnet) ולא ברשת המקורית שלו?

נכון, פעולת ה- AND.

זאת אומרת, עלי "לשקר" למחשב ולומר לו שמעכשיו אותו יעד, שעד לפני שניה התייחסת אליו כאילו הוא נמצא יחד איתך באותה רשת, עכשיו עליך לצאת החוצה דרך ה- Default Gateway כדי להגיע אליו.

פה נכנס לתמונה מושג ה- Subnetting.

## מעבר מ- 2 Level Classful Heirarchy ל- 3 Level Classless Heirarchy

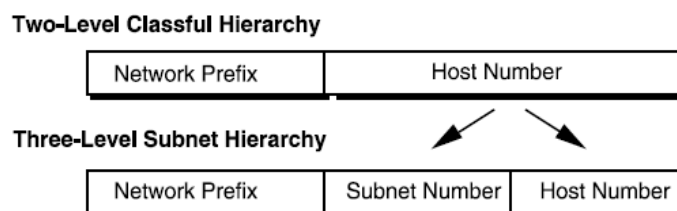
בפעולת ה- Subnetting אני לוקח Network ID מ-Class מסוים ומתנהג אליה כאילו היא מ-Class אחר, גבוה יותר. למשל, רשת של Class B שיש בה כ- 65,000 מחשבים תתנהג כאילו היא רשת המפוצלת להרבה תת-רשתות קטנות יותר, כשכל אחת מהן היא כאילו Class C, ויש בה "רק" 254 מחשבים.

שאלה: איך אפשר לגרום ל- Network ID מ-Class מסוים לחשוב שהיא מ-Class אחר?

נכון מאוד, ע"י שינוי פרמטר ה- Subnet Mask.

במקום לתת Subnet Mask של Class B, אתן לרשת Subnet Mask של Class C, וכך המחשבים "יחשבו" שהם נמצאים ב-Class C ולא ב-B.

למעשה, מה שעושים זה לעבור מתצורת Classfull IP Addressing שעבד עם 2 רמות – ה- Net ID וה- Host ID, לתצורת 3 רמות שמשתמשת, פרט ל- Net ID ול- Host ID גם ב- Subnet ID שבא ביניהם:



הסבר מקיף יגיע מיד.

**חשוב!** חשוב להבין ששינוי ה- Subnet Mask חייב להתבצע אצל כל המחשבים באותה רשת ולא רק אצל חלקם, שהרי אם חלק מהמחשבים לא יהיה מודע לשינוי שחל ב- Subnet Mks הוא עלול להמשיך ולחשוב שמחשב שנמצא בתת-רשת אחרת הוא בעצם "שכן" שלו ולכן הוא ימשיך לחפש אותו מקומית, למרות שכמובן הוא היה אמור לשלוח את המידע אל ה- Default Gateway.

## נקודות שצריך לקחת בחשבון לפני שמתחילים לבצע את פעולת ה- Subnetting:

לפני שמתחילים את החלוקה ל- Subnets צריך להביא בחשבון מספר מרכיבי מפתח שבלעדיהם אסור להתחיל בחלוקה ובתכנון:

- כמה כתובות רשת (Network ID) אני צריך? אחת עבור כל Subnet, ואחת עבור כל קשר WAN.
- כמה כתובות מחשב (Host ID) אני צריך על כל Subnet? אחת עבור כל כרטיס רשת או Interface ברשת, ואחת עבור כל רגל של נתב.
- מה יהיה הגידול הצפוי ברשת שלי? אם אני יודע שבתוך שנתיים אני צריך לפתוח עוד 5 סניפים, הרי שמראש אני אבנה רשת בחלוקה כזו שתאפשר לי גידול עתידי צפוי.

לאחר שהכנתי את הנתונים הללו צריך ליצור:

- Subnet Mask אחד עבור כל הרשת.
- Subnet ID ייחודי עבור כל תת-רשת או Subnet.
- טווח כתובות Hosts אפשרי לכל Subnet.

יצירת Subnets היא למעשה הפעולה של יצירת תת-רשתות קטנות יותר מתוך רשת-אם אחת גדולה. ארגון עם כתובת רשת אחת יכול שיהיו לו כמה Subnet Addresses עבור כל תת-רשת, אבל שיהיה לו גם Network ID אחד ראשי עבור כל הרשת כולה. כל Subnet הוא עדיין חלק מתוך הרשת הגדולה, אבל יש לו למעשה גם מאפיין או שם נוסף שמפריד אותו משאר תת-הרשתות.

אפשר להמחיש זאת באמצעות דוגמה פשוטה: נניח שיש לנו רחוב ארוך מאוד המתחיל בבית מספר 1 ונגמר בבית מספר 254. כאשר אדם המתגורר בבית מספר 10 ירצה לתקשר עם אדם הגר בבית מספר 230 באותו רחוב ברור לו שהיעד שלו נמצא באותו רחוב (הוא עשה AND, זוכרים?), ולכן יאלץ לכתת רגליו לכל אורך הרחוב עד שיגיע ליעדו. שושנה, שחייבת, אבל פשוט חייבת, לקרוא בקול גדול לכל הילדים שלה לעלות הביתה, תעשה את זה בהכרזה בקולי קולות, מה שכמובן יפריע לכל הילדים שמנסים בדיוק לנום את שנת היופי שלהם.

לעומת זאת, אם נחלק את אותו רחוב למקטעים של בלוקים של בתים, ולכל בלוק ניתן מספר מזהה, למשל בלוק מספר 1 שכולל בתים 1-30, בלוק מספר 3 שכולל בתים 65-94 וכו', עדיין לרחוב יהיה את אותו שם (זוכרים? - Network ID), אבל באותו שם יהיו כעת מספר רב של תת-שמות. "רחוב הרצל בלוק 13". אם אותו אדם מבית 10 ירצה להגיע לחברו מבית 20, הוא יצטרך ללכת ברגל כמה מטרים. אבל אם הוא ירצה להגיע לבית 200, הוא ישר ייגש לנתב (ה- Default Gateway שלו), והנתב יעביר אותו בנסיעה ישירה אל הבלוק שבו שוכן בית מספר 200. לאחר השינוי, צעקותיה הקולניות של שושנה אמנם יפריעו לשכניה לבלוק, אבל לא לכל הדיירים המתגוררים בכל הרחוב.

בדוגמה שלנו, הדוור שהולך כל בוקר ברגל מתיבת דואר אחת לשניה לא צריך לדעת על קיום כל שמות הדיירים בכל הבניינים, ולא צריך להטריד אותם בכל השמות, הכינויים והתחביבים של בני המשפחה. הוא לא צריך לזכור את כל טבלת הראוטינג, אלא רק את שם הרשת הראשית. הוא מכניס את הדואר לתוך התיבה לפי שם הרשת הראשית. אבל כשמגיע הדואר אל הנתב הראשי, הדואר הפנימי בין הבניינים ממזין לפי תיאור נוסף של מספר הבלוק באותו רחוב. בנוסף, בגלל שפעולת ה- Subnetting מאפשרת לכמה תת-רשתות להיות מאוחדות יחד, טבלת הראוטינג של אותן ארגון תהיה קטנה יותר ויעילה יותר. כמו-כן, כאמור גם הביצועים של הרשתות הקטנות ישתפרו מכיוון שהן לא מעבירות ביניהן Broadcasting וכו'.

אנחנו יוצרים Subnetting ע"י מתן Subnet ID לכל מחשב באותה רשת, ומשלבים כתובת זו בתוך הכתובת הכללית של אותו מחשב.

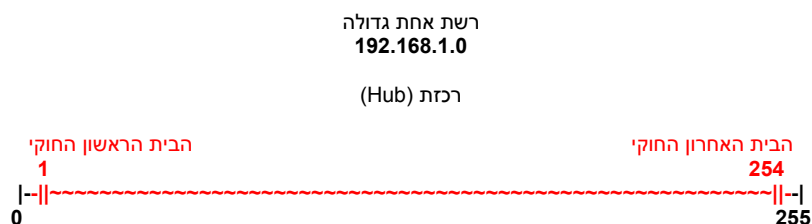
נו, אז איך יוצרים Subnet?

אמרנו כבר שאי אפשר לשנות את החלק של ה- Network ID בכתובת ה- IP, כיוון שכל המחשבים באותה רשת חייבים שתהיה להם אותה כתובת רשת כדי שיוכלו לתקשר זה עם זה. אבל אפשר בהחלט לשנות את ה- Subnet Mask המקורי שקיבלתי ולהפוך אותו לערך אחר, כזה שיגרום למחשב שלי להבין שכשהוא מנסה לפנות ליעד מסוים, עליו לצאת דרך ה- Default Gateway ולא להמשיך לחפש אותו מקומית ב- LAN שלו.

כלפי חוץ, רחוב הרצל הוא רחוב הרצל. ה- Network ID המקורי נשמר, וכן ה- Subnet Mask המקורי שהוקצה לי ע"י ה- IANA. אבל פנימית לכל המחשבים יש Subnet Mask שונה, בד"כ גדול יותר מאשר המקורי, מה שדורש להם להתבונן בצורה שונה על החלוקה בין ה- Net ID וה- Host ID.

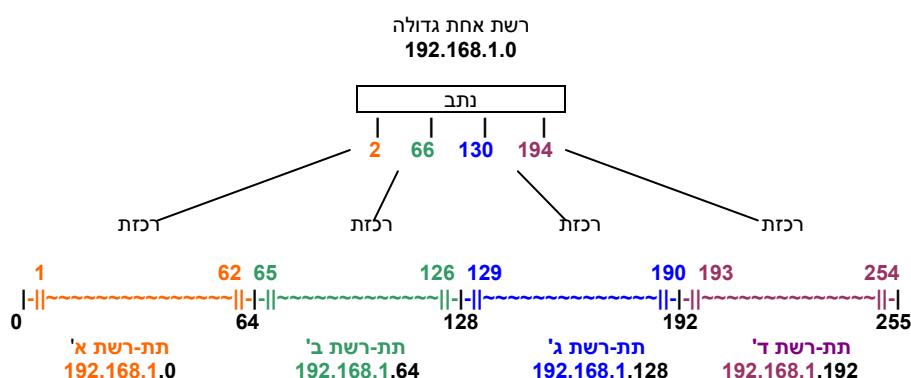
## הסבר גרפי:

## • לפני ביצוע Subnetting:



מדוע קבעתי שהבית הראשון החוקי הוא 1, והאחרון החוקי הוא 254? לא לשכוח את חוקי יסוד מספר 3 ו-4: אסור Host IDs בהם כל הביטים הם 0 או 1.

## • אחרי ביצוע Subnetting:



בדוגמה זו, במרכז הרשת יושב נתב בעל 4 כניסות (או Interfaces, או Ports – תלוי בכינוי) כשלכל Interface מוגדר כתובת IP והוא משמש כ- Default Gateway לאחת מ- 4 תת-הרשתות. כמובן שכתובות ה- IP של כל אחד מה- Default Gateways היא חלק מתוך ה- Network ID של אותה תת-רשת, אחרת לא יוכלו המחשבים הנמצאים באותה תת-רשת לתקשר החוצה עם שאר הרשתות.

**שימו לב:** הרגל או הפורט בנתב נקראת כאמור גם Interface וסביר להניח שתתקלו במושג זה די הרבה. נסו לא להתבלבל עם TCP/IP Ports שזה משהו אחר לגמרי. בנוסף, לרוב הנתבים יש שם לכל Interface, שם שבד"כ מסמל את סוג החיבור (אתרנט, סיריאלי וכו') ואת מספרו. כך בנתב בדוגמה הקודמת ניתן לקרוא לכל פורט בשם eth0, eth1, eth2, ו-eth3 בהתאם:



- לתת-רשת א', ה- interface המכונה **eth0** יישמש כ- Default Gateway עבור תת-רשת א', וכתובת ה- IP שלו תהיה **192.168.1.2**.
- לתת-רשת ב', ה- interface המכונה **eth1** יישמש כ- Default Gateway עבור תת-רשת ב', וכתובת ה- IP שלו תהיה **192.168.1.66**.
- לתת-רשת ג', ה- interface המכונה **eth2** יישמש כ- Default Gateway עבור תת-רשת ג', וכתובת ה- IP שלו תהיה **192.168.1.130**.
- לתת-רשת ד', ה- interface המכונה **eth3** יישמש כ- Default Gateway עבור תת-רשת ד', וכתובת ה- IP שלו תהיה **192.168.1.194**.

**שימו לב:** כתובות אלה הן דוגמה בלבד. אין הן מחייבות בפועל, והן משמשות רק לצורך ההמחשה. בד"כ נשתדל לבחור דווקא את הכתובות הראשונות או האחרונות, כמו 192.168.1.1, 192.168.1.65 וכו' בתור הכתובות שנקצו ל- Interfaces של הנתבים, פשוט מכיוון שזו המוסכמה הרווחת בשוק וככה יקל עלינו לזהות

אותם בסבך הכתובות. לכן, כשמגיעים להגדרת כתובות ה-IP שמשמשות עבור ה-Interfaces של הנתבים (וכאמור כ-Default Gateways עבור הסגמנטים), תמיד נשתדל לבחור את כתובת ה-IP הראשונה או האחרונה האפשרית באותו סגמנט.

**שימו לב:** מנקודת מבט של חיבור הרשת הפיזי, כמובן שככל הרשת שיוצא מכל Interface נכנס לתוך ארון תקשורת המכיל רכזות או Switches המחברים את שאר המחשבים השייכים לסגמנט הרלוונטי. כל המחשבים המחוברים לאותו ארון תקשורת רואים את כתובת ה-IP של ה-Interface הרלוונטי בנתב כשייך להם מבחינת ה-Net ID, ומן הסתם מסוגלים לתקשר איתו בצורה מקומית לאותו סגמנט.

כאשר מחשב מתת-רשת א' ירצה לתקשר עם מחשב אחר, למשל מחשב מתת-רשת ג', הוא יבצע פעולת AND בינו לבין היעד, ויבדוק האם היעד נמצא איתו באותה רשת, או שמא היעד הוא מרוחק (Remote). במידה והיעד מרוחק, המקור ישלח את המידע אל ה-Default Gateway המוגדר אצלו (במקרה זה הפורט הראשון בנתב), והנתב יבדוק לאיזה פורט הוא צריך לשלוח את התעבורה, במקרה זה לפורט השלישי בנתב (שוב, לזכור שמדובר בפורטים פיזיים שהם כניסות RJ45 בנתב עצמו, ולא בפורטים וירטואליים של TCP/IP כמו פורט 21 המשמש ל-FTP).

אגב, למה בתת-רשת א' הבית הראשון החוקי הוא 1, והאחרון החוקי הוא 62? מה קרה ל-63? לא לשכוח את הכל 0 והכל 1! אם נמיר לבינארי נראה שכתובת של 192.168.1.63 היא בעצם ביטוי של Host ID שבו כל הביטים הם 1, למרות שזה לא נראה כך על פני השטח. נלמד על כך בעוד מספר דקות.

נסו לזכור את הדוגמה הזאת. עוד נחזור אליה בהמשך.

## איך עובדת שיטת ה-Subnetting?

אחרי שהבנו למה Subnetting טוב ליהודים ולעולם בכלל בואו נראה כיצד עובדת השיטה.

אמרנו כבר שבתהליך ה-Subnetting אנחנו בונים תת-רשתות ע"י שינוי ה-Subnet Mask של המחשבים ברשת המקורית. אבל איך נשנה אותו? מה צריך להיות ה-Subnet Mask החדש? תיקף ניכנס להסברים מפורטים ואתן דוגמאות על גבי דוגמאות, אבל כדי לתאר את התהליך בקיצור, בואו ניקח דוגמה פשוטה של Subnet Mask של רשת מסוג Class B:

Network ID		Host ID	
130	57	Y	Z

ה-Subnet Mask המקורי היה:

255	255	0	0
-----	-----	---	---

או בבינארי (המרת בינארי רק את 2 האוקטטות האחרונות, את הראשונות הותרתי בתצורה עשרונית):

255	255	Host ID															
255	255	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

אנחנו רוצים "לגנוב" מה-Host ID ביטים רציפים מצידו השמאלי ביותר, ולתת אותם "מתנה" עבור ה-Subnet ID.

255	255	Host ID															
255	255	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ככל שניקח יותר ביטים מה-Host ID אנחנו מקבלים יותר ביטים שאפשר "לשחק" איתם, יותר אפשרויות, יותר Subnet IDs, כלומר יותר תת-רשתות. מצד שני, ככל שניקח יותר ביטים מה-Host ID אנחנו מקבלים פחות ביטים ב-Host ID, כלומר פחות אפשרויות, פחות Hosts לכל תת-רשת.

255	255	Host ID															
255	255	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

יותר ויותר ביטים הולכים מה-Host ID תמורת ה-Subnet ID, יותר תת-רשתות, אבל פחות ביטים נותרים ב-Host ID, ולכן פחות מחשבים בכל תת-רשת:

255	255	Host ID															
255	255	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

		Subnet ID								Host ID															
255	255	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

וכן הלאה. ניתן כמובן גם לגלוש עם ה"גניבה" של הביטים מה- Host ID גם לאוקטטה הרביעית, ואז אנחנו מגדילים עוד יותר את כמות תת-הרשתות, ומקטינים שוב את מספר ה- Hosts בכל תת-רשת.

255	255	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
255	255	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

למה אי-אפשר להתקדם עוד 2 ביטים ימינה? על כך בהמשך.

דוגמה זו טובה גם לגבי רשתות Class A, רק ששם יש לי 3 אוקטטות שאוכל "לגנוב" מהן ביטים, ולכן הרבה יותר אפשרויות לתת-רשתות. ניתן גם לעבוד עם רשתות Class C, רק ששם יש לי רק אוקטטה אחת "לגנוב" ממנה, מה שכמובן מגביל אותנו מאוד. על כך בהמשך.

**חישוב מספר הביטים שצריך "לגנוב":** בעיקרון, כשמתבקשים לחשב מספר Subnets תמיד נחפש את החזקה הקרובה ביותר של המספר 2 למספר ה- Subnets שנתבקשנו ליצור. למשל, אם אני צריך 8 תת-רשתות, אשאל את עצמי "2 בחזקת כמה שווה ל- 8?" התשובה היא כמובן 3, ולכן "אגנוב" 3 ביטים מה- Host ID לטובת ה- Subnet ID.

אם אני צריך 50 תת-רשתות אשאל את עצמי "2 בחזקת כמה שווה ל- 50?" אין תשובה מדויקת לשאלה, ולכן אחפש את החזקה הבאה של המספר 2, במקרה שלנו 64. "2 בחזקת כמה שווה ל- 64?" התשובה היא 6 כמובן, ולכן "אגנוב" 6 ביטים מה- Host ID לטובת ה- Subnet ID.

**חישוב מספר הביטים שצריך "לשמור":** אותו דבר לגבי ה- Host ID, רק שכאן אני רוצה לשמור לעצמי, בצד של ה- Host ID, לפחות את מספר הביטים שמצאתי בתשובה לשאלתי. בנוסף, מהמספר שמצאתי אני תמיד מפחית 2 (אסור Host ID שכולו 0 ו/או 1). למשל, אם אני צריך לפחות 100 מחשבים בכל תת-רשת אשאל את עצמי "2 בחזקת כמה שווה ל- 102?" התשובה היא שאין תשובה מדויקת לשאלה, ולכן אחפש את החזקה הבאה של המספר 2, במקרה שלנו 128. "2 בחזקת כמה שווה ל- 128?" התשובה היא 7 כמובן, ולכן לא משנה כמה ביטים "אגנוב" לטובת ה- Subnet ID, אדאג שבצד ימין, של ה- Host ID, ישארו לפחות 7 ביטים. במידה וארצה למשל רק 8 מחשבים בכל רשת, אשאל את עצמי "2 בחזקת כמה שווה ל- 10?" ולכן למרות שנוח לחשוב שמדובר ב- 3 ביטים בלבד, הרי שבמקרה זה אני חייב להתייחס לשאלה כאילו ביקשו ממני 16 תת-רשתות, ולכן התשובה היא 4 ולא 3.

מי שהבין את העיקרון יוכל עכשיו להבין את רוב השאלות שתיתקלו בהם במהלך המבחנים שלכם וגם במהלך עבודתכם כמנהלי רשת.

## דוגמה 1

נתבקשם ליצור מספר תת-רשתות כשבכל אחת מהן יש למשל עד 1,000 מחשבים, כל מה שאנחנו צריכים לחשב הוא כמה ביטים ב- Host ID דרושים על-מנת לציין 1,000 אפשרויות שונות.

3 ביטים? לא, כי 2 בחזקת 3 זה רק 8.

5 ביטים? לא, כי 2 בחזקת 5 זה רק 32.

9 ביטים? לא, כי 2 בחזקת 9 זה רק 512.

10 ביטים? כן! 2 בחזקת 10 זה 1,024, וזה בדיוק מה שאנחנו צריכים!

לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרנו רק עם 1,022 מחשבים בכל תת-רשת, אבל בכל מקרה זה מקיים את דרישות השאלה. אני מותיר לעצמי 10 ביטים כ-0 החל מצד ימין, וכל מה שבא החל מהביט ה-11 ואילך שמאלה יכול להפוך ל-1 ולהיות "מוענק" במתנה עבור ה- Network ID.

		Subnet ID						Host ID									
255	255	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

מה יהיה ה- Subnet Mask החדש של כל המחשבים ברשת?

נמיר את התוצאה שקיבלנו לעשרוני ונקבל:

255	255	252	0
-----	-----	-----	---

## דוגמה 2

אנחנו צריכים לבנות מערך של 128 תת-רשתות, כמה ביטים "נגנוב" מה- Host ID? 2 בחזקת כמה שווה ל-128? בדיוק 7! ולכן, "נגנוב" 7 ביטים מה- Host ID לטובת ה- Subnet ID.

		Subnet ID						Host ID									
255	255	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

מה יהיה ה- Subnet Mask החדש של כל המחשבים ברשת?

נמיר את התוצאה שקיבלנו לעשרוני ונקבל:

255	255	254	0
-----	-----	-----	---

## דוגמה 3

אנחנו צריכים לבנות מערך של 8 תת-רשתות, כמה ביטים "נגנוב" מה- Host ID? 2 בחזקת כמה שווה ל-8? בדיוק 3! ולכן, "נגנוב" 3 ביטים מה- Host ID לטובת ה- Subnet ID.

		Sub			Host ID												
255	255	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

מה יהיה ה- Subnet Mask החדש של כל המחשבים ברשת?

נמיר את התוצאה שקיבלנו לעשרוני ונקבל:

255	255	224	0
-----	-----	-----	---

## דוגמה 4

אנחנו צריכים לבנות מערך של 20 תת-רשתות, כמה ביטים "נגנוב" מה- Host ID? 2 בחזקת כמה שווה ל-20? אין מספר מדויק לסכום זה ולכן נבדוק את ה"קפיצה" הבאה בחזקות של 2, כלומר במקום לחשב לפי 20 שלא נותן לי תוצאה עגולה, נחשב לפי החזקה הבאה אחרי - 32 – ומתוך התוצאה ניקח רק 20 תת-רשתות.

לכן, 2 בחזקת כמה שווה ל-32? בדיוק 5! ולכן, "נגנוב" 5 ביטים מה- Host ID לטובת ה- Subnet ID.

		Subnet ID					Host ID										
255	255	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0

מה יהיה ה- Subnet Mask החדש של כל המחשבים ברשת?

נמיר את התוצאה שקיבלנו לעשרוני ונקבל:

255	255	248	0
-----	-----	-----	---



## דוגמה 5

אנחנו צריכים לתכנן רשת המכילה לפחות 500 תת-רשתות עם לא פחות מ-100 מחשבים לכל תת-רשת. האם זה אפשרי? הפעם נותנים לנו את 2 הערכים - גם כמות Hosts וגם כמות תת-רשתות. נבדוק 2 בחזקת כמה שווה ל-500? אין מספר מדויק לסכום זה ולכן נבדוק את ה"קפיצה" הבאה בחזקות של 2, כלומר במקום לחשב לפי 500 שלא נותן לי תוצאה עגולה, נחשב לפי החזקה הבאה אחריו - 512 - ומתוך התוצאה ניקח רק 500 תת-רשתות.

לכן, 2 בחזקת כמה שווה ל-512? בדיוק 9! ולכן, "נגנוב" 9 ביטים מה- Host ID לטובת ה- Subnet ID.

Subnet ID										Host ID							
255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

מה יהיה ה- Subnet Mask החדש של כל המחשבים ברשת?

נמיר את התוצאה שקיבלנו לעשרוני ונקבל:

255	255	255	128
-----	-----	-----	-----

אבל בואו ונבדוק האם בכל תת-רשת יש לפחות 100 מחשבים. כמה ביטים נותרו לנו לאחר ה"גניבה" ב- Host ID? 7 ביטים. 2 בחזקת 7 שווה לכמה? 2 בחזקת 7 שווה ל-128. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 126 מחשבים בכל תת-רשת. אפשר לומר שהתיכנון שהצענו אכן מאפשר ליישם את דרישות השאלה.

עכשיו, לטובת אלה שמתקשים או מעוניינים בהסבר מקיף יותר, נתחיל לעבור על כל הקומבינציות של Subnetting ברשת מסוג Class B. מי שהבין וקלט את הפטנט יוכל, אחרי דוגמה אחת או שתיים, להמשיך הלאה.

## Subnetting של אוקטטה שלמה:

בכתובת Class B רגילה, 2 האוקטטות הראשונות משמשות לכתובת הרשת, ו-2 האוקטטות האחרונות משמשות כ- Host ID. איזה Default Subnet Mask היה לכתובת כזו? ברור שמדובר בכתובת מ- Class B, ולכן ה- Subnet Mask שלה יהיה 255.255.0.0.

מה שנעשה עכשיו זה לקחת את האוקטטה השלישית, ולומר לכולם "זו תהיה כתובת תת-רשת (Subnet ID) שלכם".

איך עושים זאת? פשוט מאוד: במקום לתת Subnet Mask רגיל של 255.255.0.0, נגדיר עכשיו Subnet Mask חדש לכל הרשת ונקרא לו 255.255.255.0.

לכל המחשבים בסניף אחד ניתן כתובת כזו: 130.57.0.0, והטווח של כל רשת כזו יהיה מ-130.57.0.1 עד 130.57.0.254.

בסניף שני ניתן כתובת כזו: 130.57.1.0, והטווח יהיה מ-1 ועד 254.

בסניף שלישי ניתן כתובת כזו: 130.57.2.0, והטווח יהיה מ-1 ועד 254, וכו' וכו'.

כמה רשתות יהיו לי? 256 רשתות שונות, כשלכל אחת יש עד 254 מחשבים. כל זה במקום רשת אחת ענקית עם כ-65,000 מחשבים.

כדי להסביר בצורה יותר מוחשית, ניקח את המצב כמו שהיה, ונראה כיצד הוא השתנה לאחר היישום של ה- Subnetting:

לפני:

Network ID		Host ID	
130	57	Y	Z

ה- Subnet Mask המקורי היה:

255	255	0	0
-----	-----	---	---

יש לנו רק רשת אחת גדולה עם 2 בחזקת 16 אפשרויות ל- Host ID שונות (65534 כתובות שונות).

אחרי:

Network ID	Subnet ID	Host ID
130	57	Y
		Z

יש לנו רשת אחת גדולה שמחולקת ל- 2 בחזקת 8 תת-רשתות כשכל תת-רשת כזו יכולה להכיל 2 בחזקת 8 כתובות שונות (כמובן שיש להחסיר 2 בחישוב הסופי, כי אסור הכל 0 ואסור הכל 1).

רגע! למה התכוון המשורר באומרו "מחולקת ל- 2 בחזקת 8 תת-רשתות"? התבוננו בעמודה הכתומה, זו שעכשיו "נגנבה" מה- Host ID לטובת Subnet ID. כמה ביטים יש בה? 8 (כיוון שמדובר באוקטטה שלמה). כלומר 2 בחזקת 8 שווה 256 אפשרויות שונות.

איך עשינו את הטריק? פשוט מאוד. לקחנו את ה- Subnet Mask ושינינו אותו. כברירת מחדל ה- Subnet Mask המקורי היה **255.255.0.0**, וכעת הפכנו אותו ל- **255.255.255.0**, וכאילו "גנבנו" אוקטטה אחת מה- Host ID ונתנו אותה במתנה לחלק של ה- Network ID.

ה- Subnet Mask החדש הוא:

255	255	255	0
-----	-----	-----	---

ננסה להסביר את המושג בצורה הבינארית שלו: נסו להיזכר איך יודע המחשב האם כתובת מסויימת נמצאת על הרשת "שלו" או שמא היא נמצאת על רשת אחרת, ויש להפנות את התעבורה אליה דרך ה- Default Gateway. דיברנו על פעולת ה- AND. נחזור עליה כאן:

ניקח כתובת כלשהי, למשל **130.57.13.5**, ונראה האם היא נמצאת באותה רשת כמו **130.57.60.12**.

$$10000010.00111001.00001101.00000101 = \mathbf{130.57.13.5}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.11111111.00000000.00000000 = \mathbf{255.255.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$10000010.00111001.00000000.00000000 = \mathbf{130.57.0.0}$$

רואים שהתוצאה זהה לכתובת הרשת המקורית – **130.57.0.0**.

עכשיו, נניח שהמחשב בעל הכתובת הזו מנסה לעשות Ping למחשב אחר בעל כתובת **130.57.60.12**. לפני שהוא מבצע משלוח חומר כלשהו, המחשב שלי חייב לבצע החלטה: האם כתובת היעד נמצאת אצלי ברשת המקומית, או שמא אני צריך לשלוח את המידע החוצה לרשת המרוחקת דרך ה- Default Gateway? לכן הוא מיד מבצע פעולת AND של כתובת היעד עם ה- Subnet Mask המקורי שלי. נכתוב זאת שוב:

$$10000010.00111001.00111100.00001100 = \mathbf{130.57.60.12}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.11111111.00000000.00000000 = \mathbf{255.255.0.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$10000010.00111001.00000000.00000000 = \mathbf{130.57.0.0}$$

מה יצא? יצא שכתובת הרשת המקורית שלי זהה לכתובת הרשת של היעד. לכן מה המסקנה? היעד נמצא ברשת שלי, ואז המחשב שלי יבצע את כל הפעולות הנדרשות כדי להגיע אליו (ARP וכו'). כלומר ברשת מסוג זה, עם ה- Subnet Mask המקורי של **255.255.0.0**, שני המחשבים נמצאים באותה רשת (זה גם ברור לפי מבט חטוף בכתובת הרשת של שניהם – **130.57.0.0**).

עכשיו בואו נשתמש ב- Subnet Mask החדש – **255.255.255.0** ונראה כיצד זה משפיע על חלוקת הרשת.  
ניקח את אותן דוגמאות ממקודם, רק הפעם עם ה- Subnet Mask החדש.

ניקח את אותה כתובת - **130.57.13.5**, ונראה האם היא נמצאת באותה רשת כמו **130.57.60.12**.

$$10000010.00111001.00001101.00000101 = \mathbf{130.57.13.5}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.11111111.11111111.00000000 = \mathbf{255.255.255.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$01000000.00111001.00001101.00000000 = \mathbf{130.57.13.0}$$

רואים שהתוצאה זהה לכתובת הרשת המקורית – **130.57.13.0**. אבל - שימו לב: הפעם הרשת מקבלת שם חדש. לא רק 2 אוקטטות כמו קודם, אלא 3 אוקטטות.

עכשיו נכתוב את כתובת היעד:

$$10000010.00111001.00111100.00001100 = \mathbf{130.57.60.12}$$

מתחתיה נכתוב את ה- Subnet Mask בבינארי:

$$11111111.11111111.11111111.00000000 = \mathbf{255.255.255.0}$$

עכשיו נבצע פעולת AND או כפל בין שניהם, ביט למעלה כפול ביט למטה, ונכתוב את התוצאה מתחת:

$$10000010.00111001.00111100.00000000 = \mathbf{130.57.60.0}$$

מה יצא? יצא שכתובת הרשת השניה שונה לגמרי מכתובת הרשת הראשונה, לכן, מן הסתם, לא מדובר באותן רשתות. כלומר ע"י שינוי ה- Subnet Mask בצורה פשוטה ביותר הצלחנו לציין לשני המחשבים שהם לא נמצאים על אותה רשת כמו מקודם, למרות שלא נגענו בכתובת ה- IP עצמה. השגנו את מה שביקשנו. הפרדנו רשת אחת גדולה של Class B להרבה תת-רשתות קטנות. כמה תת-רשתות? חזור לתחילת הדוגמה וראה.

נעבור להמשך הדוגמאות, הפעם, חלוקה של חלק מהאוקטטה:

### **Subnetting של חלק מאוקטטה:**

בדוגמה הקודמת השתמשנו באוקטטה שלמה – או בייט שלם – לצורך כתובת ה- Subnet. כאמור זה משאיר לנו אוקטטה נוספת – האוקטטה הרביעית והאחרונה – לצורך קביעת ה- Host ID. ראינו שאוקטטה אחת יכולה לתת לנו 2 בחזקת 8 אפשרויות שונות, כלומר 256 אפשרויות. אך מכיוון שאסור הכל 0 ואסור הכל 1, נותרנו עם 254 אפשרויות ל- Host IDs. לכן, אם אנחנו צריכים רשת שיש לה יותר מ- 254 מחשבים, אנחנו נתקלים בבעיה. כדי לפתור את הבעיה הזו, אנחנו צריכים איכשהו "להשיג" עוד ביטים עבור ה- Host ID. מאיפה ניקח או "נגנוב" ביטים? הרי יש לנו 2 אוקטטות ל- Network ID, אוקטטה אחת ל- Subnet ID, והאוקטטה האחרונה ל- Host ID. מאיפה ניקח עוד ביטים?

התשובה היא לקצר את ה- Subnet ID, כלומר במקום מלכתחילה "לגנוב" אוקטטה שלמה מה- Host ID לצורך ה- Subnet ID, ניקח קצת פחות מאוקטטה שלמה, כלומר לא 8 ביטים, אלא פחות, כמה שנצטרך. תופעת הלוואי של השינוי הזה היא שאמנם יהיו לנו יותר ביטים עבור ה- Host ID, אבל אנחנו מקבלים פחות ביטים ל- Subnet ID, ומכאן נובע שיש לנו יותר מחשבים אבל פחות תת-רשתות.

נדגים על הכתובת הקודמת. (אני מזכיר מה היה לנו, ואח"כ נראה כיצד זה ישתנה לאחר שניקח ביטים מה- Subnet ID):

**לפני** (מתחת - ה- Subnet Mask המקורי):

Network ID		Host ID	
130	57	Y	Z
255	255	0	0

יש לנו רק רשת אחת גדולה עם 2 בחזקת 16 אפשרויות ל- Host ID שונות (65,534 כתובות שונות).

**אחרי** (מתחת - ה- Subnet Mask החדש):

Network ID		Subnet ID	Host ID
130	57	Y	Z
255	255	255	0

את המשבצת הצהובה, זו ששייכת ל- Subnet ID, אני מגדיל כאן וכותב את התוכן שלה בבינארי, ומתחתיה אני כותב את ה- Subnet Mask בבינארי (255.255.255.0):

		Subnet ID								Host ID															
255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

אם "נגנוב" את הביט הכי ימני מתוך ה- Subnet ID ונתן אותו במתנה ל- Host ID, מה זה יתן לנו?

		Subnet ID								Host ID															
255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 7 ביטים ל- Subnet ID, ו- 9 ביטים ל- Host ID.

ניקח את המספר הבינארי 11111110 ונהפוך לעשרוני. כמה יצא? 254. לכן, ה- Subnet Mask החדש יהיה 255.255.254.0

255	255	254	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 7 שווה 128, ואכן יש לנו 128 תת-רשתות אפשריות.

בואו נבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 9 ביטים לשחק איתם. לכן, 2 בחזקת 9 שווה 512. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 510 מחשבים בכל תת-רשת.

נניח לרגע שגם 510 מחשבים ברשת לא מספיקים לי. אני רוצה יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

		Subnet ID								Host ID															
255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 6 ביטים ל- Subnet ID, ו- 10 ביטים ל- Host ID.

ניקח את המספר הבינארי 11111100 ונהפוך לעשרוני. כמה יצא? 252. לכן, ה- Subnet Mask החדש יהיה 255.255.252.0

255	255	252	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 6 שווה 64, ואכן יש לנו 64 תת-רשתות אפשריות.

בואו נבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 10 ביטים לשחק איתם. לכן, 2 בחזקת 10 שווה 1,024. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 1,022 מחשבים בכל תת-רשת.

שימו לב איך אנחנו מתחילים לצמצם את מספר התת-רשתות, אבל יחד עם זאת להעלות את מספר המחשבים בכל תת-רשת כזאת. נמשיך.

נניח לרגע שגם 1,022 מחשבים ברשת לא מספיקים לי. אני רוצה יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

		Subnet ID					Host ID														
255	255	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 5 ביטים ל- Subnet ID, ו- 11 ביטים ל- Host ID.

ניקח את המספר הבינארי 11111000 ונהפוך לעשרוני. כמה יצא? 248 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.248.0

255	255	248	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 5 שווה 32, ואכן יש לנו 32 תת-רשתות אפשריות.

מעניין, בואו נבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 11 ביטים לשחק איתם. לכן, 2 בחזקת 11 שווה 2,048. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרנו רק עם 2,046 מחשבים בכל תת-רשת.

נניח לרגע שגם 2,046 מחשבים ברשת לא מספיקים לי. אני רוצה יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

		Subnet ID					Host ID														
255	255	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 4 ביטים ל- Subnet ID, ו- 12 ביטים ל- Host ID.

ניקח את המספר הבינארי 11110000 ונהפוך לעשרוני. כמה יצא? 240 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.240.0

255	255	240	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 4 שווה 16, ואכן יש לנו 16 תת-רשתות אפשריות.

בואו נבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 12 ביטים לשחק איתם. לכן, 2 בחזקת 12 שווה 4,096. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרנו רק עם 4,094 מחשבים בכל תת-רשת.

נניח לרגע שגם 4,096 מחשבים ברשת לא מספיקים לי. אני רוצה יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

		Subnet ID					Host ID														
255	255	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 3 ביטים ל- Subnet ID, ו- 13 ביטים ל- Host ID.

ניקח את המספר הבינארי 11100000 ונהפוך לעשרוני. כמה יצא? 224 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.224.0

255	255	224	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 3 שווה 8, ואכן יש לנו 8 תת-רשתות אפשריות.

כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 13 ביטים לשחק איתם. לכן, 2 בחזקת 13 שווה 8,192. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרנו רק עם 8,190 מחשבים בכל תת-רשת אפשרית.

תראו מה קורה: עכשיו יש לי רק 8 תת-רשתות, אבל 8,190 מחשבים בכל תת-רשת. נמשיך עוד:

נניח לרגע שגם 8,190 מחשבים ברשת לא מספיקים לי. אני רוצה יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

Sub		Host ID													
255	255	1	1	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 2 ביטים ל- Subnet ID, ו- 14 ביטים ל- Host ID.

ניקח את המספר הבינארי 11000000 ונהפוך לעשרוני. כמה יצא? 192 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.192.0

255	255	192	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 2 שווה 4, ואכן יש לנו 4 תת-רשתות אפשריות.

כמה כתובות אפשריות של Host ID יש לנו? עכשיו יש 14 ביטים לשחק איתם. לכן, 2 בחזקת 14 שווה 16,384. 16,384 פחות 2 שווה 16,382, ואכן יש לנו 16,382 Host ID שונים בכל תת-רשת אפשרית. הגענו למצב שבו הרשת הגדולה מפוצלת ל- 4 תת-רשתות, כשכל אחת יכולה להכיל עד 16,382 מחשבים שונים.

האם אני יכול "לגנוב" עוד ביט אחד לטובת ה- Host ID? בואו נראה: אם ניקח עוד אחד נישאר עם ביט אחד עבור ה- Subnet ID, וזה אומר 2 בחזקת 1, כלומר 2.

Sub		Host ID													
255	255	1	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק ביט אחד ל- Subnet ID, ו- 15 ביטים ל- Host ID.

ניקח את המספר הבינארי 10000000 ונהפוך לעשרוני. כמה יצא? 128 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.128.0

255	255	128	0
-----	-----	-----	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 1 שווה 2, ואכן יש לנו 2 תת-רשתות אפשריות.

אגב, במחשבים המריצים מערכת הפעלה של Windows NT 4.0 וקודמיה, אי אפשר היה להגיע לשלב האחרון הנ"ל, מכיוון שבזמן כתיבת התוכנה היתה קיימת בעיה עם נתבים שלא תמכו באופציה של חלוקה לתת-רשתות בנות ביט אחד. בחיים האמיתיים המשחק משתנה קצת, כיוון שהנתבים החדשים דווקא כן תומכים ב- Subnetting של ביט אחד בלבד (מוגדר ב- RFC 1812 - <http://www.faqs.org/rfcs/rfc1812.html>). גם מיקרוסופט לא התעלמה מהקידמה הדוהרת (עאלק...), וב- Windows 2000 ואילך איפשרה חלוקה לתת-רשתות בנות ביט אחד.

כמה כתובות אפשריות של Host ID יש לנו? עכשיו יש לנו 15 ביטים לשחק איתם. לכן, 2 בחזקת 15 שווה 32,768. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 32,766 מחשבים בכל תת-רשת.

### טבלה מסכמת ל- Subnetting של אוקטטה אחת בכתובת מ- Class B:

מס' הביטים המכוסים	Bit Pattern	כמה Subnets יש לנו?	מהו ה- Subnet Mask?	כמה Host ID לכל תת-רשת?
1	10000000	2	255.255.128.0	32,766
2	11000000	4	255.255.192.0	16,382
3	11100000	8	255.255.224.0	8,190
4	11110000	16	255.255.240.0	4,094
5	11111000	32	255.255.248.0	2,046
6	11111100	64	255.255.252.0	1,022
7	11111110	128	255.255.254.0	510
8	11111111	256	255.255.255.0	254

- מספר הביטים המכוסים הוא המספר של הביטים שנשארו לטובת ה- Subnet ID.

- ה- Bit Pattern הוא המבנה של האוקטטה השלישית, זו שעליה "רבים" ה- Subnet ID וה- Host ID.

### כתובת מ- Class A:

בואו נבצע את אותה פעולה, אבל הפעם עם כתובת אחרת, מ- Class A:

ניקח כתובת כמו 30.0.0.0 לפני (מתחת - ה- Subnet Mask המקורי):

Network ID	Host ID		
30	X	Y	Z
255	0	0	0

יש לנו רק רשת אחת גדולה עם 2 בחזקת 24 אפשרויות ל- Host ID שונות (3 אוקטטות שווה 24 ביטים), או 16,777,216 אפשרויות, אבל אסור לשכוח להוריד 2, לכן בעצם 16,777,214 אפשרויות שונות.

עכשיו "נגנוב" אוקטטה אחת לטובת ה- Subnet ID:

Network ID	Subnet ID	Host ID	
30	X	Y	Z
255	255	0	0

יש לנו רשת אחת גדולה שמחולקת ל- 2 בחזקת 8 תת-רשתות או בעצם – 256 תת-רשתות.

כמה Host ID יכולים להיות בכל אחת מהתת-רשתות השונות? יש לנו 2 אוקטטות לשחק איתן, לכן 2 בחזקת 16, כלומר 65,536 מחשבים שונים. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 65,534 מחשבים בכל תת-רשת.

להזכירכם, כל זה במקום רשת אחת ענקית עם 2 בחזקת 24 מחשבים שונים (שזה למעלה מ- 16 מיליון כתובות שונות), כעת יש לנו 256 תת-רשתות, כשכל אחת יכולה להכיל קצת למעלה מ- 65,000 מחשבים שונים. זה יכול להיות פיתרון טוב לחברות ענקיות, או לספקי אינטרנט למשל.

עכשיו בואו נניח, כמו בדוגמאות הקודמות, שכמות המחשבים הזו לא מספיקה לספק האינטרנט. הרי הוא צריך לספק 200,000 כתובות, ואם הוא יעשה Subnetting לפי 255.255.0.0 זה לא יספיק לו. נתחיל לגנוב לאט לאט, ביט ביט מתוך ה- Subnet ID וניתן אותו במתנה ל- Host ID.

אם "נגנוב" את הביט הכי ימני מתוך ה- Subnet ID ונתן אותו במתנה ל- Host ID, מה זה יתן לנו? בואו נראה:

Subnet ID								Host ID													
255	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

עכשיו יש לנו רק 7 ביטים ל- Subnet ID, ו- 17 ביטים ל- Host ID.

ניקח את המספר הבינארי 11111110 ונהפוך לעשרוני. כמה יצא? 254 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.254.0.0

255	254	0	0
-----	-----	---	---

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 7 שווה 128.

בואו נבדוק כמה כתובות אפשרויות של Host ID יש לנו: עכשיו יש לנו 17 ביטים לשחק איתם. לכן, 2 בחזקת 17 שווה 131,072. לא לשכוח להפחית 2 (אסור Host ID שכולו 0 ו/או 1) ולכן נותרו רק עם 131,072 מחשבים בכל תת-רשת.

נמשיך באופן הזה לאורך כל האוקטטה השניה. נוסיף כל פעם ביט אחד עבור ה- Host ID, עד שנגיע לביט הכי שמאלי, ול- Subnet Mask של 255.128.0.0.

אם עדין אתם לא מבינים איך זה נעשה, קחו את הדוגמאות הקודמות והמשיכו באותה דרך לגבי הרשת הנוכחית.

### טבלה מסכמת ל- Subnetting של אוקטטה אחת בכתובת מ- Class A:

מס' הביטים המכוסים	Bit Pattern	כמה Subnets יש לנו?	מהו ה- Subnet Mask?	כמה Host ID לכל רשת?
1	10000000	2	255.128.0.0	8,388,606
2	11000000	4	255.192.0.0	4,194,302
3	11100000	8	255.224.0.0	2,097,150
4	11110000	16	255.240.0.0	1,048,574
5	11111000	32	255.248.0.0	524,286
6	11111100	64	255.252.0.0	262,142
7	11111110	128	255.254.0.0	131,070
8	11111111	256	255.255.0.0	6,5534

- מספר הביטים המכוסים הוא המספר של הביטים שנשאר לטובת ה- Subnet ID.
- ה- Bit Pattern הוא המבנה של האוקטטה השניה, זו שעליה "רבים" ה- Subnet ID וה- Host ID.
- לא צריך לזכור בדיוק את הכמויות של ה- Host ID בכל תת-רשת. רק בקירוב.

### כתובת מ- Class C:

בחלוקה ל- Subnets של רשת מ- Class C המצב משתנה במקצת. יש כאן כמה בעיות שצריך לתת את הדעת עליהן, ולכן דין חלוקת Class C שונה מחלוקת Classes אחרים, למרות שהרעיון נשאר זהה.

ניקח כתובת כמו 192.80.55.0  
לפני (מתחת - ה- Subnet Mask המקורי):

Network ID			Host ID
192	80	55	Z
255	255	255	0

יש לנו רק רשת אחת עם 2 בחזקת 8 אפשרויות ל- Host ID שונות (אוקטטה אחת שווה 8 ביטים), או 256 אפשרויות, אבל אסור לשכוח להוריד 2, לכן בעצם 254 אפשרויות שונות. עכשיו "נגנוב" אוקטטה אחת לטובת ה- Subnet ID:

Network ID			Subnet ID
192	80	55	Z
255	255	255	255

מה קרה? לא נשארו לנו כלל כל ביטים לצורך ה- Host ID. כל הביטים של ה- Host ID משמשים עכשיו ל- Subnet ID, וברור שמצב כזה הוא לא מצב תקין. לכן אי אפשר לקיים רשת מ- Class C עם Subnet Mask של 255.255.255.255, ותזכרו את זה.

מה שאנחנו יכולים לעשות זה לקחת פחות מ- 8 ביטים בשביל ה- Subnet ID, ולהשאיר כמה ביטים עבור ה- Host ID. אם "נגנוב" את הביט הכי ימני מתוך ה- Subnet ID ונתן אותו במתנה ל- Host ID, מה זה יתן לנו? בואו נראה:

Network ID			Subnet ID							Host ID
255	255	255	1	1	1	1	1	1	1	0

עכשיו יש לנו רק 7 ביטים ל- Subnet ID, וביט אחד ל- Host ID.

ניקח את המספר הבינארי 11111110 ונהפוך לעשרוני. כמה יצא? 254 (בדוק!). לכן, ה- Subnet Mask החדש יהיה 255.255.255.254

255	255	255	254
-----	-----	-----	-----



כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 7 שווה 128, ואכן יש לנו 128 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו ביט אחד לשחק איתו. לכן, 2 בחזקת 1 שווה 2. אבל האם באמת יש לנו 2 כתובות שונות? לא. למה לא? כיוון שאסור לנו לשכוח שאסור להשתמש בכתובת שהיא הכל 0, וכתובת שהיא הכל 1. לכן, 2 פחות 2 שווה 0 (אפס), ולכן אי אפשר לחלק רשת מ-Class C ל-128 תת-רשתות שונות. תזכרו את זה.

### חשוב לזכור:

עד עכשיו ראינו שלכתובת מ-Class C אי אפשר לתת Subnet Mask של 255.255.255.255, וגם לא 255.255.255.254, לכן בעצם אי אפשר לחלק כתובת רשת מ-Class C ל-256 או 128 תת-רשתות כמו בשאר ה-Classes. חשוב לזכור את הנקודה הזו. אם אי אפשר לגנוב ביט אחד מה-Subnet ID, בואו ננסה לגנוב 2 ביטים:

ניקח עוד ביט אחד וניתן אותו מתנה ל-Host ID:

Subnet ID								Host ID	
255	255	255	1	1	1	1	1	0	0

עכשיו יש לנו רק 6 ביטים ל-Subnet ID, ו-2 ביטים ל-Host ID.

ניקח את המספר הבינארי 11111100 ונהפוך לעשרוני. כמה יצא? 252. לכן, ה-Subnet Mask החדש יהיה 255.255.255.252

255	255	255	252
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 6 שווה 64, ואכן יש לנו 64 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו 2 ביטים לשחק איתם. לכן, 2 בחזקת 2 שווה 4. 4 פחות 2 שווה 2, ואכן יש לנו 2 Host ID שונים בכל תת-רשת אפשרית.

64 תת-רשתות, כשלכל אחת 2 מחשבים. קצת דבילי, לא? אבל אולי דווקא זה מה שהלקוח צריך? אולי מדובר בסניפים של רשת חנויות כשבכל סניף יש רק קופה רושמת אחת ונתב שיחבר את הסניף לסניף המרכזי?

נמשיך. נניח שלא מספיקים לו 2 מחשבים בכל תת-רשת. הוא צריך יותר. אז בואו ניקח עוד ביט אחד וניתן אותו מתנה ל-Host ID:

Subnet ID								Host ID		
255	255	255	1	1	1	1	1	0	0	0

עכשיו יש לנו רק 5 ביטים ל-Subnet ID, ו-3 ביטים ל-Host ID.

ניקח את המספר הבינארי 11111000 ונהפוך לעשרוני. כמה יצא? 248. לכן, ה-Subnet Mask החדש יהיה 255.255.255.248

255	255	255	248
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 5 שווה 32, ואכן יש לנו 32 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 3 ביטים לשחק איתם. לכן, 2 בחזקת 3 שווה 8. 8 פחות 2 שווה 6, ואכן יש לנו 6 Host ID שונים בכל תת-רשת אפשרית.

לא מספיקים 6 מחשבים בכל תת-רשת. רוצה עוד. נמשיך.

ניקח עוד ביט אחד וניתן אותו מתנה ל-Host ID:

Subnet ID								Host ID			
255	255	255	1	1	1	1	1	0	0	0	0

עכשיו יש לנו רק 4 ביטים ל- Subnet ID, ו- 4 ביטים ל- Host ID.

ניקח את המספר הבינארי 11110000 ונהפוך לעשרוני. כמה יצא? 240. לכן, ה- Subnet Mask החדש יהיה 255.255.255.240

255	255	255	240
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 4 שווה 16, ואכן יש לנו 16 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 4 ביטים לשחק איתם. לכן, 2 בחזקת 4 שווה 16. 16 פחות 2 שווה 14 ואכן יש לנו 14 Host ID שונים בכל תת-רשת אפשרית.

נוצר מצב בו יש לנו עד 16 תת-רשתות כשבכל אחת מהם יש מקסימום 14 מחשבים. זה יכול להיות נוח לחברה קטנה עם כמה מחלקות קטנות.

אם זה לא מספיק, ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

			Subnet ID			Host ID				
255	255	255	1	1	1	0	0	0	0	0

עכשיו יש לנו רק 3 ביטים ל- Subnet ID, ו- 5 ביטים ל- Host ID. ניקח את המספר הבינארי 11100000 ונהפוך לעשרוני. כמה יצא? 224. לכן, ה- Subnet Mask החדש יהיה 255.255.255.224

255	255	255	224
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 3 שווה 8, ואכן יש לנו 8 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 5 ביטים לשחק איתם. לכן, 2 בחזקת 5 שווה 32. 32 פחות 2 שווה 30 ואכן יש לנו 30 Host ID שונים בכל תת-רשת אפשרית.

8 תת-רשתות עם עד 30 מחשבים בכל תת-רשת. לא מספיק? רוצים יותר מחשבים?

ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

			Sub		Host ID					
255	255	255	1	1	0	0	0	0	0	0

עכשיו יש לנו רק 2 ביטים ל- Subnet ID, ו- 6 ביטים ל- Host ID.

ניקח את המספר הבינארי 11000000 ונהפוך לעשרוני. כמה יצא? 192. לכן, ה- Subnet Mask החדש יהיה 255.255.255.192

255	255	255	192
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 2 שווה 4, ואכן יש לנו 4 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 6 ביטים לשחק איתם. לכן, 2 בחזקת 6 שווה 64. 64 פחות 2 שווה 62 ואכן יש לנו 62 Host ID שונים בכל תת-רשת אפשרית.

יש לי כעת 4 תת-רשתות, כשבכל אחת יש מקסימום 62 מחשבים. רוצים עוד?

ניקח עוד ביט אחד וניתן אותו מתנה ל- Host ID:

			Sub	Host ID						
255	255	255	1	0	0	0	0	0	0	0

עכשיו יש לנו רק ביט אחד ל- Subnet ID, ו- 7 ביטים ל- Host ID.

ניקח את המספר הבינארי 10000000 ונהפוך לעשרוני. כמה יצא? 128. לכן, ה- Subnet Mask החדש יהיה 255.255.255.128

255	255	255	128
-----	-----	-----	-----

כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? בואו נראה: 2 בחזקת מספר הביטים שנשארו, כלומר 2 בחזקת 1 שווה 2, ואכן יש לנו 2 תת-רשתות אפשריות.

עכשיו צריך לבדוק כמה כתובות אפשריות של Host ID יש לנו: עכשיו יש לנו 7 ביטים לשחק איתם. לכן, 2 בחזקת 7 שווה 128. 128 פחות 2 שווה 126 ואכן יש לנו 126 Host ID שונים בכל תת-רשת אפשרית.

### טבלה מסכמת ל- Subnetting של כתובת מ- Class C:

מס' הביטים המכוסים	Bit Pattern	כמה Subnets יש לנו?	מהו ה- Subnet Mask?	כמה Host ID לכל תת-רשת?
1	10000000	2	255.255.255.128	126
2	11000000	4	255.255.255.192	62
3	11100000	8	255.255.255.224	30
4	11110000	16	255.255.255.240	14
5	11111000	32	255.255.255.248	6
6	11111100	64	255.255.255.252	2
7	11111110	128	255.255.255.254	אי אפשר
8	11111111	256	255.255.255.255	אי אפשר

- מספר הביטים המכוסים הוא המספר של הביטים שנשארו לטובת ה- Subnet ID.
- ה- Bit Pattern הוא המבנה של האוקטטה הרביעית, זו שעליה "רבים" ה- Subnet ID וה- Host ID.
- לזכור שב- Class C לא יכול להיות Subnet Mask שנגמר ב- 255 או ב- 1254!

### תרגיל להיכרות עם טבלת ה- Subnetting:

קחו את טבלת ה- Subnetting שנמצאת אצלי באתר (ניתנת לצפייה בלינק הבא: [http://www.petri.co.il/subnetting\\_table\\_he.htm](http://www.petri.co.il/subnetting_table_he.htm)) והתבוננו בה. שימו לב למספרי ה- Subnet Mask השונים, וראו שהם בעצם חוזרים על עצמם באיזה סדר מסויים.

מניסיוני, כדאי להשקיע 5 דקות כדי לשקן את הטבלה הזו, ועוד 10 דקות ולהתאמן לצייר אותה מתוך הזיכרון. נסו פעם ועוד פעם, ואחרי הפעם השלישית תראו שלוקח לכם בדיוק חצי דקה לצייר את הטבלה כולה על דף ריק. כשתיכנסו למבחן שלכם, דבר ראשון לפני שאתם מתחילים (בהנחה שאתם לא זוכרים את הטבלה בעל-פה ובהנחה שלא נוח לכם לחשב את הערכים שלה בראש), קחו את אחד מדפי הטייטה וציירו עליו בזריזות את הטבלה. זה אמנם יבזבז חצי דקה מזמנכם, אבל יחסוך לכם הרבה כאבי ראש בהמשך המבחן כשתוכלו להציץ בטבלה ולענות כהרף עין על שאלות, בזמן שאחרים יתאמצו לחשב מספרים בינאריים וכל מיני חישובים אחרים.

**הערה לקראת מבחני ה- Windows 2000/2003:** מכיוון שמבחני ה- MCP בחלונות 2000/2003 שונים במידה רבה מאלה שהיו נהוגים ב- NT, יש לציין שכיום נוטים לשאול פחות ופחות על Subnetting נטו, ויותר שאלות שמשחילות את נושא ה- Subnetting בדלת האחורית. לכן כיום כבר כמעט ואין טעם בשינון הטבלה או בכתיבתה לפני תחילת המבחן, אלא יש להבין יותר את המנגנון שמאחוריה.

עד כאן היכרות עם ה- Subnet Mask החדש. עכשיו נעבור לנושא קצת יותר מורכב.

### כתובות וטווחים של רשתות

אחרי שהסברנו איך מתבצעת פעולת ה- Subnetting עלינו להתחיל להשתמש בזה בפועל. למי שלא מבין את הקטע הבא, אני ממליץ מאוד לחזור לתחילת המאמר ולקרוא (וגם להבין הפעם) את ההסבר לגבי כמות הרשתות והטווחים של כל רשת כשדיברנו על סוגי ה- Classes.

נניח שיש לנו רשת מ- Class B, משהו כמו 191.107.0.0, מה יהיה ה- Subnet Mask המקורי שלה? נכון, 255.255.0.0.

לכמה רשתות זה מספיק? מה זאת אומרת לכמה רשתות? הרי כתובת מ-Class B עם Subnet Mask רגיל, כמו כן, מספיקה אך ורק לרשת אחת. הרי זו כל המטרה של הפרק הזה, לא?

ולכמה Hosts תספיק הרשת הזו? קצת יותר מ-65,000 מחשבים.

נניח לרגע שאני רוצה לקחת את הרשת הזו ולחלק אותה ל- **2 תת-רשתות**. מה עושים עכשיו? אם הייתי זוכר את הטבלה ויודע לקרוא אותה (כמו שהצעתי לכם אבל אתם סירבתם, אתה הרי יותר חכם ממני, לא?), הייתי מסתכל על דף הטיטוט שעליו ציירתי את הטבלה ומיד רואה שכדי לחלק רשת כלשהי ל-2 תת-רשתות, עלי להשתמש ב- Subnet Mask שונה מהרגיל – 255.255.128.0 (ואת זה אתם כבר אמורים לדעת, הרי אתם כל כך חכמים...).

אבל מה עושים אם הייתי טמבל ושכחתי את הטבלה. מה אני עושה עכשיו? איך אני יודע את ה- Subnet Mask שאמור להיות לאותה רשת?

נלך בדרך הארוכה:

אני רוצה לחלק רשת גדולה ל- **2 תת-רשתות**. כתובת הרשת היא **191.107.0.0**. ה- Subnet Mask המקורי שלה הוא **255.255.0.0**.

Network ID		Host ID	
191	107	Y	Z
255	255	0	0

שואל את עצמי: 2 בחזקת כמה שווה ל- **2**? התשובה היא **1**. לכן, כעת נגנוב **1 ביט** מה- Host ID של ה- Subnet Mask המקורי לטובת ה- Subnet ID.

255	255	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

**1. מהו ה- Subnet Mask החדש?**

ניקח את המספר הבינארי **10000000** ונהפוך לעשרוני. ה- Subnet Mask החדש יהיה **255.255.128.0**

255	255	128	0
-----	-----	-----	---

**2. כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask?**

בואו נראה: 2 בחזקת מספר הביטים שנלקחו, כלומר 2 בחזקת **1** שווה **2**.

**3. כמה כתובות אפשריות של Host ID יש לנו בכל תת-רשת?**

פשוט מאוד. הרי "נגנבנו" ביט אחד מה- Host ID לטובת ה- Subnet ID, מה שמשאיר אותי עם 7 ביטים לטובת ה- Host ID במקום ה-8 המקוריים, אבל לא לשכוח שעדיין יש לנו את האוקטטה הרביעית שנותרה "ללא פגע". לכן, בעצם יש לנו 15 ביטים לטובת ה- Host ID.

2 בחזקת 15 שווה 32,768, אבל לא לשכוח שאי אפשר הכל 0 או הכל 1, ולכן מורידים 2. יש לנו 32,766 מחשבים בכל תת-רשת ("אם רק היתה לי הטבלה הזאת מול העיניים...").

**4. מה הכתובת של כל אחת מתת-הרשתות שנוצרו (ה- Network ID)?**

נתבונן ב- **1 ביט** שגנבנו וננסה להוציא ממנו את כל הקומבינציות האפשריות:

0  
1

אני רואה 2 אפשרויות שונות (קל לחשב: 2 בחזקת **1** שווה 2).

כלומר בדיוק 2 אפשרויות שונות. נשלב את האפשרויות האלו ב- Network ID המקורי שלי:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

והאפשרות השניה:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

שימו לב, האוקטטות הראשונות לא משתנות והן נקבעו על-ידי ספק האינטרנט שלי. החלק היחיד שמשתנה הם הביטים השמאליים ביותר של האוקטטה השלישית (במקרה של Class B, ברור שב-Class C אנחנו משחקים עם האוקטטה הרביעית, ואילו ב-Class A עם האוקטטה השנייה).

הופך לעשרוני, ומוצא את כתובות תת-הרשתות שלי:

**191.107.0.0**  
**191.107.128.0**

אלה הכתובות של תת-הרשתות החדשות שלי. מעכשיו יש לנו רשת אחת גדולה שמחולקת ל-2 תת-רשתות (או רחוב אחד ארוך שמחולק ל-2 בלוקים של בניינים), ולכל בלוק כזה יש שם. הבלוק הראשון הוא בלוק **0**, והבלוק השני הוא בלוק **128**.

### 5. מה טווחי ה-Hosts בכל אחת מתת-הרשתות הללו?

נתבונן בתת-הרשת הראשונה: הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נסיף 1:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הראשונה. כלומר:

ברשת **191.107.0.0**  
טווח כתובות ה-IP האפשריות הוא  
**191.107.0.1**  
עד  
**191.107.127.254**

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת הראשונה.

מה לגבי תת-הרשת השנייה? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נסיף 1:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הראשונה. כלומר:

ברשת **191.107.128.0**  
 טווח כתובות ה- IP האפשריות הוא  
**191.107.128.1**  
 עד  
**191.107.255.254**

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת השניה.

### עוד דוגמה?

אני רוצה לחלק רשת גדולה ל- **4 תת-רשתות**. כתובת הרשת היא **191.107.0.0**. ה- Subnet Mask המקורי שלה הוא **255.255.0.0**.

Network ID		Host ID	
191	107	Y	Z
255	255	0	0

שואל את עצמי: 2 בחזקת כמה שווה ל- **4**? התשובה היא **2**. לכן, כעת נגנוב **2 ביטים** מה- Host ID של ה- Subnet Mask המקורי לטובת ה- Subnet ID.

255	255	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

### 1. מהו ה- Subnet Mask החדש?

ניקח את המספר הבינארי **11000000** ונהפוך לעשרוני. ה- Subnet Mask החדש יהיה **255.255.192.0**

255	255	192	0
-----	-----	-----	---

### 2. כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask?

בואו נראה: 2 בחזקת מספר הביטים שנלקחו, כלומר 2 בחזקת **2** שווה **4**.

### 3. כמה כתובות אפשריות של Host ID יש לנו?

נותרו לי 14 ביטים לשחק איתם. לכן, 2 בחזקת 14 שווה 16,384. 16,384 פחות 2 שווה 16,382, ואכן יש לנו 16,382 Host ID שונים בכל תת-רשת אפשרית.

### 4. מה הכתובת של כל אחת מתת-הרשתות שנוצרו (ה- Network ID)?

נתבונן ב- **2 הביטים** שגנבנו וננסה להוציא מהם את כל הקומבינציות האפשריות:

00  
 01  
 10  
 11

נשלב את האפשרויות האלו ב- Network ID המקורי שלי:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

והאפשרות השניה:

191	107	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

והשלישית:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

והרביעית:

191	107	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הופך לעשרוני, ומוצא את כתובות תת-הרשתות שלי:

191.107.0.0  
 191.107.64.0  
 191.107.128.0  
 191.107.192.0

אלה הכתובות של תת-הרשתות החדשות שלי. מעכשיו יש לנו רשת אחת גדולה שמחולקת ל-4 תת-רשתות (או רחוב אחד ארוך שמחולק ל-4 בלוקים של בניינים), ולכל בלוק כזה יש שם. הבלוק הראשון הוא בלוק 0, והבלוק השני הוא בלוק 64, הבלוק השלישי הוא בלוק 128 והבלוק הרביעי הוא בלוק 192.

### 5. מה טווחי ה-Hosts בכל אחת מתת-הרשתות הללו?

הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הראשונה.

ברשת 191.107.0.0

טווח כתובות ה-IP האפשריות הוא

191.107.0.1

עד

191.107.63.254

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת הראשונה.

נתבונן בתת-רשת השנייה. הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השנייה.

ברשת **191.107.64.0**  
טווח כתובות ה-IP האפשריות הוא  
**191.107.64.1**  
עד  
**191.107.127.254**

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת השנייה.

מה לגבי תת-הרשת השלישית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השלישית.

ברשת **191.107.128.0**  
טווח כתובות ה-IP האפשריות הוא  
**191.107.128.1**  
עד  
**191.107.191.254**

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת השלישית.

מה לגבי תת-הרשת הרביעית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0:

191	107	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1:

191	107	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הרביעית.

ברשת **191.107.192.0**  
טווח כתובות ה-IP האפשריות הוא  
**191.107.192.1**  
עד  
**191.107.255.254**

בטווח זה אפשר להקצות כתובות IP למחשבים שאמורים להיות חלק מתת-הרשת הרביעית.





191.107.0.0  
 191.107.32.0  
 191.107.64.0  
 191.107.96.0  
 191.107.128.0  
 191.107.160.0  
 191.107.192.0  
 191.107.224.0

אלה הכתובות של תת-הרשתות החדשות שלי. מעכשיו יש לנו רשת אחת גדולה שמחולקת ל-8 תת-רשתות, ולכל תת-רשת כזו יש שם. הבלוק הראשון הוא בלוק 0, והבלוק השני הוא בלוק 32. הבלוק השלישי הוא בלוק 64 וכו'. הבלוק האחרון הוא בלוק 224.

### 5. מה טווחי ה-Hosts בכל אחת מתת-הרשתות הללו?

הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הראשונה.

ברשת 191.107.0.0  
 טווח כתובות ה-IP האפשריות הוא  
 191.107.0.1  
 עד  
 191.107.31.254

מה לגבי תת-הרשת השנייה? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השנייה.

ברשת 191.107.0.0  
 טווח כתובות ה-IP האפשריות הוא  
 191.107.32.1  
 עד  
 191.107.63.254

מה לגבי תת-הרשת השלישית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השלישית.

**ברשת 191.107.64.0**

טווח כתובות ה-IP האפשריות הוא

**191.107.64.1**

עד

**191.107.95.254**

מה לגבי תת-הרשת הרביעית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת הרביעית.

**ברשת 191.107.96.0**

טווח כתובות ה-IP האפשריות הוא

**191.107.96.1**

עד

**191.107.127.254**

מה לגבי תת-הרשת החמישית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת החמישית.

**ברשת 191.107.128.0**

טווח כתובות ה-IP האפשריות הוא

**191.107.128.1**

עד

**191.107.159.254**

מה לגבי תת-הרשת השישית? הקצה האחד של הטווח מתחיל כאשר ה-Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסף 1:

191	107	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה-Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השישית.

**ברשת 191.107.160.0**

טווח כתובות ה-IP האפשריות הוא

**191.107.160.1**

עד

**191.107.191.254**

מה לגבי תת-הרשת השביעית? הקצה האחד של הטווח מתחיל כאשר ה- Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסיף 1:

191	107	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה- Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השביעית.

ברשת **191.107.192.0**

טווח כתובות ה- IP האפשריות הוא

**191.107.192.1**

עד

**191.107.223.254**

בשאלה המקורית ביקשו 7 תת-רשתות, אבל אנחנו יודעים שיש לנו בפועל 8. אולי לא נצטרך להשתמש ברשת האחרונה, אבל בכל זאת, כדי לתרגל, נתייחס אליה כחלק מדרישות השאלה. הקצה האחד של הטווח מתחיל כאשר ה- Host ID הוא הכי נמוך, כלומר 0. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 0, לכן נוסיף 1:

191	107	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הקצה השני של הטווח נגמר כאשר ה- Host ID הוא הכי גבוה, כלומר 1. אבל כמו שאמרנו אי-אפשר Host ID שהוא הכל 1, לכן נחסיר 1:

191	107	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה התרגום של שני הקצוות לעשרוני? זהו טווח תת-הרשת השמינית.

ברשת **191.107.224.0**

טווח כתובות ה- IP האפשריות הוא

**191.107.224.1**

עד

**191.107.255.254**

## הדרך המהירה לחישובי תת-רשתות וטווחים

יש דרך מהירה וטובה לחשב את כל החישובים הללו, את ה- Subnet Mask, את כתובות תת-הרשתות, את הטווחים של כל אחת מתת-הרשתות, ואת מספר ה- Hosts המותרים בכל תת-רשת. נקודת המפתח היא הבנה מוחלטת של החוקים שנוסחו כאן, תוך כדי תרגול מתמיד ומקיף של עשרות שאלות ודוגמאות. החוקים שיוצגו לעיל טובים אך ורק אם התלמיד מבין את משמעותם ולא זוכר אותם כמו תוכי, שכן קל מאוד להתבלבל ולטעות, ואם אין לנו שום מנגנון בקרה עצמית ("זה לא יכול להיות, נראה לי חשוד, בוא נבדוק שוב") קל מאוד לטעות וללכת בדרך שאינה נכונה. לא יעזרו לכם שום טבלאות ושום נוסחאות סודיות אם המוח לא מבין את מה הוא עושה. לכן חשוב מאוד לתרגל, ורק אז תבוא ההארה. בכל זאת, כדי להקל על חלקכם (זאת מטרת המאמר, לא?) נסכם את הממצאים שלנו בטבלה קטנה:

כתובת רשת מקורית	Subnet Mask	כתובות תת-רשת	טווח	כמה מחשבים בכל תת-רשת
191.107.0.0	255.255.224.0	191.107.0.0	עד 191.107.31.254	8,190
		191.107.32.0	עד 191.107.63.254	
		191.107.64.0	עד 191.107.95.254	
		191.107.96.0	עד 191.107.127.254	
		191.107.128.0	עד 191.107.159.254	
		191.107.160.0	עד 191.107.191.254	
		191.107.192.0	עד 191.107.223.254	
		191.107.224.0	עד 191.107.255.254	

תגידו, כמה ביטים גנבנו לצורך ה- Subnet ID? **3 ביטים**. אם נכניס את הביטים האלה לטבלה בינארית, נראה תופעה מעניינת:

128	64	32	16	8	4	2	1
1	1	1	0	0	0	0	0

מהו הערך של **הביט הימני ביותר** (מאלה שגנבנו)? **32**.

128	64	32	16	8	4	2	1
0	0	1	0	0	0	0	0

חיזרו לטבלה למעלה וראו את כתובות תת-הרשת שנוצרו:

כתובות הרשת המקורית היא תמיד כתובת תת-הרשת הראשונה

191.107.0.0  
191.107.32.0  
191.107.64.0  
191.107.96.0  
191.107.128.0  
191.107.160.0  
191.107.192.0  
191.107.224.0

הערך שמצאנו ב- Subnet Mask החדש מתייג מגולם בתת-הרשת האחרונה

שימו לב באיזה קפיצות תת-הרשתות עולות. יש כאן מגמה ברורה:

191.107.0.0  
 191.107.32.0  
 191.107.64.0  
 191.107.96.0  
 191.107.128.0  
 191.107.160.0  
 191.107.192.0  
 191.107.224.0

**חוק:** כתובת הרשת הראשונה מתחילה תמיד ב- (וזה למעשה ל- Network ID המקורי שנתנו לנו), והדילוגים בין כתובות תת-הרשתות השונות קופצות בדיוק באותו מספר כמו הערך של הביט הימני ביותר מבין הביטים ה"גנובים". עד לאיזה ערך מדלגים? עד לערך של ה- Subnet Mask.

191.107.0.0  
 191.107.32.0  
 191.107.64.0  
 191.107.96.0  
 191.107.128.0  
 191.107.160.0  
 191.107.192.0  
 191.107.224.0

ה- Subnet Mask 255.255.224.0

עיבוד נוסף של החוק שמותאם לביצוע Subnetting של אוקטטה אחת בלבד (נלמד בהמשך איך ניתן לחלק רשת לפי יותר מאוקטטה אחת) הוא כדלקמן:

**חוק עזר חלופי לחוק הקודם:** ערכי הקפיצות של כתובות תת-הרשתות השונות הוא בדיוק 256 לחלק למספר תת-הרשתות הדרוש.

כלומר במקרה זה 256 לחלק ל- 8 שווה בדיוק 32, ובמקרה זהו הערך שבו קופצות הכתובות של תת-הרשתות השונות. לצערנו חוק עזר חלופי זה ישים לחלוקה של אוקטטה אחת בלבד, והוא יעבוד אך ורק אם נרצה לחשב באמצעותו חלוקות של אוקטטה אחת בלבד. מצד שני, החוק הראשון מתאים לחלוקה של כל אוקטטה, ולא דווקא הראשונה, ולכן רצוי לעבוד לפי החוק הראשון ולנסות להבין את מה שמסתתר מאחוריו.

ומה לגבי הטווחים של ה- Host ID בכל תת-רשת? שימו לב, גם כאן יש היגיון:

תת-רשת ראשונה 191.107.0.0  
 191.107.0.1  
 עד  
 191.107.31.254

תת-רשת שנייה 191.107.32.0  
 191.107.32.1  
 עד  
 191.107.63.254

תת-רשת שלישית 191.107.64.0  
 191.107.64.1  
 עד  
 191.107.95.254

תת-רשת רביעית 191.107.96.0  
 191.107.96.1  
 עד  
 191.107.127.254

וכו...

**חוק: הטווח מתחיל בכתובת תת-הרשת שלי פלוס 1, ונגמר בכתובת תת-הרשת הבאה בתור מינוס 2!!!**

שימו לב שוב!

תת-רשת ראשונה **191.107.0.0**

**191.107.0.1**

עד

**191.107.31.254**

תת-רשת שנייה **191.107.32.0**

**191.107.32.1**

עד

**191.107.63.254**

תת-רשת שלישית **191.107.64.0**

**191.107.64.1**

עד

**191.107.95.254**

תת-רשת רביעית **191.107.96.0**

**191.107.96.1**

עד

**191.107.127.254**

תת-רשת חמישית **191.107.128.0**

וכו...

למה פלוס 1? כי אם זה לא היה כך היינו מקבלים כתובת של מחשב כמו 191.107.64.0, וכבר סגרנו על זה שאסור כתובת Host שבה כל הביטים 0. שוב פעם, אם את/ה לא מבינים את המשמעות של המשפט הקודם, (כלומר אם את/ה לא מבין למה לעזאזל בכתובת תמימה כמו 191.107.64.0 נוצר מצב שבו כל הביטים של ה-Host ID הם 0), בבקשה לחזור להתחלה ולבצע את השלבים מא' ועד ת'.

למה פחות 2? כי אם זה היה כתובת הרשת הבאה פחות 1, הינו מקבלים כתובת של מחשב כמו 191.107.127.255, והרי סיכמנו כבר בהתחלה שאסור בהחלט שתהיה כתובת של Host שבה כל הביטים 1. ההמלצה לגבי הבנת החומר תקפה גם כאן.

בדרך זו אפשר למצוא בשניות את כל הכתובות וטווחי הרשתות השונות.

**הערה חשובה לגבי חוק הטווחים:** מניסיוני, אחרי שבוע שבועיים ללא תירגול בחומר הנ"ל תלמידים נוטים לשכוח מדוע יש להוסיף ביט אחד ולהחסיר 2 מהקפיצה הבאה. תמיד תנסו לזכור את החוק הזה, ואם שכחתם את הסיבה לחוק, אנא קראו שוב את הדוגמאות.

**ננסה עוד דוגמה:**

אני רוצה לחלק רשת גדולה ל- **50 תת-רשתות**. כתובת הרשת היא **191.107.0.0**. ה- Subnet Mask המקורי שלה הוא **255.255.0.0**.

Network ID		Host ID	
191	107	Y	Z
255	255	0	0

שואל את עצמי: 2 בחזקת כמה שווה ל- **50**? התשובה היא שאין כזה דבר, לכן צריך לחפש את התשובה הכי קרובה, אבל מלמעלה, לא מלמטה, כי אחרת לא יהיו לי מספיק תת-רשתות. הכי קרוב זה **64**. לכן התשובה היא בעצם **6**, כי 2 בחזקת **6** שווה ל- **64**. לכן, כעת נגנוב **6 ביטים** מה- Host ID של ה- Subnet Mask המקורי לטובת ה- Subnet ID.

255	255	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

## 1. מהו ה- Subnet Mask החדש?

ניקח את המספר הבינארי **11111100** ונהפוך לעשרוני. ה- Subnet Mask החדש יהיה **255.255.252.0**

255	255	252	0
-----	-----	-----	---

## 2. כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask?

בואו נראה: 2 בחזקת מספר הביטים שנלקחו, כלומר 2 בחזקת **6** שווה **64**. כלומר סה"כ 64 תת-רשתות שונות. רצינו 50, אבל קיבלנו 64.

## 3. כמה כתובות אפשריות של Host ID יש לנו?

יש כאן 10 ביטים לשחק איתם. לכן, 2 בחזקת 10 שווה 1,024. 1,024 פחות 2 שווה 1,022, ואכן יש לנו 1,022 Host ID שונים בכל תת-רשת אפשרית.

## 4. מה הכתובת של כל אחת מתת-הרשתות שנוצרו (ה- Network ID)?

נתבונן ב- **6 הביטים** שגנבנו וננסה להוציא מהם את כל הקומבינציות האפשריות:

000000  
000001  
000010

וכן, אבל עכשיו זה יהיה מאוד מאוד מטופש להתחיל לכתוב את כל האפשרויות השונות. לכן, נקצר כמו שלמדנו קודם.

ניקח את האפשרות הראשונה:

191	107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

מה הערך של הביט הימני ביותר שגנבנו?

128	64	32	16	8	4	2	1
0	0	0	0	0	1	0	0

לכן, לפי מה שלמדנו קודם, הכתובות של תת-הרשתות יהיו (**קפיצות של 4**):

191.107.0.0  
191.107.4.0  
191.107.8.0  
191.107.12.0  
191.107.16.0

וכן...

תת-הרשתות האחרונות:

191.107.240.0  
191.107.244.0  
191.107.248.0  
191.107.252.0

ה- Subnet Mask החדש - **255.255.252.0**

באותה מידה יכולנו לקחת 256, לחלק במספר תת-הרשתות ולקבל בדיוק **4**!

סה"כ **64 תת-רשתות** שונות (למי שמטיל דופי, נא לכתוב את כולם ולעמוד בפניה!).

**הרחבה לגבי כתובות תת-הרשתות:** לעיתים, במיוחד בשאלות הכוללות מספרים גדולים של תת-רשתות, יתכן וישאלו אותכם לגבי זהות תת-הרשת ה- 50 למשל, או הטווח של תת-הרשת ה- 35. זה יהיה מאוד



מטופש להתחיל לכתוב את כל 50 האפשרויות על נייר רק כדי להגיע לתשובה הנכונה, ולכן אפשר לקצר גם כאן:

$$\text{Hop} * (n-1)$$

למשל, תת-הרשת ה- 50 היא:

$$4 * (50-1) = 4 * 49 = 196$$

ולכן התשובה היא:

191.107.196.0

#### 5. מה טווחי ה- Hosts בכל אחת מתת-הרשתות הללו?

לפי מה שלמדנו קודם. הטווח מתחיל בכתובת תת-הרשת פלוס 1, ונגמר בכתובת תת-הרשת הבאה בתור מינוס 2:

תת-רשת ראשונה 191.107.0.0

191.107.0.1

עד

191.107.3.254

תת-רשת שנייה 191.107.4.0

191.107.4.1

עד

191.107.7.254

תת-רשת שלישית 191.107.8.0

191.107.8.1

עד

191.107.11.254

תת-רשת רביעית 191.107.12.0

191.107.12.1

עד

191.107.15.254

תת-רשת חמישית 191.107.16.0

וכו...

תת-רשת לפני אחרונה 191.107.248.0

191.107.248.1

עד

191.107.251.254

תת-רשת אחרונה 191.107.252.0

191.107.252.1

עד

191.107.255.254

**עוד דוגמה:**

הפעם נעשה דוגמה עם רשת מ-Class C, כיוון שכאן החישובים אמנם עובדים בדיוק באותה דרך, אבל למראית עין יש כאן קושי מסויים.

אני רוצה לחלק רשת גדולה ל- **20 תת-רשתות**. כתובת הרשת היא **199.80.75.0**. ה- Subnet Mask המקורי שלה הוא **255.255.255.0**.

Network ID			Host ID
199	80	75	Z
255	255	255	0

שואל את עצמי: 2 בחזקת כמה שווה ל- **20**? התשובה היא שאין כזה דבר, לכן צריך לחפש את התשובה הכי קרובה, אבל מלמעלה, לא מלמטה, כי אחרת לא יהיו לי מספיק תת-רשתות. הכי קרוב זה **32**. לכן התשובה היא בעצם **5**, כי 2 בחזקת **5** שווה ל- **32**. לכן, כעת נגנוב **5 ביטים** מה- Host ID של ה- Subnet Mask המקורי לטובת ה- Subnet ID.

255	255	255	1	1	1	1	1	0	0	0
-----	-----	-----	---	---	---	---	---	---	---	---

**1. מהו ה- Subnet Mask החדש?**

ניקח את המספר הבינארי **11111000** ונהפוך לעשרוני. ה- Subnet Mask החדש יהיה **255.255.255.248**

255	255	255	248
-----	-----	-----	-----

**2. כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask?**

בואו נראה: 2 בחזקת מספר הביטים שנלקחו, כלומר 2 בחזקת **5** שווה **32**. כלומר סה"כ 32 תת-רשתות שונות. רצינו 20, אבל קיבלנו 32.

**3. כמה כתובות אפשריות של Host ID יש לנו?**

יש כאן 3 ביטים לשחק איתם. לכן, 2 בחזקת 3 שווה 8. 8 פחות 2 שווה 6, ואכן יש לנו 6 Host ID שונים בכל תת-רשת אפשרית.

**4. מה הכתובת של כל אחת מתת-הרשתות שנוצרו (ה- Network ID)?**

נתבונן ב- **5 הביטים** שגנבנו וננסה להוציא מהם את כל הקומבינציות האפשריות:

00000  
00001  
00010

וכל. גם כאן זה יהיה מטופש להתחיל לכתוב את כל האפשרויות השונות. לכן, נקצר כמו שלמדנו קודם.

ניקח את האפשרות הראשונה:

255	255	255	0	0	0	0	0	0	0	0
-----	-----	-----	---	---	---	---	---	---	---	---

מה הערך של הביט הימני ביותר שגנבנו?

128	64	32	16	8	4	2	1
0	0	0	0	1	0	0	0

לכן, לפי מה שלמדנו קודם, הכתובות של תת-הרשתות יהיו (קפיצות של 8):

199.80.75.0  
199.80.75.8  
199.80.75.16  
199.80.75.24  
199.80.75.32

תת-רשתות האחרונות:

199.80.75.224  
199.80.75.232  
199.80.75.240  
199.80.75.248

ה- Subnet Mask החדש - 255.255.248.0

סה"כ 32 תת-רשתות שונות.

אגב, לו היו שואלים אותנו מהי כתובת תת-רשת ה-20 הייתי יכול ללכת לפי הקיצור:

$$\text{Hop} * (n-1)$$

$$8 * (20-1) = 8 * 19 = 152$$

ולכן התשובה היא:

199.80.75.152

**הערה חשובה:** ב- Subnetting של רשתות מ-Class C נוצר מצב מוזר שבו כתובות IP שלכאורה נראו לגיטימיות ו"כשרות" למהדרין הן בעצם בכלל לא כתובות IP אלא בעצם Subnet IDs.

הנה, לדוגמה הכתובת 199.80.75.24 שנראית כמו כתובת IP לכל דבר איננה בעצם יכולה לשמש ככתובת IP במידה וה- Subnet Mask שלה הוא 255.255.255.248 וזאת מכיוון שהיא מייצגת Subnet ID ולא כתובת של ממש. אם תבדקו באמצעות AND את ה- Network ID של הכתובת תגלו לחרדתכם כי התשובה זזה לכתובת עצמה, וזה אומר רק דבר אחד – כל הביטים שמשמשים את ה- Host ID בכתובת הזו הם 0, מה שכמובן לא יכול להיות.

## 5. מה טווחי ה- Hosts בכל אחת מתת-רשתות הללו?

נלך לפי מה שלמדנו קודם. הטווח מתחיל בכתובת הרשת פלוס 1, ונגמר בכתובת הרשת הבאה בתור מינוס 2. כאן צריכים להיזהר, כי קשה לנו להבחין במבט ראשון את ההתחלה והסוף של כל רשת. אין לנו כאן אוקטטה אחרונה עם 0 או 255 שנוכל לזהות כבעייתית, ולכן חייבים להיצמד לחוק הפלוס 1 מינוס 2 שלמדנו קודם:

תת-רשת ראשונה 199.80.75.0  
עד  
199.80.75.6

תת-רשת שנייה 199.80.75.8  
עד  
199.80.75.14

תת-רשת שלישית 199.80.75.16  
עד  
199.80.75.22

תת-רשת רביעית 199.80.75.24  
עד  
199.80.75.30

תת-רשת חמישית 199.80.75.32

וכן...

תת-רשת לפני אחרונה **199.80.75.240**

**199.80.75.241**

עד

**199.80.75.246**

תת-רשת אחרונה **199.80.75.248**

**199.80.75.249**

עד

**199.80.75.254**

חשוב מאוד לשים לב ל- Subnetting ברשתות של Class C, כיוון שכאמור קשה מאוד לזהות במבט ראשון התחלה וסיום של תת-רשת. לזכור את זה טוב טוב.

### עוד דוגמה, אבל הפעם דומה לשאלה אמיתית במבחן:

יש לך מחשב שמנסה לפנות לשרת מרוחק בסגמנט אחר ולא מצליח. לאף מחשב אחר באותה רשת אין בעיה להגיע לשרת, רק לתחנת העבודה הספציפית הזו. התחנה הספציפית לא נתקלת בשום קשיים בתקשורת עם שרתים מקומיים. הלכת אל התחנה, ביצעת Ipconfig וקיבלת את הנתונים הבאים:

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . :

IP Address. . . . . : 198.169.1.53

Subnet Mask . . . . . : 255.255.255.240

Default Gateway . . . . . : 198.169.1.33

מה הבעיה? מדוע לא יכול המחשב לצאת לשרת?

קודם כל, מכיוון שקיבלנו נתון חשוב שאומר ששאר המחשבים הם בסדר, ברור שהגדרות ה- Default Gateway עצמו הן בסדר. הרי אילו הוא עצמו לא היה מתפקד כהלכה, אז אף אחד לא היה יכול להגיע לשרת המרוחק.

אז איפה הבעיה? בתחנת העבודה. אבל איפה בתחנה? הגדרות ה- IP או הגדרות ה- Default Gateway?

אם נסתכל על הגדרות התחנה מיד נראה שמדובר ב- Subnetting (בגלל הערך הלא שלם של ה- Subnet Mask). לפי החוקים נתחיל לפרק את הנתונים ונראה מול מה אנחנו עומדים:

240 נראה בבינארי כך:

128	64	32	16	8	4	2	1
1	1	1	1	0	0	0	0

לכמה תת-רשתות מפוצלת הרשת המקורית? 4 ביטים, 2 בחזקת 4 שווה 16, כלומר ל- 16 תת-רשתות שונות.

מה הערך של הביט "הגנוב" הכי ימני? 16. אם כך, יש לי 16 תת-רשתות שונות, כשהכתובות שלהן נמצאות ב"קפיצות" של 16 זו מזו.

**198.169.1.0**

**198.169.1.16**

**198.169.1.32**

**198.169.1.48**

**198.169.1.64**

**198.169.1.80**

**198.169.1.96**

וכו' וכו' עד שנגיע לרשתות האחרונות

**198.169.1.208**

**198.169.1.224**

**198.169.1.240**

מהם הטווחים של כל תת-רשת? נלך לפי מה שלמדנו קודם. הטווח מתחיל בכתובת הרשת פלוס 1, ונגמר בכתובת הרשת הבאה בתור מינוס 2. כאן צריכים להיזהר, כי קשה לנו להבחין במבט ראשון את ההתחלה והסוף של כל רשת. אין לנו כאן אוקטטה אחרונה עם 0 או 255 שנוכל לזהות כבעייתית, ולכן חייבים להיצמד לחוק הפלוס 1 מינוס 2 שלמדנו קודם:

תת-רשת ראשונה **198.169.1.0**

**198.169.1.1**

עד

**198.169.1.14**

תת-רשת שנייה **198.169.1.16**

**198.169.1.17**

עד

**198.169.1.30**

תת-רשת שלישית **198.169.1.32**

**198.169.1.33**

עד

**198.169.1.46**

תת-רשת רביעית **198.169.1.48**

**198.169.1.49**

עד

**198.169.1.62**

וכו', אבל מספיק לי עד כאן לצורך השאלה הספציפית הזו.

אם נסתכל טוב טוב על הטווחים נראה שכתובת ה-IP של התחנה, **198.169.1.53**, נמצאת בטווח המתאים **לתת-הרשת הרביעית**.

לעומת זאת, כתובת ה-IP של ה-Default Gateway, **198.169.1.33**, נמצאת בטווח המתאים **לתת-הרשת השלישית**.

אי אפשר לתת למחשב כתובת IP וכתובת של Default Gateway הנמצאות בתת-רשתות או סגמנטים נפרדים. זה כמו להגיד לבנאדם "אם יש שריפה תצא מדלת החירום". הוא מיד ישאל "איפה דלת החירום", ואתה תגיד "אהה, היא נמצאת בבניין השני...". זה בלתי אפשרי.

לכן ברור מיד שכתובת ה-IP של ה-Default Gateway לא הוגדרה נכון בתחנה.

רגע! עצור! אך אנחנו יודעים שלא במקרה כתובת ה-IP של התחנה לא הוגדרה נכון? אולי אם היינו נותנים כתובת IP מהטווח הנכון אז הכל היה מסתדר, לא?

לא. מכיוון שהשאלה נתנה טיפ נוסף שאומר שלתחנה אין כל בעיות ליצור קשר עם שרתים מקומיים, מה שבהכרח אומר שכתובת ה-IP שלה נכונה ומוגדרת טוב, אחרת היה לה בלתי אפשרי לתקשר מקומית עם השרתים.

**עוד דוגמה:**

יש לך רשת מסוג Class B ואתה מתבקש לפצל אותה להרבה תת-רשתות, אבל שבכל תת-רשת לא יהיו פחות מ-**500** מחשבים. איך פותרים את זה? פשוט מאוד.

נניח שמדובר בכתובת הרשת היא **191.107.0.0**. ה-Subnet Mask המקורי שלה הוא **255.255.0.0**.

Network ID		Host ID	
191	107	Y	Z
255	255	0	0

הפעם אני לא מתחיל מה- Subnet ID, אלא דווקא מה- Host ID, כיוון שזהו הנתון היחיד שיש לי בשאלה. אני זקוק ל- **500** מחשבים לפחות בכל תת-רשת. לכן -

שואל את עצמי: 2 בחזקת כמה שווה ל- **500**? התשובה היא שאין כזה דבר, לכן צריך לחפש את התשובה הכי קרובה, אבל מלמעלה, לא מלמטה, כי אחרת לא יהיו לי מספיק מחשבים בכל תת-רשת. הכי קרוב זה 512, ו- 2 בחזקת **9** שווה 512. לכן, כעת נדאג שמה שלא יהיה, ישארו ברשותי לפחות **9 ביטים** מה- Host ID. את שאר הביטים שנותרו לי מה- Host ID המקורי, **7 ביטים** במספר, אהפוך כרגיל ל- 1.

255	255	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
-----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

#### 1. מהו ה- Subnet Mask החדש?

ניקח את המספר הבינארי **11111110** ונהפוך לעשרוני. ה- Subnet Mask החדש יהיה **255.255.254.0**

255	255	254	0
-----	-----	-----	---

באמצעות Subnet Mask כזה השארתי לעצמי לפחות **9 ביטים** לצורך ה- Host ID, ואת השאר "השאלתי" לטובת ה- Subnet ID, כרגיל.

לכמה תת-רשתות הצלחתי לחלק את הרשת המקורית, ועדין להשאיר ברשותי **500** מחשבים לכל רשת? כמה תת-רשתות יכולות להיות לנו אם נשתמש בכזה Subnet Mask? 2 בחזקת מספר הביטים שנלקחו לטובת ה- Subnet Mask, כלומר 2 בחזקת 7 שווה **128**. זאת אומרת שברשותי **128** תת-רשתות כשבכל אחת מהן יש **510** מחשבים. למה 510 ולא 512? לא לשכוח שאסור Host ID שבו כל הביטים הם 0 או 1, ולכן אני תמיד מפחית 2.

כמובן שאפשר לפתח את השאלה הזו הלאה, ולשאול מה יהיה ה- Subnet ID של כל אחת מתת-הרשתות, ומה יהיה הטווח של כל אחת מהן. אם ברצונכם תרגלו את עצמכם בשאלות מסוג אלה.

## לסיכום –

במאמר ארוך זה השתדלתי להציג נושא חשוב מאוד למנהלי רשת, נושא שלצערי איננו מוכר במאת האחוזים וגורם לבעיות אצל אנשי מקצוע רבים.

- **החלק הראשון** דנתי במרכיבים הטכניים של כתובת ה-IP, חוקי יסוד בכתובות IP, ומסביר קצת על אופן ההגדרה של הפרוטוקול. הסברתי מושגים כמו כתובת, IP, Subnet Mask, Default Gateway, Net ID ו-Host ID. בנוסף הזכרתי את השיטות השונות להקצאת כתובות IP למחשבים.
- **בחלק השני** לימדתי שיטה נוחה להמרה ממספרים בינאריים לעשרוניים ומעשרוני לבינארי. הצגתי את החלוקה ל-Classes השונים, אבחנה בין ה-Classes השונים ועל כמות הרשתות והמחשבים בכל רשת מה-Classes השונים.
- **בחלק השלישי** הזכרתי את פעולת ה-AND, משמעות ה-Subnetting, על Subnetting של אוקטטה שלמה, של חלק מהאוקטטה, של יותר מאוקטטה אחת, ולימדתי אותכם כמה חוקים שיעזרו לכם לפתור שאלות בנושא.

אני מקווה שהמאמר מצא חן בעיניכם ושהצלחתי להעביר מעט מהידע שלי בנושא. אני יודע על תלמידים ואנשי מקצוע רבים שחזרו אלי עם פידבקים מעולים על המאמר, ולכן אבקש גם מכם לשלוח אלי חוות דעת מקצועיות, תיקונים והגהות לשגיאות כתיב אם תמצאו כאלה.

הגירסה העדכנית ביותר של המאמר נמצאת להורדה מהאתר שלי בלינק הבא:

[http://www.petri.co.il/subnetting\\_tutorial\\_he.htm](http://www.petri.co.il/subnetting_tutorial_he.htm)

תצטרכו תוכנה כמו WinZip על מנת לפתוח את הקובץ המכוון, ותוכנה כמו Adobe Acrobat Reader על מנת לקרוא אותו.

בנוסף, אבקש מכל אדם שקורא מאמר זה ונתקל בו שלא בהתייחסות אלי ולשמי להודיע לי על כך בהקדם. אין לי בעיה עם כך שהמאמר מופץ בחופשיות, אבל יש לי בעיה גדולה עם גניבת זכויות יוצרים בידי אנשי מקצוע מכובדים לכאורה שמורידים את שם המחבר בשיטתיות מן המאמר המקורי ומפיצים אותו בכיתות או במקום עבודתם בצורה שקרית, כאילו הם עצמם כתבו אותו.

לסיכום, הנה טבלה שהכנתי עבור תלמידי שמסכמת את כל נושא ה-Subnetting עד כמה שאפשר. תוכלו להדפיס אותה וללמוד אותה, אבל בכל מקרה אל תסתמכו אך ורק עליה. הבנת החומר היא המפתח להצלחה, לא שינון בעל-פה של טבלאות או נוסחאות.

		# of Masked bits	# of Non-masked bits	Bit pattern	# of subnets = $2^M$	What will the new subnet mask be?	# of hosts per subnet (C Class) = $[(2^N)-2]$	# of hosts per subnet (B Class) = $[(2^N)-2]$	# of hosts per subnet (A Class) = $[(2^N)-2]$
	None	0	8	00000000	0	0	254	65,534	~16 Million
1	1	1	7	10000000	2	128	126	32,766	~8 Million
2	2	2	6	11000000	4	192	62	16,382	~4 Million
3	4	3	5	11100000	8	224	30	8,190	~2 Million
4	8	4	4	11110000	16	240	14	4,094	~1 Million
5	16	5	3	11111000	32	248	6	2,046	~520,000
6	32	6	2	11111100	64	252	2	1,022	~260,000
7	64	7	1	11111110	128	254	-	510	~130,000
8	128	8	0	11111111	256	255	-	254	~65,000

גירסה עברית של הטבלה נמצאת כאן:

[http://www.petri.co.il/subnetting\\_table\\_he.htm](http://www.petri.co.il/subnetting_table_he.htm)

גירסה אנגלית של הטבלה נמצאת כאן:

[http://www.petri.co.il/subnetting\\_table.htm](http://www.petri.co.il/subnetting_table.htm)

כתבתי קובץ שאלות ותשובות שמסכם את נושא ה- Subnetting בפרט ואת נושא ה- TCP/IP בכלל. קובץ זה לא נמצא כלינק להורדה מהאתר שלי, אבל אם תשלחו לי מכתב קצר ובו תציגו את עצמכם ותבקשו (יפה) את הקובץ נראה מה נוכל לעשות.

המשך 'בוא...

טענות, בעיות, קושיות ושאלות, הצעות ופתרונות אפשר לשלוח לדואר אלקטרוני: [mct@petri.co.il](mailto:mct@petri.co.il)

תמיד תוכלו למצוא את החומר העדכני ביותר באתר שלי הנמצא בלינק הבא: <http://www.petri.co.il>

בהצלחה.

דניאל פטרי

**גירסה נוכחית: 2.25**  
**עודכן לאחרונה: 4 לספטמבר 2005**