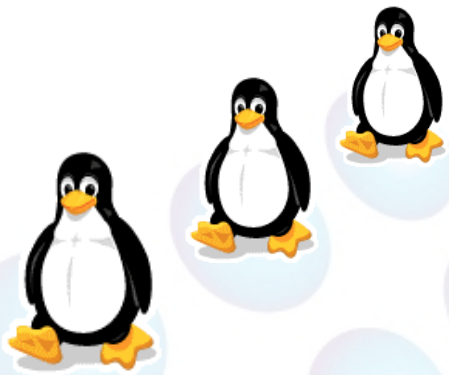
שבוע אורקל 2004

Taking the Linux road



דורון אופק
Linux & Opensource technology leader
סער מעוז
Consulting Performance Engineer - Oracle

שבוע אורקל

על מה נדבר היום ?

- מהי לינוקס
- כיצד היא צמחה
- הפן האידיאולוגי מאחורי התוכנה החופשית
- היכן רואית התקדמות בשימוש בלינוקס
- כיצד נכניס לינוקס אל הארגון
- טיפים וטריקים

מהי לינוקס ?

- לינוקס
- מהי מערכת הפעלה
- למה אני (כלקוח) , צריך את זה ?
- חשיבותם של תקנים פתוחים

מהילינוקס ?

לינוקס הינה מערכת הפעלה.

לינוקס בנויה במודל בסיסי מונולוטי (בשלב מסויים בתהליך ההתפתחות שלה, עברה שינוי למודל מודולרי).

לינוקס מתפתחת במודל של תוכנה חופשית.

מהימערכת הפעלה

- מערכת הפעלה הינה תוכנית אשר נכנסת לפעולה בזמן איתחול המחשב ונשארת פעילה עד רגע הכיבוי, ככזאת היא שולטת בפעילות התוכניות שמופעלות. היא האחראית לספק להם משאבים ולתת להם את המצע הדרוש להם לעבודה.
- מערכת ההפעלה היא התוכנית הבלעדית אשר אחראית לתקשורת עם החומרה ולפעילות הנכונה של החומרה.
- מערכת ההפעלה , היא תוכנית המכילה תוכניות משנה (פרוצדורות) אשר עשויות לטפל גם בהבטים שונים של המערכת, תמיכה בפרוטוקולי תקשורת ועוד.

מהימערכת הפעלה

- מערכת הפעלה על פי ההגדרה השמרנית:
- תוכנה המנהלת את משאבי המחשב, והמספקת כלים ניהוליים לשם כך .

מהימערכת הפעלה ?

- לפיכך האם לינוקס היא מערכת הפעלה ?
- התשובה היא , לא ! (ברמה התאורטית)
- לינוקס אמנם מספקת שליטה ברכיבי החומרה והתוכנה אך אינה מספקת כלים ניהוליים לאדמיניסטרציה של המערכת , לינוקס למעשה נשענת על כלי ניהול שפותחו תחת GNU ומאמצת אותם , שימוש בכלי GNU שונים ביחד עם לינוקס מספק מערכת הפעלה .

למה לינוקס ?

לינוקס בהגדרתה הינה תוכנית תשתיתית, היא מספקת מצע לפעילויות של אפליקציות אחרות.

"מה שמיוחד במערכת הפעלה היא שאתה לא אמור לדעת על הפעילות שלה, היא לא אמורה להפריע לך או להטריד אותך, המשתמש עובד מול תוכנית וזה מה שהוא צריך להכיר" – לינוס טרובלדס .

למה לינוקס ?

בעולם המסורתי של תוכניות, יצרן יכל לעשות (ואף עשה) דברים שאינם סטנדרטיים , על מנת לקבע את הלקוחות שלו את התוכניות שלו. במצב שכזה הלקוחות מרגע שבחרו מערכת הפעלה (או כל תוכנית אחרת) הפכו להיות "שבויים" של היצרן.

למה לינוקס ?

מודל התוכנה החופשית מעביר את מרכז הכובד אל הלקוחות .

לינוקס, אינה "כובלת" את הלקוח אל מפיץ ספציפי, תהליכי הגירה ממערכת של מפיץ אחד למערכת של מפיץ אחר היא קלה (אם בכלל קיימת) .

בעבר פרוטוקולי תקשורת בין מחשבים היו קיניניים

כתולדה מכך לא כל פרוטוקול או שיטה יכלו "לדבר" עם פרוטוקול או שיטה אחרת.

תולדה של מצב זה היתה שהלקוחות נדרשו לרכוש מספר טכנולוגיות ומספר מוצרים על מנת לקבל את היכולת לגרום למערכות שונות "לדבר" בינהן.

הכניסה של פרוטוקול ה IP הכניס כלל השוק למצב בו כל הלקוחות רצו לממש עבודה עם הפרוטוקול הנ"ל ונוצר לחץ על היצרנים לטפל בכך.

פורוטוקולי התקשורת כדוגמה לנושא של סטנדרטים אחידים וחופש

היצרנים התיישרות על פי דרישת הלקוחות והתוצאה היא שהלקוחות קיבלו את הפרוטוקול במחיר נמוך (כחלק ממערכת ההפעלה) ויותר מכך הם יכלו לגרום למערכות שונות לתקשר ע"י שימוש בסטנדרטים אחידים.

היום נראה לנו טבעי שפרוטוקול הוא רכיב תוכנה תשתיתי אשר כל הלקוחות יכולים לעשות בו שימוש – אולם צריך לזכור שלא תמיד זה היה המצב.

מה שקורה ללינוקס הוא שלב נוסף באבולוציה של הרעיון הנ"ל.

מהי "תוכנה חופשית" ולמה היא נוצרה?

תוכנה חופשית היא תוכנה שבעל זכויות היוצרים המקורי וויתר מרצון על "זכויות " שיש לו בה, הוא לכאורה העניק את הזכויות הללו לכלל הציבור , תוך השמה של מספר תנאים שישמרו את טובת הציבור בנושא התוכנה הזו. זה לא אומר שלתוכנה חופשית אין זכויות יוצרים, על תוכנה חופשית יש זכויות יוצרים !

במימד הצר , ניתן לומר כי אותו "מפתח" וויתר על השליטה בתוכנית שהוא עצמו יצר , במימד הרחב , ה"מפתח" מקבל בתמורה לוויתור הזה , סיוע של אנשי פיתוח נוספים אשר מסייעים לו לקדם בצורה טובה יותר את התוכנית שלו.

ככל שיותר אנשים יראו את התוכנית ואת הקוד שלה , האיכות של התוכנית תשתפר וכך גם היכולות שלה.

חשוב לציין כי תוכנה חופשית אינה Public Domain כלומר היא לא ניתנת לשימוש בלתי מוגבל ע"י המשתמשים אלא תוכנה שיש לה זכויות יוצרים ויוצר.

אין קשר בין היות תוכנה , תוכנה חופשית, לבי העלות הכלכלית שלה (חינמיות היא מילה גסה) .

קצת היסטוריה – היכן ולמה הגענו אל מודל התוכנה החופשית?

בשנת 69 הוציאה חברת AT&T מערכת הפעלה אשר לימים נקרא UNIX , מערכת ההפעלה הזו היתה בנויה מבליל של שפות ולא היה נוח לעבוד על המשך הפיתוח שלה. לפיכך התקבלו בחברה 2 החלטות . האחת להחזיר את המערכת אל המעבדות על מנת לבנות אותה מחדש בשפת C (בין המפתחים של המערכת ניתן למצוא כמובן דמויות שנחשבות ל"אבות הרוחניים " של שפת C) , תהליך הכתיבה מחדש הסתיים ב 72 . ההחלטה השניה היתה לספק את המערכת במחיר 0 לאוניברסיטאות ומכוני מחקר .

הרקע להחלטות הללו , היה כתוצאה מהבנה ב AT&T שיש להם מוצר טוב ביד , אך ביחד עם זאת הם לא יכלו למכור אותו (באותה תקופה החברה הוכרזה כמונופול בארה"ב , היא אולצה להתפרק למספר חברות שונות והיו תחומים שבהם היא לא יכלה לעסוק) .

קצת היסטוריה – היכן ולמה הגענו אל מודל התוכנה החופשית?

החלה לפיכך את דרכה כ"מערכת הפעלה" יחסית פתוחה , אולם רשיונות שנמכרו לחברות המסחריות הגדולות על מנת שהם יוכלו לייצר UNIX גם כן , הפכו להיות "חרב פפיות" , היצרנים רצו לקבע את הלקוחות שלהם אליהם וכל אחד הכניס שינויים במערכת עד כדי יצירת שוני כזה שמערכת אחת לא היתה דומה למערכת אחרת.

מהצד שכנגד , אוניברסיטאות הצטרפו לתהליך הפיתוח , מכוון שהקוד היה קיים אצלם והן רצו לשפר אותו (ברקלי לדוגמה) , המערכות שהתפתחו בסביבה האוניברסיטאית היו יותר "תיקניות" אך גם פחות ידודי דותיות.

2 הקווים, האוניברסיטאי והעסקי , ליוו ומלווים את עולם ה UNIX עד עצם היום הזה.

הקמת ה GNU וה FSF :

RMS

MIT

GNU

FSF

הקמת ה GNU וה FSF :

- בתחילת שנות ה 80 אדם בשם ריצ'רד סטולמן (ידוע בכינוי RMS) עבד במכון לחקר בינה מלאכותית של MIT . הוא נתקל במצבים בהם אנשי מרכז החישובים הציעו תיקונים לבאגים והצעות לשיפור, למפיצי ה UNIX השונים.המפיצים קיבלו את התיקונים והכניסו אותם למערכת אולם אז נדרשו אלו שתיקנו את המערכת לשלם בעבור התיקונים הללו – כלומר המפיץ עשה בתיקון מה שהוא מצא לנכון (מן הסתם צריך לזכור כי סביבה אוניברסיטאית היא חסכנית יותר מסביבות אחרות)

- הדברים נראו לא הגיוניים ל RMS והוא בנה מודל שונה לייצור של תוכנה , "מודל התוכנה החופשית"

- במודל שלו מציג RMS את התוכנה כסוג של שירות , בדומה לעבודת עורך דין / מעצב / אדריכל , בשונה מייצור של מוצרים ברי קיימה שהם יותר מדידים מבחינת האיכות והמחיר, מוצרי תוכנה הם מוצרים שבסיס הידע בעבורם אמור להיות משותף לכל האנושות.

אם כך מהי תוכנה חופשית ?

אם כך מהי תוכנה חופשית ?

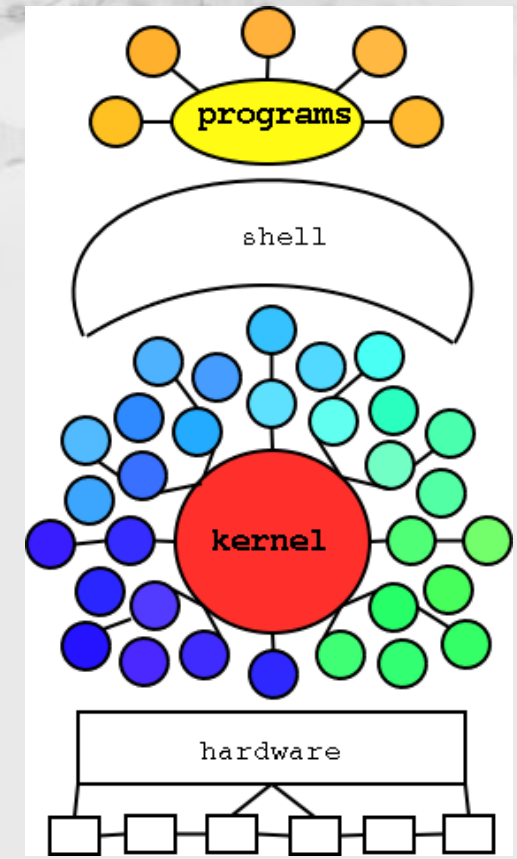
- על פי ההגדרות הקיימות (הוגדרו ע"י RMS) תוכנה חופשית היא תוכנה שעונה ל 4 קריטריונים:
- 1. החופש להפעיל את התוכנה לכל מטרה שהיא (freedom 0) .
- 2. החופש ללמוד איך התוכנית עובדת, ולשנות אותה לצרכי המשתמש (freedom 1) , גישה לקוד המקור של התוכנית הינה תנאי מקדים לחופש זה .
- 3. החופש להפיץ מחדש את התוכנית על מנת שתוכל לעזור לחברך (freedom 2)
- 4. החופש לשפר את התוכנית ולהפיץ את השיפורים לטובת כלל הציבור (freedom 3) גישה לקוד המקור של התוכנית היא תנאי מקדים לכך.

איך עושים את זה?

- RMS הקים את פרוייקט GNU כגוף ביצועי לטובת פיתוח של תוכניות כאלו
- הוקם ה FSF כגוף מנהלי שקיים מעל ה GNU
- נבנה רשיון הקרוי GPL – General Public License על מנת להקנות את האחיזה החוקית לתוכניות אלו .
- משנת 84 (הקמת ה GNU) ועד 91 הוקמה תשתית של אפליקציות על מנת לספק למשתמשים סביבת עבודה חופשית, בשנת 91 הוציא לינוס טרובלדס את ה kernel שלו תחת רשיון GNU .

מה משתמש צריך כדי לעבוד על מערכת מחשב ?

ברמה העקרונית מדובר בסדרה של תוכניות שעובדות אחת על גבי השניה , הסדרה הזו שאנחנו קוראים לה "מערכת הפעלה" אינה באמת מערכת ההפעלה – היא במקרים רבים מערכת הפעלה שאליה מוסיפים כלים , או תוכניות נוספות.

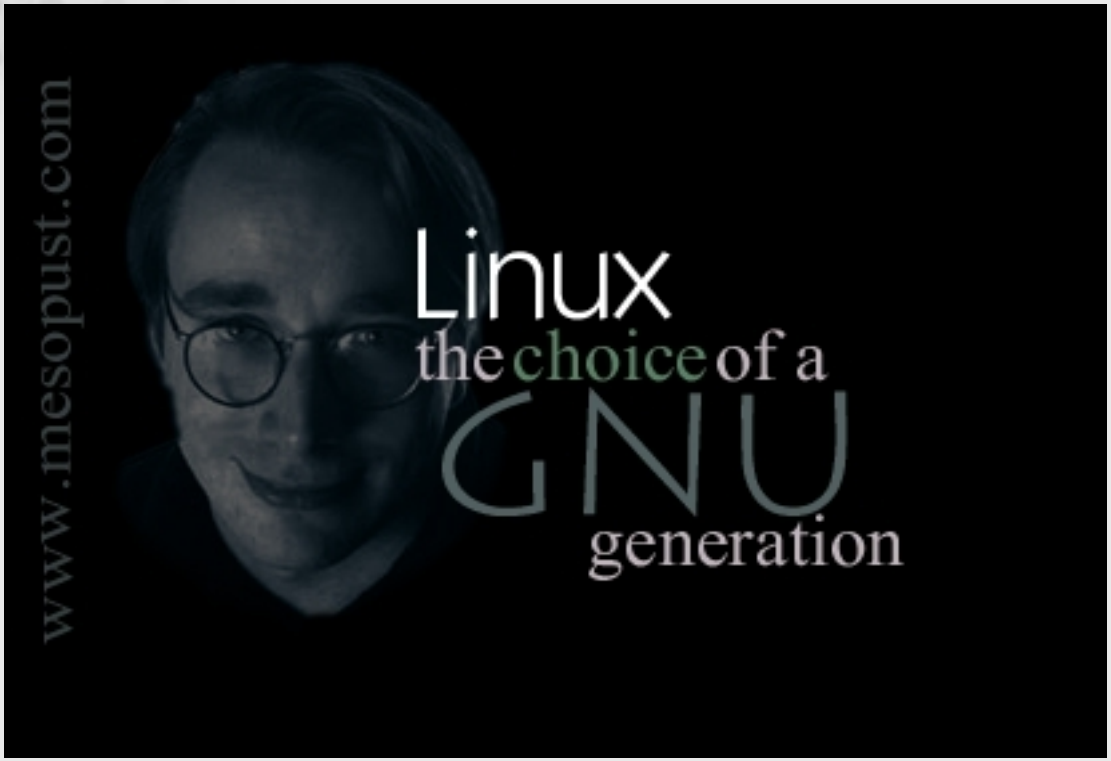


בצורה סכמתית יש מספר "שכבות" :

KERNEL – הוא למעשה מערכת ההפעלה

ה shell

תוכניות .


www.mesopust.com
Linux
the choice of a
GNU
generation

בסוף 91 היה סטודנט למדעי המחשב באוניברסיטה של הלסינקי וחבר ברשימת התפוצה של minix .

Minix היתה מערכת הפעלה שיצר פרופ' אנדו טטנבאום (מדעי המחשב) על מנת שתלמידים שלו יוכלו לתרגל באופן פרקטי שינוי ובניית רכיבים בתוך מערכת ההפעלה (הולנד)

לינוס למרות היותו סטודנט באוניברסיטת הלסינקי , נרשם לרשימת התפוצה הזו על מנת לנסות ולראות מה אפשר לעשות איתה.

בסוף 91 הוא הודיע במכתב לרשימת התפוצה שהוא כניסיון הצליח לבנות מערכת הפעלה (KERNEL) שאינה כוללת קוד של minix ושהוא מתכוון לשחרר אותה לשימוש תחת רשיון GPL .

למעשה צירוף ה KERNEL של לינוס טרובלדס (LINUX) ביחד עם תוכניות נוספות שכבר היו קיימות ב GNU סיפקו מענה לסביבת עבודה חופשית.

מודל התוכנה החופשית – האם זה מצליח ?

כעובדה , בשנים האחרונות נכנסה מערכת ההפעלה הצעירה לשימוש מאסיבי , כפתרון בסביבות השרתים ועל התחנות המקצועיות.

הן ברמה הרעיונית והן ברמה התאורטית מסתמן כי הדברים דומים לגידול אקספוננצלי, ככל שיש יותר משתמשים כוח הפיתוח עולה , כמות הפיתוחים עולה.

עולם התוכנה החופשית שבר את הקשר שבין יכולת ההשקעה של גוף יחיד בפיתוח וכמות הפיתוח ע"י מתן סביבה בסיסית פתוחה לעבודה.

ציוני דרך חשובים בעולם הקוד הפתוח

בתחילת 2001 שוחררה גרסה (רשמית) של kernel 2.4 , גרסה זו היוותה קפיצת דרך מבחינת עולם הלינוקס בגלל כמות גדולה יחסית של יכולות שהוכנסו אל גרסה זו עוד בטרם השיחרור הרשמי , מכוון שהקוד היה זמין לחברות גדולות ולמי שעוסק בתחום, נרשם גידול בשימוש בלינוקס בעיקר באירופה בארה"ב , למעשה אנשי המקצוע שמכירים את התחום צפו כי שיחרור גרסה זו, תכניס את השימוש בלינוקס אל ליבת הארגונים , ואם עד גרסה זו המערכת נחשבה למערכת שלא היתה לה נוכחות מאסיבית ביותר בליבת העסקים – שיחרור גרסה זו העלה את המערכת לרף שבו עסקים יכלו להשתמש במערכת והתייחסו אליה כשווה אל מערכות אחרות.

מאז למעשה כל מפיצי התוכנה והחומרה הגדולים החלו בהסבה מאסיבית לסביבת לינוקס.

כיום אנו נמצאים בפרק הזמן בו יוצא kernel בגרסה 2.6 לשוק , גרסה זו אינה מכניסה שפע אדיר של שינויים ברמה החיצונית אלה מייצבת בצורה טובה המון שינויים שנכנסו מאז צאת הגרסה הקודמת – למעשה ניתן לומר שהיא מהווה "הבשלה" טכנולוגית של קפיצות רבות שנעשו בשנים האחרונות.

עם כניסתם של שחקנים עסקיים לשוק הלינוקס , הליך שהחל לקרות בסוף שנות ה 90 הגיעו מספר אנשי לינוקס ברמה העולמית למסקנה שקיימת סוג של בעיה בקשר לרישוי של הלינוקס.

למעשה, עשויים להיות עסקים אשר מחד רוצים להנות מייתרונות עולם הקוד הפתוח ומאמינים שאכן שיטה זו היא השיטה הטובה, אך מאידך שימוש ברשיון GPL עשוי להכניס אותם לבעיה במודל העסקי שלהם (מסירה של הקודים של התוכניות שלהם) .

תנועת ה opensource – ההבדל בינה ובין ה"תוכנה החופשית"

למעשה תנועת הקוד הפתוח הינה תנועה שמוכנה לקבל וויתור מסויים על החופש המוענק למשתמש על מנת שחברות יוכלו להיכנס אל הליך "פתוח קוד" עם ההרגשה היותר בטוחה לגבי עתידו של הקוד.

אין מדובר כמו בדוגמה של ה GNU וה GPL על רישיון יחיד שמייצג את הרעיון , אלא על שורה של רשיונות שעונים למספר קריטריונים.

במרבית המקרים אנשי הקוד הפתוח טוענים כי הבחירה בלינוקס צריכה להיות בגלל היכולות הטכניות שלו וכן ניתן לעשות וויתור מסויים על מנת לקבל את אותן איכויות מלינוקס.

כמו שצויין קיימים שורה של רשיונות אשר אינן "קיצוניות" כמו ה GPL , הם אמנם שומרים על חלק נרחב מהמתודולוגיה שלו אולם מנסים עם זאת להיות מציאותיים יותר.

מערכת לינוקס כפי שאנו קוראים לה כיום הכוללת את ה kernel וכלים רבים נוספים הינה מערכת שבבסיסה נשענת על ה GPL אך עם זאת כוללת כלים רבים שרישיון השימוש להם הינו שונה מעט.

כאשר נשווה את לינוקס אל מערכות Unix מסורתיות נמצא מערכת שיש לה יכולות לא פחות טובות מהן ולעיתים טובות מהן (ישנה טענה שהיכולות נגזרות גם מיכולות חומרה טובות שקיימות בסביבת אינטל לעומת סביבת RISC) .

כאשר נשווה את לינוקס אל Windows נמצא מערכת ש"ירשה" רעיונות רבים מעולם ה Unix אך השכילה (ע"י "ברירה טבעית ") להשאיר את הרעיונות הטובים בלי לרשת רעיונות גרועים – לינוקס עולה בתחומים רבים על סביבת ה windows , עדיין ישנם תחומים שבהם לינוקס פחות טובה וזה בגלל התייחסות היסטורית לעולם ה Windows .

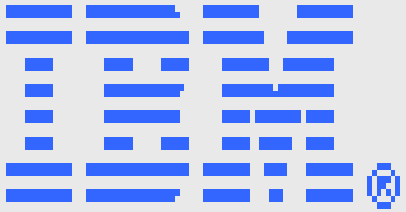
השחקנים העסקיים – מי ומה בעולם הקוד הפתוח

לינוקס וכלי ה GNU הינם בראש ובראשונה סטים של כלי תוכנה אשר ניתנה לקהילה את הזכות לשנות , ונשמרה זכותה של הקהילה בהמשך הליך הפיתוח של אותם כלים. לפיכך לינוקס היא קודם כל קהילה. כבר היו לא מעט מקרים בהם גורמים מסחריים ניסו לשנות בצורה כזו או אחרת אפליקציות שונות , ניסו לכוון לכיוון פיתוחי מסויים – וכל זאת נכשלו (גם הגדולות שבחברות) וזאת מכוון שהקהילה הינה אוטוריטה מרכזית בקוד ובתהליך הפיתוח.

המשך הליך הפיתוח אינו נתון כלל בידיהם של השחקנים העסקיים , הם אמנם יכולים לתרום לו אך בסופו של דבר מקור הסמכות הינה הקהילה.

עם קבלת התנאים הללו, חברות עסקיות יכולות להצטרף אל ההליך הפיתוחי , כלים רבים פותחו בסיוע של חברות מסחריות , בין החברות שמסייעות לפיתוח הלינוקס והתוכניות השונות ניתן למנות את כל יצרני ה UNIX לשעבר, חברות תוכנה וחומרה, חברות אשר המוצרים שלהם נושאים "מערכת משובצת" .

דוגמאות לחברות :



מיפוי הצרכנים בישראל

בישראל ניתן למצוא מספר סוגים של צרכנים:

אוניברסיאות , מכוני מחקר , ארגונים ממשלתיים.

חברות תשתית, חברות IT ,

חברות פיתוח, חברות שירות.

GOVERNMENT
LINUX

לינוקס – המגזר הממשלתי

הנושא של שימוש בקוד פתוח בתוך המגזר הממשלתי תופש תאוצה במדינות רבות בעולם .הסיבות לתהליך זה הן רבות:

אי – תלות בספק יחיד

יכולת לספק לאזרחים אלטרנטיבה סבירה במחיר מינימאלי

יכולת להוזיל את עלויות הפעלת הרשת הממשלתית

שמירה על הון מקומי בתוך אותו משק לעומן יציאת כספים כאשר מדובר במוצרים מסחריים.

יכולת לספק מקומות עבודה ועידוד הפעילות המשקית המקומית בהתבסס על קוד פתוח.

הנושא של שימוש בקוד פתוח בתוך המגזר הממשלתי תופש תאוצה במדינות רבות בעולם .הסיבות לתהליך זה הן רבות:

אי – תלות בספק יחיד

יכולת לספק לאזרחים אלטרנטיבה סבירה במחיר מינימאלי

יכולת להוזיל את עלויות הפעלת הרשת הממשלתית

שמירה על הון מקומי בתוך אותו משק לעומן יציאת כספים כאשר מדובר במוצרים מסחריים.

יכולת לספק מקומות עבודה ועידוד הפעילות המשקית המקומית בהתבסס על קוד פתוח

מצד שני ישנם גם מתנגדים לתהליך:

הנטיה לראות בכך "נקיטת עמדה" של הממשלה ביחס לספקי התוכנה הגדולים.

טענה לטובת אי המעורבות הממשלתית בפעילות במשק.

מגזרים ממשלתיים במקומות אחרים בעולם

ברחבי העולם ניתן לראות בעשור האחרון ממשלות אשר פועלות בתחום של לינוקס ותוכנה חופשית ממספר סיבות (שלא הוזכרו קודם לכן)

ממשלת ארה"ב דרך ארגונים ממשלתיים מפעילה החל משנת 95 פרוייקטים על לינוקס (פרוייקטים אשר החלו ב 95 ב NASA) - אין בכך כדי להעיד על מעורבות ממשלתית כלשהיא אלא שהגוף הנ"ל מצא את המערכת טובה דיה כדי לשמש אותו למשימות מסויימות.

בסוף שנות ה 90 הצטרף גוף נוסף לשימוש בלינוקס ואף פיתח מספר אפליקציות לכך , מדובר ב NSA ובמספר נוסף של ארגונים מהתחום שבו ה NSA עוסק אשר מצאו בלינוקס כר נוח וטוב על מנת להשתמש במערכת לטובת צרכים "מבצעיים" – הנקודה אשר דיברה אליהם הינה דווקא הנושא של אבטחת המידע שאליו ניתן להגיע עם לינוקס.

הפרויקטים של ה NSA וצורת העבודה שלהם הפכו להיות "תקן דה-פאקטו" במגזרים ביטחוניים , ממשלתיים, צבאיים וחצי צבאיים, במקומות רבים בעולם . בכללם ניתן למנות את אוסטרליה, ניו זילנד, ארה"ב , ומדינות מהגוש האירופי.

מעבר לסיבות שהוזכרו לעיל בדבר השימוש בלינוקס , יש לזכור כי תוכניות קוד פתוח באופן טבעי עובדות על פי התקנים הבסיסיים.

אם יהיה מצב שבו בתוכנית מסויימת תיהיה חריגה מהתקן , המשתמשים וארגוני הקוד הפתוח יזהו זאת במהרה ולא יוכלו לקבל מימוש שכזה , הסיבה לכך היא פשוטה , חריגה מהתקנים מתחילה תגובת שר שרת על אפליקציות רבות במערכת דבר שהוא בלתי נסבל מבחינת ארגוני הקוד הפתוח .

אצל יצרני קוד סגור קל יותר לחרוג מהתקנים ויצרנים רבים עושים זאת. נוצר מצב שבו מערכות שונות של יצרנים שונים אינם מאפשרות למשתמש ליצור סינכרוניזציה כזו או אחרת בגלל אותה חריגה מהתקנים.

כמשתמש – שמירה על תקינה בתוכניות שאיתן אני עובד מבטיחה לי שאף אחד לא יוכל "לכלוא" אותי בתוך האפליקציה שלו ושתמיד אני יוכל להשתמש באפליקציה אחת כאשר היא מתקשרת עם מערכות אחרות – ע"י מימוש של אותה תקינה.

גם האו"ם מתגייס למשימה

כגוף חיצוני לממשלות השונות ולשאיפות שלהן , החליט גם האו"ם לעשות בנושאים השייכים לטכנולוגיית המידע, בעיקר לאור העובדה שפערים הולכים וגדלים בין מדינות מפותחות ומתפתחות הינם בעיתיים, לראייתם.

בדצמבר 2003 נערך בז'נאווה כנס WSIS כנס זה הינו אחד משורה של כנסים אשר בסופו של דבר אמורים לייצר מצב בו תתקיים רגולציה בנושא של שימוש בטכנולוגיית המידע בין מדינות שונות.

בכנס, שמן הסתם היו בו לא מעט לחצים פוליטיים, הסתמן גוש ברור ביותר של מדינות מתפתחות אשר דחפו את הנושא של קוד פתוח והעלו אותו על סדר היום הציבורי.

קהילת הקוד הפתוח בישראל – מיהם הגופים הפעילים

- Iglu – הארגון הראשון שהתקיים כארגון "לינוקסי".
- Whatsup – אתר שסביבו נבנתה קהילה , עוסק בעיקר בחדשות וארועים קהילתיים.
- ה Penguin – אתר שמרכז מאמרים ומדריכים לשימוש הקהילה.
- אנשים מתוך גופים אלו היוו את הבסיס להקמת "המקור" (עמותה) , שמהווה את ארגון הגג של משתמשי הלינוקס בישראל.

IGLU
the Israeli Group of Linux Users.

קהילת הקוד הפתוח בישראל היא מן הקהילות הפעילות שיש.

בשנת 96 לערך הוקם גוף (Iglu) אשר היווה קהילה וירטואלית לקהילה המקומית, מפתחים וסטודנטים מהארץ התחילו לשתף בינהם מידע .

בשנת 98 לערך החלו להופיע פורומים (מבוססי web) אשר גם בהם התרכזו קהילות הולכות וגדלות של משתמשים.

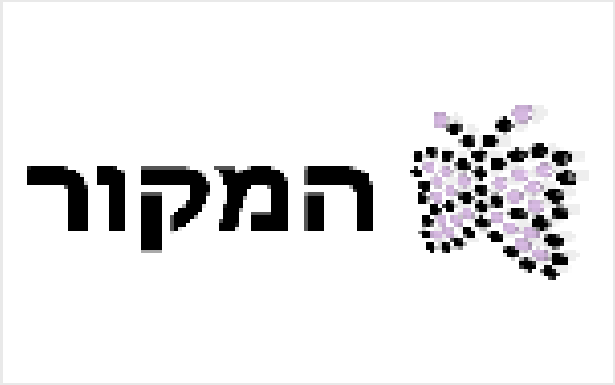
בשנת 2000-2001 , הוקמו 2 אתרים אשר מהווים היום מקום ריכוז לקהילה המקומית , האתר whatsup.org.il אשר מהווה אתר חדשות ופורומים , והאתר penguin.org.il אשר מהווה אתר למדריכים ומידע למשתמשים חדשים יותר וחדשים פחות.

בשנת 2002 , הוזמנה קבוצה של משתמשי לינוקס לדיון בוועדת הכנסת לענייני מדע , מכוון שאי אפשר היה לומר שמי מבין הקהילות מייצגת את כלל קהילת הלינוקס , הורכבה משלחת אשר הכילה בתוכה אנשים מכל הקבוצות – קבוצה זו היוותה את הגרעין להקמת ה"מקור" .

2 ארעים קדמו להקמת ה"מקור" והביאו מספר משתמשים למסקנה שיש להקים ארגון גג לכל הקבוצות הפעילות, הארוע הראשון היה הארגון של "אוגוסט פינגווין" (כנס משתמשי לינוקס) אשר בצוות הארגון שלו היו מולי בן יהודה וגלעד בן יוסף. הארוע השני היה כאמור הצורך לארגן קבוצה מאוזנת מבין קבוצות המשתמשים על מנת לנסוע לדיון בכנסת , אשר על דורון אופק היה לארגן צוות זה.

עמותת ה"מקור" – עמותה ללא כוונות רווח אשר מרכזת את הייצוג של כל קהילת הלינוקס הישראלית על שלוחותיה השונות , עמותה זו הינה הממשק של קהילת הלינוקס בכללותה אל העלום שאינו מכיר את התוכנה החופשית, בבסיס הרעיון עמותה זו מקדמת את השימוש בלינוקס ובקוד פתוח בארץ .

פעילויות העמותה , תקשורת עם הקהילה דרך הערוצים אשר היו קיימים (דהיינו פורומים, אתרים, רשימות תפוצה). הקמה , ניהול וסיוע לפרוייקטים שנועדו לקדם את השימוש בתוכנה חופשית , בכלל זה סיוע לפרוייקט ה OO של משרד האוצר, סיוע לפרוייקט "כנרת" וקשר עם משרד החינוך בנושא זה , קיום מפגשים על בסיס קבוע של כל הקהילה הישראלית ומתן סיוע לקיום מפגשים של מועדוני משתמשים מקומיים (מועדון בחיפה ומועדון בתל אביב). בנוסף על כך העמותה ממלאת מקום בנושא של תגובות לתקשורת ומהווה צינור הידברות בין התקשורת המקומית והעולמית לקהילת הלינוקס הישראלית. ולבסוף , סיוע לתפעול השוטף של האתרים הקהילתיים , טיפול שוטף באתרי mirror שנמצאים בארץ וכד'.



שם המשחק – פתיחות לטובת אי תלות (מודל ה TCP IP)

לינוקס בפרט ותוכניות קוד פתוח בכלל יוצרים שקיפות שבמהלכה נחשף המשתמש לקוד המקור של התוכנית.

שקיפות זו מסייעת ושומרת על כך שכל הפיתוחים שנעשים הינם על בסיס תקנים קיימים (נקודה זו נסקרה כבר קודם לכן)

למעשה המערכת מספקת שקיפות של המידע , בצורה זו המשתמש (הצרכן) אינו תלוי בצורה חד חד ערכית ביצרן מסויים.

אין הדברים אומרים שלא כדאי להתחבר ליצרן – אלא שהלקוח יכול לשנות את צורת עבודה שלו ולנייד את המידע ביתר קלות כאשר הוא עובד עם לינוקס .

בסופו של דבר , הלקוח יכול להיות ורוצה להיות חופשי יותר בבחירה שלו , בחירתו של ספק אחד על פני ספק אחר אינה תולדה של תלות באפליקציה מסויימת או בטכנולוגיה שרק היצרן מכיר אותה , אלא תולדה של איכות השירות המסופקת על ידי הספקים שונים.

הוזלת עלויות – הגברת התחרות

כל המתואר בסעיף הקודם יוצר מצב של תחרות , מצב שבו הלקוח הוא זה שמקבל את ההחלטה , והוא לא עושה זאת על בסיס של אילוץ.

הוזלה של עלויות היא פועל יוצא של התחרות הזו – דבר שממנו נהנה הצרכן.

עם זאת קיימות מספר בעיות או מספר נקודות שיש לתת עליהן את הדעת כאשר הולכים למודל שכזה והכוונה היא בעיקר ליצרנים וספקים אשר צריכים לבדוק את המודל הנוכחי שלהם ביחס למודלים הקיימים בעולם הקוד הפתוח.

כאשר מוצרים רבים אינם עולים כמעט כסף – הדרך הכמעט יחידה היא להרוויח משרותים אשר ניתנים כמעטפת לאותם מוצרים.

כמובן שגם על התשתית של לינוקס וקוד פתוח ניתן לשלב תוכניות קוד סגור – אך אז הדבר די דומה למודלים הקיימים כרגע בשוק.

בסופו של יום , מי שזוכר את צורת העבודה ודרך ההתנהלות של גופים שונים לפני כ 15-20 שנה , היו מצבים בהם הלקוח הרגיש לא פעם שבכלל עושים לו טובה שמוכנים לדבר איתו .

המצב כיום הוא כמובן שונה והוא אף יהיה יותר שונה בעתיד , הצרכן אינו מישהו שצריך להתייחס אל עצם קימו בטבעיות , אלא להשקיע בו על מנת שימשיך ויעבוד עם הארגון.

מוצר איכותי יותר בהיבטי -

- גמישות
- יציבות
- אבטחת מידע (ההאקרים שותפים לפיתוח)
- ניקיון והנדסת תוכנה

כי :

- הקוד חשוף בפני עיניים בוחנות יותר, במשך זמן רב יותר, לפני שהוא משתחרר
- מפתחי הקוד מקפידים יותר על איכות, בגלל התהליך

תחזוקה איכותית יותר כי -

- כח אדם מקצועי יותר – צבירת ידע חוצת פלטפורמות,
- הבנה עמוקה של המערכות כתוצאה מחשיפה לקרביים (זה גם סיכון...)
- יכולת להתאים את המערכת למצבי קצה, חומרות, **devices** בקלות יתרה.

איפה כדאילהתחיל ?

- סביבת התשתיות והשרתים
- סביבת הפיתוח
- סביבת תחנות העבודה
- מוצרים ייחודיים – תוכניות חופשיות כתחליף למוצרים קניינים
- עלייה ביעילות המערכות , עלייה ביכולות האבטחה

- עלייה ביעילות המערכות , עלייה ביכולות האבטחה , עלייה באיכות הפתרון
- גמישות
- **ROI**

סביבת השרתים

- החלפת שרתי ה **UNIX**, ניתן לחסוך בקלות בכל ההבטים : חומרה, רישוי, תחזוקה – ROI מהיר ביותר !
- החלפת סביבת ה **Windows** – מאפשרת, יציבות גדולה יותר , אבטחה טובה יותר ומערכות יעילות יותר (אבל גם סוגיית ההגירה קשה יותר)

סביבת המפתחים

- גישה לכמות כלים גדולה יותר
- גישה נרחבת יותר לקוד והיכולת לנצל קוד כתוב לשם המחזור שלו.
- לא רלוונטי למפתחי מיקרוסופט !

סביבת תחנות העבודה

- יש ללינוקס חסרונות רבים בנושא של ידידותיות
- יש ללינוקס בעיה בזמינות של תוכנות (בחלק מן המקרים)
- אם ניתן לקיים סביבה בעלת אוטומציה גבוהה , בסביבה מקובעת (למשל לקוחות של **terminal server**) – אפשרי
- **ROI** – לא מובהק

מוצרים ייחודיים

- זמינות גדולה ובלתי מוגבלת של מערכות ותוכניות **HPC** , תוכניות אבטחה שונות (**IDS** , מערכות **firewall** , שרתי **proxy** וכיוצ"ב) , זמינות של תוכניות **HA** ו – **LB** , תוכניות אבטחה מבית היוצר של ה **NSA** .
- זמינותו של הקוד ושקיפותו יצרה מצב שבו תוכניות ופרוטוקולים שנוצרו במקור בעולם הקינייני הפכו בעולם הקוד הפתוח ליעילות יותר , לדוגמה שרת **SAMBA** .
- בנושאים של אבטחה – פרוייקטים כגון **Selinux** זמינים לארגונים שונים , תוכנית זו אמורה לפי מתאר **Red Hat** להיכנס אל הדור הבא של משפחת ה **Enterprise** .

מודל ה-ROI

עלות המעבר לקוד הפתוח

מינוס

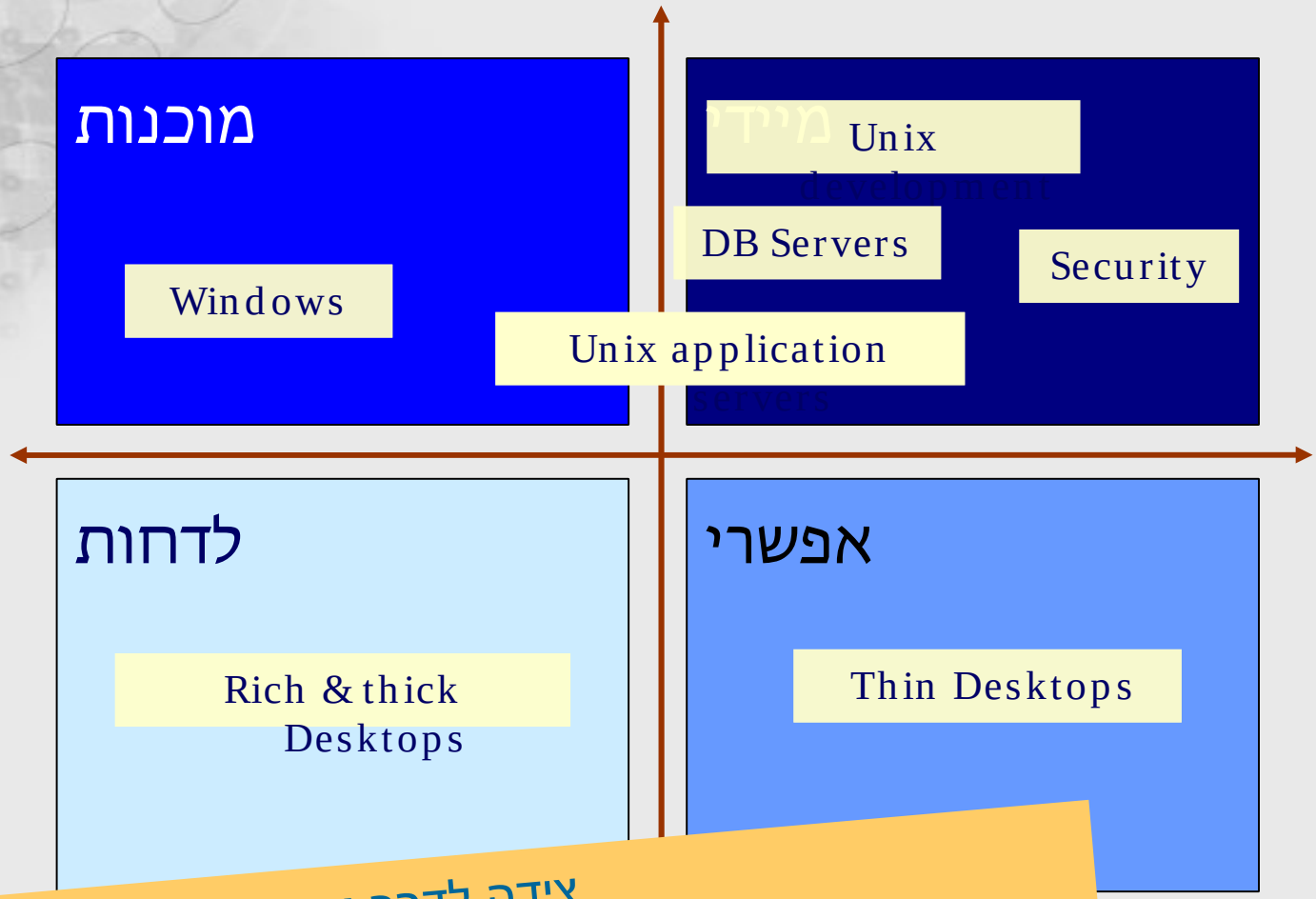
(עלות הבעלות החדשה – עלות הבעלות האלטרנטיבית)

פלוס

הערכים המוספים של יכולות ועבודה


רמת קריטיות/ תועלת גבוהה
מוכנות
מיידי
ישימות נמוכה - מאמץ גבוה
ישימות גבוהה - מאמץ נמוך
לדחות
אפשרי
רמת קריטיות/ תועלת נמוכה

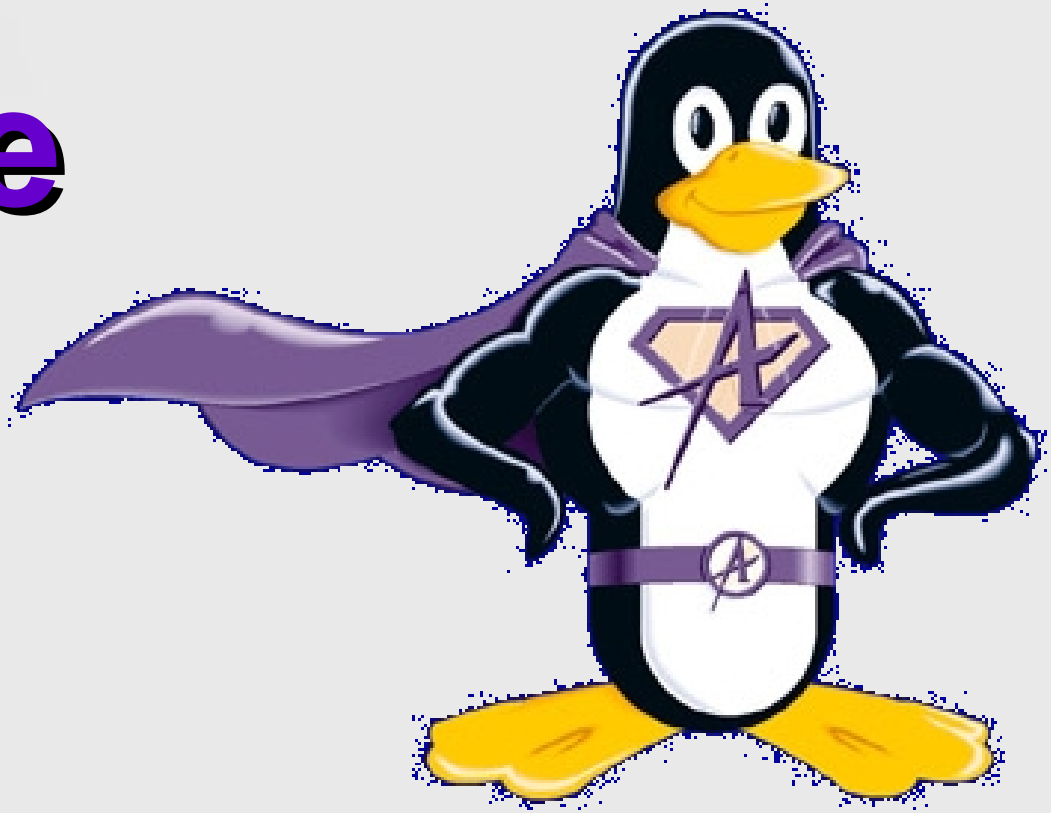
רמת קריטיות/ תועלת גבוהה



צידה לדרך :

כל פרוייקט חדש – לחשוב **linux** ולהוריד עליות במוקדי עלות גבוהים לבדוק הגירה ל-**linux**

Software as service



מה אומרים נגד הקוד הפתוח

- מה שניתן בחינם עולה בסוף יותר
- אופנה חולפת
- אין "אבא"
- אין תמיכה וארגון לא יכול להסתמך על מוצר ללא "אבא"
- עולה המון לעבור לסביבה החדשה
- יש בעיות אבטחת מידע והם רק יגדלו
- אין יישומים ואין אנשים

software as service

- משלמים על תחזוקה ואריזה – מבטיח יציבות ושירות
- ללכת בלי (מערכת הפעלה מסחרית) להרגיש עם
- Redhat הספק המוביל והוותיק בעולם (גם באירופה)
- Suse – לאחר רכישה ע"י נובל – המתחרה הבא אחריו

מה מקבלים ב-Redhat ?

- יכולת תמיכה מוכחת , יכולת לספק שינויים ושיפורים אשר אינם נתמכים בגרסאות החופשיות – הציבוריות ואשר יש להם ביקוש בשוק העסקי
- מתן מענה בנוגע לחומרות נפוצות בשוק העסקי ואשר עדיין אינן נתמכות ב kernel הרגיל.
- מתן "הסמכה" לחומרה ותוכנה אשר נמצאת בשימוש של השוק העסקי.
- מתן שירותים נלווים כגון ייעוץ ומסלולי קורסים – אשר הינן צורך בשוק העסקי
- טיפול בבעיות אבטחה , ביצוע מעקב של מאות אנשים מטעם המפיץ על אלפי רשימות תפוצה ומקורות מידע –לשיחרור תיקונים מרגע שבוצעו.
- בהמשך לסעיף 5 , גם מתן מענה של טיפול אוטומטי בתחנות /שרתים לנושאים הללו – מצריך רישום ב RHN (לחילופין אפשרי דרך ניהול של שרת מקומי שכזה)
- בנוסף על הקהילה עצמה , המפיצים לוקחים על עצמם אחריות כלפי הלקוחות שלהם.
- זיהוי טכנולוגיות חדשות או כלים חדשים , והכנסה מבוקרת שלהם למוצר.
- פעילים במקומות שבהם הקהילה אינה יכולה לפעול – למשל ביצוע מבחני הסמכת אבטחה למערכת מול גופי ממשל שונים בארה"ב

Competitive Positioning

With Red Hat at $90 million in revenue and slightly profitable, no other Linux distributor is close to it in size. SuSE, next in size with about $40 million in revenue, still is not profitable and has had management turnover, with a new CEO in January 2003.

… Red Hat's strategic advantage is its knowledge base. Much of its engineering talent derived from the original brainpower developing Linux with Linus Torvalds and still retains connections with or has active involvement in the kernel community of maintainers.

....

At this stage, Gartner feels UnitedLinux will remain a second-class .competitor,with limited market share through 2005

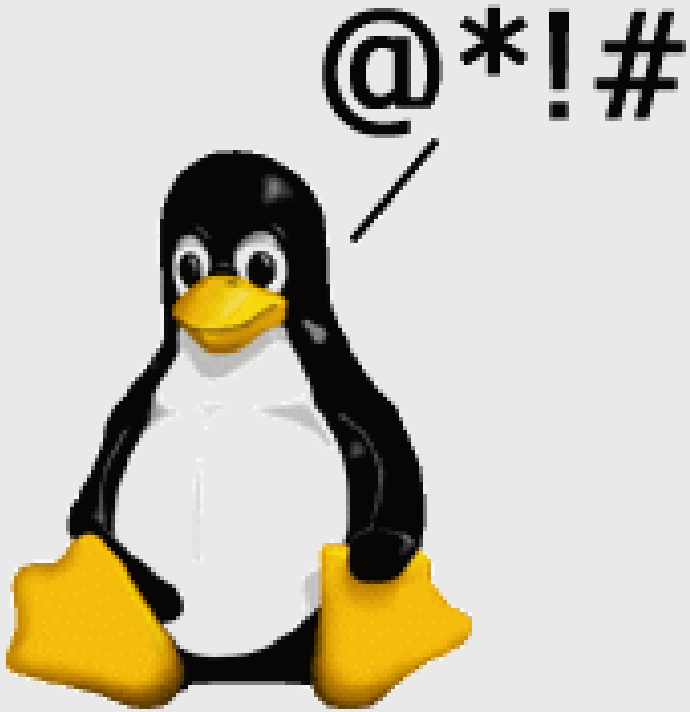
Gartner Group, march 2003

שלבים במימוש

תהליך של הגירה הינו תהליך רב שלבי וכדאי לעשותו בדירוג מסויים :

- בדיקת התשתית הפיזית הקיימת , בדיקת תשתית האפליקציות שנמצאות בשימוש ומיפוי של חומרה ושל תוכנה .
- בדיקת הפרוייקט וביצוע הערכות ראשוניות לגביי הכשרת אנשים , להתחיל מסלולי הכשרה והסמכה מותאמת תפקיד בנושאים הרלוונטיים לבעלי התפקידים השונים.
- להכין תוכנית מדורגת לביצוע של החלפה של תשתיות פיזיות ותשתיות חמרה .
- לבצע תוכנית להחלפת האפליקציות השונות.
- להתחיל בתוכנית הגירה רב שלבית כאשר בין השלבים מתבצעת עצירה על מנת לבדוק עמידה ביעדים , לאפיין בעיות להסיק מסקנות.
- מעגלי ההגירה יכולים להיות אורכיים או רוחביים – כלומר לבצע הגירה של קבוצות (מחלקות) - הגירה אורכית , או לבצע הגירה של חיתוכים (לדוגמה , שרתי ה Database , שרתי הדואר .. וכן הלאה) - הגירה רוחבית.

Tips &Tricks



- התקנה אוטומטית של מערכות לינוקס באמצעות קובץ kickstart הופכת את ההתקנה להליך פשוט ומהיר כאשר מדובר בשורה ארוכה של מחשבים או במצב בו לא נרצה שהלקוח/מתקין יתערב בהליך ההתקנה.
- כל התקנה של מערכת משאירה ב root קובץ בשם anaconda-ks.cfg קובץ זה הינו ה kickstart של המערכת שזה עתה התקנו.
- קיים היישום system-config-kickstart אשר מסייע לבנות בצורה מהירה וקלה ks.cfg .
- השמת ה kickstart לאחר שהוא מוכן אל תוך הליך ההתקנה יכול להתבצע במספר אופנים – בכל מקרה מדובר בפרמטר שמועבר ל kernel של ה installer .

- הפרמטר ks יכול להיות מקומי או רשתי, דוגמאות:
  - linux ks ks=floopy
  - linux ks ks=http://server.example.com/workstation.cfg
- הוא יכול להיות מועבר או ע"י ממשק ה CLI בשלב איתחול ההתקנה או ע"י שרת DHCP.
  - filename "/kickstart/workstation.cfg";

  next-server server.example.com;
- ניתן לבנות גם KS לכל מחשב וזאת ע"י מתן שם תיקיה כפרמטר ובניה של KS בתוך כל תיקיה על פי ה IP של התחנה.

- לפיכך:
  - התקנת kickstart יכולה להתממש כהתקנה שבה המתקין אינו צריך להתערב בהליך ההתקנה – עם זאת המתקין מתחיל את תהליך ההתקנה.
  - התקנת kickstart יכולה להתממש גם ללא מגע יד אדם כלל וזאת ע"י בניה נכונה של הסביבה (דבר שהוא עשוי להיות משתמעותי בסביבות מרובות מחשבים).

- 2 פרמטרים נוספו להליך ההתקנה בתקופה האחרונה וזאת על מנת לאפשר למשתמשים לבצע התקנות מרוחקות.
- ככלל התקנות מרוחקות אפשריות היו גם בטרם נוספו פיצ'רים אלו אולם פיצ'רים אלו מגדילים את הנוחות של המשתמש בביצוע התקנות שכאלו.
- למעשה מופעל ע"י ה installer של המערכת נושא של תצוגה גרפית המועברת אל מסך מרוחק. בעזרת תצוגה זו וכאשר השליטה על הליך ההתקנה היא במסך המרוחק מתאפשר לי כמשתמש להתקין מערכת ללא הימצאות בפועל לידה.

- 2 הפרמטרים הינם:
  - display=IP:1
  - vnc
- הפרמטר display מבצע זריקה של התצוגה למסך מרוחק, דבר שמצריך שימוש ב rhost + שיטה שאינה בטוחה ואם משתמשים בה יש לוודא את אבטחת הסביבה.
- הפרמטר vnc פותח למעשה vnc server מה שמאפשר להתחבר אליו מרחוק ומאפשר להמשיך את הליך ההתקנה בצורה מרוחקת.

- שימוש ב LVM :
- מערכות לינוקס תומכות ב LVM וניתן הן בהליך ההתקנה והן בשלבים מאוחרים יותר להוסיף מחיצות LVM למערכת ולהגדיל מחיצות לוגיות קיימות.
- החל מגרסת Enterprise 4.0 של Red Hat הליך ההתקנה משתמש ב LVM בתצורת ברירת המחדל (הגרסה צפויה לצאת ב Q 1/2005) של ההתקנה.

- ישנם מספר כלים אשר מאפשרים לשפר את הביצועים של המערכת בחיי היומיום.
- פרנטרים אשר מועברים ל kernel של המערכת.
- כלים שנכנסים לפעילות בזמן איתחול.
- פרמטרים אשר מועברים אל שירותים שונים בזמן איתחול המערכת.

- יצירת prompt צבעוני על פי המשתמש המבצע login למערכת:
  - שימוש ב /etc/bashrc או ב /etc/profile.d/file.sh כדי לנתח מיהו המשתמש ועל פי זאת לקבע את המשתנה SP1 .

- בדיקה / שיפור עבודת דיסקים ע"י התוכנית hdparm

```
root@localhost AS-3.0# hdparm -Tt /dev/hda

/dev/hda:

 Timing buffer-cache reads:   1052 MB in  2.00 seconds =
   525.82 MB/sec

 Timing buffered disk reads:   88 MB in  3.00 seconds =
   29.31 MB/sec
```

- proc הינה תיקיה המהווה ממשק אל ה kernel הפעיל.
- מתוך תיקיה זו ניתן לקבל חיווי על עבודת המערכת והפעילות
- בתוכה קיימת התיקיה proc/sys המאפשרת להעביר פרמטרים אל ה kernel ולפיכך לעצב ולשנות את התנהגותו.
- שימוש ב echo ישפיע באופן זמני בלבד (עד לביצוע כיבוי או איתחול).
- על מנת לקבע הגדרה נוכל להכניסה אל etc/sysctl.conf

- proc הינה תיקיה המהווה ממשק אל ה kernel הפעיל.
- מתוך תיקיה זו ניתן לקבל חיווי על עבודת המערכת והפעילות
- בתוכה קיימת התיקיה proc/sys המאפשרת להעביר פרמטרים אל ה kernel ולפיכך לעצב ולשנות את התנהגותו.
- שימוש ב echo ישפיע באופן זמני בלבד (עד לביצוע כיבוי או איתחול).
- על מנת לקבע הגדרה נוכל להכניסה אל etc/sysctl.conf

- תיקיית proc הדגמה
- הקובץ etc/sysctl

- התוכנית RPM נבנתה ע"י Red Hat בעבר ומשמשת כמנהל חבילות התקנה של הפצות שונות.
- בעבר, השימוש בתוכנית זו היה מוגבל יחסית בגלל יכולות חסרות שהיו קיימות בה.
- כיום פיצ'רים חדשים מוספים יכולות לתוכנית ה RPM .
- התקנת חבילה:

- rpm -i rpm_file.rpm

- הסרת חבילה:

- rpm -e rpm_file.rpm

- שידרוג חבילה:
- rpm -U rpm_file.rpm
- שאילתה על מידע כללי:
- rpm -qi rpm_file.rpm
- שאילתה על קבצים שהחבילה מוסיפה למערכת:
- rpm -ql rpm_file.rpm
- פתירת נושא של תלויות:
- rpm -Uvh rpm_file.rpm --aid

* פרמטר aid שונה מהותית מ

- עבודה מול web או ftp :
- rpm -Uvh http://server_name/directory/rpm_file.rpm
- שימוש בקובצי macro, ניתן לבנות קובצי מקרו המגדירים את מיקום המקורות כך שהתקנות ה RPM יהיו ממקורות רשתים קבועים.

- שימוש ב hosts.allow או ב hosts.deny
- שימוש ב netfilter
- שימוש ב ACL
  - ביצוע mount עם הפרמטר ACL
  - שימוש בפקודות getfacl; setfacl

- שימוש ב ssh כתחליף לשרתי ה Rhost
- שימוש ויצירת מפתח
- יצירת authorized_keys ו – authorized_keys2
- שימוש ב ssh לביצוע login או להרצת פקודות מרחוק
- שימוש ב scp או sftp

תודה רבה