##### 

##### המחלקה להנדסת תוכנה

## ניטור שרתים באמצעות מערכת Nagios

**חיבור זה מהווה חלק מהדרישות לקבלת**

**תואר ראשון בהנדסה**

מאת

### שירן בן חמו

**תמוז תשע"א יולי 2011**

****

###### המחלקה להנדסת תוכנה

###### ניטור שרתים באמצעות מערכת Nagios

**חיבור זה מהווה חלק מהדרישות לקבלת**

**תואר ראשון בהנדסה**

מאת

### שירן בן חמו

**מנחה: מר שי תבור**

**אישור המנחה: תאריך: 07.07.11**

**אישור ראש הפרויקט בתעשייה: תאריך: 07.07.11**

**אישור מרכז הפרויקטים: תאריך: 07.07.11**

1. תקציר

**פרויקט הגמר נעשה במכללה האקדמית להנדסה**, **ירושלים**, עבור צוות התשתיות אשר אחראי על התחזוקה של כלל מערכת המחשוב במכללה, בפרט שרתים רבים המספקים שירותים לכלל הסטודנטים וסגל המכללה.

**הבעיה כיום** היא שלא קיימת מערכת המנטרת את השרתים הרבים הקיימים, והתרחשותה של תקלה מתגלה רק כאשר מתקבלות תלונות בקרב המשתמשים וזאת למרות שישנם שרתים רבים המהווים יסוד חשוב ומרכזי במערכת המחשוב של המכללה ולכן חשוב לדאוג כי הם יפעלו באופן תקין.

**פרויקט הגמר כפיתרון** ייתן מענה לצורך במערכת ניטור, ייתן אינדיקציה למצב השרתים בכל רגע נתון ובכך יקל על צוות הסיסטם לאתר תקלות, ובכך יקצר את זמן טיפולן.  
הנתונים יוצגו בממשק WEB ובמקרה של חריגה תישלח התראה לתיבות מייל המוגדרות במערכת.  
הפרויקט יתבצע על בסיס מערכת בשם Nagios, שהיא מערכת ניטור מבוססת קוד פתוח.  
המערכת מאזינה לשרתים ול-Services הרצים בהם, ובעת תקלה מדווחת על כך למשתמשים.  
בכל שרת ישנו מעין רכיב אשר רץ בו באופן תמידי ובודק services ותכונות נוספות על ידי שימוש ב- plug-ins חיצוניים היושבים בשרת ה-Nagios.  
למעשה "רכיב" זה מתקשר בין שרת ה-Nagios לבין השרת, ודרכו נבדקים תכונות של השרת כמו מצב CPU, נפח, RAM וכדומה.  
בעת תקלה נשלחת התראה לכתובות מייל אשר מוגדרות במערכת.  
נתונים נוספים על המערכת כמו דוחות, מצב נוכחי של השרת, ניתן למצוא בממשק Web דרך הדפדפן.

הצהרה:

העבודה נעשתה בהנחיית מר שי תבור,

המכללה האקדמית להנדסה ירושלים- המחלקה להנדסת תוכנה

במכללה האקדמית להנדסה ירושלים במחלקת מחשוב.

החיבור מציג את עבודתי האישית

ומהווה חלק מהדרישות לקבלת תואר ראשון בהנדסה.

תודות:

* ראשית אודה למר רועי ליאני, מצוות התשתיות במכללה, על כל העזרה והתמיכה, ועל השעות הארוכות שבהן עזר לי להתגבר על כל המכשולים שעמדו בדרכי.
* ברצוני להודות למנחה הפרויקט, מר שי תבור, על הליווי המקצועי וההדרכה לאורך כל הדרך.
* ברצוני להודות למר יאיר נווה, אחראי הפרויקט, אשר תמך וסייע לנו בביצוע הפרויקט.
* ברצוני להודות לד"ר יעקב אקסמן אשר הקשיב לבעיות ושאלות ותמיד שמח לעזור.

.

תוכן עניינים

**1. תקציר........................................................................................................................3**

**2. תאור מסגרת הפרויקט.................................................................................................7**

**3. תאור הבעיה................................................................................................................8**

**4. תאור הבעיה מבחינת הנדסת תוכנה..............................................................................9**

**5. תאור פיתרון..............................................................................................................10**

5.1 טכנולוגיה................................................................................................................10

5.2 מימוש.....................................................................................................................10

**6. תאור מערכת שמומשה...............................................................................................12**

6.1 שרת הניטור.............................................................................................................12

* 1. ממשק GUI..............................................................................................................12

6.3 Add-ons: NRPE & NSClient.......................................................................................12

6.4 תוספים...................................................................................................................13

6.5 קבצי קונפיגורציה......................................................................................................13

6.6 אפליקציית אנדרואיד.................................................................................................14

6.7 תוכנת יצירת קבצי קונפיגורציה באופן אוטומטי...............................................................14

6.8 סקריפט התקנת NRPE...............................................................................................14

**7. בדיקות.....................................................................................................................15**

7.1 בדיקת התקנת NRPE................................................................................................15

7.2 בדיקת קבצי קונפיגורציה...........................................................................................15

7.3 בדיקת plug-ins........................................................................................................16

**8. השוואה לספרות........................................................................................................17**

**9. מסקנות....................................................................................................................18**

**9. רשימת ספרות...........................................................................................................19**

**10.נספחים....................................................................................................................18**

1. תיאור מסגרת הפרויקט

הפרויקט מתבצע במכללה האקדמית להנדסה ירושלים.  
במכללה יש קרוב ל 70 שרתים פיזיים ווירטואליים המספקים שירותים רבים וחשובים לפעילות סגל המכללה והסטודנטים.  
השרתים אחראים על שירותים חשובים כגון ניהול משתמשים, מיילים, אינטרנט, ומסדי נתונים.  
ללא תקינותם של שרתים אלו, העבודה השוטפת של המכללה תיפגע, דבר שהיא אינה מסוגלת להרשות לעצמה.  
על השרתים במכללה לעבוד 24/7 תוך מתן שירות במהירות ובאמינות יתרה, ולשם כך קיים הצורך במערכת התראה בזמן אמת לתקלות בשרתים אשר מהווים מרכיב עיקרי וחשוב במערכת המחשוב של המכללה.

הפרויקט נעשה עבור צוות התשתיות במכללה מאחר והם הצוות שאחראי על שירותי המחשוב ותקינותם, ולכן מערכת שכזו תקל על עבודתם.

3. תיאור הבעיה

במכללה ישנן מספר רב של שרתים המספקים שירותים חיוניים כגון שרתי storage, DB, files, exchange, web.  
שמירה על פעילות תקינה של השרתים הינה דבר קריטי.  
הבעיה: כאשר נוצרת תקלה באחד מהשרתים קשה לטפל בה כיוון שאיתורה יארך זמן מה, וזאת בכדי לאבחן את השרת הבעייתי ומה בדיוק הבעיה.  
ישנו צורך לדעת את מקור התקלה ואת מהותה בכדי לדאוג לטיפול מהיר ויעיל וזאת על ידי ממשק web שיציג את נתוני המערכת.  
בנוסף יש לשלוח התראה לצוות התשתיות במקרה של חריגה שהגדרתה תיקבע מראש. ההתראה יכולה להיות על ידי שליחת מיילים באופן אוטומטי בעת חריגה.

הפרויקט עסוק בניטור השרתים, כלומר ייתן אינדיקציה לגבי מצב השרתים ממספר בחינות שהן:  
1. זמינות השרת.  
2. נפח הדיסק.  
3. ניצול זיכרון הRAM-.  
4. עומס CPU.  
5. שירותים (Services) ופרוטוקולים: HTTP, HTTPS, SNMP, FTP, SMTP, POP3, DNS, SSH, TCP, ICMP, UDP.

ידיעת נתונים אלו על השרתים יאפשרו לחזות מבעוד מועד תקלות שעלולות להתרחש כמו נתונים על נפח דיסק המצביעים על מילוי קיבולת שרת האחסון ויאפשרו לצוות התשתיות לחזות זאת ולטפל בבעיה על ידי מחיקת נתונים ישנים מהשרת או הוספת מקור אחסון.

עד היום צוות התשתיות במכללה אינו היה מודע למצב הנוכחי של השרתים ובאם קורת תקלה יודעים הם על כך רק דרך תלונות המשתמשים, וברגע ההיוודעות לתקלה יש למצוא מה בדיוק הבעיה ואיזה שרת גורם לה.

הפרויקט בא לפתור זאת ולהציע לסיסטם מערכת נוחה למשתמש היודעת לאבחן ולהתריע על תקלות עוד לפני תלונות המשתמשים, ולדעת את מהותה רק מתוך הסתכלות על הממשק הגרפי.

1. הבעיה מבחינת הנדסת תוכנה

כאשר מתרחשת תקלה או כאשר ישנו צורך לקבל הערכת מצב לגבי מצב השרתים השונים, יש לאפשר לצוות התשתיות על ידי ממשק WEB לדעת את מצבם העדכני של השרתים, ובמקרה של תקלה כלשהי, לקבל התראה לתיבת המייל או לטלפון הסלולארי שתציין את מקורה ומהותה.

מערכת ה- Nagios הינה מערכת אשר מטרתה היא לבצע את פעולת הניטור והיא  
מותקנת על שרת שמהווה שרת ניטור (שרת Nagios), אך היא אינה מסוגלת לנטר את כל הפלטפורמות השונות והציוד השונה, אשר דרושים למכללה.  
במערכת קיימים מספר תוספים אשר יודעים לנטר פרוטוקולים שונים, אך יש להתאים את התוספים לשרתים שיבצעו את דרישות הלקוח, כלומר ינטרו אך ורק את התכונות הרלוונטיות לכל שרת וזאת מאחר וכל שרת מריץ אפליקציות שונות ועל כן יש לבדוק בכל שרת נתונים שונים.

יש להבדיל בין שרתי Win לשרתי Linux בדרך ההתקשרות של שרת ה-Nagios עימם. שרתי לינוקס מתקשרים על ידי add-on שנקרא NRPE ושרתי Win מתקשרים על ידי add-on הנקרא NSClient++.

כמו כן יש ליצור קבצי קונפיגורציה (קבצי cfg) עבור כל שרת המגדירים את ה-services והפרוטוקולים השונים אותם יש לנטר באותו שרת.  
לכל שרת יש קובץ קונפיגורציה משלו אשר מכיל את ה-services השונים אשר יש לנטר בו, ובכל service קיימת פקודה אשר קוראת ל-plug-in הרצוי בכדי שינטר את אותה תכונה.   
התוספים הם עצמאיים ובלתי תלויים אחד בשני, והם נשמרים בשרת ה-Nagios, בתיקייה שמיועדת אך ורק לתוספים.בעת ניטור service או פרוטוקול מסוים, המערכת יודעת לגשת לתיקיית ה- plug-ins ולגשת ל- plug-inהרצוי ודרכו לנטר את התכונה הרצויה בשרת.

בנוסף, ממשק ה-Nagios הינו ממשק בסיסי אשר הפונקציונאליות בו לוקה בחסר והוא אינו מסוגל להציג דוחות וסטטיסטיקות לגבי מצב השרתים.  
אין אפשרות לייצר גרפים או להגדיר סף שמעבר על סף זה יוגדר כמצב בעייתי ואז תישלח התראה למשתמש.

הבעיה: שימוש בשרת ה- Nagios לשם ניטור כל השרתים והפלטפורמות הדרושות והוספת פונקציונאליות לממשק כך שיהיה נוח ופשוט למשתמשים וייתן את האינפורמציה הדרושה באופן בהיר.

1. תיאור פיתרון

5.1 טכנולוגיה

תוכנת Nagios מבוססת קוד פתוח ותומכת שרתי Linux.  
הבחירה בה נעשתה מתוך הצורך להתאים אותה לצרכי המכללה, ניתן להרחיב אותה על ידי כתיבת תוספים שיבצעו את הניטור הדרוש.  
התוספים נכתבים לרוב בשפת C או PERL.

5.2 מימוש   
על מנת לבצע את פעולת הניטור יש להתאים חלק מהתוספים לפי התכונות הנדרשות מהשרתים.  
כל תוסף הינו עצמאי ובלתי תלוי בשני.  
יש להתאים מספר תוספים אשר כל אחד מבצע משימה שונה.  
1. בדיקת זמינות השרת- נכתב plug-in אשר משתמש בפונקציונאליות של plug-in קיים בשם is\_alive המנטר באופן בסיסי את זמינות השרת, הוא מנסה להתחבר אליו, ובאם לא הצליח שולח התראה אך ללא כל מידע על סיבת התקלה.  
ההתאמה נעשתה בכך שה-plug-in החדש מסוגל לזהות את הסיבה שההתחברות לא הצליחה:   
- מצב תקין: כאשר ההתחברות הצליחה.  
- מצב לא תקין: נובע משתי סיבות, השרת לא נמצא או שהשרת לא מחובר לרשת, כלומר יש אבחנה בין שרת Unreachable לבין שרת שלא פועל.  
2. בדיקת נפח הדיסק של השרת- ההתאמה נעשתה בהצגת נפח הדיסק שבשימוש באחוזים, כלומר כמה אחוז מהדיסק מנוצל . בעת ניצול 75% מהדיסק ישלח מצב אזהרה, בעת ניצול 95% ישלח מצב קריטי.  
3. בדיקת עומס ה-CPU של השרת- הגדרת סף של עומס CPU שעבורו ישלח מצב קריטי.  
4. בדיקת ניצול הזיכרון של השרת- הגדרת סף של ניצול זיכרון שעבורו ישלח מצב קריטי  
5. בדיקת ה-services השונים ופרוטוקולים: HTTP, HTTPS, SNMP, FTP, SMTP, POP3, DNS, SSH, TCP, ICMP, UDP.

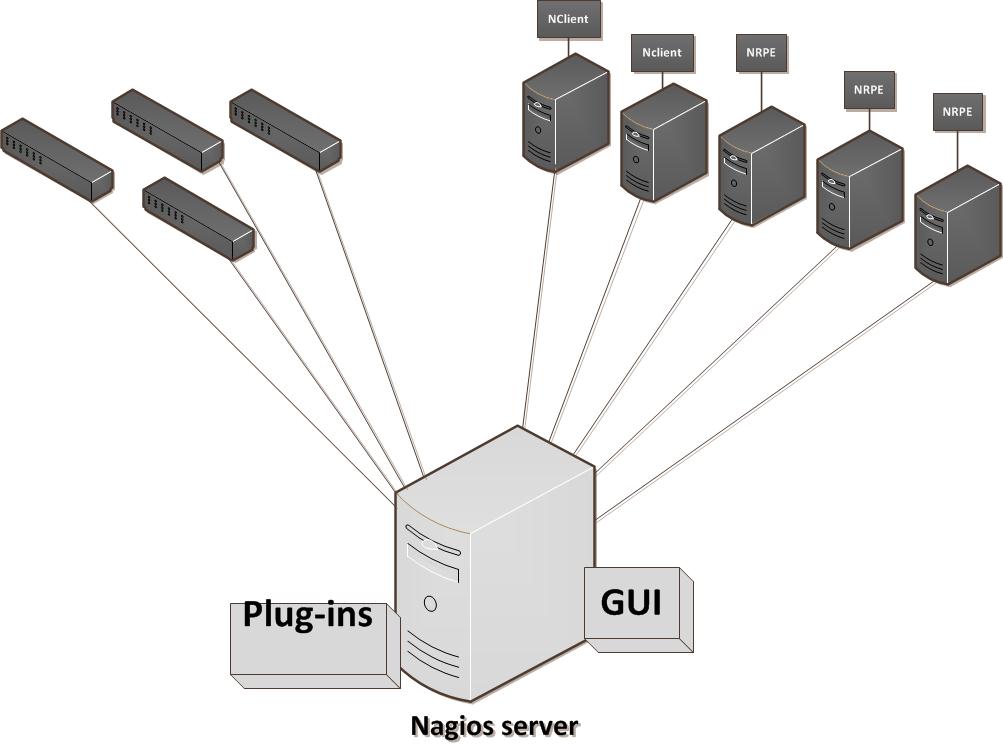
מלבד הדרישה לניטור הפרוטוקולים, כל דרישה מהווה תוסף המבצע את הדרישה.  
באשר לפרוטוקולים כל פרוטוקול יהווה תוסף בודד.

לאחר כתיבת התוספים יש לשמור אותם בתיקיית התוספים של Nagios שנמצאת בשרת ה- Nagios, ולאחר מכן עורכים קובץ קונפיגורציה לשם עדכון המערכת על תוסף חדש, ואז יש לאתחל מחדש את השרת על ידי פקודות פשוטות.  
לאחר אתחול השרת ה- Nagios יידע לגשת לתוספים החדשים ולבצע את הפונקציונאליות הכתובה בהם.  
ממשק ה-GUI יראה את הנתונים באופן אוטומטי.

.

**תרשים 5.1- רעיון הפרויקט**

שרת Nagios אשר אמור לזהות את ציוד התקשורת והשרתים.  
הפרויקט שלי יעסוק בשרתים בלבד, ושרת ה-Nagios באמצעות התוספים הקיימים בו ינטר את השרתים ובאמצעות ממשק ה-GUI שבו יציג את הנתונים לצוות התשתיותההתחברות לשרתים נעשית על ידי NRPE בשרתי Linux ו- NSClient עבור שרתי Windows.



1. תיאור מערכת שמומשה

6.1 **שרת הניטור**  
בשלב הראשון של הפרויקט הותקן ה-Nagios על עמדת מחשב שחובר לראוטר יחד עם מחשבים נוספים על מנת להתנסות במערכת ובזיהוי ציוד התקשורת והעמדות המחוברות באותה הרשת. לאחר מכן הותקן ה-Nagios על שרת שהוקצה לכך על ידי המכללה, בכדי שיהווה יחידה חזקה לניטור שרתים וציוד תקשורת.  
שרת ה-Nagios מהווה את ליבת המערכת, ועליו מותקנת המערכת .

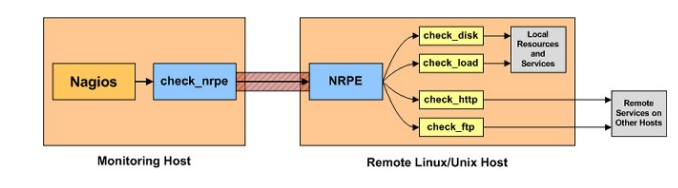
* 1. **ממשק ה-GUI**

שרת ה-Nagios מגיע עם ממשק בסיסי שמציג נתונים בצורה לא נוחה לקריאה, ולא ניתן לייצר מהנתונים דוחות או לראות גרפים הממחישים את מצב השרתים.  
התבצע מחקר באינטרנט לגבי מימוש הממשק.  
באינטרנט נמצא פרויקט מבוסס קוד פתוח בשם Ninja אשר מבוסס על ממשק ה-Nagios ומשדרג את הממשק.  
ממשק זה מאפשר את כל הפונקציונאליות שנדרשה, והינו פשוט ונוח לשימוש המאפשר יצירת דוחות וגרפים, והצגת מצב השרתים ברשת בצורה גרפית ופשוטה.  
בממשק ישנה אפשרות ליצור דוח בטווח זמן מוגדר מראש ולהגדיר סף שממנו והלאה תישלח הודעת דואר אלקטרוני למשתמשים מוגדרים.  
הממשק הוטמע ושינה את ממשק ה-Nagios.

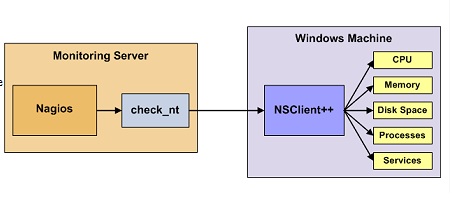
* 1. **Add-ons: NRPE & NSClient++**

מקריאה של מאמרים מצאתי שההבדל בין ניטור שרתי Linux לבין שרתי Windows היא דרך השגת המידע. בשרתי Linux אחת מהדרכים להשיג את המידע היא על ידי add-on הנקרא NRPE שיותקן על שרתי ה-Linux. כאשר יש צורך לנטר תכונה מסוימת, תוסף בשם check\_nrpe בשרת ה-Nagios יפנה ל-NRPE ויאמר לו איזו תכונה יש לנטר.  
בשרתי Windows דרך השגת המידע תתבצע על ידי add-on הנקרא NSCLIENT אשר יותקן על שרתי ה-Windows. בעת הניטור שרת ה-Nagios באמצעות תוסף הנקרא check\_nt יפנה ל- NSCLIENT ויאמר לו איזו תכונה לנטר.  
 NRPE הותקן על שרתי Linux ו-NSClient++ על שרתי Windows. ההתקנה של ה-NRPE הייתה כרוכה בהורדת ספריות והתקנתם על ידי פקודות מה-shell.

**תרשים 6.3.1**  
תרשים זה מציג את דרך הניטור בשרתי Linux.



**תרשים 6.3.2**תרשים זה מציג את דרך הניטור בשרתי Windows.



* 1. **תוספים**

בשרת ה-Nagios ישנה תיקייה אשר תפקידה להכיל את כל התוספים, לאחר הוספת תוסף חדש יש להפעיל את השרת מחדש ואז התוסף מוכר על ידי ה-Nagios באופן אוטומטי.  
בתיקייה זו ישנם התוספים אשר התאמתי לצרכי המכללה, כמו כן תוספים המובנים בתוך מערכת ה- Nagios היודעים לנטר את הפרוטוקולים השונים.   
התוספים אשר הותאמו ושוכתבו הינם:  
1. Check\_disk.c- בדיקת ניצול נפח דיסק.  
2. Check\_load.c- בדיקת ניצול CPU.  
3. Check\_ping.c- בדיקת זמינות השרת.  
4. Check\_mem.c- בדיקת ניצול זיכרון

* 1. **קבצי קונפיגורציה**

ניטור שרת כלשהו מתבטא בהתקנת ה-add-on המתאים למ"ה של השרת. בנוסף יש ליצור קובץ קונפיגורציה שבעצם מגדיר את פרטי השרת, הקבוצה אליו הוא משתייך (בפרויקט מוגדרות שתי קבוצות: שרתי Win ושרתי Linux), והגדרות של ה-services השונים שיש לנטר באותו שרת. (דוגמא לכך קיימת בנספחים).

* 1. **אפליקציית אנדרואיד**

ישנה אפליקציה מבוססת קוד פתוח אשר חוברה לשרת ה-Nagios ומציגה על גבי הטלפון הסלולארי את הבעיות וההתראות.  
האפליקציה עובדת אך ורק ברשת ה- wireless של המכללה, מאחר והאפליקציה מתחברת לכתובת ה-URL של הממשק. בעת התרחשות תקלה, תוצג למשתמש התראה עם פירוט של ה-Host הבעייתי ומהות התקלה.

* 1. **תוכנת יצירת קבצי קונפיגורציה באופן אוטומטי**

מאחר ויש ליצור קובץ cfg עבור כל שרת והקבצים די דומים במהותם וישנם שרתים רבים במכללה, נכתבה תוכנה ב- C# אשר מבקשת מהמשתמש נתונים כגון סוג השרת, כתובת IP, ותכונות לנטר, והיא יוצרת באופן אוטומטי את קובץ הקונפיגורציה.

* 1. **סקירפט התקנת NRPE**

התקנת ה-NRPE על שרתי הלינוקס הינה זהה, ואופן ההתקנה הוא על ידי הרצת פקודות ב-shell, ולכן נכתב סקריפט אשר מריץ את פקודות ההתקנה באופן אוטומטי.

מפאת רגישות יתר של השרתים במכללה, המערכת מומשה על 5 שרתים שהוקצו צורך הפרויקט, אך התאמת המערכת לכלל השרתים הקיימים הינה פשוטה ומהירה.

אציין שצוות התשתיות במכללה החליט להקים רשת ניטור נפרדת אשר תהיה מאובטחת ברמה גבוהה ביותר ותוכל להתחבר לשאר הרשתות במכללה ולנטר דרכה את כלל השרתים.  
כמו כן, נבחנת הסוגיה להתחבר ב- VPN למערכת ה-Nagios בכדי שצוות התשתיות יוכל לצפות בנתוני המערכת דרך הטלפון הסלולארי בכל רשת שהיא ולא רק בעת התחברות לרשת האלחוטית של המכללה.

7. בדיקות

7.1 **בדיקת התקנת** **NRPE**  
לאחר התקנת NRPE על שרתי Linux נבדקה ההתקנה בשני שלבים:  
1. בדיקת השרת המנוטר:   
- בדיקה כי NRPE daemon אכן רץ על שרת ומאזין לפורט 5666, שדרכו מתבצעת פעולת הניטור.  
 הבדיקה התבצעה על ידי פקודת ה- bash:" netstat -at |grep nrpe", כאשר הפלט המצופה הוא:  


- בדיקה כי NRPE deamon אכן מתפקד על ידי הפקודה: ""/usr/local/nagios/libexec/check\_nrpe -H localhost, הפלט המצופה הינו:  


2. בדיקת שרת ה-Nagios,השרת המנטר:  
- בדיקה כי ה-NRPE בשרת ה-Nagios מסוגל לתקשר עם ה- NRPE daemon שמצוי בשרת המנוטר.  
הבדיקה נעשתה ע"י הפקודה:" /usr/local/nagios/libexec/check\_nrpe -H <IP of Remote Server>"  
הפלט המצופה הינו:   
  
בשלב הזה התגלו בעיות שנפתרו על ידי הורדת ספריות בשם SSL.

7.2 **בדיקת קבצי קונפיגורציה**  
לאחר כתיבה ושינוי קבצי cfg, נבדק כי הקבצים על ידי הרצת הפקודה:   
"/usr/local/nagios/bin/nagios –v /usr/local/nagios/etc/nagios.cfg"  
במידה והקבצים תקינים הפלט יהיה:   
  
ולאחר מכן בוצע אתחול מחדש של ה- Nagios.

7.3 **בדיקת ה- plug-ins**ה-plug-ins הכתובים בשפת C נבדקו על ידי שימוש ב-plug-in וניטור באמצעותו על ידי הוספת service לקובץ קונפיגורציה של שרת מסוים, service זה בשדה ה- check command קורא ל- plug-in הרצוי עם הפרמטרים הנדרשים.  
לאחר הגדרה זו, נכונות ה- plug-in נבדקה על ידי התבוננות בממשק ה-GUI וראיית הנתונים הצפויים.  
למשל בדיקת plug-in שבודק את נפח הדיסק של השרת שבשימוש:



בדיקות אלו משמשות לשם בדיקת מערכת הניטור הכללית שמורכבת מתוספים, GUI, קבצי קונפיגורציה, NRPE ו-NSClient++.  
הבדיקות שתוארו למעשה מרכיבות את בדיקת המערכת כולה.  
לאחר שהמערכת הושלמה, נבדקו מקרים רבים שצריכים לגרום למערכת להתריע על בעיה בשרת בהם כיבוי השרת, ניצול יתר של ה-CPU, הורדת service-ים שונים והעלאת מספר התהליכים שרצים בשרת.  
המערכת הגיבה כמצופה והתריעה על מצב קריטי או אזהרה במערכת.

1. השוואה לפתרונות בספרות

בסביבת הארגון מנהלי המערכות זקוקים לכלים ייחודיים לשמירה על תקינות מערכות המידע.  
אחד הכלים הללו הוא ניטור הרשת הארגונית והרכיבים בה. בעולם הקוד הסגור אפשר למצוא מגוון רחב של כלים למשימה זו כדוגמת: GFI max, Spiceworks ואחרים. בעולם הקוד הפתוח ישנו מספר מועט יותר של פתרונות לניטור ואחד הכלים הפופולארי נקרא Nagios.  
תוכנת Nagios הינה עוצמתית, גמישה, ומבצעת בדיוק רצון המשתמש.  
האם זה אומר ש- Nagios מושלמת? לא בהכרח. התוכנה מעט מורכבת ומסורבלת להגדרה, ועשויה לגרום לכמה מנהלי רשתות להסמיק ולברוח, אך יחד עם זאת התוכנה שווה את כל זמן ההשקעה בלימודה ובהכרתה.  
מה מציעה Nagios?  
להלן מספר מאפיינים:  
- ניטור שירותי תקשורת (SMTP, POP3, HTTP, Ping ועוד...)  
- ניטור משאבי שרתים.  
- בדיקת שירותים בצורה מקבילית.  
- היררכיה טופולוגית של הרשת.  
- התראות והודעות על מצב המערכות.  
- אירועים.  
- ממשק Web לניהול ולבקרה.  
- פיתוח והטמעה של תוספים (Plug-ins).

באמצעות Nagios ניתן לנטר ציודי תקשורת כגון מתגים ונתבים. ניטור שרתי לינוקס, יוניקס, סולאריס ו- Windows, ניטור אפליקטיבי כדוגמת ניטור שרתי בסיסי נתונים Oracle, MySQL, Microsoft SQL Server ואפילו אתרי אינטרנט.

למעשה, מערכת ה-Nagios מוצעת כפיתרון חינמי הניתן לשינוי והתאמה עבור ארגונים המבקשים להגדיל את אמינות השרתים שלהם בכך שיוכלו להתמודד עם כל מצב אפשרי במהירות וביעילות.

1. מסקנות מן הפתרון והמימוש

* מלכתחילה תוכנן הפרויקט לעסוק בניטור ציוד התקשורת השונה וביצוע ממשק גרפי נוח ופשוט לשימוש אשר יחליף את ממשק ה-Nagios שהינו בעל פונקציונאליות בסיסית ביותר.  
  לאחר שנמצא כי קיים GUI אשר מציע את כל הפונקציונאליות הנדרשת, והוא מבוסס קוד פתוח לכן הוחלט להשתמש בו, להוסיף את ניטור השרתים לפרויקט, ובכך להרחיב את הפרויקט ולהציע לצוות התשתיות מערכת ניטור רחבה וכוללת יותר.  
  הפרויקט שלי מתמקד בניטור השרתים.
* דרך ניטור שרתי Linux שונה מדרך ניטור שרתי Windows.
* הקמת מערכת ה-Nagios כרוכה בעריכת קבצי קונפיגורציה רבים במערכת, בהתקנות add-ons על כל השרתים, וביצירת קובץ קונפיגורציה עבור כל אחד מהם המגדיר את התכונות השונות שיש לנטר בשרת זה.
* כל שרת אשר מעוניינים לנטר דורש יצירת קובץ קונפיגורציה המותאם אליו, לכן כדאי לכתוב תוכנה אשר מבקשת מהמשתמש את פרטי השרת (שרת windows/Linux, כתובת IP ושם השרת) ואת התכונות והשירותים אשר יש לנטר, והיא יוצרת באופן אוטומטי קובץ קונפיגורציה וזאת מאחר וקיימים שרתים רבים במכללה.
* הפרויקט דרש הקמת מערכת חדשה שלא הייתה מוכרת לי עד כה. ההתמודדות עם הקמתה נתנה לי המון ביטחון עצמי ללמוד תחומים חדשים שאף פעם לא הכרתי ואני משוכנעת שזה יתרום לי רבות בהמשך.

10. רשימת ספרות

[1] Wojciech Kocjan **"Learning Nagios 3.0"**  
 PACKT, 2008.

[2] Wolfgang Barth  
" **Nagios: System and Network Monitoring**"  
 ISBN, 2008.

[3] Chris Burges  
**"The Nagios Book"**  
Chris Burgess, 2005

[4] Ramesh Natarajan  
[**http://www.thegeekstuff.com/2008/06/how-to-monitor-remote-linux-host-using-nagios-30/**](%5b3%5d%20http://www.thegeekstuff.com/2008/06/how-to-monitor-remote-linux-host-using-nagios-30/)June 2008

[5]  [Amanda Folson](mailto:docs@linode.com)  
[**http://library.linode.com/server-monitoring/nagios/ubuntu-10.04-lucid**](http://library.linode.com/server-monitoring/nagios/ubuntu-10.04-lucid)  
May 2010

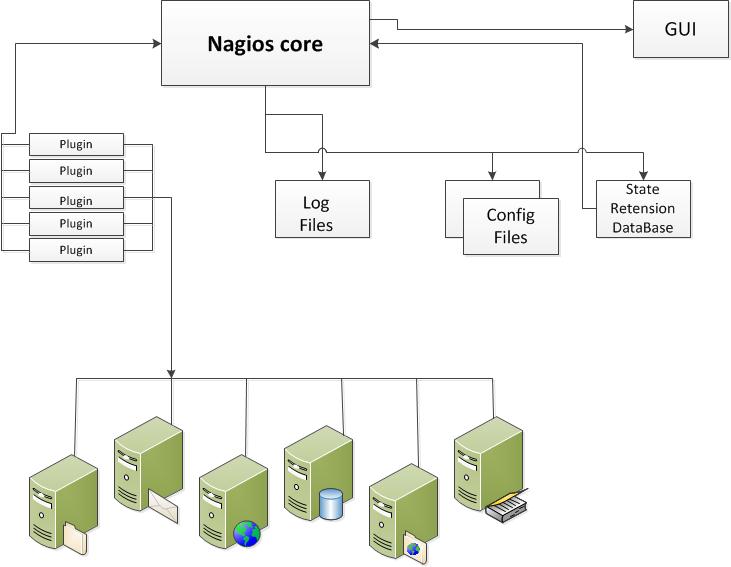
1. נספחים

**11.1 הצעות להמשך הפרויקט:**

* קיימת אפליקציה למערכת האנדרואיד שהינה מבוססת קוד פתוח.  
  האפליקציה מציגה למשתמש רק את ההתראות והבעיות הקיימות. יש לשפר את האפליקציה בכך שהיא תאפשר למשתמש יותר פונקציונאליות ותאפשר לו לבחור איזה סוג של התקנים היא רוצה לצפות בהם, שרתים או מתגים.  
  בנוסף, תינתן האפשרות להצגת השרתים אשר תקינים, מאחר וגם לפעמים מידע תקין הוא רלוונטי.
* אפשרויות הניטור הן רבות, וניתן לנטר דברים רבים ומגוונים.   
  ניתן להמשיך ולהרחיב את המערכת על ידי כתיבה והתאמת תוספים נוספים אשר ינטרו דברים שונים. לדוגמא: רישיונות תוכנה אשר תנטר ותודיע לצוות התשתיות פרק זמן מוגדר לפני שתוקף הרישיון יפוג.

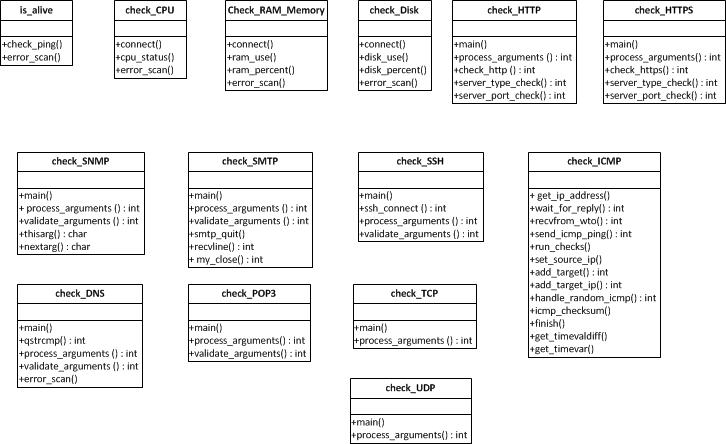
**11.2 תרשים מודולים:**

התרשים מתאר את תת המערכות במערכת ה-Nagios.  
תת המערכות כוללות את הליבה של ה-Nagios, ממשק ה-GUI, קבצי LOG, מסד נתונים אשר שומר את מצב המערכת לשמש לדוחות למשל, קבצי Configuration, והתוספים שבאמצעותם מנטרים את כלל השרתים.



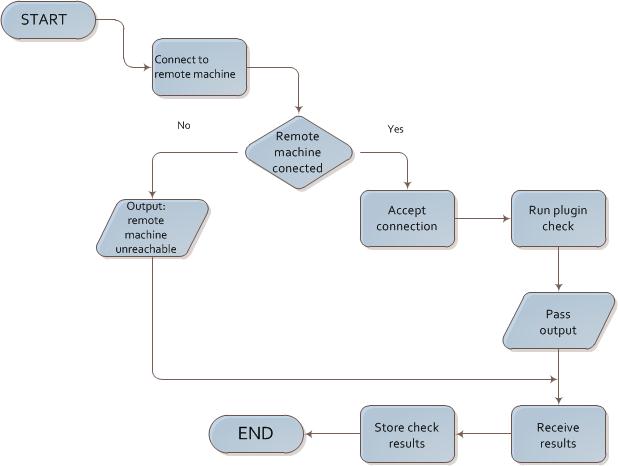
**11.3 תרשים מחלקות:**

כל תוסף מהווה יחידה עצמאית, על כן תרשים המחלקות מהווה מעין פירוט של התוספים.



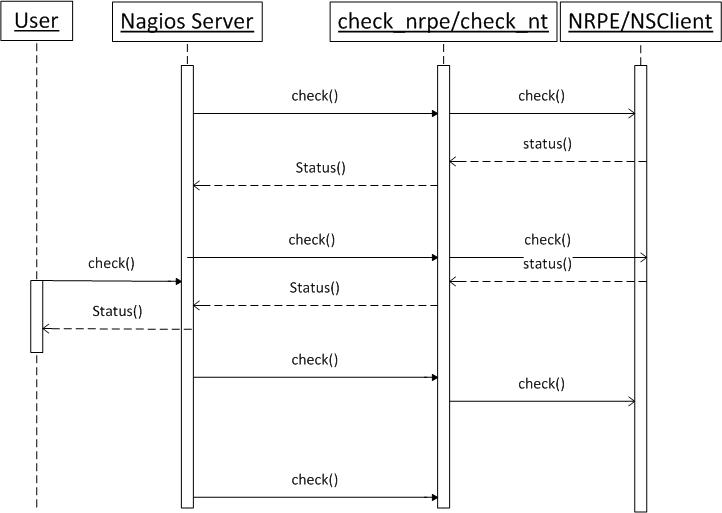
**11.4 תרשים זרימת נתונים:**

התרשים מציג את האופן בו מנוטר שרת על ידי תוסף מסויים.  
לאחר ביצוע התחברות לשרת נבדקת התכונה אוה בודק התוסף, המידע המתקבל נשלח לשרת ה-Nagios ומאוחסן במסד נתונים שקיים בשרת.  
אם לא התאפשר להתחבר לשרת, נשמרת תוצאה שהשרת לא זמין.



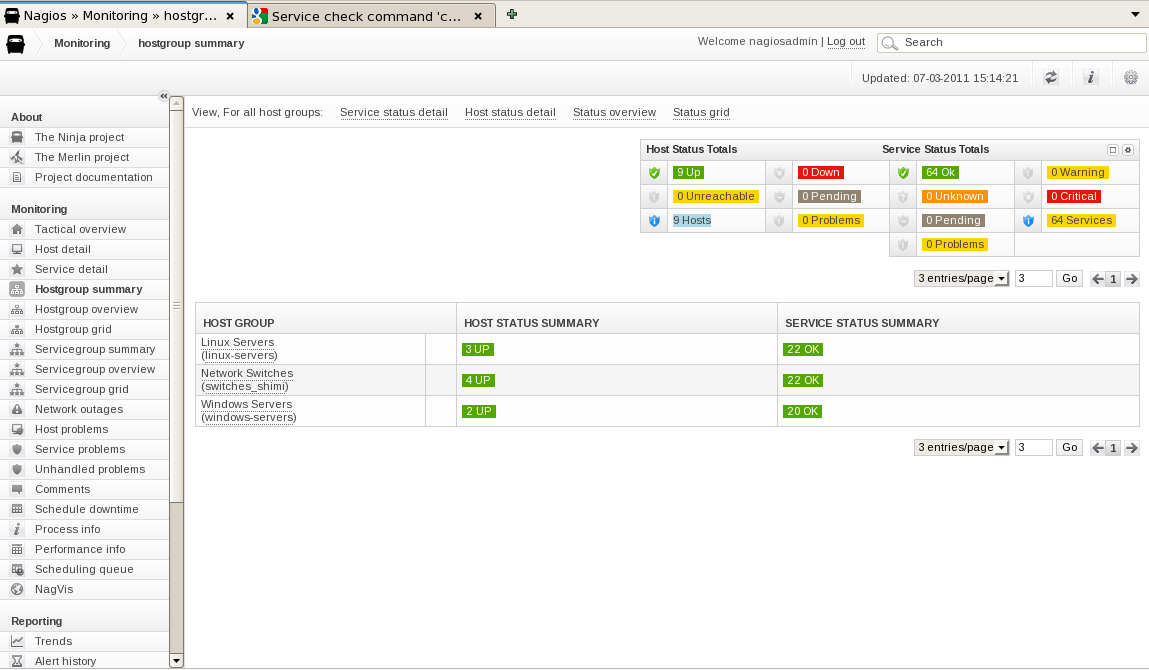
**11.5 תרשים סידרה בזמן:**

התרשים מציג את מערכת הניטור.  
שרת ה-Nagios כל פרק זמן קבוע בודק על ידי תוסף המותאם למערכת ההפעלה של השרת את מצב השרתים ביחס לתוספים אשר קיימים במערכת, הבדיקה נעשית באמצעות ה- add-on שמותקן בכל שרת.  
NRPE בשרתי Linux, ו- NSClient בשרתי Windows.  
כאשר המשתמש מעוניין בנתונים של המערכת הוא ייקבל את הנתונים שחזרו מהבדיקה האחרונה שהתבצעה על ידי שרת ה-Nagios.

****

**11.6 ממשק ה-GUI:**

כפי שמצויים בדו"ח, ישנו ממשק אשר מציג את הנתונים.לממשק יש כלים רבים המקוטלגים.  
אחד מהכלים שלו הוא הצגת השרתים במערכת ועל ידי לחיצה על אחד מהם ניתן את כל התכונות המנוטרות בו.

****

**11.7 קובץ קונפיגורציה לאחד מהשרתים:**

# HOST DEFINITIONS

#J-project1   
#Define a host for the Windows machine we'll be monitoring define host{

use windows-server ; Inherit default values from a template host\_name jproject-1 ; The name we're giving to this host alias jproject-1 server ; A longer name associated with the host address 62.90.88.145 ; IP address of the host  
}

# HOST GROUP DEFINITIONS  
# Define a hostgroup for Windows machines  
# All hosts that use the windows-server template will automatically be a member of this group

define hostgroup}

hostgroup\_name windows-servers ; The name of the hostgroup   
 alias Windows Servers ; Long name of the group  
{

# SERVICE DEFINITIONS  
# Create a service for monitoring the version of NSCLient++ that is installed

define service}

use generic-service  
 host\_name jproject-1  
 service\_description NSClient++ Version  
 check\_command check\_nt!CLIENTVERSION  
{

# Create a service for monitoring the uptime of the server

define service{

use generic-service  
 host\_name jproject-1  
 service\_description Uptime  
 check\_command check\_nt!UPTIME  
{

# Create a service for monitoring CPU load

define service}

use generic-service  
host\_name jproject-1  
service\_description CPU Load  
check\_command check\_nt!CPULOAD!-l 5,80,90  
{

# Create a service for monitoring memory usage

define service{

use generic-service  
host\_name jproject-1  
service\_description Memory Usage  
check\_command check\_nt!MEMUSE!-w 80 -c 90  
{

# Create a service for monitoring C:\ disk usage

define service}

use generic-service  
host\_name jproject-1  
service\_description C:\ Drive Space  
check\_command check\_nt!USEDDISKSPACE!-l c -w 80 -c 90  
{

# Create a service for monitoring the W3SVC service

define service{

use generic-service  
host\_name jproject-1  
service\_description W3SVC  
check\_command check\_nt!SERVICESTATE!-d SHOWALL -l W3SVC  
{

# monitoring the HTTP service on the remotehost machine:

define service{

use generic-service ; Inherit default values from a template host\_name jproject-1  
 service\_description Product Download Link  
 check\_command check\_http!-u /download/index.php -t 5  
}

1. Abstract

The final project was made at Jerusalem College of Engineering, for IT team   
that responsible for the maintenance of all College computing system, in particular many servers that provide many services to all students and staff college.

**The problem**: there are a lot of servers that constitute a major element in the computer system of the college therefore it is important to confirm that they are working proper.  
To this day the college had no system that can find and diagnose hitches. Up until the project finding, diagnosing and solving hitches was by complaints of the users

The project will give an indication to the status of the servers which will help the system crew to find the problems and will shorten the time of solving them.

When an exception occurs a massage is sent to the defined emails and the information will appear on the web site.

The monitoring system is based on an open source project named Nagios. Nagios is a host and service monitor designed to inform of network problems before the clients and users do.

Nagios is an open source tool released under the terms of the GNU (General public license).

The monitoring daemon runs intermittent checks on hosts and services you specify using external "plug-ins" which return status information to Nagios. When problems are encountered, the daemon can send notifications out to administrative contacts in email.  
Current status information, historical logs, and reports can all be accessed via web browser**.**



## SOFTWARE ENGINEERING DEPARTMENT

**Monitoring Servers Using Nagios**

**by**

### Shiran Ben Hamo

**July 2011 Tamuz 5771**

****

## SOFTWARE ENGINEERING DEPARTMENT

**Monitoring Servers Using Nagios**

**by**

**Shiran Ben Hamo**

**Supervisor: Mr. Shay Tavor**

**Approved by the Supervisor: Date: 07.07.11**

**Approved by the Industrial Supervisor: Date: 07.07.11**

**Approved by the Projects’ Coordinator: Date: 07.07.11**