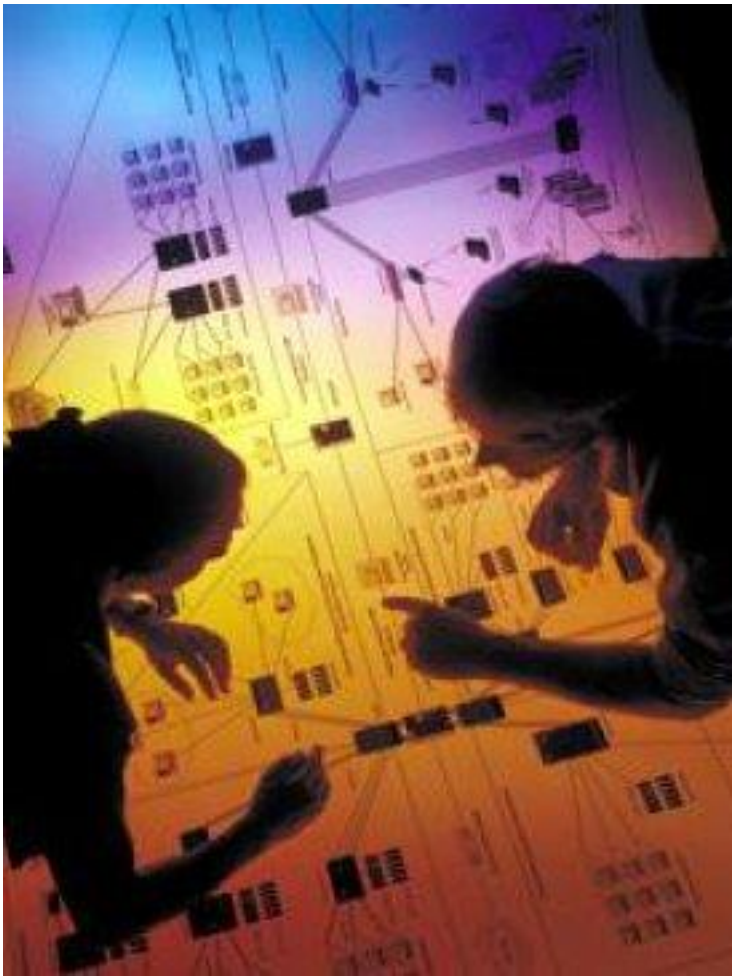
ΑΛΕΞΑΝΔΡΕΙΟ Τ. Ε. Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

Υλοποίηση συστήματος διαχείρισης δικτύου



Του φοιτητή
Κωνσταντίνου Μητρογεώργου
(011740)

Επιβλέποντες Καθηγητές:
Μπόζιος Ελευθέριος
Χαρχαλάκης Στέφανος

Θεσσαλονίκη 2011

...στην μνήμη του
Ελευθέριου Μπόζιου

Ευχαριστίες

Ευχαριστώ θερμά τον κ.Μπόζιο, παρόλο που δεν είναι πλέον μαζί μας, για την πολύτιμη βοήθεια που μου πρόσφερε καθ' όλη την διάρκεια της συγγραφής της παρούσας πτυχιακής εργασίας, καθώς επίσης και για τις πολύτιμες γνώσεις που αποκόμισα ασκώντας την πρακτική μου άσκηση υπό την επίβλεψή του στο νοσοκομείο Α.Χ.Ε.Π.Α. Στο πλαίσιο αυτό θα ήθελα να ευχαριστήσω και το προσωπικό του τμήματος πληροφορικής του Α.Χ.Ε.Π.Α. για την δυνατότητα πρόσβασης στο πληροφοριακό σύστημα του νοσοκομείου, όπως επίσης και τον συμφοιτητή μου Δημήτρη Διαμαντή, με τον οποίο δουλέψαμε μαζί, για την συνεργασία και την ανταλλαγή απόψεων πάνω στο αντικείμενο της διαχείρισης δικτύου του νοσοκομείου. Τέλος ευχαριστώ θερμά τον κ.Χαρχαλάκη για την βοήθεια που μου προσέφερε ώστε να διορθωθεί και να ολοκληρωθεί αυτή η εργασία.

Περίληψη

Ο λόγος εκπόνησης αυτής της πτυχιακής εργασίας ήτανε η υλοποίηση ενός συστήματος διαχείρισης δικτύου για το πληροφοριακό σύστημα του νοσοκομείου Α.Χ.Ε.Π.Α χρησιμοποιώντας λογισμικό ανοιχτού κώδικα, καθώς και η μελέτη διαφόρων συστημάτων και τεχνολογιών διαχείρισης για να επιλεχθεί το καλύτερο δυνατό για τις ανάγκες του προσωπικού του τμήματος πληροφορικής του νοσοκομείου. Το νέο αυτό σύστημα ήρθε να αντικαταστήσει το σύστημα διαχείρισης δικτύου Spectrum το οποίο ήτανε αρκετά παλιό, δεν ανταποκρινότανε στις σύγχρονες ανάγκες και στις νέες συσκευές που εγκαταστάθηκαν πρόσφατα στο περιβάλλον του νοσοκομείου.

Το σύστημα που επιλέχθηκε ήτανε το OpenNMS το οποίο εγκαταστάθηκε σε αφιερωμένο εξυπηρετητή και έγινε η παραμετροποίηση του, καθώς και η παραμετροποίηση των διαχειριζόμενων συσκευών ώστε να υπάρχει σωστή επικοινωνία με το σύστημα διαχείρισης δικτύου. Επίσης αναπτύχθηκαν κάποια εργαλεία χρησιμοποιώντας το περιβάλλον προγραμματισμού κελύφους bash τα οποία συμπληρώνουν την λειτουργικότητα του συστήματος διαχείρισης δικτύου δίνοντας λύση σε κάποια συγκεκριμένα προβλήματα που δεν μπορούσαν να καλυφθούν από το OpenNMS.

Η δομή της πτυχιακής εργασίας αποτελείται από τρεις ενότητες. Η πρώτη ενότητα (Κεφάλαια 1 έως 3) αποσκοπεί στην επεξήγηση για το τι είναι η διαχείριση δικτύου και πως λειτουργεί, στην παρουσίαση του πρωτόκολλου SNMP που αποτελεί το βασικό πρωτόκολλο για την διαχείριση δικτύων και στην παρουσίαση διαφόρων συστημάτων που δοκιμάστηκαν στο περιβάλλον του νοσοκομείου. Η δεύτερη ενότητα (Κεφάλαια 4 έως 10) έχει σαν σκοπό να παρουσιάσει τον τρόπο λειτουργίας του OpenNMS και να αποτελέσει εγχειρίδιο για το προσωπικό του νοσοκομείου. Τέλος, στην τρίτη ενότητα (Κεφάλαια 11 έως 13) παρουσιάζεται η παραμετροποίηση που έγινε για το συγκεκριμένο περιβάλλον και επεξηγείται ο τρόπος λειτουργίας των εργαλείων που αναπτύχθηκαν για να συμπληρώσουν την λειτουργικότητα του OpenNMS.

Abstract

The purpose of this thesis was the development of a network management system for the A.H.E.P.A. hospital information system using open source software, as well as the research of various systems and management technologies in order to choose the best possible solution and cover the staff needs of the hospital's informatics department. This new system is here to replace the Spectrum network management system which was dated, did not comply with the upcoming needs and the new devices recently installed within the hospital environment.

The system chosen was OpenNMS which was installed on a dedicated server and was configured together with the management devices in order to achieve efficient communication with the network management system. Additionally some specific tools where developed using the shell programming environment bash which complete the functionality of network management software thus resolving specific problems which could not be covered by OpenNMS.

The structure of this thesis consists of three sections. The first section (Chapters 1 to 3) aims to explain the meaning and function of network management, to present the SNMP protocol which is the basic protocol of network management and to present various systems tested within the hospital environment. The second section (Chapters 4 to 10) aims to present the way OpenNMS operates and to be used as a manual by the hospital staff. Finally, the third section (Chapters 11 to 13) presents the configuration of the specific environment and explains the operation of the tools developed in order to complete the functionality of OpenNMS.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πρώτη Ενότητα

Τι είναι η διαχείριση δικτύου, συστήματα διαχείρισης δικτύου και το πρωτόκολλο SNMP

ΚΕΦΑΛΑΙΟ 1

Διαχείριση Δικτύου

1.1. - Εισαγωγή

Τι ονομάζουμε διαχείριση δικτύων

Η διαχείριση δικτύου ορίζεται ως ένα σύνολο από λειτουργίες, ενέργειες, διαδικασίες και εργαλεία που χρησιμοποιούνται για τον έλεγχο της λειτουργίας και για την εκμετάλλευση του δικτύου.

Τι είναι διαχείριση δικτύου

Τα τελευταία χρόνια τα δίκτυα υπολογιστών και τα συστήματα κατανεμημένης επεξεργασίας έχουν γνωρίσει μεγάλη ανάπτυξη. Η τάση στην ανάπτυξη των συστημάτων αυτών είναι προς την κατεύθυνση μεγαλύτερων και περιπλοκότερων δικτύων τα οποία θα υποστηρίζουν περισσότερες εφαρμογές και περισσότερους χρήστες. Συνεπώς, έχει αυξηθεί σημαντικά η πιθανότητα να συμβεί κάποιο λάθος και έτσι ολόκληρο το δίκτυο ή ένα μέρος του να τεθεί εκτός λειτουργίας ή να μειωθεί η αξιοπιστία και η απόδοσή του. Ειδικά σε ένα μεγάλο τοπικό δίκτυο (που μπορεί να έχει έκταση ενός μεγάλου κτιριακού συγκροτήματος ή ενός Πανεπιστημίου), η συντήρηση και ο έλεγχος του μπορεί να είναι μια διαδικασία ασύμφορη, επίπονη και χρονοβόρα, που απαιτεί να ασχοληθούν αρκετοί άνθρωποι.

Οι παραπάνω λόγοι καθώς και η πολυπλοκότητα των δικτύων και η ύπαρξη συσκευών που ανήκουν σε διαφορετικούς κατασκευαστές, έχουν κάνει αναγκαία την ανάπτυξη εργαλείων που θα βοηθήσουν στην αυτόματη και αποτελεσματική διαχείριση των δικτύων. Έτσι έχουν αναπτυχθεί τα ανάλογα πρωτόκολλα και βάσεις διαχείρισης πληροφοριών καθώς και το αντίστοιχο λογισμικό το οποίο χρησιμοποιείται για να είναι εφικτή η διαχείριση του δικτύου. Με τον όρο **«Διαχείριση Δικτύου»** εννοούμε τη διαδικασία του αυτόματου (ή όσο το δυνατόν αυτοματοποιημένου) ελέγχου ενός οποιουδήποτε δικτύου υπολογιστών ώστε το κόστος συντήρησης του να είναι κατά το δυνατόν μικρότερο και η απόδοσή του η

καλύτερη δυνατή. Γενικά, οι βασικοί σκοποί της διαχείρισης του δικτύου είναι οι εξής:

1) Η διατήρηση της ικανοποιητικής και αξιόπιστης λειτουργίας ακόμη και κάτω από συνθήκες υπερφόρτωσης ή βλάβης, καθώς επίσης και κάτω από αλλαγές της διαμόρφωσης του δικτύου (εισαγωγή νέων συσκευών ή υπηρεσιών).

2) Η βελτίωση της απόδοσης του δικτύου, η οποία σχετίζεται με την ποιότητα και την ποσότητα των υπηρεσιών που παρέχονται στους χρήστες.

Τι χρειάζεται ένα σύστημα διαχείρισης

- **Network Management Console:**

– Ο σταθμός εργασίας όπου παρακολουθεί ο διαχειριστής την κατάσταση του δικτύου.

- **Network Management Protocol:**

– Το πρωτόκολλο με το οποίο θα επικοινωνεί με τις δικτυακές συσκευές.

- **Network Management Agent:**

– Το λογισμικό που εγκαθίσταται στην δικτυακή συσκευή για χρήση του πρωτοκόλλου διαχείρισης.

- **Δικτυακές συσκευές που τρέχουν SNMP agents.**

– δρομολογητές, κατανεμητές, hubs, εξυπηρετητές, εφαρμογές.

Τι προσφέρει η διαχείριση δικτύων

- **Προληπτικά**

– απομακρυσμένη διαχείριση.

– καταγραφή και σκιαγράφηση του δικτύου.

– Αλλαγές σε κομμάτια του δικτύου που ενδέχεται να υπάρξει πρόβλημα.

- **Σαν αντίδραση σε βλάβες και αστοχίες υλικού:**

– ειδοποίηση για προβλήματα στο δίκτυο.

– διάγνωση προβλημάτων.

– αυτόματη διαμόρφωση του δικτύου σε περίπτωση σφάλματος.

- **Αλληλεπιδραστικά:**

– Αλληλεπιδραστική αντιμετώπιση προβλημάτων.

Ποιες είναι οι βασικές ενέργειες διαχείρισης

- Έλεγχος λειτουργίας δικτύου
- Διοίκηση δικτύου (σε τακτικό επίπεδο)
- Ανάλυση δικτύου και βελτιστοποίηση (τακτικό και στρατηγικό επίπεδο)
- Σχεδίαση δικτύου (στρατηγικό επίπεδο)
- Ο απομακρυσμένος έλεγχος και αναδιαμόρφωση δικτυακών συσκευών
- Η παρακολούθηση του δικτύου μέσω των συσκευών που το συνθέτουν
- Η σύνθεση βάσεων δεδομένων με το “ιστορικό” της δραστηριότητας του δικτύου
- Η δυνατότητα τοποθέτησης παγίδων (traps) και συναγερμών (alarms) στις δικτυακές συσκευές

1.2. - Γενικά Χαρακτηριστικά ενός συστήματος διαχείρισης δικτύου(NMS).

Λαμβάνοντας υπόψη τις λειτουργίες που ένα διαχειριστικό σύστημα απαιτείται να υποστηρίζει, τα παρακάτω γενικά χαρακτηριστικά – περιορισμοί ενισχύουν την λειτουργικότητα του συστήματος:

Το σύστημα πρέπει να παρέχει ένα γραφικό σύστημα παρουσίασης της τοπολογίας του δικτύου. Είναι προτιμότερο η παρουσίαση να γίνεται με ιεραρχικό τρόπο και να υπάρχουν λογικές συνδέσεις μεταξύ των διαφορετικών επιπέδων της ιεραρχίας. Για παράδειγμα, σε ένα επίπεδο παρουσιάζονται μόνο τα LANs και οι συνδέσεις μεταξύ τους, ενώ σε κατώτερο επίπεδο παρουσιάζονται τα τμήματα (segments) του κάθε LAN, στο επόμενο επίπεδο οι κόμβοι των segments κ.ο.κ. Πρέπει ακόμη το σύστημα να είναι σε θέση να αναγνωρίζει τις συνδέσεις μεταξύ των επιπέδων και το πως αυτές συσχετίζονται με την απόδοση και τη λειτουργία ολόκληρου του δικτύου. Η ενοποιημένη εικόνα του διαχειριζόμενου δικτύου διατηρείται από το σύστημα, ενώ ο χρήστης μπορεί να επικεντρώνει την προσοχή του σε ορισμένα επίπεδα της ιεραρχίας. Είναι λειτουργικό, τέλος, να υπάρχει ομογενής αντιμετώπιση των στοιχείων του δικτύου σε επίπεδο διαπροσωπείας χρήστη, έστω και αν εσωτερικά υπάρχει ετερογένεια. Για παράδειγμα, σταθμοί εργασίας που διαχειρίζονται με διαφορετικά πρωτόκολλα πρέπει να παρουσιάζονται με τον ίδιο τρόπο στο χρήστη, και οι μέθοδοι άντλησης πληροφοριών για αυτούς να είναι όσο το δυνατόν παρόμοιοι. Οι ανομοιογένειες

πρέπει να κρύβονται από το χρήστη, εκτός βέβαια αν ζητηθούν ή αποτελούν αιτία προβλημάτων.

Το σύστημα πρέπει να είναι ικανό να συλλέγει όλες τις πληροφορίες από τους διαχειριζόμενους κόμβους, με όσο το δυνατόν μεγαλύτερη διαφάνεια και ιδανικά μέσω ενός μόνο πρωτοκόλλου διαχείρισης. Βέβαια, σε ετερογενή περιβάλλοντα το σύστημα πρέπει να είναι σε θέση να χρησιμοποιεί διαφορετικά πρωτόκολλα διαχείρισης και/ή proxy agents.

Η επεκτασιμότητα (expandability) και η δυνατότητα προσαρμογής σε διαφορετικές ανάγκες διαχείρισης (customization) είναι δύο ακόμη σημαντικά χαρακτηριστικά - απαιτήσεις. Δεν υπάρχει σύστημα που να καλύπτει τις ανάγκες διαχείρισης κάθε δυνατού δικτύου. Έτσι το σύστημα πρέπει να επιτρέπει την εύκολη προσθήκη νέων δυνατοτήτων και εργαλείων διαχείρισης ανάλογα με τις απαιτήσεις της κάθε εφαρμογής.

Μια ακόμη βασική λειτουργία ενός συστήματος διαχείρισης είναι η δυνατότητα ανίχνευσης και αναφοράς λαθών και προβλημάτων στο δίκτυο. Καθώς το διαχειριζόμενο δίκτυο επεκτείνεται, μια τέτοια υπηρεσία γίνεται όλο και περισσότερο πολύτιμη. Έστω και αν η διαχείριση λαθών δεν υποστηρίζεται, η ανίχνευση και η ειδοποίηση είναι απαραίτητα χαρακτηριστικά ενός διαχειριστικού συστήματος.

Το σύστημα πρέπει να παρέχει έναν αποδοτικό τρόπο φύλαξης του όγκου πληροφοριών που χρειάζεται για τη διαχείριση, ιδιαίτερα όταν τα διαχειριζόμενα δίκτυα είναι μεγάλα. Συχνά ένα σύστημα διαχείρισης βάσεων δεδομένων (DBMS) είναι απαραίτητο, καθώς εφαρμογές όπως το configuration και το accounting management είναι αδύνατον να λειτουργήσουν αποδοτικά χωρίς αυτό. Συνήθως χρησιμοποιείται το σχεσιακό μοντέλο (relational data model), ενώ γίνονται προσπάθειες να σχεδιαστούν και να υλοποιηθούν αντικειμενοστραφείς βάσεις δεδομένων (Object Oriented DBMS - OODBMS) ειδικά για χρήση σε συστήματα διαχείρισης δικτύων. Ανάλογα με την αρχιτεκτονική του NMS και τις απαιτήσεις απόδοσης μπορεί να χρησιμοποιηθεί κεντροποιημένη (centralized) ή κατανεμημένη (distributed) βάση δεδομένων.

Έχουν διατυπωθεί και άλλοι περιορισμοί που αφορούν την αλληλεπίδραση διαχειριζόμενου δικτύου και διαχειριστικού συστήματος, όπως μια μινιμαλιστική φιλοσοφία στις επιδράσεις του NMS στο διαχειριζόμενο δίκτυο που συνοψίζεται στο εξής : "Το αποτέλεσμα της εγκατάστασης ενός NMS σε ένα δίκτυο πρέπει να είναι το ελάχιστο δυνατό, αντανακλώντας τον ελάχιστο κοινό παρανομαστή" [ROSE91]. Η ανάγκη ελάχιστης επιρροής στους διαχειριζόμενους κόμβους ενισχύεται από τις μεγάλες διαφορές μεταξύ των κόμβων. Η διαδικασία άντλησης πληροφοριών και παρακολούθησης των κόμβων δεν πρέπει να προκαλεί σημαντικές καθυστερήσεις στη λειτουργία των κόμβων, καθώς κάτι τέτοιο οξύνει τις διαφορές απόδοσης. Τέλος, το φορτίο που εισάγει στο δίκτυο η λειτουργία του NMS πρέπει να είναι όσο το δυνατόν μικρότερο, αλλιώς το κέρδος της δυνατότητας διαχείρισης, αντισταθμίζεται από την παρενέργεια της πεσμένης απόδοσης και των προβλημάτων που μπορεί να προκαλέσει η συμφόρηση του δικτύου.

Μια άλλη απαίτηση είναι η βιωσιμότητα του διαχειριστικού συστήματος σε κρίσιμες καταστάσεις. Όταν το διαχειριζόμενο δίκτυο "πέφτει" και γενικά σε καταστάσεις σημαντικών προβλημάτων και λαθών, το NMS πρέπει να παραμείνει σε λειτουργία (σε όποιο βαθμό είναι αυτό δυνατό) [ROSE91]. Όσο περισσότερο ανεκτικό στα λάθη του διαχειριζόμενου δικτύου είναι το NMS, τόσο καλύτερα εκπληρώνει το ρόλο του σε περιπτώσεις προβλημάτων.

1.3. - Πρότυπα και Αρχιτεκτονικές Συστημάτων Διαχείρισης Δικτύων

Η αρχιτεκτονική που προτείνεται και χρησιμοποιείται σήμερα για τη διαχείριση τηλεπικοινωνιακών δικτύων και δικτύων υπολογιστών αποτελείται από το σύστημα διαχείρισης των δικτύων (Network Management System, NMS) ή το Σύστημα Λειτουργίας (Operation Systems, OS) και τα στοιχεία εκείνα των δικτύων (Network Elements, NE) τα οποία θέλουμε να διαχειριστούμε. Τέτοια NE΄s σε ένα δίκτυο είναι κυρίως μηχανήματα αποθήκευσης ή επεξεργασίας πληροφοριών, όπως hosts (workstation, terminals, servers κ.α.), καθώς και μηχανήματα διασύνδεσης δικτύων, όπως δρομολογητές, κατανεμητές, repeaters κ.α. στα οποία τρέχουν διαδικασίες διαχείρισης που ονομάζονται agents και είναι υπεύθυνες για την εκτέλεση των συναρτήσεων που καλούν τα συστήματα διαχείρισης. Για τη μεταφορά της πληροφορίας μεταξύ των διαχειριστικών συστημάτων και των

διαχειριζόμενων στοιχείων χρησιμοποιούνται κατάλληλα πρωτόκολλα μεταφοράς της πληροφορίας που αφορούν τη διαχείριση. Τα πρωτόκολλα αυτά καθορίζουν με σαφήνεια τον τρόπο επικοινωνίας, τη μορφή και τη σημασία των μηνυμάτων που θα ανταλλαχθούν, όπως επίσης και τον τρόπο ορισμού και περιγραφής των στοιχείων που θέλουμε να διαχειριστούμε. Ένα από τα γνωστότερα πρωτόκολλα αυτά είναι το SNMP (Simple Network Management Protocol), το οποίο συμπληρώνεται με τις προδιαγραφές για τη δομή της πληροφορίας που αφορά τη διαχείριση (Structure of Management Information, SMI) και τη βάση πληροφορίας διαχείρισης (Management Information Base, MIB), προϊόντα του Internet Architecture Board (IAB), της επιτροπής που εγκρίνει πρότυπα Request For Comments (RFCs) για την ομάδα πρωτοκόλλων TCP/IP.

1.3.1. - Γενικό Μοντέλο και οντότητες διαχείρισης - Standards

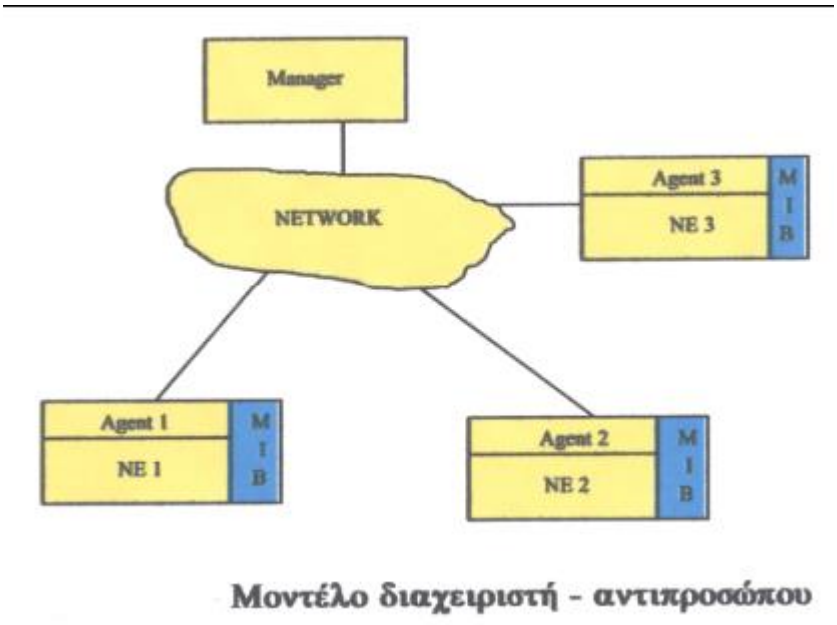
Τα συστήματα διαχείρισης ενός δικτύου αντιγράφουν τη γνωστή λογική του συστήματος **πελάτη–εξυπηρετητή**, μόνο που στην περίπτωση ενός συστήματος διαχείρισης ο πελάτης ονομάζεται **διαχειριστής** και ο εξυπηρετητής ονομάζεται **αντιπρόσωπος** (Χατζημίσιος,2007)**.**

Γενικότερα ένα σύστημα διαχείρισης αποτελείται από:

Το **διαχειριστή** (**manager)** που είναι ένα **πρόγραμμα (λογισμικό)** που εκτελείται σε κάποιο μηχάνημα του δικτύου και το οποίο χρησιμοποιεί ο υπεύθυνος συντήρησης του δικτύου (network administrator) για να στείλει εντολές διαχείρισης. Οι εντολές διαχείρισης μπορούν για παράδειγμα να αλλάζουν ρυθμίσεις σε μια δικτυακή συσκευή (χωρίς να χρειάζεται να μετακινηθούμε στο σημείο που βρίσκεται η συσκευή αυτή) ή ακόμα και να ελέγχει την κατάσταση λειτουργίας ενός τμήματος του δικτύου από μακριά.

Τα **διαχειριζόμενα στοιχεία δικτύου (Network Elements - NE)** τα οποία είναι δικτυακές συσκευές που συναντάμε σε ένα τοπικό δίκτυο όπως γέφυρες, δρομολογητές, modems, επαναλήπτες κλπ. Πολλές από αυτές τις συσκευές έχουν δυνατότητα απομακρυσμένης διαχείρισης. Για παράδειγμα, ένας δρομολογητής μπορεί να μας επιτρέπει να αλλάζουμε τις ρυθμίσεις του (π.χ. πίνακας δρομολόγησης) από κάποιο μηχάνημα του δικτύου χρησιμοποιώντας ιστοσελίδες.

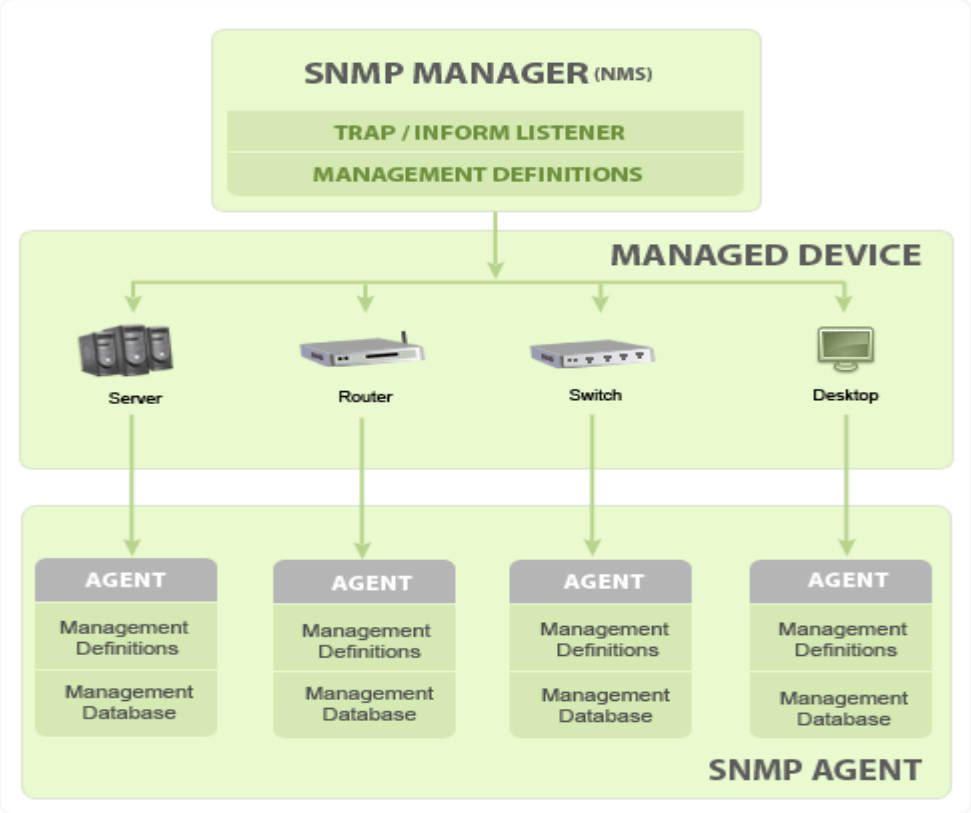
Τους **αντιπροσώπους (Agents)** που είναι επίσης **προγράμματα (λογισμικό)** τα οποία βρίσκονται εγκατεστημένα σε κάθε **διαχειριζόμενο στοιχείο δικτύου** με σκοπό να καταστήσουν δυνατή την επικοινωνία τους με το **διαχειριστή.** Η διαχείριση γίνεται με τον εξής τρόπο. Ο διαχειριστής (manager) στέλνει τις κατάλληλες εντολές διαχείρισης και ελέγχου μέσω του πρωτοκόλλου διαχείρισης δικτύου. Οι εντολές αυτές λαμβάνονται από τους agents στους οποίους απευθύνονται. Οι αντιπρόσωποι εκτελούν τις εντολές αυτές στα διαχειριζόμενα στοιχεία δικτύου (NE) που ελέγχουν.



Σχήμα 1.1 – Μοντέλο διαχειριστή - αντιπροσώπου

Τη **βάση πληροφοριών διαχείρισης (Management Information Base - MIB),** η οποία είναι μια βάση δεδομένων που μοιράζονται μεταξύ τους οι διαχειριστές και αντιπρόσωποι και η οποία περιέχει πληροφορίες σχετικά με τα διαχειριζόμενα στοιχεία δικτύου (NE). Η βάση πληροφοριών διαχείρισης περιέχει επίσης πληροφορίες που καθορίζουν τη δομή του περιεχομένου της διαχειριζόμενης πληροφορίας (Πρόκειται για μια κανονική βάση δεδομένων: Περιέχει και πίνακες που περιγράφουν τη δομή των πινάκων που περιέχουν τις πληροφορίες της βάσης). Σχεδιαστικά η MIB απεικονίζεται με μορφή δέντρου ενώ τα περιεχόμενα της αναπαρίστανται από τα φύλλα του δέντρου.

Τα **Πρωτόκολλα Διαχείρισης Δικτύου (Network Management Protocols - NMP),** με τη βοήθεια των οποίων γίνεται η διαχείριση των NE καθώς και η επικοινωνία μεταξύ του **διαχειριστή** και των **agents.** Το πρωτόκολλο που χρησιμοποιείται ευρύτατα για τη διαχείριση σε TCP/IP δίκτυα είναι το **Simple Network Management Protocol** (**SNMP**) και το οποίο θα αναλύσουμε στο επόμενο κεφάλαιο. Για δίκτυα τα οποία βασίζονται στο μοντέλο OSI έχει αναπτυχθεί το πρωτόκολλο διαχείρισης πληροφορίας (**CMIP**). Πιο εξελιγμένες εκδόσεις του SNMP αποτελούν η **SNMPv2** και η **SNMPv3**.



Σχήμα 1.2 – Μοντέλο διαχείρισης δικτύου

1.3.2. - Αρχιτεκτονικές Διαχείρισης Δικτύου

Οι αρχιτεκτονικές διαχείρισης του δικτύου που υπάρχουν είναι η **Κεντρική, η Ιεραρχική και η Κατανεμημένη.** Μια παραλλαγή τους που συνδυάζει τα δύο τελευταία είναι το δικτυωμένο NMS. Οι διαφορές αυτών των αρχιτεκτονικών αναφέρονται κυρίως στον αριθμό διαχειριστών και στον βαθμό επικοινωνίας - ανεξαρτησίας τους. Κάθε μια προσφέρει κάποια πλεονεκτήματα και μειονεκτήματα έναντι των άλλων. Η επιλογή εξαρτάται από της απαιτήσεις διαχείρισης και τον

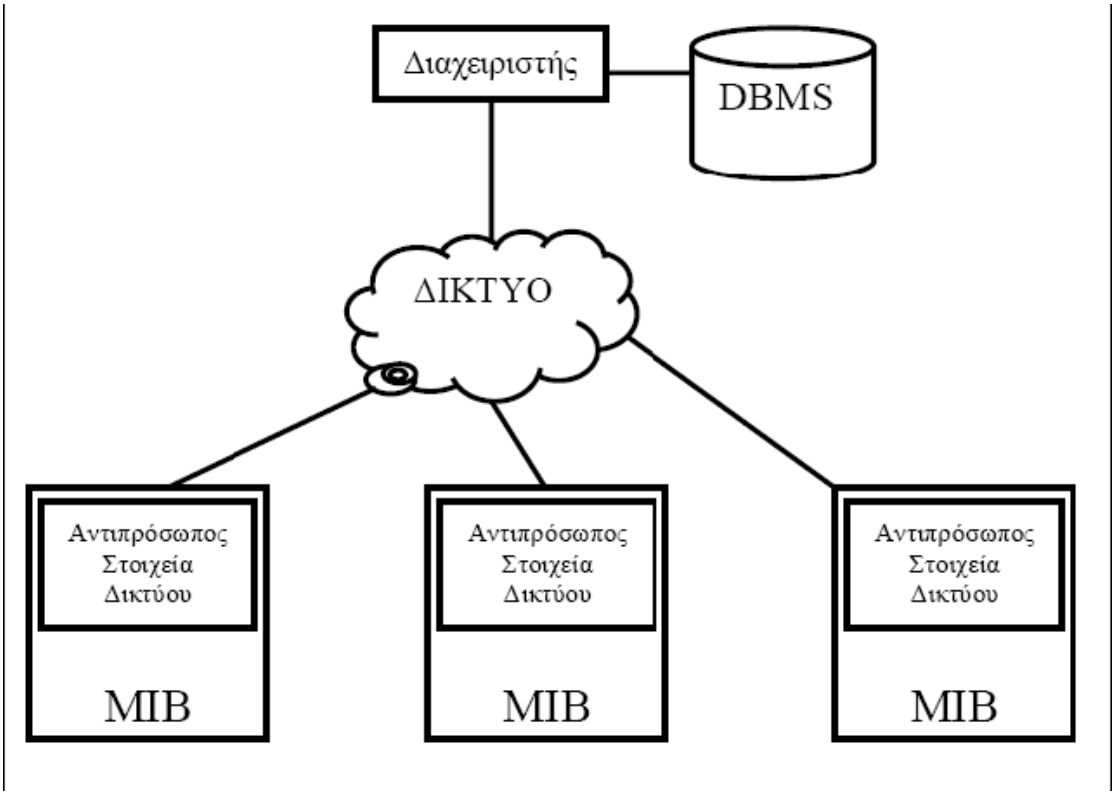
χαρακτήρα του δικτύου που απαιτείται η διαχείριση. Περιγραφή των τεσσάρων αρχιτεκτονικών, των πλεονεκτημάτων και μειονεκτημάτων τους δίνεται παρακάτω.

1.3.2.1. - Κεντρική Αρχιτεκτονική Διαχείρισης (Centralized)

Η κεντρική αρχιτεκτονική διαχείρισης έχει την πλατφόρμα διαχείρισης δικτύου σε ένα μόνο σταθμό εργασίας που είναι υπεύθυνος για όλα τα καθήκοντα διαχείρισης του δικτύου και είναι σύμφωνη με το γνωστό μοντέλο agent-manager. Περιέχει ένα κεντρικό διαχειριστή ο οποίος:

- Επικοινωνεί με όλα τα διαχειριζόμενα στοιχεία του δικτύου.
- Διαχειρίζεται την αποθήκευση των πληροφοριών του συστήματος
- Παρέχει μία ενοποιημένη εικόνα του διαχειριζόμενου δικτύου στο διαχειριστή μέσω κατάλληλου περιβάλλοντος επικοινωνίας με το χρήστη.

Η αποθήκευση δεδομένων μπορεί να είναι κεντρική ή κατανεμημένη για λόγους ασφάλειας αλλά ο έλεγχος είναι καθαρά κεντροποιημένος όπως και όλη η φιλοσοφία της αρχιτεκτονικής.



Σχήμα 1.3 – Κεντρική Αρχιτεκτονική Διαχείρισης Δικτύου

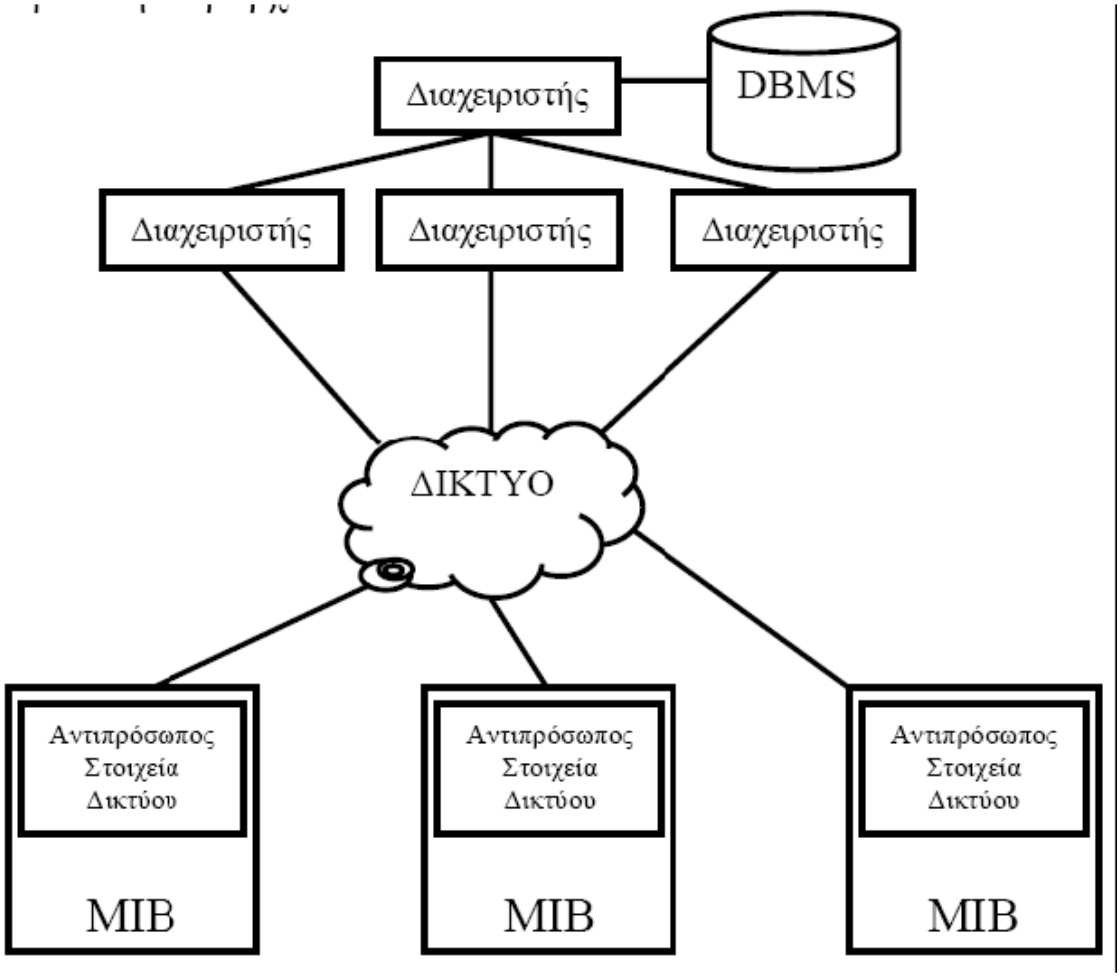
1.3.2.2. - Ιεραρχική Αρχιτεκτονική Διαχείρισης (Hierarchical)

Η ιεραρχική αρχιτεκτονική χρησιμοποιεί πολλαπλούς σταθμούς διαχείρισης. Οι πελάτες δεν έχουν πολλά Συστήματα Διαχείρισης Βάσεων Δεδομένων αλλά κάνουν χρήση του ΣΔΒΔ του κεντρικού εξυπηρετητή. Οι ομότιμοι διαχειριστές διαχειρίζονται από τον διαχειριστή που βρίσκεται στο υψηλότερο επίπεδο ιεραρχίας, τον MOM (Manager of Managers).

Κάποιες λειτουργίες εκτελούνται από τον MOM, ενώ άλλες από τους κατώτερους managers. Ο MOM έχει ανάλογο ρόλο με τον κεντρικό διαχειριστή της κεντροποιημένης αρχιτεκτονικής και συγκεντρώνει τις σημαντικές πληροφορίες, αφήνοντας τις λεπτομέρειες στους διαχειριστές του χαμηλότερου επιπέδου, οι οποίοι μπορεί να έχουν κονσόλα ή άνθρωπο χειριστή ή να παρακολουθούνται αυτόματα.

Η ιεραρχική αρχιτεκτονική διαχείρισης:

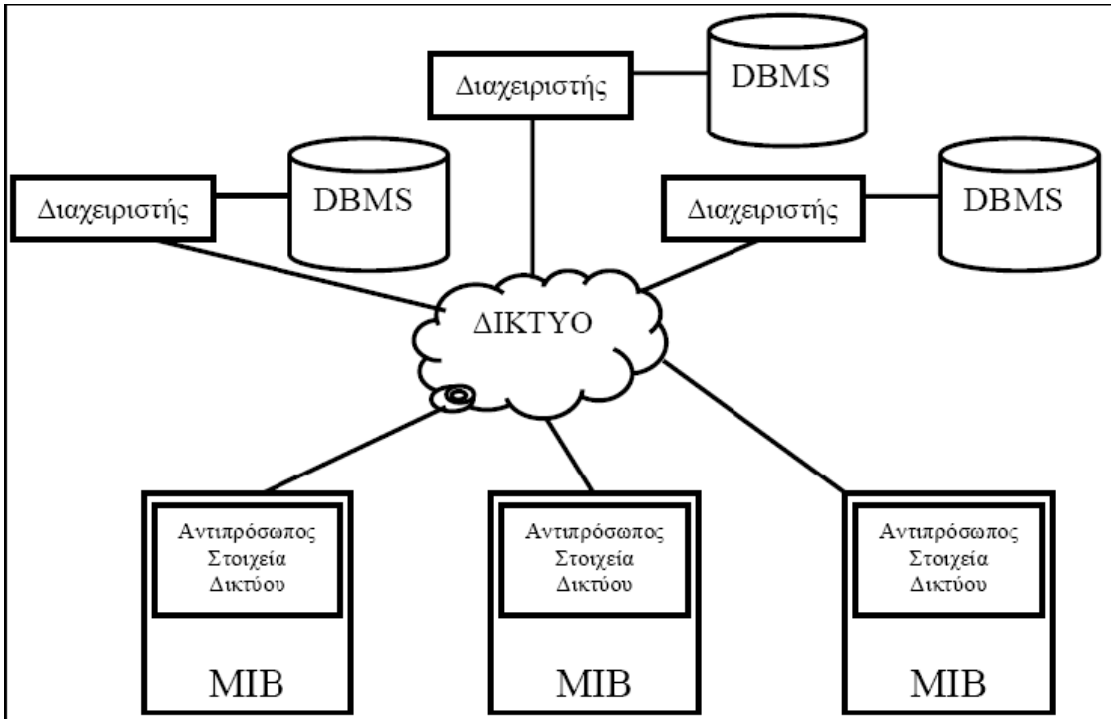
- Προσφέρει καλύτερο έλεγχο και επίδοση στο διαχειριστή του δικτύου
- Προσφέρεται για ετερογενή δίκτυα
- Βρίσκει εφαρμογή και σε δίκτυα που υπάρχει ανάγκη διαίρεσης του διαχειριζόμενου περιβάλλοντος
- Παρέχει
    - Ολοκληρωμένο διαχειριστικό περιβάλλον
    - Ενοποιημένη αναπαράσταση ετερογενούς δικτύου
    - Κοινό περιβάλλον επικοινωνίας με το χρήστη



Σχήμα 1.4 – Ιεραρχική Αρχιτεκτονική Διαχείρισης Δικτύου

1.3.2.3. - Κατανεμημένη Αρχιτεκτονική Διαχείρισης (Distributed)

Η κατανεμημένη αρχιτεκτονική διαχείρισης συνδυάζει την κεντροποιημένη με την ιεραρχική και έχει μία κεντρική πλατφόρμα διαχείρισης ή μια ιεραρχία από πλατφόρμες εξυπηρετητή-πελάτη. Χρησιμοποιεί ομότιμες πλατφόρμες διαχείρισης που καθεμιά τους χωριστά αποτελεί ένα κεντροποιημένο σύστημα. Κάθε ατομική ομότιμη πλατφόρμα μπορεί να έχει μία πλήρη ΒΔ για στοιχεία δικτύου που μπορεί να βρίσκονται σε οποιοδήποτε σημείο του δικτύου. Η διαχείριση κατανέμεται και αντιμετωπίζεται από τοπικούς διαχειριστές έχοντας λιγότερες απαιτήσεις σε υλικό και υπολογιστική ισχύ σε σχέση με την κεντροποιημένη διαχείριση.



Σχήμα 1.5 – Κατανεμημένη Αρχιτεκτονική Διαχείρισης Δικτύου

Γενικά, η κατανεμημένη αρχιτεκτονική

- Συνδυάζει την κεντροποιημένη με την ιεραρχική
- Έχει μία κεντρική πλατφόρμα διαχείρισης ή μια ιεραρχία από πλατφόρμες εξυπηρετητή πελάτη
- Χρησιμοποιεί ομότιμες πλατφόρμες διαχείρισης που καθεμιά τους χωριστά αποτελεί ένα κεντροποιημένο σύστημα.

1.3.2.4. - Δικτυωμένο NMS

Η αρχιτεκτονική αυτή συνδυάζει στοιχεία από τις αρχιτεκτονικές του κατανεμημένου και ιεραρχικού NMS. Εδώ έχουμε περισσότερους από ένα MOM (Integrated Managers), καθένας από τους οποίους διαχειρίζεται μια ομάδα managers, οι οποίοι με τη σειρά τους διαχειρίζονται μία ομάδα κόμβων (έννοια manager domain). Η αρχιτεκτονική αυτή ενθαρρύνεται από το γεγονός ότι όλο και περισσότερα διαχειριστικά συστήματα αναπτύσσουν τυποποιημένα interface, διευκολύνοντας έτσι την επικοινωνία με άλλα συστήματα. Το OSI Network Management Forum υποστηρίζει την προσπάθεια να επιτευχθεί ένα ευέλικτο και ισχυρό σύστημα που να ακολουθεί αυτή την αρχιτεκτονική. Η αρχιτεκτονική του δικτυωμένου NMS συνδυάζει τα πλεονεκτήματα των κατανεμημένων και

ιεραρχικών συστημάτων που παρουσιάστηκαν παραπάνω. Το βασικότερο μειονέκτημα είναι ότι αυξάνει τον αριθμό των διαχειριστικών συστημάτων που χρησιμοποιεί, γεγονός που αυξάνει το κόστος. Βέβαια, ο ιεραρχικός χαρακτήρας αυτής της αρχιτεκτονικής μπορεί να οδηγήσει στην μείωση του αριθμού των χειριστών στο κέντρο διαχείρισης, αλλά δεν μειώνει τον αριθμό των διαχειριστικών συστημάτων που πρέπει να αγοραστούν και να συντηρηθούν. Όπως προαναφέρθηκε, μια τεχνική που οδηγεί σε μείωση του αριθμού των συστημάτων, είναι η μέθοδος της πλατφόρμας διαχείρισης. Τέλος, αξίζει να σημειωθεί ότι οι τρεις προηγούμενες αρχιτεκτονικές προσφέρουν λύσεις στα προβλήματα της ολοκληρωμένης διαχείρισης σε ετερογενή δίκτυα.

Ειδικότερα η ιεραρχική και κατανεμημένη (peer-to-peer) αντιμετώπιση έχουν προταθεί για χρήση στην διαχείριση multi domain δικτύων. Τέτοια δίκτυα είναι σαφώς χωρισμένα σε ξεχωριστά τμήματα (domains) που διαχειρίζονται από τοπικούς διαχειριστές (εξαρτώμενοι από τους κατασκευαστές των τμημάτων). Τα προβλήματα που αντιμετωπίζονται είναι η έλλειψη ολοκληρωμένης πληροφορίας και η περιορισμένη δυνατότητα ελέγχου των τοπικών διαχειριστών. Έχουν προταθεί αλγόριθμοι που αντιμετωπίζουν αυτά τα προβλήματα και εφαρμόζονται στην κατανεμημένη αρχιτεκτονική, όπου ο κάθε διαχειριστής "βλέπει" τα άλλα τμήματα ως μοναδικούς κόμβους μέσω των τοπικών διαχειριστών.

1.4. - Απαιτήσεις του συστήματος διαχείρισης του δικτύου

Οι πιο σημαντικές λειτουργικές περιοχές στη διαχείριση ενός δικτύου όπως ορίζονται από το Διεθνή Οργανισμό Προτυποποίησης (**ISO**) είναι οι ακόλουθες:

1.4.1. - Διαχείριση σφαλμάτων ή βλαβών (Fault Management)

Η διαχείριση σφαλμάτων είναι η διαδικασία αναγνώρισης της ύπαρξης σφαλμάτων-βλαβών, ο εντοπισμός του σημείου που υπάρχει το πρόβλημα, η επίλυση του προβλήματος ή τεκμηρίωσή του και η προώθηση της περιγραφής του προβλήματος σε άλλη ομάδα. Τα προβλήματα έρχονται με τη μορφή συναγερμού και συνήθως γίνεται καταγραφή σε αρχεία ή/και με μορφή αλλαγής της χρωματικής ένδειξης σε κάποιες γραφικές απεικονίσεις του δικτύου. Η λύση των προβλημάτων διαφέρει κάθε φορά και εξαρτάται όπως είναι φυσικό από το πρόβλημα. Μπορεί να

χρειαστεί: αποσύνδεση προβληματικών συσκευών, αντικατάσταση ελαττωματικού υλικού, ρύθμιση παραμέτρων.

Όταν συμβεί κάποιο σφάλμα είναι αναγκαίο όσο το δυνατόν συντομότερα:

- Να προσδιορισθεί που βρίσκεται το σφάλμα ή η βλάβη
- Να απομονωθεί το υπόλοιπο του δικτύου, έτσι ώστε να μπορεί να λειτουργεί χωρίς παρεμβολές
- Να αναδιαμορφωθεί το δίκτυο έτσι ώστε να ελαχιστοποιηθεί η επίδραση από τη βλάβη σε κάποιο ή κάποια στοιχεία του
- Να επισκευαστεί ή να αντικατασταθεί το στοιχείο με τη βλάβη έτσι ώστε να επανέλθει το δίκτυο στην αρχική του κατάσταση

1.4.2. - Διαχείριση Διαμόρφωσης (Configuration Management)

Η διαχείριση διαμόρφωσης είναι η διαδικασία αλλαγής της τοπολογίας του δικτύου καθώς και η ρύθμιση των παραμέτρων των συσκευών που το αποτελούν, είτε σε επίπεδο υλικού, είτε σε επίπεδο λογισμικού, προκειμένου να εξασφαλίσουμε τη σωστή λειτουργία του δικτύου ανάλογα με τις εκάστοτε απαιτήσεις. Η αρχική εγκατάσταση του δικτύου και η ρύθμιση των παραμέτρων των συσκευών, δεν αποτελούν, σύμφωνα με τον επίσημο ορισμό, μέρος της διαχείρισης διαμόρφωσης, παρόλο που συχνά χρησιμοποιούνται τα ίδια εργαλεία. Ένα από τα σημαντικότερα έργα είναι η τεκμηρίωση, τόσο των συσκευών που αποτελούν το δίκτυο με τις ρυθμίσεις που έχουν, όσο και της τοπολογίας του δικτύου και του τρόπου λειτουργίας. Η τεκμηρίωση του δικτύου βοηθιέται από την ύπαρξη λογισμικού, που ανακαλύπτει και καταγράφει σε βάση δεδομένων καταλόγου υλικών όλες τις συσκευές ενός δικτύου, καθώς και τον τρόπο διασύνδεσής τους. Τα παραπάνω μπορούν να σχηματιστούν και σε γραφικές απεικονίσεις.

1.4.3. - Διαχείριση κοστολόγησης (Accounting Management)

Η διαχείριση κοστολόγησης ενός δικτύου περιλαμβάνει:

- την παρακολούθηση της χρήσης των πόρων του δικτύου
- την ανάλυση των διαθέσιμων επιπέδων των πόρων του δικτύου για συγκεκριμένες ομάδες χρηστών
- την αντιστοίχηση του κόστους της χρήσης

- την καταγραφή της χρήσης των δικτυακών πόρων από τις διάφορες ομάδες χρηστών
- την εξασφάλιση ότι οι χρήστες δεν κάνουν χρήση υπηρεσιών, που δεν έχει συμφωνηθεί ότι θα χρησιμοποιήσουν

Θα πρέπει η διαχείριση του δικτύου να μπορεί να ορίσει τις παραμέτρους που θα καταγράφονται στους διάφορους κόμβους, τα χρονικά διαστήματα στα οποία θα αποστέλλονται αυτές οι πληροφορίες σε ανώτερους κόμβους, καθώς επίσης και τον αλγόριθμό που θα χρησιμοποιηθεί για την κοστολόγηση. Θα ήταν καλό στο σημείο αυτό να γίνει σαφές ότι η διαχείριση κοστολόγησης δεν σημαίνει αναγκαστικά και χρέωση.

1.4.4. - Διαχείριση απόδοσης (Performance Management)

Για την διαχείριση της απόδοσης του δικτύου πρέπει να ορίσουμε τι θέλουμε να μετράμε, να σχεδιάσουμε τον τρόπο που θα γίνονται οι μετρήσεις και στην συνέχεια να υλοποιήσουμε αυτές τις μετρήσεις.
Σε ένα δίκτυο υπάρχουν διάφορα χαρακτηριστικά που μπορεί να θέλουμε να μετρήσουμε και να καταγράψουμε, όπως:

- το ποσοστό χρησιμοποίησης των WAN γραμμών ή διαφόρων τμημάτων τοπικού δικτύου
- η ανάλυση του ποσοστού κίνησης ανά πρωτόκολλο
- το ποσοστό λαθών σε σχέση με όλη τη κίνηση
- ο χρόνος καθυστέρησης διαφόρων σημείων του δικτύου
- ο χρόνος απόκρισης κάποιων συσκευών
- ο καθορισμός ορίων σε μερικές παραμέτρους. Αν οι τιμές των παραμέτρων υπερβούν τις τιμές των ορίων, δημιουργούνται κάποιοι συναγερμοί

Οι μετρήσεις αποθηκεύονται και αναλύονται από τους διαχειριστές του δικτύου. Σε περίπτωση που εντοπιστούν σημεία συμφόρησης ή προβληματικής λειτουργίας του δικτύου μπορεί να πραγματοποιηθεί ανασχεδίαση μερικών σημείων του δικτύου. Στην συνέχεια γίνονται νέες μετρήσεις για να ελεγχθεί κατά πόσο οι αλλαγές πέτυχαν τον σκοπό τους. Οι τιμές των μετρήσεων μπορούν να

καταγραφούν σε πίνακες ή/και σε μορφή γραφημάτων. Το κομμάτι της πρόβλεψης πιθανών προβλημάτων σχετίζεται άμεσα με τη διαχείριση απόδοσης του δικτύου.

1.4.5. - Διαχείριση ασφάλειας (Security Management)

Η διαχείριση ασφάλειας περικλείει:

- τον έλεγχο πρόσβασης σε συσκευές, δεδομένα και προγράμματα, απέναντι σε κάθε μη εξουσιοδοτημένη χρήση, ηθελημένη ή μη
- την επίβλεψη για προσπάθειες παραβίασης των κανόνων ασφαλείας
- την λήψη των απαραίτητων μέτρων προκειμένου να εξασφαλιστεί η ασφάλεια των πόρων του δικτύου

Τα μέτρα ασφαλείας πρέπει να περιλαμβάνουν όλους τους τομείς που συγκροτούν το πληροφοριακό σύστημα και πρέπει να αφορούν:

- τη φυσική προστασία των πόρων του συστήματος από μη εξουσιοδοτημένη πρόσβαση
- την ασφάλεια των συστημάτων που συνδέονται σε δίκτυο
- την ασφάλεια του δικτύου και την προστασία των δεδομένων που μεταφέρονται με αυτό

1.5. - Επίλογος

Στο κεφάλαιο αυτό έγινε αναφορά για το τι είναι η διαχείριση δικτύου συνολικά. Επεξηγήσαμε τι ακριβώς είναι η διαχείριση δικτύου σαν διαδικασία, τι προσφέρει και τι χρειαζόμαστε από την μεριά του λογισμικού για να υλοποιήσουμε ένα σύστημα διαχείρισης δικτύου. Στη συνέχεια αναφερθήκαμε στις διάφορες αρχιτεκτονικές που υπάρχουνε για την διαχείριση ενός δικτύου οι οποίες αναφορικά είναι η κεντρική, η ιεραρχική, η κατανεμημένη και το δικτυωμένο NMS που αποτελεί ένα συνδυασμό της ιεραρχικής και της κατανεμημένης αρχιτεκτονικής. Τέλος αναφερθήκαμε στις απαιτήσεις που έχουμε από ένα σύστημα διαχείρισης δικτύου οι οποίες έχουνε οριστεί από τον οργανισμό διεθνών προτύπων ISO κατά το πρότυπο FCAPS (Fault, Configuration, Accounting, Performance, Security) και έγινε μια επεξήγηση για την κάθε μία περιοχή του μοντέλου.

ΚΕΦΑΛΑΙΟ 2

SNMP

«Στα σημερινά πολύπλοκα δίκτυα γεμάτα από router, switches και servers μοιάζει πολύ κουραστική η διαχείριση όλων αυτών των συσκευών, ώστε να γνωρίζουμε όχι μόνο ότι βρίσκονται σε λειτουργία, αλλά ότι επιτελούν την εργασία τους με τον καλύτερο τρόπο…»

2.1. - Τι είναι το SNMP

Το SNMP (Simple Network Management Protocol) είναι ένα διαδικτυακό πρότυπο για την διαχείριση συσκευών σε δίκτυα βασισμένα στο πρωτόκολλο IP. Συσκευές που υποστηρίζουν το SNMP είναι συνήθως δρομολογητές, κατανεμητές, εξυπηρετητές, σταθμοί εργασίας, εκτυπωτές και άλλες. Το SNMP χρησιμοποιείται από συστήματα διαχείρισης δικτύου για την παρακολούθηση της κατάστασης των συσκευών που είναι συνδεμένες σε ένα δίκτυο και είναι μέρος της Internet Protocol Suite όπως έχει οριστεί από την Internet Engineering Task Force (IETF). Επίσης, αποτελεί μια συλλογή από πρότυπα για την διαχείριση του δικτύου, εμπεριέχει ένα πρωτόκολλο επιπέδου εφαρμογής, ένα σχήμα βάσης δεδομένων και μια συλλογή από αντικείμενα που περιέχουν την πληροφορία.

2.2. - Βασικές έννοιες του SNMP

Σε μια βασική χρήση του SNMP, έχουμε ένα ή και περισσότερους υπολογιστές που ονομάζονται διαχειριστές (managers) οι οποίοι έχουνε σαν σκοπό να παρακολουθούνε ή να διαχειρίζονται μία ομάδα από συσκευές σε ένα δίκτυο. Κάθε συσκευή η οποία παρακολουθείται εκτελεί ένα λογισμικό το οποίο ονομάζεται αντιπρόσωπος (agent), που η βασική του διεργασία είναι να προωθεί πληροφορίες μέσω του πρωτοκόλλου SNMP στον διαχειριστή.

Βασικά οι SNMP agents προωθούν την διαχειρίσιμη πληροφορία στα συστήματα διαχείρισης σαν μεταβλητές. Έτσι το SNMP έχει την δυνατότητα να τροποποιεί και να ορίζει νέες ρυθμίσεις σε συσκευές απομακρυσμένα μέσω της αλλαγής αυτών

των μεταβλητών. Οι μεταβλητές οι οποίες είναι προσβάσιμες από το SNMP είναι οργανωμένες ιεραρχικά. Η ιεραρχία καθώς και άλλα μεταδεδομένα (όπως ο τύπος και η περιγραφή μιας μεταβλητής) περιγράφονται από τις βάσεις διαχείρισης της πληροφορίας (MIBs).

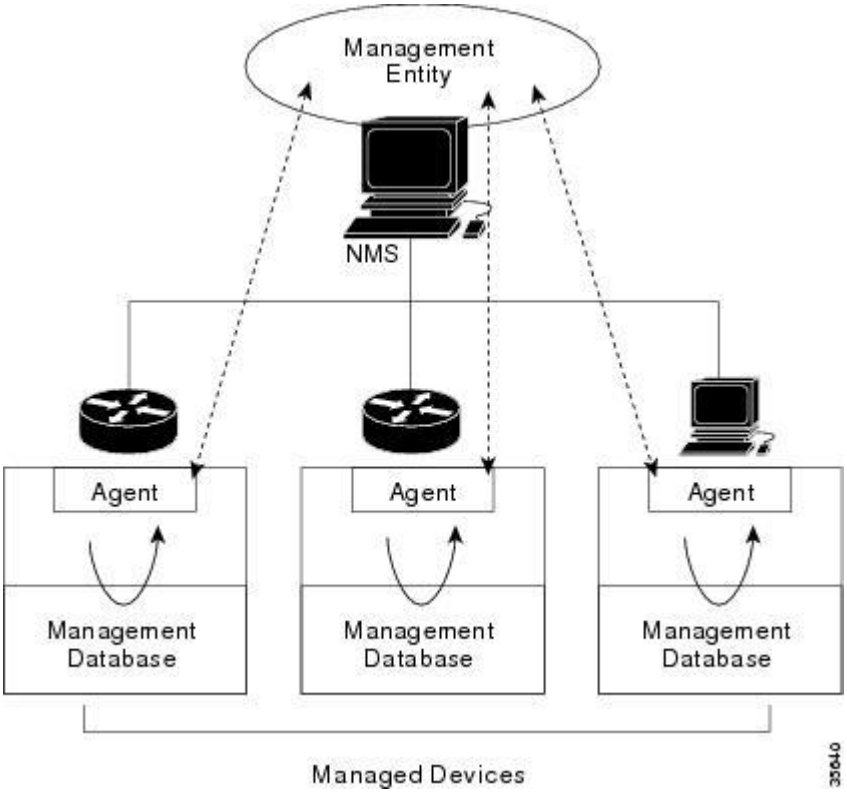
Ένα δίκτυο το οποίο διαχειρίζεται μέσω του SNMP αποτελείται από τρία βασικά συστατικά:

- Την διαχειρίσιμη συσκευή
- Agent – λογισμικό που εκτελείται από τις διαχειρίσιμες συσκευές
- Σύστημα Διαχείρισης Δικτύου (NMS) – λογισμικό που εκτελείται στον διαχειριστή.

Μια διαχειριζόμενη συσκευή είναι ένας κόμβος του δικτύου ο οποίος υλοποιεί το SNMP και επιτρέπει μονοκατευθυντική (read-only) ή διπλής κατεύθυνσης (read-write) πρόσβαση στις πληροφορίες του κόμβου. Οι διαχειριζόμενες συσκευές ανταλλάσουν πληροφορίες με το NMS και πολλές φορές ονομάζονται στοιχεία δικτύου και μπορεί να είναι διάφορα είδη συσκευών όπως δρομολογητές, εξυπηρετητές, κατανεμητές, γέφυρες, τηλέφωνα IP, κάμερες IP, εκτυπωτές και σταθμοί εργασίας.

Ένας agent είναι ένα λογισμικό διαχείρισης το οποίο εκτελείται στην διαχειριζόμενη συσκευή. Γνωρίζει την διαχειρίσιμη πληροφορία και την μετατρέπει ώστε να χρησιμοποιηθεί από το SNMP.

Το σύστημα διαχείρισης δικτύου (NMS) εκτελεί διεργασίες που παρακολουθούνε και ελέγχουν τις διαχειρίσιμες συσκευές.



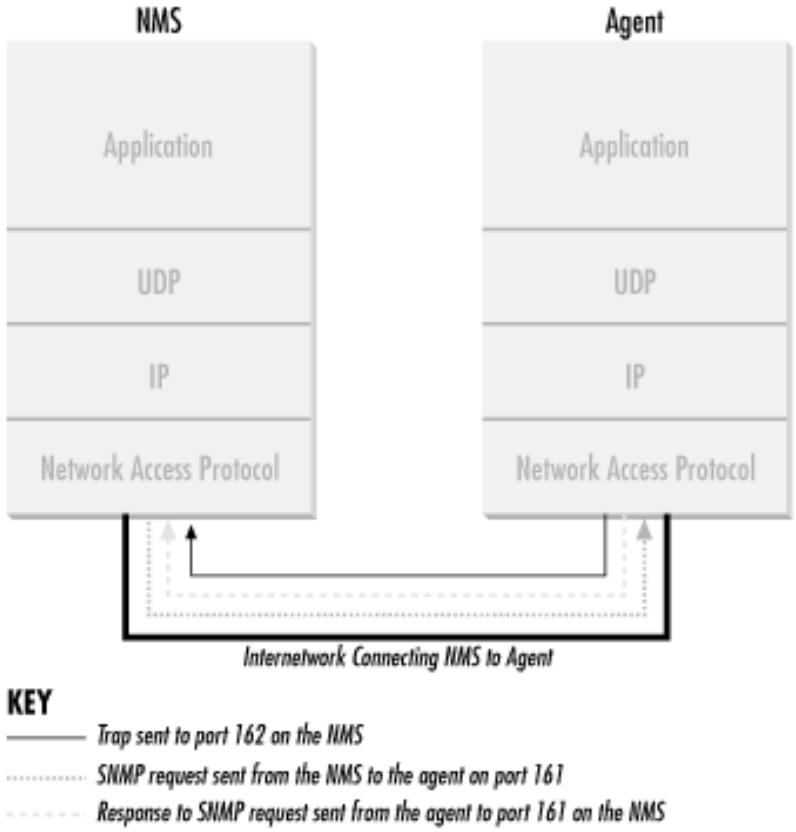
Σχήμα 2.1 – Βασική σχηματοποίηση SNMP

2.3. - SNMP and UDP

Το SNMP χρησιμοποιεί το UDP ως πρωτόκολλο μεταφοράς μεταξύ των Managers και των agents. Το UDP είναι μη αξιόπιστο πρωτόκολλο από την στιγμή που δεν υπάρχει επιβεβαίωση λήψης ενός πακέτου. Για αυτό το λόγο είναι στο έλεγχο της SNMP εφαρμογής να αναμεταδώσει πακέτα τα οποία χάθηκαν. Αυτό συνήθως υλοποιείται από ένα απλό χρονομετρητή. Το NMS στέλνει μια αίτηση σε έναν agent και περιμένει μια απάντηση. Εάν δεν πάρει μια απάντηση στο διάστημα του χρονομετρητή τότε υποθέτει ότι το πακέτο χάθηκε και επαναλαμβάνει την αίτηση. Η τιμή του χρονομετρητή καθώς και ο αριθμός των επαναλαμβανόμενων αιτήσεων μπορεί να ρυθμιστεί από τον διαχειριστή του NMS. Για τα traps τα πράγματα είναι διαφορετικά. Εάν ένας agent στείλει ένα trap και αυτό δεν καταφθάσει ποτέ δεν υπάρχει τρόπος το NMS να γνωρίζει ότι στάλθηκε. Ο agent δεν γνωρίζει ότι πρέπει να το ξαναστείλει. Το πλεονέκτημα του UDP είναι ότι έχει χαμηλό overhead οπότε η απόδοση του δικτύου δεν επηρεάζεται πολύ.

Το SNMP χρησιμοποιεί την πόρτα 161 του UDP για να στέλνει και να λαμβάνει αιτήσεις και την πόρτα 162 για να λαμβάνει traps από τις διαχειρίσιμες συσκευές.

Το σχήμα 2.2 αναπαριστά την σουίτα πρωτοκόλλων TCP/IP η οποία είναι η βάση για όλες τις επικοινωνίες TCP/IP. Κάθε συσκευή η οποία επιθυμεί να επικοινωνήσει στο Internet πρέπει να χρησιμοποιήσει αυτή τη σουίτα. Αυτό το μοντέλο συνήθως αναφέρεται και σαν στοίβα πρωτοκόλλων γιατί κάθε επίπεδο χρησιμοποιεί την πληροφορία από το αμέσως από κάτω του και παρέχει μια υπηρεσία στο ακριβώς από πάνω του.



Σχήμα 2.2 – Μοντέλο επικοινωνίας TCP/IP και SNMP

Όταν το NMS ή κάποιος agent επιθυμεί να εκτελέσει μια λειτουργία του SNMP τα ακόλουθα γεγονότα συμβαίνουν στην στοίβα πρωτοκόλλων:

Application

Αρχικά η SNMP εφαρμογή επιλέγει τι θέλει να κάνει κάθε φορά. Για παράδειγμα, μπορεί να στείλει μια SNMP αίτηση σε ένα agent, να στείλει μια απάντηση σε μία αίτηση ή να στείλει ένα trap σε ένα NMS. Αυτό το επίπεδο παρέχει πληροφορίες στον τελικό χρήστη.

UDP

Το επόμενο επίπεδο επιτρέπει την επικοινωνία μεταξύ δύο σταθμών. Η κεφαλίδα του UDP περιέχει μεταξύ άλλων την πόρτα προορισμού της συσκευής, στην οποία στέλνεται το αίτημα ή το trap. Η πόρτα προορισμού θα είναι είτε 161 ή 162.

IP

Το επίπεδο IP προσπαθεί να παραδώσει ένα SNMP πακέτο στο προορισμό του που ορίζεται από την διεύθυνση IP.

Medium Access Control (MAC)

Το τελευταίο γεγονός που πρέπει να συμβεί ώστε ένα SNMP πακέτο να φτάσει στο προορισμό του είναι να διαχειριστεί από το φυσικό δίκτυο. Το επίπεδο MAC απαρτίζεται από το υλικό και τους οδηγούς των συσκευών οι οποίοι οδηγούν την πληροφορία μέσα από ένα φυσικό κομμάτι καλωδίου, όπως είναι μία κάρτα δικτύου Ethernet.

2.4. - SNMP Communities

Το SNMPv1 και SNMPv2 χρησιμοποιούν την ιδέα των communities για να καθιερώσουν την εμπιστοσύνη μεταξύ των managers και των agents. Ένας agent ρυθμίζεται με τρία ονόματα community: read-only, read-write και trap. Τα ονόματα community είναι ουσιαστικά συνθηματικά και δεν έχουν διαφορά από ένα συνθηματικό που χρησιμοποιούμε για να προσπελάσουμε ένα λογαριασμό σε έναν υπολογιστή. Οι τρεις συμβολοσειρές community ελέγχουν τρεις διαφορετικές δραστηριότητες. Όπως και το όνομά τους υποδηλώνει, χρησιμοποιώντας το read-only community μπορούμε μόνο να διαβάσουμε και όχι να τροποποιήσουμε τιμές. Χρησιμοποιώντας το read-write community μπορούμε να διαβάσουμε και να τροποποιήσουμε τιμές. Το trap community επιτρέπει να λαμβάνουμε traps από έναν agent.

Οι περισσότεροι κατασκευαστές αποστέλλουν τον εξοπλισμό τους με προκαθορισμένες συμβολοσειρές community τα οποία είναι συνήθως public για το read-only community και private για το read-write community.

2.5. - Δομικά στοιχεία της αρχιτεκτονικής του SNMP

Μία οντότητα SNMP είναι μία συσκευή που υλοποιεί αυτήν την δυνατότητα (SNMP). Κάθε οντότητα SNMP αποτελείται από την μηχανή SNMP που υλοποιεί, και μία ή περισσότερες συνδεδεμένες με αυτήν εφαρμογή.

2.5.1. - Μηχανή SNMP

Η μηχανή SNMP παρέχει τις απαραίτητες λειτουργίες αποστολής και λήψης μηνυμάτων SNMP, αυθεντικοποίησης και κρυπτογράφησής τους και ελέγχει την πρόσβαση στις διαχειριζόμενες συσκευές. Υπάρχει μια αντιστοιχία ένα προς ένα, μεταξύ της μηχανής SNMP και της συσκευής που την περιέχει. Έτσι κάθε συσκευή SNMP έχει ένα μοναδικό αναγνωριστικό που την ταυτοποιεί σε ένα περιβάλλον πολλαπλών όμοιων συσκευών. Η μηχανή SNMP αποτελείται από:

- Τον αποστολέα (dispatcher)
- Το υποσύστημα επεξεργασίας μηνυμάτων
- Το υποσύστημα ασφαλείας
- Το υποσύστημα ελέγχου πρόσβασης

2.5.2. - Εφαρμογές SNMP

Υπάρχουν διάφοροι τύποι εφαρμογών SNMP, οι οποίες περιλαμβάνουν:

- Γεννήτριες εντολών (command generator), οι οποίες παρακολουθούν και διαχειρίζονται την παραγόμενη πληροφορία
- Γεννήτριες απόκρισης (command responder), παρέχουν πρόσβαση στην πληροφορία
- Γεννήτριες ειδοποιήσεων (notification originator), δημιουργούν ασύγχρονα μηνύματα
- Εφαρμογές διαμεταγωγής (proxy), μεταφέρουν μηνύματα μεταξύ οντοτήτων (RFC 3413,2002)

2.6. - Βάση Πληροφοριών Διαχείρισης (Management information base MIB)

Η βάση πληροφοριών διαχείρισης είναι μια ιδεατή βάση δεδομένων που χρησιμοποιείται ώστε να διαχειριζόμαστε τις οντότητες σε ένα δίκτυο δεδομένων. Τα αντικείμενα μέσα σε μια MIB είναι ορισμένα χρησιμοποιώντας ένα υποσύνολο

του Abstract Syntax Notation One (ASN.1) το οποίο ονομάζετε Structure of Management Information Version 2 (SMIv2). Το λογισμικό το οποίο αναλύει τα αντικείμενα που ορίζονται μέσω του SMIv2 είναι ένας διερμηνευτής MIB.

Η βάση πληροφοριών διαχείρισης είναι ιεραρχική (δομημένη σαν δέντρο) και οι οντότητες διακρίνονται μέσα από αναγνωριστικά αντικειμένων (Object Identifiers, OIDs).

2.6.1. - Abstract Syntax Notation One (ASN.1)

Στις τηλεπικοινωνίες και στα δίκτυα υπολογιστών, το Abstract Syntax Notation One (ASN.1) είναι ένα πρότυπο και ένα εύκαμπτο σύστημα χαρακτήρων και συμβόλων το οποίο περιγράφει δομές δεδομένων για την παρουσίαση, κωδικοποίηση, μετάδοση και αποκωδικοποίηση πληροφορίας. Παρέχει ένα σύνολο από καθορισμένους κανόνες για την περιγραφή της δομής των αντικειμένων, το οποίο είναι ανεξάρτητο από την αρχιτεκτονική των συστημάτων και τις διαφορετικές κωδικοποιήσεις τους.

Το ASN.1 είναι ένα πρότυπο που αναπτύχθηκε από κοινού από τον ISO και τον ITU-T, αρχικά ορισμένο το 1984 ως μέρος του CCITT X.409:1984. Το 1988 λόγω της μεγάλης απήχησης του, ορίστηκε σαν ξεχωριστό πρότυπο το X.208 και το 1995 δημιουργήθηκε μια ανανεωμένη έκδοση η οποία καλύπτεται από το πρότυπο X.680.

Ένα προσαρμοσμένο υποσύνολο του ASN.1 το Structure of Management Information Version 2 (SMIv2), είναι ορισμένο μέσα στο SNMP για να ορίζει σύνολα από συσχετιζόμενα αντικείμενα. Τα σύνολα αυτά ονομάζονται και MIB modules.

2.6.2. - Δομή Πληροφορίας (Structure of Management Information, SMI)

Η πληροφορία, όπως ειπώθηκε, μπορεί να αναπαρασταθεί ως συλλογή από διαχειρίσιμα αντικείμενα της ιδεατής βάσης (MIB). Οι συλλογές ομοειδών αντικειμένων της βάσης ομαδοποιούνται σε υπομονάδες ακολουθώντας την δενδρική διάταξη και την αυστηρά καθορισμένη αναπαράσταση της πληροφορίας όπως περιγράφεται λεπτομερώς στο πρότυπο ASN.1 (ITU-T, 2002). Η δομή της

πληροφορίας μπορεί να χωριστεί σε 3 βασικές κατηγορίες: module definitions, object definitions και notification definitions (RFC 3410, 2002).

Τα **Module definitions** χρησιμοποιούνται για να περιγράψουν τις υπομονάδες που παράγουν πληροφορία και χρησιμοποιούν το MODULE-IDENTITY του ASN.1 για να περιγράψουν την παραγόμενη πληροφορία

Τα **Object definitions** χρησιμοποιούνται για να περιγράψουν διαχειρίσιμες οντότητες-αντικείμενα. Το OBJECT-TYPE του ASN.1 χρησιμοποιείται για να περιγράψει την παραγόμενη πληροφορία

Τα **Notification definitions** χρησιμοποιούνται για να περιγράψουν την παραγωγή πληροφορίας άνευ προηγούμενης ζήτησης. Το NOTIFICATION-TYPE του ASN.1 χρησιμοποιείται για να περιγράψει την παραγόμενη πληροφορία (RFC 3410, 2002)

2.6.3. - Περιγραφή της πληροφορίας βάσης SMI

Οι κατηγορίες των δεδομένων που παράγει μια MIB βάση είναι:

Πίνακας 2-1: Τύποι Δεδομένων MIB

Integer32	**Gauge32**	**TimeTicks**	**STRING**	**Opaque**
Enumerated integers	Counter32	INTEGER	Unsigned32	BITS
OBJECT IDENTIFIER	Counter64	OCTET	IpAddress	

Επίσης ως δομές δεδομένων περιγράφονται και οι εξής:

IMPORTS: επιτρέπει την περιγραφή αντικειμένων μίας βάσης MIB Module, μέσα σε μία άλλη υπομονάδα MIB module.

MODULE-IDENTITY: Περιγράφει την ταυτότητα και τα χαρακτηριστικά μίας τέτοιας μονάδας

OBJECT-IDENTITY & OID: Η λεπτομερής αριθμητική δενδροειδής αναπαράσταση της συγκεκριμένης παραμέτρου της υπομονάδας.

OBJECT-TYPE: Περιγράφει το είδος των παραγόμενων δεδομένων, την κατάσταση της συσκευής και την δυνητικά παραγόμενη πληροφορία.

SEQUENCE type assignment: Περιγράφει τα διαδοχικά στοιχεία ενός παραγόμενου πίνακα

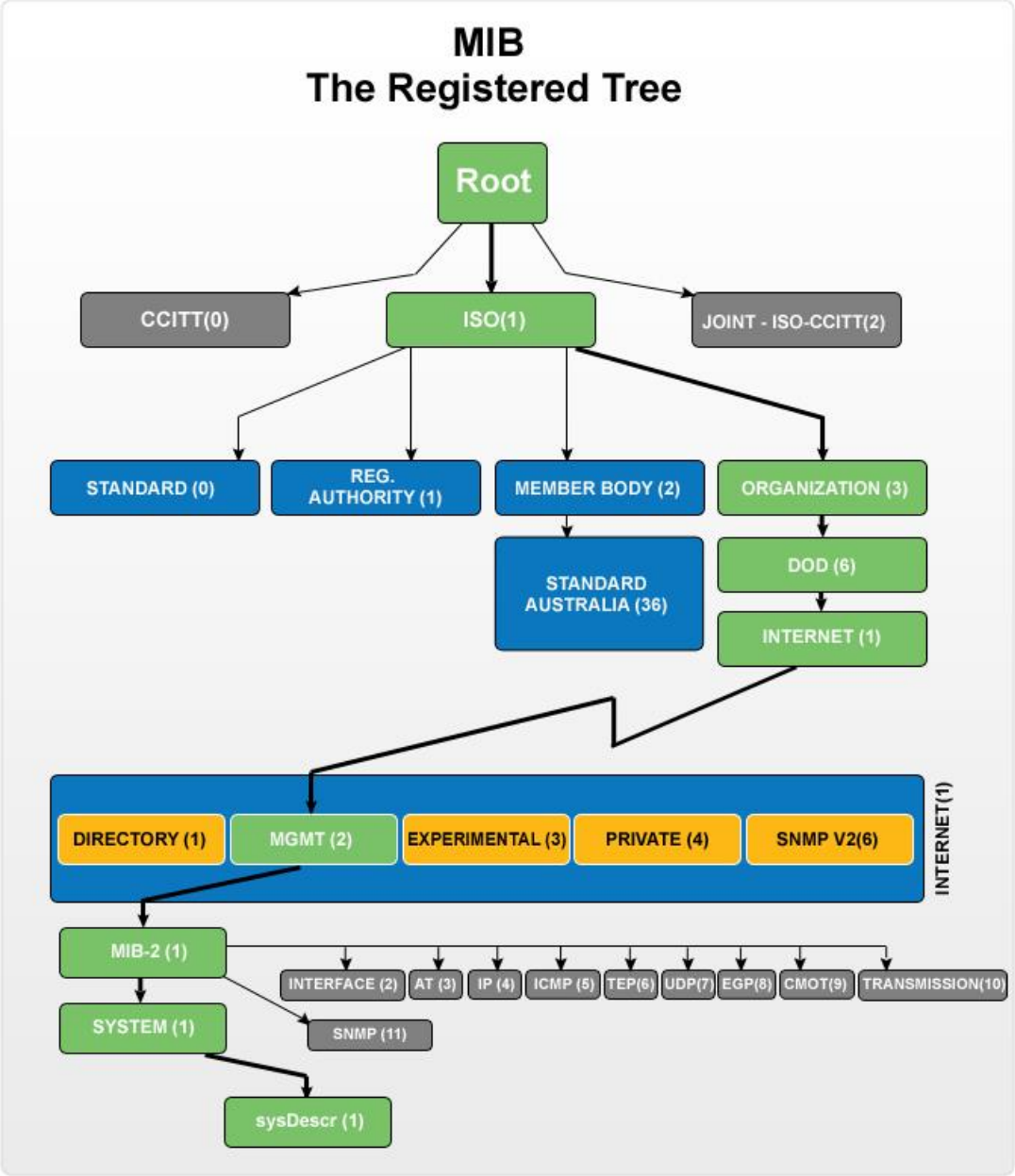
NOTIFICATION-TYPE: Περιγράφει το είδος της παραγόμενης ειδοποίησης (RFC 3410, 2002)

2.6.4. - Αρχικοποίηση αντικειμένων στο MIB

Κάθε τύπος αντικειμένου (object type) έχει ένα όνομα, μία λογική σύνταξη και μία κωδικοποίηση. Το όνομα ορίζεται μοναδικά από το OBJECT IDENTIFIER και η λογική σύνταξη του αντικειμένου καθορίζεται από τον τύπο του αντικειμένου. Π.χ. για κάποιο δεδομένο αντικείμενο μπορεί να είναι ακέραιος (INTEGER) ή οκταδική συμβολοσειρά (OCTET STRING). Η κωδικοποίηση είναι απλώς ο κωδικοποιημένος τρόπος αναπαράστασης του συγκεκριμένου αντικειμένου σύμφωνα με το προαναφερθέν ASN.1.

2.6.5. - Αναγνωριστικά Αντικειμένων (OBJECT IDENTIFIER)

Το όνομα OBJECT IDENTIFIER χρησιμοποιείται για να αναγνωρίσει μοναδικά ένα αντικείμενο ή και ολόκληρη οικογένεια αντικειμένων. Σε κάθε περίπτωση αναγνωρίζει μοναδικά, είτε μία συσκευή δικτύου όπως π.χ. ένα δρομολογητή, είτε μία κατηγορία π.χ. εγγράφων. Το όνομα είναι μία αλληλουχία ακεραίων αριθμών οι οποίοι διασχίζουν ένα μοναδικό καθολικό δέντρο. Το δέντρο ξεκινά από την ρίζα που συνδέεται με κόμβους. Ο κάθε κόμβος με την σειρά του συνδέεται με άλλους κόμβους-παιδιά. Οι κόμβοι αυτοί ονομάζονται παιδιά του προηγούμενου. Το βάθος του δέντρου δεν είναι καθορισμένο και μπορεί να συνεχίσει για όσο χρειαστεί. Ο κάθε κόμβος ονομάζεται με ένα όνομα αντιπροσωπευτικό της λειτουργίας του.



Σχήμα 2.3 – Αναπαράσταση MIB δέντρου

Η ρίζα του δέντρου δεν έχει όνομα, αλλά έχει τουλάχιστον 3 παιδιά:

- Το 1ο κόμβο τον διαχειρίζεται ο διεθνής οργανισμός προτυποποιήσεως (International Organization for Standardization)και ονομάζεται iso(1).
- Ο 2ος κόμβος διαχειρίζεται από την διεθνή επιτροπή τηλέγραφου και τηλεφώνου (International Telegraph and Telephone Consultative Committee) με το όνομα ccitt(0)
- Ο 3ος κόμβος είναι προϊόν συνδιαχείρισης των δύο προηγούμενων (ISO και CCITT) με το όνομα joint-iso-ccitt(2).

Κάτω από τον κόμβο iso(1) ο ISO έχει ορίσει ένα υποδέντρο για χρήση άλλων διεθνών οργανισμών με την ονομασία org(3). Δύο από τα παιδιά αυτού του κόμβου έχουν ανατεθεί στο Αμερικανικό Ινστιτούτο τεχνολογίας και προτύπων (U.S. National Institutes of Standards and Technology – NIST). Ένα από αυτά τα υποδέντρα έχει ανατεθεί από την NIST στο υπουργείο Άμυνας της Αμερικής ως dod(6). Αυτό με την σειρά του έχει ορίσει ένα υποδέντρο για το Διαδίκτυο, ώστε να διαχειρίζεται από την αρμόδια επιτροπή (Internet Activities Board - IAB) σύμφωνα με το παρακάτω ορισμό: internet OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }

Για τον παραπάνω λόγο, κάθε δέντρο που ασχολείται με την ονομασία οντοτήτων – αντικειμένων ξεκινά με το αναγκαστικό πρόθεμα:
1.3.6.1.xxxxx

Ακολουθώντας την κωδικοποίηση της IAB , το υποδέντρο κάτω από το κλειδί internet, έχει με την σειρά του 4 κόμβους ως εξής:
directory OBJECT IDENTIFIER ::= { internet 1 }
mgmt OBJECT IDENTIFIER ::= { internet 2 }
experimental OBJECT IDENTIFIER ::= { internet 3 }
private OBJECT IDENTIFIER ::= { internet 4 }

directory(1)
Είναι δεσμευμένο για μελλοντικές χρήσεις σε σχέση με το μοντέλο αναφοράς OSI

Mgmt(2)
Χρησιμοποιείται για αντικείμενα τα οποία ορίζονται από την IAB για την IANA (Internet Assigned Numbers Authority). Εδώ δηλαδή ορίζονται οι αριθμοί για καινούργια αντικείμενα που ορίζονται μέσω των RFC (Request for Comments).

Experimental(3)
Χρησιμοποιείται για πειραματικούς σκοπούς και έχει ανατεθεί επίσης στην IANA.

Private(4)

Χρησιμοποιείται για αντικείμενα που ορίζονται μονομερώς, ορίζονται δηλαδή από τον κατασκευαστή τους. Και αυτό το υποδέντρο έχει ανατεθεί στην IANA, όσο αφορά τον πρώτο κόμβο του δηλαδή τον

enterprises OBJECT IDENTIFIER ::= { private 1 }

Αυτό το υποδέντρο χρησιμοποιείται από τους κατασκευαστές και τις επιχειρήσεις για να περιγράψουν τα προϊόντα τους. Μόλις μια επιχείρηση αναλάβει ένα υποδέντρο, μπορεί μέσα σε αυτό όλες τις συσκευές που επιθυμεί, σε όποιο βάθος κόμβου.

2.6.6. - Τύποι δεδομένων - Primitive Types

Μόνο οι τύποι δεδομένων που περιγράφονται στο ASN.1 (Application fields of ASN.1, 2005) ως primitive types INTEGER, OCTET STRING, OBJECT IDENTIFIER, & NULL γίνονται δεκτοί.

Κατασκευαστές - Constructor Types

Επιτρέπεται η χρήση του κατασκευαστή (constructor) όπως ορίζεται στο ASN.1 με την προϋπόθεση ότι δημιουργεί (κατασκευάζει) λίστες ή πίνακες σύμφωνα με τις παρακάτω συντάξεις:

Λίστες SEQUENCE { <type1>, ..., <typeN> }

όπου <type1>, είναι ένας από τους τύπους δεδομένων που αναφέρονται παραπάνω.

Πίνακες SEQUENCE OF <entry> όπου το <entry> αντιστοιχεί σε έναν «κατασκευαστή» λίστας της προηγούμενης παραγράφου

2.6.7. - Βασικοί τύποι - Defined Types

NetworkAddress: Μία διεύθυνση από πιθανά πρωτόκολλα. Προς το παρόν υποστηρίζεται μόνο το πρωτόκολλο Internet.

IpAddress: Μήκους 32-bit διεύθυνση internet

Counter: Μη αρνητικός ακέραιος ο οποίος μονότονα αυξάνει μέχρι μία προκαθορισμένη μέγιστη τιμή και τότε μηδενίζει και ξανά-αρχίζει να αυξάνεται. Μέγιστη τιμή του είναι η 2^32-1.

Gauge Μη αρνητικός ακέραιος ο οποίος δύναται να αυξομειώνεται διαθέτοντας όμως μία μέγιστη τιμή. Αν όχι άλλως ορισμένη, τότε η μέγιστη τιμή του είναι επίσης 2^32-1.

TimeTicks: Μη αρνητικός ακέραιος ο οποίος μετράει τον χρόνο σε εκατοστά του δευτερολέπτου ξεκινώντας από κάποια ημερομηνία.

Opaque: Χρησιμοποιείται για να μπορέσει να συνταχθεί ένας τύπος αυθαίρετων δεδομένων. Τα δεδομένα χωρίζονται σε octets και μπορούν να χρησιμοποιηθούν αυτούσια από την εφαρμογή που τα παραλαμβάνει-υποδέχεται.

2.6.8. - Διαχειρίσιμα αντικείμενα - Managed Objects

Μία περιγραφή ενός αντικειμένου - συσκευής που μπορεί να ανταλλάξει SNMP πληροφορία, αποτελείται από 5 τμήματα:

OBJECT: Ένα κείμενο το ονομαζόμενο the OBJECT DESCRIPTOR, ακολουθούμενο από το OBJECT IDENTIFIER.
Syntax: Ο ορισμός του αντικειμένου που πρέπει να αντιστοιχεί σε ένα στιγμιότυπο του δέντρου των MIB όπως περιγράφεται παρακάτω
Definition: Μία λεκτική αναπαράσταση του αντικειμένου υπό περιγραφή. Θα πρέπει να είναι μοναδική ώστε να έχει νόημα σε οποιαδήποτε άλλη μηχανή υλοποιεί το πρωτόκολλο.

Access: Μπορεί να πάρει τις εξής τιμές: read-only, read-write, write-only, or not-accessible.
Status: Μπορεί να πάρει τις εξής τιμές: mandatory, optional, or obsolete.

2.7. - Λειτουργίες του SNMP

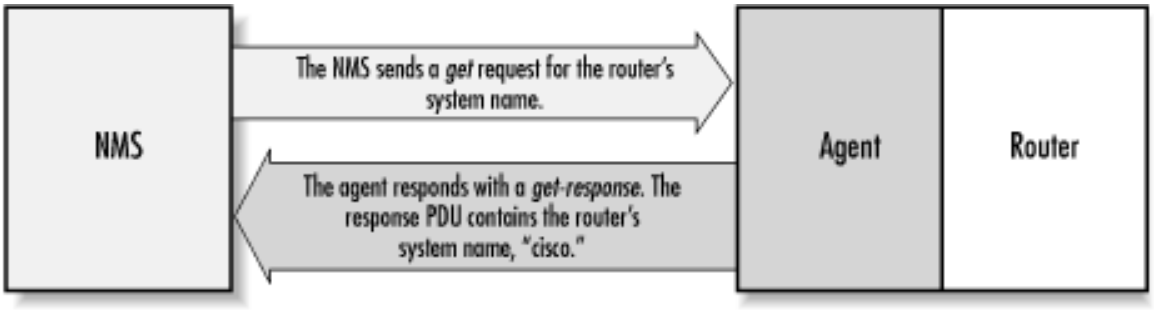
Παραπάνω αναφερθήκαμε πως το SNMP οργανώνει την πληροφορία. Σε αυτή την ενότητα θα αναφερθούμε για το πώς το SNMP συλλέγει όλη αυτή την πληροφορία. Το *Protocol Data Unit* (PDU) είναι ένα μήνυμα όπου οι managers και οι agents στέλνουν και λαμβάνουν πληροφορίες μέσω αυτού. Υπάρχει ένα συγκεκριμένο μήνυμα PDU για κάθε μία από τις παρακάτω λειτουργίες του SNMP.

- get
- get-next
- get-bulk (SNMPv2 and SNMPv3)
- set
- get-response
- trap
- notification (SNMPv2 and SNMPv3)
- inform (SNMPv2 and SNMPv3)
- report (SNMPv2 and SNMPv3)

Στις επόμενες ενότητες θα αναφέρουμε συνοπτικά την κάθε μία.

2.7.1. - Η λειτουργία get

Η λειτουργία get αρχικοποιείται από το NMS, η οποία στέλνει μια αίτηση στον agent. Ο agent λαμβάνει την αίτηση και την επεξεργάζεται. Κάποιοι agent οι οποίοι είναι πολύ απασχολημένοι και δεν είναι ικανοί να απαντήσουν στην αίτηση την απορρίπτουν. Εάν ο agent επιτυχημένα συλλέξει την πληροφορία που αιτήθηκε στέλνει πίσω στο NMS ένα get-response. Η λειτουργία παρουσιάζετε στο σχήμα 2.3.

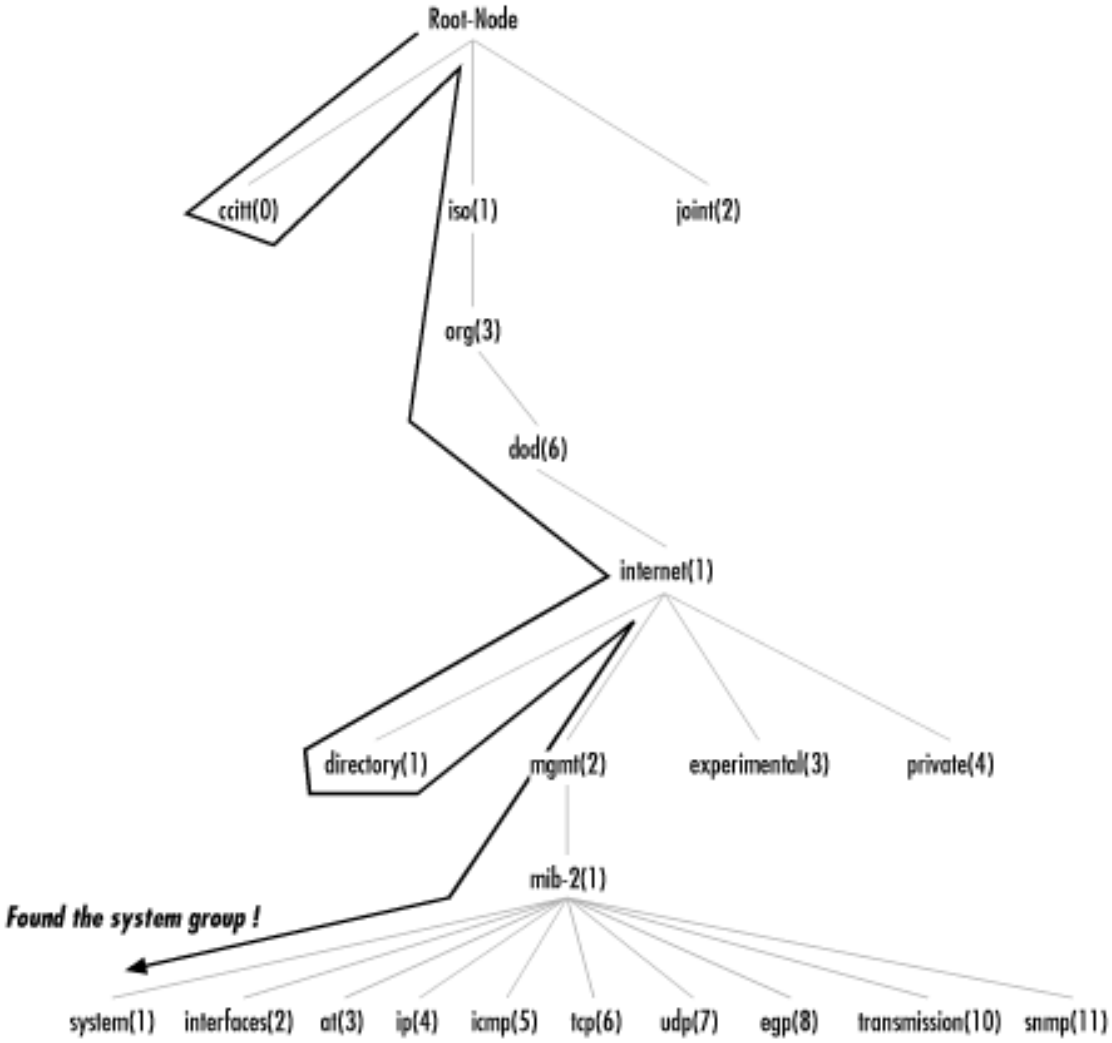


Σχήμα 2.3 - Λειτουργία get-request

Η λειτουργία get είναι χρήσιμη όταν θέλουμε να ανακτήσουμε ένα MIB αντικείμενο κάθε φορά. Η προσπάθεια να συλλέγουμε όλες τις πληροφορίες με αυτό τον τρόπο μας οδηγεί στο να χάνουμε χρόνο. Εκεί έρχεται η εντολή get-next που μας επιτρέπει να συλλέξουμε πάνω από ένα αντικείμενο την φορά.

2.7.2. - Η λειτουργία get-next

Η λειτουργία get-next μας επιτρέπει να θέσουμε μία ακολουθία από εντολές ώστε να συλλέξουμε μια ομάδα από τιμές από ένα MIB. Για κάθε MIB αντικείμενο που θέλουμε να ανακτήσουμε, ένα ξεχωριστό αίτημα get-net και get-response δημιουργείτε. Η λειτουργία get-next διασχίζει ένα υποδέντρο με λεξικογραφική σειρά. Από την στιγμή που ένα OID είναι μία ακολουθία από ακέραιους, είναι εύκολο για έναν agent να αρχίσει από την ρίζα του δέντρου των αντικειμένων και να κατεβαίνει μέχρι να φτάσει στο αντικείμενο που θέλει. Όταν το NMS λάβει μία απάντηση από τον agent για μια get-next αίτηση τότε θέτει την επόμενη get-next αίτηση. Συνεχίζει να το κάνει αυτό μέχρι ο agent να επιστρέψει ένα λάθος πράγμα που σηματοδοτεί ότι το MIB έφτασε στο τέλος του και δεν υπάρχουν άλλα αντικείμενα για να λάβουμε.

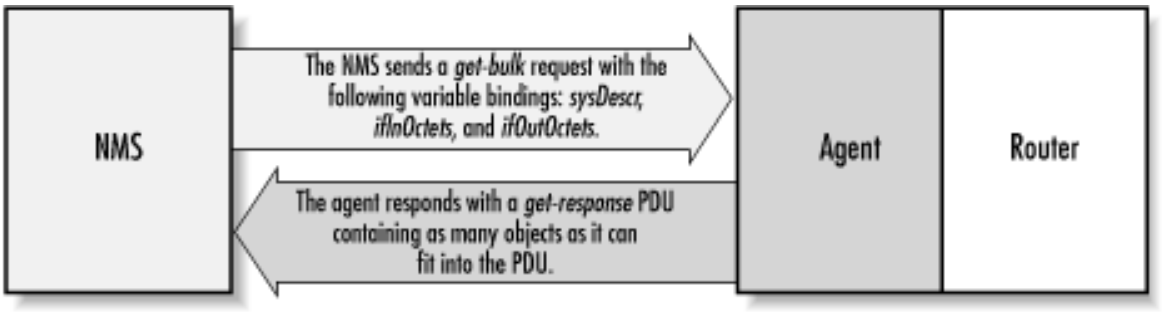


Σχήμα 2.4 - Λειτουργία get-next

2.7.3. - Η λειτουργία get-bulk

Το SNMPv2 ορίζει την λειτουργία get-bulk, η οποία επιτρέπει στον manager να ανακτήσει ένα μεγάλο μέρος ενός πίνακα μονομιάς. Η λειτουργία get μπορεί να επιχειρήσει να ανακτήσει περισσότερα από ένα αντικείμενα την φορά αλλά το μέγεθος των μηνυμάτων περιορίζεται από τις δυνατότητες του agent. Εάν ο agent δεν μπορεί να επιστρέψει όλες τις τιμές που του ζητήθηκαν επιστρέφει ένα μήνυμα λάθους χωρίς δεδομένα. Η λειτουργία get-bulk από την άλλη λέει στον agent να στείλει όση πληροφορία μπορεί. Αυτό σημαίνει ότι μη ολοκληρωμένες απαντήσεις είναι δυνατό να υπάρξουν. Υπάρχουν δύο πεδία που πρέπει να δηλωθούνε όταν θέτουμε την εντολή get-bulk: nonrepeaters και max-repetitions. Το nonrepeaters λέει στην get-bulk εντολή ότι τα πρώτα N αντικείμενα μπορούνε να ανακτηθούνε με την απλή λειτουργία get-next. Το max-repetitions λέει στην get-bulk εντολή να

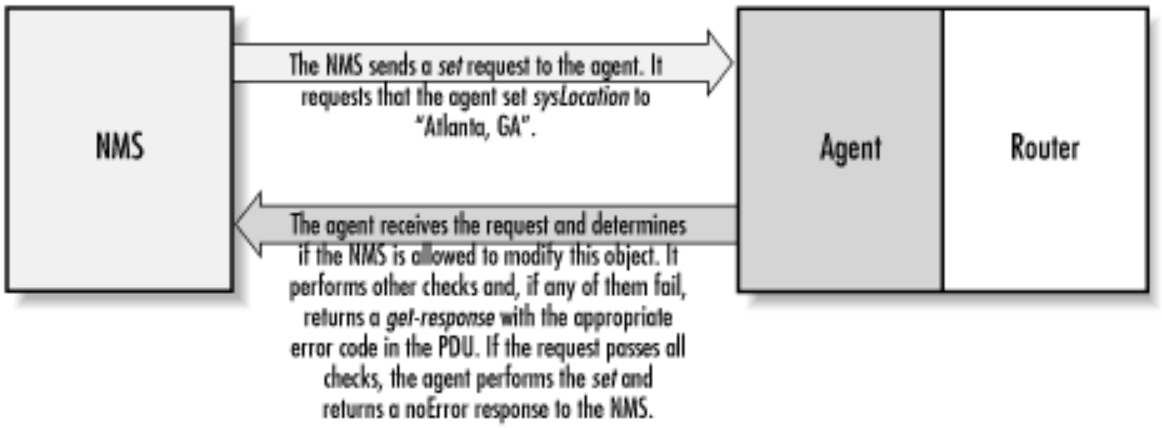
επιχειρήσει μέχρι M get-next λειτουργίες για να ανακτήσει τα υπόλοιπα αντικείμενα.



Σχήμα 2.5 - Λειτουργία get-bulk

2.7.4. - Η λειτουργία set

Η λειτουργία set χρησιμοποιείται για να αλλάξει την τιμή ενός διαχειρίσιμου αντικειμένου ή για να δημιουργήσει μία νέα γραμμή σε ένα πίνακα. Τα αντικείμενα τα οποία είναι ορισμένα σαν read-write ή write-only μπορούν να τροποποιηθούν η να δημιουργηθούν χρησιμοποιώντας αυτή την λειτουργία. Είναι δυνατόν ένα NMS να τροποποιήσει πάνω από ένα αντικείμενο την φορά.



Σχήμα 2.6 - Λειτουργία set

2.7.5. - get, get-next, get-bulk, και set Error Responses

Οι αποκρίσεις λαθών μας βοηθούν να καταλάβουμε εάν μια αίτηση get ή set εκτελέστηκε σωστά από ένα SNMP agent. Οι λειτουργίες get, get-next, και set μπορούν να επιστρέψουν τα μηνύματα λάθους που εμφανίζονται στο πίνακα 2-2. Η κατάσταση λάθους για το κάθε λάθος φαίνεται μέσα στις παρενθέσεις.

Πίνακας 2-2: SNMPv1 Μηνύματα λάθους

SNMPv1 Μήνυμα λάθους	**Περιγραφή**
noError(0)	Δεν υπήρχε κανένα πρόβλημα στην διαδικασία.
tooBig(1)	Η απόκριση της αίτησης ήταν πολύ μεγάλη για να χωρέσει σε ένα μήνυμα
noSuchName(2)	Ο agent ρωτήθηκε να εκτελέσει την λειτουργία get ή set σε ένα αντικείμενο που το OID του δεν υπάρχει
badValue(3)	Σε ένα αντικείμενο read-write ή write-only ορίστηκε μια ασυνεπής τιμή
readOnly(4)	Το λάθος αυτό δεν χρησιμοποιείται γενικώς. Το λάθος noSuchName είναι το ισότιμό του.
genErr(5)	Εάν κάποιο λάθος συμβεί που δεν ανήκει σε καμία από τα παραπάνω τότε παράγεται αυτό το λάθος.

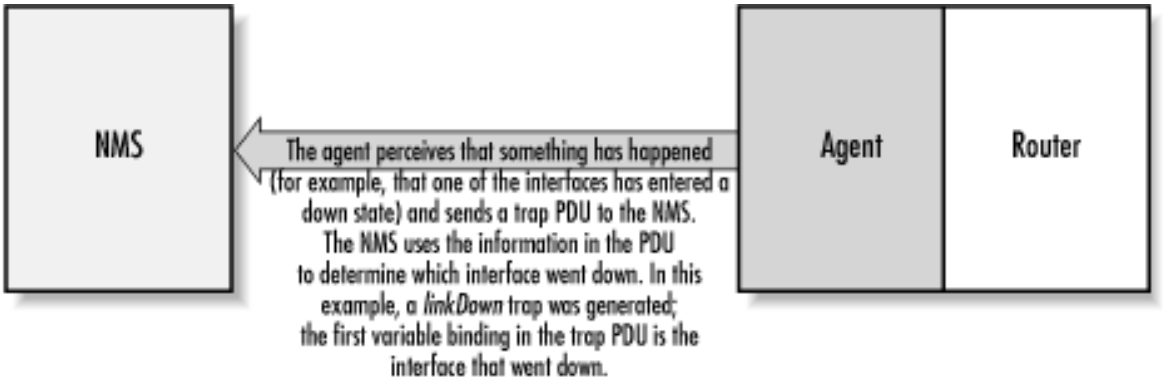
Τα μηνύματα λάθους του SNMPv1 δεν είναι και τόσο περιγραφικά. Στην προσπάθεια να διορθωθεί αυτό το πρόβλημα στο SNMPv2 προστέθηκαν κάποια μηνύματα λάθους τα οποία χρησιμοποιούνται στις λειτουργίες get, set, get-next και get-bulk, με την προϋπόθεση ότι και το NMS αλλά και ο agent υποστηρίζουν την δεύτερη έκδοση του SNMP. Τα μηνύματα παρουσιάζονται στον πίνακα 2-3.

Πίνακας 2-3: SNMPv2 Μηνύματα λάθους

SNMPv2 Μήνυμα λάθους	**Περιγραφή**
noAccess(6)	Συμβαίνει όταν πάμε να ορίσουμε τιμή σε μια μη προσβάσιμη μεταβλητή.
wrongType(7)	Σε ένα αντικείμενο ορίσαμε μια τιμή η οποία είναι διαφορετική από τον τύπο δεδομένων του.
wrongLength(8)	Σε ένα αντικείμενο ορίσαμε μια τιμή η οποία ξεπερνά το επιτρεπόμενο μέγεθος.
wrongEncoding(9)	Σε ένα αντικείμενο ορίσαμε μια τιμή η οποία έχει λάθος κωδικοποίηση.
wrongValue(10)	Θέσαμε μια τιμή στην μεταβλητή την οποία δεν καταλαβαίνει.
noCreation(11)	Προσπάθεια να θέσουμε τιμή σε μεταβλητή που δεν υπάρχει ή προσπάθεια δημιουργίας μεταβλητής η οποία δεν υπάρχει στο MIB.
inconsistentValue	Η μεταβλητή είναι σε ασυνεπή κατάσταση και δεν δέχεται την λειτουργία set.
resourceUnavailable(13)	Δεν υπάρχουν οι απαραίτητοι πόροι του συστήματος για να εκτελεστεί η λειτουργία set.
commitFailed(14)	Είναι γενικό λάθος για την αποτυχία της λειτουργίας set.
undoFailed(15)	Η λειτουργία set απέτυχε και ο agent δεν ήταν ικανός να επαναφέρει τις προηγούμενες τιμές μέχρι το σημείο της αποτυχίας.
authorizationError(16)	Μια εντολή του SNMP δεν μπορεί να αυθεντικοποιηθεί. Λάθος συμβολοσειρά community.
notWritable(17)	Η μεταβλητή δεν δέχεται αλλαγή στην τιμή της παρόλο που θα έπρεπε.
inconsistentName(18)	Η προσπάθεια να αλλάξουμε την τιμή μιας μεταβλητής απέτυχε γιατί η μεταβλητή βρίσκετε σε ασυνεπή κατάσταση.

2.7.5. - SNMP Traps

Ένα trap είναι ένας τρόπος ώστε ένας agent να ειδοποιήσει το NMS ότι κάτι κακό συνέβη. Το σχήμα 2.7 αναπαριστά την διαδικασία την δημιουργίας ενός trap.



Σχήμα 2.7 – Δημιουργία trap

Το trap ξεκινά από τον agent και κατευθύνεται προς έναν προορισμό που έχουμε εισάγει στον agent, ο οποίος είναι συνήθως η IP διεύθυνση του NMS. Δε στέλνεται κανένα μήνυμα από το NMS στον agent ώστε να αναγνωρίσει ότι έλαβε το trap, έτσι ο agent δεν μπορεί να ξέρει εάν το NMS έλαβε το trap. Από την στιγμή που το SNMP χρησιμοποιεί το UDP ως πρωτόκολλο μεταφοράς και επειδή τα traps χρησιμοποιούνται για να αναφέρουν προβλήματα στο δίκτυο, είναι αρκετά πιθανό κάποια traps να χαθούνε και να μη φτάσουν στο προορισμό τους. Ωστόσο, το γεγονός ότι κάποια traps θα χαθούνε δεν τα κάνει λιγότερο χρήσιμα και είναι αναπόσπαστο κομμάτι της διαχείρισης δικτύου. Είναι προτιμότερο για τον εξοπλισμό να προσπαθεί να μας ειδοποιήσει ότι κάτι δεν πάει καλά, έστω και αν το μήνυμα χαθεί παρά να μην γνωρίζουμε τίποτα.

Μερικές περιπτώσεις όπου ένα trap μπορεί να αναφέρει είναι:

- Μια διεπαφή δικτύου σε μια συσκευή απενεργοποιήθηκε.
- Μια διεπαφή δικτύου σε μια συσκευή ξαναενεργοποιήθηκε.
- Η εισερχόμενη κλήση σε ένα modem δεν μπόρεσε να δημιουργήσει μια σύνδεση.
- Ο ανεμιστήρας ενός κατανεμητή ή δρομολογητή έχει βλάβη.

Όταν ένα NMS λαμβάνει ένα trap πρέπει να ξέρει πώς να το διερμηνεύσει. Πρέπει να ξέρει τι σημαίνει το trap και πώς να διερμηνεύσει την πληροφορία που

κουβαλά. Το trap αρχικά αναγνωρίζεται από το generic trap number. Υπάρχουν επτά generic trap numbers(0-6) τα οποία παρουσιάζονται στον πίνακα 2-4. Το generic trap 6 είναι ένα γενικό trap για όλα τα enterprise-specific traps, τα οποία ορίζονται από κατασκευαστές ή χρήστες και δεν ανήκουν στις υπόλοιπες κατηγορίες. Τα Enterprise-specific traps αναγνωρίζονται επιπλέον από το enterprise ID και ένα συγκεκριμένο αριθμό trap, ο οποίος επιλέγεται από τον κατασκευαστή που όρισε το trap. Για παράδειγμα, όταν η εταιρία Cisco ορίζει ειδικά traps για τα ιδιωτικά της MIB, τα τοποθετεί στο συγκεκριμένο MIB υποδέντρο (*iso.org.dod.internet.private.enterprises. cisco).*

Πίνακας 2-4: Generic Traps

Γενικό όνομα και αριθμός trap	**Περιγραφή**
coldStart (0)	Υποδεικνύει ότι ένας agent έκανε επανεκκίνηση. Όλες οι διαχειρίσιμες μεταβλητές θα επαναφερθούν στην αρχική τους τιμή. Ειδικά Counters και Gauges θα γίνουνε 0.
warmStart (1)	Υποδεικνύει ότι ένας agent ξανααρχικοποίησε τον εαυτό του. Καμιά από τις διαχειρίσιμες μεταβλητές δε θα επαναφερθεί στην αρχική της τιμή.
linkDown (2)	Στέλνεται όταν μια διεπαφή δικτύου σε μια συσκευή απενεργοποιήθηκε.
linkup (3)	Στέλνεται όταν μια διεπαφή δικτύου σε μια συσκευή ξαναενεργοποιείτε
authenticationFailure (4)	Υποδεικνύει ότι κάποιος προσπάθησε να ρωτήσει τον agent με λάθος συμβολοσειρά community.
egpNeighborLoss (5)	Υποδεικνύει ότι ένας γείτονας που υποστηρίζει το Exterior Gateway Protocol (EGP) απενεργοποιήθηκε.
enterpriseSpecific (6)	Υποδεικνύει ότι το trap είναι enterprise-specific. Κατασκευαστές και χρήστες ορίζουν το δικά τους traps κάτω από το κλαδί private-enterprise στο SMI δέντρο.

2.7.6. - SNMP Notification

Σε μια προσπάθεια να τυποποιηθεί η μορφή του PDU των SNMPv1 traps, το SNMPv2 ορίζει το NOTIFICATION-TYPE. Η μορφή του PDU για το

NOTIFICATION-TYPE είναι ίδια και για την λειτουργία get και για την set. Το RFC 2863 επαναπροσδιορίσει το *linkDown* generic notification type ως εξής.

```
linkDown NOTIFICATION-TYPE
   OBJECTS { ifIndex, ifAdminStatus, ifOperStatus }
   STATUS  current
   DESCRIPTION
      "A linkDown trap signifies that the SNMPv2 entity, acting in an
       agent role, has detected that the ifOperStatus object for one
       of its communication links left the down state and transitioned
       into some other state (but not into the notPresent state). This
       other state is indicated by the included value of ifOperStatus."
   ::= { snmpTraps 3 }
```

Το πρώτο αντικείμενο είναι η συγκεκριμένη διεπαφή (ifindex) το οποίο μεταπήδησε *από την* linkDown κατάσταση σε κάποια άλλη. Το OID για αυτό το trap είναι *1.3.6.1.6.3.1.1.5.3*(iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIB Objects.snmpTraps.linkDown).

2.7.7. - SNMP inform

Το SNMPv2 παρέχει τον μηχανισμό inform ο οποίος επιτρέπει την επικοινωνία μεταξύ δύο managers. Αυτή η λειτουργία είναι χρήσιμη όταν υπάρχει η ανάγκη για περισσότερα από ένα NMS σε ένα δίκτυο. Όταν ένα μήνυμα inform στέλνεται από ένα NMS σε ένα άλλο, ο αποδέκτης στέλνει μια επιβεβαίωση στο αποστολέα ότι το έλαβε. Η συμπεριφορά δηλαδή είναι ίδια με των αιτήσεων get και set. Επίσης η λειτουργία SNMP inform μπορεί να χρησιμοποιηθεί για να στείλει SNMPv2 traps σε κάποιο NMS. Εάν χρησιμοποιήσουμε την λειτουργία inform για αυτό τον σκοπό τότε ο agent θα ειδοποιηθεί όταν το NMS λάβει το trap.

2.7.8. - SNMP report

Η λειτουργία report ορίστηκε στην πρόχειρη έκδοση του SNMPv2 αλλά ποτέ δεν υλοποιήθηκε. Τώρα είναι μέρος της SNMPv3 προδιαγραφής και προτείνεται να επιτρέψει δύο μηχανές SNMP να επικοινωνούν μεταξύ τους κυρίως για να αναφέρουν προβλήματα για την επεξεργασία SNMP μηνυμάτων.

2.8. - Εκδόσεις του SNMP

Στις παρακάτω ενότητες αναπτύσσονται συνοπτικά οι διαφορετικές εκδόσεις που αναπτύχθηκαν για το πρωτόκολλο SNMP.

2.8.1. - SNMPv1

Η πρώτη έκδοση SNMP (SNMPv1) είναι η αρχική εφαρμογή του πρωτοκόλλου SNMP. Το SNMPv1 λειτουργεί πάνω από τα πρωτόκολλα όπως το User Datagram Protocol (UDP), Internet Protocol (IP), OSI Connectionless Network Service (CLNS), AppleTalk Datagram-Delivery Protocol (DDP), και Novell Internet Packet Exchange (IPX). Το SNMPv1 χρησιμοποιείται ευρέως και είναι το de facto σαν πρωτόκολλο διαχείρισης στην κοινότητα του Διαδικτύου.

Το πρώτο RFCs για το SNMP, τώρα γνωστό ως SNMPv1, εμφανίστηκε το 1988:
RFC 1065 - Δομή και προσδιορισμός των διαχειρίσιμων πληροφοριών για TCP/IP-based internets
RFC 1066 - Βάση πληροφοριών διαχείρισης για τη διαχείριση δικτύων TCP/IP-based internets
RFC 1067 - Ένα απλό πρωτόκολλο διαχείρισης δικτύων

Αυτά τα πρωτόκολλα αντικαταστάθηκαν από:
RFC 1155 - Δομή και προσδιορισμός των διαχειρίσιμων πληροφοριών για TCP/IP-based internets
RFC 1156 - Βάση πληροφοριών διαχείρισης για τη διαχείριση δικτύων TCP/IP-based internets
RFC 1157 - Ένα απλό πρωτόκολλο διαχείρισης δικτύων

Μετά από έναν σύντομο χρονικό διάστημα το RFC 1156 (MIB-1) αντικαταστάθηκε από το συχνότερα χρησιμοποιούμενο:
RFC 1213 - Έκδοση 2 της βάσης πληροφοριών διαχείρισης(MIB-2) για τη διαχείριση δικτύων TCP/IP-based internets

Η πρώτη έκδοση έχει επικριθεί για τη φτωχή ασφάλειά της. Η αυθεντικοποίηση των πελατών εκτελείται μόνο από μια συμβολοσειρά community που στην πραγματικότητα είναι ένας τύπος συνθηματικού πρόσβασης, ο οποίος διαβιβάζεται

σαν καθαρό κείμενο. Το σχέδιο της δεκαετίας του '80 του SNMPv1 έγινε από μια ομάδα συνεργατών που είδαν την επίσημα υποστηριζόμενη από OSI/IETF/NSF (εθνικό ίδρυμα επιστήμης) προσπάθεια. Το SNMP εγκρίθηκε βασισμένο στην πεποίθηση ότι ήταν ένα προσωρινό πρωτόκολλο που απαιτήθηκε για τη λήψη μέτρων προς την επέκταση μεγάλης κλίμακας του Διαδικτύου και της εμπορευματοποίησής του.

2.8.2. - SNMPv2

Το SNMPv2 (RFC 1441–RFC 1452) αναθεωρεί την πρώτη έκδοση και περιλαμβάνει βελτιώσεις στους τομείς της απόδοσης, ασφάλειας, εμπιστευτικότητας και στην επικοινωνία μεταξύ managers. Εισάγει την λειτουργία GetBulk η οποία μπορεί να ανακτήσει μεγάλη ποσότητα διαχειρίσιμης πληροφορίας μέσα από μία αίτηση. Ωστόσο το καινούργιο σύστημα ασφάλειας του SNMPv2 θεωρήθηκε αρκετά πολύπλοκο και δεν έγινε ευρέως αποδεκτό.

Το Community-Based Simple Network Management Protocol version 2, ή SNMPv2c, ορίζετε στα RFC 1901–RFC 1908. Στα αρχικά στάδιά του, αυτό ήταν επίσης ανεπίσημα γνωστό σαν *SNMP v1.5*. Το SNMPv2c περιλαμβάνει το SNMPv2 χωρίς το αμφισβητούμενο νέο πρότυπο ασφάλειας του SNMPv2, και χρησιμοποιεί αντί αυτού το απλό σύστημα ασφάλειας του SNMPv1. Επίσημα θεωρείται ένα πρόχειρο πρότυπο αλλά είναι το ποιο ευρέως χρησιμοποιούμενο πρότυπο.

Το User-Based Simple Network Management Protocol version 2, ή SNMPv2u, ορίζεται στα RFC 1909–RFC 1910. Είναι ένας συμβιβασμός ο οποίος επιχειρεί να προσφέρει μεγαλύτερη ασφάλεια από το SNMPv1 αλλά χωρίς να εμπεριέχει την μεγάλη πολυπλοκότητα του SNMPv2. Μια παραλλαγή του εμπορευματοποιήθηκε σαν SNMPv2*, και ο μηχανισμός του τελικά υιοθετήθηκε σαν ένα από τα δυο πλαίσια ασφαλείας του SNMPv3.

2.8.3. - Διαλειτουργικότητα μεταξύ SNMPv1 και SNMPv2

Όπως μέχρι στιγμής διευκρινίζεται, το SNMPv2 είναι ασυμβίβαστο με το SNMPv1 σε δύο βασικές περιοχές, στα σχήματα μηνυμάτων και στις διαδικασίες πρωτοκόλλου. Στο SNMPv2c τα μηνύματα χρησιμοποιούν διαφορετικές κεφαλίδες

και σχήματα πρωτοκόλλου (PDU) από τα μηνύματα του SNMPv1. Το SNMPv2c χρησιμοποιεί επίσης δύο διαδικασίες πρωτοκόλλου που δεν διευκρινίζονται στο SNMPv1. Επιπλέον, το RFC 1908 καθορίζει δύο πιθανές στρατηγικές συνύπαρξης SNMPv1/v2c: πληρεξούσιοι agents και δίγλωσσα συστήματα διαχείρισης δικτύου.

2.8.3.1. - Πληρεξούσιοι agents (proxy agents)

Ένας SNMPv2 agent μπορεί να λειτουργήσει σαν proxy agent για να εξυπηρετήσει SNMPv1 συσκευές ως εξής.

- Ένα NMS SNMPv2 θέτει μια εντολή η οποία προορίζετε για ένα SNMPv1 agent.
- Το NMS στέλνει το μήνυμα SNMP σε ένα SNMPv2 proxy agent.
- Ο proxy agent προωθεί τα μηνύματα get, getnext, και set στον SNMPv1 agent απαράλλαχτα.
- Τα μηνύματα getBulk μετατρέπονται από τον proxy agent σε getnext και μετά προωθούνται στο SNMPv1 agent.

Ο proxy agent μετατρέπει τα μηνύματα των traps από SNMPv1 σε SNMPv2 και μετά τα προωθεί στο NMS.

2.8.3.2. - Τρίγλωσσα συστήματα διαχείρισης δικτύου.

Τα τρίγλωσσα συστήματα διαχείρισης δικτύου υποστηρίζουν το SNMPv1, το SNMPv2 αλλά και το SNMPv3. Για να υποστηριχτεί αυτό το τριπλό περιβάλλον το NMS επικοινωνεί συνήθως με τον agent και διαπιστώνει ποια έκδοση υποστηρίζει. Έτσι κάθε φορά που είναι να επικοινωνήσει με έναν agent το NMS συμβουλεύεται την βάση δεδομένων του και ανάλογα με την έκδοση που υποστηρίζει ο agent, το NMS χρησιμοποιεί την κατάλληλη έκδοση του SNMP.

2.8.4. - SNMPv3

Παρόλο που το SNMPv3 δεν εισάγει αλλαγές στο πρωτόκολλο εκτός από την προσθήκη της κρυπτογραφημένης ασφάλειας, οι προγραμματιστές του φρόντισαν τα πράγματα να αλλάξουν αρκετά χρησιμοποιώντας καινούργιες λεκτικές συμβάσεις, έννοιες και ετυμολογία. Το SNMPv3 βασικά πρόσθεσε ασφάλεια και την δυνατότητα στον χρήστη να κάνει ρυθμίσεις απομακρυσμένα.

Η ασφάλεια ήτανε η μεγαλύτερη αδυναμία του SNMP από την αρχή. Η αυθεντικοποίηση στις δυο πρώτες εκδόσεις δεν ήτανε τίποτα παραπάνω από ένα συνθηματικό το οποίο μάλιστα μεταδιδότανε σαν καθαρό κείμενο μεταξύ του manager και του agent. Κάθε μήνυμα SNMPv3 περιέχει παραμέτρους ασφάλειας οι οποίες είναι κωδικοποιημένες σαν οκταδικά αλφαριθμητικά. Η σημασία αυτών των παραμέτρων βασίζεται στο μοντέλο ασφάλειας που χρησιμοποιείται.

Το SNMPv3 παρέχει τα παρακάτω σημαντικά χαρακτηριστικά ασφαλείας:

- Εμπιστευτικότητα – Κρυπτογράφηση των πακέτων
- Ακεραιότητα – Εξασφαλίζει ότι το πακέτο δεν παραλλάχθηκε κατά την μεταφορά και περιλαμβάνει και ένα επιπλέον μηχανισμό προστασίας ώστε να μην μπορεί να γίνει επανάληψη του ίδιου πακέτου.
- Αυθεντικοποίηση – Εξασφαλίζει ότι το μήνυμα είναι από μια έγκυρη πηγή.

Από το 2004 η IETF αναγνωρίζει το SNMPv3 όπως ορίζεται στα RFC 3411–RFC 3418 σαν την τρέχον πρότυπη έκδοση του SNMP. Η IETF σχεδίασε το SNMPv3 σαν ένα πλήρες πρότυπο του διαδικτύου, το μεγαλύτερο επίπεδο ωριμότητας για ένα RFC και θεωρεί τις προηγούμενες εκδόσεις ξεπερασμένες. Στην πράξη βέβαια προς το παρόν, η υλοποίηση του SNMP σε ένα δίκτυο συνήθως υποστηρίζει και τις τρεις εκδόσεις (SNMPv1, SNMPv2c, SNMPv3).

2.9. - Επίλογος

Στο κεφάλαιο αυτό παρουσιάστηκε το πρωτόκολλο SNMP το οποίο αποτελεί το βασικότερο πρωτόκολλο σήμερα για την διαχείριση δικτύων. Αναφέραμε το πώς ενθυλακώνεται μέσα στο πρωτόκολλο μεταφοράς UDP και πως χρησιμοποιεί το σύστημα αυθεντικοποίησης με συμβολοσειρές community. Έγινε επεξήγηση για τους τύπους δεδομένων που χρησιμοποιεί για να μεταφέρει την πληροφορία και το πώς είναι δομημένη η σύνταξη των αντικειμένων μέσω της SMI (Structure of Management Interface) η οποία αποτελεί ένα υποσύνολο της ASN.1(Abstract Syntax Notation One). Επίσης αναφέραμε τι είναι η βάση πληροφοριών διαχείρισης (MIB) και περιγράψαμε τις λειτουργίες που μπορεί να εκτελέσει το SNMP, το πώς δηλαδή μεταφέρει την πληροφορία και τα διαφορετικά μηνύματα που μπορεί να στείλει. Τέλος αναφερθήκαμε στις τρεις διαφορετικές εκδόσεις του

SNMP, στις διαφορές τους, καθώς και τι λύσεις υπάρχουν ώστε να συνυπάρχουν και οι τρεις εκδόσεις σε ένα δίκτυο.

Τα πράγματα προς το παρόν δείχνουν ότι το SNMP σαν πρωτόκολλο έχει γνωρίσει μεγάλη αποδοχή και με την έκδοση SNMPv3 που έλυσε κάποια σημαντικά προβλήματα στην ασφάλεια αναμένεται να μείνει πολύ καιρό ακόμα στο προσκήνιο της διαχείρισης δικτύων. Στο επόμενο κεφάλαιο θα αναφερθούμε συνοπτικά σε διάφορα εργαλεία και συστήματα που μπορούμε να χρησιμοποιήσουμε για την διαχείριση ενός δικτύου, τα οποία χρησιμοποιούν το SNMP για την συλλογή των πληροφοριών.

ΚΕΦΑΛΑΙΟ 3

Εργαλεία διαχείρισης δικτύου

3.1. - Εισαγωγή

Όπως αναφέραμε στο προηγούμενο κεφάλαιο, ένα σύστημα διαχείρισης δικτύου αποτελείται από διάφορα εργαλεία που είτε είναι ενοποιημένα σε ένα περιβάλλον είτε τα χρησιμοποιούμε ξεχωριστά το καθένα για να εκτελέσουμε πιο εξειδικευμένες εργασίες. Μερικά από αυτά είναι:

ΕΡΓΑΛΕΙΑ ΑΝΟΙΧΤΟΥ ΚΩΔΙΚΑ

- PHP-Syslog-NG
- Nagios
- RRDtool
- MRTG
- Cacti
- OpenNMS
- X TACACS
- jffnms

ΕΜΠΟΡΙΚΑ ΕΡΓΑΛΕΙΑ ΚΛΕΙΣΤΟΥ ΚΩΔΙΚΑ

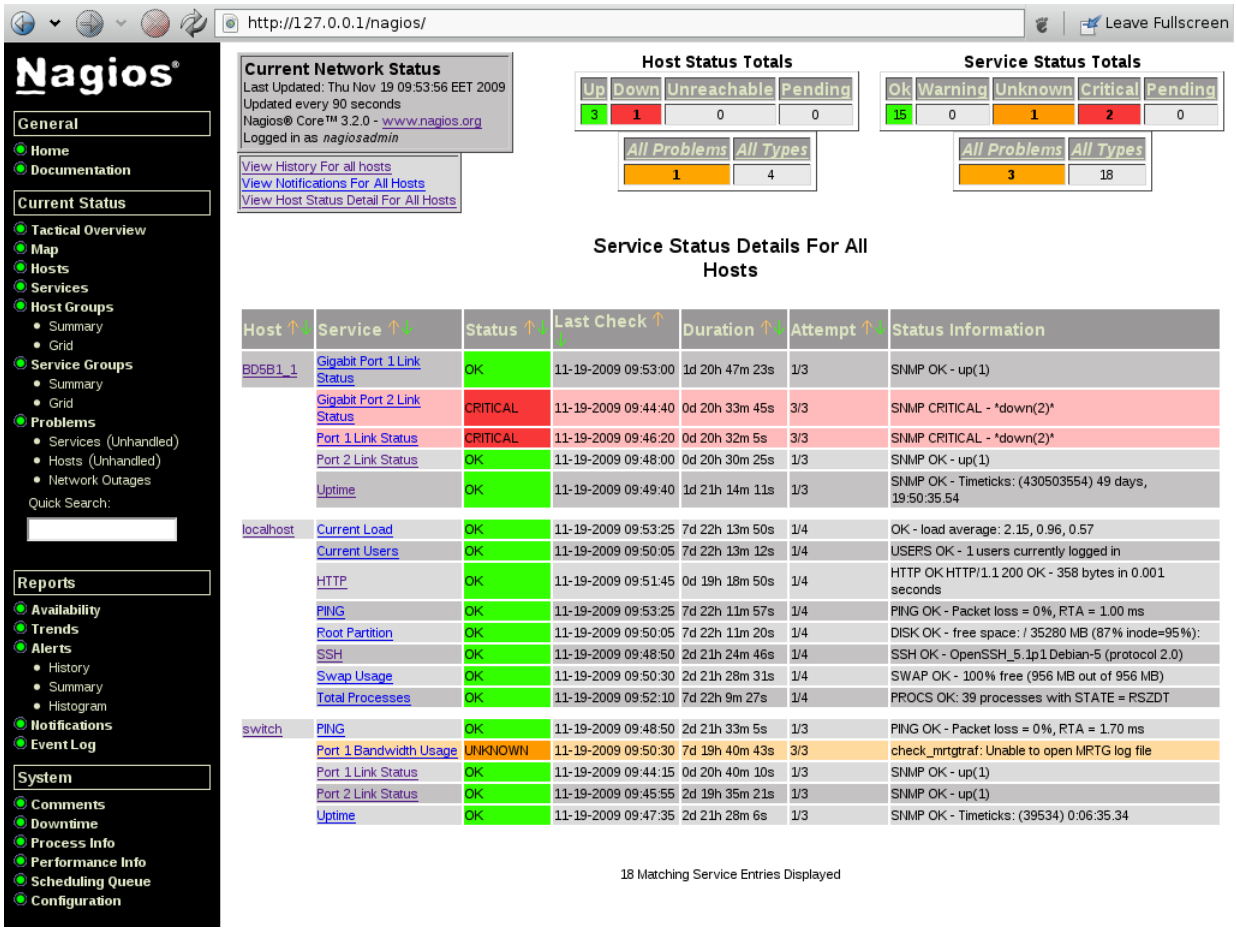
- AccelOps
- dopplerVUE
- TTI Telecom Netrac
- InfoVista VistaInsight
- NetQoS
- Opsware
- CiscoWorks
- CiscoSecure ACS
- ESM/EEM (IOS Embedded Event Manager)
- NetFlow
- IPSla
- RMON

- EMC
- CA Unicenter
- Spectrum
- FireScope BSM
- Ipswitch What's Up Gold
- Orion Solarwinds
- HP OpenView
- ECI Lightsoft
- Siemens siNMs
- PacketTrap pt360
- Itilon Moniton
- netPrefect

Στο κεφάλαιο αυτό θα παρουσιαστούν συνοπτικά μερικά από τα εργαλεία ανοιχτού κώδικα που εγκαταστάθηκαν και δοκιμαστήκαν στο περιβάλλον του νοσοκομείου Αχέπα για να καταλάβουμε στην πράξη τι παρέχει ένα σύστημα διαχείρισης δικτύου.

3.2. - Nagios

Το σύστημα διαχείρισης δικτύου Nagios παρέχει την δυνατότητα σε οργανισμούς να αναγνωρίσουν και να επιλύσουν προβλήματα της υποδομής δικτύου πριν επηρεαστούν βασικές λειτουργίες και υπηρεσίες. Παρακάτω ακολουθεί μια εικόνα από το σύστημα Nagios που μας δείχνει συνοπτικά την κατάσταση των υπηρεσιών ενός σταθμού καθώς και την κατάσταση δύο κατανεμητών και κάποιων πορτών που διαθέτουνε.



Σχήμα 3.1 – Το σύστημα διαχείρισης δικτύου Nagios

Το σύστημα διαχείρισης δικτύου NAGIOS παρέχει:

- Σχεδιασμό αναβάθμισης της υποδομής πριν κάποιο σύστημα αντιμετωπίσει προβλήματα.
- Απόκριση σε προβλήματα από την πρώτη στιγμή.
- Αυτόματη διόρθωση προβλημάτων όταν εντοπίζονται
- Συντονίζει τις διαφορετικές ομάδες τεχνικών
- Επιβεβαιώνει την ποιότητα των υπηρεσιών σε ένα δίκτυο
- Παρακολουθεί όλη την υποδομή του δικτύου και τις υπηρεσίες της.

Παρακολούθηση (Monitor)

Το Nagios μπορεί να παρακολουθεί βασικά συστατικά της υποδομής ενός δικτύου, όπως μετρικές συστημάτων, δικτυακά πρωτόκολλα, εφαρμογές, εξυπηρετητές και υπηρεσίες του δικτύου.

Ειδοποιήσεις (Alerts)

Το Nagios στέλνει ειδοποιήσεις όταν συστατικά της δικτυακής υποδομής αποτυγχάνουν ή επανέρχονται παρέχοντας έτσι την δυνατότητα στους διαχειριστές να αντιλαμβάνονται σημαντικά γεγονότα. Οι ειδοποιήσεις μπορούν να παραδίδονται είτε μέσω ηλεκτρονικού ταχυδρομείου, είτε μέσω SMS, είτε μέσω εξειδικευμένων προγραμμάτων(scripts) που έχουμε την δυνατότητα να συντάξουμε εμείς.

Εκθέσεις (Reports)

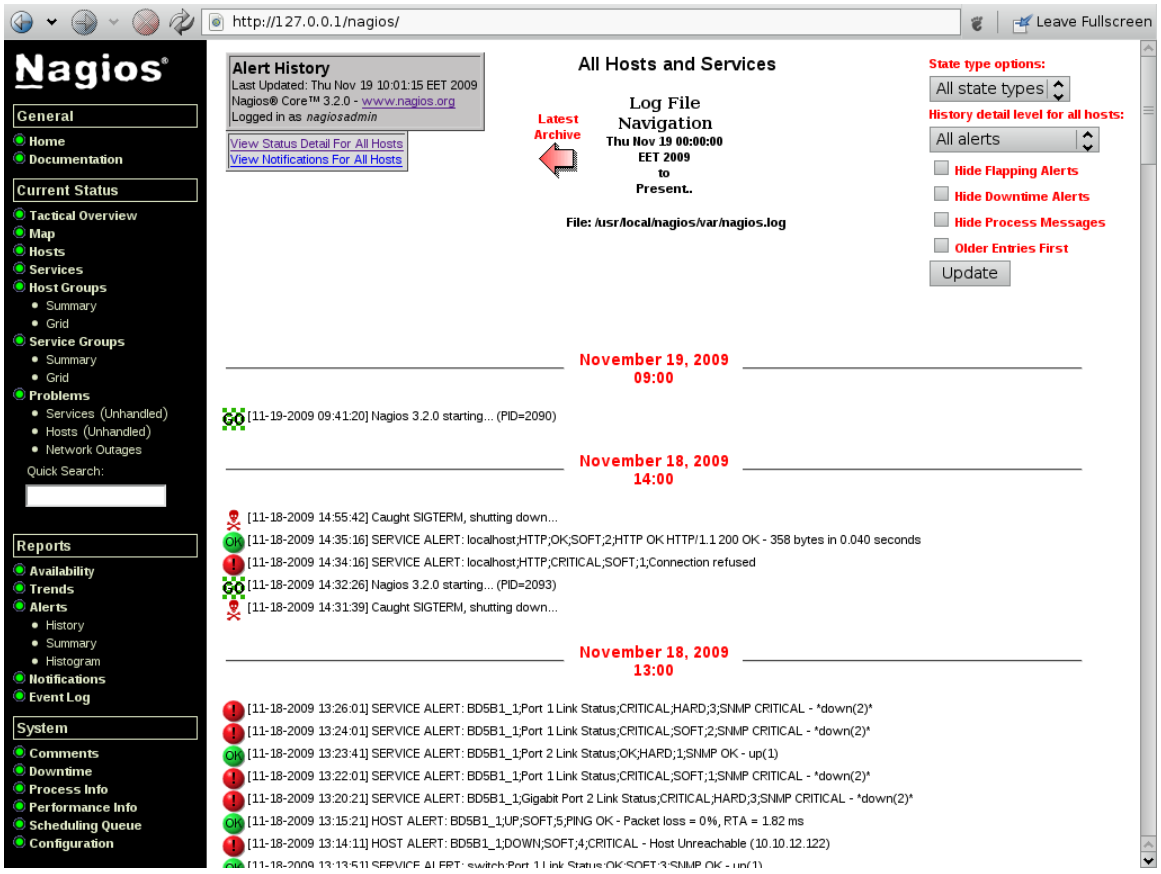
Οι εκθέσεις που μπορούν να παραχθούν μπορούν να περιέχουν ιστορικά στοιχεία για διακοπές στην λειτουργία, γεγονότα και ειδοποιήσεις. Οι εκθέσεις διαθεσιμότητας μας βοηθούν να διαπιστώσουμε εάν η ποιότητα των υπηρεσιών είναι στα επίπεδα που επιθυμούμε.

Συντήρηση (Maintance)

Έχουμε την δυνατότητα να προγραμματίσουμε το Nagios έτσι ώστε να μην στέλνει ειδοποιήσεις σε περιόδους που γίνεται συντήρηση η αναβάθμιση των συστημάτων.

Σχεδιασμός (Planning)

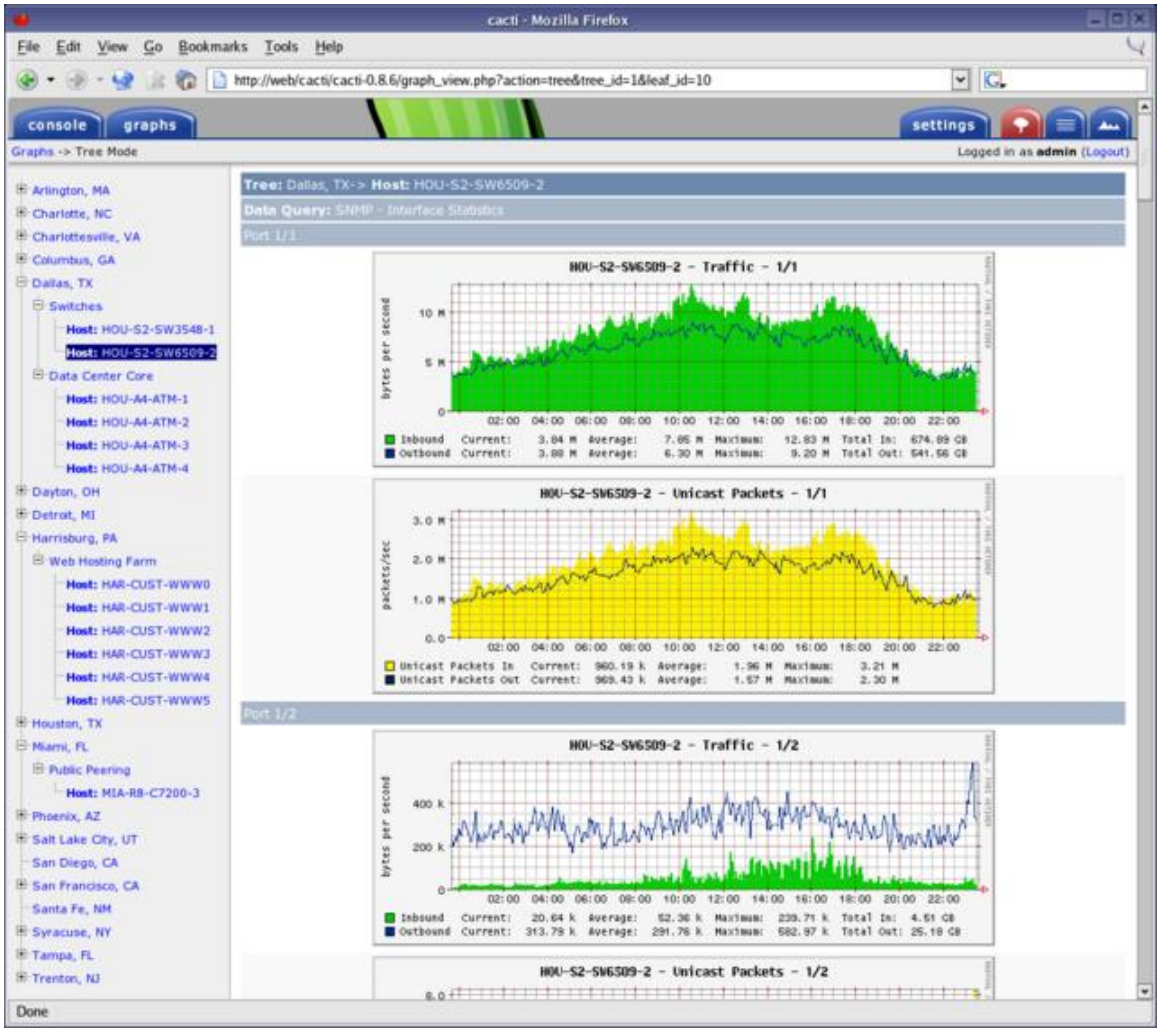
Διαγράμματα χωρητικότητας, ποιότητας και οι εκθέσεις μας παρέχουν την δυνατότητα να σχεδιάσουμε αναβαθμίσεις της υποδομής πριν κάποιο σύστημα αντιμετωπίσει προβλήματα.



Σχήμα 3.2 – Οι ειδοποιήσεις του συστήματος διαχείρισης δικτύου Nagios

3.3. - Cacti

Το Cacti είναι ένα πλήρες frontend για το εργαλείο RRDTool, το οποίο αποθηκεύει όλες τις απαραίτητες πληροφορίες για να δημιουργεί γραφήματα και να τα αποθηκεύει στην βάση δεδομένων MySQL. Η διεπαφή του προγράμματος είναι προγραμματισμένη σε PHP. Εκτός από ότι είναι δυνατόν να συντηρεί γραφήματα, πηγές πληροφορίας και Round Robin Archives σε μια βάση δεδομένων μπορεί επίσης να χειριστεί και την συλλογή της πληροφορίας.



Σχήμα 3.2 – Το σύστημα διαχείρισης δικτύου Cacti

Πηγές πληροφορίας (Data Sources)

Για να διαχειριστεί το Cacti την συλλογή πληροφορίας μπορούμε να δηλώσουμε διαδρομές προς ένα εξωτερικό πρόγραμμα-εντολή μαζί με την πληροφορία που ο χρήστης θέλει να συλλέξει. Το Cacti μετά θα εκτελέσει την εντολή μέσω του crond και θα ενημερώσει την βάση δεδομένων ή την συλλογή Round Robin.

Γραφήματα (Graphs)

Από την στιγμή που μία ή περισσότερες πηγές πληροφορίας οριστούνε, ένα γράφημα τύπου RRDTool μπορεί να δημιουργηθεί χρησιμοποιώντας την πληροφορία αυτή. Το Cacti μπορεί να δημιουργήσει σχεδόν κάθε γράφημα τύπου RRDTool που μπορούμε να φανταστούμε χρησιμοποιώντας όλα τα πρότυπα του RRDTool και τις συναρτήσεις ενοποίησης.

Διαχείριση χρηστών (User Management)

Στο Cacti μέσω ενός εργαλείου για την διαχείριση των χρηστών μπορούμε να προσθέτουμε χρήστες και να τους δώσουμε δικαιώματα για συγκεκριμένες περιοχές του. Έτσι μπορούμε να δημιουργήσουμε χρήστες οι οποίοι έχουν την δυνατότητα να αλλάζουν τις παραμέτρους των γραφημάτων ενώ κάποιοι άλλοι απλά να μπορούν να βλέπουν τα γραφήματα.

Δημιουργία προτύπων (Templating)

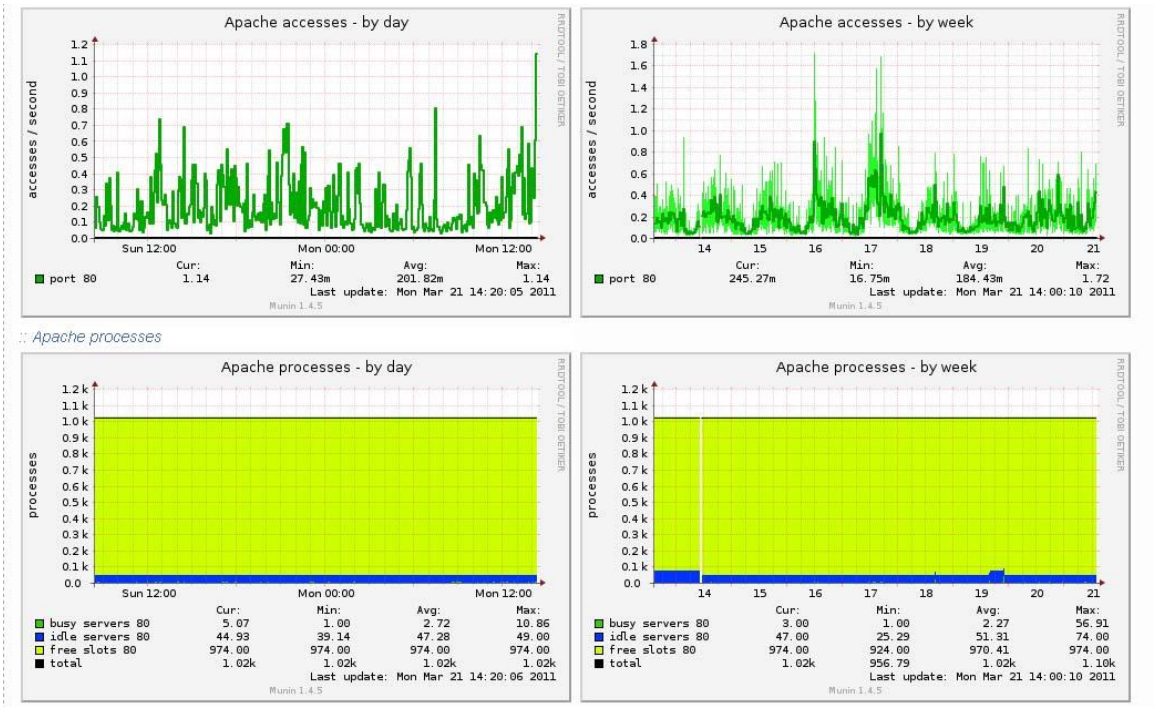
Το Cacti είναι δυνατόν να συλλέξει από μεγάλο αριθμό πηγών πληροφοριών μέσω της δυνατότητας προτύπων. Τα πρότυπα στο Cacti στην ουσία είναι μία συλλογή από δεδομένα που θέλουμε να συλλέξουμε. Έτσι δημιουργώντας ένα πρότυπο μπορούμε να το εφαρμόσουμε σε πολλούς ίδιους σταθμούς ή εξυπηρετητές χωρίς να χρειάζεται να εισάγουμε για τον καθένα ξεχωριστά τι θα συλλέξουμε.

3.4. - Munin

Το Munin είναι μία εφαρμογή για την παρακολούθηση δικτύων και συστημάτων το οποίο αναπαριστά την πληροφορία γραφημάτων μέσω ενός webinterface. Δίνει ιδιαίτερη σημασία σε δυνατότητες plug n play. Η λειτουργία του μπορεί να επεκταθεί με plugins (υπάρχουν διαθέσιμα πάνω από 500).

Χρησιμοποιώντας το Munin μπορούμε να παρακολουθήσουμε την απόδοση υπολογιστών, δικτύων και εφαρμογών. Με το Munin μπορούμε εύκολα να διαπιστώσουμε απότομες διακυμάνσεις σε ένα γράφημα που μπορεί να υπονοεί κάποιο πρόβλημα απόδοσης η κάποιο πρόβλημα έλλειψης χωρητικότητας. Όπως και το Cacti χρησιμοποιεί το RRDTool και είναι γραμμένο στην γλώσσα προγραμματισμού Perl. Ένας από τους σκοπούς του είναι η ευκολία δημιουργίας νέων plugins.

Παρακάτω βλέπουμε ένα γράφημα που παρουσιάζει τις προσπελάσεις και τις διεργασίες του λογισμικού Apache.



Σχήμα 3.3 – Γραφήματα της εφαρμογής Munin

3.5. - Net-Snmp

Το Net-Snmp είναι μια σουίτα εφαρμογών η οποία υλοποιεί το SNMPv1, SNMPv2c και SNMPv3 χρησιμοποιώντας το IPv4 αλλά και το IPv6. Η σουίτα περιλαμβάνει τα παρακάτω:

- Εφαρμογές γραμμής εντολών για:

  - Συλλογή πληροφοριών από μια συσκευή που υλοποιεί το SNMP, είτε μέσω απλών αιτήσεων (snmpget, snmpgetnext), η πολλαπλών αιτήσεων (snmpwalk, snmptable, snmpdelta).

  - Διαχείριση των ρυθμίσεων σε μια συσκευή (snmpset)

  - Συλλογή καθορισμένων πληροφοριών από μία συσκευή. (snmpdf, snmpnetstat, snmpstatus).

  - Μετατροπή μεταξύ αριθμητικών και μορφών κειμένου σε MIB OIDs, και παρουσίαση των περιεχομένων και της δομής ενός MIB (snmptranslate).
- Ένα περιηγητή με γραφικό περιβάλλον για MIB που χρησιμοποιεί Tk/perl.
- Μια εφαρμογή - δαίμονας για να λαμβάνει SNMP notifications. Τα notifications μπορούν να καταγραφούν να προωθηθούν σε ένα άλλο σύστημα διαχείρισης η να σταλθούν σε μια εξωτερική εφαρμογή.

- Ένας επεκτάσιμος SNMP Agent για να αποκρίνεται σε αιτήματα για διαχειρίσιμη πληροφορία (snmpd). Ο agent περιλαμβάνει ένα μεγάλο εύρος από MIB modules και μπορεί να επεκταθεί δυναμικά φορτώνοντας modules, εξωτερικά προγράμματα (scripts) και εντολές και από τα δύο πρωτόκολλα SNMP multiplexing (SMUX) and Agent Extensibility (AgentX).
- Μια βιβλιοθήκη για την ανάπτυξη εφαρμογών SNMP, με C και perl APIs.

Η σουίτα Net-Snmp είναι διαθέσιμη για πολλά Unix και βασισμένα στο Unix συστήματα όπως επίσης και για Microsoft Windows. Η σουίτα χρησιμοποιήθηκε για την ανάπτυξη ενός προγράμματος (mac2port) στο δίκτυο του Νοσοκομείου Αχέπα η οποία παρουσιάζεται λεπτομερώς στην τέταρτη ενότητα.

3.6. - MRTG >> RRDTool

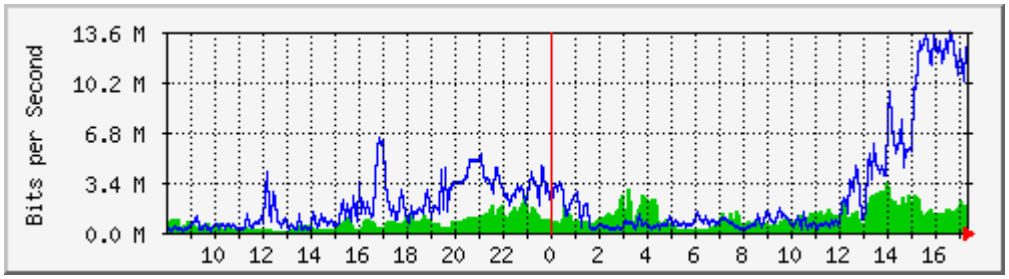
Το λογισμικό Multi Router Traffic Grapher (MRTG) αναπτύχθηκε αρχικά για την παρακολούθηση και για την δημιουργία γραφημάτων στατιστικών κίνησης σε μια σχετικά αργή και υπερφορτωμένη σύνδεση στο internet. Η έκδοση 1.0 κυκλοφόρησε το 1995 γραμμένη ολοκληρωτικά σε Perl και χρησιμοποιούσε αρκετά εξωτερικά προγράμματα για να συλλέξει τις πληροφορίες και να δημιουργήσει γραφήματα. Το MRTG έγινε γνωστό πολύ γρήγορα γιατί τα γραφήματα που δημιουργούσε αποδειχτήκαν πολύ χρήσιμα για την ενημέρωση των χρηστών μιας και μέσω ενός περιηγητή ιστού μπορούσαμε να δούμε στατιστικά ενός δρομολογητή.

Το 1997 κυκλοφόρησε η δεύτερη έκδοση με κάποια μέρη της εφαρμογής ξαναγραμμένα σε C και χρησιμοποιώντας την βιβλιοθήκη SNMP extension for Perl. Επίσης προστέθηκε και ένα πρόγραμμα που λέγεται cfgmaker με το οποίο μπορούμε να δημιουργήσουμε σκελετούς από αρχεία ρυθμίσεων χωρίς να χρειάζεται να έχουμε ιδιαίτερες γνώσεις του πρωτοκόλλου SNMP.

Παρόλο που το MRTG 2.0 ήτανε αρκετά ποιο γρήγορο υπήρχαν ακόμα περιορισμοί στην απόδοσή του και στο είδος της πληροφορίας που μπορούσε να συλλέξει. Αυτό οδήγησε στην ανάπτυξη του RRDTool, ένα σύστημα για αποδοτική ενημέρωση και ενοποίηση βάσεων δεδομένων τύπου round robin για πληροφορίες

σειριακού-χρόνου. Από την στιγμή που το RRDTool είναι αρκετά αποδοτικό το κρίσιμο κομμάτι πλέον είναι αυτό τις συλλογής της πληροφορίας.

Το MRTG 3.0 παρέχει βελτιώσεις στην συλλογή της πληροφορίας μέσω του SNMP εκτελώντας πολλαπλές ταυτόχρονες αιτήσεις ώστε να μειώσει την συνολική καθυστέρηση.



Σχήμα 3.4 – Γράφημα της εφαρμογής MRTG

3.7. - Επίλογος

Σε αυτό το κεφάλαιο αναφέρθηκαν και παρουσιάστηκαν συνοπτικά κάποια βασικά εργαλεία - συστήματα για την διαχείριση ενός δικτύου, το καθένα με λίγο διαφορετικά χαρακτηριστικά από το άλλο. Κάποια από αυτά αποτελούν μια πλήρης λύση για την διαχείριση ενός δικτύου όπως το Nagios, ενώ κάποια άλλα μπορούν να χρησιμοποιηθούν σαν επιμέρους εργαλεία για την καταγραφή κίνησης και την δημιουργία γραφημάτων όπως είναι το Cacti, το Munin και το MRTG. Σίγουρα υπάρχουν πολύ καλές λύσεις, και ολοκληρωμένες λύσεις στην μεριά των εμπορικών προγραμμάτων όπως π.χ. το HP OpenView παρόλα αυτά και οι λύσεις ανοιχτού κώδικα μπορούν να είναι εξίσου αποδοτικές με το πλεονέκτημα της επεκτασιμότητας κατά κύριο λόγο. Για το περιβάλλον του νοσοκομείου Αχέπα επιλέχθηκε το OpenNMS το οποίο δεν αναφέρεται σε αυτό το κεφάλαιο γιατί ακολουθεί εκτενέστερη αναφορά στα επόμενα. Αποτελεί και αυτό μία ολοκληρωμένη λύση ανοιχτού κώδικα και το κριτήριο επιλογής του ήτανε κατά κύριο λόγω ότι δίνει αρκετά μεγάλη σημασία στον έλεγχο τον υπηρεσιών σε ένα δίκτυο καθώς και η ευκολία κατά τις βασικές ρυθμίσεις του.

Όταν αναφερόμαστε σε ένα σύστημα διαχείρισης δεν εννοούμε πάντα ένα συγκεκριμένο λογισμικό αλλά μια πληθώρα εργαλείων τα οποία μπορεί να βρίσκονται και να εκτελούνται από το ίδιο υλικό (hardware) ή να είναι

κατανεμημένα σε πολλούς εξυπηρετητές. Η επιλογή που θα κάνουμε πάντα θα πρέπει να βασίζεται στις εκάστοτε ανάγκες και αν δεν μας καλύπτει μία λύση τότε μπορούμε να χρησιμοποιήσουμε συνοδευτικά εργαλεία. Από αυτό συμπεραίνεται ότι η διαχείριση δικτύου δεν είναι κάτι το καθορισμένο αλλά είναι μια διαδικασία που μπορεί να μεταβάλλεται συνεχώς όπως και η ασφάλεια σε ένα δίκτυο.

Τέλος το καλό είναι ότι υπάρχουν πολλές επιλογές είτε κλειστού, είτε ανοιχτού κώδικα και αυτό μας δίνει την δυνατότητα να επιλέξουμε ή και να συνδυάσουμε διάφορες λύσεις ώστε να μπορούμε να έχουμε μια συνολική ματιά για την λειτουργία και την απόδοση του δικτύου μας.

Δεύτερη Ενότητα

Παρουσίαση και τρόπος λειτουργίας του OpenNMS

ΚΕΦΑΛΑΙΟ 4

OPENNMS - Εύρεση κόμβων και υπηρεσιών

4.1. - Τι είναι το OpenNMS

Το OpenNMS είναι μια πλατφόρμα λογισμικού για την παρακολούθηση και διαχείριση ενός δικτύου η οποία αναπτύσσετε κάτω από το μοντέλο του λογισμικού ανοιχτού κώδικα και το μοντέλο ελεύθερου λογισμικού. Απαρτίζετε από μια κοινότητα ελεύθερου λογισμικού όπως επίσης και από ένα οργανισμό ο οποίος παρέχει εμπορικές υπηρεσίες, εκπαίδευση και υποστήριξη.

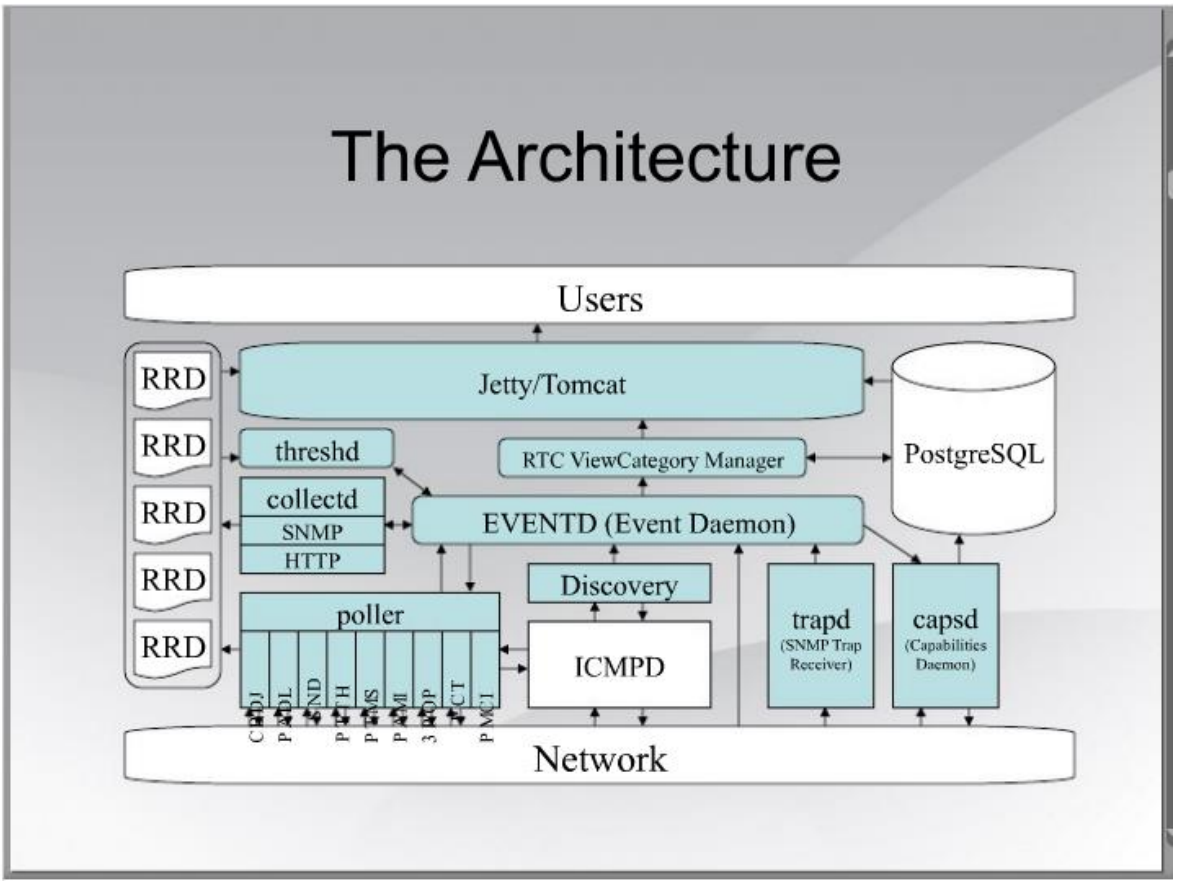
Ο σκοπός είναι το OpenNMS να είναι μια πλήρως κατανεμημένη, κλιμακούμενη πλατφόρμα που να καλύπτει όλες τις πλευρές του μοντέλου διαχείρισης δικτύου FCAPS, και να γίνει αυτή η πλατφόρμα διαθέσιμη είτε σε εμπορικές εφαρμογές, είτε σε εφαρμογές ανοιχτού κώδικα.

Όλος ο κώδικας που συσχετίζεται με το OpenNMS είναι διαθέσιμος κάτω από την άδεια GNU General Public License.

4.2. - Τι παρέχει το OpenNMS

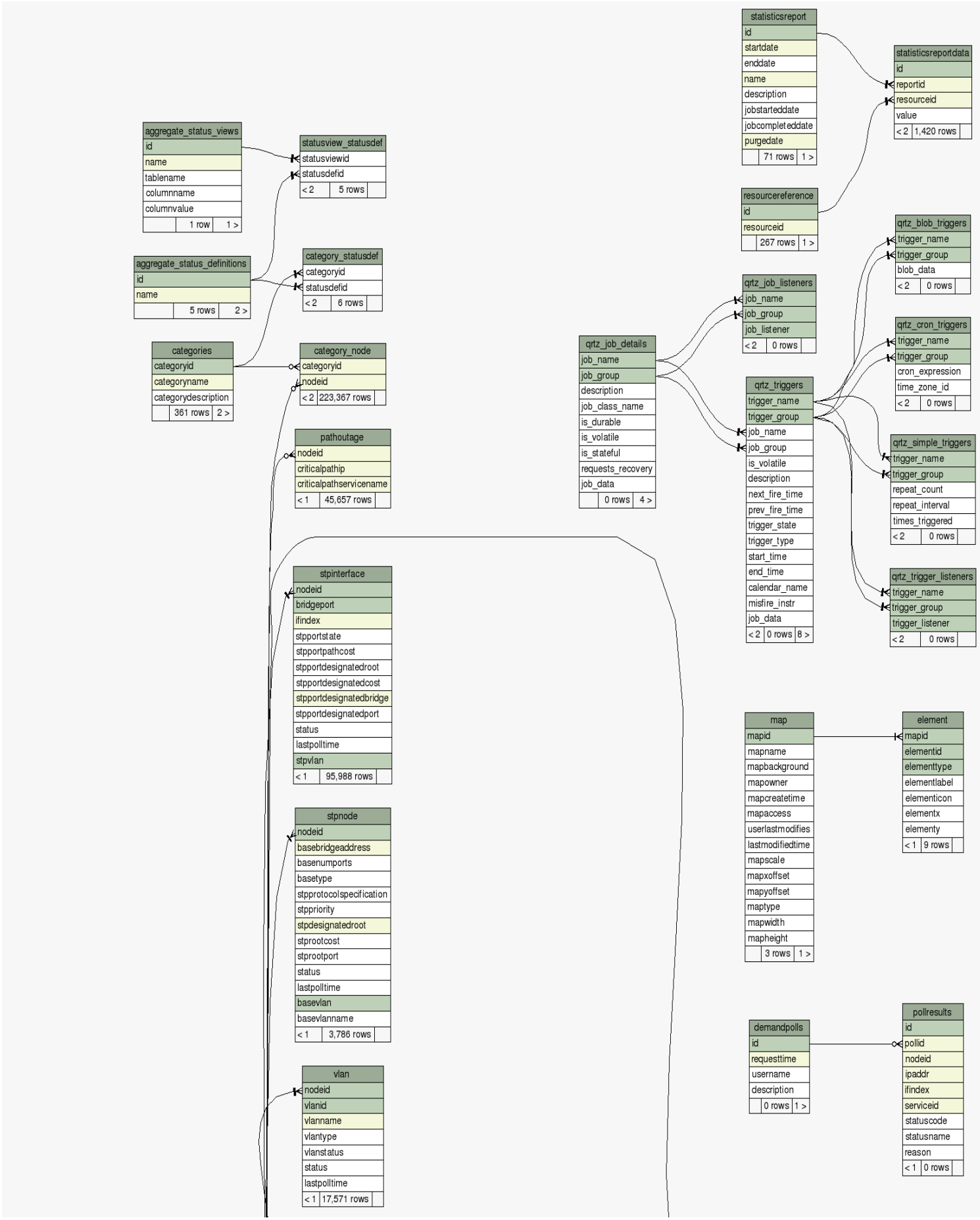
- Ενεργό έλεγχο υπηρεσιών – ο οποίος προσδιορίζει την διαθεσιμότητα και την καθυστέρηση εξυπηρέτησης.
- Συλλογή πληροφοριών – συλλογή, αποθήκευση, και αναφορά σχετικά με πληροφορίες που συλλέγει από τις δικτυακές συσκευές μέσω των πρωτοκόλλων SNMP, JMX, HTTP και NSClient
- Εξέταση της συλλεγμένης πληροφορίας για την εύρεση τιμών οι οποίες υπερβαίνουν μέγιστα η ελάχιστα όρια που έχουμε ορίσει και δημιουργία γεγονότων σε τέτοιες περιπτώσεις.
- Ειδοποιήσεις για γεγονότα και συναγερμούς μέσω ηλεκτρονικού ταχυδρομείου, XMPP κ.α.

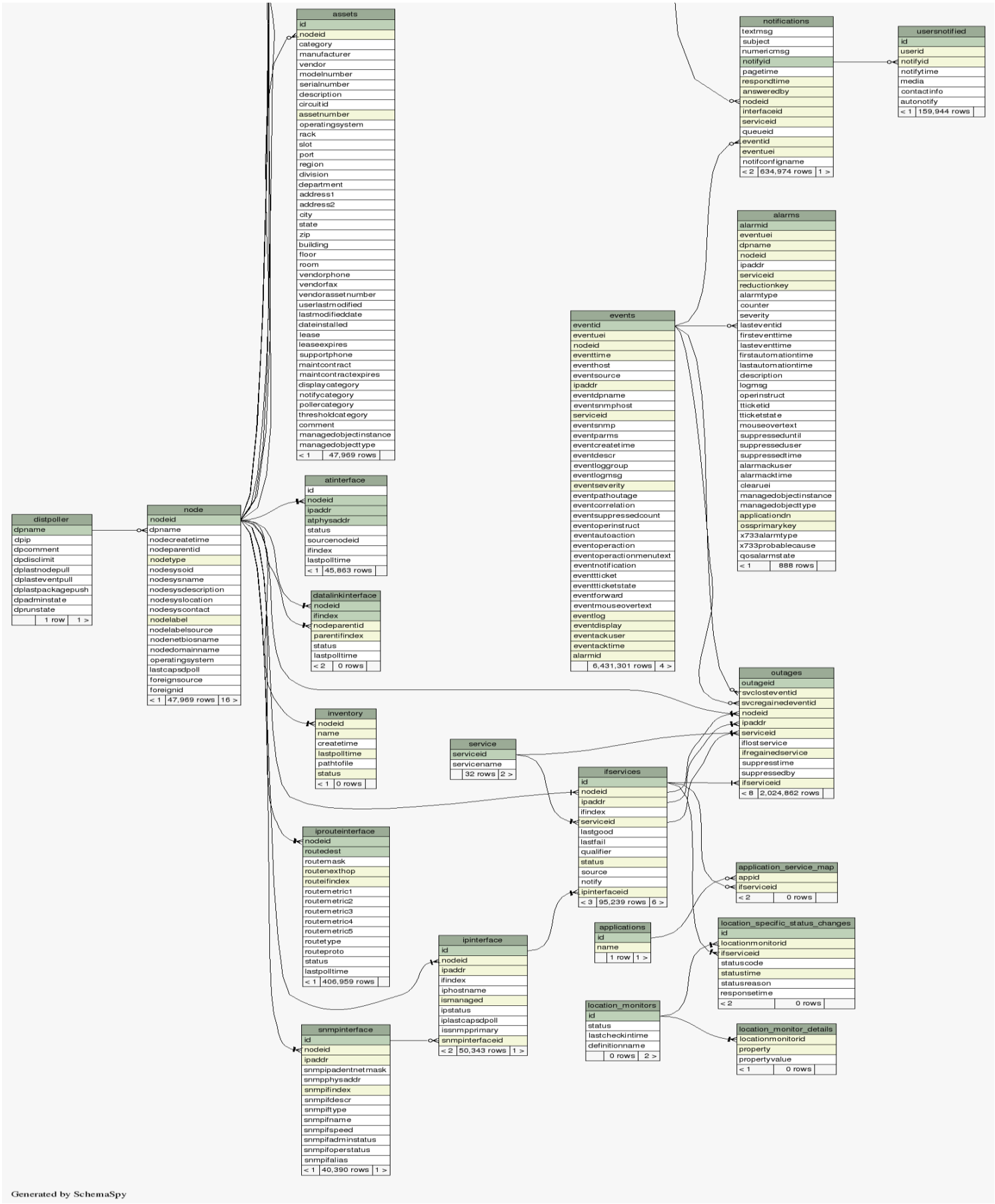
4.3. - Αρχιτεκτονική του OpenNMS



Σχήμα 4.1 – Η αρχιτεκτονική του OpenNMS

Στο σχήμα 4.1 βλέπουμε το πώς τα διάφορα προγράμματα συνδέονται και σχηματίζουν το σύστημα διαχείρισης δικτύου OpenNms. Παρατηρούμε ότι απαρτίζεται από διάφορους δαίμονες οι οποίοι κάνουν μια συγκεκριμένη εργασία και αλληλοσυνδέονται μεταξύ τους. Όπως φαίνεται και στο σχήμα οι δύο βασικές περιοχές που αποθηκεύεται η πληροφορία είναι η βάση δεδομένων PostgreSQL και τα αρχεία RRD. Στα επόμενα κεφάλαια αναλύετε η λειτουργία αυτών των δαιμόνων καθώς και τα αρχεία τα οποία επηρεάζουν τον τρόπο λειτουργίας τους. Στις δύο επόμενες σελίδες στο Σχήμα 4.2 φαίνεται το σχήμα της βάσης δεδομένων του OpenNMS.


statisticsreport
id
startdate
enddate
name
description
jobstarteddate
jobcompleteddate
purgedate
71 rows 1 >
statisticsreportdata
id
reportid
resourceid
value
< 2 1,420 rows
resourcereference
id
resourceid
267 rows 1 >
aggregate_status_views
id
name
tablename
columnname
columnvalue
1 row 1 >
statusview_statusdef
statusviewid
statusdefid
< 2 5 rows
aggregate_status_definitions
id
name
5 rows 2 >
category_statusdef
categoryid
statusdefid
< 2 6 rows
categories
categoryid
categoryname
categorydescription
361 rows 2 >
category_node
categoryid
nodeid
< 2 223,367 rows
pathoutage
nodeid
criticalpathip
criticalpathservicename
< 1 45,657 rows
qrtz_job_details
job_name
job_group
description
job_class_name
is_durable
is_volatile
is_stateful
requests_recovery
job_data
0 rows 4 >
qrtz_job_listeners
job_name
job_group
job_listener
< 2 0 rows
qrtz_triggers
trigger_name
trigger_group
job_name
job_group
is_volatile
description
next_fire_time
prev_fire_time
trigger_state
trigger_type
start_time
end_time
calendar_name
misfire_instr
job_data
< 2 0 rows 8 >
qrtz_blob_triggers
trigger_name
trigger_group
blob_data
< 2 0 rows
qrtz_cron_triggers
trigger_name
trigger_group
cron_expression
time_zone_id
< 2 0 rows
qrtz_simple_triggers
trigger_name
trigger_group
repeat_count
repeat_interval
times_triggered
< 2 0 rows
qrtz_trigger_listeners
trigger_name
trigger_group
trigger_listener
< 2 0 rows
stpinterface
nodeid
bridgeport
ifindex
stpportstate
stpportpathcost
stpportdesignatedroot
stpportdesignatedcost
stpportdesignatedbridge
stpportdesignatedport
status
lastpolltime
stpvlan
< 1 95,988 rows
stpnode
nodeid
basebridgeaddress
basenumports
basetype
stpprotocolspecification
stppriority
stpdesignatedroot
stprootcost
stprootport
status
lastpolltime
basevlan
basevlanname
< 1 3,786 rows
vlan
nodeid
vlanid
vlanname
vlantype
vlanstatus
status
lastpolltime
< 1 17,571 rows
map
mapid
mapname
mapbackground
mapowner
mapcreatetime
mapaccess
userlastmodifies
lastmodifiedtime
mapscale
mapxoffset
mapyoffset
maptype
mapwidth
mapheight
3 rows 1 >
element
mapid
elementid
elementtype
elementlabel
elementicon
elementx
elementy
< 1 9 rows
demandpolls
id
requesttime
username
description
0 rows 1 >
pollresults
id
pollid
nodeid
ipaddr
ifindex
serviceid
statuscode
statusname
reason
< 1 0 rows



Σχήμα 4.2 – Το σχήμα της βάσης δεδομένων του OpenNMS

Αντίθετα από τα παραδοσιακά συστήματα διαχείρισης δικτύου που είναι προσανατολισμένα στα στοιχεία δικτύων όπως τα interface στους κατανεμητές και τους δρομολογητές, το OpenNMS εστιάζει στις υπηρεσίες που οι πόροι του δικτύου παρέχουν. Όπως ιστοσελίδες, πρόσβαση στις βάσεις δεδομένων, DNS, DHCP, κ.λπ. (αν και οι πληροφορίες για τα στοιχεία δικτύων είναι επίσης διαθέσιμες).

Επειδή η πλειοψηφία των δικτυακών υπηρεσιών βασίζονται στο TCP/IP το OpenNMS είναι βασισμένο στο IP. Το βασικό διαχειριζόμενο “στοιχείο(element)” ονομάζετε interface και είναι μοναδικά αναγνωρισμένο με βάση μια IP διεύθυνση. Οι υπηρεσίες αντιστοιχίζονται σε interfaces και εάν ένας αριθμός interfaces βρίσκετε στην ίδια συσκευή τότε ομαδοποιούνται σε κόμβους(nodes).
Η εύρεση των κόμβων στο OpenNMS χωρίζετε σε δύο μέρη:

- Εύρεση μιας IP διεύθυνσης που θα ελέγχετε και θα παρακολουθείτε.
- Εύρεση των υπηρεσιών που υποστηρίζονται από την συγκεκριμένη IP διεύθυνση.

4.4. - Εύρεση μιας IP διεύθυνσης που θα ελέγχετε και θα παρακολουθείτε

Η εύρεση των interface και κόμβων του δικτύου με βάση την IP διεύθυνσή τους ρυθμίζετε μέσα από το αρχείο discovery-configuration.xml (βρίσκετε στο κατάλογο /usr/share/opennms/etc/) το οποίο μπορούμε να το επεξεργαστούμε είτε με ένα επεξεργαστή κειμένου είτε μέσα από to web-interface του OpenNMS. Το αρχείο αυτό ελέγχει μια διεργασία η οποία στέλνει πακέτα ICMP “ping” σε μία συγκεκριμένη ομάδα IP διευθύνσεων, εάν υπάρχει απάντηση στα ICMP πακέτα τότε δημιουργείτε ένα “new suspect” γεγονός και προσθέτει τον κόμβο στη βάση του OpenNMS. Η διεργασία αυτή ονομάζετε Discoveryd και είναι ένας από τους “δαίμονες” που απαρτίζουν το OpenNMS. Ένα παράδειγμα του αρχείου discovery-configuration έχει ως εξής:

```
<?xml version="1.0" encoding="UTF-8"?>
<discovery-configuration
   xmlns="http://xmlns.opennms.org/xsd/config/discovery" threads="2"
   packets-per-second="2" initial-sleep-time="300000"
   restart-sleep-time="86400000" retries="1" timeout="2000">
```

```
    <specific retries="1" timeout="2000">10.10.13.1</specific>
    <include-range retries="1" timeout="2000">
        <begin xmlns="">10.10.12.1</begin>
        <end xmlns="">10.10.12.254</end>
    </include-range>
    <include-range retries="1" timeout="2000">
        <begin xmlns="">10.10.0.1</begin>
        <end xmlns="">10.10.0.254</end>
    </include-range>
</discovery-configuration>
```

Οι γενικές παράμετροι του αρχείου είναι:

threads

Είναι ο αριθμός των νημάτων που θα χρησιμοποιηθούν για την εύρεση κόμβων.

packets-per-seconds

Είναι ο αριθμός πακέτων ICMP που θα παραχθεί κάθε δευτερόλεπτο. Η προεπιλογή είναι 1. Υπάρχει μια σχέση μεταξύ του packet-per-seconds και του αριθμού νημάτων. Εάν το δίκτυο έχει μια μέση καθυστέρηση των 500ms τότε αλλάζοντας των αριθμό packets-per-seconds σε 2 θα έχει ως αποτέλεσμα η εύρεση κόμβων να είναι ταχύτερη. Με την προϋπόθεση βέβαια ότι έχουμε ρυθμίσει να τρέχουν πάνω από ένα νήματα ώστε να προλαβαίνουν να στέλνονται δύο πακέτα το δευτερόλεπτο αλλιώς εάν τρέχει μόνο ένα νήμα θα στέλνει τα πακέτα όσο ταχύτερα γίνετε.

initial-sleep-time

Είναι ο χρόνος σε millisecond που η διεργασία εύρεσης κόμβων αρχίσει να στέλνει πακέτα αφότου το OpenNMS ξεκινήσει. Η προεπιλογή είναι πέντε λεπτά(300000ms). Αυτό συμβαίνει ώστε να υπάρχει χρόνος να εκκινήσει η εφαρμογή πλήρως πριν αρχίσει να αναζητά νέους κόμβους.

restart-sleep-time

Με το που ολοκληρωθεί η διεργασία εύρεσης κόμβων, αυτός είναι ο χρόνος που θα περιμένει μέχρι να ξανατρέξει. Η προεπιλογή είναι 24ώρες(86400000ms)

timeout

Είναι ο χρόνος που η διεργασία θα περιμένει για απάντηση σε ένα ICMP ”ping” πριν συμπεράνει ότι δεν υπάρχει τίποτα σε αυτή την IP διεύθυνση.

retries

Είναι ο αριθμός των προσπαθειών που θα κάνει η διεργασία να λάβει μια απάντηση σε ένα ICMP ”ping” πριν συμπεράνει ότι δεν υπάρχει τίποτα σε αυτή την IP διεύθυνση.

Εφόσον έχουμε ορίσει τις γενικές παραμέτρους μένει να πούμε στο “δαίμονα” ποια εύρη IP διευθύνσεων να ψάξει. Αυτό γίνετε με τις παρακάτω παράμετρους όπου μπορούμε σε διαφορετικά δίκτυα να ορίσουμε και διαφορετικές παραμέτρους από τις γενικές.

specific

Εδώ ορίζουμε μια IP διεύθυνση που θα ελέγχει. Μπορούμε να έχουμε πολλές <specific> ετικέτες.

<specific>*ip-address*</specific>

Include-range

Εδώ ορίζουμε ένα εύρος διευθύνσεων που θα ελεγχθούνε. Μπορούμε να έχουμε πολλές <include-range> ετικέτες.

<include-range>
  <begin>*start-ip-address*</begin>
  <end>*end-ip-address*</end>
</include-range>

exclude-range

Εδώ ορίζουμε ένα εύρος διευθύνσεων που θα δεν θέλουμε να ελεγχθούνε. Μπορούμε να έχουμε πολλές <exclude-range> ετικέτες.

<exclude-range>
  <begin>*start-ip-address*</begin>
  <end>*end-ip-address*</end>
</exclude-range>

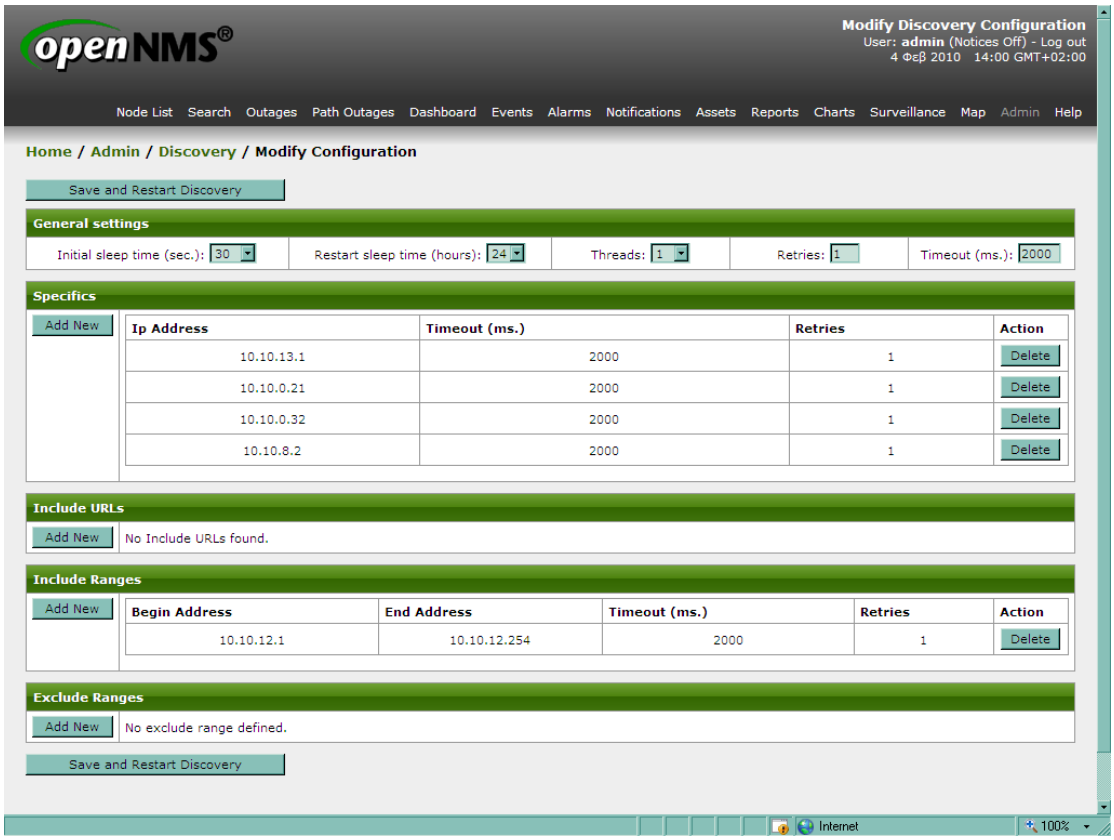
include-url

Εδώ μπορούμε να ορίσουμε ένα αρχείο το οποίο περιέχει τις IP διευθύνσεις που θέλουμε να ελέγξουμε.

<include-url>file:filename</include-url>

Όπου το “filename” είναι η διαδρομή για το αρχείο το οποίο μπορεί να περιέχει και σχόλια βάζοντας το σύμβολο # στην αρχή της γραμμής.

Ρυθμίσεις στο αρχείο discovery-configuration.xml όπως αναφέρθηκε παραπάνω, μπορούν να γίνουνε και μέσω του γραφικού περιβάλλοντος πηγαίνοντας στην κατηγορία Admin>Configure Discovery (Σχήμα 4.3)



Σχήμα 4.3 – Ρυθμίσεις ευρέσεως κόμβων

Μπορούμε να δούμε διάφορα λάθη και ειδοποιήσεις από την πορεία της διαδικασίας εύρεσης κόμβων εξετάζοντας το αρχείο discovery.log που βρίσκετε στον κατάλογο /usr/share/opennms/logs/daemons.

4.5. - Εύρεση των υπηρεσιών που παρέχονται από μία συγκεκριμένη IP διεύθυνση.

Εφόσον η διαδικασία της εύρεσης κόμβων ολοκληρωθεί, μετά αναλαμβάνει ο «δαίμονας» δυνατοτήτων ο οποίος ονομάζετε capsd. Αυτός ο «δαίμονας» είναι υπεύθυνος να βρει τις υπηρεσίες που θα παρακολουθούνται σε κάθε στοιχείο του δικτύου. Παραδείγματος χάρη ένα DNS, HTTPD, κ.τ.λ η εάν υπάρχει κάποια

διεργασία συλλογής δεδομένων όπως είναι το SNMP το οποίο είναι το μόνο που υποστηρίζετε προς το παρόν από το OpenNMS.

Ο capsd ρυθμίζετε μέσα από το αρχείο **capsd-configuration.xml** (βρίσκετε στο κατάλογο /usr/share/opennms/etc/) το οποίο περιέχει κάποιες βασικές παραμέτρους και μία συλλογή πρωτοκόλλων-υπηρεσιών όπου το OpenNMS θα προσπαθήσει να ανιχνεύσει. Εάν το πρωτόκολλο δεν υπάρχει μέσα στο αρχείο το OpenNMS δεν θα το ανιχνεύσει. Παρακάτω θα εξετάσουμε κάποιες βασικές παραμέτρους του αρχείου και μετά θα εξετάσουμε τα πρωτόκολλα:

Γενικές παράμετροι

```
<capsd-configuration
    rescan-frequency="86400000"
    initial-sleep-time="30000"
    max-suspect-thread-pool-size="6"
    max-rescan-thread-pool-size="3">
```

rescan-frequency

Ο “δαίμονας” capsd θα συνεχίσει να ελέγχει περιοδικά τους κόμβους του δικτύου ώστε να βρει νέες υπηρεσίες που έχουν προστεθεί. Η τιμή αυτής της παραμέτρου ορίζει το χρονικό διάστημα μεταξύ δύο ελέγχων. Η προεπιλεγμένη τιμή είναι 86400000 σε millisecond που ισούται με ένα εικοσιτετράωρο.

initial-sleep-time

Όπως και με την διεργασία ευρέσεως κόμβων η τιμή αυτή καθορίζει τον χρόνο που θα περιμένει η διεργασία μετά την εκκίνηση του OpenNMS πριν αρχίσει να εκτελείτε. Η προεπιλεγμένη τιμή είναι 30000 σε millisecond δηλαδή πέντε λεπτά.

management-policy

Η παράμετρος αυτή ορίζει την προεπιλεγμένη συμπεριφορά του capsd. Εάν έχει την τιμή “managed” τότε όλοι οι κόμβοι που εντοπιστήκαν από τον discoveryd θα ελέγχουνε για τις υπηρεσίες-πρωτόκολλα που παρέχουνε εκτός από αυτούς οι οποίοι βρίσκονται μέσα σε ετικέτες unmanagement στο τέλος του αρχείου. Εάν έχει την τιμή “unmanaged” τότε δεν θα γίνει ο έλεγχος για υπηρεσίες σε όλους τους κόμβους εκτός από αυτούς οι οποίοι βρίσκονται μέσα σε ετικέτες management στο τέλος του αρχείου. Μέσα στις ετικέτες μπορούμε να δηλώσουμε και εύρη

διευθύνσεων. Στην υλοποίηση μας θέλουμε να γίνετε έλεγχος σε όλους τους κόμβους οπότε αφήσαμε την προεπιλεγμένη τιμή που είναι η “management”.

max-suspect-thread-pool-size

Η τιμή αυτή ορίζει πόσα νήματα θα δημιουργηθούν από τον capsd για την εύρεση υπηρεσιών. Αυξάνοντας αυτή την τιμή έχει ως αποτέλεσμα να επιταχύνουμε την διεργασία εύρεσης υπηρεσιών χρησιμοποιώντας με κόστος τους πόρους του συστήματος.

max-rescan-thread-pool-size

Η τιμή αυτή ορίζει πόσα νήματα θα δημιουργηθούν από τον capsd για την εύρεση υπηρεσιών σε κόμβους οι οποίοι είναι ήδη γνωστοί. Ο επανέλεγχος εκτελείτε είτε κάθε “rescan-frequency” είτε χειροκίνητα μέσω του web interface.

abort-protocol-scans-if-no-route

Όταν το OpenNMS επιχειρεί να συνδεθεί σε ένα TCP port ώστε να ελέγξει μια υπηρεσία, υπάρχει η περίπτωση να λάβουμε μια εξαίρεση “no route to host”. Αυτή η εξαίρεση θεωρητικά μπορεί να σημαίνει ότι ο κόμβος είναι απροσπέλαστος αλλά μπορεί και να είναι πίσω από ένα firewall. Εάν η τιμή της μεταβλητής είναι true τότε με το που δημιουργηθεί η εξαίρεση η διεργασία περνάει στο έλεγχο του επόμενου κόμβου χωρίς να ελέγξει τις επόμενες υπηρεσίες. Εάν είναι false οι εξαιρέσεις “no route to host” θα αγνοούνται.

4.6. - Παράμετροι πρωτοκόλλων

Το OpenNMS ελέγχει για την παρουσία δικτυακών υπηρεσιών μέσα από την χρήση διαφόρων πρωτοκόλλων. Η πιο βασική περίπτωση είναι να συνδεθεί σε ένα συγκεκριμένο TCP port ώστε να λάβει ένα συγκεκριμένο μήνυμα – αλφαριθμητικό που αποδεικνύει την ύπαρξη της συγκεκριμένης υπηρεσίας. Παραδείγματος χάρη άμα πάρει ένα συγκεκριμένο μήνυμα από το TCP port 80 καταλαβαίνει ότι στο κόμβο που γίνετε ο έλεγχος υπάρχει ένας εξυπηρετητής ιστού. Βέβαια υπάρχουν και άλλες περιπτώσεις όπου η διαδικασία είναι διαφορετική για μια ποικιλία πρωτοκόλλων. Όταν ένα γεγονός NewSuspect παραλαμβάνετε από τον capsd και η IP διεύθυνση του ανήκει μέσα στο εύρος του management-policy τότε αρχίζει να εξετάζει τα πρωτόκολλα το ένα μετά το άλλο με την σειρά που βρίσκονται στο αρχείο. Παρακάτω θα παρουσιάσουμε μια λίστα με τα άμεσα υποστηριζόμενα πρωτόκολλα(χωρίς να χρειάζεται να εισάγουμε η να

εγκαταστήσουμε κάποιο) καθώς και τις ρυθμίσεις - ιδιότητες κάποιων πολύ βασικών πρωτοκόλλων.

- Citrix
- DHCP
- DNS
- Domino IIOP
- FTP
- GeneralPurpose (script based)
- HTTP
- HTTPS
- ICMP
- IMAP
- JBOSS
- JDBC
- JDBC Stored Procedure
- JSR160
- K5
- LDAP
- Microsoft Exchange
- MX4J
- Notes HTTP
- NSClient (Nagios Agent)
- NRPE (Nagios Remote Plugin Executor)
- NTP
- POP3
- Radius
- MySQL
- SQLServer
- Oracle
- Postgres
- Router
- Dell-OpenManage
- HP Insight Manager
- SMB
- SMTP
- SNMP
- SSH
- TCP
- Windows Services (SNMP-based)

4.6.1. - ΠΡΩΤΟΚΟΛΛΟ ICMP

```
<protocol-plugin protocol="ICMP"
        class-name="org.opennms.netmgt.capsd.plugins.IcmpPlugin"
        scan="on">
        <property key="timeout" value="2000"/>
        <property key="retry" value="2"/>
</protocol-plugin>
```

Κάθε πρωτόκολλο ξεκινά με την ετικέτα protocol-plugin η οποία περιέχει τέσσερις βασικές παραμέτρους.

protocol

Το όνομα του πρωτοκόλλου.

class-name

Ορίζει την java κλάση η οποία θα χρησιμοποιηθεί για να ελέγξει την υπηρεσία.

scan

Εδώ μπορούμε να ελέγξουμε την εύρεση υπηρεσιών ανά πρωτόκολλο. Στην τιμή off αποκλείουμε το συγκεκριμένο πρωτόκολλο από την εύρεση υπηρεσιών.

Κάθε protocol-plugin μπορεί να έχει ένα αριθμό ιδιοτήτων που ορίζονται σαν key και value. Ανάλογα με το πρωτόκολλο μπορούμε να έχουμε και διαφορετικές ιδιότητες αν και σχεδόν όλα έχουνε τις ιδιότητες timeout και retries, οπού είναι ο αριθμός των προσπαθειών που θα γίνουν για να ανοίξει μια σύνδεση. Μπορούμε ακόμα σε διαφορετικά εύρη IP διευθύνσεων να ορίσουμε διαφορετική συμπεριφορά στον έλεγχο του πρωτοκόλλου. Ακολουθεί ένα παράδειγμα για το ICMP:

```
<protocol-plugin protocol="ICMP"
   class-name="org.opennms.netmgt.capsd.plugins.IcmpPlugin"
    scan="on" user-defined="false">

   <protocol-configuration scan="on" user-defined="false">
        <range begin="192.168.10.0" end="192.168.10.254"/>
        <property key="timeout" value="4000"/>
        <property key="retry" value="3"/>
```

```
    </protocol-configuration>

    <protocol-configuration scan="off" user-defined="false">
        <range begin="192.168.20.0" end="192.168.20.254"/>
    </protocol-configuration>

    <property key="timeout" value="2000"/>
    <property key="retry" value="2"/>
</protocol-plugin>
```

Με την ετικέτα protocol-configuration μπορούμε όπως είπαμε και παραπάνω να ορίσουμε για διαφορετικά εύρη διευθύνσεων διαφορετικές παραμέτρους. Στο παράδειγμα φαίνεται ότι στο εύρος <range begin="192.168.10.0" end="192.168.10.254"/> έχουμε μεγάλο timeout και περισσότερα retries που θα μπορούσε να ανήκει σε ένα δίκτυο με μεγάλο χρόνο απόκρισης, ενώ στο εύρος <range begin="192.168.20.0" end="192.168.20.254"/> θέτοντας την παράμετρο scan=”off” απενεργοποιούμε την διαδικασία εύρεσης της συγκεκριμένης υπηρεσίας από αυτό.

4.6.2. - ΠΡΩΤΟΚΟΛΛΟ HTTP

```
<protocol-plugin protocol="HTTP"
class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin" scan="on">
        <property key="port" value="80" />
        <property key="timeout" value="3000" />
        <property key="retry" value="1" />
 </protocol-plugin>
```

Οι ιδιότητες του plugin για την εύρεση του HTTP πρωτόκολλου είναι ίδιες σαν το παράδειγμα που δόθηκε πριν με το ICMP εκτός από μία που έχει προστεθεί η οποία είναι το key=”port” value=”80”. Η συγκεκριμένη ιδιότητα λέει στο procotol-plugin να ψάξει στο συγκεκριμένο port ώστε να βρει εάν είναι ενεργή μια υπηρεσία η οποία επικοινωνεί μέσω του HTTP πρωτοκόλλου.

Τα procotol-plugin είναι ο κώδικας που εκτελείτε για να ελέγξει εάν υπάρχει μια συγκεκριμένη υπηρεσία, δεν είναι μία υπηρεσία. Ακολουθεί ένα παράδειγμα πως μπορούμε με το ίδιο plugin αλλά με διαφορετικές ρυθμίσεις να ελέγξουμε δύο διαφορετικές υπηρεσίες.

```
<protocol-plugin protocol="HTTP"
class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin"
scan="on">
        <property key="port" value="80" />
        <property key="timeout" value="3000" />
        <property key="retry" value="1" />
</protocol-plugin>

<protocol-plugin protocol="HTTP-8080"
class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin"
scan="on">
        <property key="port" value="8080" />
        <property key="timeout" value="3000" />
        <property key="retry" value="1" />
</protocol-plugin>
```

Αλλάζοντας το value του port μπορούμε να ελέγχουμε δύο διαφορετικές υπηρεσίες που μπορούν να επικοινωνήσουν μέσω του HTTP πρωτόκολλου αλλά περιμένουν να ανοίξει μια σύνδεση σε διαφορετικά port.

4.6.3. - ΠΡΩΤΟΚΟΛΛΟ SSH και TCP plugin

Ένα από τα ποιο χρήσιμα plugins είναι το TCP. Ακολουθεί ένα παράδειγμα για την εύρεση της υπηρεσίας SSH.

```
<protocol-plugin protocol="SSH"
class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin" scan="on"
user-defined="false">
    <property key="banner" value="SSH"/>
    <property key="port" value="22"/>
```

```
        <property key="timeout" value="3000"/>
        <property key="retry" value="3"/>
</protocol-plugin>
```

Βλέπουμε ότι μπορούμε να ορίσουμε μία ιδιότητα με όνομα banner και ως τιμή της ένα αλφαριθμητικό. Έτσι θα γίνει μια προσπάθεια σύνδεσης στην πόρτα 22 και αν η σύνδεση είναι επιτυχής θα ελεγχθεί αν οι χαρακτήρες του αλφαριθμητικού που ορίσαμε εμπεριέχονται σε αυτούς που επιστραφήκαν κατά την σύνδεση. Χρησιμοποιώντας την ιδιότητα banner μπορούμε να ορίσουμε πολλές διαφορετικές υπηρεσίες προς εύρεση αλλάζοντας την τιμή του αλφαριθμητικού.

4.6.4. - ΠΡΩΤΟΚΟΛΛΟ SNMP

Το πρωτόκολλο SNMP είναι μια ειδική περίπτωση γιατί εκτός από τον έλεγχο εάν η υπηρεσία είναι διαθέσιμη χρησιμοποιείτε και για την συλλογή δεδομένων. Παρουσιάζετε ένα παράδειγμα των ρυθμίσεων.

```
<protocol-plugin protocol="SNMP"
class-name="org.opennms.netmgt.capsd.plugins.SnmpPlugin" scan="on">
        <property key="timeout" value="2000" />
        <property key="retry" value="1" />
</protocol-plugin>
```

Από ότι βλέπουμε στις ρυθμίσεις εκτός από το ότι ορίζουμε το SnmpPlugin να βρει την υπηρεσία οι άλλες ιδιότητες είναι γνωστές σε όλα τα πρωτόκολλα. Κάτι που δεν ορίζετε εδώ είναι η έκδοση του Snmp που θα χρησιμοποιήσει το plugin. Μπορούμε να την ορίσουμε με την ιδιότητα “force version” προσθέτοντας την γραμμή:

```
        <property key="force version" value="SNMPv1"/>.
```

Έτσι θα βρούμε και τους SNMP agents οι οποίοι χρησιμοποιούν την 2η έκδοση γιατί οι δύο εκδόσεις είναι συμβατές στον συγκεκριμένο έλεγχο. Εάν βέβαια θέλουμε να δημιουργήσουμε μια νέα υπηρεσία αποκλειστικά για την 2η έκδοση του SNMP μπορούμε προσθέτοντας τις παρακάτω ρυθμίσεις.

```
<protocol-plugin protocol="SNMPv2"
 Class name="org.opennms.netmgt.capsd.plugins.SnmpPlugin"
  scan="on" user-defined="false">
        <property key="force version" value="SNMPv2"/>
        <property key="timeout" value="2000"/>
        <property key="retry" value="3"/>
</protocol-plugin>
```

Αυτή η ρύθμιση δεν έχει να κάνει με το πώς θα συλλέγονται τα δεδομένα αλλά με το να βρίσκει εάν υπάρχει κάποιος SNMP agent σε ένα κόμβο.
Οι παράμετροι που χρησιμοποιούνται για να συνδεθεί με έναν SNMP agent το OpenNMS βρίσκονται σε ένα ξεχωριστό αρχείο το οποίο λέγετε snmp-config.xml.

Όταν ο «δαίμονας» capsd ελέγχει για την υπηρεσία SNMP κάνει την προσπάθεια να λάβει και την τιμή του sysObjectID από τη συσκευή χρησιμοποιώντας σαν όνομα του community και σαν πόρτα ότι είναι ορισμένο στο αρχείο snmp-config.xml. Έάν επιτύχει το SNMP πρωτόκολλο το μαρκάρει σαν «αληθές» για την συγκεκριμένη IP-διεύθυνση. Το όνομα του community που θα χρησιμοποιήσει είναι το πρώτο που θα βρει μέσα στο αρχείο snmp-config.xml με βάση την IP διεύθυνση, ασχέτως αν η διεύθυνση είναι δηλωμένη και σε παρακάτω σημεία του αρχείου.

Μόλις ο «δαίμονας» τελειώσει με τον έλεγχο των πρωτοκόλλων στη συνέχεια ελέγχει εάν κάποια IP διεύθυνση έχει μαρκαριστεί σαν «αληθής» για το SNMP και αν υπάρχει προχωρά στους εξής ελέγχους:

1) Στέλνονται τρία SNMP-request με σκοπό να συλλέξουν το system tree, τον ipAddrTable και τον ifTable. Εάν για κάποιο λόγω οι πίνακες ipAddrTable, ifTable δεν είναι διαθέσιμοι η διαδικασία σταματά. Αυτό μπορεί να συμβεί σε UC-Davis SNMP agent όπου μόνο το system tree είναι διαθέσιμο χρησιμοποιώντας το “public” community.

2) Για όλες τις IP διευθύνσεις που βρίσκει μέσα στον πίνακα ipAddrTable ελέγχει μέσω του capsd για υπηρεσίες που μπορεί να είναι διαθέσιμες. Αυτό γίνετε χωρίς

να λαμβάνετε υπόψη η ιδιότητα managed του αρχείου. Αυτό συμβαίνει στον αρχικό έλεγχο και σε κάθε αναγκαστικό επανέλεγχο. Στους φυσιολογικούς επανελέγχους (προκαθορισμένο στις 24 ώρες) δεν ελέγχονται οι IP διευθύνσεις που έχουν οριστεί από το διαχειριστή σαν “unmanaged”.

3) Για όλες τις IP διευθύνσεις που βρίσκει μέσα στον πίνακα ipAddrTable οι οποίες υποστηρίζουνε το πρωτόκολλο SNMP ελέγχει εάν αντιστοιχίζετε με ένα έγκυρο ifindex στον πίνακα iftable. Εάν βρει αντιστοιχία τότε σημαδεύει την IP διεύθυνση σαν δευτερεύον SNMP interface και είναι υποψήφια ώστε να γίνει το βασικό SNMP interface.

Τέλος, όλες οι δευτερεύουσες SNMP διασύνδεσεις ελέγχονται εάν ταιρίαζουν σε κάποιο έγκυρο package του αρχείου collectd-configuration.xml (collectd είναι ο «δαίμονας» του OpenNMS που συλλέγει στοιχεία μέσω του SNMP). Εάν πάνω από μία IP διευθύνσεις τηρούν τα τρία κριτήρια(υποστηρίζουν SNMP, έχουν έγκυρο ifindex και εμπεριέχονται σε package του collectd), τότε η μικρότερη IP διεύθυνση επιλέγετε σαν βασική. Όλη η συλλογή πληροφοριών μέσω του SNMP γίνετε από τη βασική IP διεύθυνση.

Όταν ο έλεγχος του capsd ολοκληρωθεί, κάποια γεγονότα παράγονται όπως το NodeGainedService γεγονός. Μπορούμε να δούμε διάφορα λάθη και ειδοποιήσεις από την πορεία της διαδικασίας εύρεσης δικτυακών υπηρεσιών εξετάζοντας το αρχείο capsd.log που βρίσκετε στον κατάλογο /usr/share/opennms/logs/daemons.

4.7. - Επίλογος

Στο κεφάλαιο αυτό παρουσιάστηκε η αρχιτεκτονική του OpenNMS και αναφερθήκαμε στην δυνατότητα του να βρίσκει αυτόματα τις συσκευές που θέλουμε να διαχειριστούμε μέσω του δαίμονα discoveryd και πώς ορίζουμε εύρη διευθύνσεων προς εύρεση. Μετά αναφερθήκαμε στον δαίμονα capsd ο οποίος είναι υπεύθυνος για την εύρεση των υπηρεσιών οι οποίες παρέχονται από ένα διαχειριζόμενο κόμβο-συσκευή και αναλύσαμε το αρχείο ρυθμίσεών του καθώς και το πώς μπορούμε να τροποποιήσουμε ή να προσθέσουμε νέες υπηρεσίες προς εύρεση.

ΚΕΦΑΛΑΙΟ 5

Ενεργός έλεγχος της κατάστασης των στοιχείων του δικτύου και των υπηρεσιών που παρέχουνε (Polling)

Υπάρχουν δύο βασικοί τρόποι με τους οποίους το OpenNMS συλλέγει δεδομένα από το δίκτυο. Ο πρώτος είναι μέσω του ενεργού ελέγχου (polling). Διεργασίες οι οποίες λέγονται ελεγκτές (monitors) συνδέονται σε πόρους του δικτύου και κάνουνε έναν έλεγχο ώστε να διαπιστωθεί εάν ο πόρος ανταποκρίνεται με το σωστό τρόπο. Αν δεν ανταποκρίνεται σωστά τότε δημιουργούνται κάποια γεγονότα. Ο δεύτερος τρόπος συλλογής δεδομένων είναι μέσω του πρωτοκόλλου SNMP μέσα από διεργασίες οι οποίες λέγονται συλλογείς (collectors).

Το σκεπτικό πίσω από την διαδικασία του ενεργού ελέγχου είναι να χωρίζουμε τις δικτυακές συσκευές σε πακέτα όπου στο κάθε πακέτο μπορούμε να έχουμε διαφορικές ρυθμίσεις. Κάθε πακέτο μπορεί να περιλαμβάνει τον έλεγχο διαφορετικών υπηρεσιών ή διαφορετική συχνότητα που θα γίνετε ο έλεγχος. Αν παρουσιαστεί μια διακοπή σε κάποια υπηρεσία το κάθε πακέτο μπορεί να έχει το δικό του μοντέλο στο πώς θα αντιδράσει ο ενεργός έλεγχος δηλαδή αν θα ξαναελέγχει άμεσα, κάθε πότε και δυναμικά να αλλάξουν οι χρόνοι του ελέγχου. Τέλος κάθε πακέτο έχει ένα ημερολόγιο προγραμματισμένων διακοπών όπου μπορούμε να ορίσουμε χρονικά διαστήματα όπου ο έλεγχος θα είναι ανενεργός και την δικιά του βάση δεδομένων Round Robin όπου μπορούμε να αποθηκεύουμε τον χρόνο απόκρισης των υπηρεσιών.

Ο ενεργός έλεγχος εφαρμόζεται μόνο σε interfaces και υπηρεσίες οι οποίες ευρέθηκαν από τον «δαίμονα» capsd που παρουσιάστηκε προηγουμένως.
Όλη η διαδικασία του ενεργού ελέγχου ρυθμίζετε από τον «δαίμονα» Pollerd και το αρχείο τον ρυθμίσεων του ονομάζετε poller-configuration.xml (βρίσκετε στο κατάλογο /usr/share/opennms/etc/). Παρακάτω θα εξετάσουμε κάποιες βασικές παραμέτρους του αρχείου.

5.1. - Γενικές Μεταβλητές

```
<poller-configuration threads="30" serviceUnresponsiveEnabled="false">
      <node-outage status="on"
                  pollAllIfNoCriticalServiceDefined="true">
            <critical-service name="ICMP"/>
      </node-outage>
```

poller-configuration threads

Καθορίζει το μέγιστο αριθμό νημάτων που θα χρησιμοποιηθούν από το εικονικό περιβάλλον της JAVA για την διεργασία του ενεργού ελέγχου. Ρυθμίζετε ανάλογα με την επεξεργαστική ισχύ του εξυπηρετητή που είναι εγκατεστημένο το OpenNMS και ανάλογα το μέγεθος του δικτύου. Στην συγκεκριμένη υλοποίηση χρησιμοποιήθηκε η εξ' ορισμού τιμή η οποία είναι 30.

serviceUnresponsiveEnabled

Ο ενεργός έλεγχος είναι στην ουσία μια σύνδεση σε κάποια συγκεκριμένη πόρτα ενός interface όπου ελέγχει εάν η υπηρεσία σε αυτή την πόρτα επιστρέφει την αναμενόμενη απόκριση. Εάν η ανταπόκριση δεν ληφθεί μέσα σε ένα χρονικό διάστημα(παράμετρος timeout) η υπηρεσία θεωρείτε ανενεργή. Σε κάποια δίκτυα όμως κάποιες ασυνεχής αποτυχίες σύνδεσης είναι συνηθισμένες. Η παράμετρος που παρουσιάζετε ορίζει την εξής συμπεριφορά. Εάν είναι true τότε θεωρεί μια υπηρεσία στην πόρτα πεσμένη στην περίπτωση που η σύνδεση της αποτύχει και όχι αν δεν πάρει την απόκριση της υπηρεσίας μέσα στο επιτρεπόμενο χρονικό περιθώριο, οπού σε αυτή την περίπτωση δημιουργεί απλά ένα γεγονός "service unresponsive". Εάν είναι false τότε και για τις 2 περιπτώσεις δημιουργεί το ίδιο γεγονός το οποίο είναι το "NodeLostService".

node-outage

Το βασικό γεγονός το οποίο δημιουργείται όταν ένας έλεγχος αποτυγχάνει ονομάζετε NodeLostService. Εάν πάνω από μία υπηρεσίες δεν είναι προσβάσιμες πολλαπλά NodeLostService γεγονότα θα δημιουργηθούνε.

Εάν όλες οι υπηρεσίες σε ένα interface δεν είναι προσβάσιμες τότε έχουμε ένα γεγονός InterfaceDown και αν όλα τα interface ενός κόμβου είναι «πεσμένα» τότε ο κόμβος θεωρείτε μη προσβάσιμος. Όταν ένα γεγονός NodeDown προκύψει και η παράμετρος node-outage=”on” τότε όλα τα γεγονότα τύπου

NodeLostService και InterfaceDown θα αποσιωπηθούν και μόνο το γεγονός NodeDown θα δημιουργηθεί. Αντί ο επανέλεγχος σε ένα «πεσμένο» κόμβο να γίνει σε όλες τις υπηρεσίες μπορούμε να ορίσουμε μία υπηρεσία σαν κρίσιμη που αν δεν επανακτηθεί η λειτουργία της και η παράμετρος pollAllIfNoCriticalServiceDefined="false" ο ενεργός έλεγχος δεν συνεχίζει στις υπόλοιπες υπηρεσίες. Αν η παράμετρος pollAllIfNoCriticalServiceDefined="true" τότε θα ελεγχθεί η πρώτη υπηρεσία (εκτός της κρίσιμης) και μέχρι να επανακτηθεί η λειτουργία της ο έλεγχος θα σταματά εκεί.

5.2. - Πακέτα Ενεργού Ελέγχου (Poller Packages)

Ένα πακέτο αποτελείται από ένα όνομα, μια ομάδα από interfaces που θα ελέγχονται και τις υπηρεσίες που θα ελέγχονται στην ομάδα των interfaces.
Μπορούνε να υπάρχουνε πολλαπλά πακέτα και ένα interface μπορεί να υπάρχει σε περισσότερα από ένα. Αυτό μας δίνει μια ευελιξία ώστε να διαβαθμίσουμε τις ανάγκες του ελέγχου που θα γίνετε σε διάφορες συσκευές. Για παράδειγμα μπορούμε να ορίσουμε δύο διαφορετικά πακέτα ένα υψηλής σημασίας και ένα χαμηλής. Στο υψηλής σημασίας ο έλεγχος μπορεί να γίνετε κάθε ένα λεπτό ενώ στο χαμηλής κάθε πέντε λεπτά. Μπορούμε επίσης να ορίσουμε τον έλεγχο διαφορετικών υπηρεσιών στο κάθε πακέτο διαφορετικό “downtime” μοντέλο και “scheduled outages”.
Ο ορισμός ενός πακέτου ξεκινά με την ετικέτα του:

<package name="example1">

Και ακολουθείτε από μια λίστα ετικετών που ορίζουνε ποια interfaces (IP διευθύνσεις) θα συμπεριλαμβάνονται στο πακέτο.

Filter

IPADDR IPLIKE *.*.*.*

Κάθε πακέτο πρέπει να έχει μία ετικέτα filter που κάνει τον αρχικό έλεγχο ώστε να διαχωρίσει τα interfaces με βάση την IP διεύθυνση τους που θα συμπεριληφθούν στο πακέτο. Κάθε πακέτο μπορεί να έχει μόνο ένα φίλτρο.

specific

Εδώ ορίζουμε μια IP διεύθυνση που θα περιλαμβάνετε στο πακέτο. Μπορούμε να έχουμε πολλές <specific> ετικέτες.

<specific>*ip-address*</specific>

Include-range

Εδώ ορίζουμε ένα εύρος διευθύνσεων που θα περιλαμβάνετε στο πακέτο. Μπορούμε να έχουμε πολλές <include-range> ετικέτες.

<include-range begin=*start-ip-address* end=*end-ip-address*</include-range>

exclude-range

Εδώ ορίζουμε ένα εύρος διευθύνσεων που θα δεν θέλουμε να περιλαμβάνετε στο πακέτο. Μπορούμε να έχουμε πολλές <exclude-range> ετικέτες.

<exclude-range begin=*start-ip-address* end=*end-ip-address*</exclude-range>

include-url

Εδώ μπορούμε να ορίσουμε ένα αρχείο το οποίο περιέχει τις IP διευθύνσεις που θέλουμε να περιλαμβάνονται στο πακέτο.

<include-url>file:filename</include-url>

Όπου το “filename” είναι η διαδρομή για το αρχείο το οποίο μπορεί να περιέχει και σχόλια βάζοντας το σύμβολο # στην αρχή της γραμμής.

5.3. - Αποθήκευση του χρόνου απόκρισης των υπηρεσιών σε Βάση δεδομένων Round Robin

```
<rrd step="300">
    <rra>RRA:AVERAGE:0.5:1:2016</rra>
    <rra>RRA:AVERAGE:0.5:12:1488</rra>
    <rra>RRA:AVERAGE:0.5:288:366</rra>
    <rra>RRA:MAX:0.5:288:366</rra>
    <rra>RRA:MIN:0.5:288:366</rra>
  </rrd>
```

Με τις παραπάνω δηλώσεις δημιουργούμε μια βάση δεδομένων τύπου Round Robin όπου μπορούμε να αποθηκεύουμε τον χρόνο απόκρισης των υπηρεσιών.

<rrd step="300">

Το βήμα είναι 300 δευτερόλεπτα δηλαδή πέντε λεπτά.

<rra>RRA:AVERAGE:0.5:1:2016</rra>

Δημιουργείτε ένα αρχείο που αποθηκεύει τον μέσο όρο μία τιμής(δεν υπάρχει μέσος όρος δηλαδή) κάθε 5 λεπτά και περιέχει τις τελευταίες 2016 τιμές(1 εβδομάδα)

<rra>RRA:AVERAGE:0.5:12:1488</rra>

Δημιουργείτε ένα αρχείο που αποθηκεύει τον μέσο όρο των τελευταίων 12 τιμών δηλαδή τον μέσο όρο της τελευταίας ώρας και περιέχει τις τελευταίες 1488 τιμές(62 ημέρες)

<rra>RRA:AVERAGE:0.5:288:366</rra>

Δημιουργείτε ένα αρχείο που αποθηκεύει τον μέσο όρο των τελευταίων 288 τιμών δηλαδή τον μέσο όρο της τελευταίας ημέρας και περιέχει τις τελευταίες 366 τιμές(366 ημέρες)

<rra>RRA:MAX:0.5:288:366</rra>

Δημιουργείτε ένα αρχείο που αποθηκεύει την μέγιστη τιμή των τελευταίων 288 τιμών δηλαδή την μέγιστη τιμή της τελευταίας ημέρας και περιέχει τις τελευταίες 366 τιμές(366 ημέρες)

<rra>RRA:MIN:0.5:288:366</rra>

Δημιουργείτε ένα αρχείο που αποθηκεύει την ελάχιστη τιμή των τελευταίων 288 τιμών δηλαδή την ελάχιστη τιμή της τελευταίας ημέρας και περιέχει τις τελευταίες 366 τιμές(366 ημέρες)

Όλα τα αρχεία τύπου rra που δημιουργήσαμε παραπάνω αποθηκεύονται στην ίδια βάση δεδομένων στην οποία μπορούμε να αποθηκεύσουμε τον χρόνο απόκρισης μιας υπηρεσίας.

5.4. - Υπηρεσίες Ενεργού ελέγχου (Poller Services)

Εφόσον ορίσουμε τα interfaces (IP διευθύνσεις) που θέλουμε να εμπεριέχονται στο πακέτο μετά ορίζουμε τις υπηρεσίες οι οποίες θα ελέγχονται. Παρακάτω θα δοθούν δύο παραδείγματα για το πώς ορίζουμε υπηρεσίες ενώ το πλήρες αρχείο εμφανίζετε στην ενότητα 12.4.

ICMP

```
<service name="ICMP" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="2" />
    <parameter key="timeout" value="3000" />
```

```
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="icmp" />
    <parameter key="ds-name" value="icmp" />
</service>
```

name

Το όνομα της υπηρεσίας

interval

η συχνότητα του ελέγχου σε milliseconds

retry

ο αριθμός των προσπαθειών επανασύνδεσης αν η πρώτη φορά αποτύχει

timeout

ο χρόνος αναμονής για να ληφθεί ανταπόκριση από την υπηρεσία

rrd-repository

Η τοποθεσία αποθήκευσης του αρχείου rrd

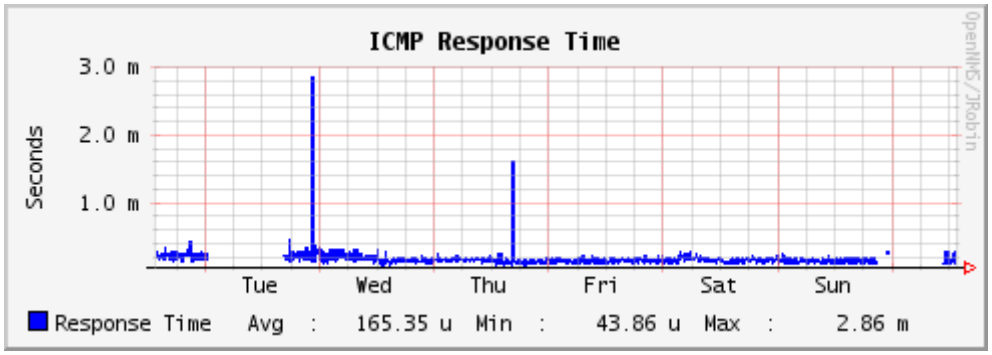
rrd-base-name

το όνομα του rrd αρχείου(χωρίς την κατάληξη .rrd ή .jrd)

ds-name

το όνομα της πηγής της πληροφορίας

Στο συγκεκριμένο παράδειγμα έχουμε τον έλεγχο της υπηρεσίας ICMP δηλαδή κάνουμε ένα ping στη συσκευή κάθε 5 λεπτά (300000 millisecond). Εάν αποτύχει η πρώτη φορά προσπαθεί άλλες δύο και για να θεωρηθεί επιτυχημένη η απόκριση πρέπει να γίνει μέσα σε χρόνο 3 δευτερολέπτων (3000 millisecond). Ο χρόνος απόκρισης της υπηρεσίας καταγράφετε σε ένα αρχείο τύπου rrd. Οι παράμετροι των υπηρεσιών δεν είναι σε όλες ίδιες και η καταγραφή του χρόνου απόκρισης συνήθως χρησιμοποιείτε μόνο σε υπηρεσίες που είναι σημαντικό να τον γνωρίζουμε.



Σχήμα 5.1 – Διάγραμμα χρόνου απόκρισης ICMP μηνυμάτων

StrafePing

```
<service name="StrafePing" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="0" />
    <parameter key="timeout" value="3000" />
    <parameter key="ping-count" value="20" />
    <parameter key="failure-ping-count" value="20" />
    <parameter key="wait-interval" value="50" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="strafeping" />
  </service>
```

Η συγκεκριμένος έλεγχος εκτελεί πολλαπλά ICMP echo requests και αποθηκεύει τον χρόνο απόκρισης καθώς και το ποσοστό πακέτων που αποτυγχάνουν σε ένα αρχείο rrd. Συνήθως είναι σε ξεχωριστό πακέτο γιατί η χρήση του σε όλες τις συσκευές του δικτύου φορτώνει αρκετά το δίκτυο καθώς και την μεταγωγή δεδομένων στο δίσκο του OpenNMS. Οι παράμετροι που ανήκουν μόνο στο συγκεκριμένο έλεγχο είναι:

ping-count

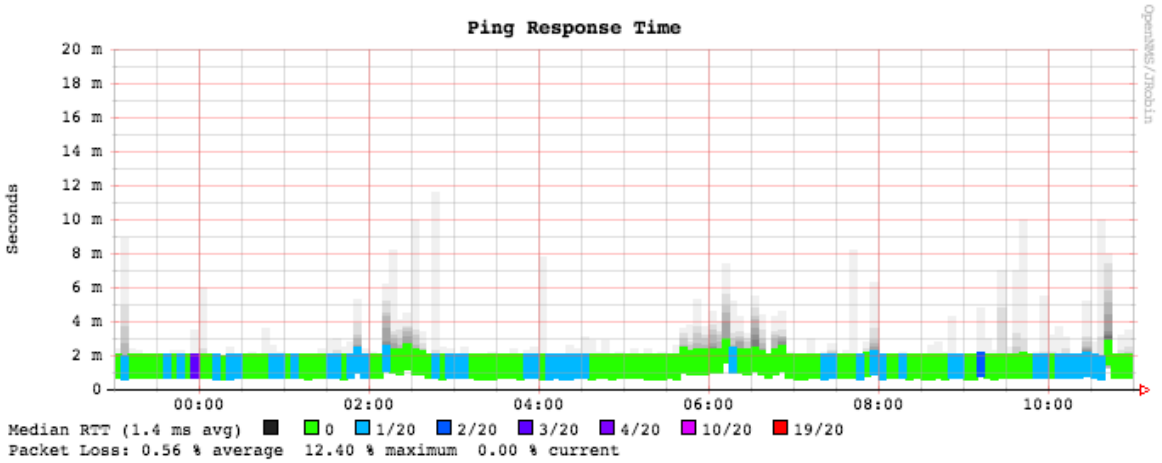
Ο αριθμός των ping που θα στέλνει κάθε χρονική περίοδο

failure-ping-count

Ο αριθμός των ping που θα αποτύχουν έτσι ώστε να θεωρηθεί η υπηρεσία ανενεργή

wait-interval

Ο χρόνος μεταξύ των ping μηνυμάτων



Σχήμα 5.2 – Διάγραμμα χρόνου απόκρισης πολλαπλών ICMP μηνυμάτων

5.5. - Μοντέλο Χρόνου Διακοπών (Downtime Model)

Ένα από τα ποιο σημαντικά χαρακτηριστικά του ενεργού ελέγχου είναι τα μοντέλα για τους χρόνους διακοπών των υπηρεσιών. Μας δίνει την δυνατότητα ο χρόνος που μεσολαβεί μεταξύ των ελέγχων να αλλάζει δυναμικά αν μια υπηρεσία δεν ανταποκρίνεται. Είναι προκαθορισμένο ο ενεργός έλεγχος να γίνετε κάθε πέντε λεπτά. Αν έχουμε διακοπή σε μια υπηρεσία ή συσκευή τότε η μικρότερη διάρκειά της θα είναι πέντε λεπτά μέχρι να ξαναελεγχεί όπου εκεί θα διαπιστώσουμε εάν επανήρθε η λειτουργία της. Οπότε εάν θέλουμε να έχουμε διαθεσιμότητα που να πλησιάζει το 100% μια διακοπή των πέντε λεπτών μπορεί να είναι αρκετά μεγάλη.

Για αυτό μέσα στο κάθε πακέτο μπορούμε να ορίσουμε ένα μοντέλο που να αλλάζει τους χρόνους του ελέγχου προσθέτοντας τα παρακάτω.

<downtime interval="30000" begin="0" end="300000" />
<!-- 30s, 0, 5m -->
Από τη στιγμή που μία διακοπή θα ξεκινήσει (begin=”0”) μέχρι πέντε λεπτά μετά (end="300000" ms) ο έλεγχος θα γίνετε κάθε τριάντα δευτερόλεπτα.

<downtime interval="300000" begin="300000" end="43200000" />
<!-- 5m, 5m, 12h -->
Μετά τα πέντε λεπτά και εφόσον η υπηρεσία ακόμα δεν είναι διαθέσιμη τότε ο έλεγχος ξαναγυρνά στην αρχική του συχνότητα (5 λεπτά) μέχρι να περάσουν δώδεκα ώρες(43200000 ms).

<downtime interval="600000" begin="43200000" end="432000000" />
<!-- 10m, 12h, 5d -->
Μετά της δώδεκα ώρες και εφόσον η υπηρεσία ακόμα δεν είναι διαθέσιμη η συχνότητα ελέγχου μειώνετε στα δέκα λεπτά μέχρι να περάσουν πέντε ημέρες (432000000 ms).

<downtime begin="432000000" delete="true" />
<!-- anything after 5 days delete -->
Τέλος εάν η υπηρεσία είναι μη διαθέσιμη μετά από πέντε ημέρες τότε διαγράφετε από το συγκεκριμένο interface. Η διαγραφή της δεν είναι απαραίτητη μπορούμε να συνεχίσουμε για όσο χρονικό διάστημα θέλουμε το έλεγχό της.

5.6. - Παρακολουθητές Ενεργού ελέγχου (Poller Monitors)

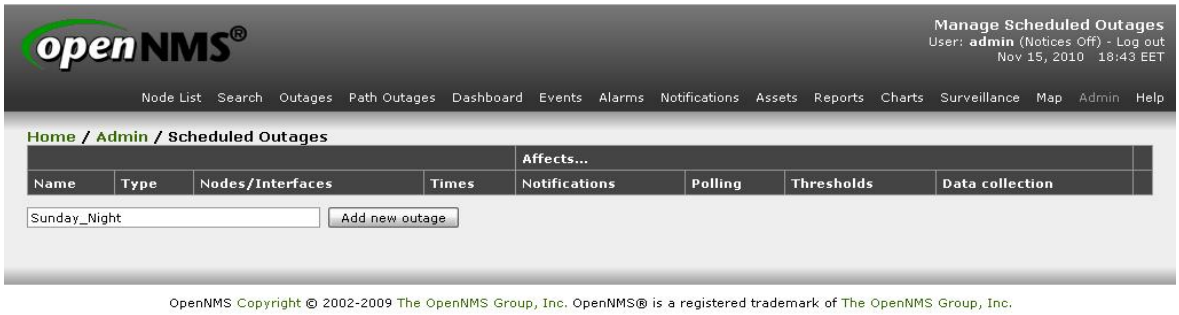
Για κάθε υπηρεσία που δηλώνουμε σε ένα πακέτο ενεργού ελέγχου πρέπει να δηλώσουμε και τον αντίστοιχο παρακολουθητή. Στον «δαίμονα» capsd που παρουσιάσαμε στο προηγούμενο κεφάλαιο η δήλωση συμπεριλαμβανότανε στην δήλωση της υπηρεσίας. Εδώ δηλώνετε στο τέλος του αρχείου μία φορά όσες και αν είναι οι φορές που η υπηρεσία δηλώθηκε μέσα στο αρχείο και σε όσα πακέτα και αν συμμετέχει. Η δήλωση των παρακολουθητών γίνετε ως εξής:

<monitor service="ICMP"
        class-name="org.opennms.netmgt.poller.IcmpMonitor"/>
<monitor service="HTTP"
        class-name="org.opennms.netmgt.poller.HttpMonitor"/>
<monitor service="HTTP-8080"
        class-name="org.opennms.netmgt.poller.HttpMonitor"/>

Εδώ βλέπουμε ότι ο παρακολουθητής "org.opennms.netmgt.poller.HttpMonitor" χρησιμοποιείτε από δύο υπηρεσίες οι οποίες έχουν διαφορετικές ρυθμίσεις. Στην μία ελέγχει για το πρωτόκολλο HTTP στη πόρτα 80 ενώ στην άλλη στη πόρτα 8080.

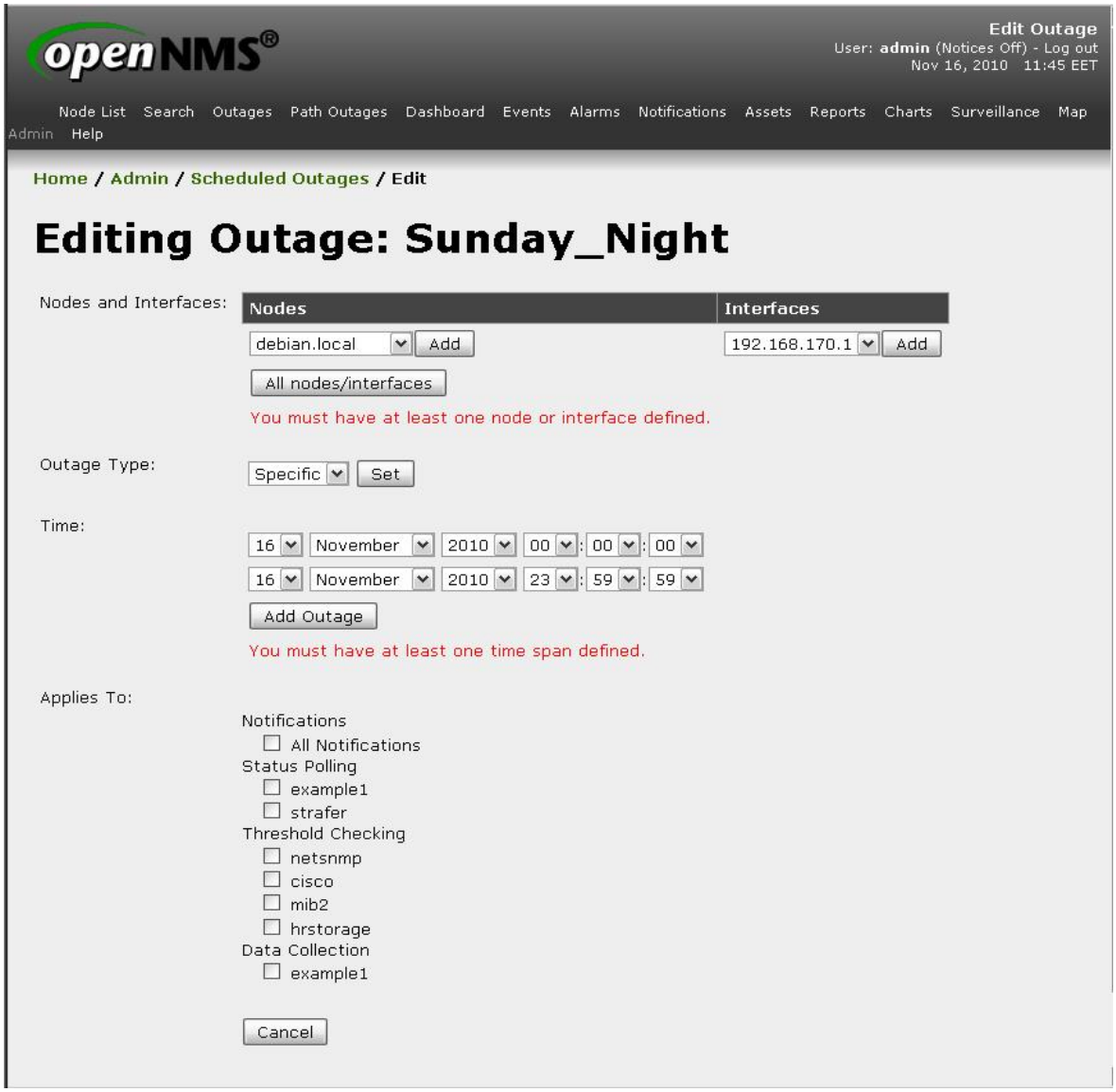
5.7. - Προγραμματισμένες Διακοπές (scheduled updates)

Για την σωστή λειτουργία των εξυπηρετητών όπως και πολλών άλλων δικτυακών συσκευών πολλές φορές είναι απαραίτητο να σταματήσουμε την λειτουργία τους για συντήρηση. Στο OpenNMS μπορούμε να δηλώσουμε ποιες χρονικές περίοδοι θα είναι αυτές ώστε να μην γίνει ο έλεγχος διαφόρων υπηρεσιών και συσκευών ώστε να μην καταγραφεί σαν διακοπή λειτουργίας. Ο ρυθμίσεις μπορούν να γίνουνε μέσα στο αρχείο poll-outages.xml όπως επίσης και μέσα από το γραφικό περιβάλλον πηγαίνοντας στην κατηγορία Admin>Scheduled Updates.



Σχήμα 5.3 – Δημιουργία προγραμματισμένης διακοπής

Στο Σχήμα 5.3 δηλώνουμε το όνομα μιας προγραμματισμένης διακοπής που θέλουμε να δημιουργήσουμε και πατώντας στο κουμπί “Add new outage” μεταβαίνουμε στο Σχήμα 5.4 για τις περαιτέρω ρυθμίσεις.



Σχήμα 5.4 – Ρυθμίσεις προγραμματισμένης διακοπής

Στο Σχήμα 5.4 βλέπουμε τις επιλογές που έχουμε για να δημιουργήσουμε μια προγραμματισμένη διακοπή. Στο Nodes and Interfaces μπορούμε να ορίσουμε ποιους κόμβους ή ποιες συγκεκριμένες επαφές θα περιλαμβάνει. Στο Outage Type αν θα είναι για μία συγκεκριμένη φορά, ημερήσια, εβδομαδιαία ή μηνιαία. Στο Time ορίζουμε την χρονική στιγμή που θα ξεκινήσει και πόσο θα διαρκέσει. Τέλος στο Applies to ορίζουμε ποια υποπρογράμματα «δαίμονες» του OpenNMS θα επηρεάσει.

Εδώ βλέπουμε και την λογική των πακέτων που διέπει το OpenNMS ο κάθε «δαίμονας» μπορεί να απαρτίζετε από πολλά πακέτα που το καθένα να έχει διαφορετικές ρυθμίσεις. Εδώ βλέπουμε το Status Polling (Pollerd) να περιέχει δύο πακέτα το example και το strafer. Το Data Collection είναι ο «δαίμονας» collectd,

το Notifications είναι ο notifd ενώ το Threshold Checking είναι ο Threshd. Έτσι δημιουργώντας μια προγραμματισμένη διακοπή μπορούμε όχι μόνο να απενεργοποιήσουμε ένα «δαίμονα» αλλά και ένα μέρος του που ορίζετε ως πακέτο.

5.8. – Επίλογος

Στο κεφάλαιο αυτό αναφερθήκαμε στο πως λειτουργεί ο ενεργός έλεγχος του OpenNMS για την κατάσταση των στοιχείων και των υπηρεσιών του δικτύου. Είδαμε πώς μπορούμε να αποθηκεύουμε σαν διάγραμμα των χρόνο απόκρισης των υπηρεσιών καθώς και πώς μπορούμε να χωρίσουμε τις υπηρεσίες σε διαφορετικά πακέτα ώστε ο τρόπος με τον οποίο ελέγχονται να είναι διαφορετικός. Επίσης αναφέραμε το πώς δουλεύει το μοντέλο με το οποίο διαχειρίζεται τις διακοπές λειτουργίας σε μια υπηρεσία το OpenNMS, δηλαδή πως αλλάζει η συχνότητα ελέγχου σε περίπτωση που μια υπηρεσία βρεθεί ανενεργή. Τέλος αναφερθήκαμε στο πως μπορούμε να ορίσουμε κάποιες προγραμματισμένες διακοπές σε περίπτωση που θέλουμε εσκεμμένα να απενεργοποιήσουμε κάποια στοιχεία του δικτύου αλλά δεν θέλουμε να λάβουμε ειδοποιήσεις και γεγονότα καθώς και να μην επηρεαστούν τα στατιστικά δεδομένα λειτουργίας για αυτή την απενεργοποίηση.

ΚΕΦΑΛΑΙΟ 6

Συλλογή Πληροφοριών (Data Collection)

Ο πρώτος τρόπος συλλογής δεδομένων είναι μέσω του ενεργού ελέγχου ο οποίος αναφέρθηκε στο προηγούμενο κεφάλαιο. Σε αυτό το κεφάλαιο θα εξετάσουμε τον δεύτερο και ποιο βασικό τρόπο συλλογής δεδομένων μέσω διεργασιών οι οποίες ονομάζονται συλλογείς δεδομένων. Οι οποίες είναι:

- SNMP
- NSClient (the Nagios Agent)
- JMX
- HTTP

6.1. - SNMP

Για να ρυθμίσουμε την λειτουργία συλλογής δεδομένων μέσω του SNMP πρωτοκόλλου δύο αρχεία πρέπει να ρυθμιστούνε. Το **snmp-config.xml** και το **datacollection-config.xml**. Το **snmp-config.xml** θα παρουσιαστεί σε αυτή την ενότητα ενώ το **datacollection-config.xml** σε παρακάτω μιας και λόγω του πλήθους των παραμέτρων του αποτελεί ενότητα από μόνο του.

snmp-config.xml

```
<snmp-config retry="3" timeout="800" read-community="public" write-
community="private">
    <definition version="v2c">
        <specific>192.168.0.5</specific>
    </definition>
    <definition retry="4" timeout="2000">
        <range begin="192.168.1.1" end="192.168.1.254"/>
        <range begin="192.168.3.1" end="192.168.3.254"/>
    </definition>
    <definition read-community="bubba" write-community="zeke">
        <range begin="192.168.2.1" end="192.168.2.254"/>
```

```
    </definition>
    <definition port="1161">
        <specific>192.168.5.50</specific>
    </definition>
</snmp-config>
```

retry

Ο αριθμός των προσπαθειών που θα γίνουν για την σύνδεση στον SNMP agent

timeout

Το χρονικό διάστημα του σε milliseconds που το OpenNMS θα περιμένει για μια ανταπόκριση από τον SNMP agent.

read-community

Το προκαθορισμένο community αλφαριθμητικό για ερωτήματα SNMP που πρόκειται να διαβάσουν. Εάν δεν οριστεί, το προκαθορισμένο είναι “public”

write-community

Το προκαθορισμένο community αλφαριθμητικό για ερωτήματα SNMP που πρόκειται να γράψουν. Εάν δεν οριστεί, το προκαθορισμένο είναι “private”.

port

Δηλώνοντας κάποια άλλη πόρτα μπορούμε να αλλάξουμε την προκαθορισμένη που είναι η 161.

version

Μπορούμε να ορίσουμε την έκδοση του SNMP με τις ορίζοντας v1 ή v2c ή v3.
Η προκαθορισμένη είναι η 1.

Για την αυθεντικοποίηση και την συλλογή μέσω του SNMPv3:

security-name

Το όνομα ασφαλείας για την αυθεντικοποίηση του SNMPv3

auth-passphrase

Το σύνθηματικό για την αυθεντικοποίηση του SNMPv3

auth-protocol

Το πρωτόκολλο αυθεντικοποίησης του SNMPv3. MD5 ή SHA. Προκαθορισμένο είναι το MD5.

privacy-passphrase

Το μυστικό συνθηματικό που χρησιμοποιείτε για την κρυπτογράφηση των πακέτων του SNMPv3

privacy-protocol

Το μυστικό πρωτόκολλο που χρησιμοποιείτε για την κρυπτογράφηση των πακέτων του SNMPv3. Είτε "DES", "AES","AES192" ή "AES256". Προκαθορισμένο είναι το DES.

engine-id

Το engine id του SNMP agent

context-name

Το όνομα του περιεχομένου που θα παρθεί η πληροφορία από τον SNMP agent.

context-engine-id

Το context engine id της οντότητας στόχου στον SNMP agent.

enterprise-id

Το enterprise id για την SNMPv3 συλλογή

proxy-host

Ορισμός ενός proxy για την επικοινωνία με SNMP agents

max-vars-per-pdu

Ο αριθμός των μεταβλητών ανά πακέτο SNMP

max-request-size

Το μέγιστο μέγεθος των εξερχόμενων SNMP αιτημάτων.

Στο παράδειγμα βλέπουμε ότι μπορούμε να ορίσουμε διαφορετικές ομάδες από IP διευθύνσεις των οποίων οι agents του SNMP είναι διαφορετικά ρυθμισμένοι, τρέχουν διαφορετικές εκδόσεις του SNMP και τα ονόματα των communities διαφέρουν. Κάθε ορισμός μπορεί να περιέχει διαφορετικές ιδιότητες.
Όλες οι γενικές μεταβλητές μπορούν επικαλυφθούνε με αυτές που ορίζονται μέσα στις ετικέτες <definition>.

6.2. - NSClient

Για να συλλέξουμε πληροφορίες από μία συσκευή η οποία τρέχει το λογισμικό NSClient πρέπει να γίνουν ρυθμίσεις σε δύο αρχεία. Στο nsclient-config.xml και στο nsclient-datacollection-config.xml. Παρουσιάζονται παρακάτω:

nsclient-config.xml

```
<?xml version="1.0"?>
<nsclient-config port="1248" retry="1" timeout="3000">
</nsclient-config>
```

retry

Ο αριθμός των προσπαθειών που θα γίνουν για την σύνδεση στον NSClient agent.

timeout

Το χρονικό διάστημα του σε milliseconds που το OpenNMS θα περιμένει για μια ανταπόκριση από τον NSClient agent.

port

Δηλώνοντας κάποια άλλη πόρτα μπορούμε να αλλάξουμε την προκαθορισμένη που είναι η 1248.

password

Το συνθηματικό (εάν υπάρχει) για να την αυθεντικοποίηση στον NSClient agent.

nsclient-datacollection-config.xml

Σε αυτό το αρχείο ορίζουμε της αντικείμενα που θέλουμε να συλλέξουμε και με ποιο τρόπο θα αποθηκευτούνε.

```
<nsclient-datacollection-config rrdRepository="/opt/opennms/share/rrd/snmp/">
  <nsclient-collection name="default">
    <rrd step="300">
      <rra>RRA:AVERAGE:0.5:1:8928</rra>
      <rra>RRA:AVERAGE:0.5:12:8784</rra>
      <rra>RRA:MIN:0.5:12:8784</rra>
      <rra>RRA:MAX:0.5:12:8784</rra>
    </rrd>
    <wpms>
      <!--  A group for collecting processor stats.
        Check the keyvalue "% Processor Time" - if it's there (should be) collect this
whole group.
      Check every recheckInterval milliseconds (3600000 = 1hr) -->
```

```
    <wpm name="Processor" keyvalue="\Processor(_Total)\% Processor Time"
recheckInterval="3600000">
        <!--  Collect these attributes.  Name is the name to pass to NSClient.
        Alias is the local name for the RRD file
        Type is used to convert values around
        maxval/minval are optional-->
        <attrib name="\Processor(_Total)\% Processor Time" alias="cpuProcTime"
type="Gauge"/>
        <attrib name="\Processor(_Total)\% Interrupt Time" alias="cpuIntrTime"
type="Gauge"/>
        <attrib name="\Processor(_Total)\% Privileged Time" alias="cpuPrivTime"
type="Gauge"/>
        <attrib name="\Processor(_Total)\% User Time" alias="cpuUserTime"
type="Gauge"/>
    </wpm>
  </nsclient-datacollection>
</nsclient-datacollection-config>
```

Οι μετρητές απόδοσης που θα συλλέξουμε ορίζονται μέσα στην ετικέτα <wpms>. Ενώ μια ομάδα από μετρητές ορίζετε μέσα σε μια ετικέτα <wpm>. Κάθε <wpm> απαρτίζετε από:

name

Ο μετρητής απόδοσης που θα συλλέξουμε.

alias

Ορίζει το όνομα του rrd στοιχείου που θα αποθηκευτεί. Μέχρι 19 χαρακτήρες.

type

Εδώ ορίζουμε τι τύπου θα είναι η πληροφορία που θα συλλέξουμε. Δηλώνοντας “gauge” είναι μια πληροφορία σε σχέση με τον χρόνο, π.χ. το ποσοστό χρησιμοποίησης ενός επεξεργαστή. Ενώ αν δηλωθεί σαν “counter” πρόκειται για μία τιμή που αυξάνετε συνεχώς, π.χ. ο αριθμός των bit που στέλνονται από μία κάρτα δικτύου.

6.3. - JMX

Είναι μια τεχνολογία με χρήση της JAVA η οποία παρέχει κάποια εργαλεία για την διαχείριση και παρακολούθηση εφαρμογών, μεταβλητών του συστήματος, συσκευών και δικτυακών υπηρεσιών. Για να συλλέξουμε πληροφορίες μέσω της JMX πρέπει να ρυθμίσουμε το αρχείο jmx-datacollection-config.xml. Αυτό το αρχείο ορίζει ποια στοιχεία θα συλλέξουμε. Στη συγκεκριμένη περίπτωση πρόκειται για MBeans.

```
<?xml version="1.0"?>
<jmx-datacollection-config
   rrdRepository = "/opt/opennms/rrd/snmp/">
   <jmx-collection name="jboss"
      maxVarsPerPdu = "50">
      <rrd step = "300">
         <rra>RRA:AVERAGE:0.5:1:8928</rra>
         <rra>RRA:AVERAGE:0.5:12:8784</rra>
         <rra>RRA:MIN:0.5:12:8784</rra>
         <rra>RRA:MAX:0.5:12:8784</rra>
      </rrd>
 <mbeans>
   <mbean name="SystemInfo" objectname="jboss.system:type=ServerInfo">
   <attrib name="FreeMemory"   alias="FreeMemory"       type="gauge"/>
   <attrib name="TotalMemory"  alias="TotalMemory"      type="gauge"/>
   </mbean>
 </mbeans>
</jmx-collection>
</jmx-datacollection-config>
```

objectname

Το όνομα του αντικειμένου το οποίο θα χρησιμοποιηθεί στον JMX agent**.**

Μέσα σε κάθε ετικέτα mbean ορίζονται τα χαρακτηριστικά του αντικειμένου που θέλουμε να συλλέξουμε. Κάθε χαρακτηριστικό ορίζετε σε μια ετικέτα attrib και έχει:

name

Το όνομα του χαρακτηριστικού που θα πάρουμε από το αντικείμενο mbean.

alias

Ορίζει το όνομα του rrd στοιχείου που θα αποθηκευτεί. Μέχρι 19 χαρακτήρες.

type

Εδώ ορίζουμε τι τύπου θα είναι η πληροφορία που θα συλλέξουμε. Δηλώνοντας “gauge” είναι μια πληροφορία σε σχέση με τον χρόνο, π.χ. το ποσοστό χρησιμοποίησης ενός επεξεργαστή. Ενώ αν δηλωθεί σαν “counter” πρόκειται για μία τιμή που αυξάνετε συνεχώς, π.χ. ο αριθμός των bit που στέλνονται από μία κάρτα δικτύου.

6.4. - HTTP

Για να συλλέξουμε πληροφορίες μέσω του HTTP πρωτοκόλλου πρέπει να ρυθμίσουμε το αρχείο http-datacollection-config.xml. Μέσα στο αρχείο μπορούμε να ορίσουμε URLs και κανονικές εκφράσεις που θα χρησιμοποιηθούν για να εξάγουν την πληροφορία από τις επιστρεφόμενες σελίδες.

```
<?xml version="1.0" encoding="UTF-8"?>
<http-datacollection-config
    xmlns:http-dc="http://xmlns.opennms.org/xsd/config/http-datacollection"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.opennms.org/xsd/config/http-datacollection
http://www.opennms.org/xsd/config/http-datacollection-config.xsd"
    rrdRepository="@install.share.dir@/rrd/snmp/" >
  <http-collection name="doc-count">
    <rrd step="300">
      <rra>RRA:AVERAGE:0.5:1:8928</rra>
      <rra>RRA:AVERAGE:0.5:12:8784</rra>
      <rra>RRA:MIN:0.5:12:8784</rra>
      <rra>RRA:MAX:0.5:12:8784</rra>
    </rrd>
    <uris>
      <uri name="document-counts">
        <url path="/test/resources/httpcolltest.html"
```

```
            user-agent="Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en)
AppleWebKit/412 (KHTML, like Gecko) Safari/412"
            matches=".*([0-9]+).*" response-range="100-399" >
        </url>
        <attributes>
          <attrib alias="documentCount" match-group="1" type="counter32"/>
        </attributes>
      </uri>
    </uris>
  </http-collection>
</http-datacollection-config>
```

6.5. - Ο «Δαιμονας» collectd

Όλη η συλλογή δεδομένων χειρίζεται από αυτή την διεργασία. Η διεργασία περιμένει για γεγονότα NodeGainedService που αφορούνε την υπηρεσία του SNMP. Όταν λάβει ένα τέτοιο γεγονός ελέγχει εάν υπάρχει βασικό SNMP interface για αυτό τον κόμβο και εάν η IP διεύθυνση του interface υπάρχει μέσα σε κάποιο πακέτο του αρχείου collectd-configuration.xml τότε η συλλογή δεδομένων αρχίζει.
Εάν δεν ορίσουμε εμείς κάποια συγκεκριμένη έκδοση του SNMP στο αρχείο snmp-config.xml τότε η διεργασία συλλογής δεδομένων θα προσπαθήσει να συλλέξει το αντικείμενο system.sysObjectID μέσω της εντολής GET-BULK η οποία υποστηρίζετε μόνο από τη δεύτερη έκδοση του SNMP. Αν αποτύχει τότε χρησιμοποιείτε η πρώτη έκδοση του SNMP.
Ή δομή του αρχείου ρυθμίσεων παρουσιάζετε παρακάτω και μοιάζει αρκετά με την δομή του αρχείου poller-configuration.xml. Ισχύει και εδώ η λογική των πακέτων, δηλαδή μπορούμε σε διαφορετικά πακέτα να ορίσουμε διαφορετικές IP διευθύνσεις που θα έχουν διαφορετικές ρυθμίσεις. Ο τρόπος επιλογής των IP διευθύνσεων γίνετε με τον ίδιο τρόπο όπως και στο αρχείο poller-configuration.xml και παρουσιάζετε και στην ενότητα 5.2. Παρακάτω βλέπουμε με ένα παράδειγμα του αρχείου και αμέσως μετά θα παρουσιαστούν κάποιες βασικές ετικέτες του.

```
<?xml version="1.0" encoding="UTF-8"?>
<collectd-configuration
```

```
   xmlns="http://xmlns.opennms.org/xsd/config/collectd" threads="50">
   <package name="example1">
      <filter>IPADDR != '0.0.0.0'</filter>
      <include-range begin="1.1.1.1" end="254.254.254.254"/>
      <service name="SNMP" interval="300000" user-defined="false" status="on">
         <parameter key="collection" value="default"/>
         <parameter key="thresholding-enabled" value="true"/>
      </service>
      <service name="NSClientpp" interval="300000"
         user-defined="false" status="on">
         <parameter key="nsclient-collection" value="default"/>
         <parameter key="timeout" value="5000"/>
      </service>
   </package>
<collector service="SNMP" class-
name="org.opennms.netmgt.collectd.SnmpCollector"/>
   <collector service="NSClientpp" class-
name="org.opennms.netmgt.collectd.NSClientCollector"/>
</collectd-configuration>
```

Services

Κάθε πακέτο έχει μια ομάδα υπηρεσιών οι οποίες συλλέγουν τα δεδομένα. Πτος το παρών βέβαια στο OpenNMS υπάρχει μόνο μία το SNMP.

name

Το όνομα τις υπηρεσίας

interval

Ορίζει τη συχνότητα συλλογής των δεδομένων (προκαθορισμένο στα 5 λεπτά)

user-defined

Για μελλοντική χρήση όπου οι χρήστες μέσω γραφικού περιβάλλοντος θα μπορούν να ορίσουν πηγές δεδομένων. π.χ. αρχεία κειμένου.

status

Ορίζουμε αν η συλλογή δεδομένων θα είναι ενεργή η όχι.

Service Parameters

Οι παρακάτω τρεις παράμετροι είναι κοινές σε όλες τις υπηρεσίες.

timeout

Ο χρόνος που θα περιμένει σε millıseconds για μια απάντηση από τον SNMP agent.

retries

Εάν ο χρόνος απόκρισης του SNMP agent λήξει. Από εδώ ορίζουμε πόσες φορές θα ξαναπροσπαθήσει να συνδεθεί.

port

Δηλώνοντας κάποια άλλη πόρτα μπορούμε να αλλάξουμε την προκαθορισμένη.

SNMP

Η SNMP υπηρεσία μπορεί να έχει της εξής παραμέτρους:

collection

Δείχνει σε μια SNMP συλλογή στο αρχείο datacollection-config.xml που ορίζει ποια Object IDs (OIDs) θα συλλεχθούνε.

JBOSS

factory

ορίζει τη μέθοδο σύνδεσης στον εξυπηρετητή JBOSS. Μπορεί να είναι είτε HTTP ή RMI.

NSClient

Η NSClient υπηρεσία έχει μια επιπλέον παράμετρο:

nsclient-collection

Δείχνει σε μια συλλογή στο αρχείο nsclient-datacollection-config.xml που ορίζει την πληροφορία που θα συλλεχθεί.

6.6. - Ρυθμίσεις για την συλλογή πληροφοριών(datacollection-config.xml)

Το αρχείο datacollection-config.xml καθορίζει τις τιμές δηλαδή τα αντικείμενα της βάσης MIB που θα συλλεχθούνε από τους agents του SNMP. Ακολουθεί την ίδια λογική με τα πακέτα όπως και στο αρχείο poller-configuration.xml αλλά εδώ το κάθε πακέτο είναι μία διαφορετική συλλογή δεδομένων και ορίζετε με την ετικέτα <snmp-collection>. Η κάθε υπηρεσία του SNMP που δηλώνουμε στο αρχείο collectd-configuration.xml δείχνει σε ένα πακέτο συλλογής δεδομένων. Στα πακέτα του αρχείου datacollection-config.xml ορίζονται ομάδες που περιέχουν αντικείμενα

της βάσης MIB(OIDs) και οι ομάδες αυτές εισάγονται σε συστήματα. Τα συστήματα μετά συσχετίζονται με interfaces με βάση το αντικείμενο systemOID. Επίσης στα πακέτα ορίζετε πως η πληροφορία θα συλλεχθεί και θα αποθηκευτεί.
Το αρχείο αρχίζει ως εξής:

```
<datacollection-config
        rrdRepository = "/var/opennms/rrd/snmp/">
```

Αυτή η δήλωση ορίζει σε ποιο κατάλογο θα αποθηκευτούν οι πληροφορίες που συλλέγουμε. Εάν θέλουμε να αλλάξουμε αυτό τον κατάλογο πρέπει να το κάνουμε επίσης και στα παρακάτω αρχεία.
poller-configuration.xml
thresholds.xml
http-datacollection-config.xml
jmx-datacollection-config.xml
nsclient-datacollection-config.xml

Αφού ορίσαμε τον προορισμό των δεδομένων αμέσως μετά ακολουθεί η δήλωση της συλλογής.

```
<snmp-collection name="default"
                maxVarsPerPdu = "50"
                snmpStorageFlag = "all">
```

Το όνομα που ορίζουμε εδώ πρέπει να είναι το ίδιο με αυτό τις παραμέτρου collection στο αρχείο collectd-configuration.xml.
Το maxVarsPdu είναι ένα όριο στο πόσες τιμές μπορεί να περιέχει ένα GET-BULK αίτημα μέσα σε ένα πακέτο. Χρειάζεται να αλλάξουμε την τιμή του σε περίπτωση που έχουμε κάποιον αρκετά αργό SNMP agent.
Το snmpStorageFlag ορίζει εάν θα συλλέγοντα δεδομένα μόνο για το βασικό interface ή για όλα τα interface. Αμέσως μετά ακολουθεί η δήλωση της βάσης δεδομένων Round Robin.

6.7. - Βάση δεδομένων Round Robin (RRD)

Το RRDTool είναι ένα εργαλείο το οποίο προήλθε από το MRTG. Δημιουργεί μια πολύ συμπαγή δομή για την αποθήκευση περιοδικής πληροφορίας σαν αυτή που συλλέγει ένα σύστημα διαχείρισης δικτύου. Το αρχεία που δημιουργούνται είναι

μεγάλα σε μέγεθος αρχικά αλλά δεν πρόκειται να μεγαλώσουν άλλο. Όταν ένα αρχείο γεμίσει τότε η ποιο παλιά πληροφορία διαγράφετε και στην θέση της μπαίνει καινούργια.

Το OpenNMS δεν χρησιμοποιεί το RRDTool αλλά μια υλοποίηση του σε JAVA που ονομάζετε JRobin. Τα αρχεία που δημιουργούνται έχουνε την κατάληξη .jrb και δεν είναι συμβατά με το RRDTool. Κάθε αρχείο απαρτίζετε από συλλογές Round Robin (RRA). Οι δηλώσεις RRA έχουν την μορφή:

RRA:Cf:xff:steps:rows

RRA

Έτσι αρχίζει μια δήλωση συλλογής Round Robin.

Cf

Αυτό το πεδίο αντιπροσωπεύει την συνάρτηση συνένωσης(consolidation function). Μπορεί να πάρει της τιμές AVERAGE, MAX, MIN, ή LAST.

xff

Είναι η παράμετρος x-files factor. Όταν πάει να συνενώσει έναν αριθμό δειγμάτων υπάρχει μια πιθανότητα κάποια δείγματα να είναι κενά επειδή η τιμή δεν συλλέχθηκε για διάφορους λόγους. Σε αυτή την περίπτωση η τιμή θα χαρακτηριστεί σαν UNKNOWN. Η παράμετρος xff καθορίζει το ποσοστό των δειγμάτων επί τοις εκατό που εάν έχουν τιμή UNKNOWN τότε το δείγμα που θα δημιουργηθεί από την συνένωση θα έχει και αυτό την τιμή UNKNOWN. Η προκαθορισμένη τιμή του είναι 0.5 ή 50%.

steps

Ο αριθμός των βημάτων που αποθηκεύονται σε μια συλλογή RRA. Για παράδειγμα εάν το βήμα είναι 300 δευτερόλεπτα (5 λεπτά) και ο αριθμός των βημάτων είναι 12, τότε το RRA 12 X 5 λεπτά = 60 λεπτά = 1 ώρα και θα αποθηκευτεί η τιμή συνένωσης για αυτή την ώρα.

rows

Το πεδίο rows ορίζει τον αριθμό των τιμών που θα αποθηκευτούν σε μια συλλογή RRA.

```
<rrd step = "300">
  <rra>RRA:AVERAGE:0.5:1:8928</rra>
```

```
  <rra>RRA:AVERAGE:0.5:12:8784</rra>
  <rra>RRA:MIN:0.5:12:8784</rra>
  <rra>RRA:MAX:0.5:12:8784</rra>
</rrd>
```

Οι δηλώσεις που βλέπουμε ορίζουν τις παραμέτρους μιας βάσης δεδομένων Round Robin για την αποθήκευση των συλλεχθέντων πληροφοριών όπου ορίζουμε σαν βήμα τα 300 δευτερόλεπτα (5 λεπτά) και μετά στην πρώτη συλλογή RRA αποθηκεύουμε τιμές από τα τελευταία 8928 πεντάλεπτα, στη δεύτερη τον μέσο όρο των τελευταίων 8784 ωρών (366 ημέρες). Στην τρίτη την μέγιστη τιμή των τελευταίων 8784 ωρών και στην τέταρτη την ελάχιστη των τελευταίων 8784 ωρών.

6.8. - Τύποι πηγών πληροφοριών (Resource Types)

Εάν θέλουμε να συλλέξουμε πληροφορίες υπό μορφή πίνακα από πίνακες στη βάση MIB πρέπει να φτιάξουμε δικούς μας ορισμούς για τον τύπο των πόρων που θα συλλέξουμε, μέσα στο αρχείο datacollection-configuration.xml. Αφού ορίσουμε τον τύπο των πόρων μετά των χρησιμοποιούμε στην συλλογή και στην αναπαράσταση της πληροφορίας που συλλέξαμε. Για την συλλογή του πίνακα MIB-2 host resources storage ο ορισμός πρέπει να είναι ως εξής:

```
<resourceType name="hrStorageIndex" label="Storage (MIB-2 Host Resources)">
        <persistenceSelectorStrategy
        class="org.opennms.netmgt.collectd.PersistAllSelectorStrategy"/>
        <storageStrategy
        class="org.opennms.netmgt.dao.support.IndexStorageStrategy"/>
</resourceType>
```

name

Ορίζουμε ένα όνομα δικιάς μας επιλογής το οποίο όμως θα το χρησιμοποιήσουμε αργότερα στο αρχείο datacollection-configuration.xml όπως και στο αρχείο snmp-graph.properties για να αναφερθούμε σε αυτό τον πόρο.

label

Χρησιμοποιείτε για να δηλώσουμε μια ποιο ευανάγνωστη ετικέτα η οποία θα εμφανίζετε στο γραφικό περιβάλλον του OpenNMS για τον συγκεκριμένο πόρο.

Τέλος οι παράμετροι persistenceSelectorStrategy και storageStrategy είναι δύο κλάσεις οι οποίες ορίζουν αν η πληροφορία θα γραφτεί στο δίσκο και το πώς θα γραφτεί. Υπάρχει μόνο μία επιλογή για την κάθε μία οπότε δεν χρειάζεται κάποια ρύθμισή από μέρους μας.

6.9. - Ομάδες πηγών πληροφοριών (Groups)

Αφού ορίσουμε τους τύπους των πηγών των πληροφοριών φτιάχνουμε τις ομάδες.

```
<group name="mib2-host-resources-storage" ifType="all">
  <mibObj oid=".1.3.6.1.2.1.25.2.3.1.3" instance="hrStorageIndex"
alias="hrStorageDescr" type="string" />
  <mibObj oid=".1.3.6.1.2.1.25.2.3.1.4" instance="hrStorageIndex"
alias="hrStorageAllocUnits" type="gauge" />
  <mibObj oid=".1.3.6.1.2.1.25.2.3.1.5" instance="hrStorageIndex"
alias="hrStorageSize" type="gauge" />
  <mibObj oid=".1.3.6.1.2.1.25.2.3.1.6" instance="hrStorageIndex"
alias="hrStorageUsed" type="gauge" />
</group>
```

Τα αντικείμενα του SNMP μπαίνουμε μέσα σε ομάδες για να είναι ποιο εύκολο να συσχετιστούν μετά με κάποια συγκεκριμένη συσκευή. Μια ομάδα(group) αποτελείτε από ένα όνομα(name) κα τον τύπο των interfaces(iftype) από τις οποίες τα αντικείμενα-μέλη (mibObj) θα συλλεχθούν.

Η παράμετρος iftype μπορεί να πάρει τις εξής τιμές:

all

Από όλους τους τύπους interfaces(Ethernet,ATM,κ.λ.π), θα συλλεχθούν τα αντικείμενα του SNMP που περιέχονται στην ομάδα.

ignore

Χρησιμοποιείτε για ομάδες τιμών οι οποίες υπάρχουν σε μια συσκευή μόνο μία φορά. Π.χ. το φόρτο του επεξεργαστή.

[specific numeric value]

Μας δίνει την δυνατότητα να συλλέξουμε πολύ συγκεκριμένα interfaces. Π.χ interfaces ATM που να μην είναι point-to-point WAN links.

<group name = "my-ATM-example" ifType = "37">

Στην ιστοσελίδα http://www.iana.org/assignments/ianaiftype-mib υπάρχει μια εκτενής λίστα από όλες τις τιμές των τύπων των interfaces.

6.10. - Συστήματα (Systems)

Από τη στιγμή που οι ομάδες των αντικειμένων οριστούνε το μόνο που μένει είναι να τις συσχετίσουμε με συστήματα τα οποία θα ελέγχονται και από αυτά θα συλλέγετε η πληροφορία. Εδώ με τον όρο σύστημα εννοούμε κάποια συσκευή. Ο κάθε SNMP agent διαθέτει ένα μοναδικό systemOID (.1.3.6.1.2.1.1.2, instance 0) το οποίο μας βοηθάει να προσδιοριστεί μοναδικά μια συσκευή. Παρακάτω ακολουθεί ο ορισμός ενός συστήματος:

```
<systems>
  <systemDef name = "Net-SNMP">
    <sysoidMask>.1.3.6.1.4.1.2021.250.</sysoidMask>
    <collect>
      <includeGroup>mib2-interfaces-net-snmp</includeGroup>
      <includeGroup>mib2-host-resources-storage</includeGroup>
      <includeGroup>mib2-host-resources-system</includeGroup>
      <includeGroup>mib2-host-resources-memory</includeGroup>
      <includeGroup>ucd-loadavg</includeGroup>
    </collect>
  </systemDef>
```

Σε αυτό τον ορισμό του συστήματος δηλώνουμε ότι όποια συσκευή έχει systemOID το οποίο αρχίζει από “.1.3.6.1.4.1.2021.250.”, τότε από αυτή τη συσκευή θα συλλεχθούνε οι ομάδες mib2-interfaces-net-snmp, mib2-host-resources-storage, mib2-host-resources-system, mib2-host-resources-memory and ucd-loadavg. Αντί για την ετικέτα sysoidMask μπορούμε να έχουμε την sysoid όπου εκεί ορίζουμε ένα συγκεκριμένο systemOID και όχι έναν αριθμό από τον οποίο θα αρχίζει το systemOID.

6.11. - Επίλογος

Στο κεφάλαιο αυτό αναλύσαμε την δομή των αρχείων που χρησιμεύουν για την συλλογή πληροφοριών από τις δικτυακές συσκευές. Είπαμε ότι μπορούμε να ορίσουμε πακέτα με διαφορετικές υπηρεσίες για την συλλογή δεδομένων μέσα στο αρχείο collectd-configuration.xml. Κάθε υπηρεσία συλλογής δεδομένων δείχνει σε μία συλλογή αντικειμένων που την ορίζουμε στο αρχείο datacollection-config.xml. Κάθε συλλογή αντικειμένων περιέχει τις ομάδες των αντικείμενων(OIDs) που θέλουμε να συλλέγουμε , τα interface από τα οποία θα γίνετε η συλλογή και ποιες ομάδες αντικειμένων είναι συσχετισμένες με ποια συστήματα με βάση το systemOID. Στο επόμενο κεφαλαίο θα αναλύσουμε την διαδικασία λήψης και δημιουργίας γεγονότων είτε από τις συσκευές δικτύου είτε από εσωτερικά γεγονότα που προκαλούνται από το ίδιο το OpenNMS.

ΚΕΦΑΛΑΙΟ 7

Γεγονότα (Events)

Τα γεγονότα είναι κύριο θέμα σε κάθε σύστημα διαχείρισης δικτύου (NMS). Στην πραγματικότητα, η κύρια λειτουργία ενός συστήματος διαχείρισης δικτύου είναι να ανακαλύπτει αλλαγές μέσα σε ένα δίκτυο, και κάθε αλλαγή μπορούμε να πούμε ότι είναι ένα γεγονός.

Το OpenNMS διαχειρίζεται τα γεγονότα μέσω ενός «δαίμονα» ο οποίος λέγεται eventd. Υπάρχουν δύο τύποι γεγονότων, αυτά τα οποία δημιουργούνται από το ίδιο το OpenNMS και αυτά που δημιουργούνται από εξωτερικά SNMP traps. Η «δαίμονες» από τους οποίους απαρτίζετε το OpenNMS δημιουργούν γεγονότα, όπως όταν η διαδικασία εύρεσης συσκευών δημιουργεί ένα γεγονός newSuspect όταν ένα interface αποκρίνετε σε ένα πακέτο ping.

Όταν ένα γεγονός δημιουργείτε, μπορούν να οριστούν διάφορες παράμετροι, όπως μια περιγραφή (description), ένα μήνυμα καταγραφής (log message) και η δριμύτητα (severity). Επίσης αυτοματοποιημένες διεργασίες μπορούν να εκτελεστούν και να στείλουν τις παραμέτρους του γεγονότος σε ένα εξωτερικό πρόγραμμα. Αυτό ορίζετε μέσα στο αρχείο eventconf.xml.

Το OpenNMS έχει επίσης ένα πλούσιο σύστημα ειδοποιήσεων (Notifications). Συγκεκριμένα γεγονότα μπορούνε να σταλθούν σαν ειδοποιήσεις με ηλεκτρονικό ταχυδρομείο ή μέσω SMS. Στο υπόλοιπο αυτού του κεφαλαίου θα αναλύσουμε τα γεγονότα με λεπτομέρεια.

7.1. Ρυθμίσεις στο αρχείο eventconf.xml

Το αρχείο eventconf.xml βρίσκετε στον κατάλογο /etc/opennms/, και είναι το αρχείο ρυθμίσεων για το πώς μεταχειρίζονται τα γεγονότα. Παρακάτω ακολουθεί η αρχή του αρχείου και στις επόμενες ενότητες θα παρουσιαστούν οι υπόλοιπες παράμετροι του αρχείου.

```
<events xmlns="http://xmlns.opennms.org/xsd/eventconf">
  <global>
```

```
    <security>
      <doNotOverride>logmsg</doNotOverride>
      <doNotOverride>operaction</doNotOverride>
      <doNotOverride>autoaction</doNotOverride>
      <doNotOverride>tticket</doNotOverride>
    </security>
</global>
```

Κάθε αρχείο που περιέχει γεγονότα αρχίζει με την ετικέτα <events> και τελειώνει με την ετικέτα </events>.

Ο «δαίμονας» eventd ακούει στην πόρτα 5817, για αυτό άλλες διεργασίες, ακόμα και αυτές που είναι εξωτερικές από το OpenNMS μπορούν να στείλουν γεγονότα στο σύστημα. Η ετικέτα <security> βοηθά ώστε να αποτρέπει γεγονότα να εκτελούνε πράξεις οι οποίες δεν είναι ορισμένες στο αρχείο eventconf.xml. Με αυτό τον τρόπο δεν μπορεί κάποιος να στείλει ένα γεγονός στο σύστημα που να έχει σαν αυτόματη πράξη το άνοιγμα ενός τερματικού του root χρήστη. Επειδή το αρχείο eventconf.xml ορίζοντας συνεχώς καινούργια γεγονότα, κατά κύριο λόγω από SNMP traps, που ενσωματώνονται θα μπορούσε να γίνει πολύ μεγάλο μπορούμε να έχουμε ξεχωριστά αρχεία που ορίζονται μέσα τους γεγονότα και να δηλώνουμε την τοποθεσία τους στο τέλος του αρχείου eventconf.xml με τον εξής τρόπο:

```
<event-file>events/Audiocodes.events.xml</event-file>
```

7.2. - Εσωτερικά γεγονότα (Internal Events)

Μετά τις γενικές παραμέτρους ακολουθεί ο ορισμός των γεγονότων για το πώς θα διαχειρίζεται το καθένα από το OpenNMS. Ακολουθεί ένα παράδειγμα ενός πολύ συνηθισμένου γεγονότος του nodeLostService:

```
<event>
  <uei>http://uei.opennms.org/nodes/nodeLostService</uei>
  <event-label>OpenNMS-defined node event: nodeLostService</event-label>
  <descr>
    &#38lt;p&#38gt;A %service% outage was identified on interface
    %interface%.&#38lt;/p&#38gt; &#38lt;p&#38gt;A new Outage record has been
```

```
    created and service level availability calculations will be
    impacted until this outage is resolved.&#38lt;/p&#38gt;
  </descr>
  <logmsg dest='logndisplay'>
        %service% outage identified on interface %interface%.
  </logmsg>
  <severity>Major</severity>
</event>
```

Κάθε γεγονός ξεκινά με την ετικέτα <event> και τελειώνει με την </event>. Μέσα σε αυτή μπορούμε να έχουμε πολλούς άλλους ορισμούς. Ακολουθεί η επεξήγηση τους.

UEI

Το "Universal Event Identifier" είναι μία ετικέτα που προσδιορίζει μοναδικά το γεγονός.

event-label

Μια ετικέτα για το γεγονός που χρησιμοποιείτε συχνά στο γραφικό περιβάλλον του OpenNMS.

descr

Εμπεριέχει μια περιγραφή για το γεγονός. Μπορεί να εμπεριέχει HTML οντότητες για την μορφοποίηση της. Μπορούμε επίσης να εισάγουμε στοιχεία όπως το %interface% και άλλα τα οποία περιγράφονται παρακάτω.

logmsg

Εμπεριέχει μια σύντομη περιγραφή ή μια περίληψη του γεγονότος. Η παράμετρος **dest** μπορεί να πάρει τις εξής τιμές:

logndisplay

Καταγράφει το γεγονός στην βάση δεδομένων και το παρουσιάζει και στο γραφικό περιβάλλον

logonly

Καταγράφει το γεγονός στην βάση δεδομένων αλλά δεν το παρουσιάζει στο γραφικό περιβάλλον

Suppress

Το γεγονός ούτε καταγράφετε ούτε παρουσιάζετε.

donotpersist

Δεν καταγράφει το γεγονός στην βάση δεδομένων αλλά το προωθεί στους «δαίμονες» που ακούνε για ένα γεγονός τέτοιου τύπου.

discardtraps

Εδώ μπορούμε να δηλώσουμε ένα trap το οποίο θέλουμε να απορρίπτεται χωρίς να δημιουργείτε κάποιο γεγονός.

severity

Αυτή η παράμετρος υποδεικνύει την δριμύτητα του γεγονότος.

7.3. - Δριμύτητες των γεγονότων (Severities)

Η δριμύτητα ενός γεγονότος καθορίζεται από διάφορους παράγοντες και πολλές φορές είναι κάτι υποκειμενικό για ένα δίκτυο. Το OpenNMS εμπεριέχει επτά διαφορετικούς βαθμούς δριμύτητας αλλά υπάρχει και η δυνατότητα να καθορίσουμε και εμείς νέους ή να παραλλάξουμε τους ήδη υπάρχοντες. Παρακάτω παρουσιάζετε μια λίστα με τους βασικούς βαθμούς.

Critical (dark red)

Σημαίνει ότι πολλές συσκευές επηρεάζονται από αυτό το γεγονός και ότι το πρόβλημα που αναφέρετε πρέπει να επιλυθεί σύντομα.

Major (light red)

Σημαίνει ότι η συσκευή είναι ανενεργή ή υπάρχει ο κίνδυνος να έρθει σε αυτή την κατάσταση.

Minor (orange)

Σημαίνει ότι ένα μέρος μιας συσκευής (μια υπηρεσία, ένα interface, ένα τροφοδοτικό ρεύματος, κτλ) έχει σταματήσει να λειτουργεί και πρέπει να δώσουμε προσοχή.

Warning (yellow)

Σημαίνει ότι το γεγονός το οποίο συνέβη μπορεί να χρειάζεται την προσοχή μας.

Normal (green)

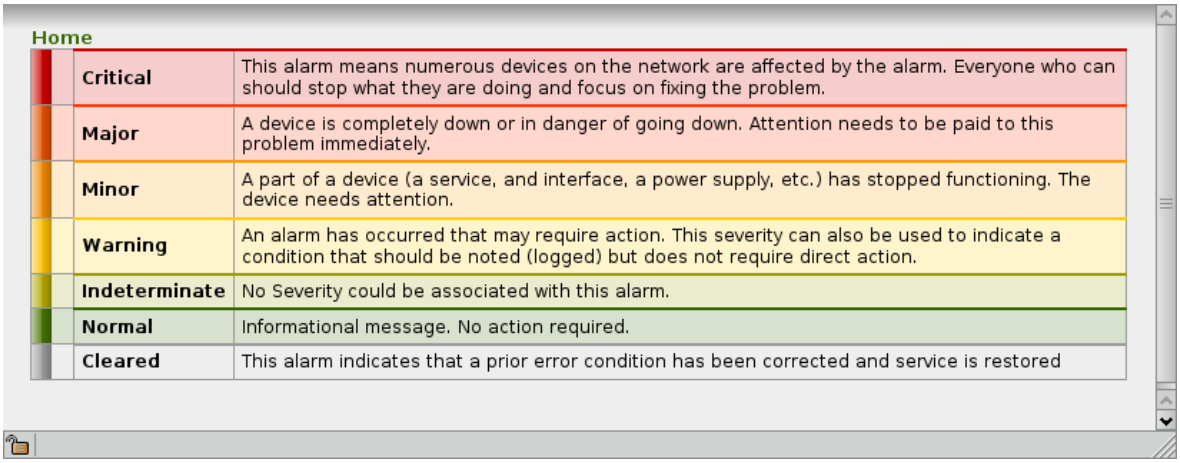
Σημαίνει κάποιο ενημερωτικό μήνυμα, Δεν χρειάζεται να κάνουμε κάτι.

Cleared (light grey)

Σημαίνει ότι μια προηγούμενη κατάσταση διορθώθηκε και μια υπηρεσία επανήλθε.

Indeterminate (yellow-green)

Σημαίνει ότι η δριμύτητα του γεγονότος δεν μπορεί να προσδιοριστεί.



Σχήμα 7.1 – Οι διαφορετικές δριμύτητες των γεγονότων του OpenNMS

7.4. - Στοιχεία (Elements)

Διάφορα στοιχεία μπορούνε να εμπεριέχονται στην περιγραφή, στα μηνύματα καταγραφής, στις οδηγίες για τον χειριστή ή σε αυτοματοποιημένες διεργασίες που εκτελούνται μόλις δημιουργηθεί ένα συγκεκριμένο γεγονός. Όλα τα στοιχεία δεν υπάρχουν για όλα τα γεγονότα και μερικά ισχύουνε μόνο για τα SNMP traps που αναφέρονται στην επόμενη ενότητα.

%uei%

Το Universal Event Identifier για αυτό το γεγονός.

%source%

Η πηγή του γεγονότος.

%time%

Η ώρα που συνέβη το γεγονός.

%nodeid%

Το nodeid της συσκευής που προκάλεσε το γεγονός.

%interface%

Το interface που συσχετίζετε με το γεγονός.

%service%

Η υπηρεσία που συσχετίζετε με το γεγονός.

%severity%

Η δριμύτητα του γεγονότος.

%snmphost%

Η συσκευή που εκτελεί τον SNMP agent ο οποίος δημιούργησε το γεγονός.

%snmp%

Οι πληροφορίες του SNMP για το γεγονός.

%id%

Το SNMP OID για το γεγονός.

%generic%

Ο γενικός αριθμός του SNMP trap για το γεγονός.

%specific%

Ο συγκεκριμένος αριθμός του SNMP trap για το γεγονός.

%community%

Η συμβολοσειρά community του SNMP trap.

%version%

Η έκδοση του SNMP με την οποία στάλθηκε το trap.

%operinstruct%

Οι οδηγίες του χειριστή για το γεγονός.

%mouseovertext%

Το κείμενο που θα εμφανίζετε όταν βάζουμε το κέρσορα από το ποντίκι πάνω στο γεγονός στο γραφικό περιβάλλον.

%nodelabel%

Το όνομα του κόμβου.

%interfaceresolv%

Το όνομα ενός interface εάν υπάρχει.

7.5. - SNMP Traps

Εκτός από τα γεγονότα τα οποία δημιουργούνται εσωτερικά το OpenNMS μπορεί να λάβει και SNMP traps μέσω του «δαίμονα» trapd. Ορίζονται μέσα στο αρχείο eventconf.xml χρησιμοποιώντας την ετικέτα <mask>. Ακολουθεί ένα παράδειγμα.

<event>

<mask>

<maskelement>

<mename>id</mename>

<mevalue>.1.3.6.1.4.1.9.9.70.2</mevalue>

</maskelement>

<maskelement>

<mename>generic</mename>

<mevalue>6</mevalue>

</maskelement>

<maskelement>

<mename>specific</mename>

<mevalue>17</mevalue>

</maskelement>

</mask>

<uei>http://uei.opennms.org/vendor/Cisco/traps/ciscoC3800SysAggregateStatusChange</uei>

<event-label>CISCO-C3800-MIB defined trap event: ciscoC3800SysAggregateStatusChange</event-label>

<descr>&#38lt;p&#38gt;Notification that the aggregate status of a node has changed.&#38lt;/p&#38gt;&#38lt;table&#38gt;&#38lt;tr&#38gt;&#38lt;td&#38gt;&#38lt;b&#38gt;
c3800SysNextTrapSeqNum&#38lt;/b&#38gt;</td&#38gt;&#38lt;td&#38gt;%parm[#1]%

&#38lt;/td&#38gt;&#38lt;td&#38gt;&#38lt;p;&#38gt;</p&#38gt;&#38lt;/td;&#38gt;&#38lt;/tr&#38gt;&#38lt;tr&#38gt;&#38lt;td&#38gt;&#38lt;b&#38gt;sysName&#38lt;/b&#38gt;&#38lt;/td&#38gt;&#38lt;td&#38gt;%parm[#2]%
&#38lt;/td&#38gt;&#38lt;td&#38gt;&#38lt;p;&#38gt;&#38lt;/p&#38gt;&#38lt;/td;&#38gt;&#38lt;/tr&#38gt;&#38lt;tr&#38gt;&#38lt;td&#38gt;&#38lt;b&#38gt;
c3800SysTrapSeverity&#38lt;/b&#38gt;&#38lt;/td&#38gt;&#38lt;td&#38gt;%parm[#3]%&#38lt;/td&#38gt;&#38lt;td&#38gt;&#38lt;p;&#38gt;

clear(1) minor(2) major(3)&#38lt;/p&#38gt;

&#38lt;/td;&#38gt;&#38lt;/tr&#38gt;&#38lt;tr&#38gt;&#38lt;td&#38gt;&#38lt;b&#38gt;
c3800SysAggregateStatus&#38lt;/b&#38gt;&#38lt;/td&#38gt;&#38lt;td&#38gt;%parm[#4]%&#38lt;/td&#38gt;&#38lt;td&#38gt;&#38lt;p;&#38gt;clear(1)minor(2)major(3) &#38lt;/td;&#38gt;&#38lt;/tr&#38gt;&#38lt;/table&#38gt;

</descr>

<logmsg dest='logndisplay'> Cisco Event: C3900: Node Status has changed. </logmsg>

<severity>Indeterminate</severity>

</event>

Το παραπάνω παράδειγμα είναι ένα γεγονός που αναφέρετε στην συσκευή C3800 της εταιρίας Cisco. Ένα μέρος του είναι όμοιο με το εσωτερικά γεγονότα με την κύρια διαφορά την ετικέτα <mask>. Μέσα στην ετικέτα <mask> ορίζουμε ετικέτες <maskelement> και όταν ληφθεί ένα SNMP trap που να ταιριάζει με τα <maskelement> τότε δημιουργείτε το γεγονός. Στο συγκεκριμένο παράδειγμα το γεγονός θα ταιριάξει με ένα SNMP trap του οποίου το enterprise OID είναι το ".1.3.6.1.4.1.9.9.70.2" η γενική τιμή του trap είναι 6 και η συγκεκριμένη τιμή του είναι το 17.

Οι τιμές που μπορεί να περιλάβει η ετικέτα <mename> είναι:

- uei
- source
- host
- snmphost
- nodeid
- interface
- service
- id
- specific
- generic
- community

Με αυτές τις παραμέτρους μπορούμε να ταιριάξουμε SNMP traps με γεγονότα. Μπορούμε δηλαδή να φτιάξουμε τα δικά μας γεγονότα και να τα ταιριάξουμε με SNMP traps.

7.6. - Δημιουργία ορισμών γεγονότων από ορισμούς SNMP traps σε mib αρχεία

Το OpenNMS έχει τη δυνατότητα να λαμβάνει SNMP traps. Για να μπορεί να καταλάβει και να διερμηνεύσει ένα trap πρέπει να υπάρχει ο αντίστοιχος ορισμός μέσα στο αρχείο eventconf.xml. Σε αντίθεση με άλλα συστήματα που μπορούμε να εισάγουμε τα mib αρχεία κατευθείαν εδώ πρέπει να μεταφραστούν σε κώδικα xml και να εισαχθούν στο αρχείο eventconf.xml. Εάν δεν υπάρχει ο ορισμός μέσα στο αρχείο τότε λαμβάνουμε μηνύματα σαν το παρακάτω.

Received unformatted enterprise event (enterprise:.1.3.6.1.4.1.2021.250.10 generic:0 specific:0). 0 args:

Το να ορίζουμε όλους αυτούς τους ορισμούς χειροκίνητα είναι αρκετά δύσκολο και επιρρεπές σε λάθη. Για αυτό υπάρχει το εργαλείο mib2opennms που μπορεί να κάνει αυτόματα την μετατροπή από την σύνταξη SMI που χρησιμοποιούν τα αρχεία mib σε κώδικα XML.

7.7. mib2opennms

Στο λειτουργικό σύστημα DEBIAN LENNY 5.03 μπορούμε να εγκαταστήσουμε το εργαλείο αυτό εκτελώντας την εντολή:

apt-get install mib2opennms

Για να μετατρέψουμε ένα απλό αρχείο mib εκτελούμε:
mib2opennms -6 *mibfile.mib* >*mibfile*.events.xml

Μετά πρέπει να επεξεργαστούμε το αρχείο mibfile.events.xml και να προσθέσουμε στην πρώτη γραμμή την ετικέτα <events> και στο τέλος την ετικέτα </events> και το μεταφέρουμε στον κατάλογο /etc/opennms/events.

Στο αρχείο eventsconf.xml προσθέτουμε την γραμμή:
<event-file>events/*mibfile*.events.xml</event-file>

Στο αρχείο eventconf.xml η γραμμή
<event-file>events/default.events.xml</event-file>
πρέπει να είναι πάντα τελευταία. Τέλος κάνουμε επανεκκίνηση το OpenNMS και τα γεγονότα είναι έτοιμα να δουλέψουν.

7.8. - Επίλογος

Σε αυτό το κεφάλαιο παρουσιάστηκε το αρχείο event.conf το οποίο είναι υπεύθυνο για την λειτουργία του δαίμονα Eventd ο οποίος διαχειρίζεται τα γεγονότα του OpenNMS. Αναφερθήκαμε στις δύο κατηγορίες γεγονότων οι οποίες είναι τα εσωτερικά (γεγονότα που παράγονται από το ίδιο το OpenNMS) και τα εξωτερικά (SNMP traps η άλλα μηνύματα προς τον Eventd). Επίσης αναφερθήκαμε στις δριμύτητες των γεγονότων και στην διαφορετική σημασία τους. Τέλος αναφερθήκαμε στον τρόπο με τον οποίο μπορούμε να εισάγουμε MIB αρχεία στο OpenNMS για την μετάφραση των SNMP traps και πως μετατρέπουμε τα αρχεία MIB σε XML με το εργαλείο **mib2opennms.**

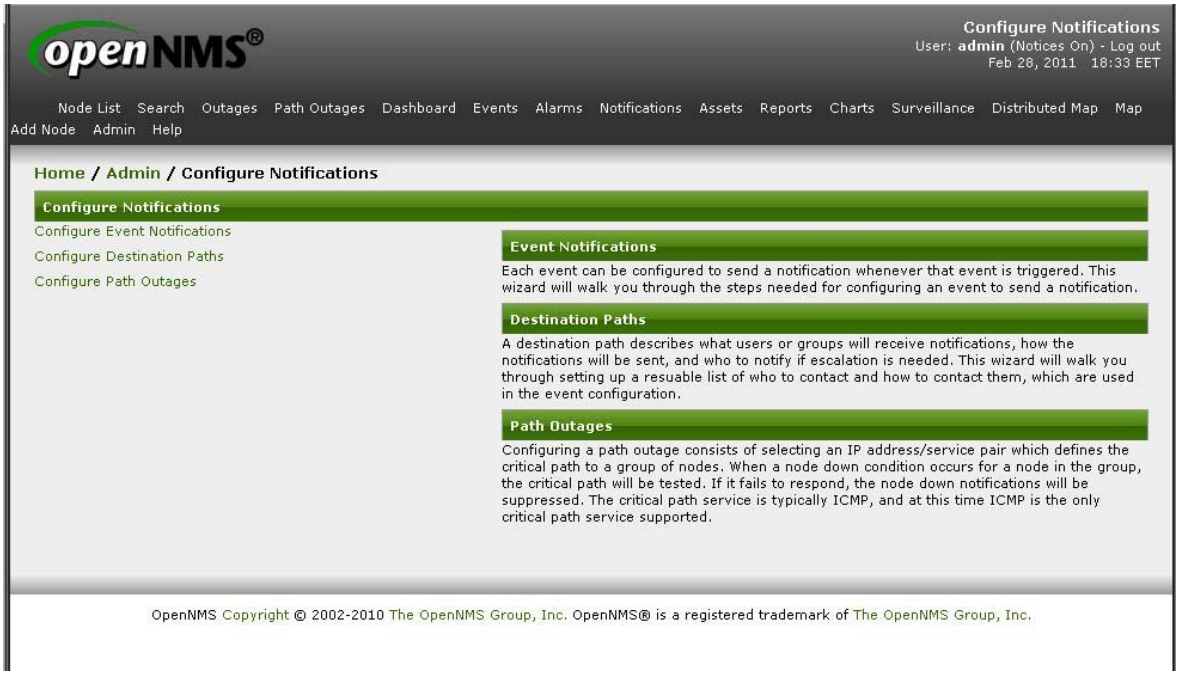
ΚΕΦΑΛΑΙΟ 8

Ειδοποιήσεις (Notifications)

Το OpenNMS χρησιμοποιεί τις ειδοποιήσεις για να ενημερώσει τους χρήστες για κάποιο γεγονός. Κοινές μέθοδοι ειδοποίησης είναι το ηλεκτρονικό ταχυδρομείο και συσκευές τηλεειδοποίησης, αλλά παρέχονται και οι εξής μηχανισμοί:

- XMPP (Jabber, πρωτόκολλο άμεσων μηνυμάτων)
- Εξωτερικά προγράμματα
- SNMP traps
- HTTP GETs/POSTs μπορούν να σταλθούν σε μια ιστοσελίδα.

Μια ειδοποίηση μπορεί να σταλθεί σε ένα χρήστη, σε ένα group χρηστών, σε κάποιο ρόλο που έχουμε ορίσει η αυθαίρετα σε μια διεύθυνση ηλεκτρονικού ταχυδρομείου. Μπορούμε να προσθέσουμε μια χρονική καθυστέρηση στην αποστολή μιας ειδοποίησης. Οι ειδοποιήσεις περιέχουν κείμενο και μερικές φορές και ένα θέμα τα οποία μπορούν να ρυθμιστούν ώστε να περιέχουν το γεγονός που ενεργοποίησε την ειδοποίηση, τον όνομα του κόμβου, διεύθυνση IP, υπηρεσία κ.α. Οι ειδοποιήσεις μπορούν να ρυθμιστούν μέσα από το γραφικό περιβάλλον πηγαίνοντας στην κατηγορία Admin>Configure Notifications.



Σχήμα 8.1 – Το μενού ρυθμίσεων των ειδοποιήσεων του OpenNMS

8.1. - Αρχεία ρυθμίσεων

Οι ειδοποιήσεις διαχειρίζονται από τον «δαίμονα» notifd. Ο notifd είναι προκαθορισμένο να ξεκινά αυτόματα, και ή συμπεριφορά του καθορίζετε από τα παρακάτω αρχεία ρυθμίσεων.

destinationPaths.xml

Ρυθμίζει διαδρομές προορισμών και καθορίζει ποιος θα ειδοποιηθεί.

notifd-configuration.xml

Ρυθμίζει γενικές ιδιότητες για τον «δαίμονα» ειδοποιήσεων όπως λεπτομέρειες για την επεξεργασία μιας ουράς ειδοποιήσεων και των αυτόματων βεβαιώσεων λήψης.

notificationCommands.xml

Ρυθμίζει τις μεθόδους ειδοποιήσεων όπως το ηλεκτρονικό ταχυδρομείο, XMPP, SNMP traps, κ.α. Παρόλο που ονομάζετε notification commands, δεν εκτελεί μόνο εξωτερικές εντολές αλλά και κλάσεις της Java η οποίες υλοποιούν μια πράξη ειδοποίησης. Η μέθοδοι ειδοποίησης της Java προτιμώνται γιατί έχουν καλύτερη απόδοση από το να καλέσουμε ένα εξωτερικό πρόγραμμα. Υπάρχει ένα πρότυπο interface το org.opennms.netmgt.notifd.NotificationStrategy το οποίο μπορούμε να χρησιμοποιήσουμε για να υλοποιήσουμε δικές μας μεθόδους ειδοποίησης. Επίσης μπορούμε να καλέσουμε προγράμματα γραμμής εντολών και shell scripts.

notifications.xml

Ρυθμίζει τις τρέχοντες ειδοποιήσεις.

8.2. - Λειτουργία ειδοποιήσεων

Ακολουθούνται τα εξής βήματα:

1) Στην εκκίνηση, ο «δαίμονας» notifd φτιάχνει μια λίστα από UEIs (Unique Event Identifier) γεγονότων για τα οποία ενδιαφέρεται βασιζόμενος στο αρχείο notifications.xml και συμφωνεί με τον «δαίμονα» γεγονότων eventd ώστε να λαμβάνει τα συγκεκριμένα γεγονότα.

2) Όταν το γεγονός ληφθεί, τα παρακάτω βήματα γίνονται.

- Είναι οι ειδοποιήσεις ενεργοποιημένες (η παράμετρος "status" στο "notifd-configuration" στοιχείο στο αρχείο notifd-configuration.xml); Εάν δεν είναι το γεγονός απορρίπτεται και καμία ειδοποίηση δεν συμβαίνει.
- Ταιριάζει το UEI του γεγονότος με ένα UEI που είναι ρυθμισμένο σε κάποια ενεργή ειδοποίηση; Εάν όχι το γεγονός απορρίπτεται και καμία ειδοποίηση δεν συμβαίνει. Το ειδικό αλφαριθμητικό UEI “MATCH-ANY-UEI” μπορεί να χρησιμοποιηθεί ώστε να ταιριάζει με όλα τα UEIs.
- Εάν η ειδοποίηση έχει κάποιο <varbind> ρυθμισμένο με όνομα και τιμή, χρησιμοποιείτε για να ταιριάξει με την αρχή του γεγονότος με το ίδιο όνομα.

3) Εάν τα παραπάνω βήματα ολοκληρωθούν τότε μια ή περισσότερες ειδοποιήσεις στέλνονται. Εάν η παράμετρος “match-all” στο αρχείο “notifd-confgiuration.xml” είναι αληθής κάθε ειδοποίηση που ταιριάζει θα εκτελεστεί, αλλιώς μόνο η πρώτη που θα βρεθεί θα εκτελεστεί.

8.3. - Διαδρομές προορισμών

Στο OpenNMS μία διαδρομή προορισμού καθορίζει το ποιος, πότε και πως μίας ειδοποίησης. Καθορίζει τους παραλήπτες μιας ειδοποίησης, την μέθοδο ειδοποίησης, και μια αρχική καθυστέρηση. Όταν ένα γεγονός λαμβάνετε του οποίου το UEI ταιριάζει σε μια ενεργή ειδοποίηση το OpenNMS διασχίζει την διαδρομή προορισμού για αυτή την ειδοποίηση. Λέμε ότι η διαδρομή προορισμού διασχίζετε γιατί είναι συνήθως μία σειρά από πράξεις που εκτελούνται εάν και μπορεί να είναι μόνο μία. Από την στιγμή που η διαδρομή προορισμού ξεκινήσει περιμένει μια αρχική καθυστέρηση (προκαθορισμένη: 0 δευτερόλεπτα) πριν στείλει την πρώτη ειδοποίηση.

8.4. - Στοιχεία μιας ειδοποίησης

Name

Ένα μοναδικό όνομα που καθορίζει μια ειδοποίηση. Χρησιμοποιείτε για την αναγνώριση στο webUI και στα μηνύματα καταγραφής. Αποθηκεύεται στην παράμετρο “name” του στοιχείου “notification” στο XML αρχείο ρυθμίσεων.

Event

Το UEI του γεγονότος που προκάλεσε την ειδοποίηση. Αποθηκεύεται στο στοιχείο “uei” στο XML αρχείο ρυθμίσεων.

Description

Μια περιγραφή του γεγονότος. Αποθηκεύεται στο στοιχείο “description” στο XML αρχείο ρυθμίσεων.

Rule

Ένα φίλτρο που πρέπει να ταιριάζει για να σταλθεί η ειδοποίηση. Συχνά είναι μια IP διεύθυνση ή μια υπηρεσία. Αποθηκεύεται στο στοιχείο “rule” στο XML αρχείο ρυθμίσεων.

Destination Path

Η διαδρομή προορισμού στην οποία η ειδοποίηση θα σταλθεί εάν ένα γεγονός ληφθεί και το φίλτρο ταιριάζει. Το όνομα της διαδρομής προορισμού αποθηκεύετε στην παράμετρο “destinationPath” στο XML αρχείο ρυθμίσεων και πρέπει να ταιριάζει μια διαδρομή προορισμού στο αρχείο destinationPaths.xml.

Subject

Το θέμα της ειδοποίησης, συγκεκριμένα, το θέμα των μηνυμάτων του ηλεκτρονικού ταχυδρομείου που δημιουργούνται από αυτή την ειδοποίηση.

Text message

Το μήνυμα κειμένου της ειδοποιήσεως. Αποθηκεύεται στο στοιχείο “text-message” στο XML αρχείο ρυθμίσεων.

On/off

Καθορίζει εάν οι ειδοποιήσεις είναι ενεργοποιημένες ή όχι. Αποθηκεύεται στην παράμετρο “status” του στοιχείου “notification” στο XML αρχείο ρυθμίσεων.

8.5. - Βεβαίωση λήψης (Acknowledgment)

Ο notifd συνεχίζει να διασχίζει το μονοπάτι προορισμού μιας ειδοποίησης μέχρι σταλθεί μια επιβεβαίωση λήψης για την συγκεκριμένη ειδοποίηση. Η επιβεβαίωση λήψης γίνετε από τον χρήστη μέσα από το webUI. Από την στιγμή που γίνει η επιβεβαίωση κανένας άλλος χρήστης ή ομάδα χρηστών δεν θα ειδοποιηθεί για την συγκεκριμένη ειδοποίηση.

8.6. - Αυτόματη βεβαίωση λήψης (Automatic Acknowledgment)

Πολλά γεγονότα τα οποία αντιπροσωπεύουν μια διακοπή στην λειτουργία ενός κόμβου επίσης έχουν ένα αντίστοιχο γεγονός το οποίο στέλνεται όταν η διακοπή λήξει και το πρόβλημα λυθεί. Ένα παράδειγμα είναι τα γεγονότα "nodeDown" και "nodeUp". Μόλις συμβεί το γεγονός “nodeUp” τότε θα γίνει αυτόματα η βεβαίωση λήψης του γεγονότος "nodeDown" και το μονοπάτι προορισμού θα σταματήσει να διασχίζεται. Επίσης θα δημιουργηθεί μία νέα ειδοποίηση ώστε να ενημερώσει τους χρήστες ότι το αρχικό πρόβλημα λύθηκε. Παρακάτω είναι ένα δείγμα από τις ρυθμίσεις από το ζευγάρι γεγονότων nodeUp/nodeDown στο notifd-configuration.xml:

```
<auto-acknowledge resolution-prefix="RESOLVED: "
                  uei="uei.opennms.org/nodes/nodeUp"
                  acknowledge="uei.opennms.org/nodes/nodeDown">
  <match>nodeid</match>
</auto-acknowledge>
```

Εδώ το στοιχείο “match” ορίζει ότι το nodeid πρέπει να είναι το ίδιο και στα δύο events για να γίνει η αυτόματη βεβαίωση λήψης.

8.7. - Επίλογος

Στο κεφάλαιο αυτό αναφερθήκαμε στις διάφορες ειδοποιήσεις που μπορεί να μας παρέχει το OpenNMS και στα αρχεία τα οποία είναι υπεύθυνα για την διαχείρισή τους τα οποία είναι τα destinationPaths.xml, notifd-confguration.xml , notificationCommands.xml και το notifications.xml. Επίσης αναφέραμε τα στοιχεία τα οποία μπορεί να μας αναφέρει η κάθε ειδοποίηση και τι μας προσφέρει το καθένα, τι είναι η επιβεβαιώσείς λήψης καθώς και πως δουλεύει η αυτόματη επιβεβαίωση λήψης σε περίπτωση που ένα πρόβλημα η μία βλάβη ανακάμψει από μόνη της.

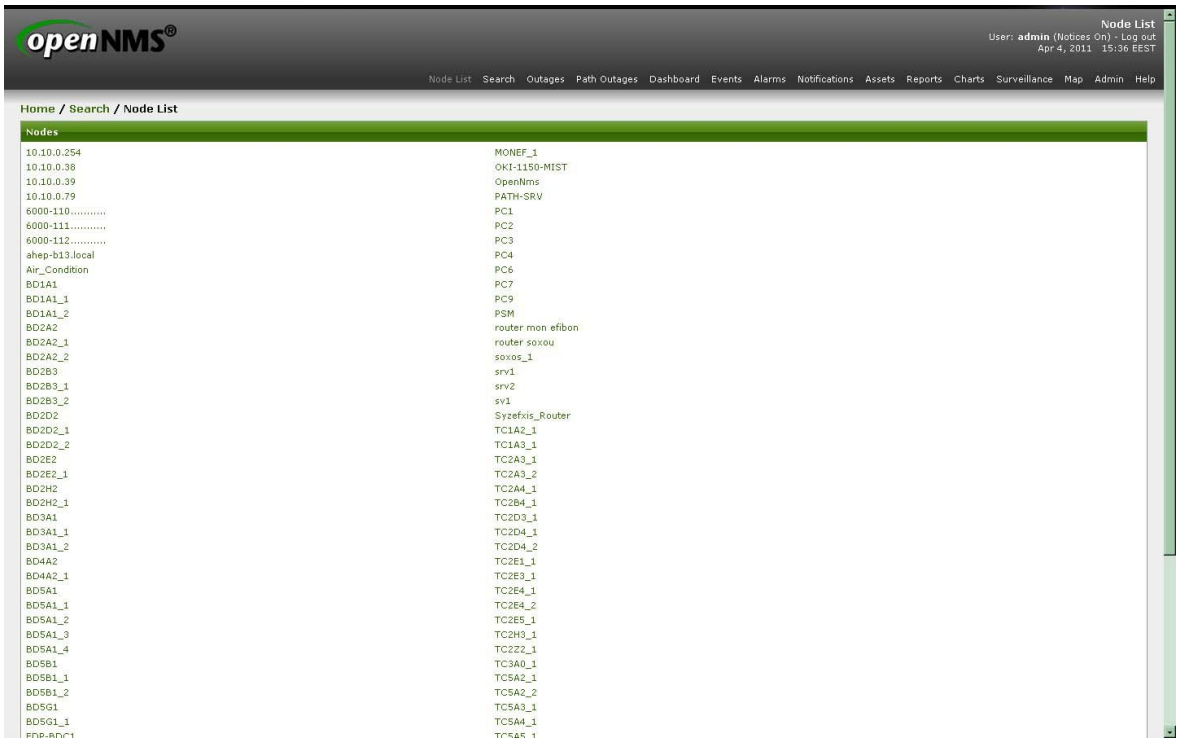
ΚΕΦΑΛΑΙΟ 9

Το Web Interface

Στο κεφάλαιο αυτό θα παρουσιαστούνε κάποια μέρη του WebInterface του OpenNMS που αποτελούνε βασικά εργαλεία για την επόπτευση του δικτύου με μια ματιά.

Η λίστα των κόμβων

Πατώντας στο Nodelist στο κεντρικό μενού μπορούμε να δούμε όλους τους κόμβους συνολικά.



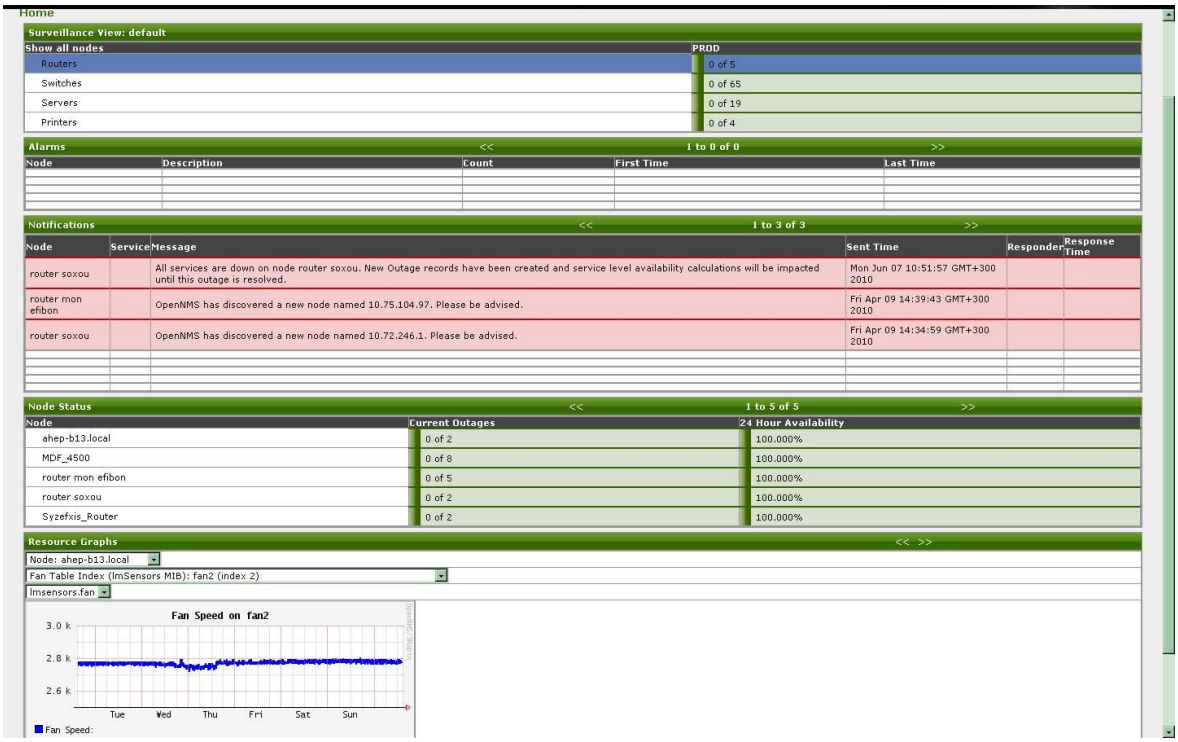
Σχήμα 9.1 – Η λίστα των κόμβων

Πατώντας σε κάποιο κόμβο μπορούμε να δούμε περαιτέρω πληροφορίες για αυτόν και πατώντας στον σύνδεσμο show interfaces τέρμα κάτω μπορούμε να δούμε και ποιες διασυνδέσεις έχει ο κόμβος καθώς και ποιες IP διευθύνσεις έχουν.

Ο πίνακας επόπτευσης (Dashboard)

Πατώντας στο Dashboard στο κεντρικό μενού εμφανίζετε ένας πίνακας επόπτευσης για τους κόμβους. Εφόσον τους έχουμε χωρίσει σε κατηγορίες

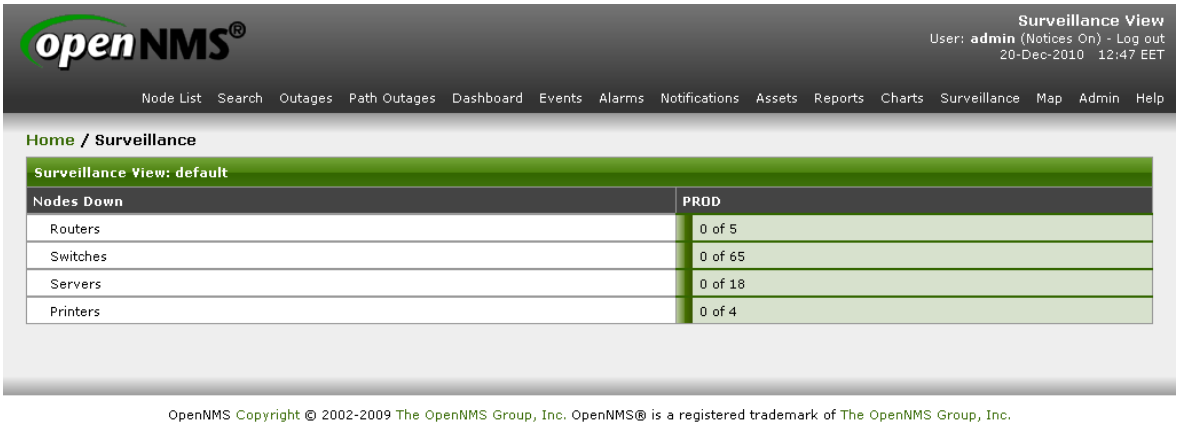
ανάλογα με την λειτουργία τους μπορούμε επιλέγοντας την κάθε κατηγορία να βλέπουμε τα γεγονότα, τις ειδοποιήσεις, την κατάσταση των κόμβων. Τέρμα κάτω μπορούμε να επιλέγουμε διάφορα διαγράμματα από κόμβους για γρήγορη επίβλεψη. Ο πίνακας επόπτευσης είναι ένα αρκετά χρήσιμο εργαλείο γιατί μας δίνει την δυνατότητα με μια ματιά να βλέπουμε σε τη κατάσταση βρίσκεται το δίκτυο.



Σχήμα 9.2 – Ο πίνακας επόπτευσης

Οι κατηγορίες των κόμβων (Surveillance)

Πατώντας στο Surveillance στο κεντρικό μενού μπορούμε να δούμε τις κατηγορίες των κόμβων καθώς και πόσους κόμβους περιέχει η καθεμία τους.



Σχήμα 9.3 – Οι κατηγορίες των κόμβων

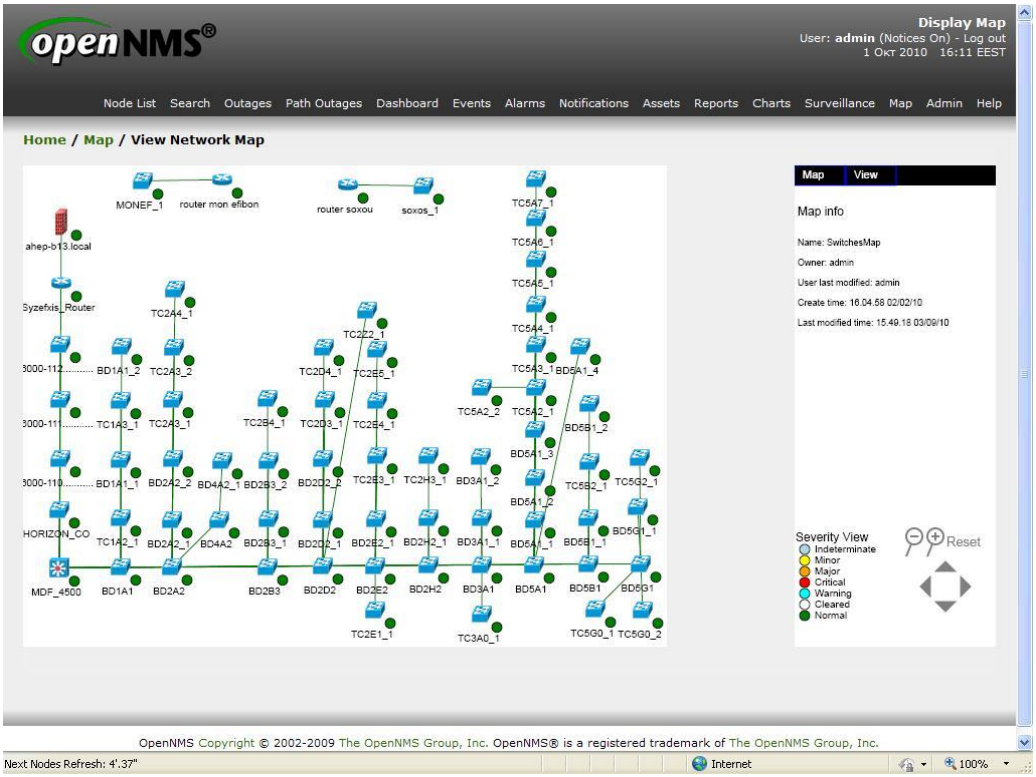
Διαγράμματα

Πατώντας στο charts στο κεντρικό μενού μπορούμε να δούμε τρία βασικά διαγράμματα ένα το οποίο μας δείχνει τον αριθμό των γεγονότων και των συναγερμών, τις διακοπές λειτουργίας σε υπηρεσίες και ποιες υπηρεσίες είχαν τις περισσότερες καθώς και των αριθμό των κόμβων, των interfaces και των υπηρεσιών που διαχειριζόμαστε συνολικά.

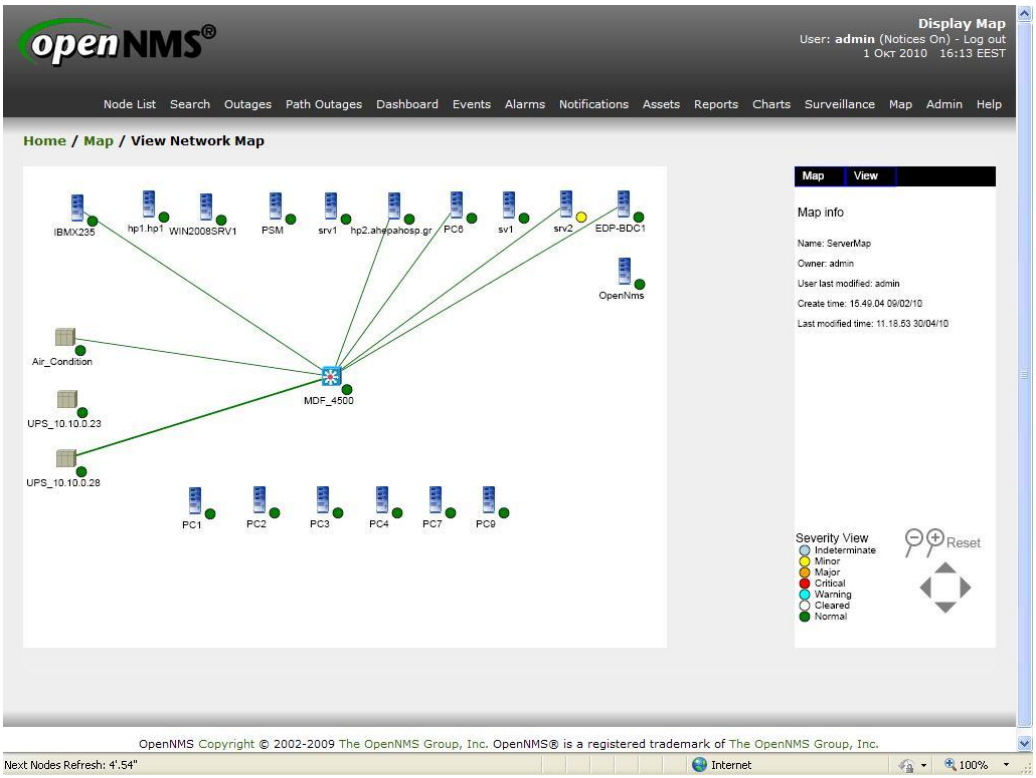
Σχήμα 9.4 – Βασικά διαγράμματα στο OpenNms

Ο χάρτης του OpenNMS

Ο χάρτης είναι ένα από τα βασικότερα εργαλεία για την διαχείριση του δικτύου γιατί μας δίνει την δυνατότητα να δούμε με μια ματιά εάν υπάρχει κανένα πρόβλημα στο δίκτυο που διαχειριζόμαστε. Για την υποδομή του νοσοκομείου Αχέπα κατασκευάστηκαν τρεις διαφορετικοί χάρτες για την εποπτεία των κατανεμητών, των εξυπηρετητών και κάποιων εκτυπωτών. Στον χάρτη εκτός από τους κόμβους παριστάνονται και οι συνδέσεις μεταξύ των κόμβων οι οποίες ανακαλύπτονται αυτόματα από των δαίμονα Linkd, καθώς και η κατάσταση που βρίσκεται ο κάθε κόμβος. Παρακάτω παρουσιάζονται οι δύο από αυτούς.



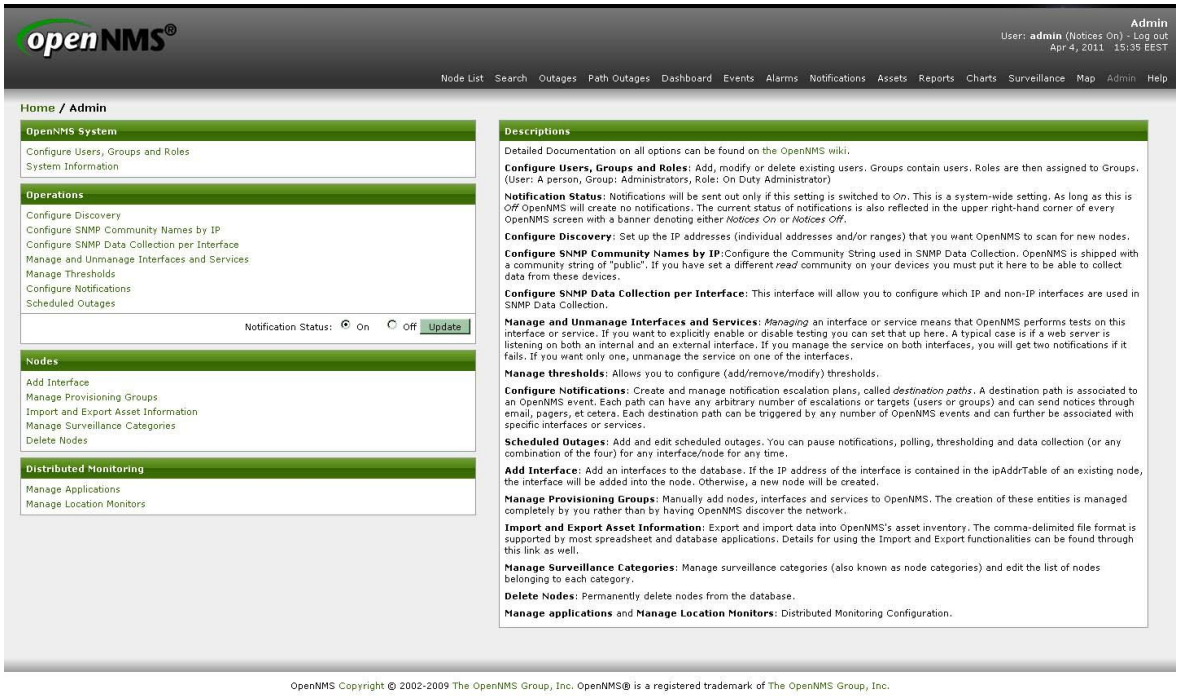
Σχήμα 9.5 – Ο χάρτης των κατανεμητών



Σχήμα 9.6 – Ο χάρτης των εξυπηρετητών

Το μενού διαχείρισης

Πατώντας στο admin στο κεντρικό μενού μπορούμε να δούμε το βασικό μενού διαχείρισης του OpenNMS. Παρουσιάζεται παρακάτω και ακολουθούν κάποιες διευκρινίσεις για το τι μπορούμε να διαχειριστούμε μέσα από αυτό το μενού.



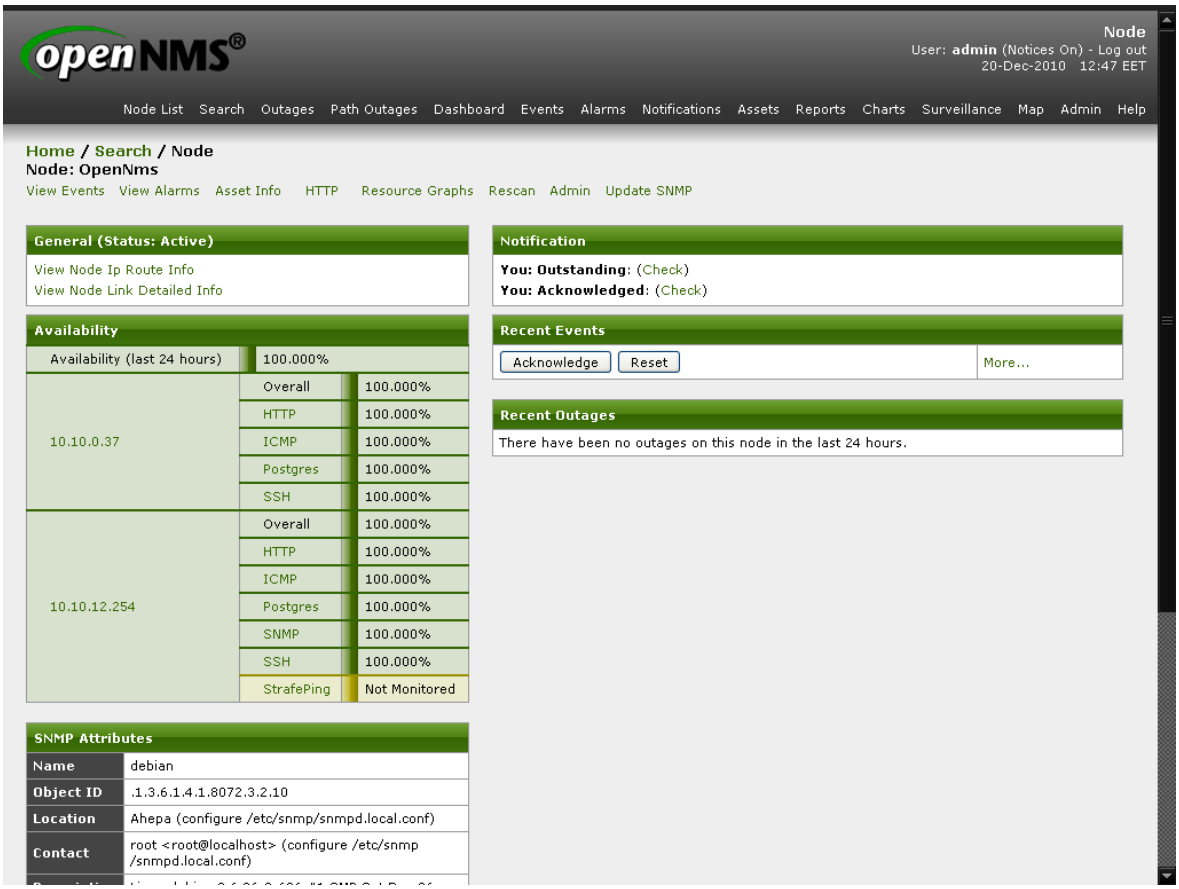
Σχήμα 9.7 – Το μενού διαχείρισης

- Προσθήκη χρηστών και ομάδες χρηστών καθώς και διάφορους ρόλους που μπορούνε να έχουνε.
- Παρουσίαση πληροφοριών σχετικά με το σύστημα
- Επιλογή διευθύνσεων που το OpenNms θα ψάχνει για συσκευές
- Μπορούμε να θέσουμε διαφορετικές συμβολοσειρές community για διαφορετικές συσκευές.
- Μπορούμε να επιλέξουμε τι πληροφορίες θα συλλέγουμε από τον κάθε κόμβο.
- Μπορούμε να επιλέξουμε εάν θέλουμε να διαχειριζόμαστε μια διασύνδεση η μία υπηρεσία μεμονωμένα.
- Μπορούμε να ορίσουμε όρια για τιμές που συλλέγουμε και να ειδοποιούμαστε εάν αυτά τα όρια παραβιάζονται.
- Ορίζουμε την συμπεριφορά των ειδοποιήσεων και εάν θα είναι ενεργές η όχι.
- Μπορούμε να καθορίσουμε προγραμματισμένες διακοπές.

- Μπορούμε να εισάγουμε απευθείας μια διασύνδεση
- Διαγραφή κόμβων
- Δημιουργία κατηγοριών κόμβων

Οι πληροφορίες του κόμβου

Μπαίνοντας σε ένα κόμβο είτε μέσα από την λίστα των κόμβων είτε πατώντας πάνω του στον χάρτη βλέπουμε συγκεκριμένες πληροφορίες για αυτόν.



Σχήμα 9.8 – Οι πληροφορίες κόμβου του NMS

Στις πληροφορίες του κόμβου μπορούμε να δούμε τις διασυνδέσεις που έχει ο κόμβος καθώς και τι υπηρεσίες παρέχει η κάθε διασύνδεση, τις ειδοποιήσεις, τα γεγονότα και τις πρόσφατες διακοπές λειτουργίας του. Επίσης μπορούμε να δούμε κάποιες παραμέτρους του πρωτόκολλου SNMP.

Τρίτη Ενότητα

Παραμετροποίηση του OpenNMS και ανάπτυξη εργαλείων για το περιβάλλον του νοσοκομείου Αχέπα

ΚΕΦΑΛΑΙΟ 10

Εγκατάσταση και ρυθμίσεις του OpenNms στο λειτουργικό σύστημα Debian 5.03 Lenny GNU/Linux

10.1. - ΕΛΑΧΙΣΤΕΣ ΑΠΑΙΤΗΣΕΙΣ ΣΥΣΤΗΜΑΤΟΣ

Αν και είναι αδύνατο να προσδιορίσουμε ακριβώς τις απαιτήσεις του OpenNMS σε ένα συγκεκριμένο περιβάλλον, αυτά που ακολουθούν αποτελούν τις ελάχιστες απαιτήσεις για την εγκατάστασή του, υποθέτοντας ότι θα διαχειριστούμε ένα δίκτυο που αποτελείται από 200 κόμβους.

Επεξεργαστής

1 GHz Pentium III(ή αντίστοιχος) ή καλύτερος. Το OPENNMS μπορεί να διαχειριστεί το πλεονέκτημα των πολλαπλών επεξεργαστών.

Μνήμη

Αν και σαν ελάχιστη μνήμη μπορούμε να χρησιμοποιήσουμε 256MB RAM τα 512MB είναι προτεινόμενα. Το OpenNMS Java Virtual Machine ωφελείται από μεγάλες χωρητικότητες μνήμης της τάξεως των 2GB και με την χρήση 64bit επεξεργαστών.

Χωρητικότητα δίσκου

Το OpenNMS απαιτεί περίπου 200MB χώρο στο δίσκο για τα αρχεία του προγράμματος. Επιπλέον, κάθε μεταβλητή συλλεχθέντος στοιχείου απαιτεί, εξ ορισμού, λίγο κάτω από 300 KB χώρο στο δίσκο. Είναι ασφαλές να υποθέσουμε ότι κάθε διεπαφή που διαχειρίζεται θα απαιτήσει περίπου 2MB του χώρου στο δίσκο, έτσι για 200 διεπαφές χρειαζόμαστε 400MB (συντηρητικά). Ανάλογα με τον αριθμό γεγονότων που αποθηκεύονται, μπορούμε να υποθέσουμε ότι 100MB με 200MB απαιτούνται για τη βάση δεδομένων. Τέλος, τα log αρχεία του OpenNMS μπορούν να γίνουν αρκετά μεγάλα, ειδικά σε debug mode.

10.2. - Εγκατάσταση του OpenNMS

Μετά την εγκατάσταση του λειτουργικού συστήματος για να εγκαταστήσουμε το OpenNMS ακολουθούμε τα εξής βήματα:

1) Προσθέτουμε στο αρχείο /etc/apt/sources.list τις παρακάτω γραμμές ώστε να ξέρει που θα το βρει το πρόγραμμα εγκατάστασης apt-get.

deb http://debian.opennms.org stable main
deb-src http://debian.opennms.org stable main

2) Η βιβλιοθήκη του OpenNMS για το Debian υπογράφεται με ένα κλειδί PGP (fingerprint 22EE DDA6 8698 B02F B2EC 50B7 062B 8A68 4C4C BBD9). Θα πρέπει να πείτε στο πρόγραμμα εγκατάστασης apt-get για το κλειδί.

wget -O - http://debian.opennms.org/OPENNMS-GPG-KEY | sudo apt-key add

3) Εφόσον η java της Sun ορίστηκε υπό την GPL μπορούμε εκτελώντας την εντολή "apt-get install opennms" από το Debian 4.0 (etch) ή Ubuntu 7.10 (gutsy) και μετά να την εγκαταστήσουμε. Οι εξαρτήσεις μεταξύ των πακέτων θα συμπληρωθούν από μονές τους προθέτοντας και την Java. Κάνουμε ενημέρωση τη λίστα των πακέτων και εγκαθιστούμε το OpenNMS.

apt-get update
apt-get install opennms

Κατά την διάρκεια της εγκατάστασης θα ερωτηθούμε εάν θέλουμε να εγκαταστήσουμε το opennms-webapp-jetty (embedded web server) ή το opennms-webapp-standalone (tomcat for the web server) πακέτο. Η προεπιλογή είναι το opennms-webapp-jetty το οποίο είναι και προτεινόμενο εκτός εάν υπάρχει η ανάγκη να εκτελέσουμε το OpenNMS σε ένα εξωτερικό servlet container.

Σε παλαιότερες εκδόσεις του Debian χρειάζεται να εγκαταστήσουμε το Java 6 JDK ξεχωριστά γιατί δεν περιλαμβάναται στην εγκατάσταση του OpenNMS.

10.3. - Ρυθμίσεις εγκατάστασης του OpenNMS

Κατ' αρχάς, για λόγους ευκολίας, πρόκειται να ορίσουμε τη μεταβλητή περιβάλλοντος **$OPENNMS_HOME** πριν τρέξουμε οποιεσδήποτε εντολές.
export OPENNMS_HOME=/usr/share/opennms

Ρυθμίσεις της βάσης δεδομένων

Το OpenNMS πρέπει να είναι σε θέση να συνδεθεί με τη βάση δεδομένων PostgreSQL ως " postgres" χρήστης (εξ ορισμού) μέσα από μια σύνδεση TCP/IP. Χρειάζεται να τροποποιήσουμε δύο αρχεία τα οποία βρίσκονται στο /etc/postgresql/X.X/main κατάλογο. (οπού X.X η έκδοση της Postgres). Εκκινούμε τη βάση:

/etc/init.d/postgresql-X.X restart

Τροποποιούμε το αρχείο pg_hba.conf (το οποίο δημιουργείται είτε κατά την εγκατάσταση της Postgres είτε την πρώτη φορά που θα εκτελεστεί) για να επιτρέψουμε στο χρήστη postgres να αυθεντικοποιηθεί στη βάση δεδομένων.
Εξ' ορισμού, θα έχει κάτι παρόμοιο στο κατώτατο σημείο:

local all all ident sameuser
host all all 127.0.0.1/32 ident sameuser
host all all ::1/128 ident sameuse

Θα πρέπει να αλλάξουμε το "ident sameuser" με το "trust"

local all all trust
host all all 127.0.0.1/32 trust
host all all ::1/128 trust

Τροποποιούμε το αρχείο postgresql.conf για να επιτρέψουμε τις συνδέσεις TCP/IP και να βελτιστοποιήσουμε την απόδοση της βάσης.

Σε παλαιότερες εκδόσεις της Postgres επιτρέπουμε τις συνδέσεις με το flag:
tcpip_socket = true

Σε νεότερες εκδόσεις επιτρέπουμε τις TCP/IP συνδέσεις με:
you can use "*" to listen on all addresses

listen_addresses = 'localhost'

Ορίζουμε το μέγιστο αριθμό συνδέσεων να είναι μεγαλύτερος από το **c3p0.maxPoolSize στο $OPENNMS_HOME/etc/c3p0.properties (50 εξ' ορισμού) +10**
max_connections = 60

Επίσης μπορούμε να κάνουμε κάποιες ρυθμίσεις ώστε να βελτιστοποιήσουμε την απόδοση της βάσης δεδομένων. Οι τιμές που καθορίσαμε για ένα σύστημα που έχει 2GB RAM είναι οι ακόλουθες:

shared_buffers = 20000
work_mem = 16348
maintenance_work_mem = 65536
vacuum_cost_delay = 50
checkpoint_segments = 20
checkpoint_timeout = 900
wal_buffers = 64
stats_start_collector = on
track_counts = on
autovacuum = on
(οι παραπάνω ρυθμίσεις ισχύουν για την έκδοση 8.3 της Postgres)

Σε αυτό το σημείο χρειάστηκε να αλλάξουμε μια παράμετρο του πυρήνα δίνοντας μεγαλύτερη τιμή στο kernel.shmmax ώστε να μπορεί το λειτουργικό σύστημα να φορτώσει ένα μεγαλύτερο κομμάτι μνήμης για τη βάση. Τροποποιούμε το αρχείο /etc/sysctl.sys και προσθέτουμε την εγγραφή:
kernel.shmmax = 176639360
Έπειτα κάνουμε επανεκκίνηση το σύστημα ώστε να πάρει τις νέες ρυθμίσεις ο πυρήνας.

Εφόσον έγιναν οι παραπάνω ρυθμίσεις κάνουμε επανεκκίνηση τη βάση δεδομένων.
/etc/init.d/postgresql-X.X restart

Δημιουργούμε την βάση που θα χρησιμοποιεί το OpenNMS με την παρακάτω εντολή:

sudo -u postgres createdb -U postgres -E UNICODE opennms

Εάν είναι η πρώτη φορά που εγκαθιστούμε το OpenNMS πρέπει να σιγουρευτούμε ότι το iplike είναι ρυθμισμένο στη βάση δεδομένων. Χρειάζεται να το εγκαταστήσουμε εκτελώντας την εντολή:

apt-get install iplike-pgsql83

(Για Debian-Based διανομές χρησιμοποιούμε το επίθεμα pgsql74, pgsql81, pgsql82,pgsql83 ανάλογα με την έκδοση που έχουμε εγκατεστημένη).

Εάν η εγκατάσταση δεν ολοκληρωθεί ή εάν δεν υπάρχει ήδη το iplike τότε πρέπει να το εγκαταστήσουμε χειροκίνητα εκτελώντας το script:

install_iplike.sh

Στο επόμενο βήμα χρειάζεται να “πούμε” στο openNMS που θα βρει την Java ώστε να μπορεί να ξεκινήσει. Πρώτα ορίζουμε το JRE της SUN σαν το προκαθορισμένο JRE για το σύστημα με την εντολή:

update-alternatives --config java

Η παρακάτω εντολή ψάχνει σε προκαθορισμένες τοποθεσίες για το JRE.

$OPENNMS_HOME/bin/runjava -s

Εάν θέλουμε να επιλέξουμε ένα μη προκαθορισμένο JRE μπορούμε να εκτελέσουμε την εντολή με την επιλογή –S:

$OPENNMS_HOME/bin/runjava -S /path_to_java_jre_from_sun

Στην περίπτωση που επιλέγουμε εμείς χειροκίνητα το JRE τότε θα πρέπει να προσθέσουμε την JAVA_HOME στο αρχείο /etc/default/opennms:

JAVA_HOME= path_to_java_jre_from_sun

10.4. - Αρχικοποίηση του OpenNMS και της βάσης δεδομένων

Πρώτα εκτελούμε το script εγκατάστασης το οποίο θα αρχικοποιήσει τη βάση δεδομένων και θα κάνει κάποιες βασικές ρυθμίσεις.

$OPENNMS_HOME/bin/install -dis

Το παραπάνω script χρειάζεται να το ξαναεκτελέσουμε σε περίπτωση που αναβαθμίσουμε το OpenNMS σε μια πιο νέα έκδοση.

Μερικές φορές μπορεί να πρέπει να πούμε στο OpenNMS πού θα βρει libjicmp.so. Στην περίπτωση αυτή, μπορούμε να χρησιμοποιήσουμε την –l επιλογή.
i386 example
$OPENNMS_HOME/bin/install -dis -l /usr/lib/jni:/usr/lib
x86_64 example
$OPENNMS_HOME/bin/install -dis -l /usr/lib64/jni:/usr/lib64

Εάν υπάρξει λάθος κατά τη διάρκεια τις εγκατάστασης που να λέει ότι η γλώσσα" plpgsql" δεν υπάρχει, εκτελούμε την εντολή:
createlang -U postgres plpgsql opennms

10.5. - Εκίνηση του openNMS

Μέχρι στιγμής έχουμε ολοκληρώσει την βασική εγκατάσταση και με την εντολή που ακολουθεί εκκινούμε το OpenNMS:
/etc/init.d/opennms start

Ανοίγουμε κάποιον φυλλομετρητή και συνδεόμαστε στη διεύθυνση :
http://yourhost:8980/opennms/ (Όπου yourhost η διεύθυνση ή το hostname του μηχανήματος που τρέχει το OpenNMS, username: admin, password: admin)

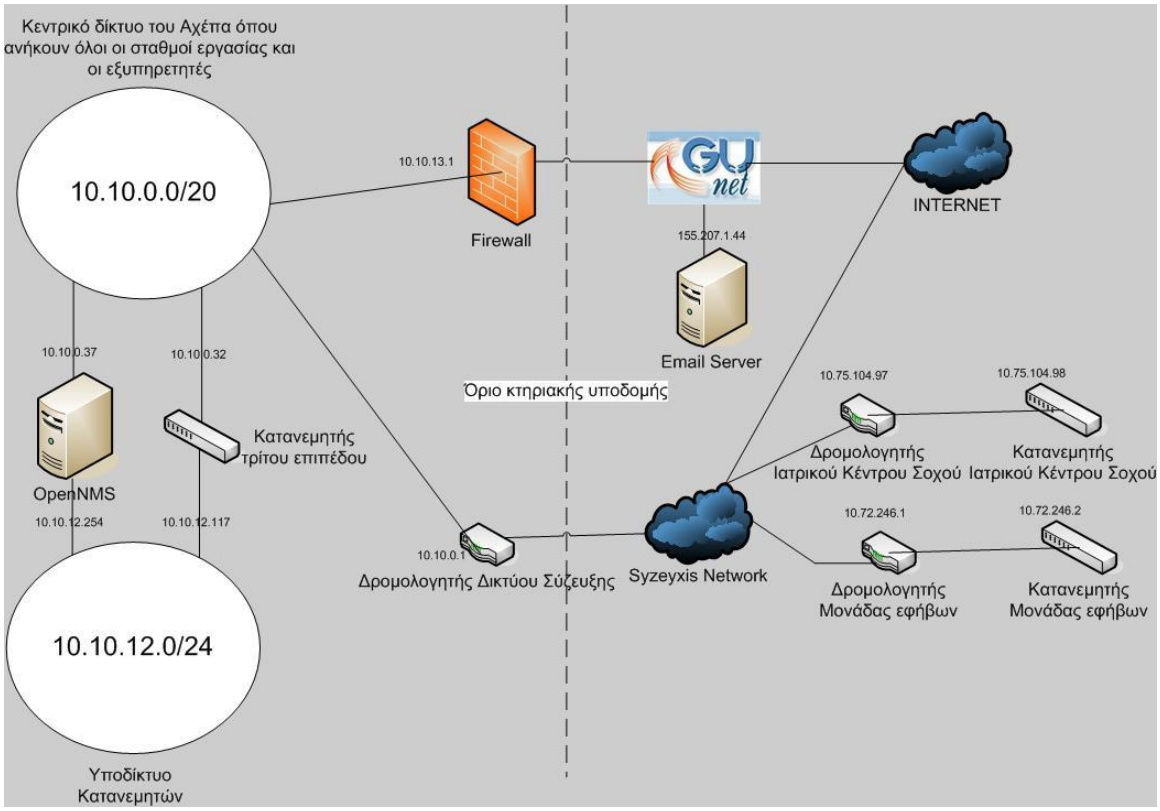
Σχήμα 10.1 – Αρχική οθόνη του OpenNms

ΚΕΦΑΛΑΙΟ 11

Τοπολογία του δικτύου

Στο κεφάλαιο αυτό θα παρουσιαστεί η τοπολογία του δικτύου του νοσοκομείου Αχέπα, οι ρυθμίσεις που έγιναν στην κάρτα δικτύου του NMS ώστε να μπορεί να επικοινωνεί με όλα τα λογικά υποδίκτυα του νοσοκομείου καθώς και με δύο δίκτυα τα οποία βρίσκονται εκτός της κτηριακής υποδομής του νοσοκομείου.

11.1. - Λογικό διάγραμμα του δικτύου



Σχήμα 11.1 – Λογική τοπολογία δικτύου

Στο Σχήμα 11.1 βλέπουμε την λογική τοπολογία του δικτύου του νοσοκομείου Αχέπα. Σίγουρα δεν αποτελεί μια πλήρης εικόνα του δικτύου γιατί έχουν συμπεριληφθεί μόνο τα στοιχεία τα οποία έπαιξαν κάποιο ρόλο στη διαχείριση του δικτύου σαν διαχειριζόμενα στοιχεία. Τα βασικά στοιχεία του παραπάνω σχήματος είναι:

- Δύο λογικά δίκτυα, το 10.10.0.0/20 και ένα υποδίκτυο του το 10.10.12.0/24 στο οποίο ανήκουν όλοι οι κατανεμητές. Το 10.10.0.0/20 περιλαμβάνει

όλους τους σταθμούς εργασίας του νοσοκομείου καθώς και τους εξυπηρετητές που παρέχουν τις διάφορες υπηρεσίες στο δίκτυο. Στους εξυπηρετητές ανατείθονται διευθύνσεις που αρχίζουν από 10.10.0.

- Ένας κατανεμητής τρίτου επιπέδου ο οποίος δρομολογεί πακέτα από το ένα λογικό δίκτυο στο άλλο.
- Ένας εξυπηρετητής ο οποίος εκτελεί το OpenNMS και περιέχει επίσης και κάποια εργαλεία δικτύου και εργαλεία διαχείρισης, δηλαδή το NMS του Δικτύου.
- Ένα δρομολογητής ο οποίος δρομολογεί πακέτα στο δίκτυο σύζευξης το οποίο είναι ένα δίκτυο όπου ενώνει διάφορες δημόσιες υπηρεσίες της χώρας.
- Το τείχος προστασίας το οποίο κάνει ταυτόχρονα και δρομολόγηση προς το πανεπιστημιακό δίκτυο GUNET.
- Ένα δρομολογητή και ένα κατανεμητή στο ιατρικό κέντρο Σοχού όπου αποτελεί παράρτημα του νοσοκομείου και η επικοινωνία γίνεται μέσω του δικτύου σύζευξης.
- Ένα δρομολογητή και ένα κατανεμητή στη μονάδα εφήβων όπου αποτελεί επίσης παράρτημα του νοσοκομείου και η επικοινωνία γίνεται μέσω του δικτύου σύζευξης.
- Ένα εξυπηρετητή ηλεκτρονικού ταχυδρομείου ο οποίος βρίσκεται στις εγκαταστάσεις του Αριστοτελείου Πανεπιστημίου και χρησιμοποιήθηκε για την αποστολή μηνυμάτων ειδοποίησης από το OpenNMS.

Τα στοιχεία που ρυθμίσαμε στο OpenNMS να διαχειρίζεται είναι όλα όσα ανήκουν στο υποδίκτυο 10.10.12.0/24 όπου εμπεριέχονται όλοι οι κατανεμητές, αρκετοί εξυπηρετητές, UPS, οι οποίοι ανήκουν στο δίκτυο 10.10.0.0/20 όπως επίσης και τους δρομολογητές και κατανεμητές στα δύο απομακρυσμένα δίκτυα στο ιατρικό κέντρο Σοχού και στην μονάδα εφήβων. Συνολικά αυτή την στιγμή γίνετε διαχείριση **19 εξυπηρετητών, 65 κατανεμητών, 4 εκτυπωτών και 5 δρομολογητών.**

11.2. - Το αρχείο interfaces στον εξυπηρετητή του OpenNMS

Όπως παρατηρούμε στο σχήμα 11.1 το OpenNMS ανήκει σε δύο διαφορετικά λογικά δίκτυα και για να το επιτύχουμε αυτό τροποποιήσαμε το αρχείο interfaces του λειτουργικού συστήματος πάνω στο οποίο τρέχει το OpenNMS έτσι ώστε να δημιουργήσουμε δύο λογικές διασυνδέσεις πάνω από μία φυσική διασύνδεση. Ο λόγος που χρησιμοποιήθηκαν δύο διαφορετικές λογικές διασυνδέσεις είναι ότι θέλαμε να διαχειριζόμαστε στοιχεία και από τα δύο λογικά δίκτυα χωρίς όμως να γίνει τροποποίηση τον ρυθμίσεων του κατανεμητή τρίτου επιπέδου. Το αρχείο interfaces παρουσιάζεται παρακάτω:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback
```

Ενεργοποίηση του loopback interface

```
# The primary network interface
auto eth0
iface eth0 inet static
        address 10.10.12.254
        netmask 255.255.255.0
        network 10.10.12.0
        broadcast 10.10.12.255
        gateway 10.10.12.117
# dns-* options are implemented by the resolvconf package, if installed
        dns-nameservers 10.10.0.21 10.10.0.22
        up route add -net 10.10.0.0 netmask 255.255.240.0 gw 10.10.12.117
2>/dev/null || true
        down route del -net 10.10.0.0 netmask 255.255.240.0 gw 10.10.12.117
2>/dev/null||true
```

Ενεργοποίηση του πρώτου λογικού interface με διεύθυνση IP 10.10.12.254 για το υποδίκτυο 10.10.12.0/24 και προσθήκη ενός κανόνα δρομολόγησης όπου ότι πακέτο προορίζεται για το δίκτυο 10.10.0.0/20 να στέλνεται στο

κατανεμητή τρίτου επιπέδου που έχει την διεύθυνση 10.10.12.117 και ενώνει λογικά τα δύο δίκτυα.

```
auto eth0:1
iface eth0:1 inet static
      address 10.10.0.37
       netmask 255.255.240.0
       network 10.10.0.0
       broadcast 10.10.15.255
# dns-* options are implemented by the resolvconf package, if installed
       up route add -host 155.207.1.44 gw 10.10.13.1 2>/dev/null||true
       down route del -host 155.207.1.44 gw 10.10.13.1 2>/dev/null||true
       up route add -net 10.75.104.96 netmask 255.255.255.224    gw 10.10.0.1
2>/dev/null||true
       down route del -host 10.75.104.96 netmask 255.255.255.224  gw 10.10.0.1
2>/dev/null||true
       up route add -net 10.72.246.0 netmask 255.255.254.0    gw 10.10.0.1
2>/dev/null||true
       down route del -host 10.72.246.0 netmask 255.255.254.0  gw 10.10.0.1
2>/dev/null||true
```

Ενεργοποίηση του δεύτερου λογικού interface με διεύθυνση IP 10.10.0.37 για το δίκτυο 10.10.0.0/20 και προσθήκη κανόνων δρομολόγησης όπου ότι πακέτο προορίζεται για την διεύθυνση 155.207.1.44 (εξυπηρετητής ηλεκτρονικού ταχυδρομείου) να προωθείται στο GUNET μέσω της πύλης 10.10.13.1. Επίσης ότι πακέτα προορίζονται για το δίκτυο 10.75.104.96/27 (ιατρικό κέντρο Σοχού) και το δίκτυο 10.72.246.0/23 (Μονάδα εφήβων) να προωθούνται στο δίκτυο σύζευξης μέσω της πύλης 10.10.0.1.

ΚΕΦΑΛΑΙΟ 12

Βασικά αρχεία ρυθμίσεων του OpenNMS και πως προσαρμόστηκαν για το περιβάλλον το Νοσοκομείου Αχέπα.

12.1. - Το αρχείο discovery-configuration.xml

Παρακάτω βλέπουμε τις ακριβείς ρυθμίσεις που έγιναν για το περιβάλλον. Οι παράμετροι του αρχείου επεξηγήθηκαν ποιο αναλυτικά στο κεφάλαιο 4.

```
<?xml version="1.0" encoding="UTF-8" ?>
<discovery-configuration xmlns="http://xmlns.opennms.org/xsd/config/discovery" threads="2"
packets-per-second="1" initial-sleep-time="30000" restart-sleep-time="86400000" retries="2"
timeout="2000">
<specific retries="2" timeout="2000">10.10.8.10</specific>
<specific retries="2" timeout="2000">10.10.4.12</specific>
<specific retries="2" timeout="2000">10.10.1.14</specific>
<specific retries="2" timeout="2000">10.10.3.27</specific>
<specific retries="2" timeout="2000">10.10.13.1</specific>
<specific retries="2" timeout="2000">10.10.2.22</specific>
<specific retries="2" timeout="2000">10.10.8.80</specific>
<specific retries="2" timeout="2000">10.10.8.71</specific>
<specific retries="2" timeout="2000">10.10.8.123</specific>
<specific retries="5" timeout="2000">10.10.2.95</specific>
<specific retries="2" timeout="2000">10.75.104.98</specific>
<specific retries="2" timeout="2000">10.72.246.1</specific>
<specific retries="2" timeout="2000">10.75.104.97</specific>

<include-range retries="1" timeout="2000">
<begin xmlns="">10.10.12.1</begin>
<end xmlns="">10.10.12.254</end>
</include-range>

<include-range retries="1" timeout="2000">
<begin xmlns="">10.10.0.0</begin>
<end xmlns="">10.10.0.254</end>
</include-range>
</discovery-configuration>
```

Όπως φαίνεται παραπάνω έχουμε ορίσει κάποιες συγκεκριμένες IP διευθύνσεις μέσα στην ετικέτα <specific> και μετά έχουμε ορίσει δύο εύρη διευθύνσεων το 10.10.12.1- 254 όπου ανήκουν όλοι οι κατανεμητές και το 10.10.0.0 – 254 όπου ανήκουν όλοι σχεδόν οι εξυπηρετητές με εξαίρεση κάποιους οι οποίοι δηλώθηκαν με την ετικέτα <specific>.

12.2. - Το αρχείο capsd-configuration.xml

Το αρχείο ρυθμίσεων capsd-configuration.xml ελέγχει των δαίμονα capsd ο οποίος είναι υπεύθυνος για την εύρεση υπηρεσιών που παρέχονται από μία διεπαφή. Η λειτουργία του αναπτύχθηκε με λεπτομέρεια στην ενότητα 4.2.

```
<?xml version="1.0" ?>
<!-- 24 hours -->
<capsd-configuration rescan-frequency="86400000" initial-sleep-time="30000" max-suspect-thread-pool-size="6" max-rescan-thread-pool-size="3">
<protocol-plugin protocol="ICMP" class-name="org.opennms.netmgt.capsd.plugins.IcmpPlugin" scan="on">
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="StrafePing" class-name="org.opennms.netmgt.capsd.plugins.IcmpPlugin" scan="off">
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="SNMP" class-name="org.opennms.netmgt.capsd.plugins.SnmpPlugin" scan="on">
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HTTP" class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin" scan="on">
  <property key="port" value="80" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HTTP-8080" class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin" scan="on">
  <property key="port" value="8080" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HTTP-8000" class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin" scan="on">
  <property key="port" value="8000" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HTTPS" class-name="org.opennms.netmgt.capsd.plugins.HttpsPlugin" scan="on">
  <property key="port" value="443" />
  <property key="timeout" value="5000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HypericAgent" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin" scan="off">
  <property key="port" value="2144" />
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HypericHQ" class-name="org.opennms.netmgt.capsd.plugins.HttpPlugin" scan="off">
  <property key="port" value="7080" />
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="FTP" class-name="org.opennms.netmgt.capsd.plugins.FtpPlugin" scan="on">
  <property key="port" value="21" />
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
```

```
  </protocol-plugin>
<protocol-plugin protocol="Telnet" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin"
scan="on">
  <property key="banner" value="*" />
  <property key="port" value="23" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="DNS" class-name="org.opennms.netmgt.capsd.plugins.DnsPlugin"
scan="on">
  <property key="port" value="53" />
  <property key="timeout" value="5000" />
  <property key="retry" value="1" />
  <property key="lookup" value="localhost" />
  </protocol-plugin>
<protocol-plugin protocol="DHCP" class-name="org.opennms.netmgt.capsd.plugins.DhcpPlugin"
scan="on">
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="IMAP" class-name="org.opennms.netmgt.capsd.plugins.ImapPlugin"
scan="on">
  <property key="port" value="143" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="MSExchange" class-
name="org.opennms.netmgt.capsd.plugins.MSExchangePlugin" scan="on">
  <property key="pop3 port" value="110" />
  <property key="imap port" value="143" />
  <property key="mapi port" value="593" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="SMTP" class-name="org.opennms.netmgt.capsd.plugins.SmtpPlugin"
scan="on">
  <property key="port" value="25" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="POP3" class-name="org.opennms.netmgt.capsd.plugins.Pop3Plugin"
scan="on">
  <property key="port" value="110" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="SSH" class-name="org.opennms.netmgt.capsd.plugins.SshPlugin"
scan="on">
  <property key="banner" value="SSH" />
  <property key="port" value="22" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="MySQL" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin"
scan="off">
  <property key="banner" value="*" />
  <property key="port" value="3306" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="SQLServer" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin"
scan="off">
  <property key="banner" value="*" />
  <property key="port" value="1433" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
```

```
<protocol-plugin protocol="Oracle" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin"
scan="on">
  <property key="banner" value="*" />
  <property key="port" value="1521" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="Postgres" class-name="org.opennms.netmgt.capsd.plugins.TcpPlugin"
scan="on">
  <property key="banner" value="*" />
  <property key="port" value="5432" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="Router" class-name="org.opennms.netmgt.capsd.plugins.SnmpPlugin"
scan="on">
  <property key="vbname" value=".1.3.6.1.2.1.4.1.0" />
  <property key="vbvalue" value="1" />
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="HP Insight Manager" class-
name="org.opennms.netmgt.capsd.plugins.TcpPlugin" scan="on">
  <property key="banner" value="*" />
  <property key="port" value="2381" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="Dell-OpenManage" class-
name="org.opennms.netmgt.capsd.plugins.TcpPlugin" scan="on">
  <property key="banner" value="*" />
  <property key="port" value="1311" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="NSClient" class-
name="org.opennms.netmgt.capsd.plugins.NsclientPlugin" scan="off">
  <property key="banner" value="*" />
  <property key="port" value="1248" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="NSClientpp" class-
name="org.opennms.netmgt.capsd.plugins.NsclientPlugin" scan="off">
  <property key="banner" value="*" />
  <property key="port" value="12489" />
  <property key="timeout" value="3000" />
  <property key="retry" value="1" />
  </protocol-plugin>
<protocol-plugin protocol="NRPE" class-name="org.opennms.netmgt.capsd.plugins.NrpePlugin"
scan="off">
  <property key="banner" value="*" />
  <property key="port" value="5666" />
  <property key="timeout" value="3000" />
  <property key="retry" value="2" />
  <property key="command" value="_NRPE_CHECK" />
  </protocol-plugin>
<protocol-plugin protocol="NRPE-NoSSL" class-
name="org.opennms.netmgt.capsd.plugins.NrpePlugin" scan="off">
  <property key="banner" value="*" />
  <property key="port" value="5666" />
  <property key="timeout" value="3000" />
  <property key="retry" value="2" />
  <property key="usessl" value="false" />
  <property key="command" value="_NRPE_CHECK" />
  </protocol-plugin>
<protocol-plugin protocol="Windows-Task-Scheduler" class-
name="org.opennms.netmgt.capsd.plugins.Win32ServicePlugin" scan="on">
```

```
  <property key="timeout" value="2000" />
  <property key="retry" value="1" />
  <property key="service-name" value="Task Scheduler" />
  </protocol-plugin>
  </capsd-configuration>
```

Στο αρχείο αυτό που πρέπει να σημειωθεί είναι ότι απενεργοποιήθηκε η αναζήτηση κάποιων υπηρεσιών οι οποίες γνωρίζουμε εξ αρχής ότι δεν υπάρχουν στο δίκτυο και έτσι η προσπάθεια ανεύρεσης τους θα φόρτιζε άδικα το δίκτυο και το NMS.

12.3. - Το αρχείο snmp-config.xml

Στο αρχείο αυτό ορίζουμε ποια έκδοση του SNMP υλοποιεί ένας agent. Το OpenNMS είναι ρυθμισμένο έτσι ώστε αρχικά να δοκιμάζει την έκδοση 2c και αν αποτύχει τότε να δοκιμάσει την 1. Εδώ δηλώνοντας κατευθείαν ποια έκδοση υλοποιεί ο κάθε agent γλιτώνουμε κάποιους επιπλέον ελέγχους που θα έκανε το OpenNMS. Επίσης αυτό το αρχείο είναι χρήσιμο όταν θέλουμε για κάποιους agent να δηλώσουμε διαφορετικά community names η να ορίσουμε usernames και passwords για την έκδοση 3 του SNMP.

```
<?xml version="1.0" encoding="UTF-8" ?>
  <snmp-config xmlns="http://xmlns.opennms.org/xsd/config/snmp" retry="1" timeout="1800"
read-community="public" version="v2c">

<definition read-community="public" port="161" version="v1">
  <specific xmlns="">10.10.0.2</specific>
  <specific xmlns="">10.10.0.9</specific>
  <specific xmlns="">10.10.0.18</specific>
  <specific xmlns="">10.10.0.24</specific>
  <specific xmlns="">10.10.0.36</specific>
  <specific xmlns="">10.10.1.14</specific>
  <specific xmlns="">10.10.2.2</specific>
  <specific xmlns="">10.10.2.22</specific>
  <specific xmlns="">10.10.3.27</specific>
  <specific xmlns="">10.10.4.12</specific>
  <specific xmlns="">10.10.8.10</specific>
  <specific xmlns="">10.10.8.71</specific>
  <specific xmlns="">10.10.8.123</specific>
  <specific xmlns="">10.10.12.24</specific>
  <specific xmlns="">10.10.12.3</specific>
  <specific xmlns="">10.10.12.4</specific>
  <specific xmlns="">10.10.12.110</specific>
  <specific xmlns="">10.10.12.111</specific>
  <specific xmlns="">10.10.12.112</specific>
  <specific xmlns="">10.10.12.116</specific>
  <specific xmlns="">10.10.12.123</specific>
  <specific xmlns="">10.10.12.133</specific>
</definition>
</snmp-config>
```

12.4. - Το αρχείο poller-configuration.xml

Στο αρχείο ορίζουμε τις υπηρεσίες στις οποίες θέλουμε να γίνεται ενεργός έλεγχος (polling) και κάποιες παραμέτρους γύρω από αυτό όπως κάθε πόση ώρα να γίνετε ο έλεγχος, πόσες προσπάθειες να γίνονται, αν θα καταγράφεται ο χρόνος απόκρισης μιας υπηρεσίας και άλλες παράμετροι ανάλογα με την υπηρεσία. Στο αρχείο αυτό αφαιρέθηκαν οι υπηρεσίες που δεν θέλαμε να γίνετε ενεργός έλεγχος αλλάζοντας την παράμετρο status=”off”. Βέβαια από κάτω στο παράδειγμα δεν υπάρχουν καν οι υπηρεσίες που είναι off λόγω του μεγάλου όγκου του αρχείου.

```
<?xml version="1.0" ?>
  <?castor class-name="org.opennms.netmgt.poller.PollerConfiguration"?>
  <poller-configuration threads="30" serviceUnresponsiveEnabled="false">
  <node-outage status="on">
  <critical-service name="ICMP" />
  </node-outage>
  <package name="example1">
  <filter>IPADDR != '0.0.0.0'</filter>
  <include-range begin="1.1.1.1" end="254.254.254.254" />
  <rrd step="300">
  <rra>RRA:AVERAGE:0.5:1:2016</rra>
  <rra>RRA:AVERAGE:0.5:12:1488</rra>
  <rra>RRA:AVERAGE:0.5:288:366</rra>
  <rra>RRA:MAX:0.5:288:366</rra>
  <rra>RRA:MIN:0.5:288:366</rra>
  </rrd>
 <service name="ICMP" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="2" />
  <parameter key="timeout" value="3000" />
  <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
  <parameter key="rrd-base-name" value="icmp" />
  <parameter key="ds-name" value="icmp" />
  </service>
 <service name="DNS" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="2" />
  <parameter key="timeout" value="5000" />
  <parameter key="port" value="53" />
  <parameter key="lookup" value="localhost" />
  <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
  <parameter key="rrd-base-name" value="dns" />
  <parameter key="ds-name" value="dns" />
  </service>
 <service name="SMTP" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="1" />
  <parameter key="timeout" value="3000" />
  <parameter key="port" value="25" />
  <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
  <parameter key="rrd-base-name" value="smtp" />
  <parameter key="ds-name" value="smtp" />
  </service>
 <service name="FTP" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="1" />
  <parameter key="timeout" value="3000" />
  <parameter key="port" value="21" />
  <parameter key="userid" value="" />
  <parameter key="password" value="" />
  </service>
 <service name="SNMP" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="2" />
  <parameter key="timeout" value="3000" />
  <parameter key="port" value="161" />
```

```
    <parameter key="oid" value=".1.3.6.1.2.1.1.2.0" />
    </service>
   <service name="HTTP" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="timeout" value="3000" />
    <parameter key="port" value="80" />
    <parameter key="url" value="/" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="http" />
    <parameter key="ds-name" value="http" />
    </service>
    <service name="HTTP-8080" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="timeout" value="3000" />
    <parameter key="port" value="8080" />
    <parameter key="url" value="/" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="http-8080" />
    <parameter key="ds-name" value="http-8080" />
    </service>
   <service name="HTTP-8000" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="timeout" value="3000" />
    <parameter key="port" value="8000" />
    <parameter key="url" value="/" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="http-8000" />
    <parameter key="ds-name" value="http-8000" />
    </service>
  <service name="HTTPS" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="timeout" value="5000" />
    <parameter key="port" value="443" />
    <parameter key="url" value="/" />
    </service>
  <service name="Oracle" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="timeout" value="3000" />
    <parameter key="port" value="1521" />
    <parameter key="banner" value="*" />
    </service>
   <service name="Postgres" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="banner" value="*" />
    <parameter key="port" value="5432" />
    <parameter key="timeout" value="3000" />
    </service>
   <service name="Telnet" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="banner" value="*" />
    <parameter key="port" value="23" />
    <parameter key="timeout" value="3000" />
    <parameter key="rrd-repository" value="/opt/opennms/share/rrd/response" />
    <parameter key="ds-name" value="telnet" />
    </service>
   <service name="SSH" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="banner" value="SSH" />
    <parameter key="port" value="22" />
    <parameter key="timeout" value="3000" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="ssh" />
    <parameter key="ds-name" value="ssh" />
    </service>
   <service name="DHCP" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="2" />
    <parameter key="timeout" value="3000" />
```

```
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="dhcp" />
    <parameter key="ds-name" value="dhcp" />
    </service>
   <service name="IMAP" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="port" value="143" />
    <parameter key="timeout" value="3000" />
    </service>
   <service name="POP3" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="1" />
    <parameter key="port" value="110" />
    <parameter key="timeout" value="3000" />
    <parameter key="rrd-repository" value="/var/lib/opennms/rrd/response" />
    <parameter key="rrd-base-name" value="pop3" />
    <parameter key="ds-name" value="pop3" />
    </service>
   <service name="Windows-Task-Scheduler" interval="300000" user-defined="false" status="on">
    <parameter key="retry" value="2" />
    <parameter key="timeout" value="3000" />
    <parameter key="port" value="161" />
    <parameter key="service-name" value="Task Scheduler" />
    </service>
    <downtime interval="30000" begin="0" end="300000" />
  <!-- 30s, 0, 5m  -->
    <downtime interval="300000" begin="300000" end="43200000" />
  <!-- 5m, 5m, 12h  -->
    <downtime interval="600000" begin="43200000" end="432000000" />
  <!-- 10m, 12h, 5d -->
  <downtime  begin="432000000" delete="true" />
  <!-- anything after 5 days delete -->
    </package>
    <monitor service="ICMP" class-name="org.opennms.netmgt.poller.monitors.IcmpMonitor" />
    <monitor service="StrafePing" class-
  name="org.opennms.netmgt.poller.monitors.StrafePingMonitor" />
    <monitor service="HTTP" class-name="org.opennms.netmgt.poller.monitors.HttpMonitor" />
    <monitor service="HTTP-8080" class-name="org.opennms.netmgt.poller.monitors.HttpMonitor" />
    <monitor service="HTTP-8000" class-name="org.opennms.netmgt.poller.monitors.HttpMonitor" />
    <monitor service="HTTPS" class-name="org.opennms.netmgt.poller.monitors.HttpsMonitor" />
    <monitor service="SMTP" class-name="org.opennms.netmgt.poller.monitors.SmtpMonitor" />
    <monitor service="DHCP" class-name="org.opennms.netmgt.poller.monitors.DhcpMonitor" />
    <monitor service="DNS" class-name="org.opennms.netmgt.poller.monitors.DnsMonitor" />
    <monitor service="FTP" class-name="org.opennms.netmgt.poller.monitors.FtpMonitor" />
    <monitor service="SNMP" class-name="org.opennms.netmgt.poller.monitors.SnmpMonitor" />
    <monitor service="Oracle" class-name="org.opennms.netmgt.poller.monitors.TcpMonitor" />
<monitor service="Postgres" class-name="org.opennms.netmgt.poller.monitors.TcpMonitor" />
<monitor service="Telnet" class-name="org.opennms.netmgt.poller.monitors.TelnetMonitor" />
<monitor service="SSH" class-name="org.opennms.netmgt.poller.monitors.SshMonitor" />
<monitor service="IMAP" class-name="org.opennms.netmgt.poller.monitors.ImapMonitor" />
<monitor service="POP3" class-name="org.opennms.netmgt.poller.monitors.Pop3Monitor" />
<monitor service="NSClient" class-name="org.opennms.netmgt.poller.monitors.NsclientMonitor" />
<monitor service="Windows-Task-Scheduler" class-
  name="org.opennms.netmgt.poller.monitors.Win32ServiceMonitor" />  </poller-configuration>
```

12.5. - Το αρχείο javamail-configuration.properties

Στο αρχείο αυτό ορίσαμε κάποιες παραμέτρους για την αποστολή μηνυμάτων μέσω ηλεκτρονικού ταχυδρομείου ώστε να στέλνονται οι διάφορες ειδοποιήσεις για γεγονότα που συμβαίνουν στα στοιχεία του δικτύου που διαχείριζόμαστε. Ακολουθεί το αρχείο μαζί με σχολιασμό για το τι κάνει η κάθε παράμετρος.

```
################################################################################
# This file is the configuration for the the JavaMailer class.  It is used to
# specify the details of the JavaMailer system properties
# This property defines system sender account.
# The default setting is root@[127.0.0.1]
org.opennms.core.utils.fromAddress=opennms@auth.gr
```

Η διεύθυνση του αποστολέα του μηνύματος

```
# These properties define the SMTP Host.
org.opennms.core.utils.mailHost=155.207.1.44
```

Η IP διεύθυνση του εξυπηρετητή ηλεκτρονικού ταχυδρομείου που χρησιμοποιούμε για την αποστολή των μηνυμάτων.

```
#org.opennms.core.utils.mailer=smtpsend
#org.opennms.core.utils.transport=smtp
#org.opennms.core.utils.debug=true
#org.opennms.core.utils.smtpport=25
#org.opennms.core.utils.smtpssl.enable=false
#org.opennms.core.utils.quitwait=true
```

Τα παραπάνω στοιχεία αποτελούν παραμέτρους για τον τρόπο αποστολής. Δεν χρειάστηκε να αφαιρεθούν τα σχόλια μιας και αποτελούν τις τιμές εξορισμού για την αποστολή.

```
# This property controls the use of the JMTA
# if it is true, mailHost will be ignored
org.opennms.core.utils.useJMTA=false
```

Εδώ ορίζουμε εάν θα χρησιμοποιήσουμε ένα τοπικό MTA (Mail Transfer Agent) γραμμένο σε Java ή εάν θα χρησιμοποιήσουμε τον MTA ενός άλλου εξυπηρετητή. Επιλέξαμε να χρησιμοποιήσουμε τον MTA του εξυπηρετητή που δηλώσαμε παραπάνω.

```
# These properties define the Mail authentication.
org.opennms.core.utils.authenticate=true
org.opennms.core.utils.authenticateUser="lmpozios"
org.opennms.core.utils.authenticatePassword="*****"
org.opennms.core.utils.starttls.enable=false
```

Σε αυτό το σημείο ορίσαμε εάν ο εξυπηρετητής ηλεκτρονικού ταχυδρομείου θέλει αυθεντικοποίηση και τα στοιχεία του χρήστη που θα χρειαστούν στην αυθεντικοποίηση.

```
# These properties configure message content
#org.opennms.core.utils.messageContentType=text/plain
#org.opennms.core.utils.charset=us-ascii
```

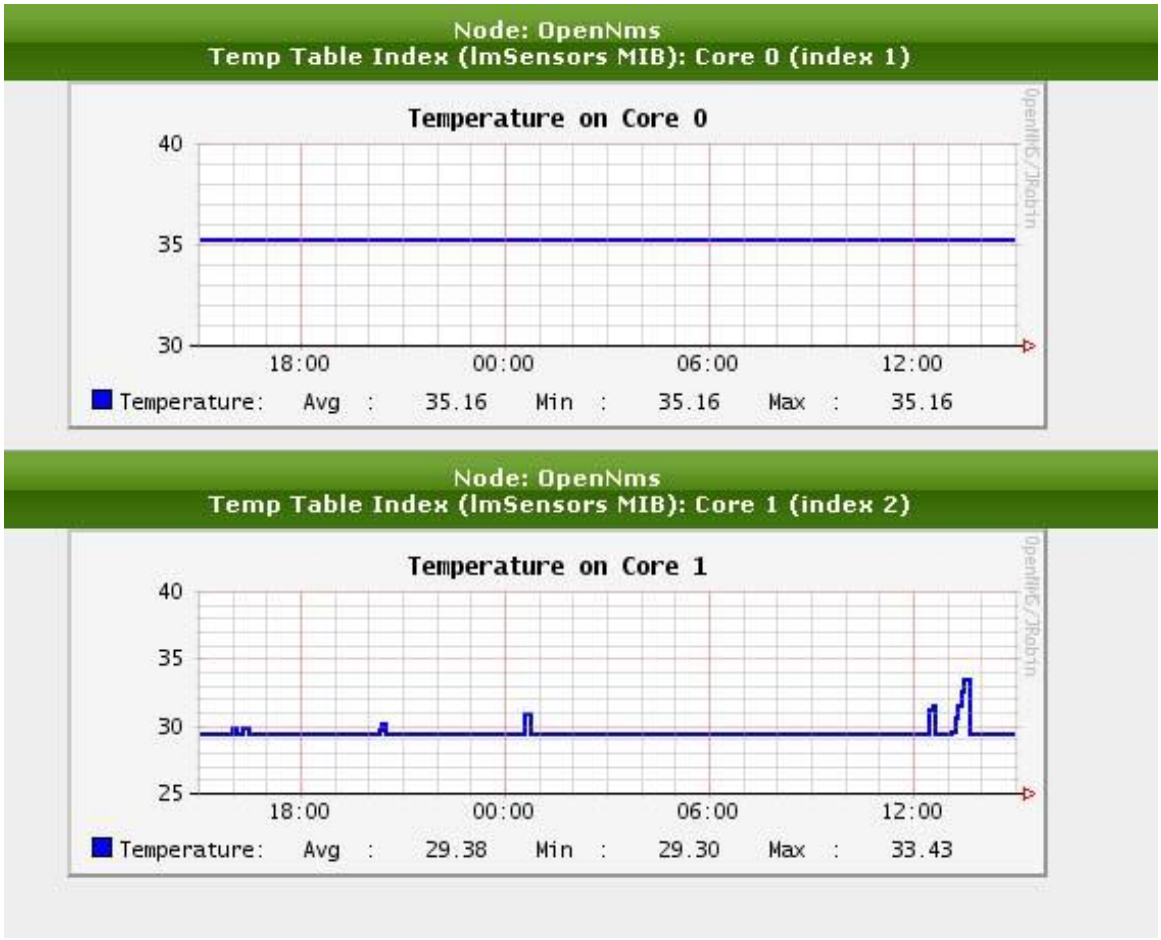
12.6. - Το αρχείο snmp-graph.properties

Στο αρχείο αυτό καθορίζεται ο τρόπος με τον οποίο θα παρουσιάζονται τα διάφορα διαγράμματα στο WebInterface του OpenNMS. Το αρχείο περιέχει τις παραμέτρους για όλα τα διαγράμματα. Στην ουσία το OpenNMS με βάση αυτό το αρχείο οπτικοποιεί την πληροφορία που συγκεντρώνει στα αρχεία jrb (Java Round Robin) τα οποία ανανεώνει ο δαίμονας collectd. Επειδή το αρχείο είναι πολύ μεγάλο δεν θα παρουσιαστεί ολόκληρο αλλά θα δοθεί ένα μικρό παράδειγμα για το πως προσθέσαμε κάποιες παραμέτρους για την δημιουργία διαγραμμάτος τα οποίο εμπεριέχει πληροφορία η οποία αφορά τις θερμοκρασίες που μπορεί να μας δώσει το πρόγραμμα lm-sensors στο λειτουργικό σύστημα GNU/Linux Debian.

Στην αρχή του αρχείου πρέπει να δηλώσουμε το όνομα του διαγράμματος έτσι εισάγουμε το όνομα lmsensors.temp. Η σειρά με την οποία δηλώνονται τα ονόματα θα πρέπει να είναι ίδια με τις παραμέτρους που απαρτίζονται δηλαδή εάν το όνομα είναι πέμπτο στη λίστα τον ονομάτων τότε πρέπει και οι παράμετροι του διαγράμματος να είναι και αυτές στην πέμπτη θέση σε σχέση με τα άλλα block παραμέτρων.

```
report.lmsensors.temp.name=lmSensors Temperature Sensor
report.lmsensors.temp.columns=lms-temp
report.lmsensors.temp.type=lmTempIndex
report.lmsensors.temp.propertiesValues=lms-tempdevice
report.lmsensors.temp.command=--title="Temperature on {lms-tempdevice}" \
 DEF:dtemp={rrd1}:lms-temp:AVERAGE \
 DEF:minDtemp={rrd1}:lms-temp:MIN \
 DEF:maxDtemp={rrd1}:lms-temp:MAX \
 CDEF:btemp=dtemp,1024,/ \
 CDEF:minBtemp=minDtemp,1024,/ \
 CDEF:maxBtemp=maxDtemp,1024,/ \
 LINE2:btemp#0000ff:"Temperature:" \
 GPRINT:btemp:AVERAGE:" Avg  \\: %8.2lf %s" \
 GPRINT:btemp:MIN:"Min  \\: %8.2lf %s" \
 GPRINT:btemp:MAX:"Max  \\: %8.2lf %s\\n" \
```

Οι ορισμοί των παραμέτρων ακολουθούν την σύνταξη του RRDTool. Οι παραπάνω γραμμές μας δημιουργούν το εξής διαγράμματα.



Σχήμα 12.1 – Διάγραμμα θερμοκρασίας επεξεργαστή

12.7. - Το αρχείο linkd-configuration.xml

Σε αυτό το αρχείο εμπεριέχονται οι ρυθμίσεις του δαίμονα Linkd ο οποίος είναι υπεύθυνος ώστε να ανακαλύπτει διασυνδέσεις μεταξύ διαφόρων συσκευών οι οποίες εμφανίζονται γραφικά στους χάρτες που μπορούμε να φτιάξουμε. Στο πρώτο μέρος του αρχείου δηλώνονται MIB αντικείμενα διαφόρων εταιριών οι οποίες κατασκευάζουν δικτυακές συσκευές ώστε να λαμβάνουμε την πληροφορία πόσα και ποια Vlans έχει ο κάθε κόμβος. Στο τέλος του αρχείου δημιουργούμε ένα package το οποίο περιέχει όλες τις διευθύνσεις που θα ελέγχουνε και με ποιο τρόπο, ώστε να βρεθεί η συσκευή με ποιες άλλες συνδέεται φυσικά. Ο Linkd ανακαλύπτει φυσικές διασυνδέσεις είτε μέσω του πίνακα προώθησης στους

κατανεμητές είτε μέσω του πίνακα δρομολόγησης στους δρομολογητές. Τους πίνακες αυτούς τους συλλέγει με την βοήθεια του SNMP.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<linkd-configuration threads="5" initial_sleep_time="3600000" snmp_poll_interval="18000000"
discovery_link_interval="300000">
<vlans>
<vendor vendor_name="3Com" sysoidRootMask=".1.3.6.1.4.1.43" class-
name="org.opennms.netmgt.linkd.snmp.ThreeComVlanTable">
  <specific>1.9.13.3.1</specific>
  <specific>10.27.4.1.2.2</specific>
  <specific>10.27.4.1.2.4</specific>
  <specific>10.27.4.1.2.11</specific>
  <specific>1.16.4.3.5</specific>
  <specific>1.16.4.3.6</specific>
  </vendor>
<vendor vendor_name="3Com3870" sysoidRootMask=".1.3.6.1.4.1.43.1" class-
name="org.opennms.netmgt.linkd.snmp.Dot1qStaticVlanTable">
  <specific>8.43</specific>
  <specific>8.61</specific>
  </vendor>
<vendor vendor_name="Nortel" sysoidRootMask=".1.3.6.1.4.1.45.3" class-
name="org.opennms.netmgt.linkd.snmp.RapidCityVlanTable">
  <specific>61.1</specific>
  <specific>35.1</specific>
  <specific>53.1</specific>
  </vendor>
<vendor vendor_name="Intel" sysoidRootMask=".1.3.6.1.4.1.343.5" class-
name="org.opennms.netmgt.linkd.snmp.IntelVlanTable">
  <specific>1.5</specific>
  </vendor>
<vendor vendor_name="HP Networks" sysoidRootMask=".1.3.6.1.4.1.11.2.3.7" class-
name="org.opennms.netmgt.linkd.snmp.Dot1qStaticVlanTable">
  <specific>11.1</specific>
  <specific>11.3</specific>
  <specific>11.13</specific>
  <specific>11.14</specific>
  <specific>11.18</specific>
  <specific>11.19</specific>
  <specific>11.20</specific>
  <specific>11.23</specific>
  <specific>11.27</specific>
  <specific>11.29</specific>
  <specific>11.32</specific>
  <specific>11.45</specific>
  <specific>11.50</specific>
  <specific>11.63</specific>
  <specific>11.66</specific>
  <specific>11.81</specific>
  <include-range begin="11.6" end="11.11" />
  </vendor>
<vendor vendor_name="cisco" sysoidRootMask=".1.3.6.1.4.1.9" class-
name="org.opennms.netmgt.linkd.snmp.CiscoVlanTable">
  <specific>1.217</specific>
  <specific>1.218</specific>
  <specific>1.222</specific>
  <specific>1.221</specific>
  <specific>1.246</specific>
  <specific>1.247</specific>
  <specific>1.248</specific>
  <specific>1.278</specific>
  <specific>1.282</specific>
  <specific>1.283</specific>
  <specific>1.300</specific>
```

```
  <specific>1.370</specific>
  <specific>1.324</specific>
  <specific>1.325</specific>
  <specific>1.366</specific>
  <specific>1.367</specific>
  <specific>1.400</specific>
  <specific>1.427</specific>
  <specific>1.428</specific>
  <specific>1.429</specific>
  <specific>1.431</specific>
  <specific>1.448</specific>
  <specific>1.502</specific>
  <specific>1.503</specific>
  <specific>1.516</specific>
  <specific>1.542</specific>
  <specific>1.561</specific>
  <specific>1.563</specific>
  <specific>1.564</specific>
  <specific>1.565</specific>
  <specific>1.578</specific>
  <specific>1.616</specific>
  <specific>1.617</specific>
  <specific>1.633</specific>
  <specific>1.634</specific>
  <specific>1.694</specific>
  <specific>1.716</specific>
  <specific>1.748</specific>
  <specific>5.7</specific>
  <specific>5.17</specific>
  <specific>5.18</specific>
  <specific>5.20</specific>
  <specific>5.28</specific>
  <specific>5.34</specific>
  <specific>5.36</specific>
  <specific>5.41</specific>
  <specific>5.42</specific>
  <specific>5.46</specific>
  <specific>5.49</specific>
  <specific>5.51</specific>
  <specific>5.59</specific>
  </vendor>
<vendor vendor_name="Extreme Networks" sysoidRootMask=".1.3.6.1.4.1.1916" class-
name="org.opennms.netmgt.linkd.snmp.ExtremeNetworkVlanTable">
  <specific>2.11</specific>
  <specific>2.14</specific>
  <specific>2.28</specific>
  <specific>2.63</specific>
  </vendor>
  </vlans>

<package name="example1">
  <filter>IPADDR != '0.0.0.0'</filter>
  <include-range begin="1.1.1.1" end="254.254.254.254" />
  </package>
<!-- Use more packages with huge network -->
<!-- here is a configuration that is valid on ethernet LANs  -->

<package name="LAN" use-ip-route-discovery="false">
  <filter>IPADDR != '0.0.0.0'</filter>
  <include-range begin="10.10.0.1" end="10.10.15.254" />
  </package>
</linkd-configuration>
```

ΚΕΦΑΛΑΙΟ 13

Ρυθμίσεις των agents και πρόσθετα εργαλεία

Παρακάτω θα παρουσιαστούνε κάποια μέρη από τα αρχεία ρυθμίσεων από SNMP agents κάποιον εξυπηρετητών και κατανεμητών που υπάρχουν στο υπολογιστικό κέντρο του νοσοκομείου ΑΧΕΠΑ και θα γίνουν κάποιες επεξηγήσεις.

13.1. - AIX SNMP agent

Ακολουθεί ένα μέρος του αρχείου ρυθμίσεων του δαίμονα snmpd στο λειτουργικό σύστημα AIX με σχόλια για το τι κάνει το κάθε μέρος του.

logging file=/usr/tmp/snmpd.log enabled
logging size=1000 level=0
#Εδώ ορίζουμε το που θα τοποθετηθεί το αρχείο καταγραφής συμβάντων #snmp.log εάν θα είναι ενεργό και τι μέγεθος θα έχει

community public 10.10.12.254 255.255.255.255 readOnly All
community private 127.0.0.1 255.255.255.255 readWrite
community system 127.0.0.1 255.255.255.255 readWrite 1.17.2
#Εδώ βλέπουμε το πώς ορίζουμε την συμβολοσειρά community και #μπορούμε να δηλώσουμε και μια διεύθυνση IP καθώς και ένα υποδίκτυο #από τα οποία μπορεί να δέχεται αιτήσεις ο agent . Στην περίπτωση μας #έχουμε ορίσει το public σαν κωδικό που να επιτρέπει το διάβασμα από #τον agent και καθορίσαμε την IP του NMS που είναι η 10.10.12.254

#view 1.17.2 system enterprises view
view All iso.3 system enterprises view internet host

#Εδώ βλέπουμε το πώς ορίζουμε views. Τα views είναι ένα σύστημα στο #οποίο μπορούμε να δηλώσουμε υποδέντρα του συνολικού δέντρου MIB #του agent και να τα συσχετίσουμε με κάποια συμβολοσειρά community. #Στην συγκεκριμένη περίπτωση έχουμε ένα view που λέγεται ALL και #περιλαμβάνει τα υποδέντρα “iso.3 system enterprises view internet host”

#και είναι συσχετισμένο με το community public όπως φαίνεται #παραπάνω.

trap public 127.0.0.1 1.2.3 fe # loopback
trap public 10.10.12.254 1.17.2 fe # loopback

#Εδώ ορίζουμε το που θέλουμε να στέλνονται τα traps και με τι #συμβολοσειρά community. Σαν συμβολοσειρά ορίσαμε το public και σαν #IP διευθύνσεις ορίσαμε το ίδιο τον εξυπηρετητή και το NMS #(10.10.12.254).

#snmpd maxpacket=1024 querytimeout=120 smuxtimeout=60
smux 1.3.6.1.4.1.2.3.1.2.1.2 gated_password # gated
smux 1.3.6.1.4.1.2.3.1.2.2.1.1.2 dpid_password #dpid

Η τελευταίες ρυθμίσεις καθορίζουν την συμπεριφορά του πρωτοκόλλου smux. Δεν αλλάχθηκαν και αφήσαμε τις προεπιλεγμένες.

13.2. - HPUX SNMP agent

Ακολουθεί ένα μέρος του αρχείου ρυθμίσεων (snmpd.conf) του δαίμονα snmpd στο λειτουργικό σύστημα HPUX με σχόλια για το τι κάνει το κάθε μέρος του.

get-community-name: public
#Το αλφαριθμητικό public καθορίζει την πρόσβαση για διάβασμα #παραμέτρων του agent

set-community-name: private
#Το αλφαριθμητικό private καθορίζει την πρόσβαση για διάβασμα, καθώς #και για τροποποίηση παραμέτρων του agent.

trap-dest: 10.10.12.254 # enter trap destination
#Εδώ καθορίσαμε τον προορισμό των traps ο οποίος είναι το NMS.

Ο agent υποστηρίζει επίσης και το σύστημα των views όπως και στο AIX αλλά στην συγκεκριμένη υλοποίηση δεν αλλάξαμε τίποτα. Εάν δεν δηλωθεί κάποιο view τότε όλο το δέντρο του MIB θεωρείται προσβάσιμο.

13.3. - NET-SNMP agent

Στο firewall και στο NMS που τρέχουν το λειτουργικό σύστημα Debian 5.0 εγκαταστάθηκε ο agent NET-SNMP με την εντολή apt-get install snmp snmpd.
Το αρχείο ρυθμίσεών του βρίσκετε στον κατάλογο /etc/snmp/ και ονομάζετε snmpd.conf. Ακολουθεί ένα μέρος του και επεξηγούνται κάποιες βασικές παράμετροί του.

```
####
# First, map the community name (COMMUNITY) into a security name
# (local and mynetwork, depending on where the request is coming
# from):
#       sec.name  source          community
com2sec local  localhost         public
com2sec localnet  10.10.12.254   public
com2sec readwrite default         private
```

#Εδώ ορίζουμε security names τα οποία τα συσχετίζουμε με #συμβολοσειρές community και με υποδίκτυα η συγκεκριμένους σταθμούς #που θα λαμβάνονται οι αιτήσεις.

```
####
# sec.model  sec.name
group MyROGroup v1         localnet
group MyROGroup v2c        localnet
group MyROGroup usm        localnet
group MyRWGroup v1         local
group MyRWGroup v2c        local
group MyRWGroup usm        local
```

#Συσχετίζουμε τα security names με ομάδες (groups) για διαφορετικές #εκδόσεις του SNMP

####
Third, create a view for us to let the groups have rights to:
incl/excl subtree mask
view all included .1 80
#Ορίζουμε views τα οποία παρακάτω θα τα συσχετίσουμε με τα groups #που φτιάξαμε παραπάνω. Το view all περιλαμβάνει όλο το υποδέντρο #μετά τον κόμβο .1 δηλαδή όλα τα αντικείμενα του MIB δέντρου.

####
Finally, grant the 2 groups access to the 1 view with different
write permissions:
context sec.model sec.level match read write notif
access MyROGroup "" any noauth exact all none none
access MyRWGroup "" any noauth exact all all none
#Συσχετίσαμε τις δύο ομάδες με το view all μόνο που στην περίπτωση του #πρώτου group view είναι read-only ενώ στο δεύτερο group είναι read-#write(όσα αντικείμενα είναι writable). ####

trap2sink 10.10.12.254
trapcommunity public
Εδώ ορίσαμε τον προορισμό των traps, ο οποίος είναι η IP διεύθυνση του NMS καθώς και την συμβολοσειρά community που θα περιέχει το trap.

13.4. - Cisco SNMP agent

Το δίκτυο του νοσοκομείου Αχέπα περιλαμβάνει 62 κατανεμητές της εταιρίας Cisco. Οι 60 βρίσκονται μέσα στην κτηριακή υποδομή του νοσοκομείου και οι άλλοι δύο βρίσκονται σε απομακρυσμένες τοποθεσίες που είναι η μονάδα εφήβων στο κέντρο της Θεσσαλονίκης καθώς και το Κέντρου Υγείας Σοχού.
Εάν και όλοι οι κατανεμητές δεν είναι το ίδιο μοντέλο οι ρυθμίσεις που αφορούνε τον SNMP agent είναι παρόμοιες. Αυτό έχει να κάνει με το λειτουργικό σύστημα του κάθε κατανεμητή, καθώς και τι υπηρεσίες προσφέρει. Παρακάτω παρουσιάζεται ένα απόσπασμα από το αρχείο ρυθμίσεων ενός κατανεμητή 2960 με σχόλια για την χρησιμότητα της κάθε γραμμής.

logging 10.10.12.254

#Εδώ καθορίζουμε σε ποια διεύθυνση θα αποστέλλονται τα σύμβαντα της #συσκευής με την μορφή traps. Ορίσαμε το NMS σαν προορισμό. Έτσι στο #NMS καταγράφονται όλα τα γεγονότα όπως π.χ. η απενεργοποίηση μιας #πόρτας του κατανεμητή.

snmp-server community public RO

snmp-server community private RW

#Εδώ καθορίσαμε την συμβολοσειρά community. Το public έχει πρόσβαση #μόνο για ανάγνωση ενώ το private και για ανάγνωση και για εγγραφή.

snmp-server enable traps snmp authentication linkdown linkup coldstart warmstart

snmp-server enable traps transceiver all

snmp-server enable traps tty

snmp-server enable traps cluster

snmp-server enable traps entity

snmp-server enable traps cpu threshold

snmp-server enable traps vtp

snmp-server enable traps vlancreate

snmp-server enable traps vlandelete

snmp-server enable traps flash insertion removal

snmp-server enable traps port-security

snmp-server enable traps envmon fan shutdown supply temperature status

snmp-server enable traps power-ethernet group 1

snmp-server enable traps power-ethernet police

snmp-server enable traps config-copy

snmp-server enable traps config

snmp-server enable traps config-ctid

snmp-server enable traps rtr

snmp-server enable traps bridge newroot topologychange

snmp-server enable traps stpx inconsistency root-inconsistency loop-inconsistency

snmp-server enable traps syslog

snmp-server enable traps mac-notification change move threshold

snmp-server enable traps vlan-membership

snmp-server enable traps errdisable

#Όλες οι παραπάνω δηλώσεις αφορούν την ενεργοποίηση διαφόρων #traps. Π.χ η τελευταία δήλωση snmp-server enable traps errdisable #ενεργοποιεί τον μηχανισμό ώστε ο SNMP agent του κατανεμητή να #αποστέλλει ένα trap στο NMS όταν κάποια πόρτα του απενεργοποιηθεί #λόγω τον πολλών λαθών που συμβαίνουν.

snmp-server host 10.10.12.254 public

#Εδώ δηλώνουμε την διεύθυνση του NMS που θα αποστέλλονται τα traps #καθώς και το αλφαριθμητικό community που το trap θα περιέχει.

Από την στιγμή που δεν θέλουμε να μας αποστέλλονται κάποια traps μπορούμε να μην τα συμπεριλάβουμε στο αρχείο ρυθμίσεων. Για το ποια ακριβώς είναι τα traps που υποστηρίζει η συσκευή μπορούμε να ανατρέξουμε στο εγχειρίδιο της συσκευής καθώς και να γνωρίσουμε ποια έκδοση του IOS χρησιμοποιεί.

13.5. - Windows XP και Windows 2003 SNMP agent

Εγκατάσταση και ενεργοποίηση της υπηρεσίας SNMP

1. Στα Windows XP και Windows 2003, κάνουμε κλικ στο κουμπί start, στη συνέχεια πηγαίνουμε στο Control Panel και μετά στο Add or Remove Programs. Στο Add or Remove Programs, κάνουμε κλικ στο κουμπί Add / Remove Windows Components και ανοίγουμε το Windows Components Wizard.

2. Στα Components των Windows XP και 2003, κάνουμε κλικ στο Management and Monitoring Tools και στη συνέχεια κάνουμε κλικ στην επιλογή Details.

3. Επιλέγουμε και σημειώνουμε το Simple Network Management Protocol ή SNMP feature.

4. Κάνουμε κλικ στο OK. Επίσης, κάνουμε κλικ στο Next. Η SNMP υπηρεσία θα εγκατασταθεί στο σύστημα. Μπορεί να χρειαστεί η εισαγωγή του CD / DVD των windows στη μονάδα οπτικού δίσκου.

Δύο νέες υπηρεσίες θα δημιουργηθούν:

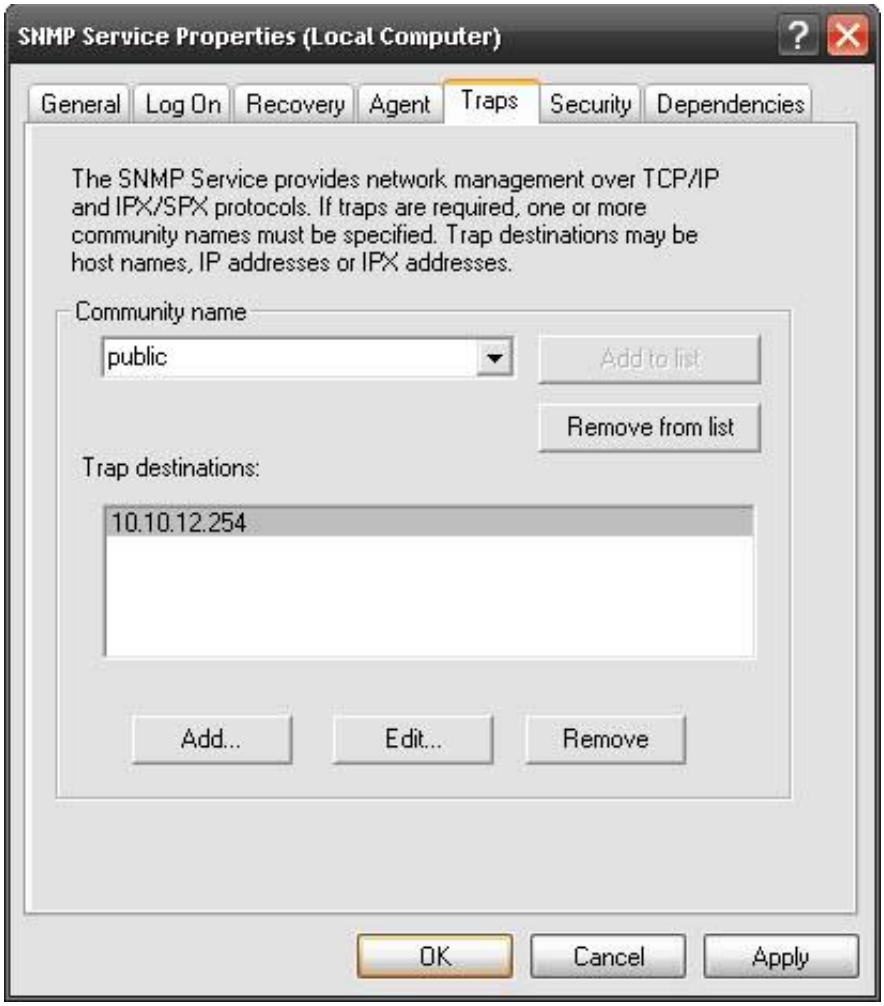
1.Η SNMP υπηρεσία η οποία είναι η κύρια μηχανή με agents που παρακολουθούν τη δραστηριότητα των συσκευών δικτύου και αναφέρουν τις πληροφορίες στο σύστημα διαχείρισης δικτύου.

2. Η SNMP υπηρεσία η οποία λαμβάνει traps που δημιουργούνται από τοπικούς ή απομακρυσμένους SNMP agents.

Τα Windows δεν αποδίδουν οποιαδήποτε συμβολοσειρά community για την υπηρεσία SNMP από προεπιλογή, και επίσης επιτρέπουν πρόσβαση μόνο από το localhost ή τοπικές συσκευές. Χρειάζεται να γίνουν περαιτέρω ρυθμίσεις για να προσθέσουμε την επιθυμητή συμβολοσειρά community, η οποία ενεργεί ως ο κωδικός πρόσβασης για τη χορήγηση απάντησης σε οποιοδήποτε αίτημα SNMP από το απομακρυσμένο σύστημα.

Ρυθμίσεις στην υπηρεσία SNMP

1. Κάνουμε κλικ στο κουμπί start, και στη συνέχεια πηγαίνουμε στον Control Panel.
2. Ανοίγουμε τα Administrator Tools.
3. Ανοίγουμε τα Services.
4. Εντοπίζουμε και κάνουμε δεξί κλικ πάνω υπηρεσία SNMP, στη συνέχεια, επιλέγουμε Properties.
5. Στην υπηρεσία SNMP στο παράθυρο Properties, κάνουμε κλικ στην καρτέλα traps.
6. Στο "Community name" κουτί κειμένου, προσθέτουμε την συμβολοσειρά public που θα χρησιμοποιούν τα traps.
7. Κάνουμε κλικ στο κουμπί Add to list.
8. Κάνουμε κλικ στο κουμπί Add και προσθέτουμε την IP διεύθυνση του NMS που στην υλοποίηση μας είναι 10.10.12.254.



Τρόπος ρύθμισης παραμέτρων ασφαλείας για την υπηρεσία SNMP για ένα community

1. Συνεχίζουμε από τα παραπάνω βήματα, κάνουμε κλικ στην καρτέλα Security.

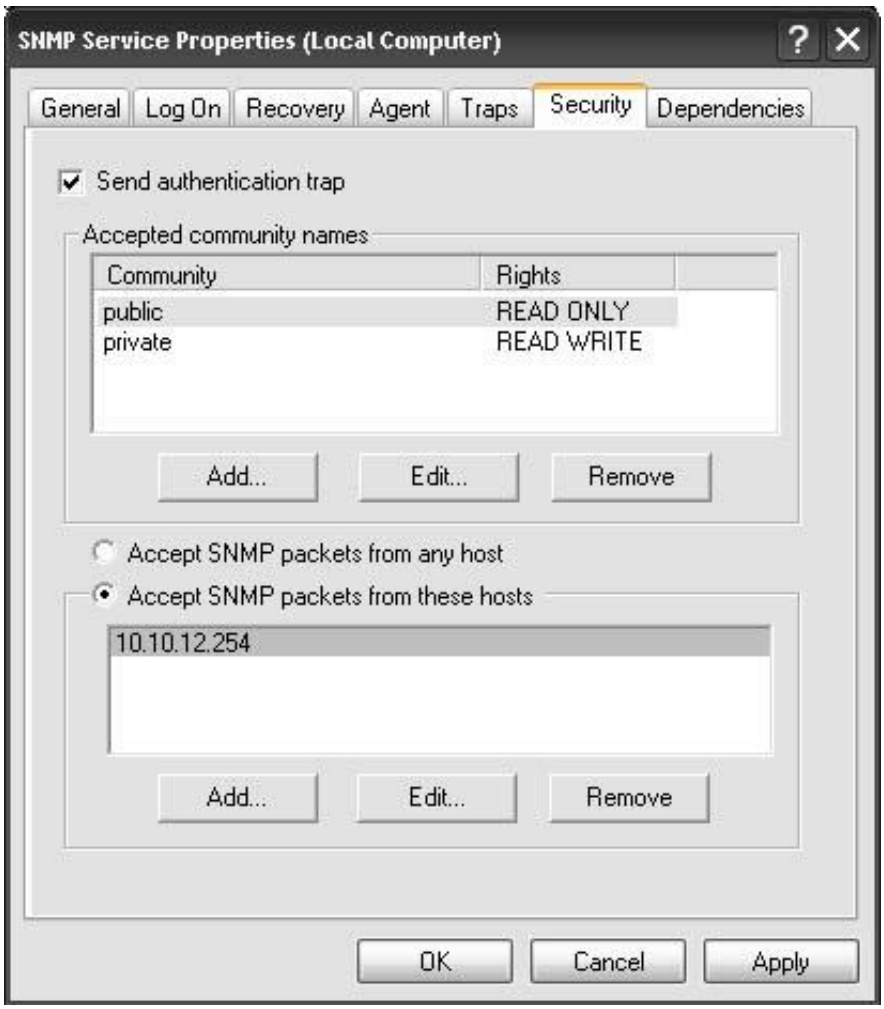
2. Στο "Accepted community names", κάνουμε κλικ στο κουμπί Add.

3. Στο Community rights επιλέξτε READ ONLY και στο Community Name εισάγουμε το public.

4. Κάνουμε ξανά κλικ στο Add επιλέγουμε στο Community rights το READ WRITE και στο Community Name εισάγουμε το private.

5. Στο "Accepted SNMP packets from these hosts", κάνουμε κλικ στο κουμπί Add, και προσθέτουμε τις IP διευθύνσεις που θέλουμε να επιτρέψουμε να κάνουν ερωτήματα στην υπηρεσία SNMP που στην συγκεκριμένη υλοποίηση είναι η διεύθυνση 10.10.12.254.

6. Εάν θέλουμε η υπηρεσία να απαντά σε οποιαδήποτε διεύθυνση επιλέγουμε το Accept SNMP packets from any host.



13.6. - Το script mac2port

Το mac2port αναπτύχθηκε συγκεκριμένα για το περιβάλλον το νοσοκομείου Αχέπα αν και είναι δυνατόν με κάποιες μικρές μετατροπές να εφαρμοστεί και σε άλλες υποδομές οι οποίες στηρίζονται πάνω σε κατανεμητές της Cisco. Ο σκοπός της ανάπτυξής του ήταν να καλύψει την ανάγκη να γνωρίζουμε που ακριβώς βρίσκεται ένας υπολογιστής στην κτηριακή υποδομή του νοσοκομείου μέσω της διεύθυνσης MAC. Χρειάστηκε πολλές φορές να βρεθεί η θέση ενός σταθμού από την MAC του και επειδή στο περιβάλλον του νοσοκομείου δεν υπήρχε η δυνατότητα καταγραφής όλων τον MAC διευθύνσεων το mac2port αποδείχθηκε ιδιαιτέρα χρήσιμο. Βασικά πολλές φορές χρειάστηκε να βρούμε την ακριβής τοποθεσία που εκπέμπονταν πακέτα από μία συγκεκριμένη IP διεύθυνση. Έτσι μέσω της IP διεύθυνσης και του εργαλείου Nmap βρίσκαμε την MAC διεύθυνση και μετά μέσω του εργαλείου mac2port βρίσκαμε σε ποιο κατανεμητή και σε ποια πόρτα του ήτανε συνδεμένος ο σταθμός.

Το εργαλείο mac2port μαζί με κάποια άλλα εργαλεία που αναπτύχθηκαν από τον συμφοιτητή μου Δημήτρη Διαμαντή τοποθετήθηκαν στον ίδιο εξυπηρετητή με το OpenNMS και από κοινού δημιουργήσαμε ένα WebInterface για ευκολία εκτέλεσης χωρίς να χρειάζεται η σύνδεση με κάποιο τερματικό στον εξυπηρετητή.
Το εργαλείο αυτό αναπτύχθηκε στην γλώσσα του τερματικού bash χρησιμοποιώντας το Net-snmp που αναφέρθηκε στο τρίτο κεφαλαίο το οποίο είναι μια υλοποίηση του SNMP για unix και βασισμένα στο unix λειτουργικά συστήματα όπως είναι το GNU/Linux Debian που χρησιμοποιήσαμε σαν λειτουργικό σύστημα του εξυπηρετητή που αποτέλεσε το NMS της υποδομής.
Τα εργαλεία αυτά συμπληρώνουν την λειτουργικότητα του NMS μιας και πρόκειται για εργαλεία δικτύου τα οποία φαίνονται χρήσιμα στην ασφάλεια και στην διαχείριση τον λογαριασμών στο δίκτυο.

Παρακάτω παρατίθεται ο κώδικας του προγράμματος με σχόλια για την κατανόηση της λειτουργίας του.

```
#!/bin/bash
```

```
#Copyright (C) 2010 Kostas Mitrogeorgos
#This is free software. You may redistribute copies of it under the terms of
#the GNU General Public License <http://www.gnu.org/licenses/gpl.html>
#There is NO WARRANTY, to the extent permitted by law.

#Link a MAC address to the port on which the address was learned.

echo ----------------Cisco_Switches----------------
echo trunking status 1=trunk 2=notrunk
printf "%-15s%-15s%-15s%s\n" Hostname Host_ip port trunkstatus
a=`echo $1 | sed "s/:/ /g"`
#Αποθήκευση της παραμέτρου (MAC Address) στην μεταβλητή a.

while read line
        do
#Εκκίνηση βρόχου ο οποίος εκτελείται για κάθε IP που περιέχεται στο
#αρχείο cisco_ip_list
      #finds the name of the hostdevice
      Hostname=`snmpget -v 2c -c public@1 ${ip[i]} .1.3.6.1.2.1.1.5.0 | cut -d " " -
f4`
#Εύρεση του hostname μέσω της εντολής snmpget

#Step 1, the MAC address is:
      tempo=`snmpwalk -v 2c -c public@1 ${ip[i]} .1.3.6.1.2.1.17.4.3.1.1 | grep -i
"$a" | cut -d " " -f1`
      tempo=`echo $tempo | sed "s/2.17.4.3.1.1./2.17.4.3.1.2./g"`
#Εύρεση εάν υπάρχει η διεύθυνση MAC στην προσωρινή μνήμη του
#κατανεμητή και αποθήκευση στην μεταβλητή tempo το OID του
#αντικειμένου που περιέχει την διεύθυνση MAC.

      if [ -n "$tempo" ] ; then
            #Step 2, the bridge port tells that the MAC address belongs to bridge
port number $tempo:
```

```
        tempo=`snmpwalk -v 2c -c public@1 ${ip[i]} .1.3.6.1.2.1.17.4.3.1.2 | grep
"$tempo" | cut -d " " -f4`
```
#Εάν υπάρχει η διεύθυνση MAC στη μνήμη του κατανεμητή τότε #βρίσκουμε το bridge port με το οποίο είναι συνδεδεμένη, δηλαδή από #όπου μαθεύτηκε αυτή η διεύθυνση MAC.

```
#Step 3, the bridge port number $tempo has ifIndex number $tempo(new):
        tempo=`snmpwalk -v 2c -c public@1 ${ip[i]} .1.3.6.1.2.1.17.1.4.1.2 | grep
"SNMPv2-SMI::mib-2.17.1.4.1.2.$tempo[[:space:]]" | cut -d " " -f4`
```
#Το bridge port είναι μια εσωτερική αρίθμηση του κατανεμητή που πολλές #φορές δεν είναι εύκολο να καταλάβουμε ποια φυσική του πόρτα είναι. #Χρησιμοποιώντας το bridge port όμως μπορούμε να βρούμε το ifindex το #οποίο αποτελεί την φυσική πόρτα του κατανεμητή

```
        #Step 4, check if the learning port is in TrunkMode
        trunkstatus=`snmpget -v 2c -c public@1 ${ip[i]}
1.3.6.1.4.1.9.9.46.1.6.1.1.14.$tempo | cut -d " " -f4`
```
#΄Ελεγχος ώστε να ξέρουμε εάν η πόρτα που βρέθηκε είναι σε κατάσταση #trunk ή όχι. Η χρησιμότητα αυτού είναι ότι συνήθως δεν υπάρχει σταθμός #συνδεμένος σε πόρτα η οποία είναι σε κατάσταση trunk.

```
 #Step 5,if the learning port isn't in TrunkMode show the port
        #if [ $trunkstatus -eq 2 ] ; then
        port=`snmpwalk -v 2c -c public@1 ${ip[i]} .1.3.6.1.2.1.31.1.1.1.1 | grep
"IF-MIB::ifName.$tempo[[:space:]]" | cut -d " " -f4`
        #echo port= $port
        #echo -e "\n ${ip[i]} \t $port \t $trunkstatus"
        printf "%-15s%-15s%-15s%s\n" $Hostname ${ip[i]} $port $trunkstatus
        #fi
```
#Εκτύπωση το όνομα του κατανεμητή, της IP διεύθυνσης του , της πόρτας #από την οποία έμαθε την MAC διεύθυνση και την κατάσταση της πόρτας #σε σχέση με την λειτουργία trunk.

```
    else
    #echo -e "\n ${ip[i]} \t\t Notfound \t\t unknow \t\t "
```

```
        printf "%-15s%-15s%-15s%s\n" $Hostname ${ip[i]} NotFound unknow
        fi
```
#Εκτύπωση το όνομα του κατανεμητή, της IP διεύθυνσης του , και ότι η #MAC διεύθυνση που ψάχνουμε δεν βρέθηκε σε αυτό τον κατανεμητή.
```
done < "/home/ftso/cisco_ip_list"

echo

echo ----------------Enterasys_Switches----------------
printf "%-15s%-15s%-15s%s\n" Hostname Host_ip port trunkstatus

while read line
        do
        #finds the name of the hostdevice
        Hostname=`snmpget -v 1 -c public ${ip_horizon[i]} .1.3.6.1.2.1.1.5.0 | cut -d "
" -f4`

        #Step 1, the MAC address is:
        tempo=`snmpwalk -v 1 -c public ${ip_horizon[i]} .1.3.6.1.2.1.17.4.3.1.1 | grep -
i "$a" | cut -d " " -f1`
        tempo=`echo $tempo | sed "s/2.17.4.3.1.1./2.17.4.3.1.2./g"`

        if [ -n "$tempo" ] ; then

          #Step 2, the bridge port tells that the MAC address belongs to bridge port
number $tempo:
                tempo=`snmpwalk -v 1 -c public ${ip_horizon[i]} .1.3.6.1.2.1.17.4.3.1.2 |
grep "$tempo" | cut -d " " -f4`

                printf "%-15s%-15s%-15s%s\n" $Hostname ${ip_horizon[i]} $tempo
        else
        printf "%-15s%-15s%-15s%s\n" $Hostname ${ip_horizon[i]} NotFound
unknow
        fi
```

done < "/home/ftso/horizon_ip_list"

#Το δεύτερο μέρος του προγράμματος είναι παρόμοιο με το πρώτο αλλά #είναι έτσι τροποποιημένο ώστε να δουλεύει για τους κατανεμητές horizon #της εταιρίας Enterasys, χρησιμοποιώντας άλλα OIDs.

13.7. - Τα script make_tftp και snmp_config_backup_tftp

Τα script που παρουσιάζονται παρακάτω χρησιμοποιούνται ώστε να αποθηκεύονται όλα τα αρχεία ρυθμίσεων των συσκευών cisco στον δίσκο του NMS μέσω ενός εξυπηρετητή tftp. Το script make_tftp δημιουργεί τα αρχεία που αρχικά είναι άδεια. Είναι προϋπόθεση του να υπάρχουν ήδη τα αρχεία στο συγκεκριμένο εξυπηρετητή. Το script snmp_config_backup_tftp δίνει την εντολή σε όλες τις συσκευές cisco να αποστείλουν το αρχείο ρυθμίσεων στο εξυπηρετητή tftp για αποθήκευση. Η εγκατάσταση ενός tftp εξυπηρετητή στο λειτουργικό σύστημα Debian Lenny είναι απλή και γίνεται εκτελώντας την εντολή

apt-get install tftp-hpa

Κατά την εγκατάσταση μας ζητάει να ορίσουμε τον κατάλογο που θα χρησιμοποιείται για την αποθήκευση των αρχείων. Στην συγκεκριμένη υλοποίηση επιλέχθηκε ο κατάλογος **/home/nms/tftp.**

Παρακάτω παρατίθεται ο κώδικας μαζί με κάποια σχόλια.

----------make_tftp----------

```
#!/bin/bash
#Initialize files for tftp, with names that retrieved with snmpget
while read line
        do
```

#Εκίνηση του βρόχου ο οποίος θα εκτελεστεί τόσες φορές όσες είναι οι IP #διευθύνσεις των συσκευών cisco που περιέχονται στο αρχείο #cisco_ip_list

```
Hostname=`snmpget -v 2c -c public $line .1.3.6.1.2.1.1.5.0 | cut -d " " -f4`
```

#Λήψη του ονόματος της συσκευής μέσω της εντολής snmpget

```
touch /home/nms/tftp/"$Hostname--$line.cfg"
```

#Δημιουργία του αρχείου με όνομα που αποτελείται από το όνομα της #συσκευής και την IP διεύθυνση.

chmod 666 /home/nms/tftp/"$Hostname--$line.cfg"
#Ορισμός των κατάλληλων δικαιωμάτων ώστε να έχει πρόσβαση σε αυτά #ο εξυπηρετητής tftp
done < "/home/ftso/cisco_ip_list"

----------snmp_config_backup_tftp----------
#!/bin/bash
simple tftp copy from the Router to the TFTP Server

COMM=private
TARGETIP=10.10.12.254
#Ορίζουμε την συμβολοσειρά community σαν private γιατί πρόκειται να #κάνουμε εγγραφή και την ίδια έχουμε ορίσει στις συσκευές σαν #συμβολοσειρά που δίνει το δικαίωμα εγγραφής. Ορίζουμε επίσης την IP #διεύθυνση του tftp εξυπηρετητή.

while read line
do
#Εκίνηση του βρόχου ο οποίος θα εκτελεστεί τόσες φορές όσες είναι οι IP #διευθύνσεις των συσκευών cisco που περιέχονται στο αρχείο #cisco_ip_list

Hostname=`snmpget -v 1 -c public $line .1.3.6.1.2.1.1.5.0 | cut -d " " -f4`
#Λήψη του ονόματος της συσκευής μέσω της εντολής snmpget

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.14.111 i 6
#Αρχικά ορίζουμε την κατάσταση αντιγραφής σε διαγραμμένη το οποίο #διαγράφει όλες τις τυχόν αποθηκεμένες πληροφορίες μέσα από το MIB.

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.2.111 i 1
#Ορίζουμε σαν πρωτόκολλο μεταφοράς το tftp

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.3.111 i 4
#Ορίζουμε ότι το αρχείο προέλευσης θα είναι το running-config

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.4.111 i 1
#Ορίζουμε ότι το αρχείο προορισμού θα είναι αρχείο που θα αποθηκευτεί #μέσω δικτύου.

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.5.111 a $TARGETIP
#Ορίζουμε την IP διεύθυνση του εξυπηρετητή tftp

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.6.111 s "$Hostname--$line.cfg"
#Ορίζουμε το όνομα του αρχείου που είναι ο συνδυασμός του ονόματος #της συσκευής και της IP διεύθυνσής του.

snmpset -c $COMM -v 1 $line 1.3.6.1.4.1.9.9.96.1.1.1.1.14.111 i 1
#Ορίζουμε την κατάσταση αντιγραφής σε ενεργή και η διαδικασία #αντιγραφής ξεκινά.
done < "/home/ftso/cisco_ip_list"

Επειδή και τα δύο script όπως επίσης και το script mac2port διαβάζουνε τις IP διευθύνσεις από το αρχείο cisco_ip_list μας είναι εύκολο να προσθέσουμε η να αφαιρέσουμε συσκευές χωρίς να επηρεαστεί η λειτουργία των script.

Τα δύο παραπάνω script εκτελούνται μία φορά την ημέρα τις δώδεκα το βράδυ μέσω του δαίμονα cron έχοντας προσθέσει μέσω της εντολής crontab –e της εγγραφές
0 0 * * * /home/ftso/make_tftp
0 0 * * * /home/ftso/snmp_ios_backup_tftp

13.8. - Το script opennms_backup και χρήσιμες εντολές διαχείρισης.

Για την δημιουργία αντιγράφων ασφαλείας των αρχείων ρυθμίσεων, των αρχείων RRD, και της βάσης δεδομένων του OpenNMS αναπτύχθηκε ένα script το οποίο

εκτελείται κάθε Κυριακή απόγευμα μέσω του crond. Παρακάτω θα δούμε τον κώδικα μαζί με κάποια σχόλια για την λειτουργία του.

---------- opennms_backup ----------

```
#!/bin/bash

opennms stop
#Σταματάμε την λειτουργία του OpenNms

su postgres -c '/usr/lib/postgresql/8.3/bin/vacuumdb -v -f -d opennms'
su postgres -c 'pg_dumpall > /tmp/onms_sql'
#Σαν χρήστης postgres εκτελούμε την εντολή vacuumdb για το #«καθάρισμα» της βάσης δεδομένων opennms και αποθηκεύουμε την #βάση μέσω της εντολής pg_dumpall

dt1="`date +%d%m%Y`_onms_etc.tar.gz"
dt2="`date +%d%m%Y`_onms_rrd.tar.gz"
dt3="`date +%d%m%Y`_onms_sql.tar.gz"
dt4=`date +%d%m%Y`
#Δημιουργία μεταβλητών για τα ονόματα των αντιγράφων ασφαλείας
#περιέχοντας την τρέχουσα ημερομηνία

cd /home/nms/opennms_backup/
#Μεταφορά στο κατάλογο όπου θα γίνει η αποθήκευση

tar cvfzP $dt1 /etc/opennms/* -R
tar cvfzP $dt2 /usr/share/opennms/share/* -R
tar cvfz $dt3 /tmp/onms_sql
#Συμπίεση και δημιουργία τριών tar συλλογών. Τα αρχεία ρυθμίσεων, τα #RRD αρχεία και την βάση δεδομένων

opennms start
#Εκινούμε  την λειτουργία του OpenNms
```

echo "scrict runned at $dt4" >> /home/nms/opennms_backup/logs/vacuumlog
#Καταγραφή στο αρχείο vaccumlog ότι το script εκτελέστηκε

---------- opennms_restore_backup ----------

Επαναφορά αντίγραφων ασφαλείας

Αφού εγκαταστήσουμε εκ νέου το λειτουργικό σύστημα Debian και το OpenNMS για να επαναφέρουμε τα αντίγραφα ασφαλείας που πήραμε με το script opennms_backup εκτελούμε τις παρακάτω εντολές:

opennms stop
Σταματάμε την λειτουργία του OpenNms

bash~#su - postgres
Αλλάζουμε χρήστη από root στον υπερχρήστη της βάσης

bash$dropdb opennms
Διαγράφουμε την τρέχουσα βάση δεδομένων του OpenNMS η οποία είναι πιθανότατα άδεια

bash~$exit
Γινόμαστε root ξανά

bash~#/usr/share/opennms/bin/install.pl -q /usr/share/opennms/etc/create.sql -l /usr/lib/postgresql/lib/opennms
Αναδημιουργούμε την βάση δεδομένων του OpenNMS

bash~#su - postgres
Αλλάζουμε χρήστη από root στον υπερχρήστη της βάσης

bash~$tar xvfz /root/******_onms_sql.tar.gz
bash~$psql -f ******_onms_sql opennms
Ανοίγουμε την συλλογή tar και επαναφέρουμε την βάση δεδομένων

Θα υπάρχουν κάποια μηνύματα λάθους ότι κάποιες δομές υπάρχουν ήδη τα οποία μπορούμε να αγνοήσουμε χωρίς να υπάρχει κάποιο πρόβλημα.

bash~$exit

Γινόμαστε root ξανά

bash~#tar xvfzP ******_onms_etc.tar.gz

Αποσυμπιέζουμε τα αρχεία ρυθμίσεων του OpenNms

bash~#rm /usr/share/opennms/share/* -rf

Σβήνουμε τα αρχεία RRD και τις αναφορές που υπάρχουν ήδη.

bash~#tar xvfzP ******_onms_rrd.tar.gz

Επαναφέρουμε τα αρχεία RRD και τις αναφορές

opennms start

Εκκινούμε την λειτουργία του OpenNms

----------logs-view, reset and backup----------

Για να εντοπίσουμε διάφορα λάθη κατά την λειτουργία του OpenNMS στα αρχεία καταγραφής εκτελούμε την παρακάτω εντολή. Επίσης μπορούμε να αντικαταστήσουμε το “ERROR” με τη λέξη “FATAL” για να δούμε ποιο κρίσιμα λάθη η με την λέξη “WARN” για να δούμε διάφορες ειδοποιήσεις.

bash#grep -r "ERROR" /var/log/opennms/* | more

Για να σβήσουμε όλα τα αρχεία καταγραφής αφού πάρουμε ένα αντίγραφο ασφαλείας εκτελούμε τις παρακάτω εντολές (σαν χρήστης root).

bash#/etc/init.d/opennms stop

Σταματάμε την λειτουργία του OpenNms

bash#tar /home/nms/opennms_backup/15052010_onms_logs.tar.gz /var/log/opennms/*

Δημιουργούμε μια συμπιεσμένη συλλογή με τα αρχεία καταγραφής του OpenNms.

bash#rm -rf /var/log/opennms/*

Σβήνουμε όλα τα αρχεία καταγραφής

bash#/etc/init.d/opennms start

Εκκινούμε το OpenNMS

Επίλογος - Συμπεράσματα

Η διαχείριση του δικτύου σε ένα δίκτυο σαν του νοσοκομείου Α.Χ.Ε.Π.Α. είναι σίγουρα ένα αναπόσπαστο κομμάτι της σωστής λειτουργίας του. Η φύση των εφαρμογών και η συνεχής λειτουργία του νοσοκομείου απαιτεί την χρήση ενός συστήματος διαχείρισης δικτύου. Με τους κανόνες δρομολόγησης που εφαρμόστηκαν έγινε δυνατή και η διαχείριση απομακρυσμένων τοποθεσιών που είναι εκτός της κτηριακής υποδομής του νοσοκομείου.

Ένα βασικό κριτήριο για την επιλογή του OpenNMS ήτανε ότι αποτελεί ελεύθερο-ανοιχτού κώδικα λογισμικό πράγμα που αποτελεί εξοικονόμηση χρημάτων για τον οργανισμό καθώς και την δυνατότητα βελτίωσης η τροποποίησης ενός τμήματος του κώδικα καθώς και την εύκολη δημιουργία πρόσθετων εφαρμογών που μπορεί να χρειαστούνε μελλοντικά. Αμέσως μετά τα βασικά πλεονεκτήματα που συντέλεσαν στην επιλογή του ήτανε η έμφαση που δίνει στις υπηρεσίες που τρέχουν στους εξυπηρετητές, το εύχρηστο γραφικό περιβάλλον, η ευκολία πρόσθήκης νέων συσκευών, καθώς επίσης και η ευκολία εγκατάστασής του αναγνωρίζοντας πολλές συσκευές απευθείας χωρίς να χρειάζονται περαιτέρω ρυθμίσεις. Ένα βασικό μειονέκτημα που σημειώθηκε κατά την διάρκεια λειτουργίας του ήτανε η δυσκολία αναβάθμισής του καθώς σε κάθε νέα έκδοση γίνονται μεγάλες αλλαγές που έχει σαν αποτέλεσμα να αλλάζει η δομή των αρχείων ρυθμίσεων κάποια από τα οποία είναι αρκετά μεγάλα σε μέγεθος και εάν έχουν τροποποιηθεί αρκετά ο χρόνος αναβάθμισης είναι αρκετά μεγάλος. Επίσης η προσθήκη MIB αρχείων για να αναγνωρίζονται τα SNMP traps συσκευών που δεν περιέχει ήδη είναι περίπλοκη καθώς χρειάζεται να μετατραπούνε πρώτα σε xml αρχεία με συγκεκριμένες ετικέτες πράγμα που κάνει αυτόματα το εργαλείο mib2opennms, αλλά σε μερικά MIB αρχεία δεν δουλεύει σωστά και είναι απαραίτητη η τροποποίηση του MIB αρχείου. Παρόλα αυτά το καλό είναι ότι περιέχει αρκετά MIBs εξ αρχής και ίσως να μην είναι απαραίτητη η εισαγωγή κάποιου επιπλέον. Ένα τελευταίο μειονέκτημα είναι ότι ο γραφικός του χάρτης δουλεύει μόνο σε Internet Explorer πράγμα που στις μετέπειτα εκδόσεις διορθώθηκε.

Το σύστημα διαχείρισης δικτύου OpenNMS δουλεύει εδώ και κάποιους μήνες χωρίς προβλήματα και έχει φανεί ιδιαίτερα χρήσιμο στους εργαζόμενους στο

τμήμα πληροφορικής του νοσοκομείου Α.Χ.Ε.Π.Α. Πέρα από την υλοποίηση του συστήματος βέβαια η παρούσα εργασία έχει σαν σκοπό να αποτελέσει εγχειρίδιο για την λειτουργία του συστήματος και μια βοήθεια σε τυχόν προβλήματα που μπορεί να προκύψουν, καθώς και για την εγκατάσταση νέων δικτυακών συσκευών.

Κάποια από τα πράγματα που θα μπορούσαν να γίνουν μελλοντικά θα ήτανε να γίνει μια προσεκτική επιλογή των interface των κατανεμητών που θέλουμε να παρακολουθούμε την κίνηση τους και να βελτιωθεί η αυτόματη δημιουργία αναφορών η οποίες μπορούν να στέλνονται καθημερινά μέσω ηλεκτρονικού ταχυδρομείου σε αρχείο PDF. Επίσης μέσα στην βάση του OpenNMS υπάρχει η δυνατότητα αποθήκευσης επιπλέον πληροφοριών για τις διαχειριζόμενες συσκευές όπως π.χ. ο σειριακός αριθμός, το μοντέλο, η ημερομηνία εγκατάστασης μια συσκευής κ.α. όπου μια πλήρης καταγραφή των κόμβων θα βοηθήσει την διαχείριση γενικότερα.

Η υλοποίηση ενός συστήματος διαχείρισης δικτύου ήτανε για εμένα πολύ ενδιαφέρον και διδακτική για τον λόγο ότι με έβαλε στην διαδικασία να καταλάβω πως είναι δομημένο ένα πληροφοριακό σύστημα, τι υπηρεσίες προσφέρει ο κάθε εξυπηρετητής και ποια πράγματα πρέπει να προσέχουμε ιδιαίτερα στην λειτουργία του. Η ανομοιογένεια των συσκευών μου έδωσε την δυνατότητα να δω και να πάρω μια ιδέα για διάφορα λειτουργικά συστήματα καθώς και το ποια είναι η χρήση τους σε ένα πληροφοριακό σύστημα. Συνοψίζοντας η αντικατάσταση του συστήματος διαχείρισης δικτύου από το Spectrum στο ποιο σύγχρονο σύστημα OpenNMS αποδείχτηκε χρήσιμη και έδωσε νέες δυνατότητες διαχείρισης στο δίκτυο του νοσοκομείου Α.Χ.Ε.Π.Α., σε συνδιασμό με την αντικατάσταση των κατανεμητών Cabletron σε κατανεμητές Cisco.

Βιβλιογραφία

- http://www.opennms.org/
- http://www.cacti.net/
- http://www.nagios.org/
- http://www.net-snmp.org/
- http://oss.oetiker.ch/mrtg/
- http://www.mrtg.org/rrdtool/
- http://www.netmode.ntua.gr
- http://www.netstatz.com/xenonms/Debian_ONMS_HOWTO_2.4-5.html
- http://www.cisco.com
- http://nms.gdd.net/index.php/Main_Page
- http://www.redbooks.ibm.com/redbooks/pdfs/sg246606.pdf
- http://docs.hp.com/en/B2355-90131/snmpd.conf.4.html
- http://ccie20728.wordpress.com/2008/05/20/get-the-cisco-configuration-over-snmp/
- Συμπληρωματικές Σημειώσεις στα πλαίσια του μαθήματος. «ΔΙΚΤΥΑ Η/Υ ΙΙΙ». (Διαχείριση Δικτύων και το πρωτόκολλο SNMP). Δρ. Περικλής Χατζημίσιος 2007
- Essential SNMP – D. Mauro, K. Schmidt, O'Reilly Media, July 2001