**חלק 1: עיוני**

מגיש: אלין ממדוב.

להלן הסבר מפורט על המערכת כולה, כולל הרישום, שליחת הודעות, קבלת הודעות, ותחזוקת שלמות, סודיות ומקוריות. התהליך יתאר כיצד הפרוטוקול עומד בתקני**End-to-End Encryption (E2EE)** ואיך הוא מגן בפני התקפות, כולל **MITM**.

1. **שיטת ההצפנה ואופן יצירת המפתחות:**

* **שימוש ב-RSA** (א-סימטרית): עבור החלפת מפתחות AES בצורה מאובטחת ואימות חתימות.
* **שימוש ב-AES** (סימטרית): להצפנת תוכן הודעות ומפתחות פרטיים.
* **KDF** (פונקציית נגזרת מפתח): יצירת מפתחות חזקים המבוססים על סיסמאות שהוכנסו על ידי הלקוח.

**מטרה**: שילוב AES ו-RSA מאפשר סודיות מהירה של הודעות תוך שימוש במפתחות א-סימטריים להגנה על מפתחות AES.

1. **תהליך הרישום הראשוני:**

**תהליך:**

1. **שלב ראשון**:
   * הלקוח שולח את מספר הטלפון שלו ואת המפתח הציבורי לשרת.
   * השרת שולח חזרה קוד רנדומלי בן 6 ספרות.
2. **שלב שני**:
   * הלקוח חותם על הקוד עם המפתח הפרטי שלו ושולח לשרת.
   * השרת מאמת את החתימה בעזרת המפתח הציבורי של הלקוח.
3. **שלב שלישי (אימות קוד סודי)**:
   * הלקוח מתבקש להכניס **קוד סודי**.
   * הלקוח מצפין את המפתח הפרטי שלו בעזרת AES, תוך שימוש בקוד הסודי כבסיס ל KDF.
   * המפתח הפרטי המוצפן והמפתח הציבורי נשלחים לשרת, שמאחסן אותם כך:
     + אינדקס: מספר הטלפון של הלקוח.
     + ערכים:
       - **מפתח ציבורי** (לא מוצפן).
       - **מפתח פרטי מוצפן** עםAES.

**איך זה משיג את המטרה:**

* **שלמות**: חתימה דיגיטלית מאמתת את מקור ההודעה.
* **סודיות**: המפתח הפרטי לעולם אינו נשלח בצורה גולמית.

1. **שליחת הודעה:**

**תהליך:**

1. הלקוח מייצר **מפתח AES רנדומלי** להצפנת תוכן ההודעה.
2. הלקוח מצפין את המפתח AES באמצעות **המפתח הציבורי RSA של הנמען** (הנמצא בשרת).
3. ההודעה נשלחת לשרת בפורמט הבא:
   * **שדות הודעה**:
     + מספר טלפון של השולח.
     + מספר טלפון של הנמען.
     + תוכן הודעה מוצפן.
4. השרת שומר את ההודעה:
   * אינדקס: מספר טלפון של הנמען.
   * ערכים:
     + מספר טלפון של השולח.
     + תוכן הודעה מוצפן.

**איך זה משיג את המטרה:**

* **סודיות**: מפתח AES מוצפן בעזרת RSA של הנמען, ורק הנמען יכול לפענח אותו.
* **מקוריות**: השרת יודע מי שלח את ההודעה בעזרת המפתח הציבורי של השולח.

1. **קבלת הודעות:**

**תהליך:**

1. הלקוח מתחבר לשרת ומבקש את ההודעות שלו.
2. השרת שולח את ההודעות השמורות, שכוללות:
   * מספר פלאפון של השולח
   * תוכן ההודעה המוצפן.
3. הלקוח נדרש להכניס את **הקוד הסודי** ששימש להצפנת המפתח הפרטי.
   * הלקוח מפענח את המפתח הפרטי שלו:
     1. פענוח AES עם קוד ה KDF.
   * המפתח AES המוצפן מפוענח עם המפתח הפרטי (RSA).
   * ההודעה מפוענחת עם מפתח AES.

**איך זה משיג את המטרה:**

* **סודיות**: רק הנמען עם הקוד הסודי שלו יכול לגשת למפתח הפרטי ולפענח את ההודעה.
* **שלמות ומקוריות**: המפתח הציבורי של השולח נדרש לאימות מקור ההודעה.

1. **אופן השגת שלמות, סודיות ומקוריות:**

**שלמות:**

* שימוש בחתימות דיגיטליות מוודא שההודעות והקוד הרנדומלי לא שונו.
* הודעות מוצפנות משויכות למספר טלפון של השולח והנמען.

**סודיות:**

* המפתח הפרטי מוגן בעזרת AES וקוד סודי(שמפיק מפתח מ KDF).
* תוכן ההודעה מוצפן בשכבות:
  1. עם מפתח RSA (ציבורי) להגנה על המפתח AES.
  2. עם מפתח AES להגנה על התוכן.

**מקוריות:**

* אימות החתימה מבטיח שההודעה נשלחה על ידי הלקוח הנכון.

1. **הגנה מפני מתקפות MITM:**

* **שימוש במפתחות RSA ציבוריים**: ההצפנה של מפתח AES מבטיחה שרק הנמען יכול לפענח את המידע.
* **חתימות דיגיטליות**: מבטיחות שההודעות לא שונו על ידי תוקף.
* **מפתחות פרטיים מוצפנים**: גם אם השרת נפרץ, התוקף לא יוכל לגשת למפתחות הפרטיים.

1. **מבנה נתונים בשרת:**

**טבלת משתמשים:**

* אינדקס: מספר טלפון.
* ערכים:
  + **מפתח ציבורי** (בפורמט PEM).
  + **מפתח פרטי מוצפן** (בפורמט PEM).

**טבלת הודעות:**

* אינדקס: מספר טלפון של הנמען.
* ערכים:
  + מספר טלפון של השולח.
  + מפתח AES מוצפן עם RSA.
  + תוכן ההודעה מוצפן עם AES.

**סיכום:**

המערכת משלבת מספר שכבות אבטחה:

1. הצפנה היברידית (RSA ו- AES)
2. חתימות דיגיטליות.
3. הגנה על מפתחות פרטיים בעזרת KDF ו-AES. הפרוטוקול עומד בסטנדרטים של E2EE ומבטיח הגנה גם במקרים של התקפות MITM.

בסופו של דבר, פרוטוקול זה מספק **הצפנה מקצה לקצה (E2EE)** על ידי שימוש בשילוב של הצפנה סימטרית וא-סימטרית, ומבצע אימות ושלמות בעזרת חתימות דיגיטליות. כל שלב מבוצע בצורה שמבטיחה שהמערכת עמידה בפני התקפות **MITM**, ושנשמרים כל הפרמטרים של **סודיות**, **שלמות** ו**מקוריות**.

**חלק 3:**

זמינות (Availability) בפרוטוקול מתייחסת ליכולת של המערכת להישאר פעילה ולתת מענה למשתמשים בכל עת, גם בתנאי עומס, תקלות או התקפות זדוניות. בפרוטוקול שתיארתי, ניתן להבטיח זמינות במידה מסוימת, אך יש גם מגבלות טבעיות. להלן ההסבר:

**איך ניתן להבטיח זמינות?**

1. **תשתית שרתים חזקה (Scalable Infrastructure):**
   * על השרת להיות חזק מספיק לתמוך בעד 10 לקוחות בו-זמנית, כפי שהוגדר בתוכנית.
   * ניתן להשתמש במנגנונים של איזון עומסים (Load Balancers) כדי לחלק את עומס התעבורה בין שרתים שונים במקרה של עומס.
2. **מנגנון חידוש מפתחות אוטומטי:**
   * אם תוקף המפתחות פג או שיש בעיה עם המפתחות הקיימים, יש ליישם מערכת שמחדשת את המפתחות באופן אוטומטי מבלי לדרוש רישום מחדש של הלקוח.
3. **שימוש בגיבויים והעתקים של מסד הנתונים:**
   * מסד הנתונים צריך להיות מגובה באופן קבוע, עם רפליקציה (Replication) לשרתים אחרים.
   * במקרה של תקלה בשרת אחד, המידע יישאר זמין בשרת אחר.
4. **התמודדות עם תקלות רשת:**
   * ניתן להשתמש בפרוטוקול שמאפשר ניסיונות חוזרים (Retries) לשליחת הודעות או חתימות שלא נמסרו.
5. **התמודדות עם התקפות מניעת שירות (DoS/DDoS):**
   * שימוש במנגנונים כמו Firewall וחוקי Rate Limiting להגבלת כמות הבקשות שיכולות להגיע מלקוח מסוים בזמן קצר.
   * מנגנוני זיהוי וחסימה של תוקפים באמצעות Machine Learning או חתימות התנהגות חשודות.

**מגבלות בהבטחת זמינות**

1. **תלות במערכת צד שלישי:**
   * אם השרת מתארח בענן, תקלה בצד ספק השירות (לדוגמה: AWS או Azure) עלולה לשבש את הפעילות.
2. **שימוש בערוץ בטוח ברישום:**
   * השימוש בערוץ בטוח להעברת הקוד הראשוני והחתימה מוגבל לפעם אחת בלבד. אם הערוץ הזה נפגע, תהליך הרישום יכשל.
3. **מתקפות מתקדמות:**
   * במידה ותוקף משתמש בטכניקות מתקדמות של מניעת שירות (כמו DDoS מבוזר עם עשרות אלפי מכשירים), השרת עלול להתקשות לעמוד בעומס גם אם יש מנגנוני .Rate Limiting

**האם ניתן להבטיח זמינות מלאה?**

לא ניתן להבטיח זמינות **מלאה** (100%) בתנאים ריאליים, משום שתמיד קיימת אפשרות לתקלות טכניות, התקפות מתוחכמות, או אירועים בלתי צפויים (כמו קריסת תשתית חשמל). עם זאת, ניתן להבטיח זמינות **גבוהה מאוד** (High Availability) על ידי יישום העקרונות שהוזכרו.

**התאמת הפרוטוקול לזמינות**

הפרוטוקול שתיארתי מתאים למערכת בעלת מספר מוגבל של לקוחות (עד 10), כך שניתן ליישם את ההמלצות בקלות יחסית. עם זאת, כדי לעמוד במטרות של זמינות גבוהה, ייתכן שיהיה צורך בשדרוגים, כמו הוספת שרתים, יישום רפליקציה של מסד הנתונים, ושימוש במנגנוני אבטחה מתקדמים להתמודדות עם התקפות.