המדריך לטרמינל בלינוקס למתחילים

Qazjap11

Version: 1.10

תוכן עניינים

הקדמה

המדריך נכתב ב 9 חלקים, ועל כן מוצג בפניכם ב 9 פרקים. המדריך נועד לעזור למשתמש המתחיל בלינוקס להסתגל לכלי שנקרא "טרמינל". המדריך נכתב ברוח חופש המידע והקוד הפתוח, ומובא לכם כמתנת מעבר למערכת הנפלאה הזאת, שקוראים לה לינוקס.

קריאה מהנה!
Qazjap11

רשיון

תודות

קודם כל, תודה למפתחי המערכת הנפלאה הזאת, למפתחי התוכנה הנפלאה שאני כותב בה עכשיו (openoffice), ליוצרי האינטרנט, ולכל מי שתומך בקוד פתוח. תודה מיוחדת לפורום hacking.org.il, שנתן במה לפרסום החלקים הראשוניים, וכן תודה מיוחדת לאנשים הבאים מהפורום הנ"ל, על שהעירו והוסיפו על המדריך: Xlib, emanuel1234, י.ש.,elicn. תודה מיוחדת למיכאל - Michael שעבר על המדריך.

פרק 1 - מבוא

המדריך נכתב עבור Ubuntu, אולם רוב הפקודות הם אוניברסליות, ויפעלו על כל הפצת לינוקס שהיא (וחלקם גם על מערכות מבוססות UNIX כמו מקינטוש).

מערכת הקבצים

נתחיל בכך שמערכת הקבצים של לינוקס שונה מזו של וינדוס או מערכות הפעלה אחרות, וכל מידע שנשמר על ה HD יכול להיות קובץ, תיקייה או קישור (כמובן יש עוד, אך לא נעסוק בזה כרגע).

קובץ - מסומן ע"י המספר 0 או - (שימושי ל CHMOD וכו')

תיקייה - מסומנת ע"י המספר 1 או d

קישור - הפנייה למיקום אחר במערכת, מסומנת ע"י המספר 2 או האות l

שימו לב שיש גם קישור קשיח, שההבדל בינו לקישור רגיל ("רך") הוא שהקישור הוא לא למיקום, אלא לקובץ עצמו, כלומר המיקום בכונן הקשיח.

ה HD מחולק למחיצות (בשפת הוינדוס: כוננים), אם בוינדוס קוראים להם C,D,E וכו', בשפת ה"לינוקס" קוראים להם sda1 sda2 sdb1 וכו'. הלינוקס בדר"כ (אלא אם כן שיחקתם עם המחיצות בהתקנה) נמצא על מחיצה אחת (בין אם זה כל הדיסק, לצד וינדוס, סתם מחיצה או מערכת אחרת). המיקום הכי גבוהה בעץ התיקיות נקרא Root directory, ובתוכה בעצם נמצאים כל הקבצים וכו' (שקול לכונן C). מה שבוינדוס היה כונן D וכו' בלינוקס נקרא סתם מחיצות, ואפשר "לפתוח" אותן בעת הצורך בעזרת פקודה.

בלינוקס, בשונה מוינדוס, התו המבדיל בין התיקיות הוא סלאש (/) ולא סלאש הפוך (\). הרוט dir (קיצור של directory) מסומן ב/, וכל תיקייה בתוכו נכתבת אחרי ה/. לדוגמה:

רוט דיר:

```
/
```

תיקיית home:

```
/home
```

התיקייה נמצאת בתוך ה root dir ולכן נכתבת אחרי הסלאש.

תיקיית ה **home**

לינוקס מאפשרת ריבוי משתמשים, כאשר לכל משתמש קיימת תיקייה משלו. בתיקייה נשמרים כל המסמכים, ההגדרות, הקונפיגים של המשתמש וכו'. המשתמש הינו בעל בעלות מלאה על תיקיית הבית

שלו, אך אינו יכול לשנות בדר"כ דבר מחוצה לה (ראו הרשאות בהמשך). תיקיית הבית של המשתמש נמצאת בתיקייה home שברוט דיר. לכן המיקום האבסולוטי שלה (כלומר יחסית לרוט דיר) הינו:

```
/home/username
```

כש username זה שם המשתמש כמובן.

אבל מסתבר שיש גם מיקום יחסי לתיקיית ה home! והוא ~ (טילדה). כלומר במקום לכתוב:

```
/home/username/folder
```

אפשר לכתוב רק:

```
~/folder
```

כלומר

```
~ = /home/username
```

~ נחשב למיקום יחסי, ואילו / מיקום אבסולוטי, כיוון שאפשר להחליף את ~ במיקום אבסולוטי דרך תיקיית הרוט. שימו לב שכל מיקום יחסי אפשר להביע בעזרת מיקום אבסולוטי (רוט דיר). תיקיית הבית של הרוט (המשתמש הכי "חשוב" במערכת) הינה יוצאת דופן והיא:

```
/root
```

סוגי קבצים

הקבצים בלינוקס, אינם מוגדרים ע"י הסיומות, ואלה קיימות למען האסטטיקה. בכל מקרה, סוג הקובץ נקבע ע"י תוכנו. לכן שמות הקבצים יכולים להיות גם בלי סיומות, ועם איזה סיומות שרק רוצים. בניגוד לוינדוס, שמסרב לפתוח קבצים בלי סיומת, אם קובץ בלינוקס מכיל טקסט, הוא יפתח ע"י עורך הטקסט בדר"כ, אלא אם כן הוגדר אחרת.

עוד מיקומים יחסיים

בכל תיקייה קיימים עוד 2 מיקומים יחסיים: "." ו"..". נקודה מסמלת את התיקייה הנוכחית, למשל נקודה בתיקייה folder שנמצאת ברוט דיר יהיה שווה ערך למיקום:

```
./ = /folder/
```

שימו לב שאפשר לכתוב ואפשר גם לא לכתוב סלאש (/) אחרי שם התיקייה, אך אם רוצים להוסיף עוד תיקייה או קובץ, חובה לכתוב / על מנת להפריד את השמות. לדוגמה:

```
folderfile X
foler/file V
```

שתי נקודות מסמלות את התיקייה מעל התיקייה הנוכחית בעץ ההירורכיה, כלומר התיקייה שמכילה את התקייה הנוכחית. לדוגמה אם התיקייה FOLDER נמצאת בתיקייה ETC שברוט דיר, אז:

```
../ = /etc/
```

שימו לב שבניגוד לתיקיית הבית שמשתנה ממשתמש למשתמש, מיקומים אלה משתנים מתיקייה

לתיקייה.

הרשאות

בניגוד לוינדוס וכו', בלינוקס יש מערכת הרשאות חכמה, שבין היתר מונעת ממשתמשים לעשות מה שהם רוצים בשרתים, וגם מקשה על פיתוח וירוסים וקודים זדוניים. ההרשאות שקיימות הן לקרוא את הקובץ (r), להריץ את הקובץ (x), ולערוך/למחוק את הקובץ (w). ההרשאות ניתנות ל 3 קבוצות: בעל הקובץ/תיקייה, המשתמשים שבקבוצה שהקובץ/תיקייה שייכת לה, וכל היתר.

כפי שניתן להבין, לכל קובץ יש בעלים, שכולל משתמש וקבוצה של משתמשים. לדוגמה, הבעלים של תיקיית הבית של כל משתמש, וגם הקבוצה השולטת, הם המשתמש עצמו, ואילו של כל שאר המערכת לרוב, המשתמש (והקבוצה) root.

ההרשאות ברשימה מלאה כוללות את סוג המידע (קובץ/תיקייה/קישור), הרשאות בעל הקובץ, הרשאות הקבוצה בעלת הקובץ והרשאות כל היתר. ההרשאות ברשימה מלאה מסומנות כ:

```
drwxrwxrwx
```

כאשר d אומר שזוהי תיקייה (אם זה קובץ אז יש -). ה rwx הראשונים אומרים שלבעל הקובץ יש הרשאות קריאה, כתיבה והרצה, השניים מתייחסים לקבוצה והשלישיים לכל היתר. אם אין הרשאה מסויימת, שמים - במקום.
לדוגמה:

```
-rwxr-xr--
```

אומר שזהו קובץ, שבעליו רשאי להריץ לכתוב ולקרוא, הקבוצה בעלת הקובץ רשאית לקרוא ולהריץ אך לא למחוק/לכתוב, וכל היתר רק לקרוא. לעיתים משמיטים את סוג הקובץ.
אבל, בגלל שלא תמיד נוח לכתוב ככה, אפשר להחליף כל זאת במספרים. המספרים הם בבסיס אוקטאלי, כאשר:

```
0 = ---
1 = --x
2 = -w-
3 = -wx
4 = r--
5 = r-x
```

```
6 = rw-
7 = rwx
```

וכן כמו שהוזכר קודם 0 = קובץ, 1 = תיקייה ו 2 = קישור.

לכן אם ננתח את ההרשאות הבאות: 0755

נראה כי זהו קובץ, שבעליו רשאי להריץ לקרוא, לערוך ולהריץ, ואילו הקבוצה והיתר רשאים רק לקרוא ולהריץ (r-x).

נראה כי הרשאה נפוצה לקבצים היא 0755, כשכל היתר יכולים לקרוא ולהריץ (למשל הרשאות של קבצים במערכת, כלומר לא בתיקיות הבית של המשתמש), ושל תיקיות היא 1644, כיוון שאין צורך בהרשאת הרצה, כיוון שאי אפשר להריץ תיקייה. (למרות שגם 1755 לא טעות). לרוב כותבים הרשאות מבלי להתייחס לסוג המידע, כלומר 755 או 644 וכו'.

בדר"כ משתמש הרוט יכול לקבל פריבילגיות על קבצים גם אם אין עליהם הרשאות.

הערות נוספות

הטרמינל רגיש לאותיות גדולות/קטנות, בין היתר הפקודות, מיקומים במערכת, שמות משתמשים וכו'.

הטרמינל מאפשר שליטה כמעט מוחלטת על המערכת, והוא כלי עוצמתי ללינוקס. יש לעיתים להעדיף להשתמש בטרמינל לביצוע פעולות, כיוון שהוא תמיד מהיר יותר.

פרק 2 - פקודות בסיסיות

הטרמינל

הטרמינל הוא כלי עוצמתי בלינוקס, שבעצם פעם הווה את התקשורת היחידה בין המחשב למשתמש. בניגוד לוינדוס שרוב התקשורת מבוססת על הממשק הגראפי, בלינוקס (שאגב הממשקים יותר ויותר נוטים לכיוון הזה), אפשר לשלוט בעזרת הטרמינל, כמעט על כל פרט במערכת.

הטרמינל נמצא ב Ubuntu, בדר"כ ב Applications שבפאנל למעלה, ב Accessorities. דרך נוספת להגיע אליו היא ללחוץ Alt+F2 ולכתוב gnome-terminal או xterm (למרות שלדעתי הראשון נוח יותר). והדרך האחרונה שאני אכתוב עליה, בדר"כ במצב שהממשק הגראפי עמוס מידי, היא ללחוץ Ctrl+Alt+F1/F2/F3 וכו', ולעשות login. כדי לחזור לחצו Ctrl+Alt+F7. (שימו לב שבעת כתיבת הססמא לא תראו אותה על המסך).

ניתוח התצוגה בטרמינל

בטרמינל, כמובן, מוצג רק טקסט, ואילו כל פקודה נכתבת בשורה חדשה (כמעט תמיד), ועל מנת להריץ פקודה יש ללחוץ enter. השורה שתראו בטרמינל תראה כמו זאת:

```
username@computername:~$
```

כאשר username הוא שם המשתמש שלכם, computername הוא שם המחשב שלכם, ואילו ה~ אחרי הנקודותיים מסמל את התיקייה שהטרמינל נמצא בה כרגע. סימן הדולר משמש מעין מפריד. כאשר אנחנו נמצאים במשתמש רוט, סימן הדולר יהפוך לסולמית (#).

פקודת **cd**

פקודת cd היא אחת הפקודות הבסיסיות ביותר, שקיימת גם ב UNIX, וגם בוינדוס. פקודה זו מנווטת אל התיקייה שצויינה. יש להבחין כאן במיקום יחסי ואבסולוטי. את הפקודה רושמים כך:

```
cd xxx
```

כאשר xxx זה המיקום של התיקייה אליה אנחנו רוצים לנווט. בכל מקרה, אפשר לרשום את התיקייה במיקום אבסולוטי, כלומר יחסית לרוט דיר. לדוגמה אם נרצה להכנס לתיקייה init.d שב etc שברוט דיר, נכתוב:

```
cd /etc/init.d
```

שימו לב שהשורה בטרמינל תשתנה ל:

```
username@computername:/etc/init.d$
```

מיקום יחסי יכול להיות גם יחסית לתיקיית הבית. לדוגמה, אם נרצה להכנס לתיקייה Desktop שבתיקיית הבית שלנו, נכתוב:

```
cd ~/Desktop
```

או בעזרת מיקום אבסולוטי:

```
cd /home/username/Desktop
```

שימו לב שבשני המקרים השורה בטרמינל תשתנה ל:

```
username@computername:~/Desktop$
```

עד כאן היה מיקום יחסי ששונה בין משתמשים. עכשיו נראה מה זאת אומרת מיקום יחסי בתיקיות.

נניח אנחנו נמצאים בתיקיית הבית (~), ויש בה את אותה תיקיית Desktop. אז אנחנו יכולים לכתוב:

```
cd Desktop
```

עוד דוגמה היא אם אנחנו נמצאים בתיקייה etc שב root dir, ונרצה להכנס לתיקייה init.d, נוכל לכתוב:

```
cd init.d
```

אבל שימו לב שאם אנחנו נמצאים בתיקיית הבית (ושם אין בדר"כ תיקייה כזאת), ונכתוב את הפקודה הנ"ל, נקבל שגיאה.

כמו כן לגבי חזרה לתיקייה קודמת (..) והתיקייה עצמה (.), לדוגמה אם נכתוב:

```
cd ../
```

כאשר אנחנו נמצאים בתיקייה ETC שברוט דיר, נחזור לרוט דיר עצמו, ושורת הטרמינל תהיה:

```
username@computername:/$
```

כלומר נכנסנו לרוט דיר (/).

פקודת ls

פקודת ls היא פקודה שגם קיימת ב UNIX, וכמובן בלינוקס ובמאק, והפלט שלה הוא בעצם רשימת הקבצים בתייקיה. יש כמה מקרים של שימוש בפקודה. אם נרשום פשוט ls, אזי הפלט יהיה רשימת הקבצים והתיקיות (שאגב צבועים בצבעים שונים) בתיקייה הנוכחית. כלומר אם עכשיו נפתח את הטרמינל (שנפתח בתיקיית הבית) ונרשום:

```
ls
```

נקבל רשימה של הקבצים והתיקיות בתיקיית הבית.

הוריאציה השנייה לשימוש, היא ציון התיקייה שאנחנו רוצים להציג את התוכן שלה, כמובן לפי חוקי היחסיות בדומה ל cd. לדוגמה, אם נרצה לראות את התוכן של התיקייה etc מבלי להכנס אליה (ויש הרבה...), נכתוב:

```
ls /etc
```

על מנת להבין את הוריאציה השלישית, יש צורך בלהבין כמה דברים. הדבר הראשון הוא פרמטרים לפקודה/תוכנית. בלינוקס אפשר להעביר פרמטרים קצרים, כלומר מקף ואות (לדוגמה a- b- וכו'), בלי ערכים, או עם ערכים שבאים אחריהם, או בצמוד. לדוגמה:

```
-a12
-a 12
```

ניתן גם להעביר פרמטרים ארוכים, שנכתבים אחרי 2 מקפים (לדוגמה try-- help--), עם או בלי ערכים, שנכתבים לאחריהם או אחרי סימן שווה (=). לדוגמה:

```
--file xxx --mode=yyy
```

ולסיכום, ככה זה נראה ביחד עם הפקודה:

```
command -s -y 12 -z14 --help --file xxx --mode=yyy
```

כש command זה שם התוכנית או הפקודה, s- הוא פרמטר בלי ערך, הערך של הפרמטר הקצר y הוא 12, ושל z הוא 14. help הוא פרמטר ארוך בלי ערך, הערך של הפרמטר הארוך file הוא xxx ושל mode הוא yyy. כרגע לא אכפת לנו איך התוכנית/פקודה מעבדת את כל זה, אנחנו רק צריכים לדעת איך להזין אותם. ושוב אני מזכיר שהטרמינל רגיש לאותיות גדולות/קטנות, גם בפרמטרים.

הדבר השני שצריך להבין, זה שישנם תיקיות וקבצים מוסתרים, שלא מוצגים ב ls רגיל (וגם לא בממשק הגראפי בדר"כ, אלא אם כן לוחצים Ctrl+H לדוגמה, בגנום), והם מתחילים בנקודה. לדוגמה הקובץ bash_history. מכיל 2000 (משתנה מהפצה להפצה) פקודות אחרונות שהזנו בטרמינל. קובץ זה לא שמיש במיוחד, ולכן אם נראה אותו כל הזמן, הוא רק יפריע. לכן הוא קובץ נסתר.

אז נחזור לענייננו. איך אנחנו נצגי קבצים נסתרים? נכתוב את הפקודה ls עם הפרמטר הקצר a-. דוגמאות:

```
ls -a
ls -a /etc
ls /etc -a
```

הדוגמה הראשונה תביא רשימה של כל הקבצים, כולל הנסתרים בתיקייה הנוכחית (תיקיית הבית אם רק פתחתם את הטרמינל), ושתי הפקודות האחרות יביאו רשימה של כל הקבצים והתיקיות כולל הנסתרים בתיקייה etc שברוט דיר. יש גם לרוב פקודה מקוצרת (שנקראת alias, כלומר קיצור) לפקודה ls -a והיא la. הפקודה עושה אותו דבר, רק אין צורך בפלאג (פרמטר).

פקודת mkdir

הפקודה הזאת בעצם יוצרת תיקייה. את התיקייה אפשר לכתוב במיקום יחסי ובמיקום אבסולוטי,

בדומה למה שלמדנו מקודם. לדוגמה אם נכתוב את הפקודה הבאה כשאנחנו בתיקיית הבית:

```
mkdir newfolder
```

ניצור תיקייה בשם newfolder בתיקיית הבית. עם זאת אנחנו יכולים ליצור תיקייה במיקום אבסולוטי, לדוגמה, אם נרצה ליצור תיקייה בשם newfolder ב etc שברוט דיר, נכתוב:

```
mkdir /etc/newfolder
```

שימו לב שאתם צריכים להיות בעלי גישות כתיבה בתיקייה etc, על מנת ליצור בתוכה תיקייה.

פקודת **rmdir**

פקודה זו מוחקת תיקייה, בדומה לפקודה הקודמת. כמובן שיש צורך בהרשאות מתאימות. יתרה מכך, הפקודה תסרב למחוק תיקייה שלא ריקה. דוגמה:

```
rmdir folder
```

תנסה למחוק את התיקייה folder שבתיקייה הנוכחית.

פקודה **touch**

הפקודה הזאת, היא הפשוטה ביותר ליצירת קובץ ריק, לפי אותם חוקי יחסיות שלמדנו קודם. כמו כן היא משמשת לעריכת זמן יצירת הקובץ (שימושית לעובדים שרק עכשיו נזכרו במסמך שהבוס ביקש לעשות אתמול). דוגמה:

```
touch file
```

תיצור קובץ ריק בתיקייה הנוכחית. שימו לב להרשאות כמובן.

פקודת **rm**

הפקודה הזאת מוחקת קובץ, ואפילו תיקיות שלמות על כל הקבצים שיש בהם. הוריאציה הראשונה כמובן היא למחוק קובץ. לדוגמה יש לי קובץ בשם xxx ונרצה למחוק אותו (בתנאי שכמובן אנחנו נמצאים בתיקייה שלו) נכתוב:

```
rm xxx
```

הוריאציה השנייה היא למחוק תיקיות על כל תוכנן, ע"י שימוש בפלאג (פרמטר קצר) r-. הפרמטר הזה מציין שאנחנו רוצים למחוק את התיקייה על כל תוכנה. לדוגמה אם אנחנו רוצים למחוק תיקייה בשם folder שמכילה הרבה קבצים ותקיות, נרשום:

```
rm -r folder
```

שימו לב שצריך הרשאות מתאימות, ושכל מה שנמחק, קשה מאוד לשחזר, לכן כדאי מאוד להזהר עם פקודה כזאת (פעם מחקתי ככה בטעות 40GB של סדרה, ואחרי שהורדתי אותה שוב קיבלתי באן מ TL).

הפקודה **cat**

הפקודה הזאת מציגה את התוכן של קובץ, והשימוש בה כמובן תואם לחוקי היחסיות כמו שראינו מקודם. לדוגמה אם נרצה לראות את תוכן הקובץ xxx, נכתוב:

```
cat xxx
```

file

הפקודה תדפיס את סוג הקובץ, לדוגמה:

```
file file.py
```

ידפיס "a python script text executable".

הערות אחרונות

כמעט לכל פקודה יש עזרה. לכן עוד לפני שהולכים לחפש בגוגל, נסו לכתוב אחת מהפקודות הבאות, על מנת להגיע לעזרה:

```
man command
command -h
command --help
command -?
```

כל הפקודות שהזכרתי קיימות ב UNIX.

ושוב אני מדגיש את **man command** חברו הטוב ביותר של המשתמש המתחיל.

פרק 3 - פקודות נוספות

עד עכשיו עסקנו בפקודות, עכשיו אני אראה לכם כמה תוכנות.

which

לפני הכל, נראה איך בעצם הטרמינל מגיע לפקודות הרצויות. הפקודות או התוכנות, כאשר אנחנו כותבים את השם שלהן, המערכת מחפשת אותם בדירים הבאים (הסדר ודירים נוספים נקבעים במשתנה $PATH)

```
/bin
/sbin
/usr/bin
/usr/sbin
```

כמובן יש עוד ויש אליאסים, אבל זה לא משנה כרגע. פקודה שימושית למצוא את מיקום הפקודה היא which. לדוגמה, אם נרצה לדעת איזה קובץ מורץ כאשר אנו כותבים ls, נכתוב:

```
which ls
```

ונקבל:

```
/bin/ls
```

שימו לב שרוב הפקודות שלמדנו קודם נמצאות בדיר bin. יוצאת דופן היא cd, שבעצם לא ממש קובץ שמורץ, אלא ממש פקודה בטרמינל. דרך נוספת להריץ קובץ בטרמינל, היא לכתוב את המיקום שלו.
לדוגמה אם נרצה להריץ את אותה הפקודה ls, נוכל לכתוב

```
/bin/ls /etc
```

ותתקבל אותה התוצאה - רשימת הקבצים בתיקייה etc.

nano

התוכנה האמיתית באמת שנלמד עליה היום היא nano. ננו היא תוכנה שפותחה בדומה ל pico ב UNIX, שלא הייתה בקוד פתוח. התוכנה היא עורך טקסטים טרמינלי, שמאוד שימושית כשרוצים לערוך קובץ מהטרמינל. ישנן 2 דרכים להשתמש בתוכנה. הראשונה היא לפתוח אותה ע"י הפקודה:

```
nano
```

לכתוב מה שצריך ולשמור בשם מסויים (כמובן צריך הרשאות). הדרך השנייה היא לפתוח קובץ קיים, או ליצור קובץ חדש בעזרת שם הקובץ, לדוגמה הפקודה:

```
nano xxx
```

תערוך את הקובץ xxx אם הוא קיים, ואילו תיצור אותו בעת השמירה, אם לא.
כאשר פותחים את עורך הטקסט הטרמינלי, מגלים מסך שונה מהטרמינל הרגיל, כלומר התוכנה "לקחה

שליטה" על הטרמינל שלנו. בתוכנה יש חוקים שונים לגמרי. נראה כמה מהם.

כמובן שהחצים מנווטים במסמך, ואילו Page down ו Page up מספקים ניווט מהיר יותר. כאשר כותבים משהו, השינויים נכנסים למסמך, ונכתבים כאשר שומרים את הקובץ. אחת הפקודות השימושיות ביותר היא פקודת היציאה (איך לא). על מנת לצאת מהעורך, יש ללחוץ Ctrl+X. אם לא שמרתם את השינויים, העורך ישאל אם אתם רוצים לשמור, אם כן שמרתם או לא נגעתם, הוא פשוט יצא, ותחזרו לחלון הטרמינל המוכר לנו.

פקודה נוספת שמאוד שכיחה, היא פקודת השמירה. על מנת לשמור לחצו Ctrl+O, הכניסו את שם הקובץ שלתוכו ברצונכם לשמור, ולחצו enter. אם אין הרשאות כתיבה, הקובץ יפתח כרגיל ותוכלו לשנות אותו, אך בעת ניסיון השמירה תתקבל שגיאה. Ctrl+W יציג שורת חיפוש, וכאשר נכניס את המילה ונלחץ enter, העורך יעבור למופע הראשון של הביטוי במסמך. Ctrl+K יגזור את השורה הנוכחית, Ctrl+U ידביק. לחצו Ctrl+G לעוד עזרה.

vi

וי הוא גם עורך מסמכים טרמינלי. בניגוד ל nano, כדאי לכתוב את השם של הקובץ, כיוון שהוא נכנס ל help קטן. בכל מקרה הנה דוגמה לשימוש בפקודה:

```
vi xxx
```

עכשיו פה הפקודות שונות מ nano. לדוגמה בשביל לצאת, צריך ללחוץ esc, אחר כך לכתוב q: (נקודותיים q). בשביל לשמור צריך לכתוב w: (אחרי שלחצנו esc), ובשביל לצאת בלי לשמור צריך ללחוץ !q: (נקודותיים q וסימן קריאה), וכמובן h: בשביל עזרה.

עוד כמה טיפים לגבי הטרמינל

אז הנה עוד כמה דברים על הטרמינל. אם תלחצו על החץ למעלה ולמטה, תופיע לכם רשימת פקודות אחרונות (השמורות ב bash_history. שהזכרתי מקודם). על מנת לראות מה היה כתוב לפני שזה זלג מהמסך, לחצו Shift+Page up/Page down (שימושי כשאין גלגלת, כלומר לא בגנום טרמינל). אם השורה של הסטטוס חצי מחוקה כי איזה מתכנת שכח לכתוב תניו ליין צ'אר והשורה דרסה את השורה הנוכחית (או הרצתם cat על קובץ ריצה או תמונה), לחצו enter, והשורה תופיע מחדש בשורה חדשה, בריאה ושלמה. על מנת להעתיק משהו (וזאת אופצייה שמורה ב gnome-terminal) נלחץ Ctrl+Shift+C, ולהדביק נלחץ Ctrl+Shift+V. על מנת לפתוח חלון חדש נלחץ Ctrl+SHIFT+T, על מנת לסגור נלחץ Ctrl+SHIFT+W, ועל מנת לצאת מכל החלונות נלחץ Ctrl+Shift+Q.

ונחזור לאופציות שרצות בכל טרמינל - פקודת יום הדין: אם בטעות הרצתם מחיקה על קבצים חשובים,

יש לכם סיכוי לא קטן לעצור אותה (או כל תוכנה/פקודה אחרת) באמצע! פשוט לחצו Ctrl+C!

בטרמינל בלינוקס יש עוד אופציה נורא שימושית והיא השלמה אוטומטית. לדוגמה, אם נכתוב את הפקודה

```
rmdi
```

(בלי ה r) ונלחץ על Tab, הוא ישלים אותה אוטומטית לפקודה rmdir. אך אם נכנס את המילה

```
mk
```

ונלחץ על Tab, לא יקרה כלום. למה? כי יש יותר מאופציה אחת. על מנת שהטרמינל יציע אותם, נלחץ על Tab פעמיים. אם יש יותר מידי אופציות, כמו במקרה של האות l, הוא ישאל אם להציג את כולם. נסו ללחוץ פעמיים טאב סתם ככה בלי להכניס כלום, והוא ישאל אם להציג לכם את כמעט 3000 (תלוי בהפצה) הפקודות שהוא מכיר.

אותו עקרון פועל גם על קבצים (וגם על חבילות - כבר לא צריך לזכור את השמות המדוייקים), אם נכתוב:

```
cd /et
```

ונלחץ Tab, הוא ישלים ל etc. אותו דבר כאשר יש יותר מאפשרות אחת יש ללחוץ פעמיים Tab, וגם כן הוא ישאל אם יש יותר מידי אפשרויות.

cp

הפקודה מעתיקה קבצים או תיקיות. קודם נראה שימוש של העתקת קובץ-קובץ:

```
cp source_file dest_file
```

הקובץ יעתיק את התוכן של source_file לתוך הקובץ dest_file. שימו לב שאם יש קובץ בשם הזה, הוא ידרס, וכמובן יש צורך בהרשאות מתאימות. השימוש השני זה העתקת תיקייה-תיקייה:

```
cp souce_folder dest_folder -r
```

שימו לב לפלאג r- שאומר שהפקודה תעתיק גם את התוכן של התיקיות. שימו לב שאפשר גם להשתמש במיקום אבסולוטי, יחסית לתיקיית הבית וכו'. לדוגמה:

```
cp .bashrc /etc/bashrc
```

יעתיק את הקובץ bashrc. במקום הקובץ bashrc שב etc. כמובן יש צורך בהרשאות מתאימות, ואל תנסו

את זה בבית, חבל סתם תהרסו משהו.

בכל מקרה הדרך השלישית זה להשתמש בקובץ-תיקייה. מה זאת אומרת? אם אני כותב:

```
cp xxx yyy
```

ו xxx הוא קובץ, ו yyy הוא תיקייה, אז הוא יעתיק את xxx לתוך התיקייה yyy וישמור באותו השם.

mv

הפקודה הזאת כמו cp, רק שמעבירה את הקובץ, בדומה ל"גזור". בכל מקרה, רק דבר אחד כדאי לדעת, וזה שכשמעבירים קובץ באותה מחיצה (כונן, דיסק וכו'), אז לא משנה כמה הוא שוקל, הוא יעבור בשנייה, בשונה מ cp, כי פה בעצם רק הנתיב של הקובץ משתנה והוא לא מועתק מה HD. כמו כן ניתן להשתמש בה לשינוי שם, ע"י העברת הקובץ הישן לחדש.

ln

הפקודה הזאת יוצרת לינקים. יש צורך להבדיל ב 2 סוגי לינקים. לינקים קשיחים וסימבולים. לינקים קשיחים מורים על אותו מיקום ב HD, ואם נמחק אחד, לא יקרה כלום לקובץ, כל עוד יש עוד לינקים קשיחים. לינק סימבולי מורה על הקובץ/תיקייה, ונמצא במקום אחר על ה HD (זוכרים בהרשאות את ה l?), ואם נמחק את קובץ המקור, הלינק לא יהיה שווה כלום. על מנת ליצור לינק קשיח נכתוב:

```
ln file_or_folder link
```

ולינק סימבולי:

```
ln -s file_or_folder link
```

כמובן, כמו cp ו mv, הוא כפוף לכללי היחסיות של מערכת הקבצים.

פרק 4 - פקודות מתקדמות

משתנים

למה שבטרמינל יהיו משתנים? מסתבר שהטרמינל מכיל בתוכו בסתר, שפה שנקראת באש (BASH) שאפשר להגיד שהיא בצורה כלשהיא בנוייה בתוך הטרמינל. שפת הבאש היא שפת הסקריפט (הכמעט) רשמית של לינוקס, והיא שיפור של ה SHELL שהייתה עוד ב UNIX. למעוניינים להרחיב ידע בנושא, אציע את הספר "Beginning Portable Shell Scripting".

משתנים בבאש, כמו בפרל וב PHP, מתחילים ב$. למרות שכאשר מגדירים אותם זה לא ככה, זה לא אמור לעניין אותנו כרגע. מה שכן מעניין אותנו זה משתנים מוגדרים כגון:

$USER
$PATH

וזהו, אני אחזור אליהם בהמשך.

echo

זו הפקודה המדפיסה את מה שביקשנו ממנה. נבחן כמה וריאציות. הפשוטה ביותר היא פשוט לכתוב מה אנחנו רוצים להדפיס. לדוגמה:

echo hi

ידפיס hi. דרך נוחה יותר לרשום, שמונעת הרבה שגיאות היא להשתמש במרכאות:

echo "buenos dias"

אם נרצה לכתוב בכמה שורות, נשתמש בתו סלאש הפוך (\) בסוף כל שורה שלא אחרונה, לדוגמה:

*echo salam *
alekum

שימו לב שזה ידפיס בשורה אחת בכל זאת. אם נרצה לעשות שורה חדשה, נשתמש במרכאות, נרשום על כמה שורות ונסגור אותם. לדוגמה:

echo "Hi
sap?
really?"

ידפיס את זה ב 3 שורות. שימו לב שבשני המקרים האחרונים, שורת הסטטוס השתנתה ל>.

עוד שימוש ל echo, הוא להדפיס משתנים. כן, אותם משתנים שדיברתי עליהם קודם. לדוגמה, אם נרצה להדפיס את המשתנה $USER, ניתן להעזר ב 2 דרכים:

echo $USER

echo "$USER"

הדרך השנייה מומלצת יותר. שימו לב שהאותיות של המשתנה הזה צריכות להיות גדולות. הפלט יהיה שם המשתמש שלכם. שימו לב כי:

echo '$USER'

עם גרש אחד, ידפיס את הביטוי $USER ולא את תוכן המשתנה. דרך נוספת לעשות זאת היא להוסיף סלאש הפוך (\) לפני הדולר ($):

echo \$USER
echo "\$USER"

נראה עוד כמה משתנים:

$SHELL מכיל את השאל שאנחנו משתמשים בו (בדר"כ באש).

$HOME מכיל את מיקום תיקיית הבית שלנו

$PATH מכיל את הדירים שבהם המערכת תחפש פקודות לפי הסדר

$ PS1 מכיל את שורת הסטטוס שלנו

כמובן יש עוד, זאת רק טעימה.

grep

הפקודה הזאת מאוד שימושית. היא מאתרת מילה או יותר נכון ביטוי רגולרי, בתוך קובץ. נראה כמה שימושים. השימוש הפשוט זה לכתוב מילה ולחפש אותה בקובץ. נעשה כך:

grep root /var/log/syslog

הפלט יהיו השורות שמכילות את המילה (או חלק ממילה) root, כאשר היא מודגשת באדום. זה היה השימוש הפשוט של מילה. נראה מה הוא ביטוי רגולרי. ביטוי רגולרי הוא תבנית שמתאימה לביטויים כלשהם. הביטויים הרגולרים בהם משתמש grep הם מסוג Basic Regular Expressions.

נקודה מסמנת כל תו שהוא (.).

קבוצה של אותיות בסוגריים מרובעים מסמלות תו מבין אלה שנמצאים שם ([abc]).

סוגריים שלפניהם סלאש הפוך (\) מסמלות תת קבוצה. ((\something\))

דוגמאות:

הביטוי הרגולרי:

ab.de

מתאים למחרוזת abcde, ולמחרוזת ab#de אך לא למחרוזת abcd.

הביטוי הרגולרי:

a[bc]de

יתאים למחרוזת abde וגם למחרוזת acde, אך לא למחרוזת abcde (כיוון שקבוצה בסוגריים מרובעים מסמלת תו אחד).

הביטוי הרגולרי:

a[a-c0-6]b

יתאים למחרוזות a0b acb aab a1b abb אך לא למחרוזת a-b.

שימו לב שמקף (-) בקבוצה מסמל מ...עד.

כוכבית (*) אומרת אפס תווים או יותר.

\{x\} אומר בדיוק x פעמים

\{x,\} אומר לפחות y פעמים

\{x,y\} אומר בין x ל y פעמים כולל

דוגמאות:

הביטוי הרגולרי:

*[a-d]**

יתאים למחרוזת aaa וגם ל abbddd וגם למחרוזת ריקה (0 פעמים), אך לא ל abcde.

הביטוי הרגולרי:

abc\{2\}

יתאים רק למחרוזת abcc

הביטוי הרגולרי:

ab[a-b]\{2,\}

יתאים למחרוזת abaa ולמחרוזת abababab אך לא למחרוזת aba. (לפחות פעמיים אחת מהאותיות a-b).

הביטוי הרגולרי

ab.\{1,2\}

יתאים למחרוזות abx2 ab12 ab1 אך לא למחרוזת ab111.

^ מסמן שהביטוי צריך להופיע בתחילת שורה

$ מסמן שהביטוי צריך להופיע בסוף שורה

$....^ מסמן שהשורה צריכה להיות בדיוק מה שרשום בין ה^ וה$.

אז אחרי שלמדנו קצת על ביטויים רגולריים, נראה איך אפשר להשתמש בהם ב grep. לדוגמה, נחפש בקובץ שורות שנגמרות אות+oot. נכתוב:

grep .oot$ /var/log/syslog

רידיירקטים

יש אפשרות בטרמינל, לכתוב פלט של פקודה לקובץ, או לקרוא קלט של פקודה מתוך קובץ. < מסמן כתיבת פלט לתוך קובץ. שימו לב שהפקודה תמחק את הקובץ הקיים, ושיש צורך בהרשאות מתאימות. לדגומה, נכתוב את הפלט של ls לתוך קובץ, ונקרא אותו:

ls /etc > file
cat file

בקובץ file יש את רשימת הקבצים והתיקיות שבתיקייה etc. << מסמן הוספה של הפלט לתוך קובץ קיים, בלי למחוק את התוכן הקודם. לדוגמה אם נרצה לכתוב לתוך קובץ 2 שורות, ניתן לעשות:

echo "Hi" > file
echo "Sap?" >> file
cat file

נראה שאכן בקובץ מופיעות 2 השורות שלנו.

> מסמן קבלת קלט מתוך קובץ. לדוגמה, אם נרצה להשתמש grep, ולייבא את התוכן מתוך קובץ (למרות שזה חסר כל הגיון), נעשה:

grep root < /var/log/syslog

כלומר הוא יחפש את המילה root בקלט שיקרא מתוך הקובץ הנתון.

דבר מאוד שימושי נוסף הוא פייפים. פייפ "מוסר" פלט של פקודה אחת כקלט של פקודה שנייה, תמיד משמאל לימין. על המסך יוצג הפלט רק של הפקודה האחרונה. פייפ נכתב כקו (|).

דוגמה:

ls /etc | grep ^h

את הפלט של הפקודה הראשונה, כלומר רשימת הקבצים בתיקייה etc, הטרמינל בלי להציג יעביר ל grep, שמיצידה תסנן ותציג רק את התיקיות והקבצים שמתחילים באות h.
דוגמה נוספת לשימוש מורכב יותר:

echo $HOME | ls | grep ^w > xxx

הפלט של ה echo, כלומר תיקיית הבית שלנו יועבד כפרמטר ל ls. הפלט של ls, כלומר רשימת הקבצים בתיקיית הבית, יועבר כפרמטר לפקודה grep, ואילו הפלט של grep, שהוא התיקיות והקבצים שמתחילים באות w, ירשם בקובץ xxx. בסופו של דבר, שום דבר לא יוצג על המסך.

sudo

הפקודה הזאת מאפשרת לשלוט על המערכת כאילו שהיית רוט, והיא בעצם הצלה מלהכנס כל פעם לרוט כשצריך משהו. את הפקודה כותבים לפני כל פקודה אחרת שאתם רוצים להריץ כ root, ולאחריה אתם תצטרכו לכתוב את הססמא שלכם (רק פעם ב...), שאותה לא תראו כי ככה בנוייה המערכת... בכל מקרה הנה דוגמה לשימוש:

sudo nano /etc/hosts

יתן לך לערוך את קובץ ה hosts, גם כשאתה נמצא במשתמש שלך, כאילו שהיית root. בכל מקרה שימו לב שעל מנת להשתמש בפקודה, יש להיות Superuser, כלומר משתמש עם פריוילגיות.

על מנת לא לכתוב כל פעם sudo לפני כל פקודה, אפשר להכנס לרוט דרך הטרמינל. לשם כך יש לכתוב su, ולהזין את הססמא של הרוט. דרך נוספת היא כתיבת sudo bash, ובזו אני משתמש כי התרגלתי.

אז לסיכום, אל תשכחו להשתמש ב sudo לפני כל פקודה שמצריכה פריוילגיות של רוט.

reboot

הפקודה מאוד קצרה. פשוט כותבים ומריצים, הפקודה מאתחלת את המחשב מחדש. שימו לב שבשביל להפעיל אותה, יש צורך בפריוילגיות רוט, לכן יש צורך לכתוב:

sudo reboot

אם אתם נמצאים במשתמש הרגיל שלכם שלא רוט.

shutdown

הפקודה הזאת מכבה את המערכת. על מנת לכבות גם את המחשב, יש להשתמש בפלאג P-. גם לפקודה

זו דרושים פריוילגיות רוט, לכן על מנת לכבות את המחשב נכתוב:

sudo shutdown -P now

צריך לשים השהייה בכיבוי. זאת הסיבה שצריך לכתוב NOW כדי לכבות מיידית. אם נכתוב x+,
המחשב יתכבה אחרי x דקות, ואם נכתוב שעה, המחשב יתכבה בה. לדוגמה:

sudo shutdown -P +5

המחשב יתכבה תוך 5 דק'. אם נכתוב:

sudo shutdown -P 17:31

המחשב יתכבה ב 17:31

exit

והפקודה האחרונה. הפקודה יוצאת מהטרמינל, ואם השתמשתם sudo bash או ב su, תצא קודם מהם.
השימוש פשוט:

exit

פרק 5 - פקודות שליטה במערכת

adduser

הפקודה הזאת יוצרת משתמש חדש במערכת. הפקודה בין היתר, תיצור את תיקיית הבית, תיצור את המשתמש וקבוצה באותו השם. דוגמה:

```
adduser xxx
```

הכניסו כמה פרטים כמו סמא וזה, והנה יש לכם משתמש חדש.

שימו לב, צריך להיות בעל הרשאות, לכן אם אנחנו במשתמש הרגיל שלנו נזין:

```
sudo adduser xxx
```

שימו לב, המשתמש אינו בעל פריוילגיות יתר, והפקודה sudo לא תעבוד לו.

שימוש נוסף:

```
sudo adduser xxx groupxxx
```

הקוד יוסיף את המשתמש xxx לתוך הקבוצה groupxxx.

addgroup

הפקודה יוצרת קבוצה ע"י:

```
sudo addgroup groupxxx
```

su

כבר ראינו שאפשר להכנס לרוט בעזרת הפקודה, אך אפשר להכנס גם למשתמשים אחרים. לדוגמה אם נרצה להכנס למשתמש xxx שיצרנו, נרשום:

```
su xxx
```

ונכניס את הסמא שלו.

שימו לב שכאשר נכנסים לרוט, הסימן $ בשורת הסטטוס משתנה ל#.

deluser

אפשר להשתמש בפקודה בשני שימושים:

למחוק את המשתמש:

```
sudo deluser xxx
```

שימו לב שהפקודה הנ"ל לא תמחק את תיקיית הבית שלו. על מנת לעשות כך נרשום:

```
sudo deluser xxx --remove-home
```

השימוש השני הוא למחוק משתמש מתוך קבוצה:

```
sudo deluser xxx groupxxx
```

תמחק את המשתמש xxx מתוך הקבוצה groupxxx.

delgroup

הפקודה מוחקת קבוצה. לדוגמה אם נרצה למחוק את הקבוצה groupxxx נרשום:

```
sudo delgroup groupxxx
```

אל תשכחו שכל הפקודות הנ"ל (מלבד su) מצריכות הרשאות root, או sudo.

passwd

הפקודה הזאת משנה את הססמא של המשתמש. יש שתי אפשרויות לשימוש:

הפקודה:

```
passwd
```

תשנה את הססמא של <u>המשתמש עצמו</u>, או יותר נכון של המשתמש שלתוכו נכנסנו כאשר אנחנו מזינים את הפקודה.

הפקודה:

```
sudo passwd user
```

תשנה את הססמא של המשתמש הרצוי (שימו לב, יש צורך בפריוילגיות sudo או root).

whoami

הפקודה תדפיס את שם המשתמש שאיתו כרגע אנחנו עובדים בטרמינל:

```
whoami
```

פשוט וקל

chmod

הפקודה משנה את ההרשאות של קובץ תיקייה וכו'. את ההרשאות למדנו מקודם עוד בחלק הראשון. אז רק דוגמאות לשימוש:

```
chmod drwxr-xr-- folder
chmod 0755 file
sudo chmod 775 file
```

שימו לב, שאם הקובץ לא שלכם, יש צורך להשתמש ב sudo.

chown

זוכרים שסיפרתי על ההרשאות לבעל הקובץ, הקבוצה וזה? אז זאת הפקודה שבדיוק קובעת מי הבעלים של הקובץ. דוגמה לשימוש:

```
chown xxx:groupxxx file
```

ישנה את הבעלים ל xxx ואת הקבוצה בעלת הקובץ ל groupxxx. שימו לב, אם אין לכם הרשאות, יש להשתמש ב sudo. אל תשכחו לרשום את המשתמש והקבוצה בגודל האותיות הנכון (אותיות גדולות/קטנות).

תהליכים

סוף סוף הגענו למשהוא משמעותי. בלינוקס כל תהליך/תוכנה, יש לו ID. ה ID בעצם עושה סדר במערכת. כל תהליך "נמצא" בתיקייה PROC שברוט דיר.

ps

הפקודה הזאת מאוד שימושית והיא מראה את התהליכים המופעלים במערכת.

נראה כמה שימושים:

```
ps -A
```

יראה את כל התהליכים הרצים.

```
ps -AH
```

יראה את כל התהליכים הרצים בהירורכיה.

הפקודה pstree תראה את כל התהליכים בצורה קצת יותר נוחה, גם כן בהירורכיה:

```
pstree
```

יש עוד הרבה פרמטרים, עליהם תוכלו לקרוא ב:

```
ps --help
```

top

הפקודה מראה את התהליכים שרצים בצורה נוחה, וכן מספקת מידע על המשאבים שכל פרוצס לוקח.

על מנת לצאת ממנו, יש ללחוץ q

```
top
```

pidof

הפקודה מחזירה את ה ID של פרוסס לפי השם. לדוגמה אם נרשום:

```
pidof firefox-bin
```

נקבל את ה ID של הפרוצס של פיירפוקס.

שימו לב שגם כאן באה לעזרתכם ההשלמה האוטומטית (טאב אם יש אפשרות אחת, פעמיים טאב אם

יש כמה).

kill

הפקודה שולחת אות לפרוצס ע"פ ה ID שלו, האומר לו להסתיים. לרוב הפרוצס יסגר ללא השהייה.
לדוגמה, אם קיבלנו שה ID של ה FF (פיירפוקס) היה 15651, נכתוב:

```
kill 15651
```

וה FF יעלם. כמובן שאם נרצה לעשות זאת לפרוצסים שלא שלנו, נשתמש ב sudo.
על מנת להרוג פרוצס ללא השעיה וללא תלות בפרוצס, נשתמש בפרמט 9-:

```
kill -9 15651
```

killall

הפקודה זהה לפקודה הנ"ל, רק שהיא פועלת ע"פ שם הפרוצס. הסיבה שיש all, היא שאם יהיו כמה פרוצסים עם אותו השם, אז היא תהרוג את כולם. דוגמה לשימוש:

```
killall firefox-bin
```

יסגור את ה FF. כמובן שאם נרצה לעשות זאת לפרוצסים שלא שלנו, נשתמש ב sudo.
על מנת להרוג פרוצס ללא השעיה וללא תלות בפרוצס, נשתמש בפרמט 9-:

```
killall -9 firefox-bin
```

אם קרה שנתקע לכם חלון, ואתם לא יודעים איזה פרוצס זה, לחצו Alt+F2, כתבו xkill, לחצו enter ולחצו על החלון. הוא יסגר מיד.

פרק 6 - ניהול חבילות באובונטו

מערכת חבילות ב **ubuntu** והפצות מבוססות **debian**

בהפצות אלה החבילות מבוססות על קבצים מסוג deb. חבילות אלה יכולות להיות מותקנות בכל ההפצות שיש להן מנהל חבילות מתאים. ישנו עוד סוג של חבילות: rpm. על חבילות אלה מתבססות הפצות כמו red hat וההפצות המבוססות עליה. מנהל חבילות ה DEB באובונטו, האחראי על הורדתם, התקנתם, מחיקתם, עדכונם ועוד הוא apt-get, שנעזר בתורו ב dpkg.

apt-get install

הפקודה הזאת מורידה ומתקינה חבילות. חבילה, יכולה להיות תוכנה, ספריה שתוכנה נעזרת בה וכו'. אפשר לכתוב מספר חבילות עם רווח ביניהם, ומנהל החבילות ינסה להוריד ולהתקין את כולם. מספר החבילות תלוי במה שההפצה שלכם מספקת לכם. ב UBUNTU הוא מאוד גדול. הפקודה אף תדע להוריד את הגרסה האחרונה ביותר, אם מותקנת לכם גרסה ישנה יותר, ולהתקין אותה. לדוגמה, אם נרצה להתקין pidgin (קליינט מסרים מידיים), שלא מותקן בברירת המחדל ב UBUNTU 10.10, נכתוב:

```
sudo apt-get install pidgin
```

שימו לב לכמה דברים. ראשית, יש צורך בהרשאות רוט. שנית, מנהל החבילות ידע להציע לכם להוריד את כל החבילות הדרושות להפעלת החבילה שאתם רוצים, כדוגמת libpurple, שהיא ספרייה שהחבילה pidgin תלוייה בה. לדבר הזה קוראים dependecy, כלומר שחבילה אחת תלוייה בנוכחות חבילה אחרת.

אבל אם לדוגמה נרצה להוריד את הגרסה החדשה של FF, כאשר הוא כבר מותקן, עדיין נרשום:

```
sudo apt-get install firefox
```

ומנהל ההורדות יהיה חכם מספיק להוריד את הגרסה האחרונה ולהתקינה. הנה דוגמה להתקנת כמה חבילות:

```
sudo apt-get install pidgin firefox sudo
```

שימו לב, גם sudo היא חבילה.

גם בחבילות באה לעזרתינו ההשלמה האוטומטית (טאב, פעמיים טאב), שבלעדיה כנראה שהיינו טובעים בים החבילות.

apt-get remove

הפקודה הזאת, מוחקת חבילות. לדוגמה אם נרצה למחוק את החבילה pidgin, נכתוב:

```
sudo apt-get remove pidgin
```

אם ננסה למחוק חבילה שחבילה אחרת תלוייה בה, כמו בדוגמה הזאת libpurple0, מנהל החבילות ימחק את כל החבילות התלויות בה גם. (זהו פלוס אדיר ללינוקס כנגד וינדוס, מחיקת תוכנה כמעט בלי להשאיר עקבות)

apt-get purge

הפקודה הזאת גם כן מוחקת חבילות. אולם ההבדל בין הפקודה הזאת לפקודה הקודמת, הוא שהיא כמעט ואינה משאירה אחריה זכר לחבילות, כלומר קבצי קונפיג, הגדרות וכו'. לכן אם לדוגמה נמחק את pidgin עם purge:

```
sudo apt-get purge pidgin
```

בפעם הבאה שנתקין אותו, המשתמשים שהגדרנו לא יהיו שם עוד.

apt-get update

הפקודה הזאת, אינה מעדכנת את החבילות במערכת, אלא רק את המידע עליהם. הפקודה מורידה את המידע מתוך שרתים, דבר הנקרא repositorities, או מקורות. מנהל החבילות מוריד מאותם מקורות את המידע על החבילות, משווה אותם לחבילות הקיימות, ומציע לעדכן חבילות.

```
sudo apt-get update
```

apt-get upgrade

הפקודה הזאת משלימה את הפקודה הקודמת, ומעדכנת את החבילות שיש להם עדכון. זכרו להריץ את הפקודה הקודמת בטרם מריצים את העדכון, אחרת איך המערכת תדע אילו חבילות לעדכן?

```
sudo apt-get upgrade
```

apt-get autoremove

הפקודה הזאת מוחקת חבילות שלא בשימוש. לדוגמה אם מחקנו את החבילה pidgin באמצעות remove, בעזרת הפקודה נוכל למחוק את החבילה libpurple0 שלא בשימוש יותר. שימו לב שמידי פעם, מנהל החבילות מדפיס מידע על חבילות שלא בשימוש.

```
sudo apt-get autoremove
```

apt-get clean

מנהל החבילות שומר התקנות של חבילות שהוא הוריד, על מנת לאפשר התקנה מהירה יותר בעת הצורך.

על מנת למחוק את הקבצים האלה, נשתמש בפקודה זו:

```
sudo apt-get clean
```

תיקון תלות שבורה

יש פעמים בהם נשברים ה dependencies, למשל כאשר קוטעים התקנת חבילה באמצע, או מחיקה לא נכונה של חבילות (למשל עם rm). מנהל החבילות apt-get מציע את הפקודה:

```
sudo apt-get install -f
```

בין היתר ע"י הורדת החבילות המתאימות. אם היא לא עוזרת, יש פקודות אחרות שנראה בהמשך.

apt-get source

הפקודה תוריד ארכיבים של קוד המקור של החבילה המבוקשת, במידה ויש כזה. לדוגמה:

```
sudo apt-get sorce pidgin
```

הפקודה תוריד את קוד המקור של החבילה pidgin.

aptitude

התוכנה הזאת בעצם סוג של ממשק טקסטואלי שמקל על ניהול החבילות במערכת:

```
sudo aptitude
```

שימוש נוסף יכול להיות במקום apt-get, לדוגמה:

```
sudo aptitude update
sudo aptitude install pidgin
```

dpkg

מנהל החבילות apt-get בעצם מוריד קבצי התקנה מסוג deb, ומשתמשת ב dpkg על מנת להתקין אותם. dpkg הוא הבסיס למערכת החבילות במערכות מבוססות דביאן (וביניהם אובונטו), ובעזרתו ניתן להתקין, למחוק, ולקבל מידע על חבילות שהורדו.

dpkg -i

הפקודה תתקין חבילה מסוג deb. לדוגמה, אם הורדנו חבילת deb בשם pidgin.deb, על מנת להתקינה נרשום:

```
sudo dpkg -i pidgin.deb
```

כמובן שיש צורך ברוט, וכל הקבצים כפופים ליחסיות מערכת הקבצים בלינוקס.

dpkg -l

הפקודה תציג את כל החבילות המותקנות:

```
dpkg -l
```

אין צורך בהרשאות רוט.

dpkg -r

הפקודה תמחק את החבילה שצויינה. יש לציין ש dpkg תסרב למחוק חבילות שיש להן חבילות שתלויות בהן. לדוגמה:

```
sudo dpkg -r packet1 packet2
```

תמחק את החבילות packet1 ו packet2.

dpkg --configure -a

גם פקודה זו תנסה לאתר ולתקן dependencies שבורים:

```
sudo dpkg --configure -a
```

tar

התוכנה הזאת משמשת לפתיחת וקיבוץ ארכיבים, בין היתר מסוג tar.gz ו tar.bz2, שדי נפוצים בלינוקס.

על מנת לפתוח ארכיבים מסוג tar.gz, נרשום:

```
tar xvfz archive.tar.gz
```

על מנת לפתוח ארכיבים מסוג tar.bz2 נרשום:

```
tar xvfj archive.tar.bz2
```

כדי ליצור ארכיב מסוג tar.gz נרשום:

```
tar cvfz archivename.tar.gz folder/
```

כדי ליצור ארכיב מסוג tar.bz2 נרשום:

```
tar cvfj archivename.tar.gz folder/
```

בגרסאות האחרונות, אפשר לפתוח ארכיבים ללא ציון סוג, לדוגמה:

```
tar xfv archive.tar.bz2
```

שימו לב שהפרמטר x מציין פתיחה (eXtract), הפרמטר c מציין קיווץ (Create), הפרמטר f מציין קובץ (הארכיב שבא אחריו), הפרמטר j מציין קיווץ מסוג bz2, הפרמטר z מציין קיווץ מסוג gz, והפרמטר v מציין הדפסת הקבצים המקווצים/מחולצים.

wget

התוכנה מורידה קובץ מ URL נתון. לדוגמה אם נרצה להוריד את הקובץ xxx משרת 127.0.0.1 נרשום:
```
wget http://127.0.0.1/xxx
```

ניתן גם לשמור אותו בשם מסויים לדוגמה:
```
wget http://127.0.0.1/xxx -o myfile
```

ניתן להריץ את התוכנה ללא פלט ע"י:
```
wget http://127.0.0.1/xxx -o myfile -q
```

ניתן לנסות להמשיך הורדה (למשל סרט), כאשר הקובץ לא הורד במלואו, משרתים תומכים כמובן, ע"י הפקודה:
```
wget http://127.0.0.1/xxx -o myfile -c
```

ניתן לשנות את ה User Agent ב HTTP האדר, ע"י הפקודה:
```
wget http://127.0.0.1/xxx -o myfile -c --user-agent="agent"
```

ניתן להכליל קוקיז בבקשה ע"י הפקודה:
```
wget http://127.0.0.1/xxx -o myfile -c --load-cookies="file"
```

ניתן לשלוח מידע דרך POST, ע"י הפקודה:
```
wget http://127.0.0.1/xxx -o myfile -c --post-data="par=val&par2=val2"
```

ניתן להתעלם מסרטיפיקט לא מאומת ע"י הפקודה:
```
wget http://127.0.0.1/xxx -o myfile -c --no-check-certificate
```

כמובן ניתן לשלב את כל הפרמטרים הנ"ל ביחד.

make

והנה הגענו לחלק המסובך ביותר, נפלאות הלינוקס. הפקודה הזאת מקמפלת קבצי מקור, לקבצים שאותם ניתן להתקין במערכת. על מנת שיהיה אפשר לעשות כן, צריך MAKEFILE, ולפעמים גם עוד, אולם כשאתם מורידים קבצי מקור בלינוקס, בדרך כלל כבר דאגו לכך. בשביל לקמפל קבצי מקור, יש צורך בחבילה build-essential, שכוללת את המקומפיילר gcc, ועוד כמה חבילות שימושיות. לכן קודם לכן, נתקין את החבילה:

```
sudo apt-get install build-essential
```

על מנת לקמפל את קבצי המקור, ולבנות מהם קבצי הרצה, נכנס לתיקיית קבצי המקור בעזרת cd, ונרשום:

```
make
```

שימו לב, תהליך זה יכול לקחת זמן, תלוי בגודל הפרוייקט.

על מנת להתקין את הקבצים שקימפלתם כרגע, רישמו:

```
sudo make install
```

פעולה שלרוב פשוט מעתיקה את הקבצים המקומפלים למקומות המתאימים במערכת, ולכן דרשת פריוילגיות sudo.

לעיתים יש צורך לקנפג את קבצי המקור בהתאם למערכת. סקריפט מתאים לרוב נקרא בשם configure. כמו כן, לעיתים צריך למחוק את הקבצים המקומפלים ולקמפל מחדש. על מנת לעשות זאת נשתמש ב make clean. דוגמה:

```
./configure
make
sudo make install
make clean
```

פרק 7 - פקודות מתקדמות במערכת

היררכיית התיקיות בלינוקס

אתם יודעים שברוט דיר יש הרבה תיקיות, אבל האם תהיתם אי פעם מה בדיוק יש שם? פה בדיוק אני אסביר על זה.

בתיקייה bin יש קבצי הרצה של פקודות נפוצות בלינוקס, כמו ls, ב cat וכו'.

בתיקייה boot יש קבצים של הבוט לואדר, שאותם המערכת מעלה כשמעלים אותה.

בתיקייה dev מין קישורים לכל הדיבייסים המחוברים, כגון ה HD, ה cdrom, USB, וכו'.

בתיקייה etc יש את קבצי ההגדרה של רוב התוכנות במערכת.

בתיקייה home נמצאות תיקיות הבית של המשתמשים.

בתיקייה lib יש ספריות שימושיות וגם מודולים.

בתיקייה media יש התקני מדיה כגון פלאש דיסק, דיסק רגיל, כאשר הם "נפתחים" ע"י המערכת.

בתיקייה mnt אין כלום, אבל נוח להשתמש בה ב mount.

בתיקייה root יש את תיקיית הבית של המשתמש root.

בתיקייה sbin יש קבצי הרצה של התוכנות של המערכת.

בתיקייה srv יש מידע לסרביסים מסויימים.

בתיקייה tmp יש קבצים זמניים.

בתיקייה var יש מידע משתנה כמו לוגים וכו'.

בתיקייה usr יש תת הירורכיה:

(מעכשיו כל התיקיות הן בתוך usr)

בתיקייה bin יש את רוב קבצי הריצה של התוכנות במערכת כמו gcc ופיירפוקס.

בתיקייה include יש קבצי האדרים שאנחנו משתמשים בהם לקימפול תוכנות ב C, C++ וכו'.

בתיקייה lib יש את הספריות שהתוכנות המותקנות משתמשות בהם.

בתיקייה sbin יש קבצי ריצה של תוכנות של המערכת, כגון adduser, גנום (הממש הגראפי של Ubuntu כברירת המחדל), ועוד.

בתיקייה share יש בעיקר הרבה דוקומנטציה.

בתיקייה src יש סורסים של הקרנל וההאדרים שלו, דבר החיוני לבניית מודולים.

lsusb

הפקודה הזאת מציגה את כל ה USB המחוברים למחשב שלכם. הפקודה חיונית להוצאת ה ID של

המכשיר המחובר, ובכך למקד את החיפוש בגוגל אם יש בעיה כלשהי. לדוגמה, ה ID של העכבר שלי הוא: 046d:c041.

אפשר לסנן קצת תוצאות בעזרת grep ופייפ, כמו שלמדנו במדריכים הקודמים:

```
lsusb | grep Logitech
```

ידפיס רק את השורה עם העכבר.

lspci

הפקודה הזאת מדפיסה את החומרה המחוברת לכם בתוך המחשב. כדאי להשתמש עם grep לחיפוש מידע ספציפי:

```
lspci | grep Audio
```

אצלי הוא ידפיס:

```
00:1b.0 Audio device: Intel Corporation 82801I (ICH9 Family) HD Audio
Controller (rev 02)
```

מודולים

מודולים הם בעצם קבצים הנטענים ומעובדים ע"י הליבה של המערכת. מודולים משמשים פעמים רבות כדרייברים, המשמשים להפעלת חומרה. בלינוקס, יש הרבה מאוד מודולים שתפקידם לדאוג לעבודה תקינה של חומרה. כיוון שלרוב החברות המייצרות אותה לא מפיצות דרייברים ללינוקס, רוב המודולים שנכתבו ע"י הקהילה כבר נמצאים שם, ולכן לרוב אין צורך בהתקנת דרייברים (כמו בוינדוס) כשמחברים חומרה חדשה. המודולים נמצאים לרוב בתיקייה:

```
/lib/modules
```

modprobe

הפקודה הזאת טוענת מודול לתוך הקרנל. לדוגמה, אם נרצה לטעון את המודול gspca, שאגב אחראי על הפעלת לא מעט מצלמות אינטרנט, נרשום:

```
sudo modprobe gspca_main
```

שימו לב, פעולה זו דורשת פריווילגיות. גם במודולים באה לעזרתינו ההשלמה האוטומטית (טאב/פעמיים טאב).

rmmod

הפקודה הזאת מוציאה את המודול מן הקרנל. לדוגמה, אם נרצה להוציא את המודול gspca, נרשום:

```
sudo rmmod gspca_main
```

שימו לב, פעולה זו דורשת פריוילגיות.

lsmod

ופקודה זו, כמו שוודאי ניחשתם, מוציאה רשימה של כל המודולים הטעונים לתוך הקרנל:

```
lsmod
```

אם נרצה לחפש משהו ספציפי יותר, נשתמש ב grep עם פייפ:

```
lsmod | grep rt
```

הפלט יהיה בין היתר המודולים של כרטיס הרשת שלי.

טעינה אוטומטית של מודולים

לרוב, כאשר אתם מבצעים פעולה כלשהי המצריכה טעינה של מודול, כגון הכנסת חומרה ל USB, המערכת תזהה את המודול הדרוש, ותטען אותו. אבל, יש מקרים שבהם אתם רוצים לטעון דווקא מודול אחר, ולא אם המודול שהמערכת מתעקשת לטעון אותו, לכן אנחנו צריכים להכניס אותו לרשימה השחורה. על מנת לעשות זאת, נערוך את הקובץ הבא:

```
/etc/modprobe.d/blacklist.conf
```

עם עורך הטקסט המועדף עלינו, למשל nano, ונוסיף את השורה:

```
blacklist module_name
```

דרך פשוטה יותר, היא ע"י שימוש ברידיירקט ההוספה (וקודם לכן כניסה לרוט):

```
su
echo "blacklist module_name" >> /etc/modprobe.d/blacklist.conf
```

mount

הפקודה הזאת, טוענת את התוכן של דיבייס כלשהו, לדוגמה דיסק או מחיצה, לתוך תיקייה כלשהי, כך שבעצם אנחנו יכולים לראות ולערוך (או לא במקרה של דיסק) את תוכנה. השימוש הוא כדקלמן:

```
mount device_path mount_path
```

עכשיו נכנס כאן הקטע של כל ה DEVים. לדוגמה, אם ל CDROM שלי קוראים cdrom0 ב DEV (דרך שימושית היא להשתמש בהשלמה האוטומטית - טאב, פעמיים טאב, ולראות מה באמת יש שם), אני ארשום:

```
sudo mount /dev/cdrom0 /mnt
```

שימו לב שיש צורך בפריוילגיות רוט. זה גם לא משנה באיזה תיקייה אני "אשים" את התוכן, העיקר

שהיא תהיה ריקה ולא בשימוש (ותיקיית MNT מתאימה לצורך זה באופן מושלם).

דוגמה נוספת היא אם נרצה לפתוח מחיצה. זוכרים אמרנו שלמחיצות בלינוקס קוראים sda1 sda2 וכו'?

אז נגיד יש לכם מחיצה ב sda2, שפעם הייתה דיסק D שלכם, ואתם רוצים לראות את תוכנה. נרשום:

```
sudo mount /dev/sda2 /mnt
```

והנה נוכל עכשיו לעיין בקבצים ובסרטים ששמרתם בדיסק D ב"ימים הטובים" של וינדוס.

שימוש נוסף הוא רשימה של כל הדיבייסים שבוצעה עליהם הפקודה הנ"ל, ע"י הפקודה:

```
mount
```

umount

הפקודה הזאת בעצם מבטלת את הפקודה הקודמת. לדוגמה, אחרי שסיימנו לעיין בסרטים מהימים הטובים, נרשום:

```
sudo umount /mnt
```

ואין יותר סרטים, וגם אפשר לפתוח לשם עוד משהו. שימו לב שאין צורך לדעת של מי התוכן, וכן יש צורך בפריוילגיות יתר.

פרק 8 - חבילות **RPM** ופקודות אינטרנט

rpm

RPM נקראת גם Red Hat Packet Manager, היא מערכת החבילות במערכות מבוססות Red Hat, בין היתר Fedora, Mandriva ו CentOS. היא מאוד דומה ל DEBים, אבל יש גם שוני.

yum

yum הוא מנהל החבילות די נפוץ, במערכות כמו פדורה, סנטוס ועוד, לכן אני אכתוב דווקא עליו. אז נתחיל:

yum install

הפקודה תתקין את החבילה או החבילות המצויינות, אשר הוא יחפש במקורות: repositories. לדוגמה אם נרצה להתקין את החבילות xxx ו yyy, נכתוב:

```
yum install xxx yyy
```

כאשר אנחנו נמצאים ברוט (במערכות כמו פדורה וסנטוס לעיתים אין sudo).

yum localinstall

הפקודה הזאת מתקינה חבילה מתוך קובץ RPM. לדוגמה, אם יש לנו קובץ בשם package.rpm, נרשום (כמובן ברוט):

```
yum localinstall package.rpm
```

הפקודות הנ"ל כמו כן יחפשו את ה dependencies המתאימים ויציעו להתקין לכם את החבילות החסרות.

yum check-update

פקודה זו מעדכנת את ה repositories של המערכת. לא לשכוח לעשות זאת לפני עדכון המערכת:

```
yum check-update
```

yum update

פקודה זו מעדכנת את המערכת, אחרי שעדכנו את המידע על החבילות הזמינות:

```
yum update
```

על מנת לעדכן חבילה מסויימת, למשל בשם foo, נרשום:

```
yum update foo
```

yum remove

הפקודה מוחקת חבילות. לדוגמה, על מנת למחוק את החבילה xxx, נרשום:

```
yum remove xxx
```

yum list

הפקודה מוציאה רשימה של חבילות מסויימות, לדוגמה הפקודה:

```
yum list installed
```

תדפיס רשימה של כל החבילות המותקנות, והפקודה:

```
yum list updates
```

תדפיס את רשימת העדכונים הזמינים.

yum info

הפקודה תדפיס מידע על חבילה, למשל הפקודה:

```
yum info xxx
```

תדפיס מידע על החבילה xxx...

yum clean all

הפקודה תמחק את ה Cache של מנהל החבילות:

```
yum clean all
```

שימו לב, שלכל הפקודות הנ"ל יש צורך בהרשאות רוט!

הסט הבא של הפקודות יתייחס לפקודות רשת:

ping

הפקודה שולחת פינג לדומיין/כתובת IP כלשהי:

```
ping google.com
```

finger

הפקודה משמשת להוצאת מידע ממשתמש, לרוב ssh. לדוגמה, על מנת להוציא מידע על המשתמש שלנו, נכתוב:

```
finger username
```

ועל מנת להוציא מידע על משתמש שיושב על שרת עם IP מסויים (בתנאי שיש לו ssh) נכתוב:

```
finger user@255.255.255.255
```

בעזרת הפקודה אפשר לנסות לקבל מידע גם על אימיילים, שרתי FTP וכל סרביס אחר שרק מנדב מידע.

ifconfig

בעזרת הפקודה הזאת ניתן לשלוט על האינטרפייסים של האינטרנט שלכם. אינטרפייסים יכולים להיות התקן חוטי, אלחוטי וכו', ולרוב מקבלים שמות כמו eth0 לחוטי, ו wlan0 לאלחוטי. על מנת להציג את כל האינטרפייסים יש לרשום:

```
ifconfig
```

על מנת לקבל מידע על אינטרפייס מסויים נרשום

```
ifconfig wlan0
```

לדוגמה.

על מנת לכבות אינטרפייס מסויים נרשום:

```
sudo ifconfig wlan0 down
```

לדוגמה. יש צורך בהרשאות רוט, כיוון שבעצם אנחנו משפיעים על כל המערכת בפקודה זו.

על מנת להדליק בחזרה אינטרפייס מסויים, נרשום לדוגמה:

```
sudo ifconfig wlan0 up
```

iwconfig

הפקודה עוזרת בין היתר להתחבר לרשתות אלחוטיות. אם נרשום:

```
iwconfig
```

הפלט יהיה מידע על האינטרפייסים האלחוטיים.

על מנת להתחבר לרשת בשם xxx נרשום:

```
sudo iwconfig wlan0 essid xxx
```

ולאחר מכן נעזר dhclient על מנת לקבל IP וכו':

```
sudo dhclient wlan0
```

כמובן שאפשר להחבר עם ססמאות, ולהגדיר את כל ה IP והסאבנט מאסק וה DNS, אבל אני לא אכתוב על זה. ניתן לקרוא זה בהרחבה יותר בפקודה:

```
man iwconfig
```

netstat

הפקודה תציג מידע על כל הפרוצסים המשתמשים באינטרנט שלכם:

```
netstat
```

כדאי ומומלץ לקרוא עוד מידע:

```
man netstat
```

פרק 9 - שונות

daemons

הכוונה לפרוצסים שרצים ברקע ללא התערבות המשתמש. לדוגמה, הדאמון gdm אחראי על הסביבה הגראפית gnome, והדאמון networking אחראי על ניהול הרשת במערכת. רוב הדאמונים נמצאים בתיקייה:

```
/etc/init.d
```

והם נשלטים בצורה הבאה:

```
/etc/init.d/daemon stop
/etc/init.d/daemon start
/etc/init.d/daemon restart
/etc/init.d/daemon status
```

כשהפקודות הם עצירה, אתחול, ואתחול מחדש, והדפסת מצב בהתאמה. כמובן יש צורך ברוט, ועל כן אפשר להשתמש ב sudo. לדוגמה, על מנת לאתחל מחדש את הדאמון gdm, נכתוב:

```
sudo /etc/init.d/gdm restart
```

מומלץ לנסות כשאין חלונות חשובים פתוחים (דפדפן, מסמך וכו').

דרך נוספת לשלוט על הדאמונים היא באמצעות הפקודה service. לדוגמה, אם נכתוב:

```
sudo service gdm restart
```

גם פקודה זו תאתחל מחדש את הסביבה הגראפית. דרך שימושית לצאת ממצב של תקלה בסביבה הגראפית, או אם היא נתקעה, מבלי לאתחל מחדש את המחשב, היא להכנס לטרמינל דרך Ctrl+Alt+F1, להכנס למשתמש, ולכתוב את הפקודה הנ"ל.

הדאמנוים מופעילים לרוב ע"י הפרוצס init, שמורץ ראשון על המערכת, ולכן ה ID שלו גם 1.

chroot

הפקודה הזו משנה את הרוט דיר, לרוט דיר אחר, שביחס אליו יבנו כל היחסיות הנדרשת. הפקודה שימושית כשעושים mount למחיצה מסויימת שמותקנת עליה לינוקס. הפקודה מצריכה המצאות הקובץ:

```
/bin/bash
```

ברוט החדש, ולכן כמובן תעבוד רק עם לינוקס. לדוגמה, אם לא עולה לנו המערכת, נעלה Live Cd

ונרשום:

```
sudo bash
mount /dev/sda2 /mnt
mount --bind /proc /mnt/proc
mount --bind /dev /mnt/dev
chroot /mnt
```

הפקודה השלישית מביאה לנו את האפשרות להשתמש בפרוצסים הפועלים על ה Live Cd גם ב mnt. הפקודה הרביעית עושה אותו דבר על הדיבייסים המחוברים (מחיצות, דיסקים וכו'). אחרי שכתבנו את הפקודה האחרונה, אנחנו יכולים להתנהג כאילו שאנחנו נמצאים במערכת שלנו, כיוון שהכל מתייחס לרוט דיר של המערכת, ולא של ה Live Cd. בכל מקרה, שימו לב שאי אפשר לעשות זאת על מערכות עם סיביות שונה (32 ביט ו 64 ביט), וכן מומלץ להשתמש ב Live Cd של מערכת קרובה ככול האפשר למערכת שאנחנו רוצים לתקן.

על מנת להחזיר את הרוט דיר למצב הקודם, נכתוב:

```
exit
```

fdisk

בגדול הפקודה משמשת ליצירת מחיצות על הכונן הקשיח, אני לא אפרט על זה, אבל הפקודה השימושית יותר שלו היא:

```
sudo fdisk -l
```

שהיא מדפיסה את המחיצות הקיימות על ה HD.

grub

גראב הוא הבוט לואדר הנפוץ ביותר של המערכת. הבוט לואדר נותן לבחור איזה מערכת להעלות, איזה קרנל להעלות, וכו', ובגדול מעלה את המערכת. הגרסה העדכנית שלו היא 2, ומותקנת על ההפצות החדשות של Ubuntu כברירת מחדל. גרסה זו שונה מאוד מהגרסה הקודמת, וכיוון שהיא חדישה יותר, אני אכתוב דווקא עליה.

קודם כל, אם אתם לא רואים בחירה, לחצו והחזיקו את המקש Shift בעת הפעלת המחשב, עד שתופיע לכם הבחירה. משם אתם יכולים להעלות אפילו דרך קומנד ליין קטן כל מה שרק תרצו.

על מנת לעדכן את הרשומות ב grub, שיודע אף לזהות מערכות אחרות כמו וינדוס (בניגוד לוינודס), יש לרשום:

```
sudo update-grub2
```

את הרשומות שאתם רוצים להוסיף יש להוסיף בקובץ:

```
/etc/grub.d/40_custom
```

שצריך להסתיים בשורה ריקה, והוא יכלל בקובץ
/boot/grub/grub.cfg

שכמו שמצויין בתחילתו, לא כדאי לערוך אותו, כיוון שהוא נמחק כל פעם כשמופעלת פקודת העדכון (למשל אחרי עדכון קרנל).

אני לא אכתוב כאן על איך כותבים רשומה, על זה יש אינספור מדריכים, אולם אני אראה פתרון לבעיה שכיחה, והיא התקנת וינדוס. סתם, למרות שהתקנת וינדוס בעיה לשל עצמה, הבעיה היא מחיקת grub, והתקנת הבוט לואדר של וינדוס שלא יודע לזהות אפילו את המחיצה של לינוקס...
בכל מקרה, על מנת לשחזר את ה grub2, נפעיל מ Live Cd, וכמו שהראתי מקודם על איך מגיעים למחיצת הלינוקס, נוסיף על זה רק את הפקודה grub-install, והתוצאה תראה כ:

```
sudo bash
mount /dev/sda2 /mnt
mount --bind /proc /mnt/proc
mount --bind /dev /mnt/dev
chroot /mnt
grub-install /dev/sda
exit
reboot
```

בהנחה שמחיצת הלינוקס היא sda3, וה HD הרצוי הוא sda. שימו לב, אתם מתקינים grub על ה HD ולא על המחיצה של הלינוקס.

הרצת פקודות ברקע

על מנת להריץ פקודות/תוכנות ברקע, כלומר בצורה שתאפשר לנו להמשיך ולעבוד עם הטרמינל, נוסיף אחרי הפקודה את הסימן &. לדוגמה אם נרצה להריץ את פיירפוקס ברקע, נכתוב:

```
firefox &
```

סיכום

ובכן, זהו המדריך הקטן שלי למתחילים בלינוקס, שלא יודעים למצוא דרכם בכלי השימושי שנקרא טרמינל. אני מקווה שהמדריך יעזור לכם להפוך ללינוקסרים אדוקים יותר, וישמש אותכם בעתיד.

הערות אפשר לשלוח אלי לאימייל: Qazjap11 שטרודל ג'ימייל נקודה קום.

בברכה,

Qazjap11