



TEL AVIV UNIVERSITY

מערכות הפעלה

מרצה אורח: חזי ישורון

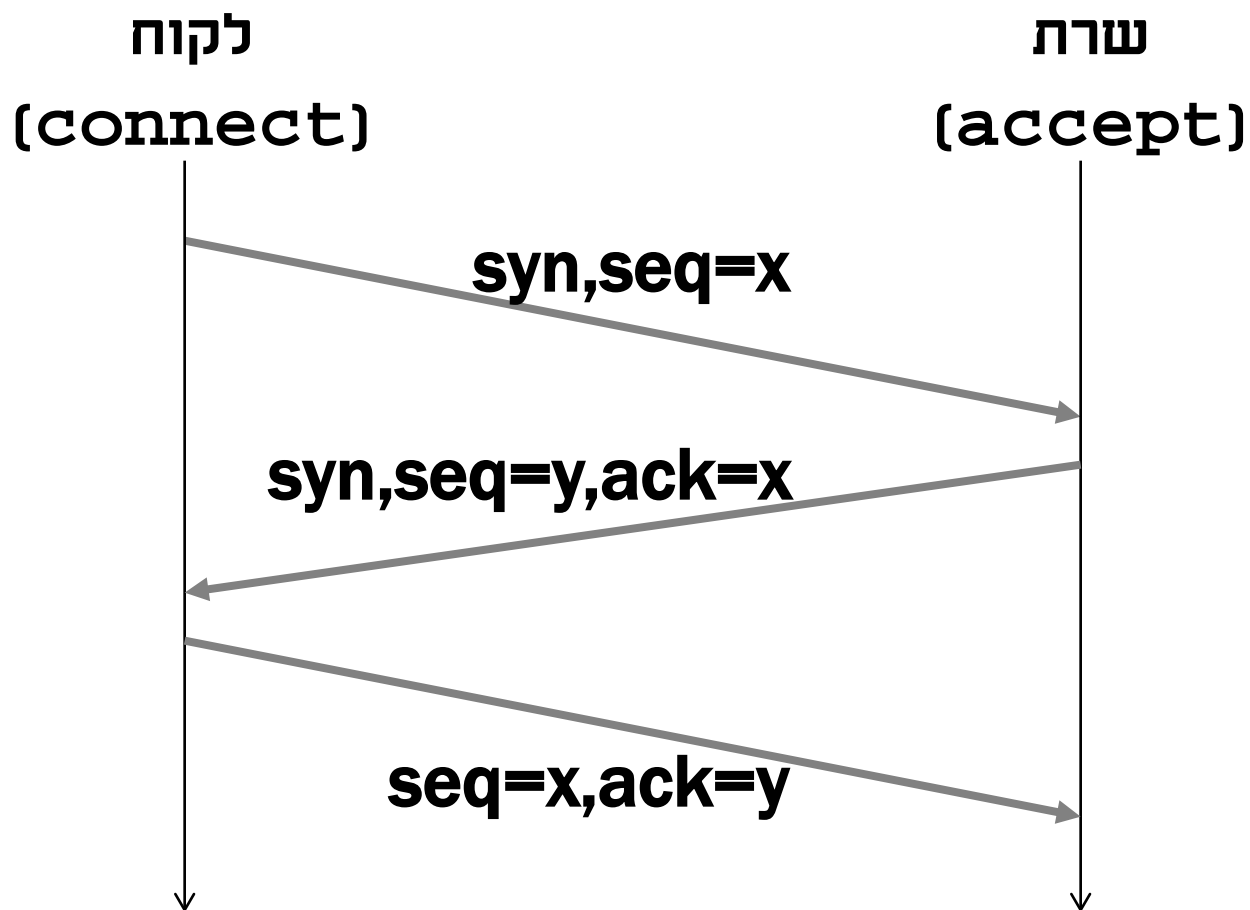
סמסטר א' תשע"ב

הרצאה 11

פרוטוקול TCP (המשך)

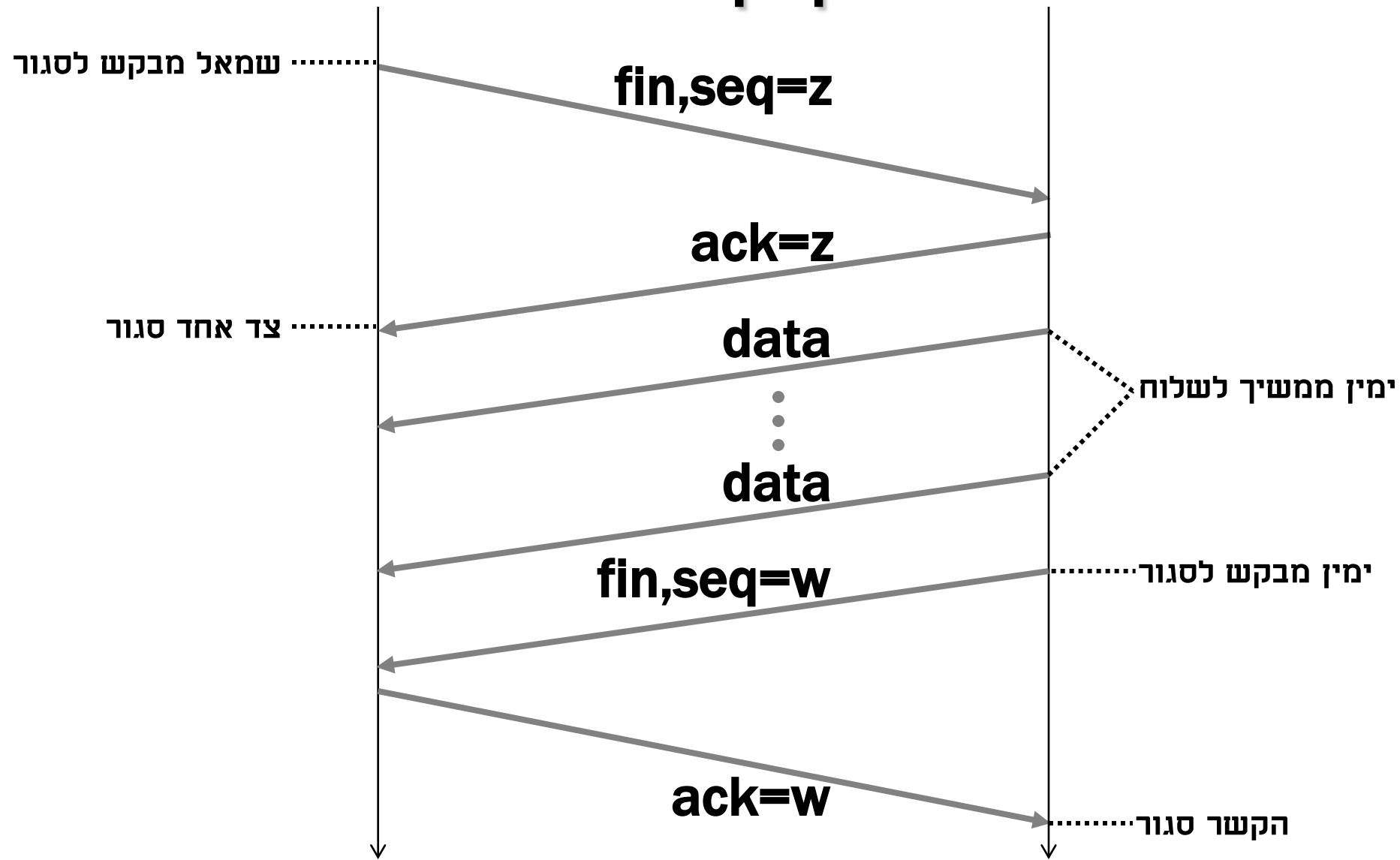
מערכות קבצים מבוזרות

יצירת קשר TCP



הרצפים מתחילים ממיקום אקראי כדי להקשות על התחזות

פירוק קשר TCP



מנות מקשר מפורק

- ❖ מנות שהתעכבו ברשת עלולות להגיע לאחר שהקשר פורק
- ❖ אם האישור ל-fin אובד, מנת חן נוסף תגיע בהמשך למחשב שאישר קבלת חן ושמבחינתו הקשר סגור
- ❖ אי לכך, מערכת ההפעלה שומרת על מבנה הנתונים שייצג את הקשר זמן מה לאחר סגירתו כדי שניתן יהיה לטפל במנות כאלו
- ❖ כדי שניתן יהיה לטפל במנות כאלו, מערכת ההפעלה אינה מתירה ליצור קשר נוסף עם אותם פרמטרים (כתובת ip ומספר שער מקומיים ומרוחקים) באותו פרק זמן
- ❖ מערכות הפעלה רבות אינן מאפשרות למחזר את הפרמטרים המקומיים; ניתן להורות להן למחזר על ידי קריאה ל-setsockopt

מנות מקשר שנשכח

- ❖ שמירה לזמן מה על מבנה הנתונים של הקשר לאחר סגירתו מאפשרת לטפל ברוב המנות שמגיעות אחרי סגירה
- ❖ אבל לעיתים מנות עלולות להגיע זמן רב לאחר סגירה
- ❖ בנוסף, תקלות במחשבים אחרים (או ניסיונות שיבוש מכוונים) עלולות לגרום לקבלת מנות של קשרים שלא היו קיימים מעולם
- ❖ מערכת ההפעלה עונה למנות כאלה במנת מיוחדת שמסומנת בסיבית reset
- ❖ קבלת מנה כזו צריכה לגרום לשולח להפסיק לשלוח מנות בקשר

מודעות לקיום קשר

- ❖ כאשר אין צורך להעביר מידע בקשר TCP, מנות אינן נשלחות כלל
- ❖ התנהגות זו גורמת לחוסר מודעות להתנתקות הקשר עקב נפילת אחד הצדדים ועקב התנתקות של הרשת
- ❖ ביישומים שבהם דרושה מודעות לקיום או אי-קיום הקשר, ניתן להורות למערכת ההפעלה לשלוח מנה מדי פעם לבדיקת קיום
- ❖ המנגנון מופעל באמצעות שירות היקיצה `keepalive`

פרק 9

מערכות קבצים מבוזרות

מערכות מבוזרות

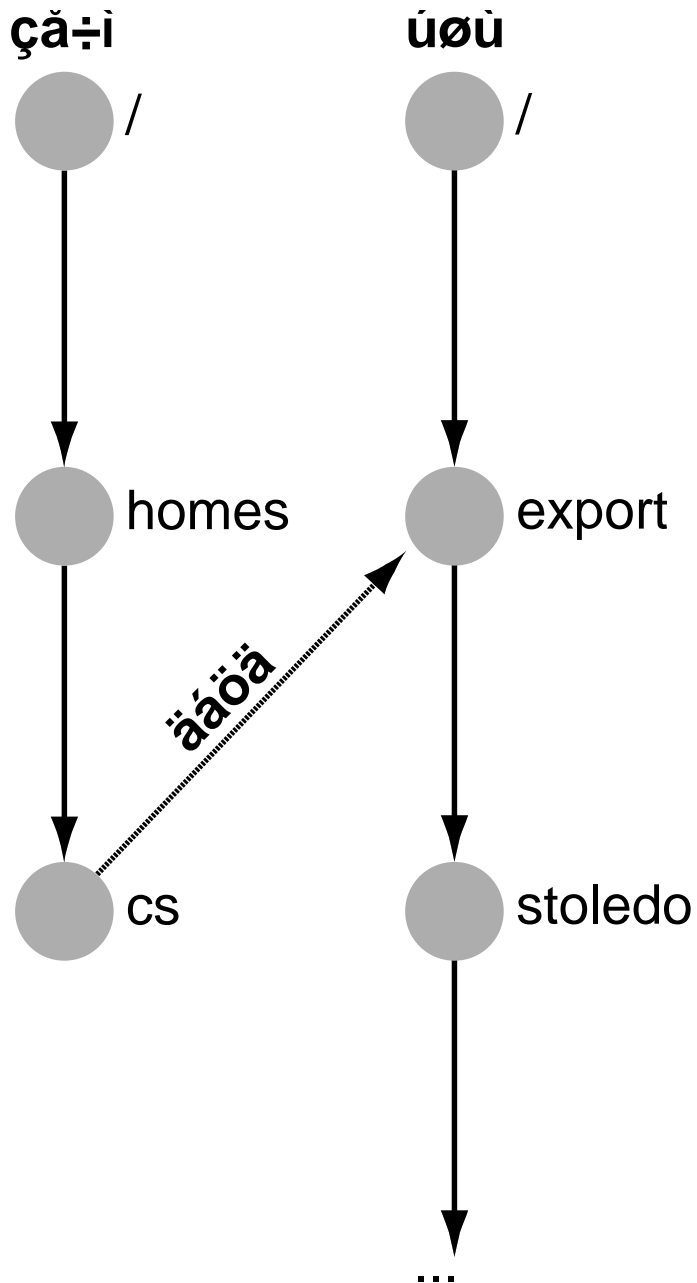
- ❖ קבוצה של מחשבים וציוד אחר (מדפסות למשל) מחוברים ברשת תקשורת.
- ❖ אנו רוצים לאפשר למשתמשים לנצל חומרה דרך הרשת ולא רק חומרה מקומית.
- ❖ בדרך כלל לא נפעיל את כל החומרה תחת מערכת הפעלה אחת.

למה לא מערכת הפעלה אחת?

- ❖ במחשב יחיד כל רכיבי החומרה פועלים או שהמחשב מכובה/מקולקל.
- ❖ במערכת מבוזרת יתכנו מצבים רבים שבהם חלק מהרכיבים מקולקלים או כבויים ועדיין אנו רוצים להשתמש בשאר.
- ❖ במחשב יחיד ההגנה של מערכת ההפעלה מוחלטת.
- ❖ במערכת מבוזרת עלולות להסתנן לרשת הודעות שלא סונו על ידי מערכת ההפעלה.
- ❖ צריך להתחשב באיטיות רשת התקשורת.

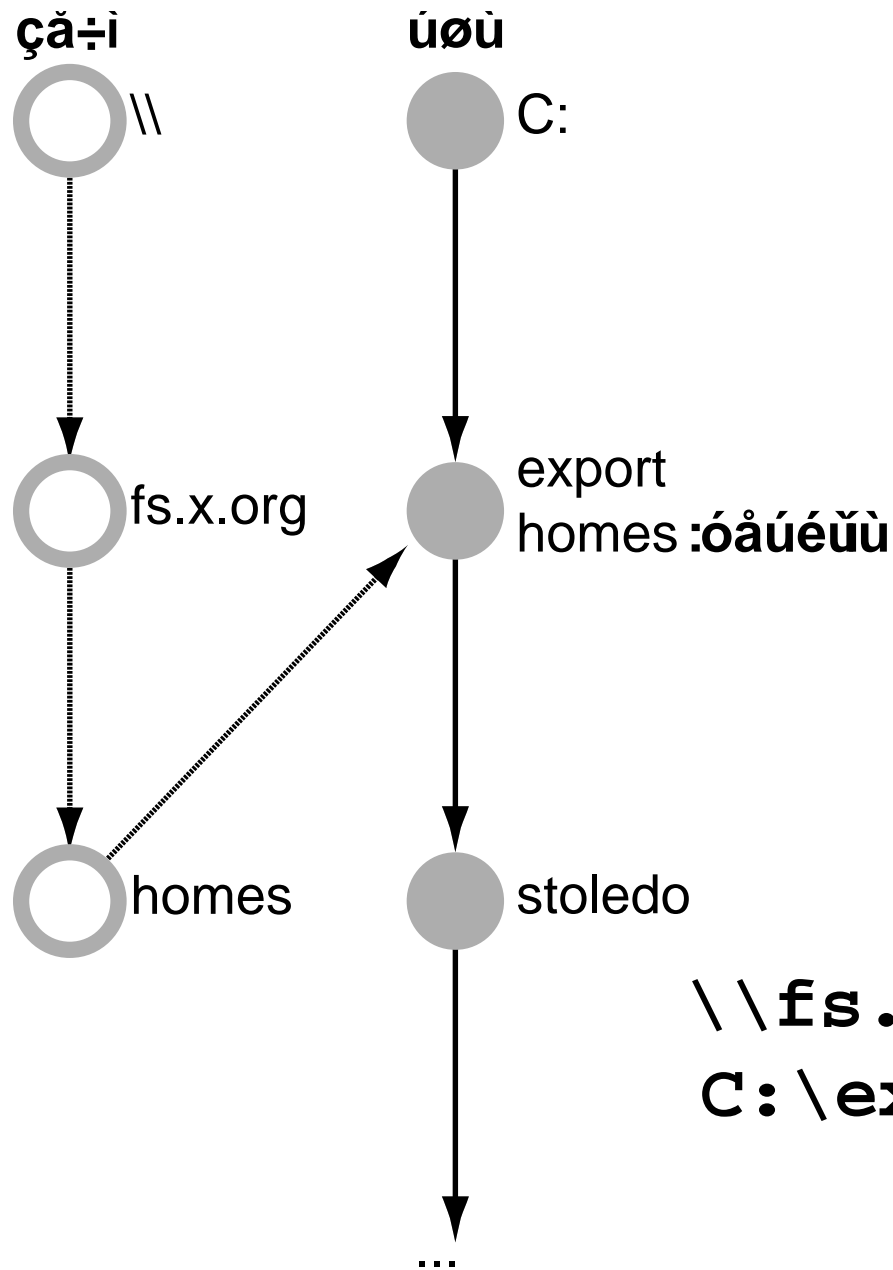
מערכות קבצים מבוזרות

- ❖ מאפשרות לגשת ממספר מחשבים (לקוחות) לקבצים שמאוחסנים במחשב אחר (שרת).
- ❖ מאפשרות לפזר עומס חישובי בין לקוחות.
- ❖ מאפשרות למשתמשים גישה לקבציהם ממקומות גיאוגרפיים שונים (מחדרים שונים ועד יבשות שונות).
- ❖ לעיתים קל יותר לנהל קבצים בשרת מרכזי (למשל, קל יותר להבטיח שקבצים יגובו או שתהיה יתירות).
- ❖ לעיתים זול יותר לרכוש נפח דיסקים עבור שרת מרכזי. למשל, יתכן שאין צורך בדיסק שלם עבור כל תחנת עבודה. מאידך, לעיתים דיסקים לשרתים יקרים יותר מדיסקים לתחנות עבודה.



הצבה שרירותית במרחב השמות

מערכת הקבצים ששורשה ב-
/export בשרת מוצבת על גבי
/homes/cs בלקוח



שיקוף שם שרת במרחב השמות

בלקוח, השם

`\\fs.x.org\homes\stoledo`

מתייחס ל-`C:\export\stoledo`

בשרת `fs.x.org`

עוד על שמות קבצים מרוחקים

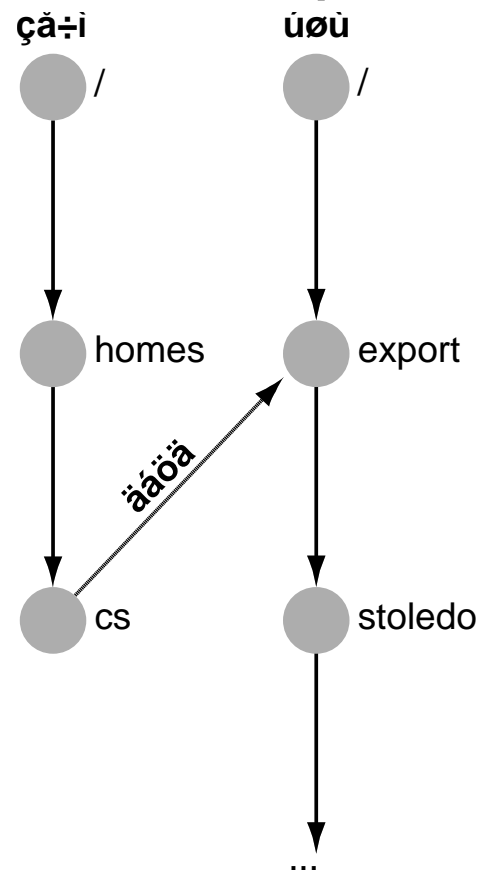
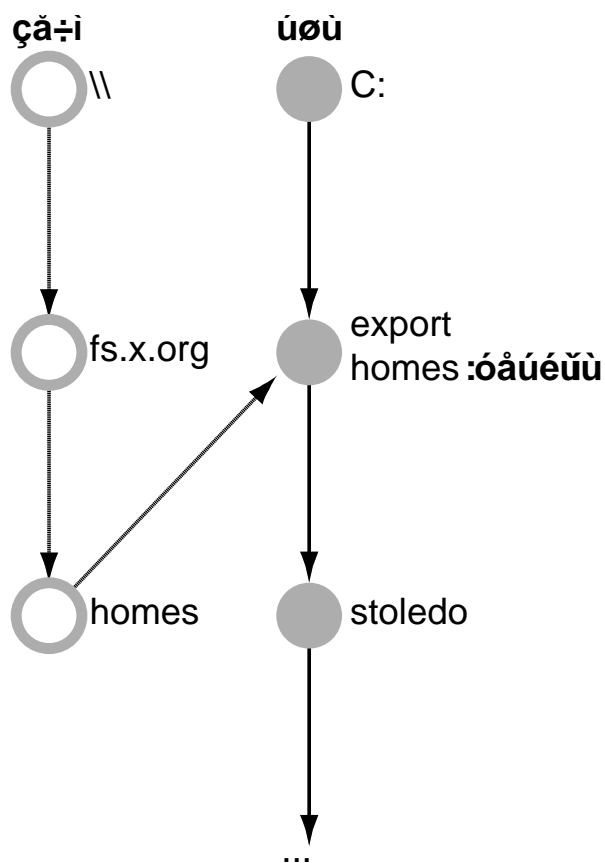
- ❖ מערכת AFS בעולם היוניקס:
`/afs/ibm.com/stoledo/a.out`
- ❖ בחלונות ניתן להציב קבצים מרוחקים רק באות כונן; החל מחלונות 2000 ניתן להציב בנקודה שרירותית.
- ❖ שרת חלונות משתמש בשם חיצוני (share name) עבור תתי מרחבים שהוא מספק.
- ❖ שיקוף שמות שרתים משמש גם בגישה לקבצים באינטרנט דרך URL-ים, לגישה לדואר אלקטרוני בפרוטוקולי POP ו-IMAP, וכדומה.

השוואה

שם קובץ משקף את השרת	הצבה שרירותית
שם הקובץ מכיל גם את מיקומו	שם הקובץ מציין את רק את תוכנו
המשתמש יכול לגשת לכל שרת שיש לו הרשאות לגשת אליו	הצבה דורשת לפעמים התערבות של מנהל המערכת
שם קובץ קבוע	בלבול אם לא ניתן להציב במקום מסוים או אם קבצים מוצבים בנקודות שונות בלקוחות שונים
משתמשים מודעים להחלפת שרת	החלפת שרת ועדכון ההצבה ללא מודעות של המשתמשים

פענוח שמות קבצים מרוחקים

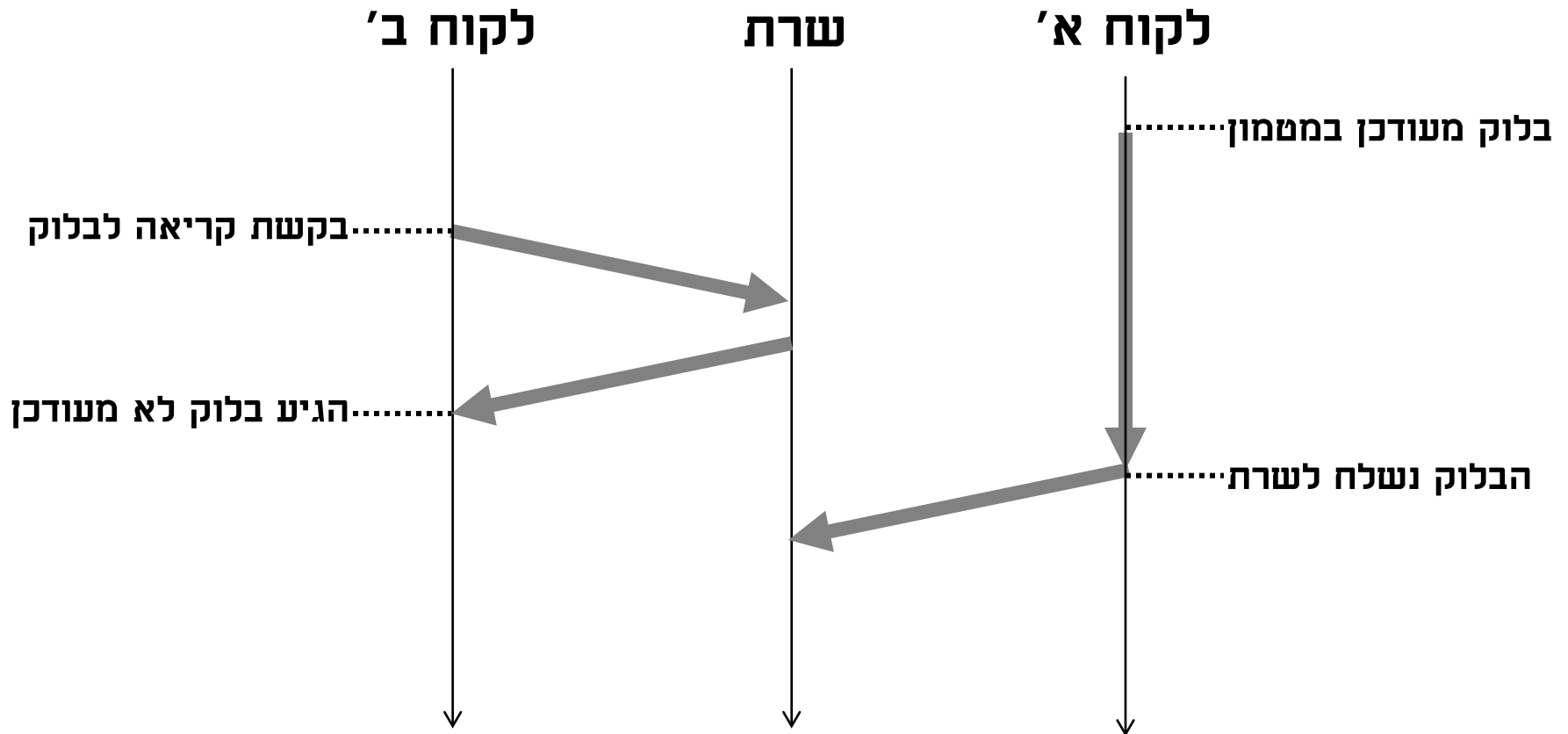
מערכת ההפעלה של הלקוח מזהה נקודת הצבה של מערכת קבצים מרוחקת או תחילית שמציינת שמות מרוחקים, שולחת את סוף השם לשרת לפענוח, ומקבלת חזרה מזהה לקובץ.



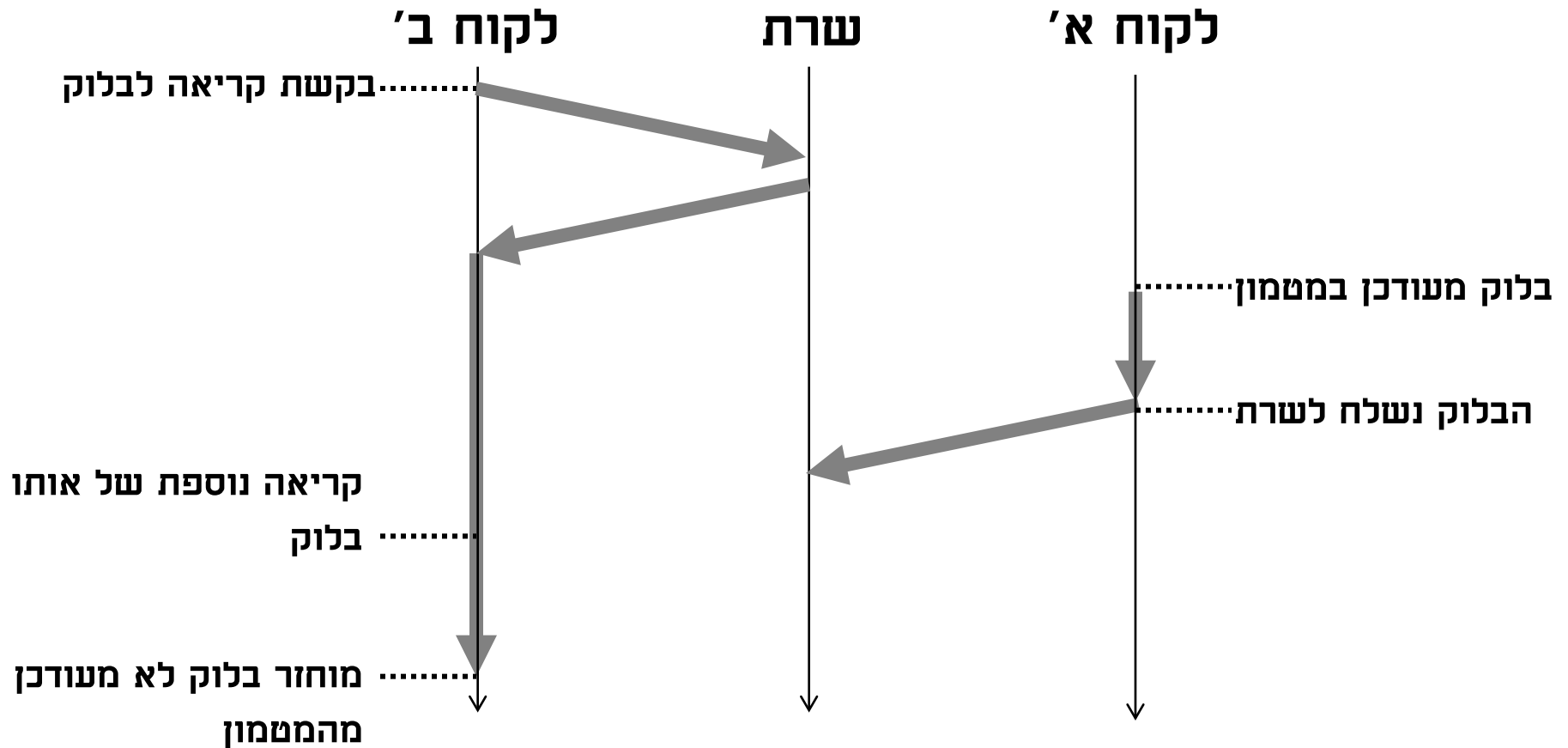
הטמנת נתונים בלקוח

- ❖ הלקוח מטמין נתונים (לקריאה חוזרת או ולחציצת כתיבות) על מנת להתגבר על שתי בעיות:
 - תקשורת איטית בין השרת והלקוח
 - עומס על השרת ממספר לקוחות
- ❖ דוגמה: רשת של 100 Mb/s לעומת דיסק מקומי עם קצב העברה של של 160 Mb/s = 20 MB/s
- ❖ דוגמה: רשת של 1 Gb/s ושרת עם מערך דיסקים עם קצב העברה כולל של 800 Mb/s = 100 MB/s

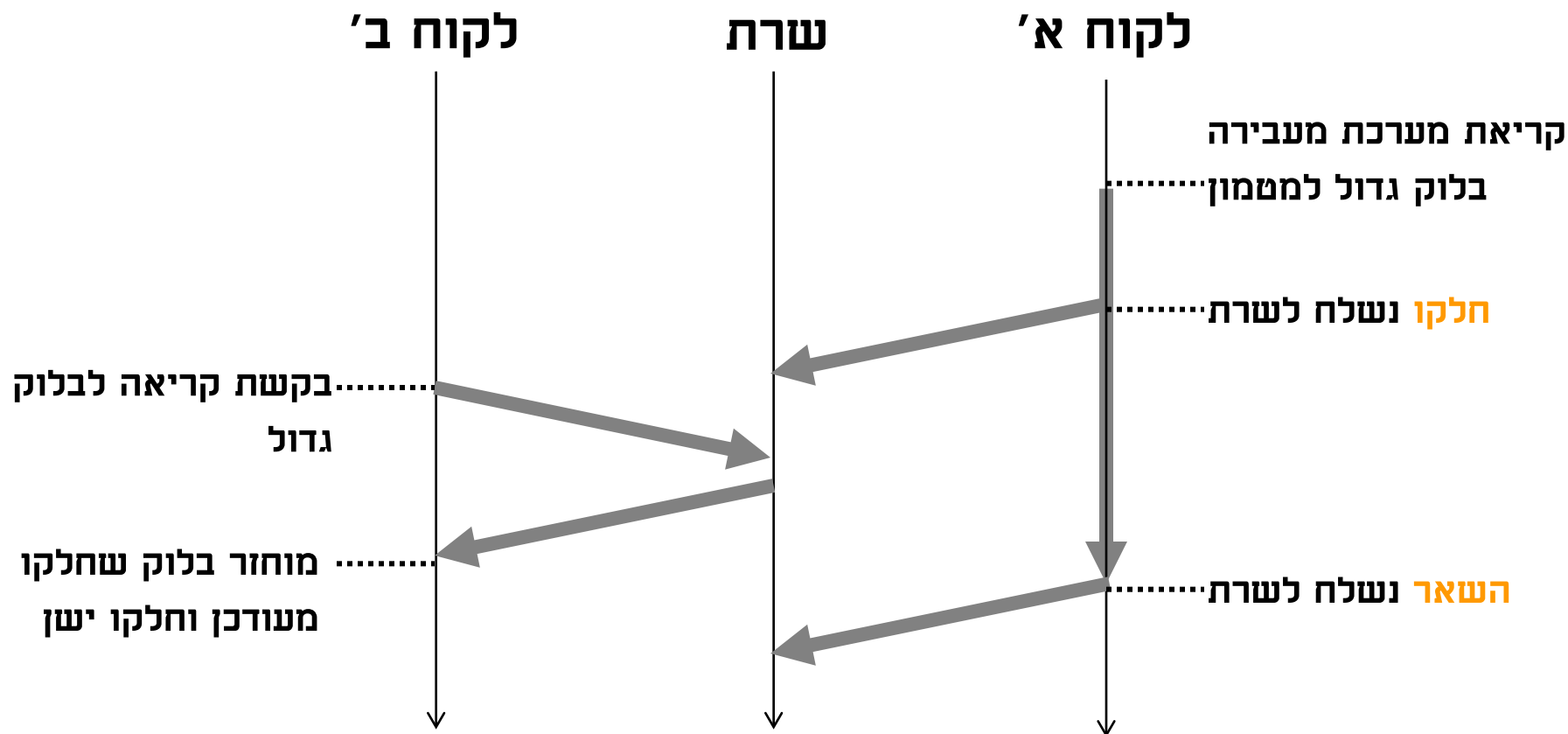
הטמנה עלולה לפגוע בסמנטיקה (1)



הטמנה עלולה לפגוע בסמנטיקה (2)



הטמנה עלולה לפגוע בסמנטיקה (3)



פתרונות לבעיית הסמנטיקה

- ❖ לא להטמין נתונים כלל
 - כל בקשת גישה מועברת לשרת
 - עלולה לצוץ בעיית ביצועים קשה, כמעט לא בשימוש
- ❖ session semantics במערכת הקבצים AFS:
 - הקובץ מועתק ללקוח כאשר פותחים אותו
 - פעולות בלקוח מתבצעות על העותק המקומי
 - עותק מעודכן מוחזר לשרת כאשר סוגרים את הקובץ
 - ביצועים טובים אלא אם ניגשים לחלקים של קבצים
- סמנטיקה חלשה לשיתוף נתונים (צריך לחכות לסגירה) אבל ברורה
- ❖ פרוטוקול מיוחד לשמירה על קונסיסטנטיות (נתאר בהמשך)
- ❖ התעלמות מבעיות הסמנטיקה על מנת למרב ביצועים: NFS

הפרוטוקול חסר המצב של NFS

- ❖ במערכת הקבצים המבוזרת NFS השרת אינו זוכר דבר אודות לקוחות פרט לזהות המחשבים שמותר לו לשרת, ולקוחות אינם זוכרים דבר אודות השרת פרט לזהותו
- ❖ כתוצאה מכך, נפילה או התנתקות מהרשת של לקוחות אינה משפיעה על השרת כלל
- ❖ נפילה של שרת אינה משפיעה על לקוחות אם הוא עולה חזרה תוך פרק זמן סביר
- ❖ הפרוטוקול חסר המצב משיג שרידות גבוהה במערכות מבוזרות עם מרכיבים לא אמינים

פתיחת קובץ NFS

- ❖ כשתהליך פותח בלקוח קובץ, הלקוח שולח לשרת בקשת פענוח עם שם הקובץ
- ❖ השרת מודא שהלקוח מורשה ושהמשתמש רשאי לגשת לקובץ
- ❖ השרת מחזיר ללקוח מזהה שכולל את מיקום הקובץ בדיסק (inode) ואת זמן יצירת הקובץ
- ❖ המזהה תקף עבור גישות לשרת כל זמן שלא נוצר בשרת קובץ חדש באותו מיקום; המזהה תקף גם אם השרת נפל ועלה חזרה

פעולות על קבצי NFS

- ❖ בקשות קריאה מקבצי NFS פתוחים מועברות לשרת (עם המזהה והמיקום בקובץ שממנו קוראים)
- ❖ הלקוח שומר במטמון בלוקים שנקראו ונכתבו לאחרונה
- ❖ בקשת קריאה לבלוק שנשמר במטמון מחזירה את תוכנו מהמטמון אלא אם הוא ישן מאוד (חצי דקה)
- ❖ בלוקים נכתבים לשרת בהדרגה אבל לפני ש-close חוזר
- ❖ אין צורך להודיע לשרת שקובץ נסגר
- ❖ כל התקשורת בין השרת ולקוחות מבוססת על בלוקים בגודל קבוע
- ❖ השרת מאשר קבלת וביצוע כל בקשת שירות, כולל כתיבת בלוקים ופעולות על מרחב השמות

וידוא ביצוע ב-NFS

- ❖ הלקוח חייב לוודא קבלת אישור על כל בקשת שירות
 - NFS רץ בדרך כלל על UDP שאינו אמין (אין בעיה כזו ב-TCP)
 - יתכן שהשרת נפל ולא קיבל את הבקשה או שקיבל ולא הספיק לבצע
- ❖ אי קבלת אישור לאחר זמן סביר גורמת למשלוח בקשה חוזרת
- ❖ השרת עלול לקבל בקשה יותר מפעם אחת
 - יתכן שהבקשה הראשונה או האישור עליה פשוט התעכבו ברשת
 - יתכן שהשרת ביצע את הפעולה בבקשה הראשונה אבל האישור אבד
- ❖ מכיון שהשרת חסר מצב הוא אינו זוכר שביצע כבר את הבקשה
- ❖ השרת עשוי לבצע את אותה פעולה יותר מפעם אחת
- ❖ במידת האפשר, הבקשות צריכות להיות אידמפוטנטיות

פעולות לא אידמפוטנטיות

- ❖ רוב הבקשות מגדירות פעולות אידמפוטנטיות, כתיבה למשל
- ❖ בקשות מסוימות לא ניתן להגדיר כאידמפוטנטיות, למשל יצירת קובץ תוך וידוא שאינו קיים (`open` עם דגלי `O_CREAT`, `O_EXCL`)
 - הביצוע הראשון ייצור את הקובץ
 - הביצוע השני ייכשל כי הקובץ קיים ויחזיר קוד שגיאה ללקוח
- ❖ הלקוח צריך להיות מוכן לקבל הודעות שגיאה לא נכונות ולטפל בהן בצורה סבירה
 - בדוגמה, לברר את תכונות הקובץ, מי יצר אותו ומתי, או
 - לדווח למשתמש על הבעיה אבל גם להסביר בדיווח שאולי הוא שיקרי
- ❖ עוד הבדל בסמנטיקה: קבצים פתוחים עלולים להימחק כי השרת אינו מודע כלל לכך פתוחים

קיבוץ כתיבות

❖ אישור על כתיבה מורה שהשרת כתב את הבלוק למדיה יציבה

- הלקוח יכול למחוק את הבלוק המעודכן מהמטמון
 - כתיבת בלוק בודד עלולה להצריך יותר מגישה אחת לדיסק (אם מגדילים קובץ) ולא ניתן לעכב אותה כדי לא לעכב את האישור
 - כתיבה מיידית כזו גורמת לבעיית ביצועים בדיסקים של השרת
- ❖ בגרסה 3 הלקוח יכול להרשות לשרת לעכב את הכתיבה
- כאשר הלקוח רוצה לכפות כתיבה בזמן סגירת קובץ או על מנת לפנות מקום במטמון, הוא שולח הודעת commit
 - השרת כותב את כל הבלוקים ומחזיר את זהות הבלוקים שנכתבו
 - הלקוח צריך להיות מסוגל לטפל בכתיבה חלקית (עקב נפילה של השרת, למשל) ולשלוח את הבלוקים החסרים שוב
- ❖ מאפשר לשרת לבצע מספר כתיבות בסדר יותר יעיל
- ❖ פרוטוקול עם מצב, אבל מצב שמותר לשכוח

סוגי הצבות של מערכות NFS

❖ הצבה קשיחה

- הלקוח לא מוותר לעולם
- קריאת מערכת חוזרת רק אחרי שהשרת מבצע את השירות
- תהליכים שמבצעים פעולות מול שרת שנפל נתקעים

❖ הצבה רכה

- קריאות מערכת חוזרות עם קוד שגיאה אם השרת לא עונה תוך זמן סביר

❖ הצבה ניתנת לפסיקה

- קריאת מערכת חוזרת אחרי שהשרת מבצע את השירות או שהתהליך מקבל איתות (למשל control-c); בהעדר איתות ממשיכים לחכות

שימוש בסוגי הצבה שונים

- ❖ הצבה קשיחה: בעיקר כאשר הלקוח לא יכול למעשה לתפקד ללא השרת, כמו במקרה שבו כל מערכת הקבצים של הלקוח מרוחקת
- ❖ הצבה קשיחה מתאימה גם לסביבות אצווה (batch) שתוכניות רצות בהן זמן רב, ורצוי שפשוט יחכו לעליית השרת כשהוא נופל
- ❖ הצבה ניתנת לפסיקה: מתאימה לסביבות אינטראקטיביות שבהן תוכניות לא מתוכננות לטפל בנפילת שרת; המשתמש יחליט
- ❖ הצבה רכה: מתאימה לסביבות אינטראקטיביות שבהן תוכניות מתוכננות לטפל בנפילת שרת, למשל על ידי הודעה למשתמש ושמירת עותק של הקובץ בדיסק מקומי (בזמן "שמור")
- ❖ הצבה רכה עלולה לגרום כשלים כשהשרת עמוס מאוד

חוזי שכירות קצרי מועד ב-NQNFs

- ❖ לקוחות מבקשים "חוזה שכירות" על קבצים שהם ניגשים אליהם
 - חוזה קריאה עם הטמנה מתיר ללקוח לקרוא מהקובץ ולהטמין בלוקים
 - חוזה כתיבה עם הטמנה מתיר ללקוח לקרוא ולכתוב ולהטמין בלוקים
 - חוזה ללא הטמנה מחייב את הלקוח לתקשר עם השרת בכל גישה לקובץ
- ❖ אם יש חוזה כתיבה עם הטמנה ולקוח מבקש חוזה, השרת מבטל את החוזה הקיים, מודיע על כך לבעליו, ומעניק חוזים ללא הטמנה
- ❖ אם יש חוזה עם הטמנה כלשהו בתוקף ולקוח מבקש חוזה כתיבה, השרת מבטל את החוזים הקיימים ומעניק חוזים ללא הטמנה
- ❖ אם יש חוזה כתיבה ללא הטמנה בתוקף ולקוח מבקש חוזה הוא מקבל חוזה ללא הטמנה
- ❖ אם יש רק חוזי קריאה ולקוח מבקש חוזה קריאה, הוא יורשה להטמין
- ❖ תוקף כל החוזים מוגבל בזמן (כדקה) ויש לחדש אותם

חוזי שכירות ושרידות

- ❖ חוזים אבודים (הלקוח והשרת לא מסוגלים לתקשר) עלולים לגרום לשרת לסרב להעניק חוזים חדשים או לפגיעה בקונסיסטנטיות
- ❖ שרת שצריך לבטל חוזה ואינו מצליח לתקשר עם הלקוח פשוט ממתין לפקיעת החוזה; שני הצדדים יודעים אז שהחוזה לא בתוקף
- ❖ לקוח שהיה בידו חוזה כתיבה עם הטמנה והחוזה מבוטל (מפורשות או בשל תפוגה) חייב לשלוח מייד את העדכונים לשרת
- ❖ שרת שנופל ועולה ואינו יודע איזה חוזים יש אולי בתוקף אצל לקוחות פשוט מחכה לזמן הפקיעה המקסימלי לפני שהוא מתחיל להעניק חוזים חדשים
- ❖ סיכום: הטמנה רק אם אין העברת מידע בין לקוחות דרך קבצים, יש מצב בשרת ולקוחות אבל שיכחון אינו אסון (commit כמו)