**Overseer**

חלק א - מסמך אפיון

אילן דוברומילסקי

0.1v

11/02/2020



**היסטוריית גרסאות המסמך**

|  |  |  |
| --- | --- | --- |
| **תאריך** | גרסה | **תקציר השינויים** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**תוכן עניינים**

**הקדמה..............................................................................................................................4**

**תיאור כללי.........................................................................................................................5  
דרישות מפורטות................................................................................................................6**

**1. הקדמה**

החלק הזה ישמש להצגה כללית של מסמך האפיון - הוא נועד לאפשר לקורא להבין במה מדובר - מה זה בכלל הפרויקט הזה ומה מופיע בהמשך המסמך.

1.1 מטרה

מטרת המסמך הינו הצגה כללית של הפרויקט, כיצד הוא עובד ולמה הוא מיועד.

1.2 המוצר

*שם מוצר: Overseer*

*מה המוצר עושה: המוצר מייצר דוחות המסכמים פרטים על תעבורת הרשת בפרקי זמן או כמות המוגדרים על ידי המשתמש. בנוסף, יכול להציג סיכום של כל הדוחות.*

יתרונות: דוחות המוצר מוצגים בצורה קריא המאפשר ניתוח והבנה פשוטה ומהירה. המוצר קל לשימוש. המוצר הוא בעל רמת דיוק גבוהה.

1.3 הגדרות

שרת: השרת הוא ה"מוח" של המערכת. הוא דואג לניתוח ויצירת הדוחות. לצידו פועל האתר.

אתר: מציג את הדוחות הקיימים למשתמש.

סוכן: נמצא על מחשב העובדים. אחראי על קליטת וסיכום התעבורה ושליחתו לשרת.

פקטה: יחידת מידע העוברת ברשת.

DB(database) – קובץ המכיל json של מידע דרוש לפעולות בין הרצות.

**1.4** תקציר

*הסבר מה מופיע בהמשך המסמך ופירוט כיצד זה מסודר לאורך המסמך*

1. תיאור כללי

החלק הזה מתאר את הגורמים העיקריים המשפיעים על המוצר ועל דרישותיו. חלק הזה לא מפרט דרישות ספציפיות אלא רק עוזר להבין את הדרישות.

2.1 פונקציונליות

שרת:

* מריץ את האתר.
* מנתח את החבילות שנשלחות אליו מכל הסוכנים.
* מייצר דוח.

אתר:

* מציג את הדוחות.
* מייצר דוח - סיכום דוחות.

סוכן:

* מסניף פקטות היוצאות ונכנסות למחשב.
* עושה סיכום של כל פקטה על פי שדות קבועות מראש.
* שולח לשרת את לניתוח כללי את הפקטה במסוכמת.

2.2 קהל היעד

המוצר מיועד לכל ארגון המעוניין לפקח/להשיג מידע על תעוברת הרשת וכל עובד בפרט.

דרוש מעט ידע על רשתות. המשתמש מריץ את השרת – ביכולתו לבחור פורט אליו הסוכנים יפנו, כל כמה זמן ייצר דוח ובאיזה אופן. סוכן – פורט אליו יפנו הסוכנים בשרת, כתובת השרת.

על פי הגדרות אלו, המשתמש צריך לדעת למצוא את כתובת אינטרנט של השרת, ולדעת באיזה פורט לבחור כך שלא יפגע בתוכנות אחרות.

2.3 אילוצים עיקריים

בשביל שהמוצר יעבוד כמו שצריך, דרושים לפחות שתי מחשבים – אחד שיריץ את השרת, ואחד שיריץ את הסוכן.

על המחשבים דרושה השפה פייתון גרסת 3.7.4 לפחות עם הספריות os, socket, json, time, threading, subprocess, sys, requests, re, scapy, flask. לשרת צריך להיות מספר פנוי של פורטים ככמות הסוכנים +1.

2.4 הנחות ותלויות

המוצר עובד בעיקרו מעל הרשת. הוא נרשם ב windows אך אני מאמין שיכול לעבוד מעל כל המערכת הפעלה התומכת באילוצים.

**2.5** סיקור מצב השוק כיום (אופציונלי)

https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/network-traffic-analysis

wireshark

1. דרישות מפורטות

3.1 דרישות פונקציונליות

**יצירת דוח**

זה הפיצ'ר המרכזי של המערכת, יצירת דוחות המפרטות על התעבורה במחשבים וברשת.  
יצירת הדוח נעשת בשרת, הפונקציה מקבלת מספר מוגדר (או לפי זמן) של מידע ומשתמשת ב template מוכן. הפונקציה ממלאת מידע בtemplate, ואת התוצר שומרת כדוח.

**יצירת סיכום**

הסיכום הוא תוספת לפרוייקט. מטרתו היא להקל אף יותר למשתמש ולייצר דוח מיוחד המסכם את המידע בכל הדוחות שנאספו כבר.   
הסיכום נשען על בסיס הנתונים. האתר מייצר את הסיכום. הוא עושה זאת על ידי קריאת בסיס הנתונים וכמו הדוח שימוש בtemplate ומילואו. התוצר נשלח ישירות למשתמש ואינו נשמר בשום מקום.  
שגיאות שיכולות להתרחש: שגאית קריאה מבסיס הנתונים. במקרה כזה, יוחזר למשתמש הודעה מתאימה.  
לא מספיק מידע ליצירת דוח סיכום. גם פה תוחזר הודעה מתאימה.

**שמירת נתונים**

שמירת הנתונים הוא חלק בעייתי יחסית. יש לוודאה שהנתונים שנצברו תקינים, ושלא תהיה בעיה בשמירתם והוספם לבסיס הנתונים. בסיס הנתונים שלי, לשם נוחיות, נשמר כ JSON. כך הקריאה והפיכתו למשתנה שמיש, ושמירתו נעשים במהירות ונוחיות. בחרתי לשמור את המידע ישר אחרי יצירת כל דוח. כך ניתן לקטלג את המידע בקלות – לפי דוח, וגם העדכון נעשה כמה שיותר כדי למנוע איבוד מידע.

**multithreading**

ה multithreading הוא הפיצ'ר המאפשר את הפרוייקט. כך השרת יכול לתקשר עם כמה סוכנים בו זמנית, שיפור ביצועים, והקטנת זמני חישוב. השרת לא יתקע במקרה חריג, אלה רק thread וכך המערכת יכולה להמשיך ולתפקד.  
הפיצ'ר הזה לא מקבלת פלט, וגם לא יוצר פלט, אך הוא עוזר רבות בטיפול שגיאות, והעבודה הכללית של הפרוייקט.

3.2 דרישות של ממשקים חיצוניים

**3.2.1 ממשקי משתמש** (למשל: GUI)

האתר משמש כממשק משתמש.  
כקלט הוא מקבל נתיב לתקיה בה מוכלים הדוחות להצגה.  
המשתמש מבצע שאילתת HTTP פשוטה (דרך הדפדפן כנראה) והאתר יחזיר את הדוחות/עמוד מתאים.  
כאשר מתרחשת שגיאה באתר, הוא ישנה ערכים בעמוד האומרות שהתרחשה שגיאה.

**3.2.2 ממשקי חומרה**

מחשב שמסוגל להריץ python עם הספריות os, socket, json, time, threading, subprocess, sys, requests, re, scapy, flask.

**3.2.3 ממשקי תוכנה** (אם התכנה מחולקת לכמה תוכנות שונות שצריכות לתקשר בניהן)

הפרויקט מחולק ל 3 תוכנות. הסוכן, השרת והאתר. הסוכן מדבר עם השרת, השרת עם האתר והסוכן, והאתר עם האינטרנט.

ראשית השרת צריך להעלות, וביחד איתו הוא יתחיל את האתר. לאחר מכן הסוכן מקבל כקלט את כתובת השרת ויפתח שיחה איתו. הסוכן משם ישלח מידע לשרת, השרת ייצר דוחות, והאתר מציג אותם.

אם שגיאה תתרחש אצל הסוכן הוא יעצור. שגיאה בשרת תסגור את השיחה בינו לסוכן. שגיאה באתר יחזיר עמוד מותאם.

**3.2.4 ממשקי תקשורת (למשל תקשורת בין שר**ת ללקוחות)

התקשורת בין הסוכן והשרת מתנהלת בפרוטוקול JSON, האתר ב HTTP.

3.3 דרישות לא פונקציונליות

*דרישות מהמערכת שלא מתבטאות בפיצ'ר ספציפי או בתהליך ספציפי שמתרחש במערכת אבל משפיעות על אופן עיצובה ומימושה, לדוגמא:*

**3.3.1 דרישות ביצועים (performance)**

קוד השרת בנוי ברובו מ threading. החילוק משפר ביצועים

**3.3.2 דרישות מהימנות (reliability)**

**לאורך הקוד יש הקפדה על שמירת נתונים בזמנים אופטימלים, דאגה על כך שהקוד לא יקרוס משגיאות, ודאגה ל**multithreading **כך שלא יתקעו.**

**3.3.3 דרישות זמינות (Availability)**

**כל עוד השרת עומד, הסוכנים יכולים להתחבר.**

**3.3.4 דרישות אבטחה (security)**

**3.3.5 דרישות תחזוקה (maintainability)**

**לפני תחילת ריצת התוכנית המשתמש יכול לקבוע הגדרות להתאמות אישיות, דבר שעוזר על תחזוקה נוחה יותר. בנוסף, במקרה של קריסה, יודפס הודעה עם מידע שיעזור לטיפול הבעייה.**

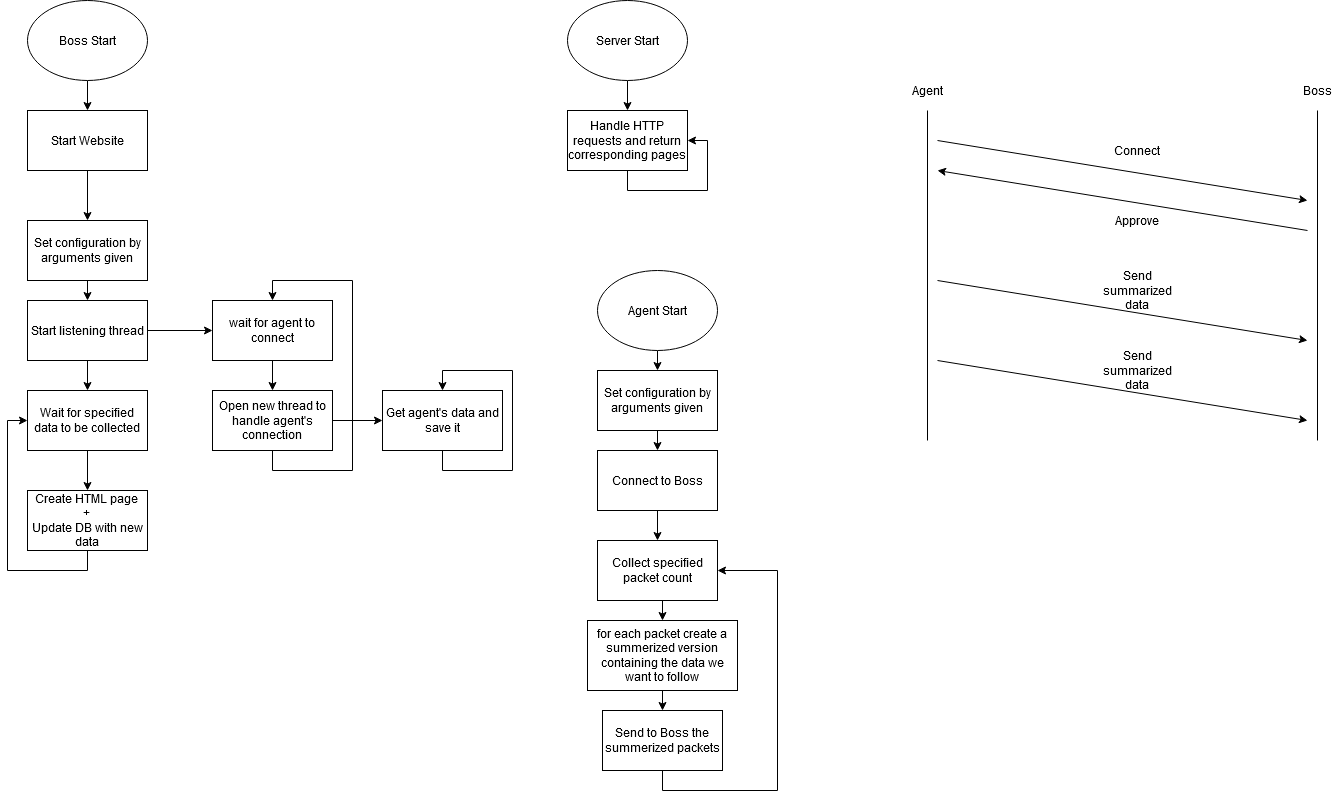
3.4 דרישות בסיס נתונים

*הנתונים הנכנסים לאחסון הם כל השדות הדרושות ליצור דוח. כך אם יש צורך בשחזור דוח, אפשר להעיר את הנתונים לפונקציה בשרת והוא ידע ליצר את הדוח בשנית.*

*הנתונים נשמרים כ json בקובץ מסוג .dat. אין חשיבות לסוג הקובץ, אלה רק למבנה התוכן שחשוב שהיה json שפייתון יכול לקרוא.*

*נפח האחסון הדרוש הוא תלוי זמן ריצה. ככל שהשרת רץ יותר זמן, כך ייצר יותר דוחות שישמרו ב DB. אומנם משקלו של הנתונים של כל דוח אינו הרבה, עם הזמן יגדל.*

*הקובץ עצמו אינו מוצפן בשום צורה ואף קריא יחסית.*

1. כלי ניתוח נוספים 

5. נספחים