Ανάπτυξη web-based εργαλείου συλλογής πληροφοριών με τη χρήση του SNMP

18 Μαρτίου 2010

πτυχιακή εργασία των φοιτητών:
Παπαδόπουλος Χαράλαμπος <chpapa04@yahoo.co.uk>,
Σιαπέρας Δημήτριος <dimis_983@yahoo.gr>
Θεσσαλονίκη, Μάρτιος 2010

Περιεχόμενα

Πρόλογος

Καθώς το διαδίκτυο γνώριζε μεγάλη ανάπτυξη στη δεκαετία του 1980, έπρεπε να βρεθούν τρόποι για την εποπτεία των δικτύων έτσι ώστε να αντιμετωπισθούν κάποια προβλήματα. Ένα από τα προβλήματα αυτά ήταν ο έλεγχος της ομαλής λειτουργίας των δικτυακών συσκευών από απόσταση. Πριν εμφανισθεί κάποιο πρωτόκολλο διαχείρισης δικτύου, τέτοιου είδους έλεγχοι γίνονταν με τη χρησιμοποίηση του πρωτοκόλλου ICMP με τη βοήθεια μηνυμάτων ελέγχου προσπελασιμότητας και κατάστασης προορισμού, τα μηνύματα αίτησης/απάντησης αντήχησης και άλλα. Επίσης, όσον αφορά τον έλεγχο και την απλή διαχείριση των δικτύων, γινόταν χρήση των ifconfig, netstat, πράγμα που με την ταυτόχρονη ανάπτυξη του διαδικτύου και της τεχνολογίας, δεν ήταν πρακτικό.

Μία πρώτη μορφή για την εποπτεία δικτύων, εμφανίστηκε το 1987 με την έκδοση του Simple Gateway Monitoring Protocol (SGMP). Το SGMP χρησιμοποιήθηκε για την ανάκτηση ή την αλλαγή τιμών των δρομολογητών διαδικτύου. Τα μηνύματα του SGMP, χρησιμοποιούσαν το πρωτόκολλο UDP και τη θύρα 153 για αποστολή. Περιλαμβάνει τέσσερις μορφές μηνυμάτων, οι οποίες είναι οι ακόλουθες: Get Request, για ανάκτηση πληροφοριών, Get Response, η οποία είναι η απάντηση στην αίτηση για κάποια πληροφορία και περιλαμβάνει την ίδια την πληροφορία, Trap Request, που αιτείται την απάντηση του παραλήπτη όταν συμβεί κάποιο γεγονός και Set Request, η οποία θέτει τιμές σε μεταβλητές του δρομολογητή διαδικτύου. Το πρωτόκολλο SGMP χρησιμοποιήθηκε σαν βάση για την ανάπτυξη του πρωτοκόλλου SNMP. Οι πληροφορίες που έχουν να κάνουν με το SGMP, αφορούν:

- Τύπους των θυρών των δρομολογητών
- Κατάσταση θυρών
- Τύπο διαδρομών
- Πρωτόκολλα δρομολόγησης

Οι πληροφορίες που επιλέχθηκαν να συλλέγονται, είναι καθορισμένες για κάθε δικτυακή συσκευή. Αυτό συμβαίνει γιατί κάθε συσκευή έχει μία ή περισσότερες θύρες οι οποίες έχουν μια κατάσταση λειτουργίας, χρησιμοποιούν πρωτόκολλα και αποθηκευουν στατιστικές πληροφορίες που αφορούν τις θύρες.

Με την έκδοση του RFC1028 για το SGMP, ξεκίνησε η ανάπτυξη του πρωτοκόλλου SNMP το οποίο δημοσιεύτηκε το 1988 και δέχθηκε ευρείας αποδοχής.

Το Simple Network Management Protocol (SNMP) είναι ένα πρωτόκολλο το οποίο χρησιμοποιείται ευρέως για τη συλλογή πληροφοριών από δικτυακές συσκευές. Το τελευταίο λειτουργεί μέσω μιας δενδροειδούς δομής, τη βάση δεδομένων διαχείρισης πληροφοριών (MIB), που κατηγοριοποιεί όλες τις πιθανές πληροφορίες που μπορούμε να λάβουμε από διάφορες συσκευές δικτύων.

Περίληψη

Με την παρούσα πτυχιακή εργασία δημιουργήθηκε ένα web – based εργαλείο συλλογής πληροφοριών. Σκοπός της εφαρμογής αυτής είναι η συλλογή χρήσιμων πληροφοριών από δικτυακές συσκευές. Η λήψη αυτών των πληροφοριών έγινε με τη χρήση του πρωτοκόλλου SNMP (Simple Network Management Protocol).

Στο πρώτο κεφάλαιο γίνεται αναφορά στα εργαλεία που χρησιμοποιήθηκαν για τη δημιουργία της εφαρμογής, καθώς και μια μικρή περιγραφή για το καθένα απο αυτά. Στο δεύτερο κεφάλαιο γίνεται εισαγωγή στο πρωτόκολλο SNMP και επεξήγηση των βασικών εννοιών που το χαρακτηρίζουν. Στη συνέχεια περιγράφεται ο τρόπος οργάνωσης των διαθέσιμων πληροφοριών μέσω του δένδρου MIB και αναφέρονται οι βασικές λειτουργίες του πρωτοκόλλου. Στο τρίτο κεφάλαιο πραγματοποιείται ιστορική αναδρομή στο SNMP και επεξήγηση των λειτουργιών των βασικών εκδόσεων του SNMP καθώς και περιγραφή του SNMP PDU της κάθε έκδοσης.

Στο τέταρτο κεφάλαιο αναλύεται η διαδικασία υλοποίησης της εφαρμογής. Γίνεται περιγραφή της διαδικασίας ελέγχου και χρήσης της IP διεύθυνσης και του community name. Στη συνέχεια επεξηγείται η ροή του προγράμματος και ο τρόπος λήψης και εμφάνισης των πληροφοριών. Επισής δίνονται πληροφορίες σχετικά με το μενού που χρησιμοποιήθηκε στην εφαρμογή και τέλος, ανάλυση της χρήσης του Session στην ιστοσελίδα.

Στο πέμπτο κεφάλαιο γίνεται περιγραφή του πρωτοκόλλου CDP της εταιρίας Cisco. Στο έκτο κεφάλαιο περιγράφεται ο τρόπος ανεύρεσης του CDP δικτύου καθώς και οι συναρτήσεις που χρησιμοποιήθηκαν για την οργάνωση των χρήσιμων πληροφοριών αυτού. Στο έβδομο κεφάλαιο γίνεται αναφορά των συναρτήσεων που χρησιμοποιήθηκαν στην εφαρμογή και επεξηγείται ο τρόπος λειτουργίας τους.

Στο όγδοο κεφάλαιο δίνεται η δομή της ιστοσελίδας και οι πληροφορίες εγκατασταση της σε κάποιο server. Στο ένατο κεφάλαιο παρατίθεται ο οδηγός (manual) της ιστοσελίδας. Τέλος, δίνονται κάποια συμπερασματικά στοιχεία σχετικά με το δίκτυο του TEI.

Abstract

This thesis is about the creation of a web – based information collector tool. The purpose of this application is the collection of information from network devices. Obtaining such information was done by using the protocol SNMP (Simple Network Management Protocol).

The first chapter refers to the tools that have been used to create the application. A short description of each tool is also included. The second chapter is an introduction to the SNMP protocol and expains the basic concepts which characterize that protocol. It continues with the description of the organised structure of the available information through the Management Information Base tree and the essential functions of the protocol are refered. The third chapter describes the history of the SNMP and explains the functions of the basic versions of SNMP and an SNMP PDU description is being made.

The fourth chapter discusses the implementation process of the application. A description of the control and use of IP addresses and community name is being made. Then, the flow of the application, the reception and the display of the useful information, is explained . Also Information is given for the menu which is used in the implementation of the application and finally, an analysis of the use of Session on the site is being made.

The fifth chapter contains a description of the Cisco protocol, CDP and the sixth chapter describes how to find the CDP network and the functions used to organize this useful information. In the seventh chapter a reference of the functions used in the application and their operation is explained.

In the eighth chapter, the structure of the website is given as well as information about the installation process in a server. In the ninth chapter, the manual of the application is given. Finally, in the tenth chapter, some statistical information about the Technological Educational Institute of Thessaloniki network is given.

1 Εισαγωγή

Το θέμα της εργασίας αυτής είναι η ανάπτυξη ενός web-based εργαλείου συλλογής πληροφοριών με τη χρήση του SNMP. Η δημιουργία του εργαλείου αυτού έγινε με τη βοήθεια της PHP, της HTML, του Smarty καθώς και του CSS για θέματα εμφάνισης ενώ η συλλογή των απαραίτητων πληροφοριών πραγματοποιήθηκε μέσω του net-snmp καθώς και του phpsnmp. Η ανάλυση των παραπάνω παρατίθεται στις επόμενες σελίδες ενώ δίνονται και πληροφορίες σχετικά με τον αριθμό των εκδόσεων.

Επίσης,θα ακολουθήσει ανάλυση ως προς το πώς χρησιμοποιήσαμε το net-snmp και το phpsnmp μέσω της php καθώς και ποιες διαδικασίες ακολουθήσαμε ώστε να πάρουμε το επιθυμητό αποτέλεσμα. Επίσης θα δοθούν πληροφορίες σχετικές με την επεξεργασία και την εμφάνιση των δεδομένων που συλλέξαμε.

Στη συνέχεια, θα αναλυθούν κάποια κύρια προβλήματα που αντιμετωπίσαμε κατά τη δημιουργία αυτού του web-based εργαλείου. Τα προβλήματα αυτά μπορεί να είναι λογικά ή προγραμματιστικά.

Τέλος, θα δοθεί ένας οδηγός (manual) στον οποίο αναλύονται οι διαδικασίες που πρέπει να ακολουθηθούν για να πάρει ένας απλός χρήστης ή ένας διαχειριστής δικτύου το επιθυμητό αποτέλεσμα. Ειδικότερα για τους απλούς χρήστες θα δοθεί και μια μικρή ανάλυση του κάθε πίνακα που μπορεί να εμφανιστεί.

1.1 Περιβάλλον ανάπτυξης

Η ανάπτυξη της εφαρμογής πραγματοποιήθηκε σε περιβάλλον Linux και συγκεκριμένα στην έκδοση UBUNTU 8.04. Για την αρχική ανάπτυξη της εφαρμογής εγκαταστάθηκε ο Apache server έκδοση 2.0, τόσο για τον έλεγχο των πρώτων σταδίων της υλοποίησης του προγράμματος όσο και για τη διενέργεια των αρχικών δοκιμών μας. Το αρχικό τοπικό δίκτυο που χρησιμοποιήθηκε αποτελούνταν από τρεις υπολογιστές και ένα ADSL router. Στα πρώτα στάδια για την ανάπτυξη του κώδικα της εφαρμογής χρησιμοποιήθηκε η PHP έκδοση 5 καθώς και η HTML ενώ στα επόμενα στάδια ο κώδικας επαναδιατυπώθηκε με τη χρήση του Smarty για τον πλήρη διαχωρισμό της HTML από την PHP.

Για τη συλλογή των πληροφοριών του δικτύου χρησιμοποιήθηκε το πακέτο net-snmp μέσω της PHP. Με αυτόν τον τρόπο λαμβάνουμε τις πληροφορίες των δικτυακών συσκευών του δικτύου. Στη συνέχεια χρησιμοποιήθηκε ένα δεύτερο πακέτο, το php-snmp, λόγω των συγκεκριμένων δυνατοτήτων που μας παρέχει. Οι λόγοι χρήσης του κάθε πακέτου θα αναλυθούν εκτενώς στα επόμενα κεφάλαια.

Για την μορφοποίηση της σελίδας χρησιμοποιήθηκε το CSS (Cascading Style Sheets).

1.2 Βασικές πληροφορίες εργαλείων

Αρχικά παρατίθεται ο ορισμός της PHP, η οποία είναι μια ευρέως διαδεδομένη γλώσσα προγραμματισμού για τη δημιουργία ιστοσελίδων με δυναμικό περιεχόμενο και μπορεί να ενσωματωθεί μέσα σε HTML.

Η PHP δημιουργήθηκε το 1995 από τον Rasmus Lerdorf για προσωπική του χρήση χρησιμοποιώντας τη γλώσσα Perl. Η πρώτη της χρήση αφορούσε τη μέτρηση και την αποθήκευση της κίνησης στη σελίδα του και αρχικά ονομαζόταν personal home page tools. Αργότερα έγραψε πολλά προγράμματα σε γλώσσα C, τα οποία μπορούσαν να επικοινωνήσουν με βάση δεδομένων ενώ ταυτόχρονα επέτρεψε στους χρήστες να αναπτύξουν απλές δυναμικές ιστοσελίδες. Αποφάσισε ακόμα, να διαθέσει στο κοινό τον πηγαίο κώδικα της php/fi ώστε να μπορούν να τον χρησιμοποιήσουν όλοι. Η πρώτη επίσημη έκδοση παρουσιάστηκε στις 8 Ιουνίου 1995 και ονομαζόταν PHP Version 2. Στη συνέχεια οι δύο Ισραηλινοί Zeev Suraski και Andi Gutmans βελτίωσαν την υπάρχουσα έκδοση της PHP και το 1997 εκδόθηκε η PHP Version 3 όπου PHP σημαίνει Hypertext Preprocessor. Η επίσημη έκδοση της version 3 έγινε εννιά μήνες μετά την πρώτη δημόσια κυκλοφορία. Για τη δημιουργία της έκδοσης 4 (Μάιος 2000) και έκδοσης 5 (Ιούλιος 2004) αναδιαρθρώθηκε ο πυρήνας της PHP, ο οποίος πήρε το όνομα Zend από τα αρχικά των μικρών ονομάτων των δημιουργών της κι έτσι οριστικοποιήθηκε η PHP στη σημερινή της μορφή. Η παρούσα εφαρμογή βασίστηκε στην PHP έκδοση 5.2.4.

Το smarty είναι μια μηχανή προτύπων για την PHP. Στην ουσία διαχωρίζει τη λογική της εφαρμογής από την παρουσίαση. Ένα από τα κύρια πλεονεκτήματα είναι ότι μεταφράζει ξεχωριστά και μόνο σε περίπτωση αλλαγών τα μεταφράζει ξανά. Η λογική είναι ότι μπορείς να ασχοληθείς χωριστά με τον κώδικα της εφαρμογής και σε διαφορετικό αρχείο να ασχοληθείς με την παρουσίαση της εφαρμογής έτσι ώστε να επικεντρωνόμαστε σε κάθε πρόβλημα ξεχωριστά. Μια άλλη δυνατότητα που παρέχει το smarty είναι ότι είναι πολύ γρήγορο και παράλληλα με τη χρήση πολλών διαδικασιών (functions) και μεταβλητών (variables) μπορούμε να επεκτείνουμε τη χρήση του εκμεταλλευόμενοι τις πολλές δυνατότητες που μας προσφέρει. Για την εφαρμογή μας χρησιμοποιήσαμε την έκδοση 2.6.20 του Smarty.

Για τη συλλογή των πληροφοριών από το δίκτυο προχωρήσαμε στη χρήση του πρωτοκόλλου SNMP μέσω της PHP. Το πρώτο πακέτο που χρησιμοποιήθηκε είναι το net-snmp έτσι ώστε να γίνει δυνατή η λήψη πληροφορίων από μία συγκεκριμένη συσκευή κάθε φορά. Για την παραπάνω διαδικασία, καθώς και την εμφάνιση των πληροφοριών, τα SNMP ερωτήματα ενσωματώθηκαν μέσα στην PHP. Στη συνέχεια, για να είναι εφικτή η λήψη πληροφορίων από πολλές συσκευές και σε σύντομο χρονικό διάστημα χρησιμοποιήθηκε το php-snmp. Μια από τις βασικές λειτουργίες του τελευταίου είναι η αποστολή ενός SNMP αιτήματος σε πολλές συσκευές ταυτόχρονα ή η αποστολή πολλών SNMP αιτημάτων σε μια συσκευή. Ακολούθως, γίνεται εκτενέστερη αναφορά σχετικά με το net-snmp και με το phpsnmp δίνονται παραδείγματα για το πώς μπορούμε να θέτουμε ερωτήματα σε μια συσκευή ή σε περισσότερες.

2 SNMP

Όπως αναφέρθηκε ήδη, το πρωτόκολλο που χρησιμοποιήθηκε για τη λήψη των επιθυμητών πληροφοριών, είναι το Simple Management Network Protocol (SNMP). Σε αυτήν την ενότητα αναλύεται η λειτουργία του πρωτοκόλλου αυτού, καθώς και οι υπάρχουσες εκδόσεις του (SNMPv1, SNMPv2 και SNMPv3). Τέλος, εξηγούνται οι λόγοι για τη χρήση της έκδοσης SNMPv2, SNMPv2c στην παρούσα εφαρμογή.

Το πρωτόκολλο διαχείρισης απλών δικτύων (SNMP) είναι ένα πρωτόκολλο εποπτείας συσκευών δικτύου και συλλογής πληροφοριών απόδοσης για τις βάσεις δεδομένων διαχείρισης πληροφοριών (MIB). Έχει σχεδιαστεί με βάση το TCP/IP και λειτουργεί στο επίπεδο εφαρμογών.

TCP/IP Architecture Layer	Protocols
Application	HTTP, POP3, SMTP, **SNMP** , DNS, FTP
Transport	TCP, UDP
Inter network	IP, ARP, RARP, ICMP
Network interface	Ethernet, Frame Relay

Το SNMP χρησιμοποιεί UDP πακέτα για την αποστολή αιτημάτων στις συσκευές δικτύου και αποτελείται από τρία βασικά μέρη:

1. Διαχειρίσιμες συσκευές (managed devices)
2. Πράκτορες (agents)
3. Σταθμοί διαχείρισης δικτύου (network management stations)

Οι managed devices είναι οι συσκευές από τις οποίες θέλουμε να πάρουμε πληροφορίες. Οι πληροφορίες που λαμβάνουμε γίνονται άμεσα διαθέσιμες στους network management stations (NMS). Οι NMS είναι οι σταθμοί όπου καταλήγουν οι πληροφορίες οι οποίες λαμβάνονται απο τις διαχειρίσιμες συσκευές. Αξίζει να σημειωθεί ότι δεν είναι απαραίτητο να υπάρχει μόνο ένα NMS σε κάθε δίκτυο. Μερικά παραδείγματα διαχειρίσιμων συσκευών είναι: οι routers, τα switches, τα ip-phones, οι υπολογιστές, οι εκτυπωτές κλπ.

Ένας agent είναι ένα λογισμικό το οποίο είναι εγκατεστημένο σε κάθε συσκευή απ΄ την οποία θέλουμε να παίρνουμε πληροφορίες. Το λογισμικό αυτό γνωρίζει τις απαραίτητες πληροφορίες της συγκεκριμένης συσκευής και όταν ο χρήστης κάνει ένα αίτημα σε αυτή, ο agent μετατρέπει την πληροφορία σε συμβατή μορφή χρησιμοποιώντας το SNMP.

Το SNMP χρησιμοποιεί ένα δικό του τρόπο για την οργάνωση των διαθέσιμων πληροφοριών προκειμένου να οργανώσει τις πληροφορίες που επιθυμεί να λάβει. Αυτό γίνεται μέσω των MIBs (Management Information Bases) τα οποία περιγράφουν τη δομή των διαχειρίσιμων πληροφοριών. Τα MIBs χρησιμοποιούν ένα ιεραρχικό τρόπο για την κατηγοριοποίηση του κάθε στοιχείου που θέλουμε να πάρουμε. Το κάθε στοιχείο δηλώνεται μοναδικά και ονομάζεται OID (Object Identifier).

Προκειμένου να γίνει κατανοητή η λειτουργία του MIB θα πρέπει να εξηγηθεί καλύτερα τι είναι τα OID. Ένα OID είναι η τελική διαδρομή που δείχνει το που μπορεί να βρεθεί το επιθυμητό στοιχείο και εκπροσωπείται από ένα σύνολο αριθμών χωρισμένων με τελείες.

Π.χ 1.3.6.1.2.1.1.3

Τα OID αντιπροσωπεύουν ένα πλήθος πληροφοριών, αλλά το κάθε OID είναι μοναδικό, πράγμα που συνεπάγεται οτι αντιστοιχεί μόνο σε μία συγκεκριμένη πληροφορία ή έναν πίνακα πληροφοριών. Το δέντρο των MIB περιγράφει τα εκάστοτε OID.

2.1 MIB δένδρο

Το δέντρο των MIB είναι μια δομή η οποία κατηγοριοποιεί όλα τα OID ιεραρχικά. Για την ανεύρεση του συγκεκριμένου OID πρέπει να ανατρέξουμε από τη ρίζα του δέντρου προς το συγκεκριμένο αντικείμενο. Το κάθε στοιχείο του δέντρου αντιπροσωπεύεται από έναν αριθμό. Η διαδρομή από τη ρίζα μέχρι το ζητούμενο στοιχείο είναι το σύνολο των αριθμών που αναφέραμε παραπάνω και ονομάζεται OID.

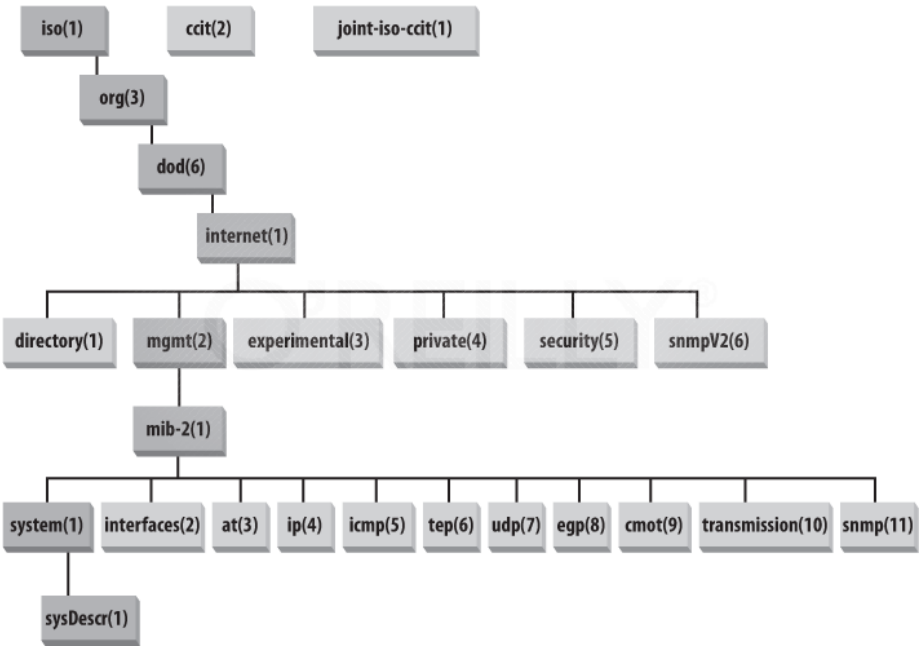
Για παράδειγμα, η διαδρομή που μας δείχνει την ώρα μιας συσκευής είναι η εξής: 1.3.6.1.2.1.1.3.

Το κάθε νούμερο της διαδρομής αυτής αντιστοιχεί σε μια υποκατηγορία/επίπεδο του δέντρου:

iso(1).indent-org(3).dod(6).internet(1).mgmt(2).mib(1).system(1).sysUpTime(3).

Παίρνοντας μόνο τους αριθμούς 1.3.6.1.2.1.1.3 και χρησιμοποιώντας τις κατάλληλες εντολές μπορούμε να πάρουμε τη συγκεκριμένη πληροφορία.

Το δέντρο των MIB μοιάζει με το παρακάτω:



Ακολουθώντας τα διαθέσιμα νούμερα παρατηρείται μέχρι ενός σημείου η επιθυμητή διαδρομή. Σημειώνεται οτι δεν είναι φανερή ολόκληρη η διαδρομή καθώς

το MIB είναι πάρα πολύ μεγάλο, οπότε δεν είναι δυνατόν να παρουσιασθεί σε όλη του την έκταση.

2.2 Εξήγηση του MIB tree

Όπως είναι εμφανές η ρίζα του δέντρου είναι κενή. Οι «μεταβλητές» που υπάρχουν στο MIB δένδρο είναι επεκτάσιμες. Αυτό σημαίνει ότι οι κατασκευαστές μπορούν να προσθέσουν νέες μεταβλητές στο δένδρο. Το δένδρο MIB είναι το ίδιο παγκοσμίως και συντηρείται από τον ISO (International Standards Organization).

2.3 Κατηγορίες

Οι συσκευές δικτύων δεν είναι όλες κατασκευασμένες από έναν κατασκευαστή, ενώ τα χαρακτηριστικά των συσκευών δικτύων ποικίλλουν μεταξύ των κατασκευαστών. Επίσης οι συσκευές δεν είναι όλες ίδιες ως προς τις λειτουργίες τους. Η χρήση του SNMP θα πρέπει να πραγματοποιείται ανεξάρτητα απο το γεγονός αυτό. Αυτή είναι δουλειά του δένδρου και των κατασκευαστών. Στο MIB δένδρο υπάρχουν κατηγορίες ανάλογα με τον τύπο των συσκευών αλλά και ανάλογα με τον κατασκευαστή αν αυτό είναι επιθυμητό από τον ίδιο τον κατασκευαστή. Πέρα από τις γενικές πληροφορίες μπορούν οι κατασκευαστές να προσθέσουν τα δικά τους OID ανάλογα ανάλογα με τα χαρακτηριστικά των συσκευών που κατασκευάζουν. Έτσι, όπως γίνεται εμφανές και στην εικόνα του MIB δένδρου, έχουν δοθεί «διαδρομές» OIDS που ανήκουν σε κατασκευαστές. Για παράδειγμα, για να πάρουμε πληροφορίες από μία Cisco συσκευή που αφορά το πρωτόκολλο CDP που μόνο οι συσκευές της Cisco χρησιμοποιούν, θα πρέπει να βρούμε τη διαδρομή που έχει καθορίσει η ίδια η Cisco για να πάρουμε αυτήν την πληροφορία.

Έτσι, για τη συγκεκριμένη πληροφορία θα πρέπει να δοθεί η παρακάτω διαδρομή:

```
1.3.6.1.4.1.9.X (όπου X, η πληροφορία της συσκευής)
```

η οποία διαδρομή μεταφράζεται ως:

```
iso(1).indent-org(3).dod(6).internet(1).private(4)

.enterprise(1).cisco(9).[πληροφορία που θέλουμε]
```

Παρατηρούμε ότι υπάρχει η κατηγορία private, σε υποκατηγορίες της οποίας υπάρχουν τα MIBs των κατασκευαστών.

Πέρα από τους διάφορους κατασκευαστές, το MIB δένδρο περιέχει τις γενικές πληροφορίες που είναι διαθέσιμες. Πολλές από αυτές είναι κοινές σε συσκευές δικτύου και ανεξάρτητες από τον κατασκευαστή της συσκευής. Για παράδειγμα, σε περίπτωση που θέλουμε να πάρουμε πληροφορίες όπως την ώρα του συστήματος ή την ηλεκτρονική διεύθυνση του διαχειριστή από κάποιες συσκευές, θα πρέπει να γίνει από τη διαδρομή προς τις πληροφορίες συστήματος. Η διαδρομή αυτή είναι:

```
1.3.6.1.2.1.1.X
```

η οποία μεταφράζεται ως:

```
iso(1)
  .org(3)
    .dod(6)
      .internet(1)
        .mgmt(2)
          .mib(1)
            .system(1)
              .[information we want]
```

και μας δίνει πρόσβαση στις πληροφορίες συστήματος.

Ωστόσο, μέσω της παραπάνω διαδρομής, παρέχεται η δυνατότητα να ληφθούν και άλλου είδους πληροφορίες. Αν θέλουμε για παράδειγμα, να πάρουμε πληροφορίες που αφορούν τη δικτυακή λειτουργία της συγκεκριμένης συσκευής σε όλα τα επίπεδα (μοντέλο TCP/IP), μπορούμε μέσω του δένδρου να βρούμε τις κατάλληλες διαδρομές προς αυτήν την πληροφορία. Αν λοιπόν αντικατασταθεί από την προηγούμενη διαδρομή το system(1) και χρησιμοποιηθεί από το δένδρο κάποιο άλλο νούμερο (που αντιπροσωπεύει μια διαφορετική κατηγορία), έχουμε τη δυνατότητα λήψης τέτοιου είδους πληροφοριών. Παρακάτω παρουσιάζονται κάποιες από τις κατηγορίες της διαδρομής 1.3.6.1.2.1.1:

OID
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).interfaces(2)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).at(3)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).ip(4)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).icmp(5)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).tcp(6)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).udp(7)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).transmission(10)
iso(1).org(3).dod(6).internet(1).mgmt(2).mib(1).snmp(11)

Όπως φαίνεται και από το όνομα της κάθε διαδρομής, είναι κατανοητό το είδος των πληροφοριών που μπορούν να ληφθούν από κάθε κατηγορία. Για παράδειγμα, αν επιθυμούμε τη συλλογή πληροφοριών που αφορούν τις θύρες μιας δικτυακής συσκευής, όπως ένας δρομολογητής, θα επιλέξουμε τη διαδρομή 1.3.6.1.2.1.2.X. Αξίζει να τονισθεί ότι οι διαδρομές που χρησιμοποιήθηκαν έως τώρα, αναφέρονται σε κατηγόριες και όχι σε συγκεκριμένες τιμές. Έτσι, για τη λήψη μιας συγκεκριμένης πληροφορίας από την κατηγορία των θυρών που χρησιμοποιήθηκαν προηγουμένως, είναι απαραίτητο να δωθεί και ο αριθμός που θα καθορίζει ποια πληροφορία επιθυμούμε.

2.3.1 Ανακεφαλαίωση

Το SNMP χρησιμοποιεί το δένδρο MIB για να οργανώσει ιεραρχικά τις διαθέσιμες πληροφορίες που μπορεί να λάβει ένα NMS. Το Δένδρο MIB είναι ένα δένδρο που έχει κενό όνομα ρίζας και κάθε στοιχείο του αντιπροσωπεύεται από έναν αριθμό. Μια διαδρομή MIB μας δείχνει τον δρόμο προς την πληροφορία που θέλουμε και αντιπροσωπεύεται απο τα ονόματα των κατηγοριών, χωρισμένα με τελείες. Αν αντικαταστήσουμε τα ονόματα των κατηγοριών με τους αντίστοιχους αριθμούς,

τότε θα έχουμε το OID της διαδρομής μας. Το δένδρο είναι υπό την επίβλεψη του ISO και είναι κοινό σε ολόκληρο τον κόσμο.

2.4 Οι λειτουργίες του SNMP

Παρακάτω παρατίθενται κάποιες από τις βασικές λειτουργίες του SNMP. Το SNMP υποστηρίζει κυρίως 3 τύπους εντολών.

1. Read
2. Write
3. Trap

Η λειτουργία read του SNMP χρησιμοποιείται από τις συσκευές διαχείρισης (NMS) για τη λήψη των πληροφοριών που επιθυμεί ο κάθε χρήστης.

Η λειτουργία write, χρησιμοποιείται για τη διαχείριση των συσκευών από τα NMS, ενώ παρέχει και τη δυνατότητα αλλαγής τιμών που υπάρχουν στο MIB δένδρο.

Η λειτουργία trap χρησιμοποιείται για να ενημερώσει τους NMS για συγκεκριμένα γεγονότα που λαμβάνουν χώρα σε μία ή περισσότερες συσκευές δικτύου. Η διαδικασία είναι η εξής, καθορίζουμε το γεγονός το οποίο θα προκαλέσει την trap, καθορίζουμε το εύρος των συσκευών που μπορούν να προκαλέσουν μια trap. Μόλις πραγματοποιηθεί αυτό το συγκεκριμένο γεγονός, το managed system δημιουργεί μια trap και την αποστέλλει στο NMS. Στην ουσία μία trap (παγίδα) είναι μία αναφορά προς τον NMS από τις Managed Systems, που δημιουργείται και αποστέλλεται όταν συμβούν συγκεκριμένα γεγονότα.

3 Ιστορία του SNMP

Η ιστορία του SNMP ξεκίνησε το 1988. Ιδρυτής ήταν ο Jeffrey Case ο οποίος το 1987 ήταν διαχειριστής δικτύων και καθηγητής της επιστήμης της πληροφορικής στο πανεπιστήμιο του Tennessee. Το 1988 μαζί με τον Ken Key, ο οποίος ήταν απόφοιτος πληροφορικής και στη συνέχεια συνεργάτης του Case, δημιούργησαν το πρωτόκολλο διαχείρισης δικτύων. Αρχικά, είχε δημιουργηθεί το SGMP (Simple Gateway Management Protocol), το οποίο το 1988 αναδημιουργήθηκε και ονομάστηκε SNMP. Η κύρια λειτουργία του ήταν η διαχείριση και εποπτεία δικτύων και συσκευών δικτύων. Το SNMP χρησιμοποιεί το μοντέλο managed device, agent, network-management systems (NMSs). Οι managed device είναι κάποιες συσκευές δικτύου οι οποίες περιέχουν ένα snmp agent. Στον υπολογιστή, μέσω του οποίου θα γίνει η διαχείριση των συσκευών, υπάρχει το λογισμικό διαχείρισης του SNMP, ενώ στις συσκευές που θέλουμε να διαχειριστούμε υπάρχει εγκατεστημένο το λογισμικό του agent. Η επικοινωνία μεταξύ των συσκευών γίνεται μέσω του πρωτοκόλλου SNMP.

Το SNMP λειτουργεί στο επίπεδο εφαρμογών του TCP/IP. Τα μηνύματα στέλνονται μέσω UDP και χρησιμοποιούνται εξ' ορισμού τα UDP ports 161 για τα αιτήματα και τις απαντήσεις, και 162 ,για τα trap (παγίδες). Η αρχική έκδοση του SNMP ήταν η SNMPv1 που χρησιμοποιούσε τα βασικά αιτήματα/ απαντήσεις, οι οποίες είναι οι ακόλουθες: get-request, set-request, getnext-request, get-response και trap. Το SNMPv1 περιγράφεται από το RFC 1157. Από το NMS στέλνονται τα αιτήματα και οι agent επιστρέφουν τις απαντήσεις. Όπως έχει ήδη αναφερθεί τα βασικά ερωτήματα είναι τα get, set, getnext, get-response και trap. Το πακέτο SNMPv1 αποτελείται από δύο μέρη, την κεφαλίδα (header) και το SNMP PDU.

Η κεφαλίδα περιέχει δύο πεδία: τον αριθμό έκδοσης και το community name. Ο αριθμός έκδοσης περιγράφει την έκδοση του SNMP που χρησιμοποιεί το πακέτο. Το community name χρησιμοποιείται για την ασφάλεια των συσκευών και το SNMP το χρησιμοποιεί σαν "κωδικό" ώστε να έχει πρόσβαση ο διαχειριστής στη συσκευή. Συνήθως, χρησιμοποιείται σε περιοχές διαχείρισης στις οποίες όλες οι συσκευές έχουν το ίδιο community name, όπως για παράδειγμα τα router και switches σε μια εταιρία όπου εξ' ορισμού το community name είναι public. Σε περίπτωση λανθασμένου community name δεν είναι δυνατό να πάρουμε πληροφορίες από τη συσκευή, γι' αυτό λέγεται ότι είναι σαν μια αδύναμη μορφή αυθεντικοποίησης. Το SNMP PDU περιλαμβάνει πέντε πεδία από τα οποία τα τέσσερα είναι σταθερού μήκους 32 bit και το πέμπτο μεταβλητού μεγέθους.

Το πρώτο πεδίο είναι το PDU type (pdu τύπος), το οποίο είναι ένας ακέραιος αριθμός που μας δείχνει τον τύπο του SNMP μηνύματος. Οι τιμές που παίρνει είναι από μηδέν έως τρία.

- 0 - GetRequest-PDU χρησιμοποιείται μόνο από τον NMS για τη συλλογή ενός ή περισσότερων στιγμιότυπων από έναν agent. Σε περίπτωση που δεν υπάρχει το συγκεκριμένο στιγμιότυπο που ζητάμε στον agent ή δε μπορεί να πάρει απάντηση, τότε δεν παρέχει καμία πληροφορία οπότε και επιστρέφει κενό μήνυμα.
- 1 – GetNextRequest χρησιμοποιείται από το NMS στον agent για να επιστρέψει το αμέσως επόμενο στιγμιότυπο από το OID που περιέχει το πακέτο. Δηλαδή, στέλνοντας ένα GetNextRequest είναι σα να στέλνουμε πολλά GetRequest, καθένα από το οποίο ζητάει την αμέσως επόμενη τιμή του προηγούμενου αιτήματος, πράγμα που συνεχίζεται μέχρι την τελευταία εγγραφή

του πίνακα. Μόλις τελειώσει ο πίνακας στέλνεται ένα μήνυμα που υποδηλώνει ότι ο πίνακας έχει ολοκληρωθεί και δε ζητάει πλέον πληροφορίες.

- 2 – GetResponse-PDU το μήνυμα αυτό δημιουργείται στον agent και είναι η απάντηση στα μηνύματα get που στέλνονται από τον agent στον manager.

- 3 – SetRequest αποστέλλεται από τον manager στον agent για να θέσουμε τιμές σε στιγμιότυπα αντικειμένων.

Το επόμενο πεδίο είναι το Request Identifier το οποίο είναι ένας αριθμός που δημιουργεί ο manager όταν πρόκειται να στείλει ένα αίτημα και τον αντιγράφει ο agent στην απάντηση. Το error status είναι ένας ακέραιος αριθμός που χρησιμοποιείται επίσης από τον agent στην απάντηση για να ενημερώσει το manager για το αποτέλεσμα του αιτήματος. Παίρνει τις ακόλουθες έξι τιμές.

- 0 – noError υποδεικνύει ότι δε συνέβη κανένα σφάλμα. Επίσης χρησιμοποιείται σε όλα τα αιτήματα γιατί γίνεται να αναφερθεί κάτι λάθος σε αυτά.

- 1 – tooBig υποδεικνύει ότι το μέγεθος της απάντησης είναι πολύ μεγάλο για να μεταφερθεί.

- 2 – noSuchName το όνομα του αντικειμένου που ζητήθηκε δεν υπάρχει.

- 3 – badValue η τιμή στην αίτηση δεν είναι ίδιου τύπου με αυτή που είχε ζητήσει ο παραλήπτης.

- 4 – readOnly υποδεικνύει ότι στάλθηκε ένα setRequest αίτημα σε μια μεταβλητή που είναι μόνο για ανάγνωση (read only).

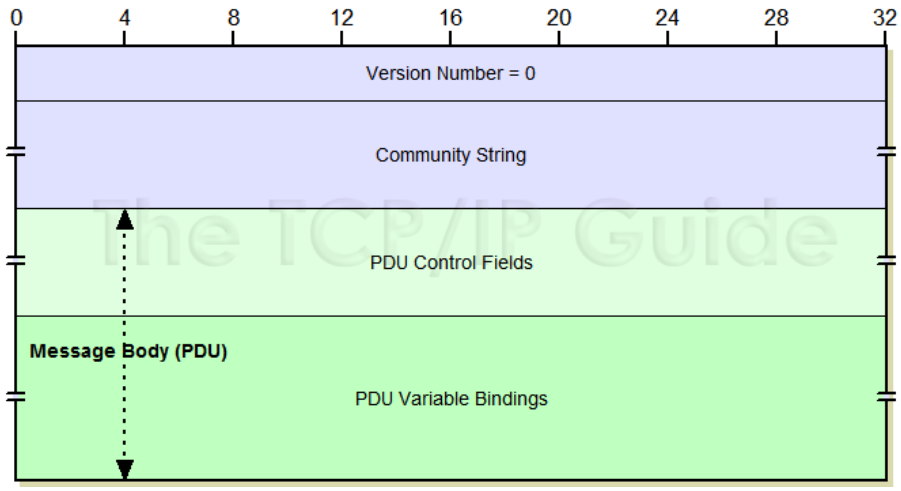
- 5 – genErr υποδεικνύει ότι έγινε κάποιο διαφορετικό σφάλμα από τα προαναφερθέντα.

Error Index. Όταν το πεδίο error status (κατάσταση λάθους) δεν είναι μηδέν, συνεπάγεται ότι το πεδίο αυτό περιέχει ένα δείκτη που μας λέει ποιο αντικείμενο δημιούργησε το λάθος.

Variable Bindings περιέχει ζευγάρια ονόματος και τιμής. Πιο συγκεκριμένα, το όνομα αντιπροσωπεύει το όνομα του αντικειμένου που θέλουμε (MIB Object) και η τιμή αντιπροσωπεύει τη ζητούμενη πληροφορία. Τα ζευγάρια αυτά υπάρχουν μόνο σε απαντήσεις ενώ στα set και τα get περιέχεται απλώς το όνομα που ζητάμε.

3.1 SNMP Version 1 (SNMPv1) Common PDU Format

Η SNMP Version 1 χρησιμοποιήθηκε για αρκετά χρόνια. Σταδιακά παρατηρήθηκαν κάποια προβλήματα και κάποιοι τομείς απαιτούσαν βελτίωση. Αυτό οδήγησε στην ανάπτυξη του SNMP Version 2 το οποίο βελτίωσε το SNMPv1 σε πολλούς τομείς όπως για παράδειγμα, στους καθορισμούς των αντικειμένων του MIB, στις λειτουργίες πρωτοκόλλου και στην ασφάλεια. Η αρχική έκδοση του SNMPv2 έγινε το 1993 και παρείχε κάποιες βελτιώσεις σε σχέση με την πρώτη έκδοση του SNMP. Οι βασικές λειτουργίες παρέμειναν ίδιες (get και set) και απλά προστέθηκαν άλλες δυο, (οι οποίες είναι) η getBulk και η Inform.



Μετά την έκδοση του SNMPv1 παρατηρήθηκε ότι υπήρχε μεγάλο πρόβλημα στον τομέα της ασφάλειας. Για το λόγο αυτό αμέσως μετά την έκδοση του ξεκίνησε η ανάπτυξη μιας νέας έκδοσης η οποία θα διόρθωνε το παραπάνω πρόβλημα. Αρχικά, ανέκυψαν διαφωνίες ως προς τον τρόπο αντιμετώπισης της μειωμένης ασφάλειας, που χαρακτήριζε τη νέα έκδοση του SNMP. Έτσι, το 1992 εκδόθηκε το SNMPsec το οποίο χρησιμοποιούσε ένα νέο μηχανισμό ασφάλειας. Αυτός ο μηχανισμός παρείχε μεγαλύτερη ασφάλεια από την πρώτη έκδοση του SNMP αλλά δεν έτυχε ποτέ ευρείας αποδοχής και πλέον δε χρησιμοποιείται. Αν και το SNMPsec δε χρησιμοποιήθηκε, η ιδέα του μηχανισμού ασφάλειας που χρησιμοποιούσε (party based security) έγινε η βάση για την ανάπτυξη της πλήρης έκδοσης του νέου SNMP κι έτσι τον Απρίλιο του 1993 κυκλοφόρησε για πρώτη φορά η δεύτερη έκδοση του πρωτοκόλλου. Δυστυχώς όμως, και αυτή η έκδοση δεν έγινε αποδεκτή παγκοσμίως. Συνεπώς, κάποιες επιτροπές ξεκίνησαν την ανάπτυξη διαφορετικών εκδοχών του SNMPv2.

Η βασική έκδοση είναι η SNMPv2p όπου με το p να αναφέρεται στο party based security το οποίο θα αναλυθεί παρακάτω. Έχουν κυκλοφορήσει τρεις βασικές εκδοχές οι οποίες είναι:

1. Η SNMPv1.5, που προσπάθησε να λύσει κάποια προβλήματα το SNMPv2p, ήταν ο πρόγονος του SNMPv2c και χρησιμοποιεί τη μέθοδο των community strings όπως και η v1.

2. Η SNMPv2c η οποία χρησιμοποιεί τα community strings αντί για το party based security.

3. Η SNMPv2u (user based), η οποία χρησιμοποιεί τους χρήστες αντί για τα community strings. Θεωρείται πιο απλή από την party based αλλά και πιο ασφαλής από την community strings security. Καθορίζεται από τα RFC1909 και RFC1910 και βρίσκεται επίσημα σε πειραματικό στάδιο.

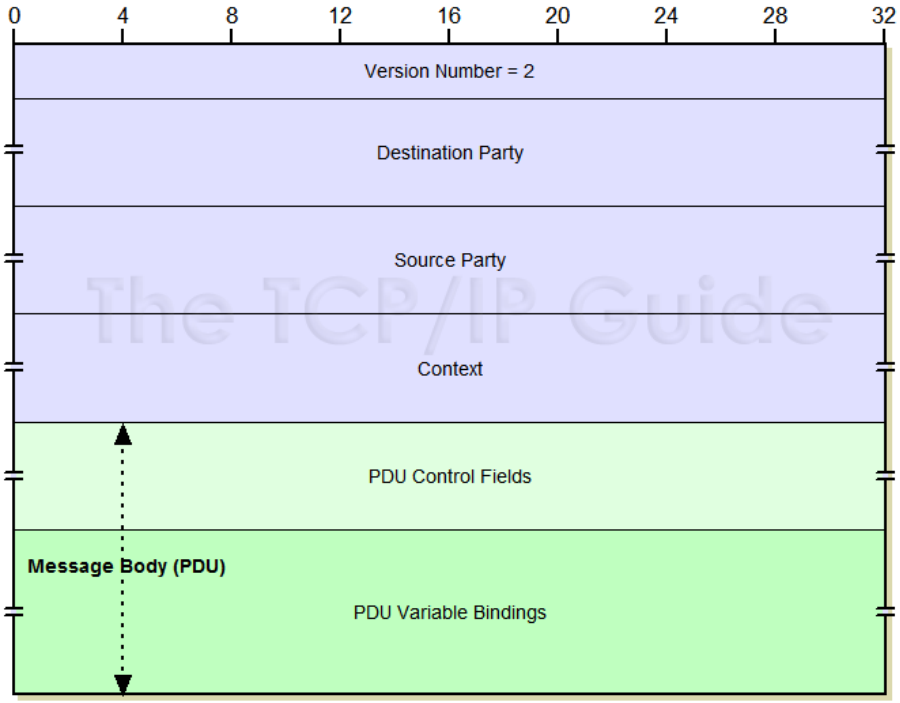
Λόγω αυτών των διαφορετικών εκδόσεων δεν υπάρχει μόνο ένα PDU της version 2. Παρακάτω θα γίνει η ανάλυση του μηνύματος των τριών βασικών εκδόσεων του SNMP της SNMPv2p, SNMPv2c και της SNMPv2u.

3.2 SNMPv2p

Το πρώτο πεδίο είναι ο αριθμός έκδοσης (version). Μεγέθους 32bit και χρησιμοποιείται για να διασφαλιστεί η συμβατότητα μεταξύ των εκδόσεων. Για το SNMPv2p η τιμή είναι 2.

Το επόμενο πεδίο είναι το Destination Party το οποίο χρησιμοποιείται από τον party based μηχανισμό ασφάλειας προκειμένου να βρεθεί ο προορισμός του μηνύματος. Στη συνέχεια υπάρχει το Source Party πού δείχνει τον αποστολέα του μηνύματος.

Ακολουθεί το Context, το οποίο καθορίζει το πλήθος των MIB αντικειμένων που μπορούν να έχουν πρόσβαση σε μια συσκευή. Αμέσως επόμενο είναι το PDU, το οποίο θα αναλυθεί παρακάτω γιατί είναι κοινό για όλες τις εκδόσεις. Να σημειωθεί ότι όλα τα πεδία είναι μεταβλητού μεγέθους εκτός από τον αριθμό έκδοσης.



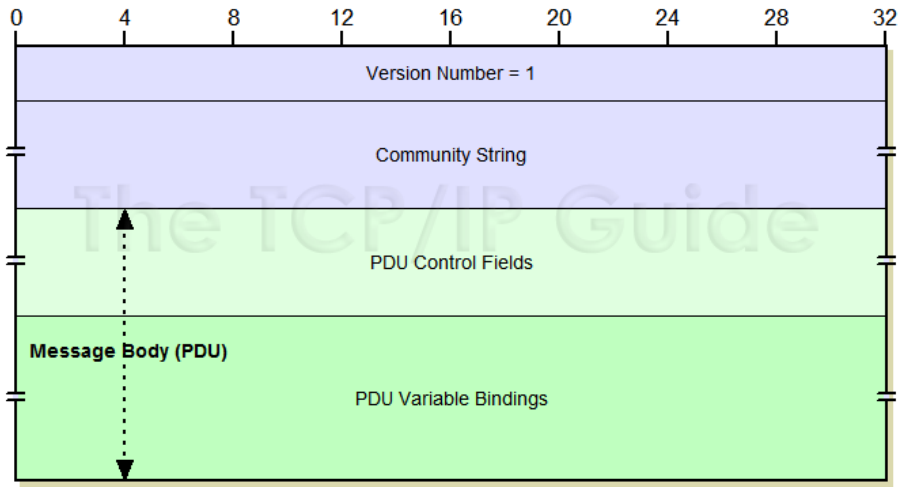
3.3 SNMPv2c

Το SNMPv2c περιλαμβάνει τρία πεδία.

Το πρώτο πεδίο είναι ο αριθμός έκδοσης (version) το οποίο είναι μεγέθους 32bit και χρησιμοποιείται για να διασφαλιστεί η συμβατότητα μεταξύ των εκδόσεων. Η μόνη διαφορά είναι ότι στην έκδοση SNMPv2c η τιμή είναι 1.

Το επόμενο πεδίο είναι το Community String το οποίο αναγνωρίζει σε ποια περιοχή (community) βρίσκονται ο αποστολέας και ο παραλήπτης του μηνύματος.

Και το τελευταίο πεδίο είναι το PDU το οποίο θεωρείται ως το σώμα του μηνύματος. Όπως και στην προηγούμενη έκδοση μόνο ο αριθμός έκδοσης είναι σταθερού μήκους ενώ τα υπόλοιπα πεδία είναι μεταβλητού μήκους.



3.4 SNMPv2u

Το SNMPv2u μήνυμα περιέχει τρία βασικά πεδία.

Το πρώτο πεδίο είναι ο αριθμός έκδοσης (version) που είναι μεγέθους 32bit και χρησιμοποιείται για να διασφαλιστεί η συμβατότητα μεταξύ των εκδόσεων. Για το SNMPv2c η τιμή είναι 2 η οποία όπως παρατηρούμε είναι ίδια με την έκδοση SNMPv2p.

Το δεύτερο πεδίο είναι οι παράμετροι οι οποίοι χρησιμοποιούνται για την υλοποίηση του user based μοντέλου επικοινωνίας. Οι παράμετροι που χρησιμοποιούνται είναι οι εξής:

1. Model με μέγεθος 1byte. Όταν βρίσκεται στην τιμή ένα υποδεικνύει ότι χρησιμοποιείται το user based μοντέλο.

2. Qos (quality of service) με μέγεθος 1byte. Υποδεικνύει αν χρησιμοποιείται πιστοποίηση και κωδικοποίηση και αν επιτρέπεται η δημιουργία απάντησης στο συγκεκριμένο αίτημα.

3. AgentID με μέγεθος 12bytes. Χρησιμοποιείται για την αναγνώριση του agent που στέλνει το μήνυμα. Είναι χρήσιμο για την αντιμετώπιση συγκεκριμένων επιθέσεων ασφαλείας.

4. Agent Boots με μέγεθος 4bytes. Είναι ο αριθμός που φορτώθηκε ή ξαναφορτώθηκε ο agent από τη στιγμή που τέθηκε το agent id του και χρησιμοποιείται για την αντιμετώπιση επιθέσεων ασφαλείας.

5. Agent Time με μέγεθος 4bytes. Είναι ο αριθμός των δευτερολέπτων από την τελευταία εκκίνηση του agent. Χρησιμοποιείται επίσης, για την αντιμετώπιση προβλημάτων ασφαλείας.

6. Max Size με μέγεθος 2bytes. Είναι το μέγεθος του μηνύματος που ο αποστολέας μπορεί να δεχτεί.

7. User Length με μέγεθος 1byte. Είναι το μήκος του ονόματος χρήστη (user name).

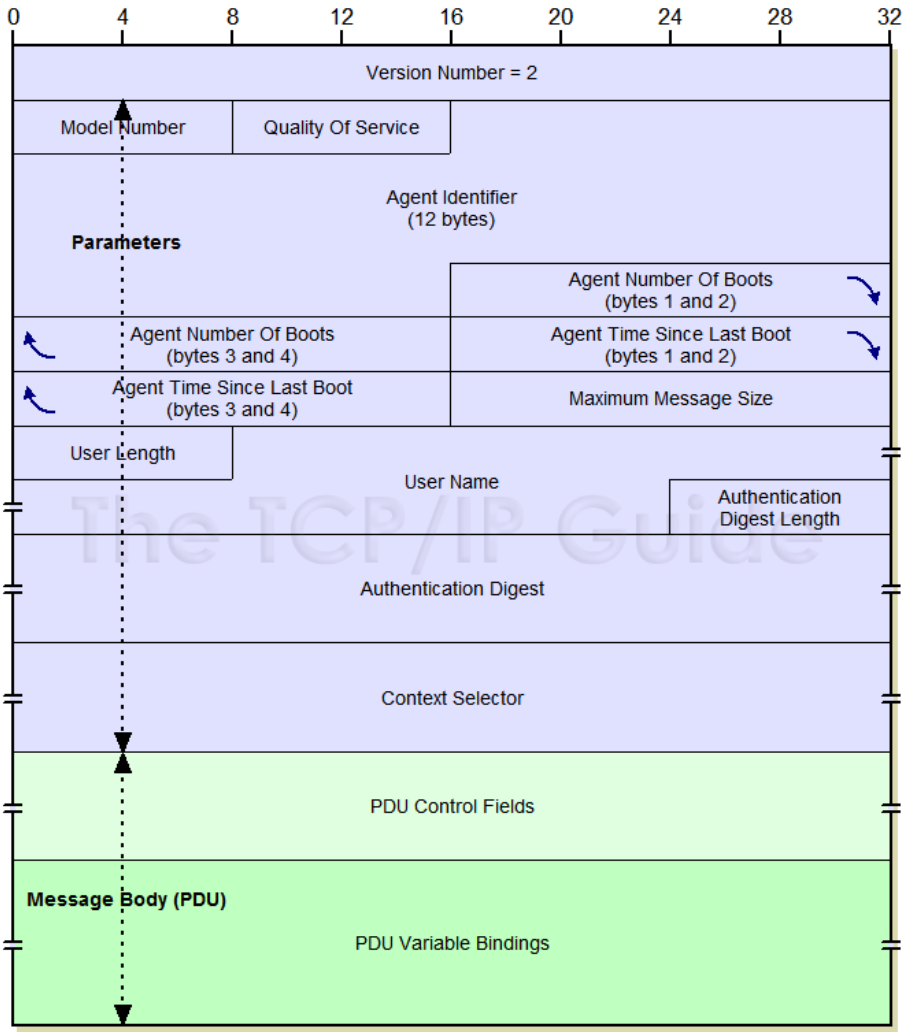
8. User Name με μέγεθος από 1 εώς 16 bytes. Είναι το όνομα του χρήστη από τον οποίο αποστέλλεται το μήνυμα.

9. Authentication Length με μέγεθος 1 byte. Μας δείχνει το μήκος του πεδίου authentication digest.

10. Authentication Digest με μήκος από 0 εώς 255 bytes. Είναι ένας αριθμός πιστοποίησης για την εξακρίβωση της ταυτότητας και της γνησιότητας του μηνύματος. Έχει τη τιμή μηδέν όταν δε χρησιμοποιείται αυθεντικοποίηση (πιστοποίηση).

11. Context Selector με μέγεθος 0 έως 40 bytes. Είναι ένα string το οποίο συνδυάζεται με το agent id για να καθορίσει ένα συγκεκριμένο κομμάτι που περιέχει τις πληροφορίες διαχείρισης που περιέχονται σε αυτό το μήνυμα.

Και μετά ακολουθεί το PDU του SNMPv2u το οποίο μπορεί να είναι είτε κρυπτογραφημένο είτε όχι.



Στη συνέχεια θα περιγράψουμε τη μορφή του SNMPv2 PDU. Όπως ήδη αναφέραμε, το PDU είναι το ίδιο για όλες τις εκδόσεις του SNMP εκτός από το GetBulkRequest μήνυμα. PDU Type έχει μέγεθος 4byte και περιγράφει τον τύπο του PDU. Οι τιμές που μπορεί να πάρει είναι οι εξής:

PDU TYPE VALUE	PDU TYPE
0	GetRequest
1	GetNextRequest
2	Response
3	SetRequest
4	δεν χρησιμοποιείται πλέον (ήταν το trap PDU στην έκδοση 1)
5	GetBulkRequest
6	InformRequest
7	Trapv2
8	Report

Το επόμενο πεδίο είναι το Request Id, το οποίο έχει μέγεθος 4bytes και είναι ένας αριθμός που χρησιμοποιείται για την αντιστοίχιση των αιτημάτων με τις απαντήσεις. Η συσκευή που στέλνει το αίτημα θέτει το Request Id και η συσκευή που στέλνει την απάντηση το αντιγράφει σε αυτό το πεδίο.

Στη συνέχεια, υπάρχει το Error Status, το οποίο είναι ένας ακέραιος αριθμός που στέλνεται από τη συσκευή που απαντάει για να μας δηλώσει το αποτέλεσμα του αιτήματος. Αν έχει τιμή μηδέν δηλώνει ότι δε συνέβη κανένα σφάλμα. Οι πρώτες έξι τιμές είναι ίδιες με την έκδοση 1 ενώ έχουν προστεθεί πολλές καινούριες τιμές. Όλες οι τιμές των σφαλμάτων δίνονται στον παρακάτω πίνακα.

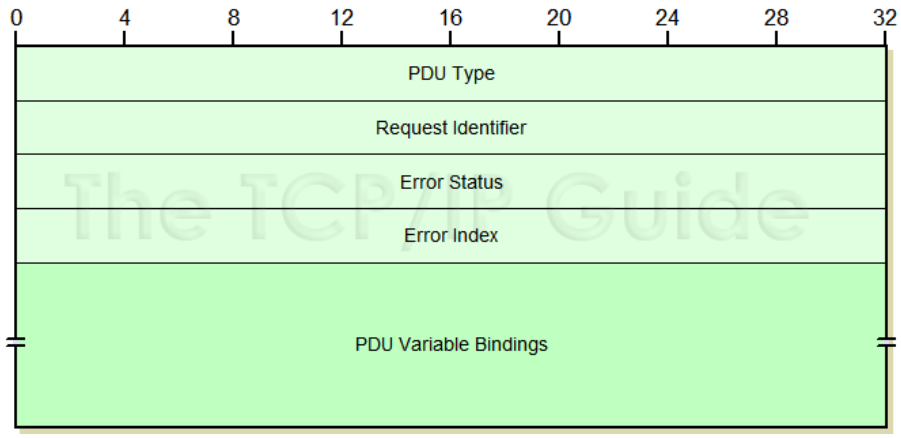
Error Status Value	Error Code	Description
0	noError	No error occurred. This code is also used in all request PDUs, since they have no error status to report.
1	tooBig	The size of the Response-PDU would be too large to transport
2	noSuchName	The name of a requested object was not found.
3	badValue	A value in the request didn't match the structure that the recipient of the request had for the object. For example, an object in the request was specified with an incorrect length or type.
4	readOnly	An attempt was made to set a variable that has an Access value indicating that it is read-only.
5	genErr	An error occurred other than one indicated by a more specific error code in this table.
6	noAccess	Access was denied to the object for security reasons.
7	wrongType	The object type in a variable binding is incorrect for the object
8	wrongLength	A variable binding specifies a length incorrect for the object.
9	wrongEncoding	A variable binding specifies an encoding incorrect for the object.
10	wrongValue	The value given in a variable binding is not possible for the object.
11	noCreation	A specified variable does not exist and cannot be created.
12	inconsistentValue	A variable binding specifies a value that could be held by the variable but cannot be assigned to it at this time.
13	resourceUnavailable	An attempt to set a variable required a resource that is not available.
14	commitFailed	An attempt to set a particular variable failed.
15	undoFailed	An attempt to set a particular variable as part of a group of variables failed, and the attempt to then undo the setting of other variables was not successful.
16	authorizationError	A problem occurred in authorization.
17	notWritable	The variable cannot be written or created.
18	inconsistentName	The name in a variable binding specifies a variable that does not exist.

Το επόμενο πεδίο είναι το Error Index το οποίο έχει μέγεθος 4 bytes. Όταν

αυτό το πεδίο δεν είναι μηδέν περιέχει το δείκτη στο αντικείμενο που δημιούργησε το σφάλμα.

Έπειτα, έχουμε το Variable Bindings, το οποίο είναι μεταβλητού μεγέθους. Είναι ζευγάρια ονόματος τιμών τα οποία αναγνωρίζουν τα MIB αντικείμενα στο PDU. Αν είναι απάντηση και όχι αίτημα το πεδίο αυτό περιέχει τις τιμές των MIB Oobjects.

Η εικόνα του μηνύματος φαίνεται παρακάτω:



Όσον αφορά το GetBulkRequest στο οποίο δεν είχαμε αναφερθεί προηγουμένως, χρησιμοποιείται για την ανάκτηση μεγάλου όγκου πληροφοριών και κυρίως για μεγάλους πίνακες. Συγκεκριμένα, εκτελεί μια συνεχόμενη GetNext λειτουργία με βάση την τιμή του πεδίου max repetitions.

3.5 SNMPv3

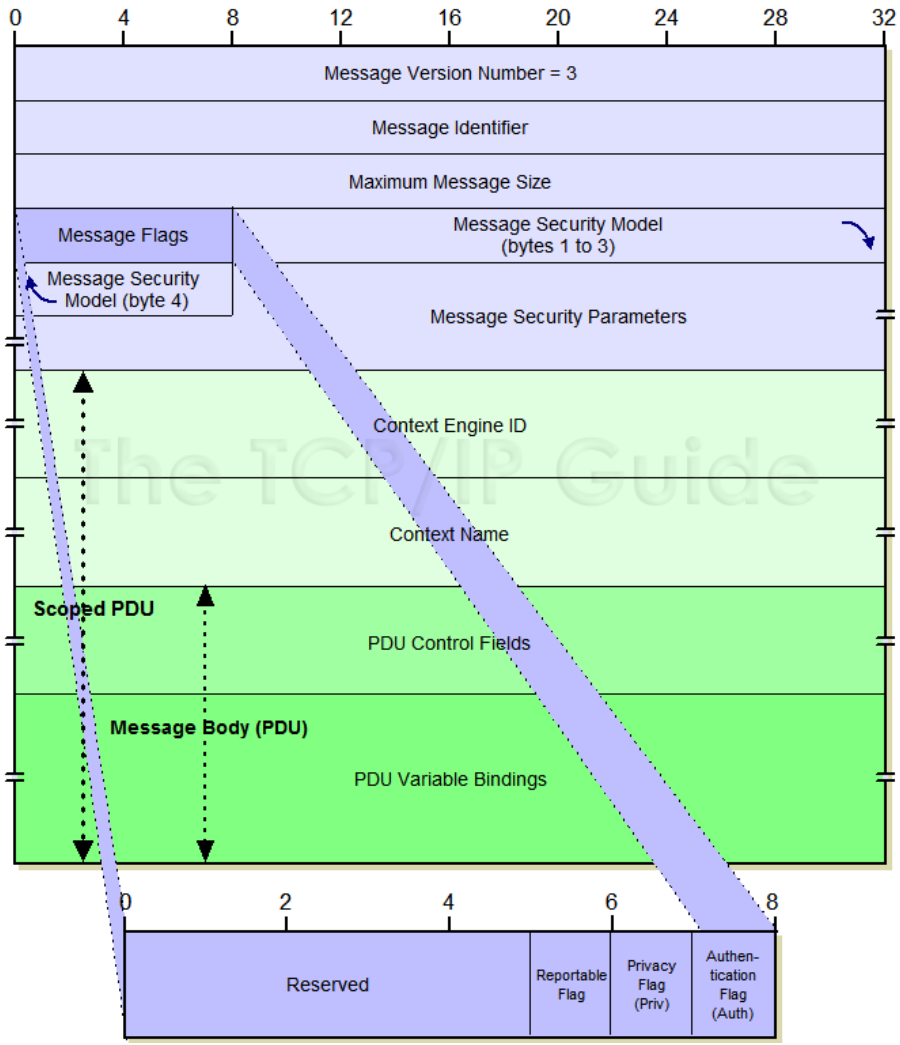
Όπως ήδη αναφέραμε, η έκδοση SNMPv2c είναι η πιο διαδεδομένη. Γνωρίζουμε ότι η SNMPv2 εκδόθηκε για να καλύψει τα κενά στον τομέα της ασφάλειας. Παρά τις βελτιώσεις που έγιναν σε σχέση με την πρώτη έκδοση, και πάλι υπήρχαν προβλήματα στο συγκεκριμένο τομέα. Γι' αυτό το 2004 εκδόθηκε το SNMPv3. Η SNMPv3 παρέχει τρία σημαντικά στοιχεία όσον αφορά την ασφάλεια.

- Ακεραιότητα μηνύματος (message integrity). Πιστοποιεί ότι το μήνυμα δε θα αλλοιωθεί κατά τη διάρκεια της όλης διαδικασίας.
- Αυθεντικοποιήση. Παρέχει την πιστοποίηση ότι το μήνυμα προέρχεται από έγκυρη πηγή.
- Κρυπτογράφηση. Κρυπτογραφεί τα μηνύματα έτσι ώστε να μην μπορούν να διαβαστούν από τρίτους.

3.5.1 ΜΟΡΦΗ SNMPv3

Η βασική μορφή είναι όμοια με τις προηγούμενες εκδόσεις δηλαδή έχουμε μια κεφαλίδα και ένα ενθυλακωμένο PDU. Τα πεδία περιγράφονται παρακάτω.

<table>
<tr><th>Field Name</th><th>Description</th></tr>
<tr><td>Msg Version</td><td>Message Version έχει μέγεθος 4byte είναι ο αριθμός έκδοσης του SNMP και για την έκδοση τρία είναι το νούμερο τρία.</td></tr>
<tr><td>Msg ID</td><td>Message Identifier έχει μέγεθος 4byte και έχει σχεδόν την ίδια χρήση όπως τις προηγούμενες εκδόσεις αλλά δεν είναι όμοιες. Το πεδίο αυτό δημιουργήθηκε για να επιτρέψει την αντιστοίχηση στο στάδιο επεξεργασίας του μηνύματος ξεχωριστά από τα περιεχόμενα του PDU για την προστασία ενάντια σε επιθέσεις ασφαλείας.</td></tr>
<tr><td>Msg Max Size</td><td>MSG max size έχει μέγεθος 4 byte και είναι το μέγιστο μέγεθος μηνύματος που μπορεί να παραλάβει ο αποστολέας. Η ελάχιστη τιμή που μπορεί να πάρει είναι 484 bit.</td></tr>
<tr><td>Msg Flags</td><td>MSG flags έχει μέγεθος 1 byte και είναι ένα πλήθος σημαιών(flags) για τον έλεγχο της επεξεργασίας του μηνύματος. Η δομή του byte αυτού είναι η εξής:
<table>
<tr><th>Subfield name</th><th>Size</th><th>Description</th></tr>
<tr><td>Reserved</td><td>5 bit</td><td>Είναι δεσμευμένο για μελλοντική χρήση.</td></tr>
<tr><td>Reportable Flag</td><td>1 bit</td><td>Όταν έχει τιμή 1, η συσκευή που λαμβάνει αυτό το μήνυμα πρέπει να στείλει πίσω ένα report PDU κάτω από οποιεσδήποτε συνθήκες.</td></tr>
<tr><td>Privacy Flag</td><td>1 bit</td><td>Όταν έχει τιμή 1, υποδηλώνει ότι το μήνυμα είναι κρυπτογραφημένο. Το bit αυτό είναι ενεργό μόνο όταν και το authentication flag έχει τιμή 1.</td></tr>
<tr><td>Authentication Flag</td><td>1 bit</td><td>Όταν έχει τιμή 1, δηλώνει ότι χρησιμοποιήθηκε αυθεντικοποίηση για την προστασία του μηνύματος.</td></tr>
</table></td></tr>
<tr><td>Msg Security Model</td><td>MSG Security Model το οποίο έχει μέγεθος 4byte. Δηλώνει ποιο μοντέλο ασφαλείας χρησιμοποιείται για το μήνυμα.</td></tr>
<tr><td>Msg Security Parameters</td><td>MSG Security Parameters έχει μεταβλητό μέγεθος και περιέχει παραμέτρους που χρησιμοποιούνται για την υλοποίηση του μοντέλου ασφαλείας που χρησιμοποιείται στο μήνυμα.</td></tr>
<tr><td>Scoped PDU</td><td>Scoped PDU έχει και αυτό μεταβλητό μέγεθος και περιέχει το PDU που πρόκειται να αποσταλεί μαζί με κάποιες παραμέτρους που αναγνωρίζουν ένα SNMP περιεχόμενο το οποίο περιγράφει ένα πλήθος διαχειρίσιμων πληροφοριών, προσβάσιμων από μια συγκεκριμένη οντότητα (είναι θέματα ασφάλειας και δε θα δοθεί μεγαλύτερη έκταση στο ζήτημα) .</td></tr>
</table>


0
4
8
12
16
20
24
28
32
Message Version Number = 3
Message Identifier
Maximum Message Size
Message Flags
Message Security Model (bytes 1 to 3)
Message Security Model (byte 4)
Message Security Parameters
Context Engine ID
Context Name
Scoped PDU
PDU Control Fields
Message Body (PDU)
PDU Variable Bindings
0
2
4
6
8
Reserved
Reportable Flag
Privacy Flag (Priv)
Authen-tication Flag (Auth)

4 Διαδικασία υλοποίησης πτυχιακής

Αφού έγινε η εγκατάσταση όλων των απαραίτητων προγραμμάτων, ξεκινήσαμε να πειραματιζόμαστε με τις εντολές του SNMP σε περιβάλλον Linux. Πειραματιστήκαμε με διάφορους τρόπους ώστε να εξοικειωθούμε με τις εντολές και τις λειτουργίες του SNMP.

Αρχικά, έγινε προσπάθεια να πάρουμε πληροφορίες για το ADSL Router του σπιτιού μας. Ήταν επιτυχής η απόκτηση πληροφορίων που αφορούν τα interface του, καθώς και γενικών πληροφοριών για το ίδιο το Router.

Σε δεύτερη φάση, δημιουργήθηκε ένα τοπικό δίκτυο τριών υπολογιστών και ενός ADSL Router, ώστε να μπορούμε να αποκτήσουμε πληροφορίες από ολόκληρο το δίκτυο ή ξεχωριστά από κάθε υπολογιστή. Καταφέραμε και εγκαταστήσαμε το NET-SNMP για περιβάλλον MS Windows με αποτέλεσμα να είναι δυνατή η ανταλλαγή των πληροφοριών που χρειαζόμασταν. Ως αποτέλεσμα αυτού, συλλέξαμε πληροφορίες από περιβάλλον Linux, MS Windows XP και MS Windows Vista.

Μετά από τους διάφορους πειραματισμούς, ξεκινήσαμε με τη γρήγορη εκμάθηση της PHP και τη δημιουργία απλών δυναμικών ιστοσελίδων, με αρχικό σκοπό την εξοικείωση με αυτήν τη γλώσσα. Μετά την εξοικείωση με την PHP, ξεκινήσαμε την ανάπτυξη του κώδικά μας με τη χρήση της PHP και χρησιμοποιήσαμε το πακέτο NET SNMP για την εφαρμογή του SNMP. Η ιστοσελίδα στην αρχή ήταν πολύ απλή και χωρίς πολλές λειτουργίες. Τελικά, καταφέραμε να παίρνουμε πληροφορίες από τα τοπικά μας συστήματα. Οι πληροφορίες αυτές αφορούσαν το σύστημα και όχι το δίκτυο. Η πρώτη μας δυσκολία ήταν η χρήση της IP το οποίο αναλύεται παρακάτω.

IP

Αρχικά, χρησιμοποιήσαμε τέσσερις διαφορετικές μεταβλητές ώστε να αποθηκεύουμε την κάθε οκτάδα και να τη συγκρίνουμε ξεχωριστά. Έτσι, καταλήξαμε στο συμπέρασμα ότι έπρεπε να βρεθεί ένας τρόπος που να μας επιτρέπει την αποθήκευση. Όμως, αυτό χρειαζόταν τη χρήση μιας βάσης δεδομένων, πράγμα που δεν ήταν καθόλου χρήσιμο για την εφαρμογή μας, γιατί δε θα χρειαζόταν αυτή η βάση δεδομένων κάπου αλλού. Όπως είναι λογικό η δημιουργία μιας βάσης δεδομένων για την αποθήκευση μόνο μιας πληροφορίας σημαίνει αυτομάτως ότι δεν ήταν η καλύτερη λύση για το πρόβλημα μας. Οπότε και δεν προχωρήσαμε στη χρησιμοποίηση αυτής και οδηγηθήκαμε σε διαφορετική.

Λόγω του ότι θέλαμε να χρησιμοποιούμε μία IP, και με αυτή να παίρνουμε πληροφορίες πολλές φορές και για διαφορετικά πράγματα, δε μπορούσαμε να βρούμε έναν τρόπο ώστε να «αποθηκεύουμε» την IP για την επαναχρησιμοποίησή της. Η λύση αυτή δόθηκε με τη χρήση της IP στο browser, με τη χρήση της εντολής GET της PHP.

```
$ip=$_GET["ip"];
$community=$_GET["cn"];
$ch=$_GET["ch"];
list($ip1, $ip2, $ip3, $ip4) = explode(".",$ip);
```

Και τη χρήση της φόρμας HTML:

```
<form action="index.php" method="GET">
 <table align='center' width="30%" border="0">
        <tr>
            <td>
                Enter the IP address:
            </td>
            <td>
            <input type="text" name="ip" size="15"
                        maxlength="15" value="">
           </td>
       </tr>
<input type="submit" name="submit" value="Submit">
```

Έτσι, όταν δίναμε την IP στη φόρμα και πατούσαμε το κουμπί Submit, αυτομάτως το URL της σελίδας μας είχε την παρακάτω μορφή (Στο συγκεκριμένο παράδειγμα έχουμε δώσει την IP 195.251.123.1).

http://aetos.it.teithe.gr/~dsiaper/myapp/index.php?ip=195.251.123.1

Βέβαια στη συνέχεια, προστέθηκαν και άλλες μεταβλητές με ένα τελικό αποτέλεσμα να εμφανίζεται ως εξής:

http://aetos.it.teithe.gr/~dsiaper/myapp/index.php?ip=195.251.123.1&cn=public&ch=1

Με τον τρόπο αυτό, καταφέραμε να χρησιμοποιούμε την IP του εκάστοτε συστήματος όποτε τη χρειαζόμασταν και επιπλέον, να πραγματοποιούμε ελέγχους εγκυρότητας στην IP. Όπως ήταν αναμενόμενο, αμέσως μετά από αυτό και με τον ίδιο τρόπο, έγινε διαθέσιμο και το community name στον browser. Αφού έγινε δυνατή η λήψη της IP και του community name, ξεκινήσαμε τον έλεγχο της εγκυρότητας.

Για τον έλεγχο της IP χρησιμοποιήθηκε η κλάση Net_CheckIP() που πήραμε από το site

http://pear.php.net/package/Net_CheckIP2/docs/latest/Net_CheckIP2/_Net_CheckIP2-1.0.0RC2---examples---check-ip.php.html .

Στη συνέχεια, έγιναν οι απαραίτητες ρυθμίσεις ώστε να μπορεί να χρησιμοποιηθεί στην ιστοσελίδα μας ο συγκεκριμένος έλεγχος. Η διαδικασία που χρησιμοποιεί η κλάση αυτή δίνεται παρακάτω:

```
function check_ip($ip)
  {
      $oct = explode('.', $ip);
      if (count($oct) != 4) {
          return false;
      }
      for ($i = 0; $i < 4; $i++) {
          if (!preg_match("/^[0-9]+$/", $oct[$i])) {
              return false;
          }
          if ($oct[$i] < 0 || $oct[$i] > 255) {
              return false;
```

```
            }
        }
        return true;
    }
```

Η συνάρτηση αυτή παίρνει ως όρισμα την IP που θέλουμε να ελέγξουμε και τη χωρίζει σε 4 κομμάτια, που όπως είναι γνωστό είναι τα byte τις IP. Αν τα μέρη στα οποία χωρίσαμε την IP είναι 4, τότε προχωράμε στους επόμενους δύο ελέγχους εγκυρότητας της IP. Για κάθε κομμάτι (Byte) ελέγχει αν είναι αριθμός, και τέλος, ένας τρίτος έλεγχος δείχνει αν βρίσκεται μεταξύ των ορίων που καθορίζει η μορφή της IP (0-255).

Η κλήση της συνάρτησης αυτής γίνεται από την αρχική σελίδα index.php μέσω της εντολής:

```
if (Net_CheckIP::check_ip($host) &&
    DevExist($host,$community))
```

Μέσα στον έλεγχο της εντολής if, στην οποία βρίσκεται η κλήση της συνάρτησης γίνεται και η κλήση της συνάρτησης DevExists() η οποία ελέγχει αν η IP που δώσαμε ανταποκρίνεται σε κάποια συσκευή. Αυτό γίνεται με ένα snmpget ερώτημα, το οποίο στέλνουμε με μειωμένο χρόνο απόκρισης ώστε να μην υπάρχει μεγάλη καθυστέρηση.

Στο ερώτημα αυτό ζητείται να επιστραφεί το όνομα της συσκευής, στοιχείο που είναι πάντα διαθέσιμο από όλες τις συσκευές που έχουν SNMP agent.

Με τον τρόπο που περιγράψαμε παραπάνω, εξασφαλίζεται η εγκυρότητα της IP, καθώς και η ύπαρξη της συσκευής που αντιστοιχεί η IP που δώσαμε. Σε περίπτωση λανθασμένης IP ή ανύπαρκτης συσκευής, έχει προστεθεί ένα παράθυρο, όπου εμφανίζει μήνυμα λάθους.

Όσον αφορά το community name, δε γίνεται κάποιος έλεγχος, παρά μόνο το αν υπάρχει ή όχι. Το community name, όπως περιγράφηκε σε προηγούμενο κεφάλαιο, είναι ένας «κωδικός», οπότε και μπορεί να είναι οποιοσδήποτε, χωρίς περιορισμούς και έτσι δεν πραγματοποιείται κάποιος έλεγχος εγκυρότητας γι αυτό. Εξ΄ ορισμού σε πολλές περιπτώσεις το community name είναι το public.

4.1 Η ροή του προγράμματος

Η αρχική έκδοση της ιστοσελίδας περιελάμβανε ένα αρχείο PHP από το οποίο πραγματοποιούνταν όλες οι βασικές λειτουργίες. Έτσι, η ροή ξεκινούσε από την αρχή και σειριακά προχωρούσε προς το τέλος. Αυτό όμως δεν ήταν λειτουργικό διότι, δε μπορούσαμε αφενός να αλλάξουμε τη ροή και αφετέρου να διαχειριστούμε τον όγκο των πληροφοριών. Αυτό γιατί όταν ανοίγαμε την ιστοσελίδα και δίναμε μια IP και ένα community name, αυτομάτως παίρναμε όλες τις πληροφορίες, απαραίτητες ή όχι. Όπως είναι αναμενόμενο, ένα τέτοιο πρόβλημα ήταν αρκετά βασικό για τη λειτουργία της ιστοσελίδας μας και έπρεπε να λυθεί άμεσα.

Στη συνέχεια, το πρόβλημα που ανέκυψε ήταν ότι έπρεπε να διαχειριστούμε όλον αυτόν τον όγκο πληροφοριών και να τον οργανώσουμε με τρόπο κατανοητό έτσι ώστε να γίνεται εύκολος στην ανάγνωση. Αυτό είχε τεράστιο κόστος, τόσο λειτουργικό όσο και χρονικό.

Η λύση ήταν η οργάνωση της σελίδας σε μία κύρια, αρχική σελίδα (index.php) από την οποία θα αλλάζει η ροή του προγράμματος κάθε φορά που θα εκτελούμε μία νέα λειτουργία. Έτσι, μας δίνεται η δυνατότητα να επιλέξουμε τα δεδομένα

που θέλουμε να πάρουμε από κάποια συσκευή του δικτύου χωρίς απαραίτητα να χρησιμοποιήσουμε όλα τα διαθέσιμα δεδομένα.

Έτσι, η αρχική μας σελίδα οργανώθηκε σε κατηγορίες, σύμφωνα με τις οποίες μπορούμε να πάρουμε συγκεκριμένες πληροφορίες, ανάλογα με την επιλογή μας, για το ίδιο το σύστημα και τη δικτυακή του λειτουργία. Στην περίπτωση τώρα που έχουμε μια Cisco συσκευή μπορούμε να εκτελέσουμε κάποιες πρόσθετες λειτουργίες τις οποίες όμως, θα αναλύσουμε σε επόμενη ενότητα. Ο τρόπος επιλογής γίνεται με τη χρήση ενός μενού του οποίου η λειτουργία θα αναλυθεί στη συνέχεια.

Ο τρόπος που αλλάζει η ροή μας κατ' επιλογή πραγματοποιείται με τη χρήση μιας switch/case και με τη χρήση μεταβλητών ώστε κάθε φορά να ανοίγει η συγκεκριμένη ιστοσελίδα που θέλουμε.

```
switch ($ch){
        case '1':
             include_once 'files/SNMP/sysTable.php';
             $show=showTable();
             $smarty->assign('show', $show);
        break;
        case '15':
             include_once  'files/SNMP/cisco.php';
             $show=showTable();
             $smarty->assign('show', $show);
        break;
        case '16':
             include  'files/SNMP/ciscoNetwork.php';
             $show=showTable();
             $smarty->assign('show', $show);
        break;
        default :
             $show='';
        break;
}
```

Όπως βλέπουμε στον κώδικα, επιλέγοντας από το μενού την απαραίτητη κατηγορία πληροφοριών που θέλουμε να δούμε (routing table, arp table) , αυτομάτως θέτουμε την τιμή της μεταβλητής που χρησιμοποιούμε για την switch, τη μεταβλητή «ch» στην κατάλληλη τιμή. Για παράδειγμα, αν διαλέξουμε από το μενού την επιλογή Cisco, αυτομάτως η μεταβλητή ch (μεταβλητή της επιλογής) παίρνει την τιμή 15 οπότε ενεργοποιείται η δέκατη πέμπτη περίπτωση του switch στο Index.php, το οποίο έχει σαν αποτέλεσμα την εκτέλεση του αρχείου «Cisco.php» και την εμφάνιση των επιθυμητών αποτελεσμάτων. Η διαδικασία που ακολουθείται για τη λήψη της μεταβλητής ch, έχει να κάνει με τη διαρρύθμιση του μενού μας και θα εξηγηθεί στο κεφάλαιο του μενού.

```
(1) include_once 'files/SNMP/ARPTable.php';
(2) $show=showTable();
(3) $smarty->assign('show', $show);
```

Στον παραπάνω κώδικα, βλέπουμε μία από τις περιπτώσεις του switch. Στην γραμμή (1) γίνεται η συμπερίληψη του κώδικα του αρχείου «sysTable.php» στο index.php.

Στη συνέχεια, στη γραμμή (2), δημιουργούμε μία μεταβλητή με το όνομα show η οποία περιέχει το αποτέλεσμα της συνάρτησης showTable(). Η showTable() είναι μία συνάρτηση η οποία σαν σκοπό έχει την εμφάνιση των δεδομένων που θέλουμε. Αυτό γίνεται ως εξής:

Κάθε αρχείο PHP που μας δίνει τις πληροφορίες του SNMP, είναι στην ουσία η συνάρτηση showTable(). Έτσι, καλώντας την από το index.php ξεκινάει η λήψη των δεδομένων καθώς και η επεξεργασία και εμφάνισή τους. Αμέσως μετά τη λήψη των δεδομένων, τα προωθούμε στο Smarty ώστε να τα λάβει το ARPTable.tpl. Τέλος, καλούμε το αρχείο ARP.tpl, το οποίο είναι υπεύθυνο για την εμφάνιση των δεδομένων, και το επιστρέφουμε ολόκληρο στο index.php.

Στην γραμμή (3) παίρνουμε το περιεχόμενο της μεταβλητής show, το οποίο αφορά τα αποτελέσματα της showTable() και το αποθηκεύουμε στην Smarty μεταβλητή show μέσω του assign. Τονίζεται ότι το 'show' και το $show είναι διαφορετικά πράγματα, με το πρώτο να είναι μεταβλητή του Smarty και το δεύτερο μεταβλητή της PHP. Με τον τρόπο αυτόν έχουμε στο index.php τα δεδομένα μας έτοιμα προς εμφάνιση. Το τελευταίο βήμα είναι το πού θα το εμφανίσουμε. Η εμφάνιση γίνεται στο τέλος του αρχείου μας, με τη χρήση της εντολής :

```
$smarty->display('index.tpl');
```

Έτσι στην ουσία, εμφανίζουμε ολόκληρη την σελίδα, μαζί με τα δεδομένα που θέλουμε όπως τα καθορίσαμε μέσα στο index.tpl.

4.1.1 Αποτελέσματα

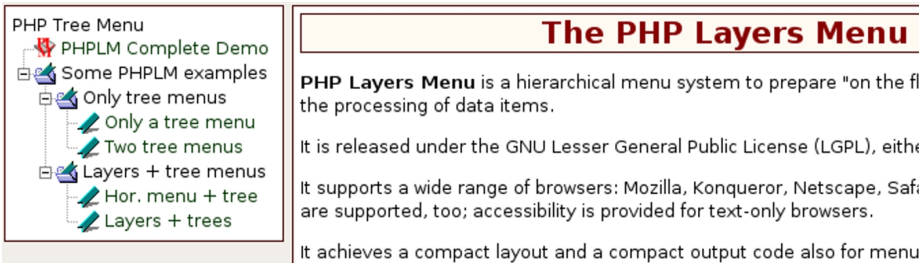
Με τη χρήση πολλών αρχείων για τη λήψη δεδομένων που αλλάζει τη ροή του προγράμματος, λύθηκαν κάποια σημαντικά προβλήματα. Όπως ήδη αναφέρθηκε, είναι κατανοητή η ροή του προγράμματος χωρίς να υπάρχει σύγχυση μεταξύ των αρχείων μας. Το σημαντικότερο πρόβλημα που αντιμετωπίζεται έτσι είναι ότι στην περίπτωση κάποιου σφάλματος σε κάποιο από τα αρχεία, δε σταματάει η λειτουργία του προγράμματος, εμφανίζοντας σφάλμα στη σελίδα, αλλά μόνο το συγκεκριμένο κομμάτι που έχει πρόβλημα δε λειτουργεί, δίνοντάς μας έτσι τη δυνατότητα να «ξανατρέξουμε» την ιστοσελίδα από την αρχή μη μπορώντας απλώς να χρησιμοποιήσουμε τη συγκεκριμένη λειτουργία. Αυτό βοηθάει ακόμη και στον εντοπισμό του προβλήματος γρήγορα, ώστε να διορθωθεί άμεσα.

Ένα άλλο πρόβλημα που λύνεται με τον τρόπο αυτόν είναι η ταχύτητα της λήψης των δεδομένων, αφού έτσι παίρνουμε συγκεκριμένες πληροφορίες κάθε φορά και όχι όλες τις πληροφορίες ταυτόχρονα, πράγμα που θα έκανε την ιστοσελίδα να είναι πιο «αργή» και να χρειάζεται μεγαλύτερη επεξεργασία των δεδομένων.

Επίσης, λύνεται και το πρόβλημα των πολλαπλών παραθύρων, όπου αρχικά χρειαζόταν κάθε φορά που επιλέγαμε μία κατηγορία, αυτή να ανοίγει ένα νέο παράθυρο και να εμφανίζει στο browser το όνομα του αρχείου που χρησιμοποιούμε κάθε φορά για κάθε νέα κατηγορία. . Το να εμφανίζεται μόνο το αρχείο index.php στο browser εκτός από την ευχρηστία του να εμφανίζονται όλα σε ένα παράθυρο, προσθέτει και ασφάλεια, καθώς δε γνωρίζει ο χρήστης τα αρχεία που χρησιμοποιεί η ιστοσελίδα.

4.2 Το μενού στην ιστοσελίδα

Όπως ήδη έχουμε αναφέρει, ο βασικός σκοπός της ιστοσελίδας μας είναι να μπορεί ο διαχειριστής ή ο χρήστης να παίρνει πληροφορίες για τις συσκευές του δικτύου. Για τον παραπάνω λόγο αποφασίσαμε να χρησιμοποιήσουμε ένα μενού σε PHP και όχι σε κάποια άλλη γλώσσα προγραμματισμού, όπως για παράδειγμα την java script, διότι μας ενδιαφέρει η λειτουργικότητά του. Είναι προφανές ότι, το μενού δεν έχει δημιουργηθεί από την αρχή, αλλά χρησιμοποιήσαμε σα βάση ένα παράδειγμα ενός δενδροειδούς μενού από τη σελίδα http://phplayersmenu.sourceforge.net με αρκετές διαφοροποιήσεις γιατί, όπως είναι λογικό, δε μπορούμε απλώς να χρησιμοποιήσουμε ένα μενού που να έχει τις λειτουργίες που εμείς χρειαζόμαστε, αλλά πρέπει να γίνουν αρκετές, και ταυτόχρονα σημαντικές αλλαγές.



Το μενού αυτό χρησιμοποιεί ένα αρχείο txt για την εμφάνιση των επιλογών του. Αυτό μας δημιουργούσε πρόβλημα διότι έπρεπε να χρησιμοποιήσουμε μεταβλητές για τις επιλογές μας, όπως επίσης και για την πραγματοποίηση ελέγχων. Έτσι, έπρεπε να ενσωματώσουμε το txt αρχείο στην PHP ώστε να μπορούμε να το χρησιμοποιήσουμε όπως εμείς θέλουμε. Η λύση που βρέθηκε, ήταν η τοποθέτηση του txt αρχείου του μενού, σε μία string μεταβλητή, μέσα στο αρχείο menu.php της ιστοσελίδας μας, ώστε να γίνεται δυνατή η χρήση php κώδικα σε αυτό.

Το αρχείο menu.php της ιστοσελίδας μας, είναι αυτό που δημιουργεί το μενού. Αρχικά απλώς δημιουργεί ένα αντικείμενο το οποίο είναι το μενού μας, και στη συνέχεια γίνεται ο έλεγχος διαθεσιμότητας της συσκευής. Αυτό γίνεται με την παρακάτω συνάρτηση find().

```
$userMenu=find($host, $community);
function find($host, $community){
      if ($host!=null && $community!=null)
            $doNow=existence($host, $community);
      if ($doNow=='0')
            $userMenu="";
      elseif ($doNow=='1'){
               $userMenu=
           ".|Host␣Resources|index.php?ip=$host&cn=$community|
The␣Host␣Resources␣menu||||1\n".
          "..|Host␣Resources␣System|index.php?ip=$host&cn=$community&ch=11|
Show␣the␣HR␣System␣info\n".
          "..|Host␣Resources␣Device|index.php?ip=$host&cn=$community&ch=12|
Show␣the␣HR␣Device␣info\n".
        "..|Host␣Resources␣Storage|index.php?ip=$host&cn=$community&ch=13|
Show␣the␣HR␣Storage␣info\n".
     "..|Running␣Software|index.php?ip=$host&cn=$community&ch=14|
Show␣the␣Running␣Software\n";
      }
```

```
     elseif($doNow=='2')
            $userMenu=
         ".|Cisco|index.php?ip=$host&amp;cn=$community|
The Cisco menu\n".
"..|Cisco|index.php?ip=$host&amp;cn=$community&amp;ch=15|
CDP of the device||||1\n".
"..|CDP Network|index.php?ip=$host&amp;cn=$community&amp;ch=16|
The whole Network||||1\n";
elseif ($doNow=='3'){
          $userMenu="";

        }
return $userMenu;
}
```

Η συνάρτηση find() ελέγχει αν έχει δοθεί IP και community name. Σε περίπτωση που αυτά υπάρχουν, μας οδηγεί στη συνάρτηση existence() η οποία χρησιμοποιεί με τη σειρά της την IP και το community name που τα παίρνει από τη συνάρτηση find() ώστε να βρει αν η συγκεκριμένη συσκευή είναι local υπολογιστής ή CISCO συσκευή.

```
function existance($host, $community){
   $old=ini_get('error_reporting');
#<-- the old error reporting for restoration
   ini_set('error_reporting',$old & (~E_WARNING));
#<-- turns off the warnings

    $sysName= snmpget($host,$community,
                "1.3.6.1.2.1.1.5.0",1000000 , 1 );
    $hrSystemUptime = snmpget($host,$community,
             "1.3.6.1.2.1.25.1.1.0",1000000 , 1 );
$cdpCacheVersion = snmpwalk($host,$community,
     "1.3.6.1.4.1.9.9.23.1.2.1.1.5",1000000 , 1 );
     if ($sysName!=null && $hrSystemUptime!=null)
           return 1;
     elseif ($sysName!=null &&
             $cdpCacheVersion!=null)
           return 2;
     ini_set('error_reporting',$old);
#<-- restores the warnings
return 0;
}
```

Αυτό γίνεται με τον εξής τρόπο. Πρώτα απενεργοποιούμε την εμφάνιση των προειδοποιήσεων από την PHP (WARNINGS) με την εντολή:

```
ini_set('error_reporting',$old & (~E_WARNING)
```

Με το ini_set() μπορούμε να αλλάξουμε τις τιμές της PHP. Έτσι, μπορούμε να αλλάξουμε την εμφάνιση των λαθών στην PHP. Με την παραπάνω εντολή απενεργοποιούμε την εμφάνιση των προειδοποιήσεων έτσι ώστε να κάνουμε τον έλεγχο που επιθυμούμε. Το επόμενο βήμα μας είναι να κάνουμε τρία SNMP requests προκειμένου να δούμε ποιά συσκευή δικτύου αντιστοιχεί στην IP που δώσαμε.

```
$sysName=snmpget($host,$community,
                    "1.3.6.1.2.1.1.5.0",1000000,1);

$hrSystemUptime=snmpget($host,$community,
                "1.3.6.1.2.1.25.1.1.0",1000000,1);

$cdpCacheVersion=snmpwalk($host,$community,
      "1.3.6.1.4.1.9.9.23.1.2.1.1.5", 1000000, 1);
```

Με το πρώτο request, ζητείται απο το SNMP να μας επιστρέψει το όνομα του συστήματος. Με το δεύτερο, παίρνουμε την τοπική ώρα του συστήματος και με το τρίτο, λαμβάνουμε τις πληροφορίες του πρωτοκόλλου CDP από τη συσκευή. Αν επιστραφεί τιμή στο πρώτο request, καταλαβαίνουμε ότι πρώτον, υπάρχει η συγκεκριμένη συσκευή και δεύτερον, ότι χρησιμοποιεί το πρωτόκολλο SNMP.

Για τα άλλα δύο request, περιμένουμε να μας επιστραφεί μόνο η μία από τις δύο τιμές. Αν επιστραφεί τιμή από το πρώτο, σημαίνει ότι είναι τοπικός υπολογιστής, οπότε και αλλάζει η εμφάνιση του μενού. Αν επιστραφεί από την δεύτερη, σημαίνει ότι έχουμε μια CISCO συσκευή. Αξίζει να σημειωθεί ότι και στην περίπτωση αυτή αλλάζει η εμφάνιση του μενού κατάλληλα.

Σε περίπτωση που δεν επιστραφεί τίποτα, τότε η συσκευή δεν υποστηρίζει SNMP ή δεν είναι ενεργή. Το μενού τότε είναι και πάλι διαφορετικό.

Η συνάρτηση αυτή, επιστρέφει τρεις διαφορετικές τιμές, μία για κάθε περίπτωση. Αν έχουμε τοπικό σύστημα, το μενού παίρνει τη μορφή που χρειάζεται για να πάρουμε πληροφορίες για το σύστημα. Αυτό γίνεται από τη συνάρτηση find() χρησιμοποιώντας μόνο την εντολή ελέγχου. Αφού επιλεγεί η μορφή του μενού, το αποτέλεσμα αποθηκεύεται στη μεταβλητή usermenu και επιστρέφεται.

Στη συνέχεια, το πρόγραμμα συνεχίζει τη ροή του από την επόμενη εντολή.

```
$struct= (
".|Enter␣New␣IP|index.php|Enter␣new␣IP␣for␣info\n".

".|System|index.php?ip=$host&amp;cn=$community|||
|\n".

"..|Generalinformation|index.php?ip=$host&amp;
cn=$community&amp;ch=1|
opening␣the␣system␣general␣information|||8\n");
```

Με αυτόν τον τρόπο, μπορούμε σε μία μεταβλητή να έχουμε ολόκληρο το μενού μας και αν προσθέσουμε την μεταβλητή που περιέχει το τροποποιημένο μενού, να έχουμε το τελικό μενού που θα εμφανίσουμε στην ιστοσελίδα μας.

```
$struct= (
".|Enter␣New␣IP|index.php|Enter␣new␣IP␣for␣info\n".

".|System|index.php?ip=$host&amp;cn=$community|||
|\n".
```

```
"..|General␣information|index.php?
ip=$host&amp;cn=$community&amp;ch=1|
opening␣the␣system␣genberal␣information|||8\n".
"...|SNMP|index.php?ip=$host&amp;cn=$community&amp;
ch=10|opening␣the␣Interface␣Information|||1\n".

$userMenu
);

"..|ARP␣Table|index.php?ip=$host&amp;
cn=$community&amp;ch=2|Show␣the␣ARP␣Table||||1\n".
```

Παραπάνω βλέπουμε την εντολή που δημιουργεί τις επιλογές του μενού. Οι τελείες στην αρχή, καθορίζουν το αν θα είναι κύριο μενού ή υπό-μενού. Ό,τι γράψουμε μετά την πρώτη κάθετο, είναι και η ετικέτα που θα εμφανίσει στο μενού. Από τη δεύτερη κάθετο και μετά, βάζουμε τον προορισμό που επιθυμούμε. Αυτός είναι και ο λόγος που διαφοροποιήσαμε το μενού και τοποθετήσαμε το αρχείο txt μέσα στην PHP. Με τον τρόπο αυτό έχουμε τον προορισμό ακολουθούμενο από την IP και το community name όπως επίσης και τη μεταβλητή ch με την τιμή της.

Η μεταβλητή ch είναι αυτή που καθορίζει στο index.php την επιλογή του switch/case και μας οδηγεί στο κατάλληλο αρχείο php ώστε να πάρουμε τις πληροφορίες που θέλουμε. Στην τρίτη κάθετο, μπορούμε να βάλουμε κάποια σύντομη περιγραφή της συγκεκριμένης καρτέλας. Τέλος, ακολουθούν κάποιες εντολές δημιουργίας του μενού. Το αντικείμενο που δημιουργήθηκε το μενού δίνει κάποιες λειτουργίες. Αυτές αφορούν το τι θα περιλαμβάνει το μενού, το μέγεθος των εικονιδίων, τις επεκτεινόμενες επιλογές κ.α που δεν είναι απαραίτητα να περιγραφούν. Αμέσως μετά τη δημιουργία του μενού γίνεται η εμφάνισή του μέσω του index.tpl.

4.3 Session στην ιστοσελίδα

Ένα από τα προβλήματα που αντιμετωπίσαμε κατά την ανάπτυξη της ιστοσελίδας μας ήταν αυτό του χρόνου και της επανάληψης της εκτέλεσης ίδιων κομματιών του κώδικα. Κατά την εκκίνηση, ο κώδικας «έτρεχε» κάθε φορά και ζητούσε τις ίδιες πληροφορίες για το ίδιο σύστημα από την αρχή. Αυτό όπως είναι φυσικό, είχε σαν αποτέλεσμα τη σπατάλη χρόνου και την επεξεργασία των δεδομένων από την αρχή κάθε φορά. Έτσι, ήταν αναγκαία η εισαγωγή κάποιου συστήματος προσωρινής αποθήκευσης των δεδομένων. Για το λόγο αυτόν χρησιμοποιήσαμε τα session της PHP ώστε να λύσουμε το συγκεκριμένο πρόβλημα.

Κάθε φορά που τρέχει η ιστοσελίδα μας ξεκινάει το SESSION με την εντολή:

```
session_start();
```

Έτσι δημιουργούμε έναν χώρο στον SERVER/PC ώστε να κρατάμε κάποια δεδομένα. Αυτά αφορούν κάθε συσκευή που θέλουμε ξεχωριστά. Όταν δίνουμε μια IP στην ιστοσελίδα μας, αυτή αποθηκεύεται στον πίνακα των SESSION. Ένα παράδειγμα του πίνακα των SESSION δίνεται στην παρακάτω εικόνα:

```
SESSION          -> [IP1]     -> Systable
                                -> ARPtable
                 -> [IP2]     -> Systable
                                -> ARPtable
                 -> [IP3]     -> Systable
                                -> ARPtable
```

4.3.1 Λειτουργία του Session

Κάθε φορά που μπαίνουμε στην ιστοσελίδα και αφού ξεκινήσει το SESSION, γίνεται ένας έλεγχος για το αν έχουμε επισκεφτεί ξανά τη συσκευή με τη συγκεκριμένη IP. Αυτό γίνεται με την εντολή της PHP:

```
isset($_SESSION[$host]['views'])
```

Έτσι, με έναν απλό έλεγχο μπορούμε να δούμε αν έχουμε ξαναπάρει πληροφορίες από τη συγκεκριμένη συσκευή. Στην περίπτωση που έχουμε επισκεφτεί τη συσκευή αυτή, ο μετρητής αυξάνεται κατά ένα και μας δείχνει το πόσες φορές την έχουμε επισκεφτεί αλλιώς ο μετρητής μας δείχνει απλώς ότι είναι η πρώτη φορά που επισκεφτήκαμε τη συγκεκριμένη συσκευή αυτήν.

Στη συνέχεια της ροής της ιστοσελίδας, αφού έχουμε δώσει νέα IP και Community name (πρώτη φορά), γίνεται ένας έλεγχος για το αν είναι ήδη διαθέσιμες οι πληροφορίες, πράγμα που σημαίνει ότι έχουμε ξαναδεί αυτήν τη συσκευή και απλώς να τις εμφανίσουμε, ή αν δεν έχουμε ξαναδεί τη συσκευή αυτή να δημιουργήσουμε από την αρχή πίνακες στο SESSION για την IP της συσκευής.

```
if(isset($_SESSION[$host]['sysTable'])){

       $smarty->assign('sysTable',
$_SESSION[$host]['sysTable']);
}
else {

$_SESSION[$host]['sysTable'] =
array ('host' =>array ('Host␣IP', $host),
   'sysName'  =>array ('Name', snmpget("$host",
"$community","1.3.6.1.2.1.1.5.0")),
 'sysDescr' =>array ('Desrciption', , snmpget("$host",
"$community","1.3.6.1.2.1.1.3.0")));
   $smarty->assign('sysTable',
$_SESSION[$host]['sysTable']);
     }
```

Το παραπάνω παράδειγμα κώδικα δείχνει τον τρόπο με τον οποίο αποθηκεύουμε τα δεδομένα μας σε πίνακες του SESSION. Έτσι για κάθε IP έχουμε υπό-πίνακες που δείχνουν σε στοιχεία που έχουμε λάβει από τις συσκευές. Όταν δημιουργείται εκ νέου ένας πίνακας από IP, έχουμε έναν πίνακα SESSION[host] ο οποίος περιέχει έναν πίνακα SESSION[Host][Systable] που περιέχει τα δεδομένα που θέλουμε. Με

τις εντολές του SNMP, snmpget() παίρνουμε τα δεδομένα και τα αποθηκεύουμε απευθείας στο SESSION. Αφού ολοκληρωθεί η λήψη των δεδομένων, αναθέτουμε τον πίνακα του SESSION στο Smarty για εμφάνιση.

Στο σημείο αυτό θα πρέπει να σημειώσουμε ότι στις περισσότερες περιπτώσεις δεν αναθέτουμε τιμές στο SESSION, αλλά ολόκληρους πίνακες. Αυτό γίνεται διότι όταν παίρνουμε τιμές από μία συσκευή, όπως για παράδειγμα, ένα Routing Table (πίνακας δρομολόγησης) , δε θέλουμε μόνο μία τιμή αλλά ολόκληρο το Routing Table κάθε φορά. Έτσι, κάθε φορά που θέτουμε ένα πίνακα στο SESSION, χρειάζεται η κατάλληλη επεξεργασία ώστε να γίνουν κατανοητά και ορθά τα δεδομένα.

Ένα ακόμα στοιχείο που περιέχεται στην ιστοσελίδα και συγκεκριμένα στο index.php, αλλά εμφανίζεται παντού ώστε να είναι διαθέσιμο, είναι το παράθυρο Visited IP. Σε αυτό εμφανίζεται το ιστορικό των τελευταίων πέντε συσκευών που έχουμε επιτυχώς επισκεφθεί. Αυτό πραγματοποιήθηκε με τη χρήση του session.

Κατά την πρώτη φορά που ανοίγει η ιστοσελίδα, γίνεται ένας έλεγχος για το αν έχει δοθεί IP διεύθυνση και community name. Εφόσον είναι η πρώτη φορά, δεν υπάρχει περίπτωση να υπάρχει IP διεύθυνση και έτσι περνάει σε διαδικασία όπου πρέπει να εισάγουμε δεδομένα. Μετά τον έλεγχο, γίνεται ένας πρόσθετος έλεγχος για να δούμε αν έχει δημιουργηθεί ο πίνακας του session $_SESSION[Vhosts]. Αυτός ο πίνακας θα περιέχει τις διευθύνσεις που επισκεπτόμαστε κάθε φορά. Υπενθυμίζουμε ότι όταν ανοίγει η σελίδα για πρώτη φορά, γίνεται έλεγχος για το αν έχει δοθεί IP αλλά και όταν επιλέγουμε να δώσουμε νέα IP, ξαναγίνεται ο ίδιος έλεγχος. Έτσι, εφόσον είναι η πρώτη φορά που ανοίξαμε την ιστοσελίδα και δεν έχει δημιουργηθει ο πίνακας Vhosts, δημιουργείται επίσης και ο πίνακας VChosts, ο οποίος περιέχει τα community names για την κάθε visited IP του πίνακα και τέλος, αρχικοποιούμε την πρώτη θέση του πίνακα στην τιμή 'No IP given'. Στη συνέχεια, αναθέτουμε στο Smarty την εμφάνιση του παράθυρου αυτού.

Έχουμε ήδη αναφέρει το πώς γίνεται ο έλεγχος της IP διεύθυνσης που εισάγουμε κάθε φορά. Αμέσως μετά από αυτόν τον έλεγχο, χρησιμοποιείται ένας νέος έλεγχος για το Vhosts. Ο έλεγχος αυτός περιλαμβάνει τρεις περιπτώσεις.

1. Αν στην πρώτη θέση του πίνακα αυτού, υπάρχει η τιμή 'No IP given', τότε αντικαθιστά αυτήν την τιμή με την IP διεύθυνση που δώσαμε και θέτει στην πρώτη θέση του πίνακα VChosts το αντίστοιχο community name.

2. Ελέγχουμε αν το πλήθος των στοιχείων του πίνακα Vhosts είναι ίσο με πέντε και αν στον πίνακα περιέχεται η διεύθυνση IP που χρησιμοποιείται τη συγκεκριμένη χρονική στιγμή. Αν το πλήθος είναι ίσο με πέντε και δεν υπάρχει ήδη η διεύθυνση IP στον πίνακα, τότε με έναν βρόχο αντικαθίσταται η πρώτη τιμή του πίνακα από τη δεύτερη, η δεύτερη τιμή του πίνακα με την τρίτη και ούτω καθεξής, όπως και για τον πίνακα VChosts και τέλος, στην πέμπτη τιμή των δύο αυτων πινάκων, τοποθετείται η IP και το community name.

3. Γίνεται απλώς ο έλεγχος για το αν υπάρχει η IP αυτή στον πίνακα. Αν δεν υπάρχει τότε τον τοποθετεί και συνεχίζεται η ροή του προγράμματος.

Για την υλοποίηση αυτής της λειτουργίας, χρησιμοποιούμε το session το οποίο μας δίνει τη δυνατότητα να «κρατάμε» τις τιμές για άλλες χρήσεις. Ο κώδικας που υλοποιήθηκε η διαδικασία αυτή δίνεται παρακάτω.

```
if ($_SESSION['Vhosts'][0]=='No␣IP␣given'){
```

```
        $_SESSION['Vhosts'][0]=$host;
        $_SESSION['VChosts'][0]=$community;
}
elseif(count($_SESSION['Vhosts'])==5 &&
!in_array($host,$_SESSION['Vhosts'])){
        $temp=array();
        for($i=0;$i<4;$i++){
                $temp[0][$i]=$_SESSION['Vhosts'][$i+1];
$temp[1][$i]=$_SESSION['VChosts'][$i+1];
        }
        $temp[0][4]=$host;
        $temp[1][4]=$community;
        $_SESSION['Vhosts']=null;
        $_SESSION['VChosts']=null;
        $_SESSION['Vhosts']=$temp[0];
        $_SESSION['VChosts']=$temp[1];
} // elseif
elseif(!in_array($host,$_SESSION['Vhosts']))
$_SESSION['Vhosts'][count($_SESSION['Vhosts'])]=$host;
          $_SESSION['VChosts']
          [count($_SESSION['VChosts'])]=$community;
} //if (Net_CheckIP::check_ip($host))
   else {
          $smarty->assign('errMsg', "Not␣a␣valid␣IP");
          $smarty->assign('IPbox', $IPbox);
   }

if (!isset($_SESSION['Vhosts'])){
          $_SESSION['Vhosts']=array();
          $_SESSION['VChosts']=array();
          $_SESSION['Vhosts'][0]='No␣IP␣given';
}
$smarty->assign('IPbox', $IPbox);
}
```

4.3.2 Αποτελέσματα σχετικά με το Session

Φαίνεται ξεκάθαρα ότι με τη χρήση των SESSION γλιτώνουμε κόστος τόσο σε χρόνο όσο και σε επεξεργασία αφού δε χρειάζεται να επαναλαμβάνεται η χρήση ίδιων κομματιών κώδικα για να πάρουμε τις ίδιες πληροφορίες. Παρ'ολα αυτά, σε περίπτωση που απαιτείται η λήψη των δεδομένων από την αρχή, δημιουργήθηκε μία επιλογή στο μενού (επιλογή Reset Session) ώστε να διαγράφεται το SESSION και να γίνεται εκ νέου η λήψη των δεδομένων. Για παράδειγμα ένα Routing Table (Πίνακας Δρομολόγησης) είναι μεταβαλλόμενο ανά τακτά χρονικά διαστήματα. Έτσι, αν χρειαζόμαστε κάποιο νέο στιγμιότυπο του πίνακα, απλώς πρέπει να διαγραφεί το SESSION και να επαναληφθεί η λήψη των διαφόρων πινάκων που αφορούν το Routing Table, να ξαναγίνει η επεξεργασία και να σταλούν στο Smarty όλοι οι πίνακες για να γίνει σωστή και ανανεωμένη εμφάνιση του πίνακα δρομολόγησης.

5 Cisco συσκευές

Ένα από τα σημαντικά κομμάτια της πτυχιακής αφορά τις Cisco συσκευές καθώς και την ανακάλυψη ολόκληρου του δικτύου. Όπως είναι λογικό, μέσω του SNMP μπορούμε να πάρουμε πληροφορίες από πολλές διαφορετικές συσκευές. Ο σκοπός της πτυχιακής ήταν να πάρουμε πληροφορίες από διάφορες συσκευές, να τις επεξεργαστούμε και στη συνέχεια να δώσουμε βαρύτητα στις συσκευές της εταιρίας Cisco που είναι μια από τις μεγαλύτερες εταιρίες στο χώρο των δικτύων.

Είναι προφανές ότι ζητώντας τις κατάλληλες πληροφορίες μπορούμε να πάρουμε δεδομένα από οποιαδήποτε εταιρία θέλουμε. Το πώς μπορούμε να κάνουμε κάτι τέτοιο θα το εξηγήσουμε παρακάτω. Έχουμε ήδη αναφερθεί σε κάποια βασικά πράγματα για τη λειτουργία της σελίδας μας. Όταν ο χρήστης εισάγει μια διεύθυνση IP, εκτός των άλλων γίνεται και ο έλεγχος για το αν είναι μια συσκευή Cisco και στη συνέχεια εμφανίζεται η επιλογή Cisco ή όχι στο μενού. Ήδη έχει εξηγηθεί το πώς υλοποιούμε την διαδικασία αυτή με την χρήση της existence() όταν εξηγήθηκε η δημιουργία του μενού.

Αφού έχει πραγματοποιηθεί ο έλεγχος της IP και του community name, καθώς και η κλήση της existence() θα δημιουργηθεί το μενού με τις επιλογές για τις συσκευές Cisco. Όπως είπαμε, οι πληροφορίες που παίρνουμε για τις συσκευές Cisco είναι από το πρωτόκολλο CDP. Έτσι, μπορούμε να πάρουμε διάφορες πληροφορίες σχετικά με μια Cisco συσκευή χρησιμοποιώντας απλώς το αντίστοιχο MIB της Cisco, το οποίο, είναι διαδεδομένο και μπορούμε να το χρησιμοποιήσουμε. Ένα σημαντικό στοιχείο είναι ότι μπορούμε να τα χρησιμοποιήσουμε απευθείας από το διαδίκτυο, χωρίς να χρειαστεί να αποθηκεύσουμε ολόκληρο το MIB, απλώς γράφοντας το αντίστοιχο OID της πληροφορίας που μας ενδιαφέρει.

5.1 CDP

Για να πάρουμε πληροφορίες σχετικά με τις Cisco συσκευές, έγινε η χρήση του πρωτοκόλλου CDP.

Το πρωτόκολλο CDP (Cisco Discovery Protocol) είναι ένα πρωτόκολλο που δημιουργήθηκε από την Cisco και χρησιμοποιείται απο τις συσκευές της. Η δουλειά του είναι να ανακαλύπτει τις άμεσα συνδεδεμένες συσκευές που βρίσκονται στην αρχική συσκευή που χρησιμοποιούμε. Όταν μία συσκευή Cisco έχει ενεργό το πρωτόκολλο CDP, στέλνει μηνύματα προς τους γείτονες (άμεσα συνδεδεμένες συσκευές), ώστε να κοινοποιήσει την παρουσία της και να ανακαλύψει τους γείτονές της. Αυτό βέβαια, ισχύει μόνο για τις Cisco συσκευές οι οποίες έχουν ενεργό το πρωτόκολλο αυτό.

Στην ιστοσελίδα μας θέλαμε να δούμε ποιες συσκευές είναι άμεσα συνδεδεμένες στη συγκεκριμένη συσκευή που θέλουμε. Έτσι, χρησιμοποιήσαμε το πρωτόκολλο CDP ώστε να ανακαλύψουμε τις συσκευές αυτές. Αυτό που κάναμε είναι να βρούμε ποιες είναι αυτές οι συσκευές μέσω των IP διευθύνσεων που υπάρχουν διαθέσιμες στους πίνακες που κρατάει το CDP και να κάνουμε ερωτήματα στις συσκευές αυτές, για να πάρουμε τις πληροφορίες που θέλουμε. Παρακάτω βλέπουμε ένα κομμάτι των αποτελεσμάτων αυτής της διαδικασίας. Οι βασικές πληροφορίες που πήραμε για τις Cisco συσκευές είναι οι εξής (τα δεδομένα τα πήραμε από την ιστοσελίδα μας και απλώς εδώ τα εμφανίζουμε με τη μορφή πίνακα):

Address	DeviceId	DevPort	Platform	IosVer
192.168.16.131	it2	GigabitEthernet0/1	Cisco WS-C3550-12G	12.2(50)SE3

Η πρώτη πληροφορία που ζητούμε είναι η cdpCacheAddress η οποία ουσιαστικά μέσω του CDP μας επιστρέφει όλους τους άμεσα συνδεδεμένους γείτονες της συσκευής από την οποία έχουμε ζητήσει τις πληροφορίες.

Στη συνέχεια παίρνουμε πληροφορίες σχετικά με το όνομα της συσκευής. Αυτό το ζητούμε μέσω της cdpCacheDeviceId. Αμέσως μετά παίρνουμε την πληροφορία σχετικά με τη θύρα του γείτονα στην οποία είμαστε συνδεδεμένοι, διαδικασία που πραγματοποιείται μέσω του cdpCacheDevicePort. Η επόμενη στήλη αναφέρεται στο μοντέλο της συσκευής του γείτονα και στη συνέχεια παίρνουμε και το ios version του κάθε γείτονα. Παρακάτω θα παραθέσουμε έναν πίνακα που σχετίζεται με την πληροφορία που ζητούμε καθώς και με το αντίστοιχο OID για καθεμία πληροφορία της συγκεκριμένης καρτέλας.

Αρχικά, έχουμε βάλει ένα τμήμα του OID σε μια μεταβλητή για πρακτικούς λόγους. Στη συνέχεια, καλούμε ολόκληρο το OID για τη συγκεκριμένη πληροφορία, έχοντας δώσει φυσικά και το host αλλά και το community name.

```
$mib="1.3.6.1.4.1.9.9.23.1.2.1.1";
$_SESSION[$host]['ciscoTable']['CacheAddressType']
=snmpwalk($host,$community,"$mib.3");
$_SESSION[$host]['ciscoTable']['CacheAddress']
=snmpwalk($host,$community,"$mib.4");
$_SESSION[$host]['ciscoTable']['CacheDevicePort']
=snmpwalk($host,$community,"$mib.7");
$_SESSION[$host]['ciscoTable']['CachePlatform']
=snmpwalk($host,$community,"$mib.8");
```

Τα παραπάνω είναι απλά παραδείγματα για να δούμε πως ζητούμε πληροφορίες σχετικά με μια Cisco συσκευή. Όπως παρατηρούμε η διαδικασία δε διαφέρει σε κάτι από το να ζητήσουμε μια πληροφορία από μια οποιαδήποτε συσκευή.

6 Εύρεση Cisco δικτύου

Η επόμενη επιλογή στο μενού μας σχετικά με τη Cisco είναι να πάρουμε μέσω του πρωτοκόλλου cdp, πληροφορίες σχετικά με τις συσκευές που υλοποιούν το πρωτόκολλο αυτό το οποίο ουσιαστικά μας δίνει ολόκληρο το δίκτυο (CDP Network).

Η διαδικασία που ακολουθείται για να ανακαλυφθεί ολόκληρο το CDP Network περιγράφεται αναλυτικά παρακάτω ενώ ακολούθως αναλύονται όλες οι διαδικασίες και οι συναρτήσεις που χρησιμοποιήθηκαν για να βγάλουμε ένα σωστό αποτέλεσμα. Η όλη διαδικασία γίνεται με τη χρήση τριών πινάκων.

```
Temp -' πίνακας για προσωρινή αποθήκευση ip
Visited -' πίνακας με τις ip του δικτύου που έχουμε
          ήδη επισκεφτεί
To_Search -' οι ip τις οποίες θα ψάξουμε να βρούμε
```

Η διαδικασία αυτή, με τη χρήση κατάλληλων συναρτήσεων μας δίνει στον πίνακα visited, όλες τις IP των Cisco συσκευών του δικτύου. Θα δούμε όλα τα βήματα ένα προς ένα για να διαπιστώσουμε πως φτάνουμε στην τελική μορφή έχοντας ολόκληρο το δίκτυο.

Εν πρώτοις, έχουμε μια IP την οποία εισάγουμε στον to_search. Η IP αυτή είναι η IP της Cisco συσκευής από την οποία παίρνουμε τις πληροφορίες. Στη συνέχεια, μέσω του SNMP ζητάμε όλες τις IP των γειτόνων της συγκεκριμένης συσκευής και εισάγουμε το αποτέλεσμα αυτής της αναζήτησης στον πίνακα tmp. Ο tmp τώρα περιέχει τους γείτονες της αρχικής συσκευής που είχαμε. Ακολούθως προσθέτουμε στο visited την IP από την οποία πήραμε τους γείτονες. Αυτό γίνεται με τη χρήση μιας συνάρτησης η οποία δίνει στο visited μόνο τις IP τις οποίες δεν περιέχει ήδη. Η λειτουργία της αναλύεται παρακάτω. Μετά από αυτή τη διαδικασία συγκρίνουμε τον tmp με τον visited έτσι ώστε ο tmp να μην «ψάξει» ξανά IP που περιέχονται στο visited. Στη συνέχεια ο to_search παίρνει τις τιμές του tmp ώστε να ψάξουμε για αυτές τις IP. Η διαδικασία αυτή επαναλαμβάνεται όσο ο to_search έχει εγγραφές, ενώ με τη χρήση κατάλληλων συναρτήσεων μας δίνει, στον πίνακα visited, όλες τις IP των Cisco συσκευών του δικτύου. Θα δούμε όλα τα βήματα ένα προς ένα για να διαπιστώσουμε πως φτάνουμε στην τελική μορφή έχοντας ολόκληρο το δίκτυο. Ακολουθεί ο κώδικας για αυτή τη διαδικασία.

```
while($to_search!=null && is_array($to_search)
&& count($to_search)>0 ){
    $tmp=$snmp->multi_walk($to_search,
 "1.3.6.1.4.1.9..9.23.1.2.1.1.4", $community);
    $tmp=fixTmp($tmp);
    $visited=addNewIP($visited, $to_search);
    $tmp=array_diff($tmp, $visited);
    $to_search=$tmp;
    $to_search=clear_to_search($to_search);
} //while
```

Στον κώδικά μας παρεμβάλλονται και κάποιες συναρτήσεις οι οποίες αναλύονται παρακάτω. Η πρώτη συνάρτηση η οποία συναντάμε είναι η fixTmp η οποία σβήνει τους υπό-πίνακες και βάζει όλα τα στοιχεία σε ένα πίνακα και στο τέλος μας επιστρέφει τον πίνακα αυτό. Παραθέτουμε τον κώδικα της συνάρτησης:

```
function fixTmp($tmp){
    $key=0;
    $tmp2=array();
    foreach ($tmp as $temp){
        foreach($temp as $t){
            if (strlen($t)==11)
                $tmp2[$key++]=$t;
        }
    }
    $tmp=null;
    $tmp=$tmp2;
    $tmp=fixIpHexDec($tmp);
return $tmp;
}
```

Όπως παρατηρούμε η fixTmp καλεί την fixIpHexDec της οποίας η λειτουργία είναι η μετατροπή των IP από δεκαεξαδική μορφή σε δεκαδική. Η συνάρτηση fixIpHexDec() χρησιμοποιείται και από το Cisco.php όπου ουσιαστικά, το SNMP επιστρέφει την ίδια μορφή IP. Επειδή υπάρχουν κάποιες συναρτήσεις που χρησιμοποιούνται από περισσότερα από ένα αρχεία, θεωρήσαμε σκόπιμο να δημιουργήσουμε ένα νέο αρχείο, το οποίο θα περιλαμβάνει τις κοινά χρησιμοποιούμενες συναρτήσεις, ώστε να μην χρειάζεται να αντιγράφονται. Παρακάτω δίνεται ο κώδικας της συνάρτησης αυτής και αμέσως μετά εξηγείται η λειτουργία της.

```
function fixIpHexDec($ipToFix){
   for ($x=0;$x<sizeof($ipToFix);$x++){
       if($ipToFix[$x]!='"' || $ipToFix[$x]!=''
   || $ipToFix[$x]!=null){
$ipHexDecArr=str_split($ipToFix[$x],"3");
       $ipHexDecArr[0] = str_replace('"', "",
       $ipHexDecArr[0]);
           $ipHexDecArr[4] =null;

       for ($i=0;$i<sizeof($ipHexDecArr);$i++){
         $ipHexDecArr[$i]=hexdec($ipHexDecArr[$i]);

         }
     $ipToFix[$x]="$ipHexDecArr[0].$ipHexDecArr[1].
$ipHexDecArr[2].$ipHexDecArr[3]";
       }
      else {$ipToFix[$x]='no IP';
     }
  }
return $ipToFix;
}
```

Η συνάρτηση χρησιμοποιώντας ένα βρόχο επανάληψης, χωρίζει κάθε τιμή του πίνακα $ipToFix, που αντιστοιχεί σε μία δεκαεξαδική IP, σε ένα νέο πίνακα ανά τρείς χαρακτήρες, δηλαδή η IP σε δεκαεξαδική μορφή:

"C0 A8 10 83 "

Θα μπει σε ένα πίνακα όπου θα είναι:

[0] –>"C0
[1] –>A8
[2] –>10
[3] –>83
[4] –> "

Από αυτόν τον πίνακα ζητάμε να αντικατασταθούν τα " με κενό από την πρώτη θέση του πίνακα και επίσης ορίζουμε σαν null την τελευταία θέση του πίνακα όπου είναι πάντα ' " ' . Αυτό γίνεται γιατί η IP είναι σταθερού μεγέθους οπότε και μπορούμε να έχουμε στατικά την θέση αυτή του πίνακα. Μετά με έναν βρόχο επανάληψης αλλάζουμε την κάθε τιμή του πίνακα από δεκαεξαδικό αριθμό σε δεκαδικό. Τέλος στον παλιό πίνακα μας ενώνουμε την IP διεύθυνση και την επιστρέφουμε.

Η επόμενη συνάρτηση που συναντάμε είναι η addNewIP με την οποία ελέγχονται οι πίνακες visited και to_search αν έχουν ίδια στοιχεία. Αν υπάρχουν ίδια στοιχεία τότε τα παραβλέπει, διαφορετικά βάζει την IP, την οποία ελέγχουμε στο visited, και στο τέλος μας επιστρέφει τον visited.

```
function addNewIP($v, $t_s){
    foreach ($t_s as $t){
        $key=sizeof($v);
        for ($i=0; $i<=$key; $i++){
            if (array_search($t, $v)===false
|| !(in_array($t, $v))|| $t!=$v[$i]|| $t!='0.0.0.0')
                $v[$key]=$t;
        }
    }
return $v;
}
```

Βλέπουμε ότι στη addNewIP καλούμε επίσης και την array_search η οποία ψάχνει αν υπάρχει το $t (IP του πίνακα to_search) στον πίνακα $v (πίνακας $visited) . Στην συνέχεια καλούμε και την in_array, και ελέγχουμε επίσης αν το $t με το $v[$i] είναι διαφορετικό ενώ στο τέλος γίνεται έλεγχος αν υπάρχει μηδενική IP.

Ακολούθως, στην κανονική ροή του προγράμματος, χρησιμοποιούμε την array_diff η οποία είναι μια συνάρτηση της PHP και μας επιστρέφει τις διαφορές του πρώτου πίνακα και του δεύτερου. Στη δικιά μας περίπτωση μας επιστρέφει τις διαφορές του tmp από το visited.

Η τελευταία συνάρτηση που χρησιμοποιούμε μέσα στον βρόχο while είναι η clear_to_search η οποία « καθαρίζει» τον πίνακα to_search από μηδενικές IP και από null τιμές.

```
function clear_to_search($to_search){
    $s=0;
    $arrTs=array();
    $size=sizeof($to_search);
    for ($i=0;$i<=sizeof($to_search);$i++){
        if ($to_search[$i]!='0.0.0.0'
```

```
|| $to_search[$i]!=null ||
$to_search[$i]!='' || $to_search[$i]!="")
{
          $temp=$to_search[$i];
          $s+=1;
      }
      for ($s; $s<=$size;$s++
          if($temp==$to_search[$s]
|| $to_search[$s]=='0.0.0.0')
                $to_search[$s]=null;
        $s=$i;
        $arrTs[sizeof($arrTs)]=$temp;
    }
$arrTs=fixArray($arrTs);
return $arrTs;
}
```

Στην clear_to_search παρατηρούμε επίσης ότι καλούμε και την fixArray η οποία καθαρίζει από null τιμές τον πίνακα.

```
function fixArray($arr){
    $x=0;
    $temp=array();
    foreach ($arr as $a){
        if($a!=null || $a!='' || $a!="")
            $temp[$x++]=$a;
    }     $arr=null;
    $arr=$temp;
return $arr;
}
```

Στη συνέχεια της ροής του προγράμματος, παίρνουμε τις πληροφορίες που χρειαζόμαστε από τις συσκευές οι οποίες είναι οι εξής:

Device Ip	System Name	Description	System Time	IOS Version	Interfaces
195.251.123.1	it	catalyst355012G	19days 11hours	12.2(44)SE3	27

Οι πληροφορίες που παίρνουμε είναι η Device IP, την οποία παίρνουμε από τον πίνακα visited και είναι οι IP όλου του δικτύου. Στη συνέχεια, παίρνουμε το system name που είναι το όνομα της συσκευής. Αμέσως μετά, ζητάμε το μοντέλο της συσκευής και έπειτα το χρόνο που είναι ανοιχτή η συγκεκριμένη συσκευή. Το IOS Version το παίρνουμε μέσα από μια διαφορετική διαδικασία η οποία θα αναλυθεί παρακάτω και τέλος, παίρνουμε τον αριθμό των interface.

Πρέπει να σημειώσουμε εδώ την ύπαρξη μιας διαφοροποίησης. Λόγω του ότι υπήρχαν προβλήματα με τη διαχείριση των κενών τιμών των πινάκων, λόγω

συσκευών που δεν επέστρεφαν τιμές, ενώ είναι CISCO συσκευές, όπως για παράδειγμα IP τηλέφωνα, έπρεπε να έχουμε τον πίνακα $visited με όλες τις χρήσιμες IP, δηλαδή, αυτές που μπορούν να επιστρέψουν πληροφορίες. Οπότε έγινε δυνατή η χρήση του $cdpDescr, ο οποίος είναι ένας πίνακας που έχει όλα τα description από όλες τις CISCO συσκευές που λειτουργούν και μπορούν να δώσουν πληροφορίες σε ολόκληρο το δίκτυο. Αφού υπήρχαν διαθέσιμες οι πληροφορίες με τη χρήση του phpSnmp, μπορούσαμε να κάνουμε διαφορετική επεξεργασία σε σχέση με αυτήν του NET-SNMP.

Όταν επέστρεφε πληροφορίες το NET-SNMP με τη χρήση της εντολής snmpwalk(), το αποτέλεσμα ήταν ένας πίνακας όπου κάθε δείκτης ήταν ένας αύξων αριθμός και σαν τιμή είχε την πληροφορία ακόμα και όταν δεν υπήρχε κάποια πληροφορία. Στο παράδειγμα παρακάτω γίνεται απόλυτα κατανοητός ο τρόπος που γίνεται αυτό.

Στιγμιότυπο του πίνακα ARP_Table :

```
Array
(
    [ifIndex] => Array
        (
            [0] => 1
            [1] => 2
                .
                .
                .
            [13] => 14
            [14] => 15
            [15] => 16
    [ifDescr] => Array
        (
            [0] => GigabitEthernet0/1
            [1] => GigabitEthernet0/2
                .
                .
                .
            [13] => Vlan1
            [14] => Tunnel1
            [15] => Loopback0
    [ifSpeed] => Array
        (
            [0] => 1Gbps
            [1] => 1Gbps
                .
                .
                .
            [13] => 1Gbps
            [14] => null
            [15] => 4.29Gbps
```

Ενώ με την χρήση του phpSnmp, η εμφάνιση των πινάκων γίνεται ως εξής:

```
Array
(
    [195.251.123.1] => Array
        (
            [.1.3.6.1.2.1.1.2.0] =>
                    .1.3.6.1.4.1.9.1.431
        )
    [195.251.240.129] => Array
        (
         [.1.3.6.1.2.1.1.2.0] =>
                    .1.3.6.1.4.1.9.1.502
       )
    [195.251.121.94] => Array
        (
         [.1.3.6.1.2.1.1.2.0] =>
                    .1.3.6.1.4.1.9.1.325
        )
    [192.168.229.202] => Array
        (
        )
```

Βλέπουμε την διαφορά με τη χρήση NET-SNMP και phpSnmp.

Όταν χρησιμοποιούμε το phpsnmp οι πληροφορίες που μας επιστρέφει έχουν διαφορετική μορφή από αυτή του net snmp. Το phpsnmp μας επιστρέφει ένα πίνακα με ένα δείκτη την IP όπου έχει σαν υπό-πίνακα, ένα πίνακα με δείκτη το OID το οποίο έχει την τιμή που θέλουμε.

Για παράδειγμα:

```
Array (
       [195.251.123.1] =>
                       [1.3.6.1.2.1.1.2.0] =>
                                       catalyst355012G
)
```

Προκειμένου να έχουμε τον πίνακα σε μια μορφή που να μπορούμε να τον επεξεργαστούμε εύκολα και γρήγορα, χρησιμοποιούμε τη συνάρτηση fixTheArray:

```
function fixTheArray($arrayToFix, $oid, $ip){
    $vis3=array();
    for ($i=0;$i<sizeof($arrayToFix);$i++){
          $vis3[$i]=$arrayToFix[$ip[$i]];
    }     $arrayToFix=null;
    for ($i=0;$i<sizeof($vis3);$i++){
        if($vis3[$i][$oid]!=null)
            $arrayToFix[$i]= $vis3[$i][$oid];
        else
              $arrayToFix[$i]= null;
    }
return $arrayToFix;
}
```

η οποία στην ουσία παίρνει τη χρήσιμη πληροφορία και φτιάχνει ένα πίνακα με δεκαδικούς δείκτες και τιμές την πληροφορία που θέλουμε.

Ένα από τα προβλήματα που παρουσιάστηκαν κατά την εμφάνιση του δικτύου ήταν η μορφή του χρόνου. Αρχικά, είχαμε το χρόνο σε δευτερόλεπτα με τα κλάσματα του δευτερολέπτου στο τέλος. Όπως ήταν αναμενόμενο, παρουσιάζοντας έτσι το χρόνο ήταν δυσνόητος από τον οποιοδήποτε και δεν εξυπηρετούσε κάποιο σκοπό. Για το λόγο αυτό δημιουργήσαμε μια συνάρτηση σύμφωνα με την οποία επεξεργαζόμαστε αυτόν τον αριθμό και τον εμφανίζουμε στην κατάλληλη μορφή. Αυτό που κάνουμε αρχικά, είναι να υπολογίσουμε τις μέρες, στη συνέχεια τις ώρες και τελικά τα δευτερόλεπτα αφού πρώτα έχουμε αφαιρέσει από το αρχικό νούμερο τα κλάσματα του δευτερολέπτου. Στη συνέχεια γίνεται ο έλεγχος για την ύπαρξη μηδενικών τιμών των ημερών ή των ωρών. Σε αυτή την περίπτωση αποθηκεύονται μόνο οι μή μηδενικές τιμές της μεταβλητής του χρόνου. Στο τέλος, γίνεται η αποθήκευση σε μια εγγραφή του πίνακα του κάθε αποτελέσματος και η εμφανισή του στην τελική του μορφή.

```
function calcTime($sec) {
$str=array();
$count=strlen($sec);
$sec= substr_replace($sec, '', ($count-2),2);
          $d = intval($sec/86400);
          $sec -= $d*86400;
          $h = intval($sec/3600);
          $sec -= $h*3600;
          $m = intval($sec/60);
          $sec -= $m*60;
        if($d>0)
               $str = $d . ' days '.$h . ' hours ';
        elseif($d==0 && $h>0)
          $str = $h . ' hours '.$m . ' min ';
        elseif($d==0 && $h==0 && $m>0)
         $str = $m . ' min '.$sec . ' sec';
  return $str;
}
```

Μια ακόμα συνάρτηση που δημιουργήσαμε αφορά στη σωστή εμφάνιση του IOS Version της κάθε συσκευής. Για να πάρουμε το IOS Version, καθώς δεν είναι δεδομένο ότι υπάρχει για κάθε συσκευή χρησιμοποιήσαμε το System Description και από εκεί μέσα εμφανίσαμε την πληροφορία που μας ενδιέφερε. Το System Description εμφανίζει πληροφορίες όπως φαίνεται παρακάτω.

```
Cisco IOS Software, C3550 Software (C3550-
IPSERVICESK9-M),Version 12.2(44)SE3, RELEASE
SOFTWARE (fc2) Copyright (c)1986-2008 by
Cisco Systems,Inc. Compiled Mon 29-Sep-08
01:21 by nachen
```

Από όλες αυτές τις πληροφορίες, χρειαζόμασταν μόνο το IOS Version. Για να μπορέσουμε να πάρουμε τη συγκεκριμένη πληροφορία έπρεπε πρώτα να επεξεργαστούμε την έξοδο του System Description. Για το λόγο αυτό δημιουργήθηκε μια συνάρτηση η getVersion η οποία φαίνεται παρακάτω.

```
function getVersion($io){
  $i=0;
    foreach ($io as $sys){
      $sysDfix[$i]  = explode(",␣Version␣", $sys);
      $temp[$i]=$sysDfix[$i][1];
      $i++;
   }
   $i=0;
   foreach($temp as $t){
       $t2[$i]= explode(",␣", $t);
       $io[$i]=$t2[$i][0];
       $i++;
   }
return $io;
}
```

Αυτό στο οποίο μας διευκολύνει η συγκεκριμένη συνάρτηση είναι να εντοπίζουμε την πληροφορία της έκδοσης (Version) και να κόβουμε την πρόταση μέχρι το αμέσως επόμενο κόμμα. Αυτό, λόγώ της μορφής του System Description μας δίνει μόνο την πληροφορία της έκδοσης. Στη συνέχεια, απλώς κόβουμε ότι υπάρχει μετά το Version. Αυτό γίνεται αναζητώντας το πρώτο κόμμα μετά το Version και στη συνέχεια κόβοντας ό,τι υπάρχει μετά από αυτό. Έτσι, ουσιαστικά, έχουμε μόνο την πληροφορία που μας ενδιαφέρει και αυτό είναι το IOS Version στην παρακάτω μορφή:

```
12.2(44)SE3
```

Αφού χρησιμοποιήσαμε διάφορες συναρτήσεις και το SNMP, ώστε να πάρουμε πληροφορίες για το πρωτόκολλο CDP, καταλήξαμε στο να έχουμε ολόκληρο το δίκτυο, και στην προκειμένη περίπτωση εκείνο του TEI. Μέρος του αποτελέσματος φαίνεται στον πίνακα παρακάτω. Να σημειωθεί ότι, χρησιμοποιώντας την ιστοσελίδα σε διαφορετικό δίκτυο, αν αυτό είναι CISCO δίκτυο ή οι συσκευές που υποστηρίζουν το πρωτόκολλο CDP, μπορούμε επίσης να πάρουμε την ίδια πληροφορία.

Device Ip	System Name	Description	System Time	IOS Version	Interfaces
195.251.123.1	it	catalyst355012G	21days 7hours	12.2(44)SE3	27
192.168.16.131	it2	catalyst355012G	3days 14hours	12.2(50)SE3	19
192.168.16.151	it-info210a.it.teithe.gr	catalyst297024TS	4days 21hours	12.2(25)SEE2	30
192.168.16.145	it-info201a	catalyst295048G	4days 21hours	12.1(22)EA4a	52

7 Περιγραφή και ανάλυση μεθόδων

Στο κεφάλαιο αυτό γίνεται περιγραφή των συναρτήσεων και των διαδικασιών που χρησιμοποιήσαμε για την οργάνωση των δεδομένων μας, από πρωτογενή μορφή στο τελικό αποτέλεσμα.

Θα δούμε τις σημαντικότερες διαδικασίες και συναρτήσεις που δημιουργήσαμε, προκειμένου να έχουμε το επιθυμητό αποτέλεσμα για κάθε αρχείο που τις χρησιμοποιεί και θα αναλύουμε κάθε συνάρτηση που χρησιμοποιήσαμε.

ARP Table

Ξεκινάμε με το αρχείο ARP.php και η πρώτη συνάρτηση που θα δούμε είναι η fixMac() η οποία διορθώνει τη MAC address που μας δίνει το SNMP request. Όταν κάνουμε ένα SNMP request για να πάρουμε τη MAC address ενός σταθμού, η διεύθυνση αυτή επιστρέφεται στην ακόλουθη μορφή:

```
0:d:66:fe:ec:6f
```

Βλέπουμε ότι η MAC address παραπάνω, δεν είναι στην απαιτούμενη μορφή, λόγω του ότι η MAC address είναι πάντα σε δεκαεξαδική μορφή 12 χαρακτήρων και όχι λιγότερων όπως φαίνεται εδώ. Η σωστή μορφή εμφάνισης της διεύθυνσης αυτής, θα έπρεπε να είναι ως εξής:

```
00:0d:66:fe:ec:6f
```

Λόγω του προβλήματος αυτού, δημιουργήσαμε τη συνάρτηση fixMac(), η οποία ουσιαστικά διορθώνει το πρόβλημα της εμφάνισης της MAC address. Η συνάρτηση βρίσκεται στο αρχείο Funcs.php για τον λόγω του ότι χρησιμοποιείται και από το αρχείο ifTable.php, το οποίο θα αναλυθεί αργότερα. Παρακάτω βλέπουμε τη συνάρτηση αυτή.

```
function fixMac($macW){
$c=0;
foreach ($macW as $mac){
   $toF = $mac ;
   $toFa=explode(":", $toF);
               for($i=0;$i<=sizeof($toFa); $i++){
                      if (strlen($toFa[$i])==1){
                      $toFa[$i]="0".$toFa[$i];
                      }
               }
$macW[$c]=$toFa[0].":".$toFa[1].":".$toFa[2].":"
.$toFa[3].":".$toFa[4].":".$toFa[5];
$c++;
}
return $macW;
}
```

Η συνάρτηση δουλεύει με τον εξής τρόπο: αρχικά, καλούμε τη συνάρτηση δίνοντας ως παράμετρο τον πίνακα με τα στοιχεία που μας έδωσε το SNMP. Στη

συνέχεια, για κάθε ένα στοιχείο του πίνακα αυτού, γίνεται ξεχωριστά η επεξεργασία. Παίρνουμε κάθε μία MAC, τη χωρίζουμε ανάλογα με τα διαλυτικά και την αποθηκεύουμε σε έναν ξεχωριστό πίνακα, τον $toFa[]. Για αυτόν τον πίνακα, διαβάζουμε την κάθε τιμή του και ελέγχουμε αν το πλήθος της κάθε τιμής του πίνακα είναι ίσο με ένα (1). Στην περίπτωση αυτή, βάζουμε ένα μηδενικό μπροστά στην τιμή, ακολουθούμενο από την τιμή της μεταβλητής. Είναι γνωστό ότι το μηδενικό δεν αλλάζει κάτι στη διεύθυνση όταν βρίσκεται από αριστερά. Οπότε έχουμε έναν πίνακα με ολόκληρη την MAC address χωρισμένη σε δυάδες. Στη συνέχεια αυτό που μένει είναι να ενώσουμε αυτά τα κομμάτια ώστε να πάρουμε την MAC address. Έτσι, βάζουν την MAC address στον πίνακα $macW[] στην θέση $c.

Η μεταβλητή $c είναι ένας μετρητής ο οποίος αλλάζει τιμή κάθε φορά που εισάγουμε μία MAC address στον πίνακα. Η αρχικοποίηση της μεταβλητής αυτής γίνεται στην αρχή της συνάρτησης και χρησιμοποιείται ακόμα και σα δείκτης στον πίνακα $macW[]. Έτσι, στον πίνακα $macW[] απλώς τοποθετούμε με τη σειρά τα στοιχεία του πίνακα $toFa[] ακολουθούμενα από διαλυτικά (:). Τέλος, ο πίνακας επιστρέφεται διορθωμένος και η πορεία του προγράμματος συνεχίζεται κανονικά.

Routing table

Στο αρχείο routTable.php υπάρχουν αρκετές συναρτήσεις. Οι συναρτήσεις εκείνες που χρησιμοποιήθηκαν δημιουργήθηκαν βάση των πληροφοριών που παρέχει ένα routing table (πίνακας δρομολόγησης). Καταρχήν, πρέπει να βλέπουμε τον τύπο της σύνδεσης της συγκεκριμένης συσκευής στην συσκευή με την IP που δώσαμε (routType) . Οι τύποι που μπορούμε να δούμε είναι οι direct, indirect, other και invalid.

- Direct(Connected) – Ο τύπος αυτός σημαίνει ότι η συσκευή στην οποία αναφέρεται η IP διεύθυνση είναι άμεσα συνδεδεμένη με τη συσκευή που χρησιμοποιούμε ή ότι η IP διεύθυνση είναι υποδίκτυο (subnet).

- Indirect – Η συσκευή που αναφέρεται η IP δεν είναι άμεσα συνδεδεμένη. Δηλαδή μεσολαβεί άλλη συσκευή/ες.

- Invalid – Σημαίνει ότι η διαδρομή προς την IP αυτή, δεν είναι έγκυρη.

- Other – Όταν δεν εμπίπτει σε καμιά από της παραπάνω κατηγορίες.

Επιπροσθέτως, ένας πίνακας δρομολόγησης περιέχει τις αντιστοιχίες IP προορισμού(destination IP) – IP μέσω της οποίας θα φτάσουμε εκεί (next hop IP). Ο τρόπος που θα φτάσουμε στον προορισμό που μας ενδιαφέρει, μας παρέχεται από το Interface από το οποίο θα αποστείλουμε τα δεδομένα που θέλουμε. Η απόσταση που βρίσκεται ο προορισμός είναι το metric, το οποίο ορίζεται ως μονάδα μέτρησης απόστασης του προορισμού. Πρέπει επίσης να γνωρίζουμε ποιο πρωτόκολλο χρησιμοποιεί η συσκευή μας για τη δρομολόγηση, για παράδειγμα, RIP, OSPF κτλ. Καθώς επίσης, και η Subnet mask του δικτύου η οποία δηλώνει σε ποιο δίκτυο βρίσκεται η συσκευή προορισμού.

Έτσι, μια εγγραφή του πίνακα δρομολόγησης θα έχει την εξής μορφή:

Route Type	Destination IP	NextHop Address	Outgoing Interface	Metric	Protocol
Indirect	172.16.0.0/20	195.251.240.9	Fast Ethernet 0/0	54016	ciscoIgrp

Η πρώτη συνάρτηση που χρησιμοποιήσαμε είναι η toPrefix(), η οποία υπολογίζει το prefix από το subnet mask. Όταν κάναμε ένα SNMP request για να πάρουμε τη μάσκα υπό δικτύου, μας επέστρεφε την μάσκα στην μορφή: 255.255.0.0. Αυτό που θέλαμε εμείς ήταν να παίρνουμε τη μάσκα σαν ένα δεκαδικό αριθμό, και να τον προσαρτήσουμε ακολούθως, στην IP προορισμού. Έτσι, χρησιμοποιήσαμε τη συνάρτηση toPrefix() την οποία βλέπουμε παρακάτω:

```
function toPrefix($RouteMask){
    $prefix=$RouteMask;
    $prefix=str_split($prefix, 4);
    for ($i=0; $i<=3;$i++)
         $prefix[$i]=decbin($prefix[$i]);
    $pre="$prefix[0]"."$prefix[1]"."$prefix[2]".
"$prefix[3]";
    $prefixArr=str_split($pre);
    $count=0;
    for ($i=0; $i<=31; $i++)
         if ($prefixArr[$i]==1)$count++;
 return $count;
```

Στη συνάρτηση αυτή, δίνεται ως όρισμα η κάθε τιμή της μάσκας υποδικτύου. Αμέσως μετά, χωρίζεται αυτή η τιμή σε τέσσερα μέρη, τεσσάρων χαρακτήρων το κάθε ένα (συμπεριλαμβάνονται και οι τελείες). Στη συνέχεια, για κάθε μέρος του πίνακα χρησιμοποιούμε τη συνάρτηση της PHP decbin() ώστε να μεταφράσουμε τη δεκαδική μορφή σε δυαδική. Ακολούθως, με την str_split() χωρίζουμε όλες τις τιμές της μεταφρασμένης σε δυαδική μορφή μάσκας, σε έναν πίνακα και χρησιμοποιούμε ένα μετρητή ώστε να μετρήσουμε το πλήθος των '1' στον πίνακα αυτόν. Έτσι, επιστρέφουμε την τιμή του μετρητή και έχουμε τη μάσκα υποδικτύου σε δεκαδική μορφή δύο ψηφίων.

Επίσης, όπως ήδη γνωρίζουμε, ένας πίνακας δρομολόγησης περιέχει συνήθως μία διαδρομή με μηδενική IP διεύθυνση. Αυτή η διεύθυνση είναι η Default Route. Οι τιμές στα πεδία Route Type και η IP address για αυτήν τη διεύθυνση που μας επιστρέφει το SNMP, είναι αντίστοιχα η indirect και 0.0.0.0. Οπότε, χρειαζόμαστε μία συνάρτηση η οποία θα βρίσκει αυτήν τη διεύθυνση και θα την ορίζει ως Default Route. Ταυτόχρονα, η συγκεκριμένη συνάρτηση διορθώνει τα ονόματα των τύπων των συνδέσεων που μας επιστρέφει το SNMP request. Ουσιαστικά, η συνάρτηση αυτή κάνει τον πίνακα δρομολόγησης πιο ευανάγνωστο, οπότε την παραθέτουμε στη συνέχεια:

```
function defRoute($destIp, $ipRouteType){
   for($i=0; $i<=sizeof($destIp);$i++){
```

```
        if($destIp[$i]=="0.0.0.0" &&
$ipRouteType[$i]=='indirect')
$ipRouteType[$i]="DefRoute";
        else if($ipRouteType[$i]=='direct')
                $ipRouteType[$i]="Connected";
        else if($ipRouteType[$i]=='indirect')
                $ipRouteType[$i]="Indirect";
      }
return $ipRouteType;
}
```

Όπως βλέπουμε, η συνάρτηση παίρνει δύο ορίσματα. Αυτά είναι η διεύθυνση προορισμού ($destIp) και ο τύπος της σύνδεσης ($ipRouteType) τα οποία αποτελούν όσα χρειαζόμαστε για να καθορίσουμε τις διαδρομές μας. Σε πρώτη φάση, διαβάζουμε τον πίνακα των IP προορισμού προκειμένου να βρούμε τη μηδενική διεύθυνση.Έπειτα, προχωρούμε στη διενέργεια ελέγχου για τη διεύθυνση αυτή. Αν εντοπίσουμε τη διεύθυνση, ακολουθεί ο έλεγχος του πίνακα των τύπων διευθύνσεων ώστε να είναι indirect. Βλέπουμε πως ο δείκτης $i μας δίνει την αντίστοιχη θέση και στους δύο πίνακες. Αν όντως βρεθεί το 0.0.0.0, τότε αλλάζουμε την τιμή του τύπου του σε DefRoute ώστε να δηλώσουμε τη διεύθυνση αυτή ως εξόρισμού διεύθυνση. Σε κάθε έλεγχο, για οποιαδήποτε διεύθυνση βρεθεί, αλλάζουμε απλώς το όνομα του τύπου, ώστε να γίνει πιο ευανάγνωστο. Για παράδειγμα, από direct σε Connected. Τέλος, επιστρέφουμε τον πίνακα με τις διορθωμένες του καταχωρίσεις.

Η επόμενη συνάρτηση έχει να κάνει με τις εξερχόμενες διεπαφές. Τα request μας επιστρέφουν σε ακέραια μορφή τον αριθμό των διεπαφών ενώ εμείς θέλαμε το όνομα της κάθε διεπαφής. Έτσι, χρησιμοποιήσαμε τη συνάρτηση getIfIndex για αυτόν τον σκοπό. Η συνάρτηση αυτή, χρησιμοποιείται και από το ARP table για τον ίδιο ακριβώς λόγο και έτσι τη γενικεύσαμε στο αρχείο Funcs.php ώστε να είναι διαθέσιμη και στα δύο αρχεία.

```
function getIfIndex($IfIndex, $host, $community){
   for($i=0; $i<sizeof($IfIndex);$i++){
$IfIndex[$i] = snmpget("$host","$community",
"1.3.6.1.2.1.2.2.1.2.$IfIndex[$i]");
     }
return $IfIndex;
}
```

Σε αυτήν τη συνάρτηση χρειαζόμαστε τρεις παραμέτρους. Αυτές είναι ο πίνακας με τους ακέραιους που δηλώνουν τη διεπαφή, η IP της συσκευής και το community name ώστε να τα χρησιμοποιήσουμε για να πάρουμε τα ονόματα των διεπαφών. Η λειτουργία είναι απλή. Για κάθε τιμή του πίνακα κάνουμε ένα ξεχωριστό SNMPget request για να επιστραφεί η περιγραφή της κάθε διεπαφής. Έτσι, αντικαθίσταται κάθε ακέραιος αριθμός της διεπαφής με το όνομά της και επιστρέφεται ο πίνακας.

Interface section

Ο πίνακας των διεπαφών (Interface) μας δίνει πληροφορίες που αφορούν τις διεπαφές της συγκεκριμένης συσκευής που θέλουμε. Οι πληροφορίες αυτές αφορούν την

ταχύτητα της διεπαφής, τον τύπο της, την MAC διεύθυνσής της και την κατάσταση στην οποία βρίσκεται. Από τον πίνακα που εμφανίζεται στην αρχή, παίρνουμε γενικές πληροφορίες που αφορούν τη συσκευή ως προς τις διεπαφές της. Τέτοιες πληροφορίες είναι για παράδειγμα, ο αριθμός των διεπαφών της συσκευής, ο αριθμός των ενεργών και των μη ενεργών διεπαφών.

Ένα παράδειγμα μιας διεπαφής είναι η παρακάτω:

Desrc iption	Type	Alias	Mtu	Speed	Admin Status	Oper Status	Last Changed	MAC Address
Gigabit Ethernet 0/1	ethernet Csmacd	sw-it2	1500	1Gbps	up	up	17days 16hours	00:0a:41:0b:4e:01

Αυτό μας δηλώνει την κατάσταση και δίνει τις πληροφορίες της συγκεκριμένης διεπαφής. Το μόνο πρόβλημα που αντιμετωπίσαμε εδώ, ήταν η ταχύτητα της διεπαφής. Η εμφάνιση της ταχύτητας της διεπαφής αρχικά ήταν σε δεκαδικό αριθμό σε bit ανά δευτερόλεπτο. Για να είναι πιο ευανάγνωστο προς τον χρήστη, δημιουργήσαμε μία συνάρτηση η οποία μετατρέπει τα bit ανά δευτερόλεπτο, σε Mbps ή σε Gbps, ανάλογα με την ταχύτητα της διεπαφής. Η συνάρτηση δίνεται παρακάτω:

```
function ifSpeedFix($ifSpeed){
     for ($i=0; $i<count($ifSpeed);$i++){
       if ($ifSpeed[$i]>=10000000 &&
$ifSpeed[$i]<1000000000)
     $ifSpeed[$i]= $ifSpeed[$i]/1000000 ."Mbps";
       elseif ($ifSpeed[$i]>=1000000000 &&
$ifSpeed[$i]<10000000000)
     $ifSpeed[$i]= $ifSpeed[$i]/1000000000 ."Gbps";
         else $ifSpeed[$i]='null';
      }
      return $ifSpeed;
 }
```

Είναι εμφανές ότι στη συνάρτηση αυτή, δίνουμε ως παράμετρο τον πίνακα με τις ταχύτητες των διεπαφών, και γίνονται για κάθε μία τιμή οι υπολογισμοί των ταχυτήτων τους. Έτσι, αν έχουμε μία διεπαφή της οποίας η ταχύτητα βρίσκεται μεταξύ των 10.000.000 και 1.000.000.000 bit ανά δευτερόλεπτο, τότε αυτό που γίνεται είναι να διαιρέσουμε την ταχύτητα αυτή με το 1.000.000 ώστε να μεταφράσουμε την ταχύτητα αυτή σε Mbps. Αντίστοιχα, ακολουθείται η ίδια διαδικασία για μεγαλύτερες ταχύτητες, δηλαδή για Gbps. Τέλος, αν δεν υπάρχει ταχύτητα στη διεπαφή, τότε παίρνει την τιμή null.

Επίσης, χρησιμοποιούμε τη συνάρτηση fixMac() που περιγράψαμε στον πίνακα δρομολόγησης, για να διορθώσουμε την εμφάνιση της MAC διεύθυνσης.

Traffic

Στην καρτέλα Interface Information, υπάρχει ένα κουμπί το οποίο μας δείχνει την εισερχόμενη και εξερχόμενη κίνηση του κάθε interface της συσκευής που βρισκόμαστε. Η συγκεκριμένη σελίδα εμφανίζεται σε νέο παράθυρο και ανανεώνεται ανά πέντε δευτερόλεπτα. Ένα στιγμιότυπο του πίνακα δίνεται παρακάτω:

ID	Desrciption	Alias	Traffic Down	Traffic UP
1	GigabitEthernet0/1	sw-it2	2.69 Mbps	165.14 Kbps
5	GigabitEthernet0/5	it-info211	256 bps	4.52 Kbps

Προκειμένου να πάρουμε την κίνηση ενός οποιουδήποτε interface, θα πρέπει να έχουμε κάποιες συγκρίσιμες τιμές. Οι τιμές αυτές είναι ένας χρόνος και ο αριθμός των δεδομένων που έχουν σταλεί/ληφθεί τη συγκεκριμένη χρονική στιγμή. Αν πάρουμε μετά από κάποιο χρονικό διάστημα τις ίδιες πληροφορίες και αφαιρέσουμε το νέο χρόνο από τον παλιό και τα νέα δεδομένα από τα παλιά, θα έχουμε τη διαφορά που υπήρξε σε αυτό το χρονικό διάστημα. Επίσης, ξέρουμε ότι το πηλίκο των δεδομένων προς τον χρόνο μας δίνει τον αριθμό των δεδομένων ανα το χρόνο. Οπότε για να βρούμε την κίνηση που επιθυμούμε, η συνάρτηση που χρειαζόμαστε είναι η εξής:

Data Per Time = DataNew – DataOld /TimeNew – TimeOld

Οι πληροφορίες που μπορούμε να πάρουμε από τα interface, μας δίνουν τη δυνατότητα να κάνουμε αυτούς τους υπολογισμούς ώστε να καταφέρουμε να πάρουμε την κίνηση των interface. Χρησιμοποιήσαμε τον αριθμό των εισερχόμενων Οκτάδων(Bytes) για αυτόν τον σκοπό, καθώς και την τοπική ώρα του server ώστε να έχουμε το χρόνο που χρειαζόμαστε για να κάνουμε τις πράξεις. Έτσι, σε μια μεταβλητή t_NEW έχουμε το χρόνο και στη μεταβλητή PackIn_NEW έχουμε τον αριθμό των byte. Στα επόμενα πέντε δευτερόλεπτα, η σελίδα ανανεώνεται αυτόματα και οι μεταβλητές αυτές, αποθηκεύονται στο SESSION για να χρησιμοποιηθούν στην επόμενη μέτρηση και τέλος, γίνεται η λήψη των νέων τιμών. Έτσι, έχουμε τέσσερις μεταβλητές.

$PackIn_NEW

$t_NEW

$_SESSION[$host][‘ifTraffic’][‘Traffic][‘t_OLD’]

$_SESSION[$host][‘ifTraffic’][‘Traffic][‘PackIn_OLD’]

Τώρα μπορούμε να κάνουμε τις πράξεις και να πάρουμε την πρώτη τιμή της κίνησης. Η ίδια διαδικασία, γίνεται για τον υπολογισμό της εισερχόμενης αλλά και εξερχόμενης κίνησης. Παρακάτω αναλύεται η διαδικασία που χρησιμοποιήθηκε καθώς και οι συναρτήσεις ώστε να έχουμε το επιθυμητό αποτέλεσμα.

Για να πάρουμε τους χρόνους χρησιμοποιήσαμε τη συνάρτηση της PHP time() που επιστρέφει την ώρα του συστήματος σε δευτερόλεπτα.

Για τη λήψη του εισερχόμενου αριθμού των byte χρησιμοποιήσαμε το OID 1.3.6.1.2.1.2.2.1.10 και για τα εξερχόμενα το OID 1.3.6.1.2.1.2.2.1.16.

Χρησιμοποιείται η συνάρτηση Traffic(), η οποία κάνει την πράξη που είδαμε παραπάνω. Ο κώδικας είναι ο παρακάτω:

```
function Traffic($timeOLD, $timeNEW, $bytesOLD,
$bytesNEW){
$speed=array();
if (isset($timeNEW) && isset($bytesNEW))
   for($i=0;$i<count($bytesOLD);$i++){
            $speed[$i]=0;
          $speed[$i]=
  (($bytesNEW[$i]-$bytesOLD[$i])/($timeNEW - $timeOLD));
      }
return $speed;
```

```
}
```

Στη συνάρτηση αυτή, εφόσον υπάρχουν οι μεταβλητές νέου χρόνου και νέου αριθμού byte, για κάθε τιμή του πίνακα των byte ανά interface γίνεται η πράξη. Στο τέλος επιστρέφεται ο πίνακας με τις τιμές σε bytes per second (Bps).

Το επόμενο βήμα είναι να υπολογιστεί ο αριθμός αυτός σε bps, Kbps και Mbps. Αυτό γίνεται με την παρακάτω συνάρτηση:

```
function ifBps($bps){
       for($i=0;$i<count($bps);$i++){
            $bps[$i]*=8;
            if($bps[$i]<1000){
                    $bps[$i]=round($bps[$i], 2);
                    $bps[$i]=
"<font␣color=#909090>$bps[$i]␣bps</font>";
            }
            elseif ($bps[$i]<1000000){
                    $bps[$i]/=1000;
                    $bps[$i]=round($bps[$i], 2);
                    if($bps[$i]<=1000)
                              $bps[$i]=
"<font␣color=#909090>$bps[$i]␣Kbps</font>";
            }
            else{
                    $bps[$i]/=1000000;
                    $bps[$i]=round($bps[$i], 2);
                    if($bps[$i]>5)
                        $bps[$i]="<b>$bps[$i]␣Mbps</b>";
                    else
                        $bps[$i]="$bps[$i]␣Mbps";
          }
      }
return $bps;
}
```

Η συνάρτηση αυτή, υπολογίζει τις τιμές σε bits και κάνει τις διαιρέσεις ώστε να πάρουν οι τιμές την κατάλληλη μορφή ανάλογα με το μέγεθος που δίνεται και επίσης, «χρωματίζει» τις τιμές, ανάλογα με το εύρος που βρίσκονται ώστε να γίνονται αντιληπτές οι διαφορές από το χρήστη.

Ταξινόμηση

Σε κάποιες περιπτώσεις που έχουμε πίνακες δεδομένων, όπως είναι ο πίνακας ARP και ο πίνακας των Interface, παίρνουμε πολλά δεδομένα και τα εμφανίζουμε. Θεωρείται πρακτικό και λειτουργικό το να μπορούμε να ταξινομήσουμε αυτούς τους πίνακες, ώστε να μπορούμε να δούμε ξεκάθαρα κάποια πράγματα. Για παράδειγμα είναι πιο πρακτικό να δούμε ταξινομημένα τα πρωτόκολλα ενός πίνακα δρομολόγησης ώστε να φανεί ξεκάθαρα ποιά πρωτόκολλα χρησιμοποιούνται. Όπως επίσης θα ήταν πιο εύκολο να εντοπιστούν τα κλειστά/ανοικτά interface αν τα ταξινομούσαμε. Παρακάτω θα αναλυθεί πως πραγματοποιηθήκε η ταξινόμηση.

Γνωρίζουμε ότι για την παρουσίαση των δεδομένων υπό μορφή πίνακα, όπως για παράδειγμα ο πίνακας ARP, βρίσκεται στην μορφή:

Interface Index	Entry Type	MAC Address	IP Address
GigabitEthernet0/10	dynamic	00:0e:38:38:9f:bf	195.251.240.65
GigabitEthernet0/10	static	00:0a:41:0b:4e:00	195.251.240.66
GigabitEthernet0/12	static	00:0a:41:0b:4e:00	195.251.240.69

Σε μία συσκευή δικτύου, τις περισσότερες των περιπτώσεων, δεν υπάρχουν λίγες εγγραφές στον πίνακα όπως το παράδειγμα αλλά πάρα πολλές. Για να εμφανιστεί ο πίνακας ARP, χρειάστηκαν τέσσερις διαφορετικοί πίνακες. Προκειμένου να γίνει ταξινόμηση σε τέτοιου είδους πίνακες, χρειάζεται να χρησιμοποιηθεί μία συνάρτηση της PHP η array_multysort() η οποία χρησιμοποιείται για την ταξινόμηση πολλών πινάκων ταυτόχρονα. Η διαδικασία που χρησιμοποιήθηκε, ήταν να δώσουμε ως παραμέτρους στην συνάρτηση τους πίνακες, με πρώτο αυτόν τον οποίο θέλουμε να γίνει η ταξινόμηση και στην συνέχεια τους υπόλοιπους πίνακες. Επίσης δίνεται ως παράμετρος ο τύπος της ταξινόμησης που θέλουμε και τέλος η σειρά της ταξινόμησης. Έτσι για τον πίνακα ARP η συνάρτηση array_multysort() θα έχει τη μορφή:

```
array_multisort($arpTable2['ifname'],
SORT_ASC, SORT_STRING,
$arpTable2['ipNetToMediaType'],
$arpTable2['ipNetToMediaPhysAddress'],
$arpTable2['ipNetToMediaNetAddress']);
```

Με αυτόν τον τρόπο, ταξινομήθηκαν όλοι οι πίνακες με βάση τον πρώτο πίνακα που δηλώθηκε ως String, κατά αύξουσα σειρά και αντιστοιχήθηκαν οι τιμές των άλλων πινάκων. Για να γίνει η ταξινόμηση και με βάση κάποιον άλλον πίνακα, αυτό που έπρεπε να γίνει, ήταν το να γίνει αλλαγή των παραμέτρων στην συνάρτηση. Έτσι αν για παράδειγμα ζητήσουμε ταξινόμηση με βάση την στήλη με τις MAC διευθύνσεις, τότε θα μπει ως πρώτη παράμετρος η στήλη αυτή και μετά με την σειρά όλες οι άλλες. Προκειμένου να γίνει ταξινόμηση για όλες τις περιπτώσεις, κατά αύξουσα ή φθίνουσα σειρά, χρησιμοποιούνται τόσες περιπτώσεις όσοι και οι πίνακες επί δύο. Δηλαδή για τέσσερις πίνακες έχουμε οκτώ διαφορετικές περιπτώσεις. Λόγω του ότι έχουμε περιπτώσεις, χρησιμοποιήθηκε μια switch ώστε να αλλάζουμε περιπτώσεις. Η μεταβλητή που χρησιμοποιεί η switch είναι η $ss. Αυτή η μεταβλητή δίνεται από την HTML μόλις πατηθεί το κουμπί (βελάκι) στην σελίδα. Αμέσως μετά από το πάτημα αυτό, η μεταβλητή αυτή παίρνει την κατάλληλη τιμή και τίθεται στην διαθεσή μας απο το link του browser.

http://aetos.it.teithe.gr/~dsiaper/myapp/index.php?ip=195.251.123.1
&cn=public&ch=2&ss=3

Έτσι όπως έχει εξηγηθεί προηγουμένως στο κεφάλαιο για την IP και στο κεφάλαιο για την ροή του προγράμματος, γίνεται η λήψη της μεταβλητής αυτής και μπαίνει ως παράμετρος στην συνάρτηση sIndex() η οποία πραγματοποιεί έχει όλα τα κατάλληλα στοιχεία πλέον, ώστε να γίνει η ταξινόμηση. Οι παράμετροι αυτοί είναι η μεταβλητή $ss και ο κάθε πίνακας με τους υποπίνακές του που αποτελούν κάθε φορά την οθόνη που βλέπουμε.

$_SESSION[$host][‘arpTable’]=sIndex($ss, $_SESSION[$host][‘arpTable’]);

8 Δομή ιστοσελίδας

Στο παρακάτω κεφάλαιο θα γίνει μια μικρή περιγραφή της δομής της ιστοσελίδας μας, καθώς θα δούμε ποια αρχεία χρησιμοποιούνται προκειμένου να είναι διαθέσιμη η ιστοσελίδα. Επίσης θα εξηγηθεί το πώς μπορούμε να εγκαταστήσουμε την ιστοσελίδα μας σε ένα http server ενός δικτύου ώστε να μπορούμε να το χρησιμοποιήσουμε στο δίκτυο που θέλουμε.

Ξεκινώντας δίνουμε έναν πίνακα με τα αρχεία που απαιτούνται προκειμένου να λειτουργήσει η ιστοσελίδα.

public_html
Smarty
->Αρχεία που αφορούν το Smarty
myapp
->index.php
->files
-> ciscoDevices.php
-> incSmarty.php
-> ip.php
-> menu.php
-> tpl
->about.tpl
->ARP.tpl
->cisco.tpl
->ciscoNetwork.tpl
->hrDevTable.tpl
->hrStoreTable.tpl
->hrSWRunTable.tpl
->hrSysTable.tpl
->ifTraffic.tpl
->ifTable.tpl
->index.tpl
->ip.tpl
->ref.tpl
->routTable.tpl
->sysTable.tpl
->TCP_IPTable.tpl
-> SNMP
->about.php
->ARP.php
->cisco.php
->ciscoNetwork.php
->Funch.php
->hrDevTable.php
->hrStoreTable.php
->hrSWRunTable.php
->ifTraffic.php
->ifTable.php
->routTable.php
->sysTable.php
->TCP_IPTable.php

-> lib
->CSS
->layersmenu-demo.css
->layerstreemenu.css
->menu
->Τα αρχεία του μενού
->phpsnmp
->Τα αρχεία του phpSnmp
->smarty
-> cache
-> configs
-> templates
-> templates_c
->icons

Με έντονα γράμματα είναι οι φάκελοι και τα υπόλοιπα είναι τα αρχεία με τις επεκτάσεις τους. Η πρόσβαση στην ιστοσελίδα γίνεται από τη διεύθυνση http://localhost/myapp/index.php, όπου localhost η διεύθυνση του server που φιλοξενείται η ιστοσελίδα. Στην προκειμένη περίπτωση είναι ο server του ΑΤΕΙΘ, ο οποίος είναι ο aetos.it.teithe.gr, οπότε το link είναι το

http://aetos.it.teithe.gr/~dsiaper/myapp/index.php.

Για τη λειτουργία της ιστοσελίδας χρειάζεται να έχουμε:

Το πακέτο Smarty, το πακέτο NET-SNMP εγκατεστημένο στον server και το πακέτο phpsnmp. Τα πακέτα Smarty και phpSnmp, υπάρχουν μέσα στην ιστοσελίδα μας, οπότε δεν είναι απαραίτητο να προϋπάρχουν. Προκειμένου να είναι διαθέσιμη στον server πρέπει:

1. Από το CD να γίνει αντιγραφή του φακέλου myapp, στον φάκελο του server όπου έχει οριστεί να φιλοξενεί τις ιστοσελίδες. Για παράδειγμα, αν αυτός ο φάκελος είναι ο " /home/mySites/" τότε πρέπει να γίνει η αντιγραφή μέσα στον φάκελο mySites.

2. Αλλαγή της διαδρομής για τη συμπερίληψη των αρχείων που χρησιμοποιεί η ιστοσελίδα, στην τοπική διεύθυνση φιλοξενίας των ιστοσελίδων. Αυτό γίνεται μέσα από το αρχείο incSmarty.php το οποίο βρίσκεται στο φάκελο "myapp/files/incSmarty.php".

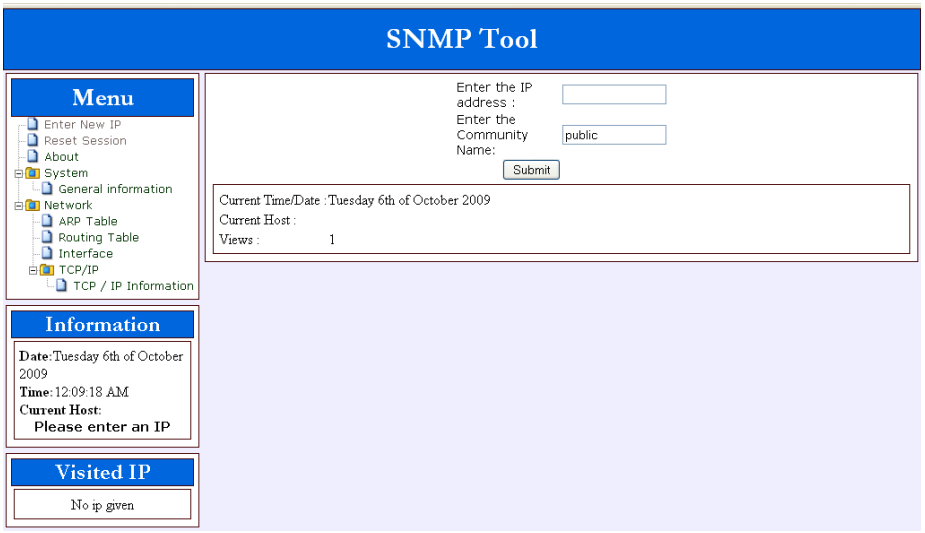
3. Ανοίγουμε το αρχείο με έναν Text Editor και αλλάγή της μεταβλητής $localhost από $localhost='new_server_path' σε $localhost='local_server_path'. Όπου local_server_path, ο φάκελος που φιλοξενούνται οι ιστοσελίδες, για παράδειγμα /home/mySites.

Για την υλοποίηση της πτυχιακής στο δικό μας χώρο, στον server aetos.it.teithe.gr, χρησιμοποιούμε τη διαδρομή

$localhost='/home/student/x0203/dsiaper/public_html'·

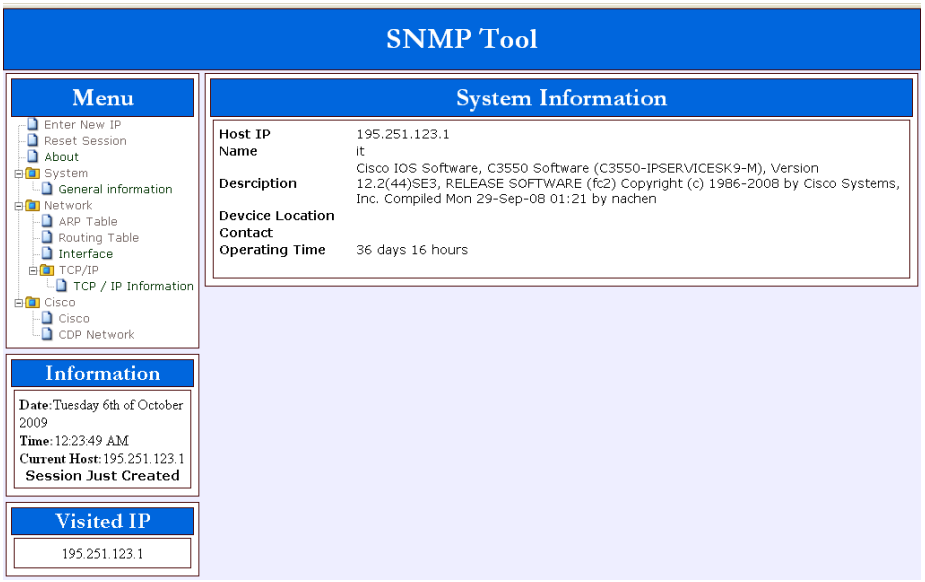
9 Manual

Αρχικά, ανοίγουμε τον browser και μπαίνουμε στη σελίδα στην οποία βρίσκεται η ιστοσελίδα και είναι ο παρακάτω σύνδεσμος http://aetos.it.teithe.gr/~dsiaper/myapp/index.php. Αυτο που εμφανίζεται είναι αρχικά ο τίτλος της σελίδας. Αριστερά βρίσκεται το μενυ πλοήγησης, ένα παράθυρο πληροφοριών και ένας πίνακας με τις IP που έχουμε επισκεφθεί. Στο κέντρο της οθόνης μας ζητούνται η ip address και το community name. Αφού πληκτρολογίσουμε μια έγκυρη IP και το community name της συσκευής που θέλουμε να πάρουμε πληροφορίες στη συνέχεια πατάμε το κουμπί submit.



9.1 System Information

Η αρχική οθόνη που βλέπουμε αφου πατήσουμε το κουμπί submit είναι το system information, το οποίο μας δίνει πληροφορίες για τη συσκευή της οποίας την IP δώσαμε προηγουμένος. Οι πληροφορίες που παίρνουμε είναι :



1. Host ip : Η IP της συσκευής.

2. Name : Το όνομα της συσκευής. Αυτό δίνεται από το διαχειριστή.

3. Description : Μια σύντομη περιγραφή της συσκευής όπως την επιστρέφει το SNMP.

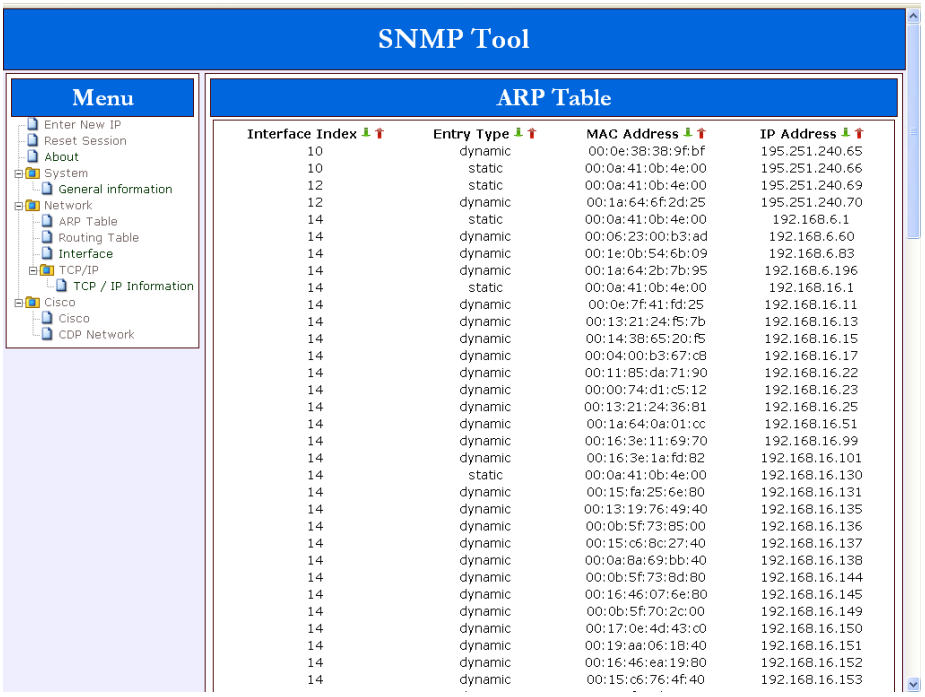
4. Device Location : Τοποθεσία της συσκευής (καθορίζεται από το διαχειριστή).

5. Contact: Τρόπος επικοινωνίας με το διαχειριστή (καθορίζεται από το διαχειριστή).

6. Operating Time: Ο χρόνος λειτουργίας της συσκευής.

Η πρώτη οθόνη είναι ίδια με το General information του menu.

9.2 ARP Table λειτουργεί σε 2 επίπεδο (TCP/IP).



Το ARP Table περιέχει τις αντιστοιχίσεις των Mac address με τις IP addresses.

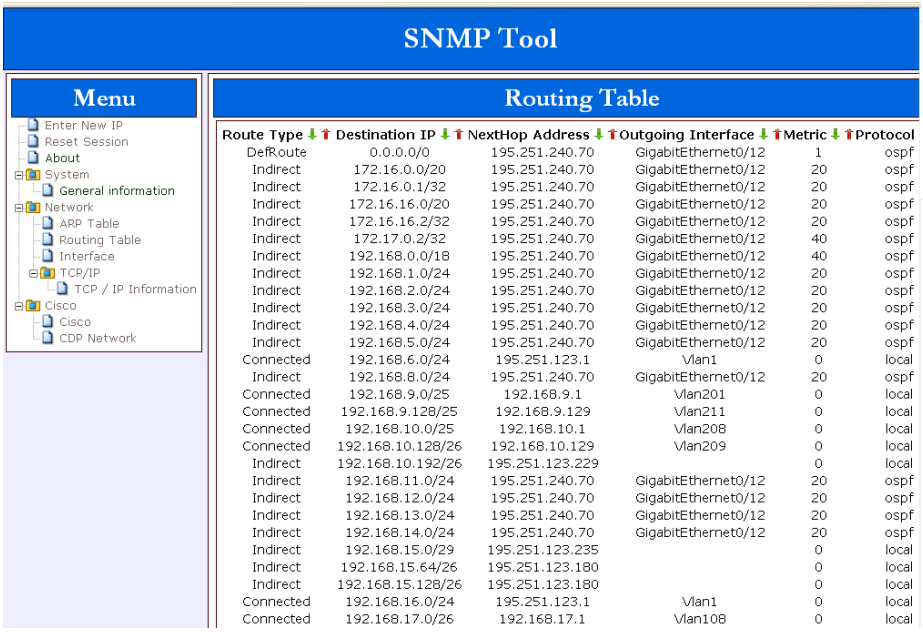
1. Interface Index : Αναγνωριστικό του interface στο οποίο αναφέρεται η εγγραφή στο ARP Table. Η τιμή αυτή καθορίζεται από το MIB.

2. Entry Type : dynamic: καθορίζεται από το πρωτόκολλο. Static: καθορίζεται από το διαχειριστή.

3. MAC Address : Η MAC διεύθυνση της IP στην οποία αντιστοιχεί.

4. IP Address : Η αντίστοιχη IP διεύθυνση.

9.3 Routing Table

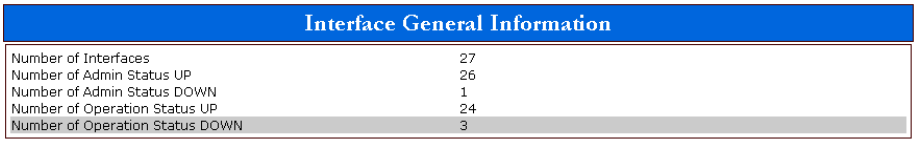


Το Routing Table περιέχει πληροφορίες δρομολόγησης. Δηλαδή το που και το πώς θα σταλθούν τα πακέτα.

1. Route Type : Τύπος διαδρομής
2. Destination IP : Διεύθυνση προορισμού και Subnnet mask.
3. NextHop Address : Ο επόμενος δρομολογητής.
4. Outgoing interface : Εξερχόμενο interface.
5. Metric : Μονάδα μέτρησης της απόστασης ανάλογα με το πρωτόκολλο που χρησιμοποιεί.
6. Protocol : Το πρωτόκολλο δρομολόγησης που χρησιμοποιείται.

9.4 Interface

Το Interface General Information μας δίνει πληροφορίες για το σύνολο των interface της κάθε συσκευής.



1. Number of interfaces : ο αριθμός των interfaces συνολικά.
2. Number of admin Status UP: τα ενεργά interface σε λογικό επίπεδο.
3. Number of admin Status DOWN: τα ανενεργά interface σε λογικό επίπεδο.

4. Number of Operation Status UP: τα ενεργά σε φυσικό επίπεδο.

5. Number of Operation Status Down: τα ανενεργά σε φυσικό επίπεδο.

Εδώ βλέπουμε για κάθε interface τις πληροφορίες ξεχωριστά.

Interface Information

ID	Desrciption	Type	Alias	Mtu	Speed	AdminS	OperS	LastChanged	MACAddress
1	GigabitEthernet0/1	ethernetCsmacd	sw-it2	1500	1Gbps	up	up	17 days 16 hours	00:0a:41:0b:4e:01
2	GigabitEthernet0/2	ethernetCsmacd	sw-it2	1500	1Gbps	up	up	17 days 16 hours	00:0a:41:0b:4e:02
3	GigabitEthernet0/3	ethernetCsmacd	it-info210A	1500	1Gbps	up	up	26 days 8 hours	00:0a:41:0b:4e:03
4	GigabitEthernet0/4	ethernetCsmacd	it-info201	1500	1Gbps	up	up	33 days 6 hours	00:0a:41:0b:4e:04
5	GigabitEthernet0/5	ethernetCsmacd	it-info211	1500	1Gbps	up	up	26 days 8 hours	00:0a:41:0b:4e:05
6	GigabitEthernet0/6	ethernetCsmacd	it-info202A	1500	1Gbps	up	down	33 days 8 hours	00:0a:41:0b:4e:06
7	GigabitEthernet0/7	ethernetCsmacd	it-staff-A	1500	1Gbps	up	up	26 days 8 hours	00:0a:41:0b:4e:07
8	GigabitEthernet0/8	ethernetCsmacd	it-info208A	1500	1Gbps	up	up	26 days 8 hours	00:0a:41:0b:4e:08
9	GigabitEthernet0/9	ethernetCsmacd	it-info301	1500	1Gbps	up	down	33 days 23 hours	00:0a:41:0b:4e:09
10	GigabitEthernet0/10	ethernetCsmacd	uplink-loudias	1500	1Gbps	up	up	23 days 6 hours	00:0a:41:0b:4e:00
11	GigabitEthernet0/11	ethernetCsmacd	hydra-stats	1500	10Mbps	up	down	3 min 17 sec	00:0a:41:0b:4e:0b
12	GigabitEthernet0/12	ethernetCsmacd	uplink-fw	1500	1Gbps	up	up	21 days 6 hours	00:0a:41:0b:4e:00
13	Null0	other	No Alias	1500	4.29Gbps	up	up	1 min 50 sec	
14	Vlan1	propVirtual	No Alias	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
15	Tunnel1	tunnel	den yposthrizei 6to4	1514	null	down	down	3 min 18 sec	
16	Loopback0	softwareLoopback	No Alias	1514	4.29Gbps	up	up	3 min 18 sec	
17	Port-channel1	propVirtual	sw-it2	1500	2Gbps	up	up	17 days 16 hours	00:0a:41:0b:4e:02
18	Vlan108	propVirtual	Lab 108 (autom)	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
19	Vlan201	propVirtual	Lab 201	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
20	Vlan202	propVirtual	Lab 202	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
21	Vlan208	propVirtual	Lab 208	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
22	Vlan209	propVirtual	Lab 209	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
23	Vlan210	propVirtual	Lab 210	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
24	Vlan211	propVirtual	Lab 211	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
25	Vlan301	propVirtual	Lab 301	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
26	Vlan800	propVirtual	Open wireless network - Dedalos	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00
27	Vlan900	propVirtual	VoIP	1500	1Gbps	up	up	3 min 22 sec	00:0a:41:0b:4e:00

1. ID : Το αναγνωριστικό του interface όπως καθορίζεται από το MIB.

2. Description : Η περιγραφή του interface.

3. Type : Ο τύπος του interface.

4. Alias : “ψευδώνυμο” για το interface (καθορίζεται από το διαχειριστή).

5. MTU : Μέγιστο μέγεθος πλαισίου που υποστηρίζει το interface.

6. Speed : Ταχύτητα του interface.

7. AdminStatus : κατάσταση interface σε λογικό επίπεδο.

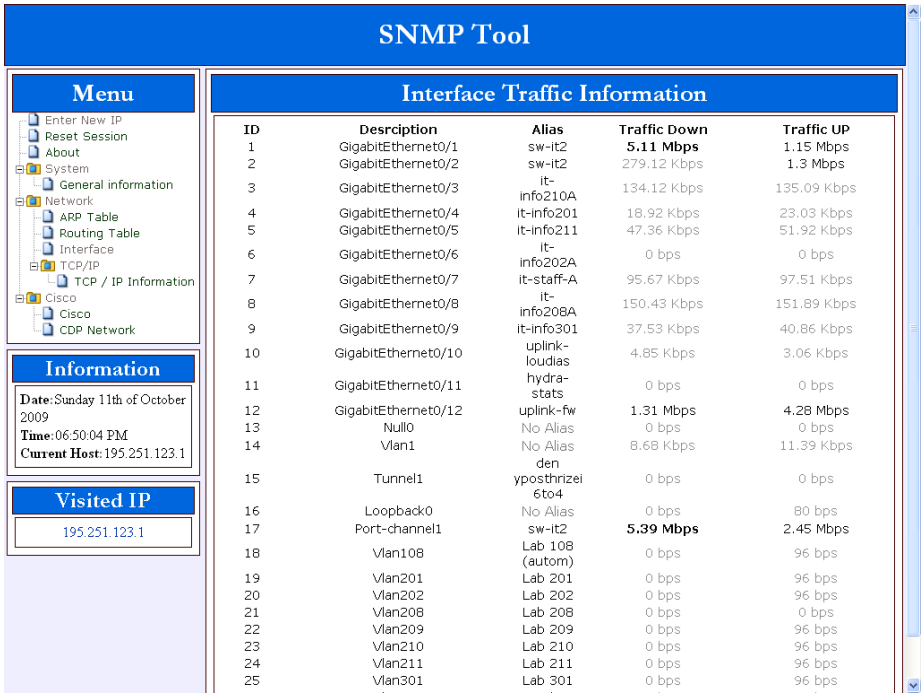
8. OperStatus : κατάσταση interface σε φυσικό επίπεδο.

9. LastChanged : Τελευταία αλλαγή που επήλθε στο interface.

10. MACAddress : Η Mac διεύθυνση του interface.

Με το κουμπί Traffic ανοίγει ένα νέο παράθυρο το interface bitrate information.

9.5 Interface Traffic Information



1. ID : όπως και στα interface.
2. Description : όπως και στα interface.

3. Type : όπως και στα interface.

4. Alias : όπως και στα interface.

5. Traffic down*: ταχύτητα με την οποία δέχεται τα δεδομένα.

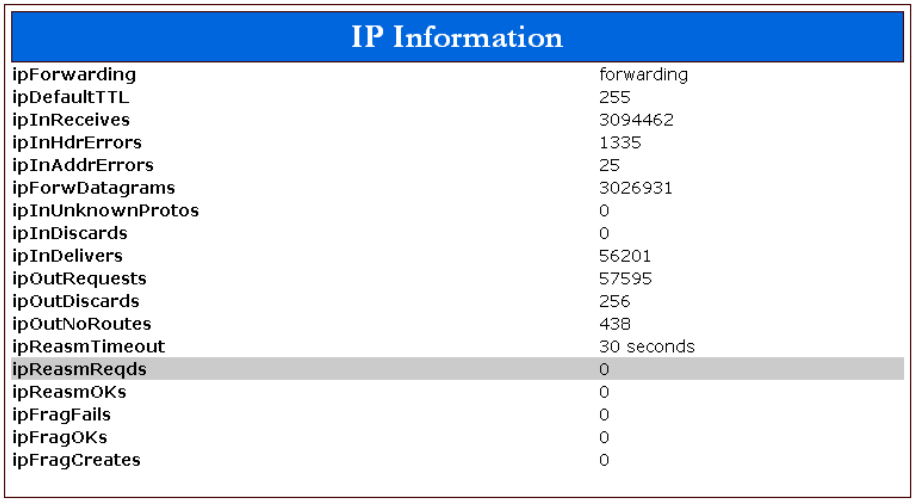
6. Traffic up*: ταχύτητα με την οποία στέλνει τα δεδομένα.

*Η αρχική τιμή είναι μηδενική οπότε και δεν εμφανίζει πληροφορίες. Η σελίδα ανανεώνεται ανά 5 δευτερόλεπτα ώστε να έχουμε σαφή εικόνα για την ταχύτητα.

9.6 TCP/IP

Στην επιλογή αυτήν παίρνουμε πληροφορίες για τα πρωτόκολλα της πλατφόρμας TCP/IP. Αυτά αφορούν τα IP, TCP, UDP, ICMP και SNMP.

9.6.1 IP



IP Information	
ipForwarding	forwarding
ipDefaultTTL	255
ipInReceives	3094462
ipInHdrErrors	1335
ipInAddrErrors	25
ipForwDatagrams	3026931
ipInUnknownProtos	0
ipInDiscards	0
ipInDelivers	56201
ipOutRequests	57595
ipOutDiscards	256
ipOutNoRoutes	438
ipReasmTimeout	30 seconds
ipReasmReqds	0
ipReasmOKs	0
ipFragFails	0
ipFragOKs	0
ipFragCreates	0

1. IPForwarding : Forwarding/ Not Forwarding. Δηλώνει το αν η συσκευή δρομολογεί πακέτα IP ή όχι.

2. IpDefaultTTL : Επιστρέφει την τιμή την οποία δίνει εξορισμού στο πεδίο του IP πακέτου TTL(Time To Live) σε περίπτωση που δεν έχει δωθεί τιμή από το πρωτόκολλο ανώτερου επιπέδου.

3. IpInReceives : Ο αριθμός των εισερχόμενων πακέτων από όλα τα interface της συσκευής, συμπεριλαμβανομένων των εσφαλμένων πακέτων.

4. ipInHdrErrors : Τα πακέτα που απορρίφθηκαν λόγω λανθασμένων πεδίων ελέγχων στην κεφαλίδα του πακέτου IP.

5. ipInAddrErrors : Τα πακέτα που απορρίφθηκαν λόγω IP διεύθυνσης προορισμού διαφορετικής από την συσκευή. Συμπεριλαμβάνονται μη έγκυρες IP όπως 0.0.0.0 και μη συμβατές κλάσεις IP όπως class E.

6. ipForwDatagrams : Το πλήθος των πακέτων όπου δεν έχουν ως τελικό προορισμό την συγκεκριμένη συσκευή αλλά η συσκευή τα προώθησε προς τον τελικό του προορισμό.

7. ipInUnknownProtos : Το πλήθος των πακέτων που απορρίφθηκαν λόγω πρωτοκόλλου ανωτέρου επιπέδου που δεν υποστηρίζεται από την συσκευή.

8. ipInDiscards : Το πλήθος των πακέτων IP που απορρίφθηκαν.

9. ipInDelivers : Το πλήθος των IP πακέτων που παραδώθηκαν επιτυχώς στα πρωτόκολλα ανώτερου επιπέδου, συμπεριλαμβανομένων και των πακέτων ICMP.

10. ipInDelivers : Το πλήθος των IP πακέτων που δόθηκαν από τα πρωτόκολλα ανωτέρων επιπέδων προς αποστολή. Συμπεριλαμβάνονται και τα πακέτα ICMP.

11. ipOutDiscards : Το πλήθος των IP πακέτων που δόθηκαν από τα πρωτόκολλα ανωτέρων επιπέδων προς αποστολή και απορρίφθηκαν.

12. ipOutNoRoutes : Το πλήθος των IP πακέτων που δόθηκαν από τα πρωτόκολλα ανωτέρων επιπέδων προς αποστολή και απορρίφθηκαν, λόγω μη εύρεσης κάποιας διαδρομής.

13. ipReasmTimeout : Ο αριθμός των δευτερολέπτων αναμονής των κατατετμημένων πακέτων πριν την απόρριψή τους.

14. ipReasmReqds : Το πλήθος των IP πακέτων που παραλήφθηκαν και αναμένετε η επαναδημιουργία τους.

15. ipReasmOKs : Το πλήθος των IP πακέτων που αναδημιουργήθηκαν επιτυχώς.

16. ipReasmFails : Το πλήθος των IP πακέτων που δεν αναδημιουργήθηκαν επιτυχώς.

17. ipFragOKs : Το πλήθος των IP πακέτων που κατατμήθηκαν επιτυχώς.

18. ipFragFails : Το πλήθος των IP πακέτων που δεν κατατμήθηκαν επιτυχώς.

19. ipFragCreates : Ο αριθμός των κατατμημένων πακέτων που δημιουργήθηκαν επιτυχώς.

9.6.2 ICMP

ICMP Information			
ICMP Incoming Messages		ICMP Outgoing Messages	
icmpInMsgs	778	icmpOutMsgs	1211
icmpInErrors	0	icmpOutErrors	0
icmpInDestUnreachs	0	icmpOutDestUnreachs	435
icmpInTimeExcds	0	icmpOutTimeExcds	8
icmpInParmProbs	0	icmpOutParmProbs	0
icmpInSrcQuenchs	0	icmpOutSrcQuenchs	0
icmpInRedirects	0	icmpOutRedirects	0
icmpInEchos	778	icmpOutEchos	0
icmpInEchoReps	0	icmpOutEchoReps	778
icmpInTimestamps	0	icmpOutTimestamps	0
icmpInTimestampReps	0	icmpOutTimestampReps	0
icmpInAddrMasks	0	icmpOutAddrMasks	0
icmpInAddrMaskReps	0	icmpOutAddrMaskReps	0

1. icmpInMsgs : Ο συνολικός αριθμός των ICMP μηνυμάτων που ελήφθησαν.
2. icmpInErrors : Ο συνολικός αριθμός των εσφαλμένων ICMP μηνυμάτων που ελήφθησαν.
3. icmpInDestUnreachs : Ο αριθμός των μηνυμάτων ICMP 'Destination Host Unreachable' που ελήφθησαν.
4. icmpInTimeExcds : Ο αριθμός των μηνυμάτων ICMP 'Time Exceeded' που ελήφθησαν.
5. icmpInParmProbs : Ο αριθμός των μηνυμάτων ICMP 'Parameter Problem' που ελήφθησαν.
6. icmpInSrcQuenchs : Ο αριθμός των μηνυμάτων ICMP 'Source Quench' που ελήφθησαν.
7. icmpInRedirects : Ο αριθμός των μηνυμάτων ICMP 'Redirect' που ελήφθησαν.
8. icmpInEchos : Ο αριθμός των μηνυμάτων ICMP 'Echo request' που ελήφθησαν.
9. icmpInEchoReps : Ο αριθμός των μηνυμάτων ICMP 'Echo reply' που ελήφθησαν.
10. icmpInTimestamps : Ο αριθμός των μηνυμάτων ICMP 'Timestamp request' που ελήφθησαν.
11. icmpInTimestampReps : Ο αριθμός των μηνυμάτων ICMP 'Timestamp reply' που ελήφθησαν.
12. icmpInAddrMasks : Ο αριθμός των μηνυμάτων ICMP 'Address Mask request' που ελήφθησαν.
13. icmpInAddrMaskReps : Ο αριθμός των μηνυμάτων ICMP 'Address Mask reply' που ελήφθησαν.
14. icmpOutMsgs : Ο συνολικός αριθμός των ICMP μηνυμάτων που αποστάλθηκαν.
15. icmpOutErrors : Ο συνολικός αριθμός ICMP μηνυμάτων που δεν αποστάλθηκαν λόγω εσωτερικών προβλημάτων του πρωτοκόλλου.

16. icmpOutDestUnreachs : Ο αριθμός των μηνυμάτων ICMP 'Destination Host Unreachable' που αποστάλθηκαν.

17. icmpOutTimeExcds : Ο αριθμός των μηνυμάτων ICMP 'Time Exceeded' που αποστάλθηκαν.

18. icmpOutParmProbs : Ο αριθμός των μηνυμάτων ICMP 'Parameter Problems' που αποστάλθηκαν.

19. icmpOutSrcQuenchs : Ο αριθμός των μηνυμάτων ICMP 'Source Quench' που αποστάλθηκαν.

20. icmpOutRedirects : Ο αριθμός των μηνυμάτων ICMP 'Redirect' που αποστάλθηκαν.

21. icmpOutEchos : Ο αριθμός των μηνυμάτων ICMP 'Echo request' που αποστάλθηκαν.

22. icmpOutEchoReps : Ο αριθμός των μηνυμάτων ICMP 'Echo reply' που αποστάλθηκαν.

23. icmpOutTimestamps : Ο αριθμός των μηνυμάτων ICMP 'Timestamp request' που αποστάλθηκαν.

24. icmpOutTimestampReps : Ο αριθμός των μηνυμάτων ICMP 'Timestamp reply' που αποστάλθηκαν.

25. icmpOutAddrMasks : Ο αριθμός των μηνυμάτων ICMP 'Address Masks request' που αποστάλθηκαν.

26. icmpOutAddrMaskReps : Ο αριθμός των μηνυμάτων ICMP 'Address masks reply' που αποστάλθηκαν.

9.6.3 TCP Active Connections

Στον πίνακα αυτόν παρουσιάζονται οι ενεργές TCP σύνοδοι.

TCP Active Connections				
Local Address	Local Port	Remote Address	Remote Port	Connection State
0.0.0.0	22	0.0.0.0	0	listen
0.0.0.0	25	0.0.0.0	0	listen
0.0.0.0	111	0.0.0.0	0	listen
0.0.0.0	587	0.0.0.0	0	listen
0.0.0.0	2401	0.0.0.0	0	listen
0.0.0.0	3493	0.0.0.0	0	listen
0.0.0.0	4949	0.0.0.0	0	listen
0.0.0.0	5222	0.0.0.0	0	listen
0.0.0.0	5269	0.0.0.0	0	listen
0.0.0.0	5432	0.0.0.0	0	listen
0.0.0.0	9999	0.0.0.0	0	listen
0.0.0.0	50110	0.0.0.0	0	listen
127.0.0.1	587	127.0.0.1	59866	timeWait
127.0.0.1	631	0.0.0.0	0	listen
127.0.0.1	3306	0.0.0.0	0	listen
127.0.0.1	3493	127.0.0.1	36004	established
127.0.0.1	36004	127.0.0.1	3493	established
195.251.123.236	3493	195.251.123.232	56314	established
195.251.123.236	3493	195.251.123.234	42433	established
195.251.123.236	3493	195.251.123.235	43408	established
195.251.123.236	3493	195.251.123.236	57036	timeWait
195.251.123.236	3493	195.251.123.236	57037	timeWait
195.251.123.236	3493	195.251.123.236	57038	timeWait
195.251.123.236	3493	195.251.123.236	57039	timeWait
195.251.123.236	3493	195.251.123.236	57040	timeWait
195.251.123.236	3493	195.251.123.240	55915	established
195.251.123.236	3493	195.251.123.241	1032	established

1. Local Address : Η τοπική IP διεύθυνση για την TCP σύνοδο

2. Local Port : Η τοπική πόρτα που χρησιμοποιείται για την TCP σύνοδο

3. Remote Address : Η απομακρυσμένη IP διεύθυνση που χρησιμοποιείται για την σύνοδο.

4. Remote Port : Η απομακρυσμένη πόρτα που χρησιμοποιείται για την TCP σύνοδο

5. Connection State : Η κατάσταση της συγκεκριμένης συνόδου.

6. tcpInErrs : Το πλήθος των εισερχόμενων τμημάτων που ελήφθησαν με σφάλματα.

7. tcpOutRsts : Ο αριθμός των εξερχόμενων τμημάτων που εστάλεισαν με το bit RST στην κεφαλίδα TCP ενεργό.

9.6.4 TCP Algorithm

Στον πίνακα αυτόν εμφανίζονται πληροφορίες που αφορούν τον αλγόριθμο του TCP.

TCP Information	
tcpRtoAlgorithm	other
tcpRtoMin	200 milliseconds
tcpRtoMax	120000 milliseconds
tcpActiveOpens	-1
tcpPassiveOpens	1386
tcpAttemptFails	2557
tcpEstabResets	34
tcpCurrEstab	211
tcpInSegs	16
tcpOutSegs	96879
tcpRetransSegs	334

1. tcpRtoAlgorithm : Ο αλγόριθμος που χρησιμοποιείται για να καθοριστεί ο χρόνος επαναποστολής των τμημάτων.

2. tcpRtoMin : Η μικρότερη επιτρεπτή τιμή για τον χρόνο επαναποστολής των τμημάτων ανάλογα με τον αλγόριθμο που χρησιμοποιείται.

3. tcpRtoMax : Η μεγαλύτερη επιτρεπτή τιμή για τον χρόνο επαναποστολής των τμημάτων ανάλογα με τον αλγόριθμο που χρησιμοποιείται.

4. tcpMaxConn : Το μεγιστο επιτρεπτό όριο των ενεργών συνόδων που επιτρέπεται σε αυτόν τον σταθμό. Αν ο αριθμός αυτός υπολογίζεται δυναμικά, τότε η τιμή αυτή θα είναι -1.

5. tcpActiveOpens : Το πλήθος των ενεργών ανοιγμάτων που έχει κάνει αυτός ο σταθμός. Ορίζεται ως το πλήθος των τμημάτων που έχει στείλει με ενεργό το bit SYN-SENT από την καταστση CLOSED.

6. tcpPassiveOpens : Το πλήθος των παθητικών ανοιγμάτων που έχει κάνει αυτός ο σταθμός. Ορίζεται ως το πλήθος των τμημάτων που έχει στείλει με ενεργό το bit SYN-RCVD από την καταστση LISTEN.

7. tcpAttemptFails : Ο αριθμός που δηλώνει το πόσες φορές έχει αλλάξει την κατάσταση σε CLOSED από την κατάσταση SYN-SENT ή από την κατάσταση SYN-RCVD.

8. tcpEstabResets : Ο αριθμός των αλλαγών κατάστασης από ESTABLISHED ή CLOSED-WAIT σε CLOSED.

9. tcpCurrEstab : Ο αριθμός των συνόδων που βρίσκονται σε κατάσταση ESTASBLISHED ή CLOSED-WAIT την συγκεκριμένη χρονική στιγμή.

10. tcpInSegs : Το πλήθος των εισερχόμενων τμημάτων που ελήφθησαν, ακόμα και αυτά που ελήφθησαν με σφάλμα.

11. tcpOutSegs : Το πλήθος των εξερχόμενων τμημάτων.

12. tcpRetransSegs : Το πλήθος των τμημάτων που έχουν αναμεταδοθεί.

9.6.5 UDP

Ο πίνακας αυτός εμφανίζει πληροφορίες για το πρωτόκολλο UDP.

UDP Information	
udpInDatagrams	479750
udpOutDatagrams	480883
udpNoPorts	39
udpInErrors	0

UDP Information

1. udpInDatagrams : Ο αριθμός των UPD πακέτων που έχουν ληφθεί.

2. udpOutDatagrams : Ο αριθμός των UPD πακέτων που έχουν αποσταλεί.

3. udpNoPorts : Το πλήθος των UDP πακέτων που έχουν παραληφθεί και δεν υπήρχε κάποια εφαρμογή στην θύρα που έχει γίνει η παράδοσή τους.

4. udpInErrors : Το πλήθος των UDP πακέτων που δεν έχουν παραληφθεί σωστά για άλλους λόγους από την έλλειψη κατάλληλης εφαρμογής.

UDP Table

UDP Table	
Local Address	Local Port
0.0.0.0	67
0.0.0.0	69
0.0.0.0	111
0.0.0.0	137
0.0.0.0	138
0.0.0.0	161
0.0.0.0	514
0.0.0.0	631
0.0.0.0	880
0.0.0.0	35449
0.0.0.0	36868
0.0.0.0	52108
0.0.0.0	53710
195.251.123.183	137
195.251.123.183	138
195.251.123.183	3478
195.251.123.183	3479
195.251.123.236	137
195.251.123.236	138
195.251.123.236	500
195.251.123.236	3478
195.251.123.236	3479
195.251.123.236	4500

1. udpLocalAddress : Η τοπική IP διεύθυνση αποδοχής UDP πακέτων.

2. udpLocalPort : Η τοπική θύρα αποδοχής UDP πακέτων.

SNMP

SNMP Information

SNMP Incoming Messages		SNMP Outgoing Messages	
SNMP Incoming Packets	2526	SNMP Outgoing Packets	2545
snmpInBadVersions	0	snmpOutTooBigs	0
Incoming Bad Community Names	0	snmpOutNoSuchNames	3
snmpInBadCommunityUses	0	snmpOutBadValues	0
snmpInASNParseErrs	0	snmpOutGenErrs	0
snmpInTooBigs	0	Get Requests	0
snmpInNoSuchNames	0	Get Next Requests	0
snmpInBadValues	0	Set Requests	0
snmpInReadOnlys	0	Get Responses	2553
snmpInGenErrs	0	snmpOutTraps	0
snmpInTotalReqVars	2532		
snmpInTotalSetVars	0		
Get Requests	337		
Get Nexts	2201		
Set Requests	0		
Get Responses	0		
snmpInTraps	0		
snmpEnableAuthenTraps	disabled		
snmpSilentDrops	0		
snmpProxyDrops	0		

1. snmpInPkts : Το πλήθος των εισερχόμενων SNMP πακέτων.

2. snmpInBadVersions : Το πλήθος των εισερχόμενων SNMP μηνυμάτων που παραλήφθηκαν με έκδοση πρωτοκόλλου που δεν υποστιρίζεται από την συγκεκριμένη συσκευή.

3. snmpInBadCommunityNames : Το πλήθος των εισερχόμενων SNMP μηνυμάτων που παραλήφθηκαν με λάθος community name.

4. snmpInBadCommunityUses : Το πλήθος των εισερχόμενων SNMP μηνυμάτων που παραλήφθηκαν και το community name δεν επιτρέπει να εκτελεστεί η λειτουργία που ζητείται.

5. snmpInTooBigs : Το πλήθος των εισερχόμενων SNMP πακέτων τα οποία επέστρεψαν μήνυμα σφάλματος 'tooBig' s

6. snmpInNoSuchNames : Το πλήθος των εισερχόμενων SNMP πακέτων τα οποία επέστρεψαν μήνυμα σφάλματος 'NoSuchName'.

7. snmpInBadValues : Το πλήθος των εισερχόμενων SNMP πακέτων τα οποία επέστρεψαν μήνυμα σφάλματος 'badValues'.

8. snmpInReadOnlys : Το πλήθος των εισερχόμενων SNMP πακέτων τα οποία επέστρεψαν μήνυμα σφάλματος 'readOnly'. Το αποτέλεσμα αυτό οδηγεί σε συμπεράσματα λάθους υλοποίησης του πρωτοκόλλου στην συσκευή.

9. snmpInGenErrs : Το πλήθος των εισερχόμενων SNMP πακέτων τα οποία επέστρεψαν μήνυμα σφάλματος 'genErr'.

10. snmpInTotalReqVars : Το πλήθος των πακέτων SNMP που παραλήφθηκαν σωστά ως αποτέλεσμα των απεσταλμένων SNMPget request ή SNMPgetNext request.

11. snmpInTotalSetVars : Το πλήθος των MIB αντικειμένων που άλλαξαν, ως αποτέλεσμα των SNMPset requests.

12. snmpInGetRequests : Το πλήθος των εισερχόμενων SNMPget requests που παρελήφθησαν και επεξεργάστηκαν από την συσκευή.

13. snmpInGetNexts : Το πλήθος των εισερχόμενων SNMPgetNext requests που παρελήφθησαν και επεξεργάστηκαν από την συσκευή.

14. snmpInSetRequests : Το πλήθος των εισερχόμενων SNMPset requests που παρελήφθησαν και επεξεργάστηκαν από την συσκευή.

15. snmpInGetResponses : Το πλήθος των εισερχόμενων SNMPget response που παρελήφθησαν και επεξεργάστηκαν από την συσκευή.

16. snmpInTraps : Το πλήθος των εισερχόμενων SNMPtrap που παρελήφθησαν και επεξεργάστηκαν από την συσκευή.

17. snmpOutPkts : Το πλήθος των εξερχόμενων SNMP πακέτων.

18. snmpOutTooBigs : Το πλήθος των εξερχόμενων SNMP πακέτων τα οποία στέλνονται με μήνυμα σφάλματος 'tooBig'.

19. snmpOutNoSuchNames : Το πλήθος των εξερχόμενων SNMP πακέτων τα οποία στέλνονται με μήνυμα σφάλματος 'noSuchName'.

20. snmpOutBadValues : Το πλήθος των εξερχόμενων SNMP πακέτων τα οποία στέλνονται με μήνυμα σφάλματος 'badValues'.

21. snmpOutGenErrs : Το πλήθος των εξερχόμενων SNMP πακέτων τα οποία στέλνονται με μήνυμα σφάλματος 'genErr'.

22. snmpOutGetRequests : Το πλήθος των εξερχόμενων SNMPget requests που δημιουργήθηκαν και εστάλησαν από την συσκευή.

23. snmpOutGetNexts : Το πλήθος των εξερχόμενων SNMPgetNext requests που δημιουργήθηκαν και εστάλησαν από την συσκευή.

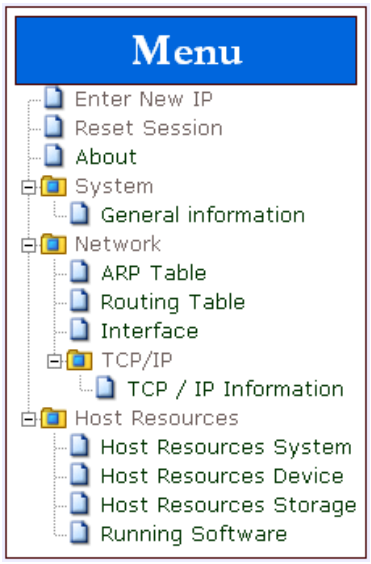
24. snmpOutSetRequests : Το πλήθος των εξερχόμενων SNMPset requests που δημιουργήθηκαν και εστάλησαν από την συσκευή.

25. snmpOutGetResponses : Το πλήθος των εξερχόμενων SNMPget response που δημιουργήθηκαν και εστάλησαν από την συσκευή.

26. snmpOutTraps : Το πλήθος των εξερχόμενων SNMPtrap που δημιουργήθηκαν και εστάλησαν από την συσκευή.

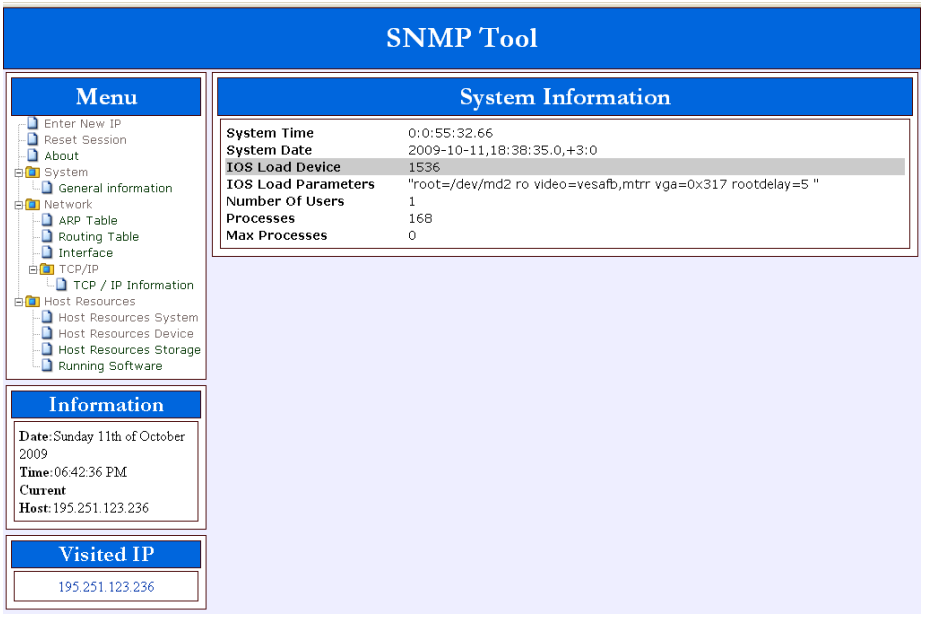
9.7 Host Resources

Μας δίνει πληροφορίες για την τερματική συσκευή της οποίας έχουμε δώσει την IP Address.



9.7.1 Host Resources System

Πληροφορίες σχετικά με το σύστημα.



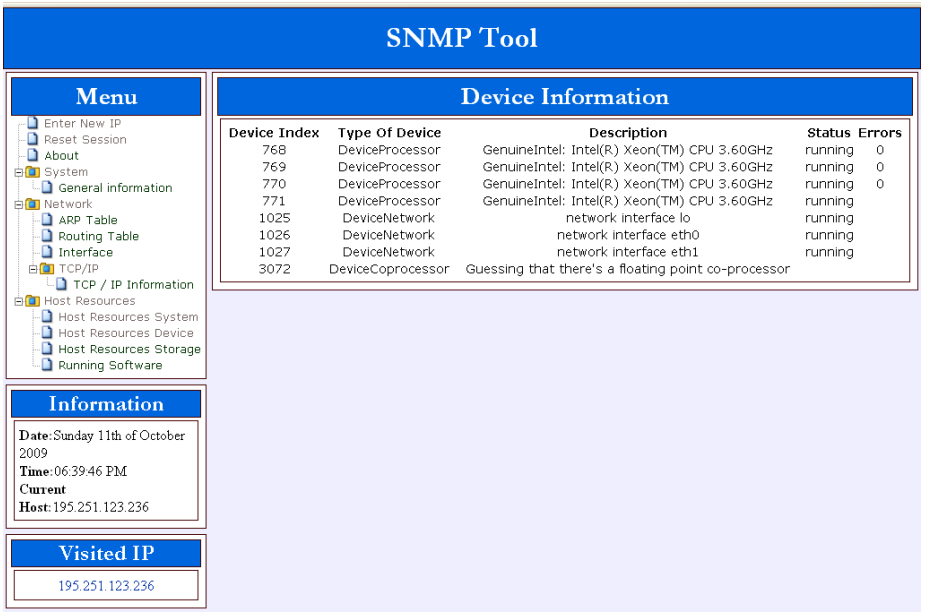
1. System Time: Η ώρα του συστήματος
2. System Date : Η ημερομηνία του συστήματος
3. IOS Load Device : Ακέραιος αριθμός που δηλώνει την συσκευή από την οποία ζητείται να φορτωθεί το ΛΣ
4. IOS Load Parameters :Οι παράμετροι που θα χρησιμοποιηθούν κατά την φόρτωση του ΛΣ.

5. Number Of Users : Ο αριθμός των χρηστών

6. Processes :Ο αριθμός των διεργασιών που τρέχουν στο σύστημα την συγκεκριμένη χρονική στιγμή

7. Max Processes : Ο αριθμός των μέγιστων διεργασιών για το συγκεκριμένο σύστημα. Εξορισμού 0.

9.7.2 Host Resources Device



Στον πίνακα αυτόν μας δίνονται πληροφορίες σχετικά με τις συσκευές του συστήματος.

1. Device Index : Το αναγνωριστικό της συσκευής.

2. Type Of Device : Ο τύπος της συσκευής.

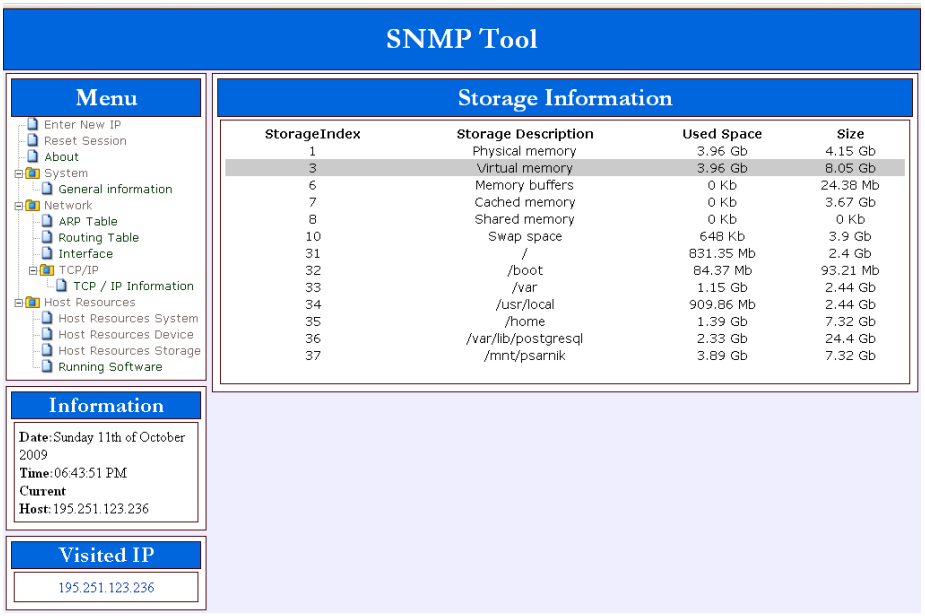
3. Description : Περιγραφή της συσκευής.

4. Status : Κατάσταση της συσκευής.

5. Errors : Ο αριθμός των σφαλμάτων της συγκεκριμένης συσκευής.

9.7.3 Storage Information

Στον πίνακα αυτόν, μας δίνονται πληροφορίες σχετικά με τις μνήμες και με τα συστήματα αρχείων της τερματικής συσκευής.



1. Index : Το αναγνωριστικό της συσκευής.

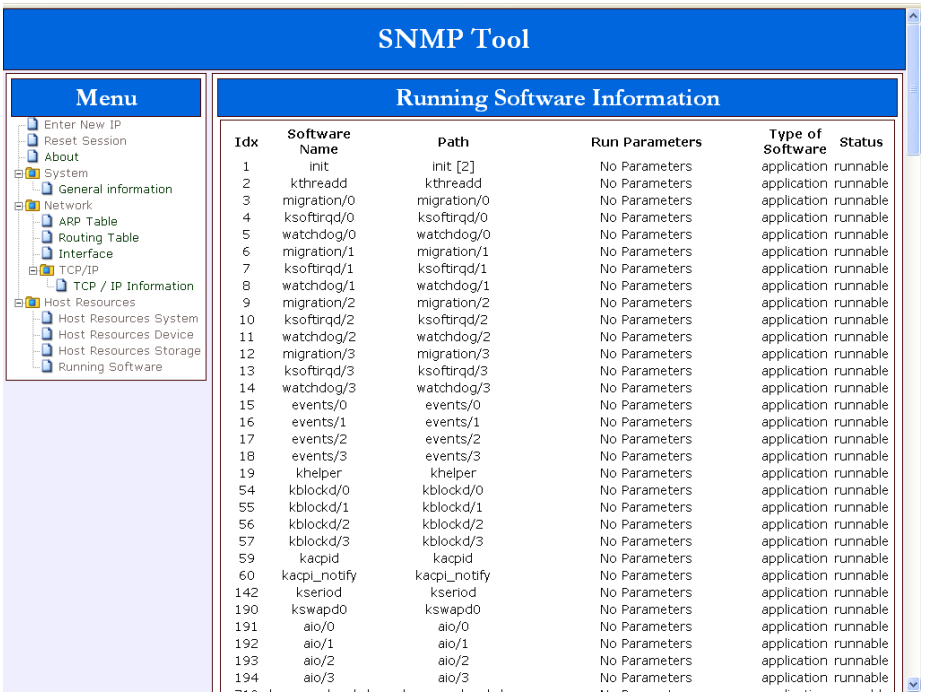
2. Description : Περιγραφή του τύπου της συσκευής και/ή το όνομα του.

3. Used space : χώρος που χρησιμοποιείται.

4. Size : Συνολικό μέγεθος

9.7.4 Running Software Information

Εμφανίζει το τρέχον λογισμικό του συστήματος.



1. Index : Αναγνωριστικό του λογισμικού.

2. Software Name : Η ονομασία του λογισμικού

3. Software Run Path : Η τοποθεσία από την οποία φορτώθηκε το λογισμικό

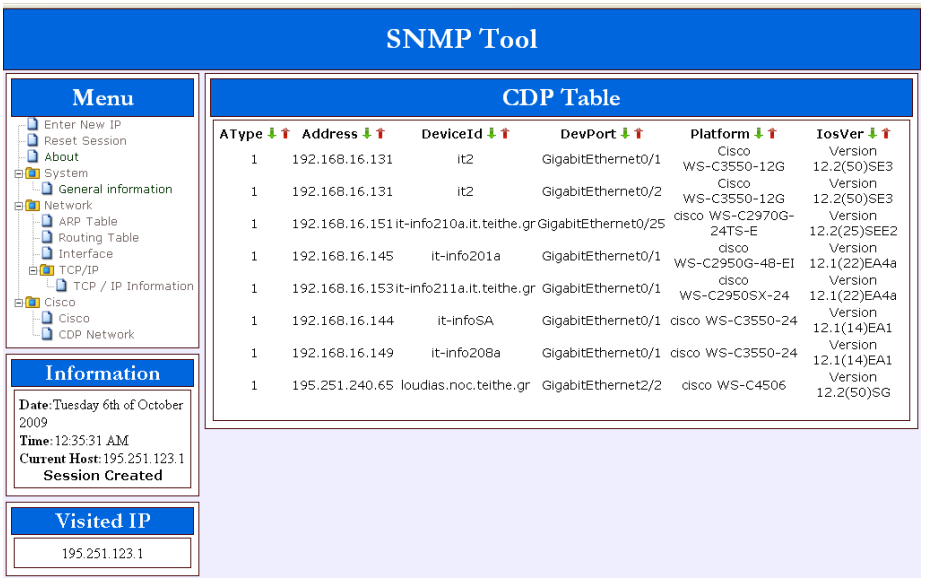
4. Parameters : Οι παράμετροι που δόθηκαν κατά την φόρτωση του λογισμικού.

5. Software Type : Ο τύπος του λογισμικού.

6. Run Status : Η κατάσταση του λογισμικού.

9.8 CISCO

Σε αυτόν τον πίνακα εμφανίζονται πληροφορίες σχετικά με το πρωτόκολλο CDP (Cisco Discovery Protocol). Οι πληροφορίες αυτές αναφέρονται στις άμεσα συνδεδεμένες συσκευές, από την συσκευή της οποίας δώσαμε την IP. Οι πληροφορίες είναι διαθέσιμες για CISCO συσκευές ή και για συσκευές που υποστηρίζουν το πρωτόκολλο CDP.



1. Address : Η IP διεύθυνση της άμεσα συνδεδεμένης συσκευής.

2. DeviceId : Το όνομα της συσκευής όπως καθορίστηκε από τον διαχειριστή.

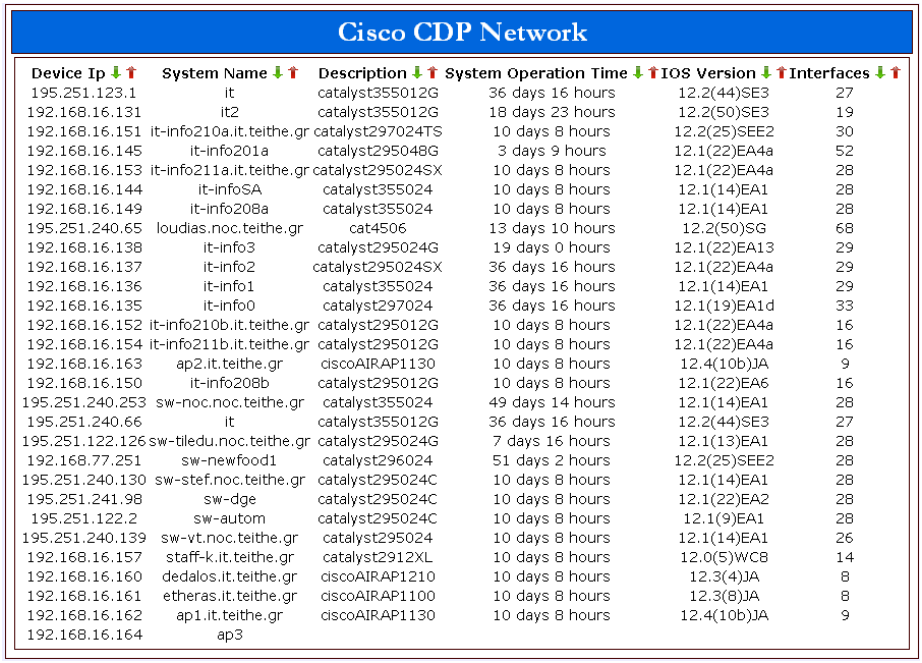
3. DevPort : Το Interface της άμεσα συνδεδεμένης συσκευής που είναι σε επικοινωνία η συσκευή μας.

4. Platform : Ο τύπος της συσκευής.

5. IOS Version : Η έκδοση του Λειτουργικού Συστήματος της συσκευής.

9.8.1 CDP network

Σε αυτόν τον πίνακα παίρνουμε πληροφορίες για ολόκληρο το CDP δίκτυο της συσκευής που δώσαμε την IP.



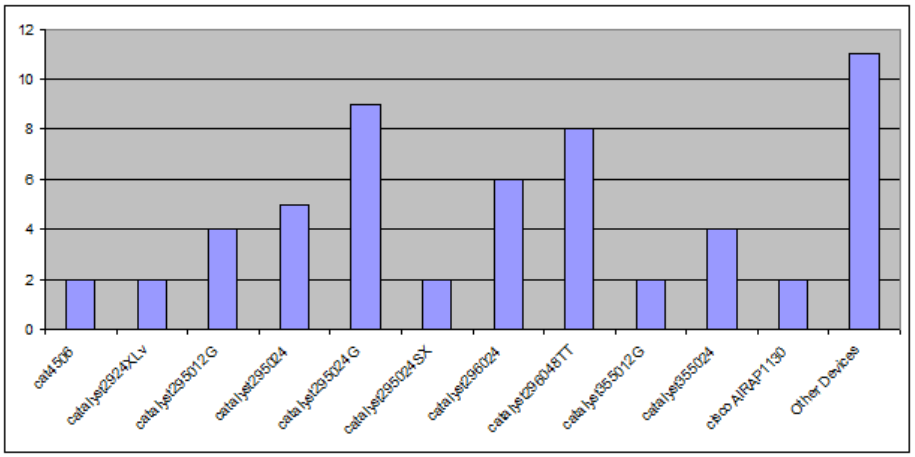
Cisco CDP Network

Device Ip	System Name	Description	System Operation Time	IOS Version	Interfaces
195.251.123.1	it	catalyst355012G	36 days 16 hours	12.2(44)SE3	27
192.168.16.131	it2	catalyst355012G	18 days 23 hours	12.2(50)SE3	19
192.168.16.151	it-info210a.it.teithe.gr	catalyst297024TS	10 days 8 hours	12.2(25)SEE2	30
192.168.16.145	it-info201a	catalyst295048G	3 days 9 hours	12.1(22)EA4a	52
192.168.16.153	it-info211a.it.teithe.gr	catalyst295024SX	10 days 8 hours	12.1(22)EA4a	28
192.168.16.144	it-infoSA	catalyst355024	10 days 8 hours	12.1(14)EA1	28
192.168.16.149	it-info208a	catalyst355024	10 days 8 hours	12.1(14)EA1	28
195.251.240.65	loudias.noc.teithe.gr	cat4506	13 days 10 hours	12.2(50)SG	68
192.168.16.138	it-info3	catalyst295024G	19 days 0 hours	12.1(22)EA13	29
192.168.16.137	it-info2	catalyst295024SX	36 days 16 hours	12.1(22)EA4a	29
192.168.16.136	it-info1	catalyst355024	36 days 16 hours	12.1(14)EA1	29
192.168.16.135	it-info0	catalyst297024	36 days 16 hours	12.1(19)EA1d	33
192.168.16.152	it-info210b.it.teithe.gr	catalyst295012G	10 days 8 hours	12.1(22)EA4a	16
192.168.16.154	it-info211b.it.teithe.gr	catalyst295012G	10 days 8 hours	12.1(22)EA4a	16
192.168.16.163	ap2.it.teithe.gr	ciscoAIRAP1130	10 days 8 hours	12.4(10b)JA	9
192.168.16.150	it-info208b	catalyst295012G	10 days 8 hours	12.1(22)EA6	16
195.251.240.253	sw-noc.noc.teithe.gr	catalyst355024	49 days 14 hours	12.1(14)EA1	28
195.251.240.66	it	catalyst355012G	36 days 16 hours	12.2(44)SE3	27
195.251.122.126	sw-tiledu.noc.teithe.gr	catalyst295024G	7 days 16 hours	12.1(13)EA1	28
192.168.77.251	sw-newfood1	catalyst296024	51 days 2 hours	12.2(25)SEE2	28
195.251.240.130	sw-stef.noc.teithe.gr	catalyst295024C	10 days 8 hours	12.1(14)EA1	28
195.251.241.98	sw-dge	catalyst295024C	10 days 8 hours	12.1(22)EA2	28
195.251.122.2	sw-autom	catalyst295024C	10 days 8 hours	12.1(9)EA1	28
195.251.240.139	sw-vt.noc.teithe.gr	catalyst295024	10 days 8 hours	12.1(14)EA1	26
192.168.16.157	staff-k.it.teithe.gr	catalyst2912XL	10 days 8 hours	12.0(5)WC8	14
192.168.16.160	dedalos.it.teithe.gr	ciscoAIRAP1210	10 days 8 hours	12.3(4)JA	8
192.168.16.161	etheras.it.teithe.gr	ciscoAIRAP1100	10 days 8 hours	12.3(8)JA	8
192.168.16.162	ap1.it.teithe.gr	ciscoAIRAP1130	10 days 8 hours	12.4(10b)JA	9
192.168.16.164	ap3				

1. Device IP : Οι διευθύνσεις IP των συσκευών του δικτύου
2. System Name : Τα ονόματα των συσκευών όπως καθορίστικαν από τον διαχειριστή.
3. Description : Ο τύπος των συσκευών
4. System Operation Time : Η ώρα λειτουργίας των συσκευών
5. IOS Version : Η έκδοση των λειτουργικών συστημάτων των συσκευών.
6. Interfaces : Ο αριθμός των interfaces της κάθε συσκευής του δικτύου.

10 Συμπεράσματα

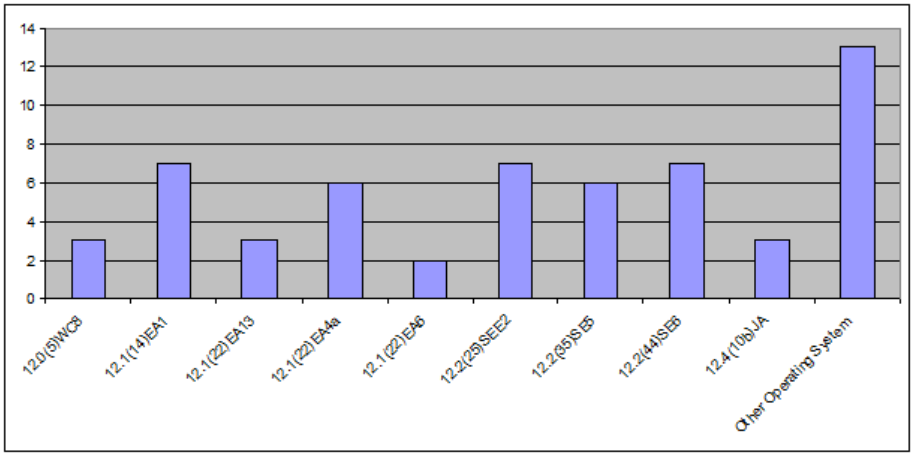
Μέσω του CDP Network πήραμε πληροφορίες για ολόκληρο το δίκτυο του TEI. Δίνοντας μια οποιαδήποτε IP του TEI και επιλέγοντας απο το μενού το CDP Network μπορούμε να βρούμε όλες τις συμβατές με CDP συσκευές του TEI. Έτσι μπορούμε να βγάλουμε συμπεράσματα σχετικά με το δίκτυο. Τρέξαμε την εφαρμογή κάποια τυχαία χρονική στιγμή και οι μετρήσεις έδειξαν τα παρακάτω:



Την Δευτέρα στις 8 Μαρτίου το απόγευμα βρέθηκαν 57 δικτυακές συσκευές σε λειτουργία οι οποίες υποστηρίζουν πρωτόκολλο SNMP. Συγκεκριμένα όλες οι συσκευές φαίνονται στον παρακάτω πίνακα.

Συσκευές	**Πλήθος**	**Συσκευές**	**Πλήθος**
cat4506	2	catalyst2960G8TC	1
catalyst2924XLv	2	catalyst297024	1
catalyst295012G	4	catalyst297024TS	1
catalyst295024	5	catalyst6kMsfc2	1
catalyst295024G	9	cisco2509	1
catalyst295024SX	2	cisco3662Ac	1
catalyst296024	6	ciscoAIRAP1100	1
catalyst296048TT	8	ciscoAIRAP1210	1
catalyst355012G	2	ciscoAIRAP1240	1
catalyst355024	4	catalyst295048G	1
ciscoAIRAP1130	2	catalyst2912XL	1

Την ίδια χρονική στιγμή αναλύθηκαν και οι πληροφορίες σχετικά με το IOS Version των συσκευών αυτών. Στο παρακάτω διάγραμμα φαίνεται οτι οι περισσότερες συσκευές χρησιμοποιούν τα 12.1(14)EA1, 12.2(25)SEE2 και 12.2(44)SE6 IOS Version.



Βέβαια όπως φαίνεται και στον παρακάτω πίνακα υπάρχουν αρκετές συσκευές οι οποίες χρησιμοποιούν άλλα IOS Version. Παρακάτω δίνεται αναλυτικά ο πίνακας των IOS Version όλων αυτών των συσκεών.

IOS Version	**Πλήθος**	**IOS Version**	**Πλήθος**
12.0(5)WC8	3	12.1(22)EA1	1
12.1(14)EA1	7	12.1(22)EA2	1
12.1(22)EA13	3	12.1(26)E4	1
12.1(22)EA4a	6	12.1(9)EA1	1
12.1(22)EA6	2	12.2(25)SG	1
12.2(25)SEE2	7	12.2(31)	1
12.2(35)SE5	6	12.2(44)SE3	1
12.2(44)SE6	7	12.2(50)SE3	1
12.4(10b)JA	3	12.2(50)SG	1
12.0(27)	1	12.3(4)JA	1
12.1(19)EA1c	1	12.3(8)JA	1

Επίλογος

Η παρούσα πτυχιακή εργασία είχε σαν στόχο την δημιουργία ενός web – based εργαλείου συλλογής πληροφοριών. Σκοπός της εφαρμογής είναι η συλλογή χρήσιμων πληροφοριών από δικτυακές συσκευές με τη χρήση του πρωτοκόλλου SNMP (Simple Network Management Protocol). Η συλλογή τέτοιων πληροφοριών παρέχει τη δυνατότητα εποπτείας της ομαλής λειτουργίας του δικτύου. Επίσης, παρέχεται σε κάθε χρήστη η δυνατότητα λήψης βασικών πληροφοριών για το εκάστοτε σύστημα, με την προυπόθεση να είναι εφοδιασμένος με βασικές γνώσεις στο θέμα των δικτύων.

Η υλοποίηση της συγκεκριμένης εφαρμογής απευθύνεται κυρίως σε διαχειριστές δικτύων, αλλά και σε έμπειρους χρήστες σε θέματα αυτών. Παρόλο που είναι δυνατή η συλλογή πληροφοριών από πολλές συσκευές, διαφόρων εταιριών, για την ανεύρεση του δικτύου δόθηκε μεγαλύτερη βαρύτητα στις συσκευές Cisco, χωρίς όμως αυτό να σημαίνει ότι οι συσκευές των άλλων εταιριών δεν υποστηρίζονται.

Η περαιτέρω μελλοντική ανάπτυξη της εφαρμογής θα μπορεί να χαρτογραφεί όλο το δικτύο παρέχοντας μια οπτική απεικόνιση αυτού. Επίσης θα είναι δυνατή η συλλογή πληροφοριών απο συσκευές περισσότερων κατασκευαστών καθώς και η δυνατότητα λήψης πληροφοριών σε μεγαλύτερο βάθος.

Αναφορές

[1] PHP, http://www.php.net/

[2] Net-SNMP, http://net-snmp.sourceforge.net/

[3] phpsnmp, http://eder.us/projects/phpsnmp/

[4] Smarty Template Engine, http://www.smarty.net/

[5] W3Schools Online Web Tutorials, http://www.w3schools.com/html/default.asp

[6] Simple Network Management Protocol,http://en.wikipedia.org/wiki/ Simple_Network_Management_Protocol

[7] Laura Thomson και Luke Welling: Ανάπτυξη Web εφαρμογών με PHP και MySQL, τρίτη έκδοση

[8] CISCO (MIB locator), http://tools.cisco.com/Support/SNMP /do/BrowseOID.do?local=en

[9] The TCP/IP Guide, http://www.tcpipguide.com/

[10] PEAR-PHP_Extension_and_Application_Repository, http://pear.php.net/package/Net_CheckIP2/docs/latest/Net_CheckIP2 /_Net_CheckIP2-1.0.0RC2—examples—check-ip.php.html

[11] SNMP in Network Troubleshooting, http://support.3com.com/infodeli/tools /netmgt/tncsunix/product/091500/c15snmp.htm

[12] Simple Network Management Protocol , http://www.cisco.com/en/US/docs/internetworking/technology /handbook/SNMP.html#wp1022871

[13] What is SNMP?, http://www.tech-faq.com/snmp.shtml

[14] SNMP Tutorial Part 1: An Introduction to SNMP, http://www.dpstele.com/layers/l2/snmp_l2_tut_part1.php

[15] Network Management ,http://www.pms.ifi.lmu.de/mitarbeiter /ohlbach/-multimedia/IT/IBMtutorial/3376c414.html

[16] RFC1098 - Simple Network Management Protocol (SNMP), http://www.faqs.org/rfcs/rfc1098.html

[17] RFC1067 - Simple Network Management Protocol, http://www.faqs.org/rfcs/rfc1067.html

[18] SNMP PDUs and operations , http://ou800doc.caldera.com/en /NET_snmp/snmpC.pdus.html#snmp.flow

[19] RFC1155 - Structure and identification of management information, http://www.faqs.org/rfcs/rfc1155.html

[20] SNMPv3: A Security Enhancement for SNMP, http://www.comsoc.org/livepubs/surveys/public/4q98issue/stallings.html

[21] Α. Αλεξόπουλος και Γ. Λαγογιάννης: Τηλεπικοινωνίες και δίκτυα υπολογιστών, έκτη έκδοση

[22] Douglas E. Comer: Διαδίκτυα με TCP/IP, Τέταρτη Αμερικάνικη έκδοση

[23] Mark A. Dye και Rick McDonald και Antoon W. Rufi: Network Fundamentals, εκδόθηκε απο τη Cisco

[24] Rick Graziani και Allan Jonshon: Routing Protocols and Concepts, εκδόθηκε απο τη Cisco

[25] Wayne Lewis: LAN Switching and Wireless,εκδόθηκε απο τη Cisco

[26] Bob Vachon και Rick Graziani, Accessing the WAN,εκδόθηκε απο τη Cisco