SIMPLE NETWORK MANAGEMENT PROTOCOL
SNMPV3

VERSION 3

ΑΛΜΠΑΝΗ Α. ΜΑΡΙΑ

Α.Μ.: Μ2/98

ΕΙΣΑΓΩΓΗ

Το *simple network management protocol (SNMP)* δημιουργήθηκε το 1988 με σκοπό να παράγει έναν εύκολο τρόπο διαχείρισης του δικτύου διοίκησης των δρομολογητών, των ηλεκτρονικών υπολογιστών , των σταθμών εργασίας και άλλων ερευνητικών δικτύων. Το πρωτόκολλο αυτό ειδικεύεται στα εξής θέματα:

- *Ορίζει ένα πρωτόκολλο για την ανταλλαγή πληροφοριών ανάμεσα σε ένα ή περισσότερα διευθυντικά συστήματα και σε ένα αριθμό πρακτόρων (agents).*
- *Παράγει ένα σκελετό για το φορμάρισμα και την αποθήκευση διοικητικών πληροφοριών .*
- *Ορίζει έναν αριθμό ποικίλων διοικητικών πληροφοριών γενικού-σκοπού ή αντικειμένων.*

Η γνήσια έκδοση του SNMP έγινε αμέσως γνωστή και χρησιμοποιήθηκε ευρέως από το ανεξάρτητο δίκτυο διοικητικού πλαισίου. Καθώς όμως το πρωτόκολλο άρχισε να γίνεται γνωστό τα ελαττώματα του άρχισαν να γίνονται αντιληπτά. Αυτά είναι, η ελλιπής επικοινωνία manager to manager , η ανικανότητα να μεταφέρει σώμα δεδομένων , και τέλος η ελλιπής ασφάλεια (security).

Έτσι το 1993 μεταφερόμαστε στη νέα έκδοση του SNMP στη version 2. Αυτή έρχεται να αντικαταστήσει τη version 1 και τα ελαττώματα που προαναφέραμε. Στην αρχή η βελτιωμένη αυτή έκδοση αγκαλιάστηκε γρήγορα από τους πελάτες , όμως η ικανότητα της ασφάλειας που παρείχε δεν ικανοποίησε τους σχεδιαστές γιατί τη βρήκαν πολύπλοκη. Το αποτέλεσμα της ομάδας εργασίας ήταν να δημιουργήσουν ένα tune –up σχέδιο για το SNMPv2 το οποίο είχε ένα μεγάλο μειονέκτημα και ένα μικρό πλεονέκτημα. Το μικρό πλεονέκτημα είναι το tune-up του των λειτουργικών θεμάτων και το μεγάλο μειονέκτημα είναι στη περιοχή της ασφάλειας. Η ομάδα εργασίας φάνηκε ανίκανη να ξαναλύσει το πρόβλημα και έτσι δύο συναγωνίσιμες προσεγγίσεις εμφανίσθηκαν. Με αυτό το tune-up το λειτουργικό τμήμα του snmp παρήχθη το 1996 από το draft internet standard status.

Κατά αυτό τον τρόπο φθάνουμε αισίως στο 1997 και στην αναπτυσσόμενη έκδοση του SNMPv3 η οποία παρέχει ελάχιστες λειτουργικές ανακαινίσεις, αλλά έχει την ικανότητα να μας παρέχει μια δυνατή προσέγγιση ασφάλειας. Στην εργασία μου θα αναπτύξω τη προσέγγιση του SNMPv3 η οποία έχει πολλά πλεονεκτήματα σε σχέση με τις προηγούμενες εκδόσεις και ίσως να χρησιμοποιηθεί μέσα στο επόμενο εξάμηνο εκτενέστερα. Θα ήταν καλό να τονίσουμε ότι ορισμένες λειτουργικές συναρτήσεις, σκέψεις και η αρχιτεκτονική είναι κοινές και στις τρεις εκδόσεις και θα τις αναφέρουμε πληροφοριακά . Όπως επίσης θα δώσουμε περισσότερη έμφαση στις καινοτομίες που φέρνει το νέο αυτό πρωτόκολλο στο σύστημα της ασφάλειας

Μία επιχείρηση που αναπτύσσεται στο internet- standard management framework περιλαμβάνει τέσσερα βασικά συστατικά:

- *Διάφορους διοικούμενους κόμβους , που ο καθένας έχει την οντότητα του SNMP οι οποίοι παρέχουν αποδοχή στους management agents (πράκτορες).*
- *Τουλάχιστον μία οντότητα SNMP με management applications- manager.*
- *Το πρωτόκολλο διεύθυνσης χρησιμοποιείται να για να μεταβιβάζει πληροφορίες διεύθυνσης ανάμεσα στις οντότητες του SNMP και*
- *Στην πληροφορία διεύθυνσης.*

Επομένως το μοντέλο του SNMP v3 περιλαμβάνει τα παρακάτω βασικά κλειδιά:

- **Management station**
- **Management agent**
- **Management information base**
- **Network management protocol**

Το simple network management protocol όπως παρατηρούμε περιλαμβάνει τη λέξη κλειδί ΄΄simple΄΄ που ως γνωστόν στα ελληνικά μεταφράζεται ως απλό και αυτή είναι η ιδιαιτερότητα και το χαρακτηριστικό που έχει το πρωτόκολλο αυτό . Σχεδιάστηκε με τέτοιο τρόπο ώστε να είναι απλό στην εκτέλεση και στο να καταναλώνει ελάχιστη επεξεργασία και έρευνα στο δίκτυο. Αποτελεί επομένως ένα εργαλείο στο να χτίσουμε το σκελετό του δικτύου διεύθυνσης.

Το SNMP είναι ο θεμέλιος λίθος των σημερινών σύγχρονων επιχειρήσεων που ασχολούνται με διοικητικά συστήματα δικτύου. Είναι ένα ανοιχτό πρότυπο το οποίο ορίζει ένα τρόπο στους διευθύνοντες (managers) του δικτύου να επιτύχουν μια ειδική παρουσίαση και το σχεδιασμό της πληροφορίας που θέλουν από ένα λειτουργικό πράκτορα κάτοικο ενός απομακρυσμένου δικτύου μηχανής. Όπως και τα γνωστά πρότυπα διεύθυνσης των δεδομένων των δικτύων (OSI-based cousin, CMIP, που κυριαρχούν στα στο δίκτυο διεύθυνσης τηλεπικοινωνιών) , έτσι και το SNMP διευθύνει τα πάντα , από τις αιτήσεις τω υπηρεσιών δικτύου μέχρι σε χιλιάδες υποδομές μηχανών δικτύων επιχειρήσεων. Το SNMP παράγει μια ευρέως διαδεδομένη επικοινωνία ανάμεσα στις φόρμες διεύθυνσης και στις μηχανές και στις υπηρεσίες ατομικών μηχανών δικτύου .

Το SNMPv3 δέχθηκε δυνατή υποστήριξη από τη κοινότητα των πελατών. Περισσότερο προς έκπληξή μας , διάφορες αναφορές εκτελέσεων των πρακτόρων και των λειτουργικών διευθυντών του SNMP υπήρχαν ήδη σε πρωτότυπο τη τελευταία άνοιξη , όταν ρίξαμε μια βιαστική ματιά που παρουσιάσθηκε στο Las Vegas για το SNMPv3 hotspot στο NetWorld+Interop trade show. Έτσι υποστηρικτές του SNMPv3 εμφανίζονται οι Bay Networks, BMC Software, Cisco Systems, Hewlett-Packard, Liebert Corp. and Tivoli Systems.Οι πράκτορες της έκδοσης 3 είναι ακόμη πρωτότυποι σε ώρες αιχμής αλλά η αυξανόμενη υποστήριξη της έκδοσης αυτής ανάμεσα στα δίκτυα της διεύθυνσης και στην υποδομή των πελατών θα συνεχιστεί περισσότερο κατά τη διάρκεια του 1999. Ίσως να αναρωτηθεί κανείς αν χρειάζεται όλη αυτή η επίδειξη . Το απλό πρωτόκολλο διεύθυνσης δικτύου έκδοσης 3 σίγουρα ξεχειλίζει από υποσχέσεις. Στην εργασία μας θα αποδείξουμε ότι δεν είναι απλά

μια επίτευξη που υπόσχεται μόνο , αλλά παράλληλα υλοποιεί αυτά που υπόσχεται.

ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΤΟΥ SNMP

Για να εμποδίσουμε το Internet , να γίνει ένα μεγάλο και χαοτικό περιβάλλον, η διαχείριση του είναι ίσως το σημαντικότερο θέμα που προκύπτει. Η επεξεργασία της διαχείρισης πρέπει να διοικείται από μία ολοκληρωμένη οργάνωση –επιχείρηση η οποία να περιλαμβάνει διάφορα τμήματα τα οποία με κάποιο τρόπο νοιάζονται για τη σωστή διαχείριση και ύπαρξη του Internet . Τέτοια επιχείρηση υπάρχει και ονομάζεται Internet Engineering Task Force (IETF).

Η IETF είναι μία τεράστια ανοιχτή κοινότητα ανθρώπων, οι οποίοι κάνουν συμφωνίες ή καλύτερα διαχειρίζονται το Internet με ποικίλους τρόπους. Αυτοί οι άνθρωποι είναι για παράδειγμα , σχεδιαστές δικτύου, χειριστές, πελάτες και ερευνητές που σχετίζονται με την εξέλιξη του Internet. Η τεχνική δουλειά γίνεται από ομάδες εργασίας (Working Groups) όπου κάθε μία στηρίζει ένα διαφορετικό θέμα . Η ομάδα εργασίας επικοινωνεί κυρίως μέσω της λίστας του ταχυδρομείου (mailing list) κ.τ.λ. και συναντιούνται όταν η IETF καθορίζει μια συνάντηση , η οποία λαμβάνει χώρα περίπου τρεις φορές το χρόνο. Μία τέτοια ομάδα καθορίστηκε για να εργαστεί πάνω στο SNMPv3 . Γι' αυτό το λόγο θα πούμε κάποια πράγματα για την ιστορία και την αναγκαιότητα του πρωτοκόλλου αυτό γενικά και στη συνέχεια θα διευκρινίσουμε γιατί η τελευταία έκδοση είναι τόσο αναγκαία .

Στη δεκαετία των ογδόντα και πιο συγκεκριμένα στο δεύτερο μισό αυτής, η IETF αδυνατούσε στη διαχείριση του δικτύου του Internet. Μετά από κάποια συζήτηση συμπέραναν να χρησιμοποιούν το OSI' s CMIP. Όμως για να μπορέσει να ταιριάξει αυτό το πρωτόκολλο στο TCP/IP Internet ήταν αναγκαίες κάποιες αλλαγές. Το αποτέλεσμα αυτών των αλλαγών ήταν το CMOT

(Common Management Over TCP/IP). Η ανάπτυξη της διαχείρισης του OSI πήρε αρκετό χρόνο. Και αυτό γιατί η IETF δεν ήθελε να κάθεται και να περιμένει κάποια αποτελέσματα , έτσι αποφάσισαν να επεκτείνουν το ήδη υπάρχον και αναπτυγμένο πρωτόκολλο SGMP(Simple Gateway Monitoring Protocol) και να το χρησιμοποιήσουν σαν μία βραχυχρόνια λύση. Πρόθεση τους ήταν ότι μετά από κάποιο διάστημα αυτή η λύση θα αντικαταστούταν από μία δομική λύση βασισμένη στο OSI.

Το πρωτόκολλο διαχείρισης που σχεδιάστηκε SGMP ονομάστηκε αργότερα SNMP (Simple Network Management Protocol) και ακολουθεί τα παρακάτω κριτήρια :

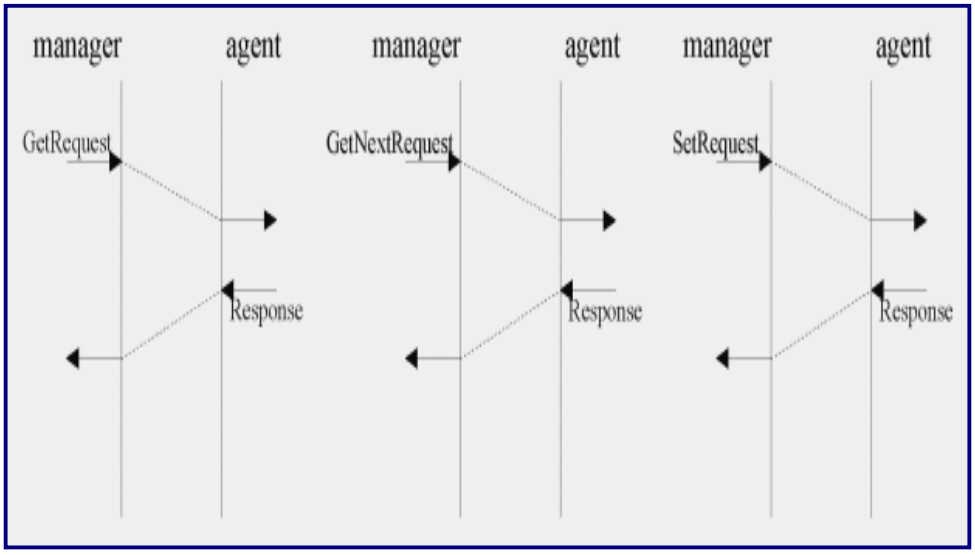
- *Το SNMP μπορεί να διαχειρίζεται κάθε σύστημα το οποίο είναι συνδεδεμένο με το Internet*
- *Το κόστος εκτέλεσης του SNMP είναι ελάχιστο*
- *Ορίζοντας νέα ΄΄διαχειρίσιμα αντικείμενα ΄΄, οι ικανότητες διαχείρισης μπορούν να επεκταθούν εύκολα*

- *Το Snmp είναι δυνατό, ακόμα και στο γεγονός της αποτυχίας ο διαχειριστής μπορεί να συνεχίσει να εργάζεται (ακόμα και όταν λίγη περισσότερη προσπάθεια ίσως να χρειάζεται εκείνη τη στιγμή*)

Πιστεύω ότι το πρωτόκολλο SNMP ήταν η σωστή λύση τη σωστή στιγμή. Μέσα στα χρόνια που ακολούθησαν κατάφερε να αποδείξει ότι είχε την ικανότητα να διαχειριστεί τη πλειοψηφία των κατασκευαστών που ήταν συνδεδεμένοι με το Internet. Σήμερα οι παραγωγοί των κατασκευαστών δεδομένων επικοινωνίας ενσωματώθηκαν στο snmp κατά λάθος. Το πρωτόκολλο αυτό έχει γίνει σήμερα το πιο σημαντικό πρότυπο στη διαχείριση δικτύων.

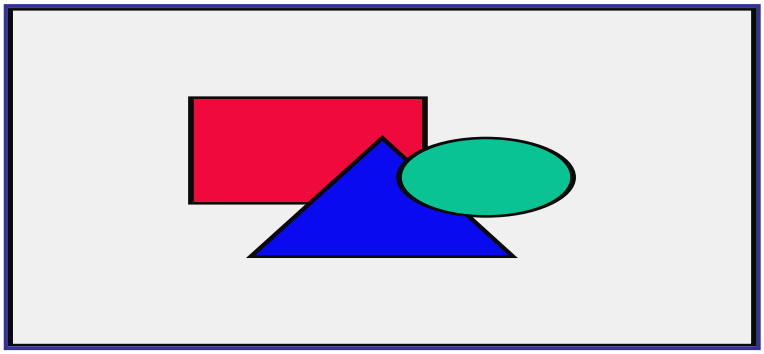
Η επιτυχία του πρωτοκόλλου δεν μπορούσε να προβλεφθεί από τον οποιονδήποτε , ούτε ακόμα και από την ίδια την IETF . Εξαιτίας αυτής της επιτυχίας , το αρχικό σχέδιο να αντικαταστήσουν το SNMP με το CMOT καταρρίφθηκε το 1992. Αυτή τη στιγμή είναι απίθανο αν το OSI χρησιμοποιηθεί ποτέ ξανά για το δίκτυο διαχείρισης.

Η πρωτοβουλία για να στέλνουν πληροφορίες διαχείρισης ανήκει συνήθως στη πλευρά των διαχειριστών (managers). Και αυτό για να πετύχουν μία συγκεκριμένη πληροφορία ο διαχειριστής θα στείλει την εντολή 'GetRequest' –'GetNextRequest' στον πράκτορα (agent). Η πληροφορία ζήτησης μπορεί να επιστραφεί από το πράκτορα με το'Response'. Εάν ο διαχειριστής θελήσει να αλλάξει τη πληροφορία που περιέχει ο πράκτοράς τότε θα στείλει το 'SetRequest'. Για να αναφέρει οποιαδήποτε λάθη , ο πράκτορας θα αντιδράσει και σ' αυτή τη περίπτωση με το 'Response'.



Ο διαχειριστής παίρνει πρωτοβουλία

Σε μερικές περιπτώσεις ο πράκτορας μπορεί να πάρει τη πρωτοβουλία να στείλει τη πληροφορία επίσης. Για να το κάνει λοιπόν αυτό ο πράκτορας στέλνει ένα μήνυμα 'Trap' που φυσικά θα εξηγήσουμε εκτενέστερα τις εντολές αυτές. Ο διαχειριστής παρόλα αυτά δεν αντιδρά σ' αυτό με μια απάντηση. Παραδείγματα αυτής της περίπτωσης είναι η αρχή των νέων συστημάτων , το resetting των συστημάτων και το fall-out των συνδέσεων ανάμεσα στα συστήματα .



Ο πράκτορας παίρνει πρωτοβουλία

Σημαντικό για το SNMP είναι ότι αυτά τα μηνύματα τα οποία στέλνονται ανάμεσα στο πράκτορα και στο διαχειριστή μπορεί να χαθούν . Εκ πρώτης όψεως αυτό φαίνεται σαν ένα ανεπιθύμητο χαρακτηριστικό, αλλά οι σχεδιαστές ρητά επέλεξαν να το κάνουν με αυτό τον τρόπο . Για να κάνουμε τις πληροφορίες να ανταλλάσσονται με αξιοπιστία, πρέπει να χρησιμοποιήσουμε συναρτήσεις πρωτοκόλλου οι οποίες είναι με το δικό τους τρόπο εύτρωτες όταν άλλα προβλήματα δικτύου πραγματοποιούνται. Για να το εμποδίσουμε, πρέπει να αναπτύξουμε πολύπλοκες λύσεις , αλλά αυτό θα μας οδηγούσε στο να αφαιρέσουμε τη λέξη 'απλό' από την ονομασία του πρωτοκόλλου. Σαν αποτέλεσμα αυτού, είναι υπευθυνότητα των διαχειριστών να ξαναστέλνουν τα χαμένα μηνύματα.

Από στιγμή που το SNMP καθιερώθηκε σαν πρότυπο , πολλαπλές προτάσεις για ανάπτυξη αυτού εμφανίστηκαν . Το 1992 η IETF πακετάρισε έναν αριθμό αυτών των προτάσεων και την ανάπτυξη της νέας έκδοσης του SNMP που εμφανίστηκε σαν έκδοση δεύτερη (SNMPv2). Σε σύγκριση με τη γνήσια έκδοση του SNMP , αυτή η νέα έκδοση υποτίθεται ότι θα πρέπει να έχει τις παρακάτω ικανότητες:

- *Ασφάλεια διαχείρισης(μέσω της επικύρωσης, της απόκρυψης και της αποδοχής ελέγχου κάθε αντικειμένου διαχείρισης)*
- *Μεταφορά των διαχειριστικών πληροφοριών με πιο ικανοποιητικό τρόπο (με τη νέα εντολή 'GetBulk')*
- *Το χτίσιμο της ιεραρχίας των διαχειριστών*

Το SNMPv2 επίσης περιλαμβάνει ένα μεγάλο αριθμό μικρών επιτευγμάτων.

ΑΣΦΑΛΕΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

Το SNMPv1 πρωτόκολλο εκτέθηκε σε πολλές προειδοποιήσεις. Αυτές οι απειλές ή προειδοποιήσεις μπορούν να διαχωριστούν σε αρχικές , σε δευτερεύουσες και σε απειλές που είναι με λιγότερη σημασία. Οι αρχικές και οι δευτερεύουσες απειλές έθεσαν σε ώριμη σκέψη το snmpv2 το οποίο καλείται να παρέχει προστασία από αυτού του είδους τις απειλές. Οι αρχικές απειλές ενάντια στις οποίες το νέο πρωτόκολλο παρέχει προστασία είναι οι εξής:

- **Αλλαγή της πληροφορίας**

Η απειλή της τροπολογίας είναι ένας κίνδυνος στον οποίο μια μη εξουσιοδοτημένη οντότητα μπορεί να αλλάξει κατά τη μεταφορά της . Τα μηνύματα του SNMPv2 παράγονται εκ μέρους ενός εξουσιοδοτημένου χρήστη με τέτοιο τρόπο ώστε να έχουν σαν αποτέλεσμα μη εξουσιοδοτημένες λειτουργίες διαχείρισης, περιλαμβάνοντας νοθευμένη την αξία του αντικειμένου διαχείρισης

- **Masquerade**

Η απειλή του μασκαρέματος είναι κίνδυνος στις λειτουργίες διαχείρισης μη εξουσιοδοτημένες από κάποιον χρήστη ο οποίος επιχειρεί να αποκτήσει τη ταυτότητα κάποιου άλλου χρήστη ο οποίος έχει τις κατάλληλες εξουσιοδοτήσεις.

Επίσης η νέα αυτή έκδοση του πρωτοκόλλου παρέχει προστασία ενάντια στις παρακάτω δευτερεύουσες απειλές:

- **Message stream modification**

Το snmpv2 πρωτόκολλο είναι τυπικά βασισμένο πάνω σε μία υπηρεσία μεταφοράς χωρίς σύνδεση η οποία λειτουργεί πάνω σε μια οποιαδήποτε υπηρεσία υποδικτύου. Η παραγγελία , η επανάληψη της παραγγελίας , η καθυστέρηση των μηνυμάτων μπορεί και συμβαίνει πάνω διαμέσου φυσικών λειτουργιών πολλών τέτοιων υπηρεσιών υποδικτύων.

- **Αποκάλυψη (disclosure)**

Είναι η αποκάλυψη των ανταλλαγών ανάμεσα στους αντιπροσώπους ή πράκτορες και στους σταθμούς διαχείρισης. Η προστασία ενάντια σ' αυτή την απειλή μπορεί να θεωρηθεί σαν ένα συμβάν της τοπικής πολιτικής.

Οι επόμενες απειλές ανήκουν έξω από το όριο αναφοράς ή σ' αυτές που θεωρούνται μικρής σημασίας στις οποίες η νέα έκδοση του πρωτοκόλλου δεν χρειάζεται να παρέχει προστασία ενάντια τους. Αυτές είναι:

- Άρνηση των υπηρεσιών

Το πρωτόκολλο δεν χρειάζεται να προσπαθεί να διευθύνει τα ευρέα πεδία επιθέσεων των οποίων η υπηρεσία των εκ μέρους των εξουσιοδοτημένων χρηστών αρνήθηκε. Πράγματι, αυτή η άρνηση των υπηρεσιών είναι σε πολλές περιπτώσεις δυσδιάκριτη.

- Ανάλυση συμφόρησης

Εν συντομία το νέο αυτό πρωτόκολλο της δεύτερης έκδοσης μας εξασφαλίζει τα εξής:

- Ένα μήνυμα λαμβάνων το οποίο λεει από πού έχει έρθει
- Ένα μήνυμα λαμβάνων το οποίο δεν έχει μεταλλαχθεί κατά τη διάρκεια της μεταφοράς
- Ένα μήνυμα το οποίο έχει ληφθεί μέσα σε ένα συγκεκριμένο time-window
- Ένα μήνυμα το οποίο περιλαμβάνει μια μυστική πληροφορία δεν μπορεί να διαβαστεί από οποιοδήποτε άλλο παρά από τον αποδέκτη ενώ διασχίζει κατά μήκος το δίκτυο.

Έτσι το 1993, το SNMPv2 έγινε ένα προτεινόμενο πρότυπο . Στο μεταξύ πολλαπλές ομάδες έρευνας έχτιζαν καινούργια πρότυπα. Πολύ σύντομα έγινε ξεκάθαρο ότι η έκδοση αυτή ήταν περισσότερο πολύπλοκη από ότι ο κόσμος περίμενε να είναι. Όταν η ερώτηση τέθηκε το 1994 , ενώ υπήρχε ικανοποιητική υποστήριξη για τη προώθηση του πρωτοκόλλου αυτού , άρχισε να γίνεται συζήτηση για τη πολυπλοκότητα του. Ο κύριος ντόρος γινόταν γύρω από ένα διοικητικό μοντέλο το οποίο να περιγράφει πως τα δεδομένα χρειάζονται ασφάλεια. Σαν αποτέλεσμα δύο διαφορετικές προσεγγίσεις αναδύθηκαν η USEC και η v2*. Αν και καμία από τις δύο δεν παρείχε τη ζητούμενη ασφάλεια . Έτσι οι δύο προσεγγίσεις έγιναν μια και ονομάστηκε SNMPv3 ή όπως αλλιώς είναι γνωστή σαν Next Generation. Τον Απρίλιο του 1998 έγινε πρόταση προτύπου.

TABLE OF SNMP EVOLUTION

ABSTRACT

The purpose of this assignment is to present Version 3 of the Simple Network Management Protocol (SNMPv3). SNMPv3 is the most recent version of the Management Network and provides robust attributes of security and access control. It has been designed with simplicity in mind at the point of implementation and therefore requires less process and research in the network.As a result, it is a tool on which the basic framework of the management network can be built. SNMP is a cornerstone of today's enterprises that use management network systems. SNMPv3 consists of the following basic components:

- Management Station
- Management Agent
- Management Information Base
- Network Management Protocol

SNMPv3 will probably be installed on the Network within the next six months and already has the necessary support from such organizations and companies as Hewlett-Packard, Liebert Corp. and Tivoli Systems. This protocol is targeted towards the most popular protocol of Internet TCP/IP. It provides the language of queries and transits the engines of queries to agents which can run in management devices.

ΠΕΡΙΛΗΨΗ

Ο σκοπός αυτής της εργασίας είναι να παρουσιάσει την έκδοση 3 του Απλού Δικτύου Διαχείρισης Πρωτόκολλο (SNMPv3). Το SNMPv3 είναι η πιο πρόσφατη έκδοση του δικτύου διαχείρισης και παρέχει δυνατά χαρακτηριστικά στην ασφάλεια και στην αποδοχή ελέγχου. Έχει σχεδιαστεί με απλότητα σκέψης όσον αφορά την εκτέλεση και γι' αυτό λοιπόν απαιτεί ελάχιστη επεξεργασία και έρευνα στο δίκτυο. Σαν αποτέλεσμα , είναι ένα εργαλείο πάνω στο οποίο ο βασικός σκελετός της διαχείρισης δικτύου μπορεί να χτιστεί. Το πρωτόκολλο απλής διαχείρισης δικτύου είναι ένας θεμέλιος λίθος των σημερινών επιχειρήσεων οι οποίες χρησιμοποιούν συστήματα διαχείρισης δικτύου. Το πρωτόκολλο αυτό αποτελείται από τα επόμενα βασικά συστατικά:

- Σταθμός διαχείρισης
- Πράκτορας διαχείρισης
- Βάση πληροφορίας διαχείρισης
- Πρωτόκολλο διαχείρισης δικτύου

ίσως να εγκατασταθεί στο δίκτυο τους επόμενους έξι μήνες , ενώ ήδη έχει την απαραίτητη υποστήριξη που χρειάζεται από επιχειρήσεις και οργανισμούς όπως οι Hewlett-Packard, Liebert Corp. and Tivoli Systems.Το SNMPv3 στοχεύει στα πιο δημοφιλή πρωτόκολλα του Internet το TCP/IP.Παρέχει τη γλώσσα των ερωτήσεων και της μεταφοράς αυτών από τους πράκτορες στους διαχειριστές οι οποίοι τρέχουν σε μηχανές διαχείρισης.

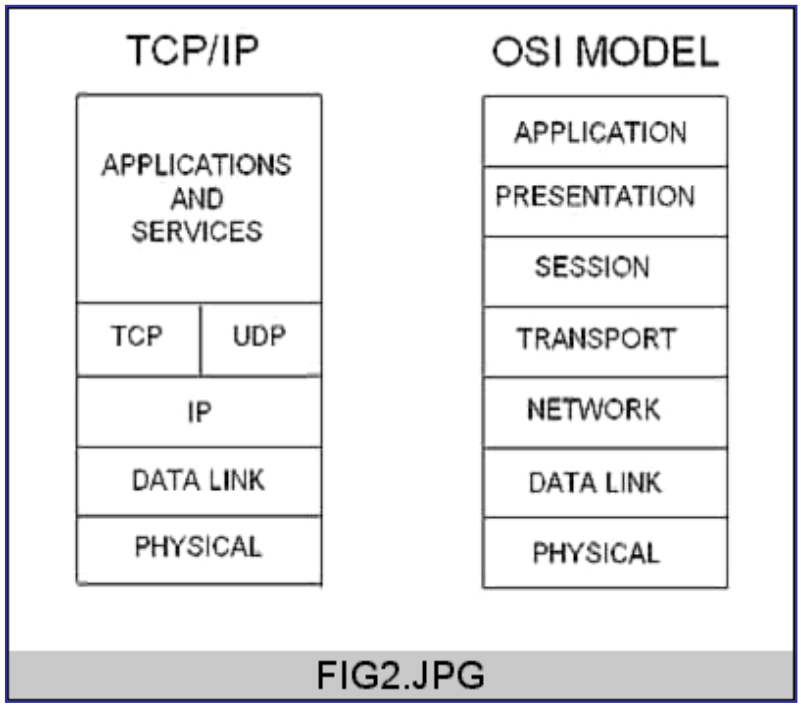
Η ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ SNMPv3

Σήμερα κάθε εμπορικό περιοδικό και κάθε διαφήμιση θα δηλώσει το δικό της hub, τη δική της bridge, router, multiplexor , switch ή οτιδήποτε άλλο θα μπορούσε να συνεργαστεί με το πρωτόκολλο SNMP. Πως όμως συμβαίνει αυτό; κάποια ειδικά χαρακτηριστικά του σκελετού του SNMP βοηθούν με τη δική τους εκπληκτική ανάπτυξη , **έρευνα και ευρεία χρήση**:

- Οποιαδήποτε μηχανή η οποία μπορεί να υποστηρίξει ένα πολύ μικρό σύνολο λειτουργικού μπορεί να συμμετέχει
- Οποιοδήποτε λειτουργικό developer μπορεί να χρησιμοποιήσει δημοσίως τα διαθέσιμα πρότυπα πρωτοκόλλου και εργαλείων με σκοπό να χτίσουν ένα σταθμό διαχείρισης δικτύου και να το προσφέρουν σαν προϊόν
- Ο σταθμός διαχείρισης μιλά με όλες τις μηχανές με τον ίδιο τρόπο , και έτσι προσωρινά μπορεί να διαχειριστεί οποιαδήποτε μηχανή επιθυμεί
- Η επέκταση της εμβέλειας της διαχείρισης είναι εύκολη.

Συγκεντρώνοντας τη κοινότητα των πελατών που ενδιαφέρονται και των εξειδικευμένων τεχνολόγων , με σκοπό να τους θέσουμε να γράψουν ένα τμήμα με ορισμούς .

Το πρωτόκολλο που μελετούμε έχει σαν στόχο το περισσότερο διαδεδομένο πρωτόκολλο του Internet το TCP/IP. Παρακάτω θα δείξουμε το πώς ταιριάζει το TCP/IP στο SNMP αλλά και την αλληλεπίδρασή τους. Το International Organization for Standardization (ISO) Open System Interconnect (OSI) model, όπως παρατηρούμε στο παρακάτω σχήμα είναι χρήσιμο για το καθορισμό της ταυτότητας των βασικών συστατικών του TCP/IP πρωτοκόλλου που ταιριάζουν στο :



TCP/IP and OSI layers

Φυσικό επίπεδο(physical layer)

Το φυσικό επίπεδο ή επίπεδο 1 έρχεται σε επαφή με την υπογραμμισμένη επικοινωνία του hardware συστήματος. Το επίπεδο 1 περιλαμβάνει λεπτομερής περιγραφές των μέσων μαζικής ενημέρωσης όπως είναι το coaxial cable, twisted pair, fiber. Αυτά το πρότυπα καθορίζουν φυσικές συνδέσεις και ξεχωριστές μεθόδους. Ο σταθμός διαχείρισης του snmp μπορεί να προκαλέσει tests τα οποία να ελέγχουν τη μεσαία κατάσταση ή τη μεσαία διασύνδεση.

Το επίπεδο διασύνδεσης των δεδομένων (the data link layer)

Το επίπεδο διασύνδεσης των δεδομένων ή επίπεδο 2 όπως αλλιώς ονομάζεται είναι υπεύθυνο για τη μεταφορά της πληροφορίας ανάμεσα σε δύο συστήματα τα οποία είναι συνδεδεμένα με ΄μια σύνδεση point-to-point , LAN (ETHERNET) ή packet network circuit (frame relay). Οι βασικές συναρτήσεις του επιπέδου διασύνδεσης των δεδομένων είναι οι:

- Ένα πακέτο πληροφοριών μέσα σε πλαίσιο
- Παράγει φυσικές πληροφορίες καθορίζοντας τις πηγές και τις διευθύνσεις

Ένας σταθμός διαχείρισης του snmp μπορεί να χρησιμοποιηθεί για το σχεδιασμό την ενεργοποίηση την απενεργοποίηση των διασυνδέσεων. Μπορεί επίσης να επιτύχει την είσοδο και την έξοδο πλαισίων οχταδικών καθώς επίσης και λαθών για κάθε διασύνδεση.

Το επίπεδο δικτύου (the network layer)

Το Internet Protocol IP αντιπροσωπεύει το επίπεδο 3 και έχει τη δουλειά της δρομολόγησης των δεδομένων μέσα στο δίκτυο. Οποιοδήποτε δίκτυο το οποίο βασίζεται στο IP καλείται internet. Η σωστή επιχείρηση του IP εξαρτάται από :

- Τη μοναδική ανάθεση εργασίας των διευθύνσεων του IP
- Την ικανότητα της μετάφρασης των διευθύνσεων του IP σε φυσικές διευθύνσεις (ARP)
- Ακριβολογεί εισόδους σε πίνακες δρομολόγησης

Το επίπεδο μεταφοράς (the transport layer)

Η πρωτεύουσα υποχρέωση του επιπέδου μεταφοράς ή επιπέδου 4 είναι να παρέχει επικοινωνία από μία αίτηση ενός προγράμματος σε άλλο. Τέτοιου είδους επικοινωνία καλείται end-to- end. Δύο πρωτόκολλα το TCP και το UDP καθορίζουν το επίπεδο 4. Ο σταθμός διαχείρισης του SNMP παρακολουθεί τον αριθμό και τη διάρκεια των συνδέσεων του TCP στο σύστημα και ιχνηλατεί ποιος μιλάει με ποιον. Επίσης επιτυγχάνει αθροίσματα κυκλοφοριακών λαθών του TCP και του UDP.

Αιτήσεις (applications)

Στα υψηλότερα επίπεδα, οι χρήστες εμπλέκουν προγράμματα αιτήσεων τα οποία υπηρεσίες διαθέσιμες στο TCP/IP internet. Οι αιτήσεις TCP/IP περιλαμβάνουν electronic mail, file transfer, terminal access.

Στη συνέχεια θα δώσουμε μερικά τυπικά συστατικά του περιβάλλοντος του πρωτοκόλλου του SNMP:

Μηχανές διαχείρησης (managed devices)

Μηχανές δικτύου όπως hubs, routers, bridges που τρέχουν πράκτορες (agents) οι οποίοι συλλέγουν πληροφορίες σχετικές με τις επιχειρήσεις τους . Οι πράκτορες κάνουν λίγη δουλειά από μόνοι τους , εκτός από τη παρακολούθηση και τον έλεγχο των σημαντικών προγραμμάτων και γεγονότων που συμβαίνουν στην μηχανή. Πιο συγκεκριμένα ο πράκτορας κάνει τις παρακάτω εργασίες:

- Συλλέγει και συντηρεί τις σχετικές πληροφορίες με το τοπικό του περιβάλλον
- Προωθεί αυτή τη πληροφορία στο διαχειριστή (manager) , είτε για απάντηση στην αίτηση είτε για μια αζήτητη διαμόρφωση όταν κάτι αξιοπρόσεκτο συμβαίνει
- Απαντά στις παραγγελίες του διαχειριστή για αλλαγή στο τοπικό σχηματισμό ή στις λειτουργικές παραμέτρους

Επειδή αναφέρουμε πολύ συχνά την έννοια του διαχειριστή (manager) στην εργασία μας θα ήταν καλό στο σημείο αυτό να ρεζουμάρουμε ή καλύτερα να οριοθετήσουμε τις λειτουργίες του διαχειριστή , μια και ο ρόλος του στο πρωτόκολλο που αναλύουμε είναι πολύ σημαντικός. Ένας σχηματισμός μπορεί να περιλαμβάνει ένα ή και περισσότερους διαχειριστές ή σταθμούς διαχείρισης δικτύου. Ο σταθμός διαχείρισης γενικά παράγει διασύνδεση στο χρήστη έτσι ώστε ένα ανθρώπινο δίκτυο διαχείρισης να μπορεί να ελέγχει και να παρατηρεί την επεξεργασία του δικτύου διαχείρισης. Αυτή η διασύνδεση επιτρέπει στον χρήστη να εκθέτει παραγγελίες ή εντολές (π.χ. απενεργοποίηση σύνδεσης , να συλλέγει στατιστικά στοιχεία μιας παρουσίασης κ.ο.κ.). Επίσης του δίνει τη δυνατότητα να παράγει μια λογική σειρά στη περίληψη και στο φορμάρισμα της πληροφορίας που συλλέγεται από το σύστημα. Έτσι χωρίς να το καταλάβουμε περνούμε στο δεύτερο σημαντικό συστατικό του περιβάλλοντος που εξετάζουμε το:

Σύστημα διαχείρισης(management systems)

Ένα διοικητικό σύστημα το οποίο συλλέγει πληροφορίες από τους από τους πράκτορες οι οποίοι τρέχουν στις μηχανές διαχείρισης (management devices). Κοινές συναρτήσεις της κονσόλας διαχείρισης περιλαμβάνουν το χάρτη της τοπολογίας του δικτύου, γεγονότα που παγιδεύονται με συναγερμούς , παρακολούθηση της κυκλοφοριακής συμφόρησης , διάγνωση συναρτήσεων δικτύου , γεννήτορες αναφοράς , ιστορική αναφορά διαχείρισης , και την ανάλυση της γενικής κατεύθυνσης. Η καρδιά ενός συστήματος διαχείρισης δικτύου είναι ένα σύνολο από αιτήσεις οι οποίες συναντούν τις έντονες ανάγκες για διαχείριση δικτύου. Στο ελάχιστο, ένα σύστημα θα περιλαμβάνει βασικές

αιτήσεις για παρουσίαση παρακολούθησης και ελέγχου , σχηματισμό ελέγχου, και λογιστική. Περισσότερο μελετημένα συστήματα θα περιλαμβάνουν περισσότερο επεξεργάσιμες αιτήσεις σ' αυτές τις κατηγορίες , συν τη διευκόλυνση για ελαττωματική απομόνωση και διόρθωση , και για τη διαχείριση τα χαρακτηριστικά της ασφάλειας του δικτύου.

SNMP

Αυτό είναι ένα πρωτόκολλο το οποίο παράγει τη γλώσσα ερωτήσεων και μεταφέρει τους μηχανισμούς για ερωτήσεις στους πράκτορες οι οποίοι τρέχουν στις μηχανές διαχείρισης . Το SNMP χρησιμοποιεί μία ερώτηση και μια απάντηση επεξεργασίας για να επικοινωνήσει με τους πράκτορες και για να επιτύχει αξίες ή να αλλάξει τη κατάσταση των στοιχείων του δικτύου. Το SNMP χρησιμοποιεί UDP (User Datagram Protocol) για να κουβαλήσει τα μηνύματα του και τα δεδομένα του μέσα στο δίκτυο . Είναι ένα πρωτόκολλο μεταφοράς χωρίς σύνδεση με ένα απλό σύνολο εντολών . Γι' αυτό το λόγο είναι ένα πρωτόκολλο ικανοποιητικό και το οποίο μπορεί να λειτουργήσει πάνω στα διάφορα δίκτυα τα οποία είναι υπερφορτωμένα και αποτυχημένα.

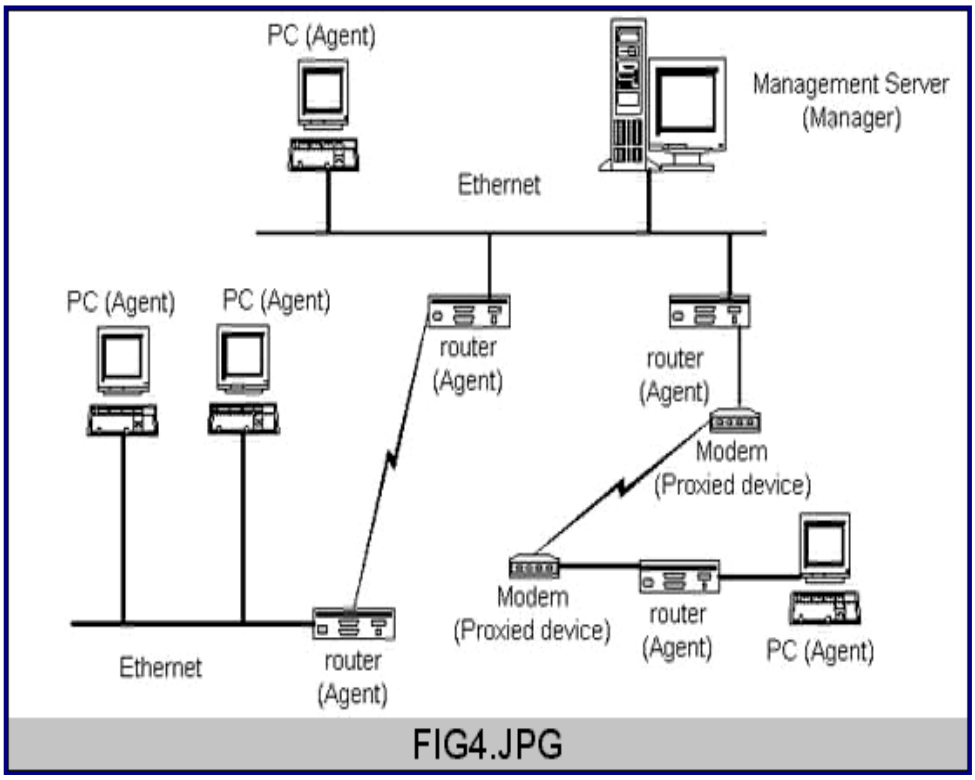
Όλες οι εφαρμογές (applications) του δικτύου διαχείρισης γενικά μοιράζονται ένα κοινό πρωτόκολλο διαχείρισης δικτύου. Αυτό το πρωτόκολλο παράγει τις βασικές συναρτήσεις για σώσιμο της πληροφορίας διαχείρισης από τους πράκτορες και για θεματολογία εντολών των πρακτόρων . Αυτό το πρωτόκολλο , απεναντίας , κάνει εύκολη τη χρήση επικοινωνίας , όπως το TCP/IP ή το OSI.

Τέλος, όπως προαναφέραμε κάθε πράκτορας συντηρεί τη βάση πληροφορίας διαχείρισης (γνωστή ως management information base- MIB) η οποία περιέχει συνεχιζόμενες και ιστορικές πληροφορίες σχετικές με το τοπικό σχηματισμό της και τη κυκλοφοριακή συμφόρηση της. Ο σταθμός διαχείρισης θα συντηρήσει ένα γενικό MIB με μια περιληπτική πληροφορία από όλους τους πράκτορες. **Η MIB αποτελείται από ένα σύνολο στοιχείων που ονομάζονται διαχειρίσιμα αντικείμενα που περιέχουν πληροφορίες διαχείρισης συγκεκριμένων κόμβων** . Θα μιλήσουμε γι' αυτή εκτενέστερα παρακάτω , αφού η παρουσία της είναι πολύ σημαντική και στις τρεις εκδόσεις του πρωτοκόλλου που εξετάζουμε.

Πληρεξούσιος πράκτορας (proxy agent)

Κάποιες μηχανές δεν έχουν την ικανότητα να τρέξουν ένα πράκτορα . ο πληρεξούσιος πράκτορας είναι μια μηχανή η οποία μπορεί να τρέξει ένα πράκτορα όπως μια μηχανή και να το παράγει με συναρτήσεις SNMP. Ο πληρεξούσιος πράκτορας είναι επίσης χρήσιμος για το διαχειρισμό και τη παρακολούθηση υποδικτύων τα οποία πρέπει να διατηρούν ασφάλεια. Ένας κεντρικός σταθμός διαχείρισης τοποθετείται έξω από το υποδύκτιο έτσι ώστε να μπορεί να επικοινωνεί με ένα εσωτερικό σταθμό διαχείρισης ο οποίος περιορίζει τη συμφόρηση που δημιουργείται από τον έξω κόσμο. Ένα άλλο σενάριο , θα μπορούσε να περιλαμβάνει παλιότερες μηχανές οι οποίες να μην μπορούν μα διαχειριστούν από το SNMP άμεσα αλλά διαμέσω ενός άλλου ιδιόκτητου πρωτοκόλλου το οποίο λειτουργεί μέσα σε ένα κατά παραγγελία σχεδιασμένο σύστημα διαχείρισης. Ο πληρεξούσιος πράκτορας τότε μεταφράζει

τις ερωτήσεις του SNMP μέσα στις ιδιόκτητες ερωτήσεις και ξαναφορμάρει τα εσωτερικά δεδομένα της μηχανής σε μορφή MIB.



Περιβάλλον SNMP παράδειγμα

Η αρχιτεκτονική του SNMP αποτελείται από κατανεμημένα αλληλοεπιδρόμενα τμήματα του SNMP. Κάθε ολότητα εκτελεί ένα τμήμα της ικανότητας του SNMP και ίσως να δρά σαν κόμβος πράκτορα, ή σαν συνδυασμό και των δύο. Κάθε ολότητα αποτελείται από μια συλλογή τμημάτων ή στοιχείων (modules) τα οποία αλληλεπιδρούν μεταξύ τους με σκοπό να παράγουν τις απαραίτητες υπηρεσίες. Αυτές οι αλληλεπιδράσεις μπορούν να μοντελοποιηθούν σαν ένα σύνολο περιληπτικών φυσικών στοιχείων και παραμέτρων. Ο **σχεδιασμός του SNMPv3** είναι ο εξής:

1. Επιτρέπει την εκτέλεση πάνω σε ένα ευρύ πλαίσιο λειτουργικών περιβαλλόντων , κάποια από τα οποία χρειάζονται ελάχιστη , φθηνή λειτουργικότητα και μερικά από αυτά ίσως υποστηρίζουν πρόσθετα χαρακτηριστικά για τη διαχείριση μεγάλων δικτύων
2. Κάνει ικανή τη μετακίνηση τμημάτων της αρχιτεκτονικής μπροστά από τα πρότυπα ακόμα και όταν η πλειοψηφούσα γνώμη δεν έχει ακόμα φθάσει σε όλα τα κομμάτια
3. Περιθάλπει εναλλακτικά μοντέλα ασφάλειας

Τα τμήματα (modules) είναι τα παρακάτω:

Module 1 : Network Management Architectures

The need for network management
- Growth in network size and complexity
- Business-critical nature of networks

Network management framework
- The OSI Reference Model
- Five areas of network management
- TCP/IP network management model

Module 2 : The Simple Network Management Protocol

SNMP evolution
- The Internet standards process
- Original goals and purpose of SNMP

Introduction to MIB definitions
- Structure of Management Information (SMI)
- MIB I, MIB II and extensions
- Examples of MIB definitions

SNMPv1, SNMPv2 and SNMPv3 functions
- Get
- Get-Next
- Set
- Trap
- Get-Bulk
- Inform Request
- Authentication and other requirements

Module 3 : Management Information Bases

MIB definition: rules and results
- Abstract Syntax Notation #1(ASN.1)
- The MIB definition tree
- Standard and proprietary MIBs
- Defining your own MIB

MIB encoding
- Basic Encoding Rules for ASN.1
- Application to object definitions and values

SNMP in the TCP/IP protocol suite
- SNMP PDU definitions
- Transport mappings

Module 4 : Managing Network Components

Managing interfaces

- Ethernet/IEEE 802.3
- Token Ring
- FDDI
- DS1, E1, X.25, DS3, frame relay

Managing network components: hubs, concentrators, bridges and routers

- Functional and practical needs
- Monitoring traffic levels
- Measuring error rates
- Address translations
- Routing tables
- Device and protocol configuration
- Operational statistics
- Component-specific MIBs and proprietary extensions

Managing the unmanageable

- Modems
- Proprietary devices

Module 5 : Managing Large Networks

Enterprisewide internetworks

- Management by proxy
- Need for multiple management stations
- Setting up management domains
- Manager-to-manager MIB
- Remote monitoring(RMON MIB)
- RMON-1 and RMON-2
- Security, authentication and privacy

Migrating from SNMPv1 to SNMPv2 or SNMPv3

- MIB compatibility
- Dual protocol agents
- Strategic considerations

Module 6 : Managing Systems

PCs and workstations

- PC network management needs
- Managing user workstations with the HR MIB
- Using SNMP in non-TCP/IP environments

Window NT

- Monitoring IP address allocation with the DHCP MIB
- Using SNMP to manage the WINS Server

Module 7 : Performance And Troubleshooting

Performance analysis

- Using SNMP to gather data for performance modeling
- Impact of SNMP on performance

Troubleshooting the network

- Determining deviations from normal
- Identifying the problem source
- Taking corrective action
- Troubleshooting methodology and tools

Module 8 : Network Manager Platforms

Current capabilities & future directions

- SUNnet Manager
- HP Open View
- IBM NetView/AIX
- Other players

Το πρωτόκολλο προάγει τις εξής τέσσερις συναρτήσεις :

Get: χρησιμοποιείται από το διαχειριστή για να σώσει ένα θέμα από τη MIB τοου πράκτορα

Set: χρησιμοποιείται από το διαχειριστή για να θέσει μια αξία στη MIB του πράκτορα δηλ. όταν εγώ επηρεάζω το δίκτυο

Trap: χρησιμοποιείται από το πράκτορα για να στείλει συναγερμό στο διαχειριστή. Είναι με άλλα λόγια ασύγχρονες αναφορές που μου δίνουν τα engine για να μου δείξουν ότι κάτι συμβαίνει

Inform: χρησιμοποιείται από το διαχειριστή για να στείλει συναγερμό σε έναν άλλον διαχειριστή

Αυτό είναι τόσο απλό όσο θα μπορούσαμε να επιτύχουμε. Αυτό που δίνει το πρωτόκολλο είναι η δύναμη του και το εκτεταμένο σύνολο της δομής της MIB την οποία την έχουμε ήδη ορίσει . Η MIB διατάσει το πράκτορα τη πληροφορία ο πράκτορας θα συλλέξει και θα αποθηκεύσει. Για παράδειγμα , υπάρχει ένας αριθμός ποικιλιών στη βάση της MIB η οποία σχετίζεται τη λειτουργία τη βασική των πρωτοκόλλων του TCP και του IP , περιλαμβάνοντας έναν αριθμό πακέτων τα οποία έχουν σταλεί ή έχουν ληφθεί, όπως επίσης πακέτα λανθασμένα κ.ο.κ.. Από τη στιγμή που όλοι οι πράκτορες συντηρούν το ίδιο σύνολο δεδομένων , οι εφαρμογές ή αιτήσεις (applications) γράφθηκαν στο σταθμό διαχείρισης για να εκθέσουν αυτή τη πληροφορία.

Υπάρχουν πέντε βασικές ομάδες:

SNMPv 1 statistics

SNMPv 2 statistics

Object resources

Traps

Set

Τα πλεονεκτήματα του SNMPv2 πάνω στο SNMPv1 είναι τα εξής:

- Επεκτεταμένη τύποι δεδομένων: 64bit counter
- Αποδεικνύει επάρκεια και παρουσίαση :get-bulk operator
- Επιβεβαιώνει το γεγονός της κοινοποίησης : inform operator
- Πλουσιότερος τρόπος χειρισμού του λάθους: errors and exceptions
- Αποδεικνύει σύνολα: especially row creation and deletion
- Πολύ καλά προσαρμοσμένη γλώσσα ορισμού δεδομένων

Τα χαρακτηριστικά της ασφάλειας είναι :

- Επικύρωση
- Απομόνωση ή ιδιωτικοποίηση
- Εξουσιοδότηση και έλεγχος πρόσβασης και τέλος
- Καταλληλότητα για απομακρυσμένο σχηματισμό και ικανότητα διοίκησης για αυτά τα χαρακτηριστικά

Το SNMPv3 επομένως είναι ίσο με :

SNMPv3=SNMPv2+security +administration

- *Security*

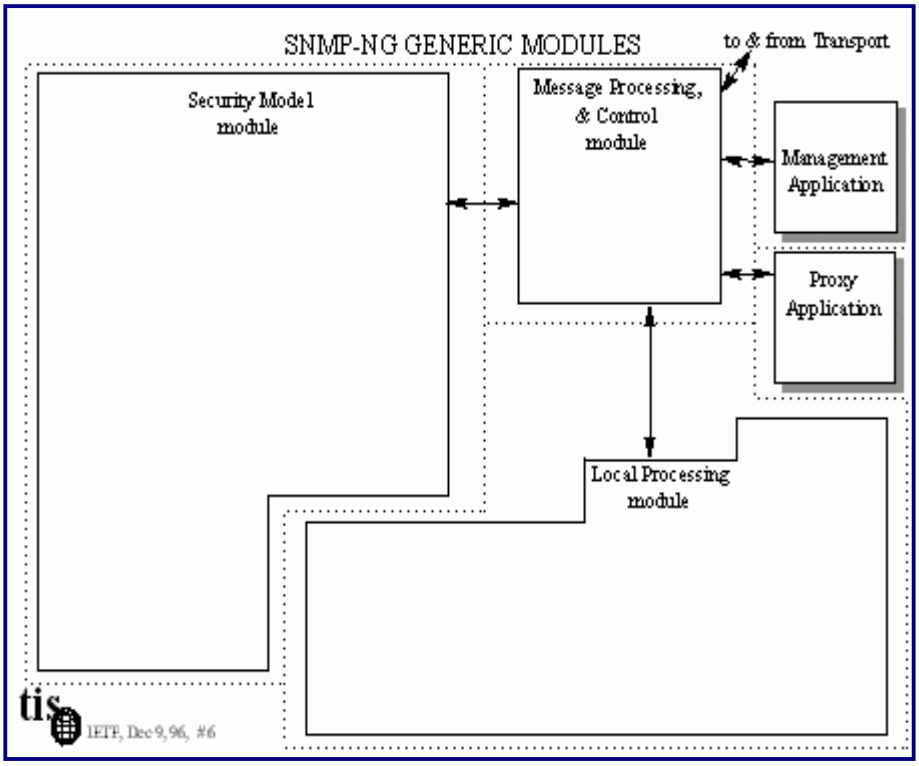
- Επικύρωση και απομόνωση
- Εξουσιοδότηση και έλεγχος πρόσβασης

- *Administrative framework*

- Ονομασία των οντοτήτων
- Άνθρωποι και πολιτικές
- Ονόματα χρηστών και κλειδί διαχείρησης
- Κοινοποίηση αποστάσεων
- Πληρεξούσιες σχέσεις
- Απομακρυσμένη σχηματοποίηση μέσω του SNMP λειτουργιών

Από τη μελέτη πολλών εργασιών καταλήξαμε ότι το SNMPv3 αποτελείται από τρία βασικά τμήματα τα οποία βέβαια αλλάζουν ανάλογα με τις ανάγκες της επιχείρησης ή του οργανισμού αυτά είναι :

- Message processing & control module
- Security model module
- Local processing module



άποψη του αρχιτεκτονικού μοντέλου του SNMP3

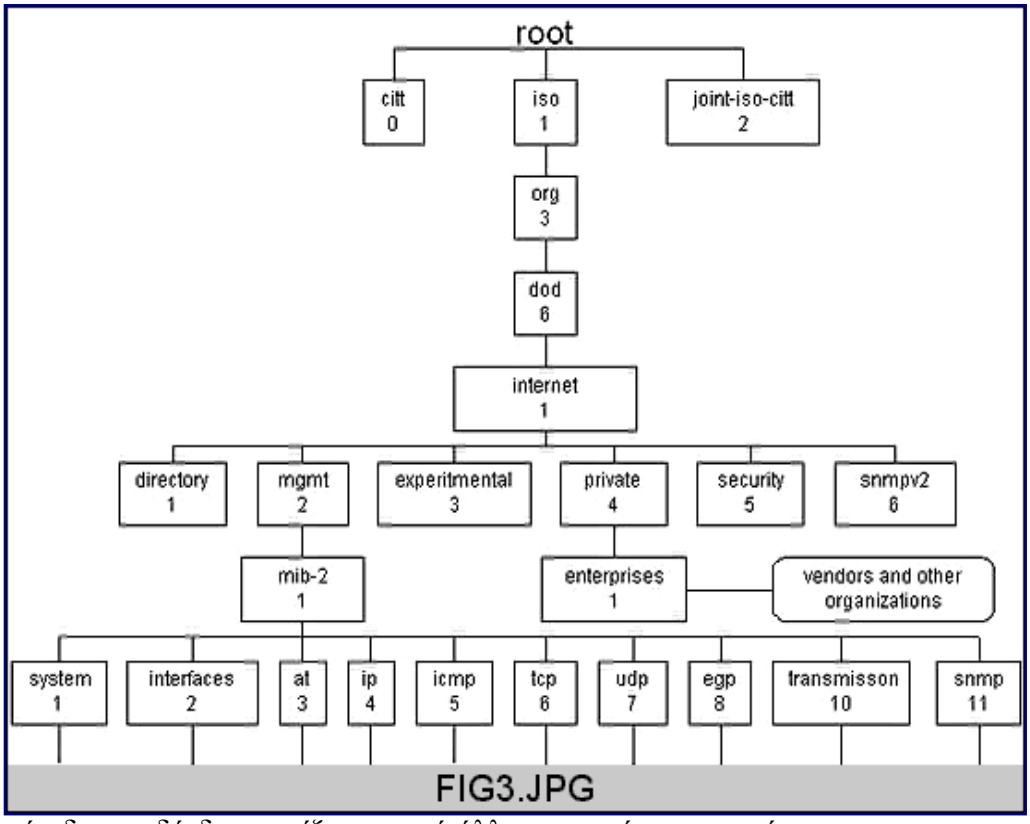
τα τρία τμήματα παίζουν το καθένα ξεχωριστά το δικό του ξεχωριστό ρόλο στην αρχιτεκτονική του πρωτοκόλλου. Το πιο σημαντικό είναι το message processing & control module, επειδή αυτό το τμήμα αλληλεπιδρά με τα άλλα δύο τμήματα και πιθανόν με τις εφαρμογές του διαχειριστή , τις εφαρμογές του πληρεξούσιου , και το επίπεδο μεταφοράς. Το message processing & control module κρίνει σε πια τμήματα ή εφαρμογές ένα μήνυμα δίνεται όταν στέλνεται ή λαμβάνεται. Στα παρακάτω κεφάλαια όλα τα τρία τμήματα θα εξηγηθούν αναλυτικά αλλά επειδή είναι το πιο σημαντικό θα το εξηγήσουμε πρώτο.

Αλλά προηγουμένως ας κάνουμε μια αναφορά στη MIB όπως είχαμε υποσχεθεί.

Management Information Base (MIB)

Είναι βολικό να σκεφθούμε το σχηματισμό , τη κατάσταση , και τις στατιστικές πληροφορίες μέσα σε μια μηχανή καθώς καθορίζουμε τη βάση δεδομένων αυτής. Στη πραγματικότητα η πληροφορία ίσως να αποθηκεύεται μέσα σε μια μηχανή σαν ένας συνδυασμός αγώγιμων συνόλων, hardware counters, in memory variables, in memory tables or files. Στο πρότυπο SNMP αυτή η λογική βάση δεδομένων του δικτύου διαχείρισης πληροφορίας καλείται Management Information Base (MIB). Δεν είναι τόσο σημαντικό το πώς τα δεδομένα αποθηκεύονται εσωτερικά στη μηχανή. Εμείς ενδιαφερόμαστε μόνο για την αποδοχή των δεδομένων . Όπως ορίσαμε και προηγουμένως η MIB αποτελείται από ένα σύνολο στοιχείων που ονομάζονται διαχειρίσιμα αντικείμενα που περιέχουν πληροφορίες διαχείρισης συγκεκριμένων κόμβων .

Η MIB έχει τη δική της ιεραρχική δομή. Υπάρχουν γενικής αρχής και ειδικά για πώληση διαχειρίσιμα αντικείμενα στη MIB. Πάνω από 1,000 διαφορετικά αντικείμενα έχουν εγγραφεί με τη κοινότητα του Internet σαν διαχειρίσιμα αντικείμενα. Ένα διαχειρίσιμο αντικείμενο είναι μια λογική αντιπροσώπευση μιας πραγματικά αληθινής φυσικής ενότητας πάνω στο δίκτυο. Κάθε αντικείμενο συλλέγει μια εξειδικευμένη πληροφορία με το ρόλο του πάνω στο δίκτυο. Η MIB έχει δομή δένδρου , και η κορυφή του δένδρου ορίζεται από το ISO όπως φαίνεται στο πιο κάτω σχήμα . τα χαμηλότερα



επίπεδα του δένδρου ορίζονται από άλλες οργανώσεις και κάποια υποκαταστήματα ορίζουν τα ειδικά για πώληση αντικείμενα.

ISO/CITT δένδρο object identifiers

Τα αντικείμενα που θέλουμε να διαχειριστούμε αντιπροσωπεύονται από κόμβους φυλλώματα στο γενικό ISO/CITT δένδρο. Κάθε κόμβος στο δένδρο έχει μια ετικέτα η οποία αποτελείται από ένα ακέραιο και το κύριο τμήμα περιγραφής. Η αναγνώριση του αντικειμένου είναι μια σειρά από ακέραιους οι οποίοι υπογραμμίζουν το μονοπάτι από τη ρίζα προς το αντικείμενο. Στη γλώσσα των κατασκευαστών αυτό καλείται ως OBJECT IDENTIFIER (αντικείμενο αναγνώρισης). Το μέρος του κειμένου της ετικέτας του κάθε κόμβοου έχει τη πρόθεση να βοηθήσει τους ανθρώπους που ίσως έχουν πρόβλημα στο να θυμούνται και να αναγνωρίζουν μεγάλα strings αριθμών . Αυτή η φιλική φόρμα για το χρήστη από τον identifier συχνά γράφεται σαν μια σειρά από ετικέτες με υπογράμμιση.

To string

Iso_org_dod_internet_mgmt_mib2_interfaces_ifTable_ifEntry_ifOperStatus

Αναγνωρίζει τη λειτουργική κατάσταση της διασύνδεσης. Και φυσικά είναι πιο εύκολο να το καταλάβουμε από το 1.3.6.1.2.1.2.2.1.8. Ο instance identifier or variable name χρησιμοποιείται για να δηλώσει την ακριβή αξία την οποία θέλουμε να δούμε. Η μεταβλητή έχει τη φόρμα :

OBJECT IDENTIFIER . which one

Σ' αυτή τη περίπτωση, για να πάρουμε τη συγκεκριμένη διασύνδεση , προσθέτουμε το τελευταίο ψηφιακό στον identifier. Το μονοπάτι του ifOperStatus είναι:

1.3.6.1.2.1.2.2.1.8.

εάν θέλουμε τη λειτουργική κατάσταση της *τρίτης* διασύνδεσης της μηχανής , ζητούμε για:

1.3.6.1.2.1.2.2.1.8.*3*

Συχνά η πληροφορία του δικτύου έχει μια φυσική δομή σαν μία σειρά από γραμμές και στήλες που είναι οργανωμένες σε ένα πίνακα. Κάθε σειρά στο πίνακα καλείται *entry*.Ενας πίνακας δεν χρησιμοποιείται συχνά χωρίς το index του το οποίο σου επιτρέπει να βγάλεις έξω μια συγκεκριμένη σειρά του πίνακα. Το φυσικό index σχετίζεται με τα δεδομένα που είναι αποθηκευμένα στο πίνακα. Για παράδειγμα οι διευθύνσεις του IP θα ήταν τέλειοι δείκτες για τη δρομολόγηση του πίνακα. Ο σκοπός του index είναι να αναγνωρίζει ένα συγκεκριμένο entry μέσα στο πίνακα. Οι άνθρωποι που είναι οικείοι με τις πινακοειδής βάσεις δεδομένων γνωρίζουν ότι κάποιες φορές ότι ο index πρέπει να φτιάχνεται από περισσότερες από μια στήλες. Με άλλα λόγια δεν μπορείς να ξεχωρίσεις μια μοναδική γραμμή έκτός και αν καθορίσεις τις αξίες ορισμένων στηλών.

Η πρώτη MIB , σήμερα καλείται σαν **MIB-I** συγκεντρώνεται σε εξειδικευμένες πληροφορίες του TCP/IP. Εδώ είναι κάποιες *μεταβλητές* δείγματα τα οποία το γνήσιο MIB περιλαμβάνει:

- Τη *περιγραφή συστήματος*
- *Τις διευθύνσεις του IP σε συνεργασία με τη διασύνδεση*
- *Ο αριθμός των διασυνδέσεων του δικτύου*
- *Η μέτρηση των εισόδων και των εξόδων των σχεδιαγραμμάτων*
- *Ένα πίνακα των ενεργητικών συνδέσεων του TCP*

Επομένως οτιδήποτε κάνω πρέπει να ακολουθεί μια ορισμένη τυποποίηση σύμφωνα με το MIB. Τα φύλλα του MIB είναι συγκεκριμένα objects και δεν μπορεί κάποιος να επηρεάσει τη δενδροειδή μορφή και τα φύλλα της MIB. Η ομάδα MIB αποτελείται από ορισμούς οι οποίοι μπορούν να χρησιμοποιηθούν περισσότερες από μια φορές.

STRUCTURE OF MANAGEMENT INFORMATION (SMI)

Η MIB ορίζεται από τη SMI, η οποία προάγει την έννοια της δημιουργίας και της αναγνώρισης των τύπων των δεδομένων και τις περιλήψεις των αντιπροσωπευτικών δεδομένων. Πιθανόν οι τύποι των δεδομένων είναι ως εξής integer values, network addresses, counters, gauges, time ticks, table information. **SMI** επίσης ορίζει μια ιεραρχική δομή ονομασίας για την αναγνώριση κάθε αντικειμένου διαχείρισης. Τα αντικείμενα έχουν συγκεκριμένα ονόματα και η τοποθεσία στην ιεραρχική δομή μας βοηθά στο να αναγνωρίσουμε το τι κάνει το αντικείμενο και ποια είναι η ρίζα του. Τα αντικείμενα επίσης περιλαμβάνουν πεδία τα οποία ορίζουν πως η πληροφορία αποθηκεύεται και πως η πληροφορία έγινε αποδεκτή(read-only, read/write). Ένα αντικείμενο ορίζεται μέσω του SMI και μεταγλωττίζεται για να δημιουργήσει κωδικό ο οποίος να μπορεί να ενοποιηθεί με ένα πράκτορα ο οποίος τρέχει πάνω σε μια συγκεκριμένη μηχανή. Το αντικείμενο τότε συλλέγει την ορισμένη πληροφορία και τη κάνει κατάλληλη στα συστήματα διαχείρισης.

MIB –II

Μετά την ανάπτυξη μετώπου στον αληθινό κόσμο, οι βασικοί ορισμοί MIB που διευκρινίστηκαν και πολλοί νέοι ορισμοί που προστέθηκαν, τα αποτελέσματα είναι κοινώς αποδεκτά σαν **MIB-II** η οποία αποδείχθηκε να είναι πιο δυνατή βάση για τη διαχείριση του TCP/IP.

MESSAGE PROCESSING AND CONTROL MODULE

Μια SNMPv3 protocol engine ΠΡΕΠΕΙ να υποστηρίζει τουλάχιστον ένα Message Processing Model. Μια SNMPv3 protocol engine ΙΣΩΣ να υποστηρίζει περισσότερα από ένα , για παράδειγμα σ' ένα multilingual σύστημα το οποίο παρέχει στιγμιαία υποστήριξη στα SNMPv3, SNMPv1, SNMPv2c.

Η παραγωγή του μηνύματος στα διαφορετικά τμήματα θα συζητηθεί για κάθε τύπο μηνύματος κάθε φορά. Αυτά είναι:

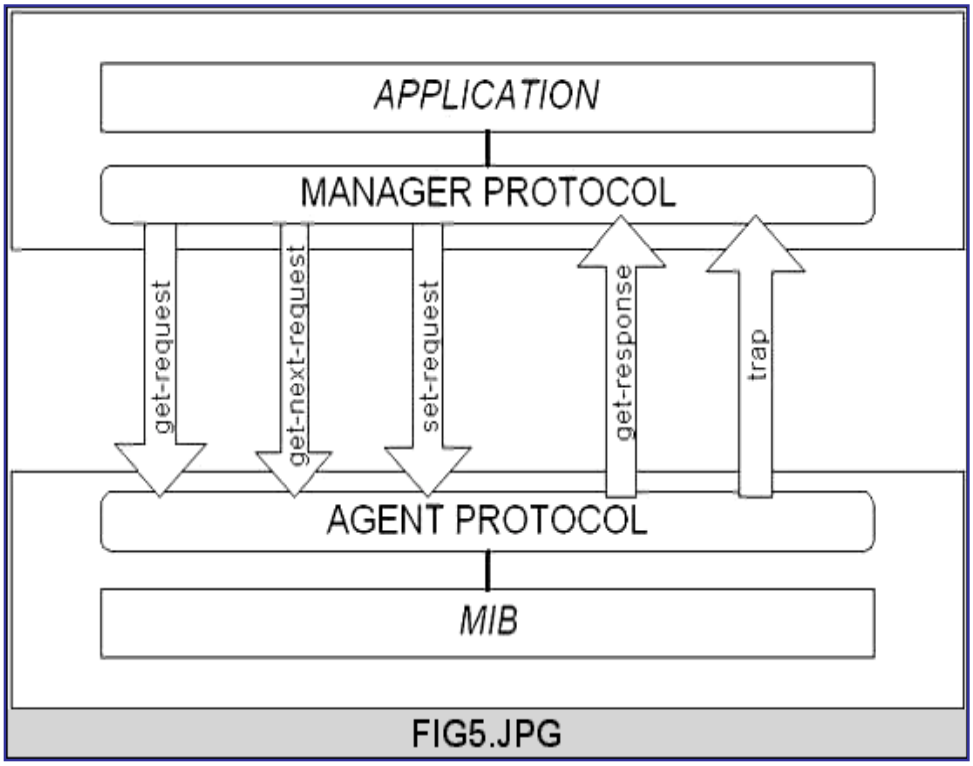
- **Request.** Ο διαχειριστής στέλνει ένα μήνυμα ερώτηση στο πράκτορα εάν ο διαχειριστής θέλει να πάρει ή να αλλάξει τη πληροφορία η οποία έχει αποθηκευθεί στο πράκτορα.
- **Inform.** Η inform βασικά είναι η ίδια με τη request , αλλά η inform στέλνεται από το διαχειριστή σε έναν άλλο διαχειριστή και όχι στον πράκτορα .
- **Response.** Έναν πράκτορας απαντά με ένα μήνυμα απάντηση σε όλα τα request μηνύματα .Ένα τέτοιο μήνυμα περιέχει είτε τα δεδομένα της ζήτησης όπως τα

- Get-request: σώζει μια ή περισσότερες αξίες από το κόμβο MIB διαχείρισης
- Get-next-request:κάνει ικανό το διαχειριστή να σώσει αξίες διαδοχικά. Μια δημοφιλής χρήση της εντολής αυτής είναι να διαβάζει τις σειρές του πίνακα.

- Set-request: κάνει ικανό το διαχειριστή να βελτιώνει τις κατάλληλες μεταβλητές.
- get-response:επιστρέφει τα αποτελέσματα μιας get-request , get-next-request, set request operation
- trap: κάνει ικανό το πράκτορα να αναφέρει αυθόρμητα σημαντικά γεγονότα ή προβλήματα
- get bulk request

ή απλά επικυρώνει ότι οι αξίες έχουν αλλάξει στη περίπτωση του set request a response επίσης επιστρέφεται από ένα διαχειριστή σε απάντηση στο inform

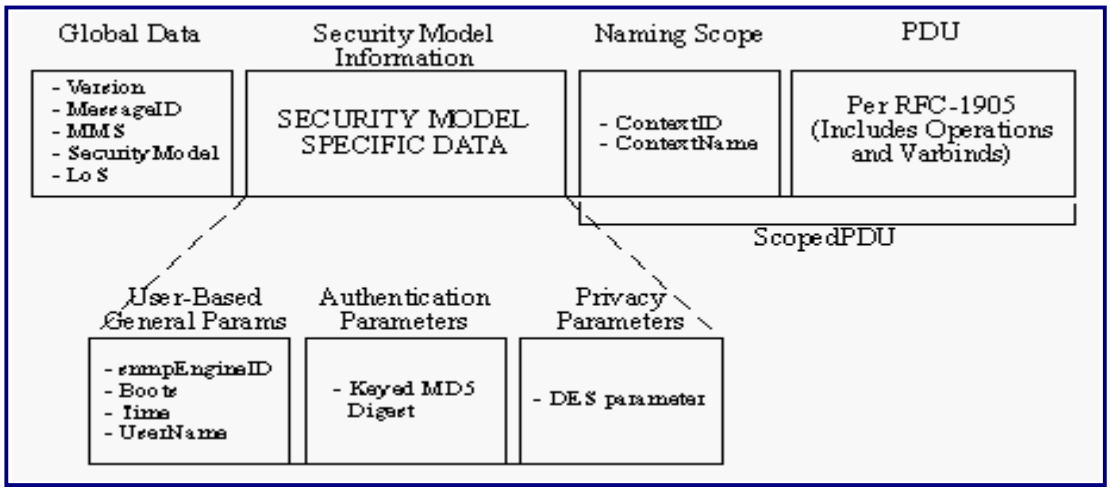
- **Trap.** Μια εφαρμογή μπορεί να αποφασίσει να στείλει ένα trap μήνυμα στο διαχειριστή όταν συγκεκριμένα γεγονότα συμβαίνουν. Ένα τέτοιο μήνυμα συνήθως περιέχει μια πληροφορία κατεστημένη και δεν υπάρχει απάντηση στο trap.
- **Report.** Εάν ένα λάθος συμβεί κατά τη διάρκεια της επεξεργασίας του λαμβανόμενου request ή inform message,το report μήνυμα ίσως να δημιουργηθεί και να σταλθεί πίσω στη ενότητα που το έστειλε.



Μυνήματα snmp

Τα πρώτα πέντε πεδία που δημιουργούνται από το μοντέλο επεξεργασίας μηνύματος σε εξερχόμενα μηνύματα και σε εσερχόμενα. :

- **msgVersion:** Set to snmpv3(3).
- **msgID:** A unique identifier used between two SNMP entities to coordinate request and response messages, and by the message processor to coordinate the processing of the message by different subsystem models within the architecture. The range of this ID is 0 through 231-1.
- **msgMaxSize:** Conveys the maximum size of a message in octets supported by the sender of the message, with a range of 484 through 231-1. This is the maximum segment size that the sender can accept from another SNMP engine (whether a response or some other message type).



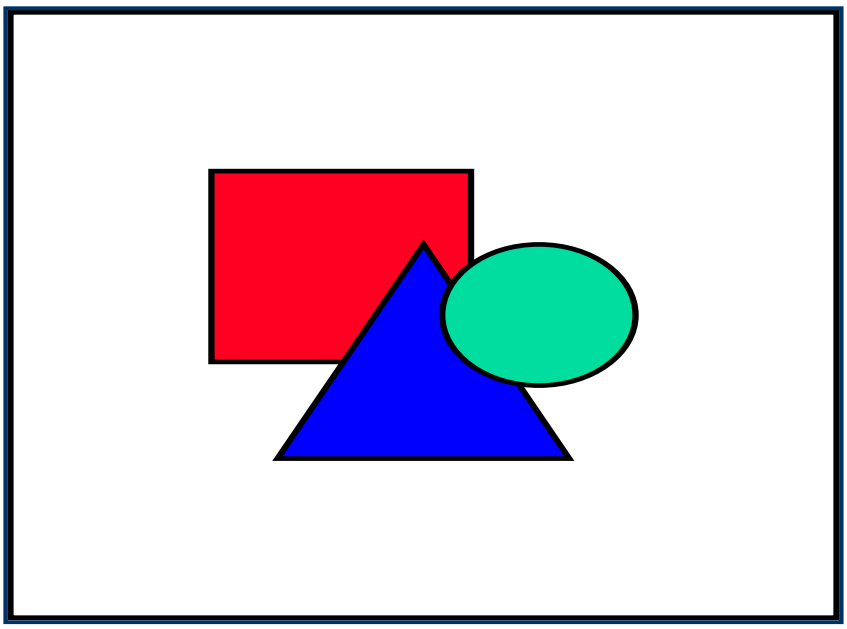
- **msgFlags:** An octet string containing three flags in the least significant three bits: reportableFlag, privFlag, authFlag. If reportableFlag = 1, then a Report PDU must be returned to the sender under those conditions that can cause the generation of a Report PDU; when the flag is zero, a Report PDU may not be sent. The reportableFlag is set to 1 by the sender in all messages containing a request (Get, Set) or an Inform, and set to 0 for messages containing a Response, a Trap, or a Report PDU. The reportableFlag is a secondary aid in determining when to send a Report. It is only used in cases in which the PDU portion of the message cannot be decoded (e.g., when decryption fails due to incorrect key). The privFlag and authFlag are set by the sender to indicate the security level that was applied to the message. For privFlag = 1, encryption was applied and for privFlag = 0, authentication was applied. All combinations are allowed except (privFlag = 1 AND authFlag = 0); that is, encryption without authentication is not allowed.
- **msgSecurityModel:** An identifier in the range of 0 through 231-1 that indicates which security model was used by the sender to prepare this message and therefore which security model must be used by the receiver to process this message. Reserved values include 1 for SNMPv1, 2 for SNMPv2c, and 3 for SNMPv3.

Το message processing control module (MPC) αποτελεί το κέντρο της SNMPv3 engine μηχανής. Παίρνει μηνύματα από το επίπεδο μεταφοράς ,

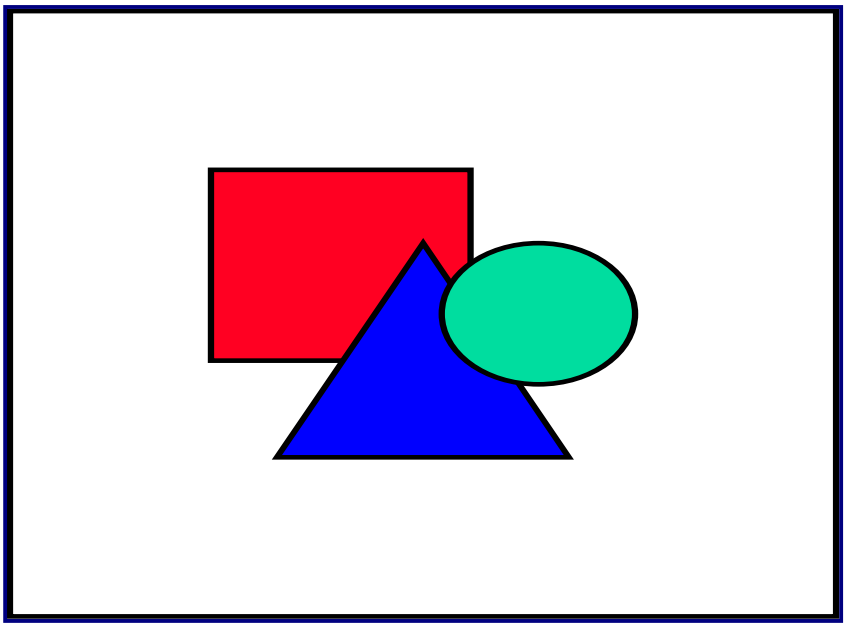
αφήνει το κατάλληλο τμήμα ασφάλειας να ελέγξει το μήνυμα και στη συνέχεια περνάει το αναλυμένο αποτέλεσμα στο PDU (το μέρος εκείνο του μηνύματος που περιέχει τη πληροφορία της διαχείρισης) στο τμήμα της τοπικής επεξεργασίας (local processing module) ή στην εφαρμογή που αποσκοπεί .Φυσικά αυτή η επεξεργασία μπορεί να γίνει αντιστρόφως , τότε παίρνει το αντικείμενο από το PDU από το τμήμα της τοπικής επεξεργασίας ή τη φόρμα της εφαρμογής , αφήνει το κατάλληλο τμήμα ασφάλειας να το ασφαλίσει και στη συνέχεια το περνάει στο επίπεδο μεταφοράς .

Η ΕΠΕΞΕΡΓΑΣΙΑ ΤΩΝ ΕΞΕΡΧΟΜΕΝΩΝ ΜΗΝΥΜΑΤΩΝ

Μετά από τη πρώτη φάση, το MPC module περνάει τα δεδομένα στο επιλεγμένο τμήμα ασφάλειας. Στη συνέχεια αυτό θα προσθέσει την επικύρωση και θα αποκρύψει το μήνυμα εάν αυτό ζητηθεί. Το τμήμα της ασφάλειας επιστρέφει με ένα ολοκληρωμένο μήνυμα. Το μήνυμα τότε είναι έτοιμο για μεταφορά και περνά στο επίπεδο μεταφοράς, μαζί με τη διεύθυνση της κατεύθυνσης.



εξερχόμενα μηνύματα



εισερχόμενα μηνύματα

Εάν η πληρεξούσια /διαχειριστική εφαρμογή ή το τμήμα της τοπικής επεξεργασίας θέλει να στείλει ένα μήνυμα, θα το περάσει το μήνυμα από το τμήμα του MPC. Αυτό στη συνέχεια θα στείλει το μήνυμα στο τμήμα ασφάλειας για να το ασφαλίσει , και στη συνέχεια θα περάσει στο επίπεδο μεταφοράς το οποίο θα φέρει το μήνυμα στη κατεύθυνση του. Εάν ένα μήνυμα λαμβάνεται , πρώτα το τμήμα του MPC θα στείλει μήνυμα στο τμήμα ασφάλειας για να το ασφαλίσει. Όταν αυτό γίνει τότε θα περάσει στην εφαρμογή του πληρεξούσιου , στην εφαρμογή της διαχείρισης ή στο τμήμα της τοπικής επεξεργασίας.

REQUEST MESSAGE

Όταν μια αίτηση εφαρμογής θέλει να στείλει ένα μήνυμα αναφοράς , περνά από το scoped PDU , a LoS, a maximum message size (MMS) , a destination address, and a security cookie to the MPC module. Το τμήμα ασφάλειας μπορούμε να το δούμε σαν ένα είδος ασφάλειας ανεξάρτητο από την ενότητα της ασφάλειας αναγνώρισης. Εξαρτάται από του εκτελεστές το πώς θα ορίσουν το τμήμα της ασφάλειας, εφόσον είναι το μοναδικό πιστοποιητικό αναγνώρισης.

Το MPC τμήμα αρχίζει με τη δημιουργία ταυτότητας id του μηνύματος. Κάθε μήνυμα πρέπει να κουβαλάει μια ταυτότητα μηνύματος (message ID) μοναδική στην ενότητα που στέλνεται. Αυτό σε ένα μήνυμα ερώτησης αντιγράφεται σε ένα μήνυμα απάντησης , αυτή είναι η SNMPv3 engine η οποία

λαμβάνει απάντηση η οποία κρίνει ποια εφαρμογή στάλθηκε από τη γνήσια ερώτηση.

Μετά τη δημιουργία ταυτότητας του μηνύματος το τμήμα MPC θα πάρει τα γενικά δεδομένα (version 3 , message ID, Los, MMS and security model number), η διεύθυνση κατεύθυνσης, το τμήμα ασφάλειας, scoped PDU, cache this data. Εάν ένα λάθος αναφοράς ληφθεί , ή εκτός χρόνου κάτι συμβεί, τότε το τμήμα που μελετάμε ξαναστέλνει το μήνυμα. Μετά από αυτό, τα γενικά δεδομένα , και το scoped PDU είναι ber-encode.Η επεξεργασία τότε συνεχίζεται στο τμήμα της ασφάλειας.

INFORM MESSAGE

Η επεξεργασία του μηνύματος αυτού δεν διαφέρει πολύ από το μήνυμα αναφοράς. Η μόνη διαφορά εδώ είναι η επικοινωνία διαχειριστή με διαχειριστή, και η ερώτηση –αναφορά είναι για επικοινωνία διαχειριστή προς πράκτορα.

RESPONSE MESSAGE

Όταν το τμήμα MPC λαμβάνει ένα μήνυμα αναφοράς, το περνάει στο τμήμα τοπικής επεξεργασίας. Το τμήμα αυτό στη συνέχεια θα δημιουργήσει μια απάντηση . Πριν όμως το περάσει στο τμήμα τοπικής επεξεργασίας, το MPC σώζει όλα τα γενικά δεδομένα που βρίσκονται μέσα στο μήνυμα. Μετά τη λήψη της απάντησης PDU γυρίζει πίσω από το τμήμα τοπικής επεξεργασίας, το MPC τότε χρησιμοποιεί τα δεδομένα για να δημιουργήσει ένα μήνυμα με τα ίδια γενικά δεδομένα όπως στη γνήσια αναφορά ή αίτηση. Ξανά τα γενικά δεδομένα και το scoped PDU αποκρύπτονται και περνούν στο κατάλληλο τμήμα ασφαλείας. Κανένα σώσιμο οποιουδήποτε αρχείου παίρνει μέρος εδώ , έτσι το χαμένο μήνυμα αίτησης δεν μπορεί να ξανασταλθεί.

TRAP MESSAGE

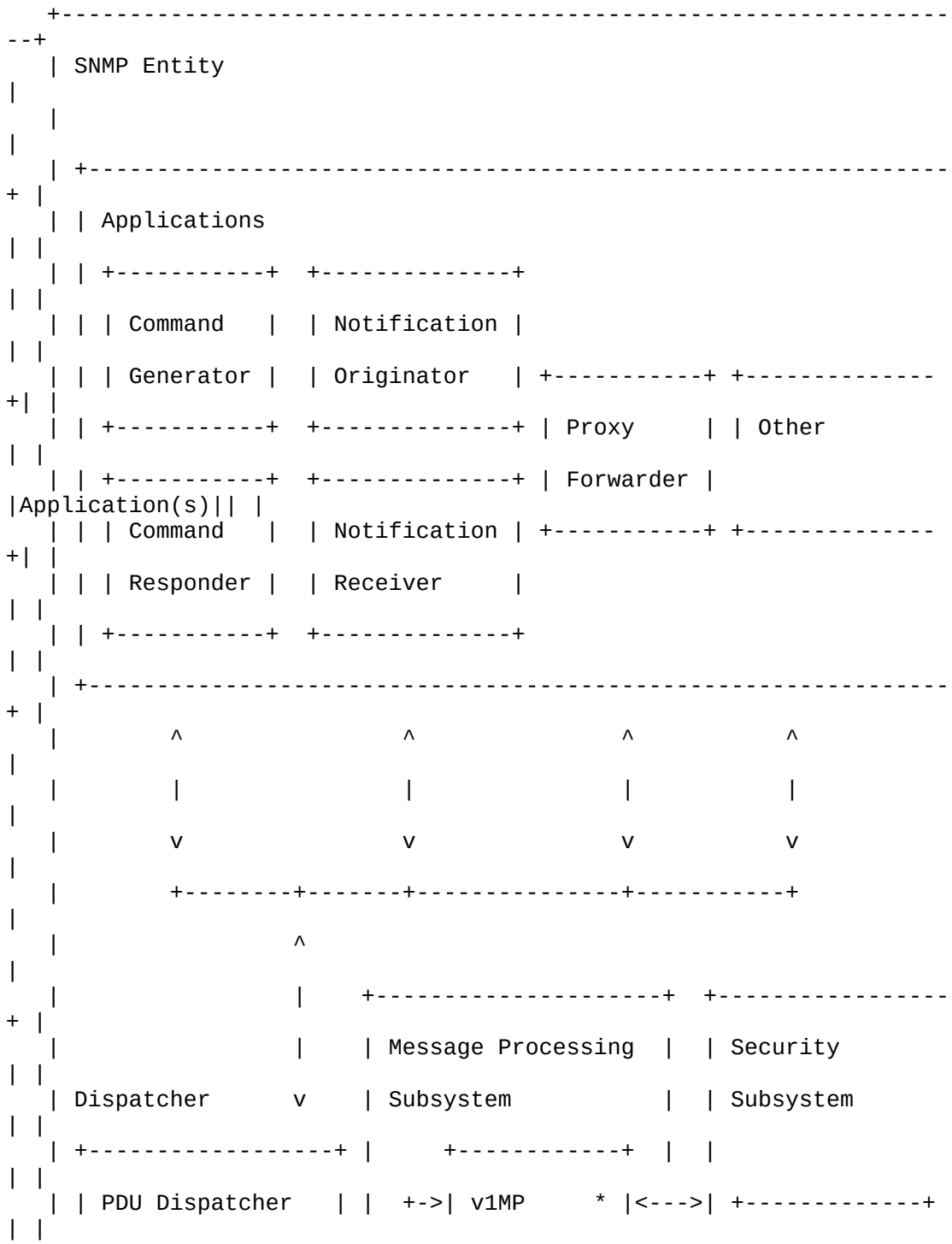
Σε οποιοδήποτε λεπτό μια εφαρμογή μπορεί να αποφασίσει να στείλει πληροφορία στο διαχειριστή. Αυτό γίνεται με το trap message μια εφαρμογή περνά το SCOPED PDU που περιέχει τη πληροφορία

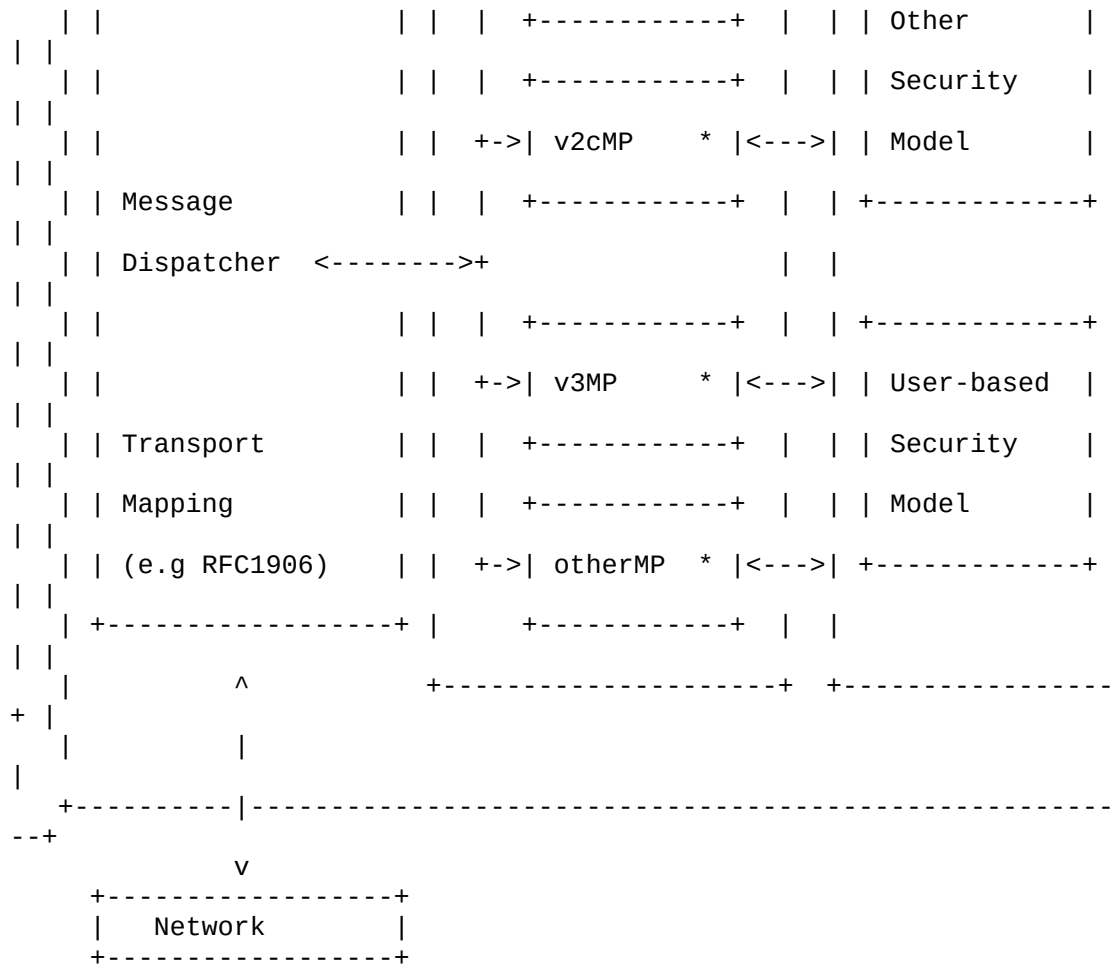
REPORT MESSAGE

Όταν συμβαίνει ένα λάθος κατά τη διάρκεια της επεξεργασίας ενός λαμβάνων μηνύματος το μήνυμα αναφοράς μπορεί να γεννηθεί και να επιστρέψει στον αποστολέα του μηνύματος. Το MPC τμήμα θα αρχίσει με το σκηνικό των λαθών στο PDU. Στη συνέχεια το MPC τμήμα θα ελέγξει εάν η αίτηση ID από το λαμβάνων τμήμα είναι γνωστή ή όχι. Εάν είναι γνωστή , αυτή η αξία χρησιμοποιείται , εάν δεν είναι η requestID in the PDU θα μπεί στην αξία 214183647.

Η ΕΠΕΞΕΡΓΑΣΙΑ ΤΩΝ ΕΙΣΕΡΧΟΜΕΝΩΝ ΜΗΝΥΜΑΤΩΝ

Η παρακάτω απεικόνιση παρουσιάζει το Message Processing σε σχέση με τις SNMP applications, the Security Subsystem and Transport Mappings.





Στοιχεία του Message Processing and Dispatching

See [SNMP-ARCH] for the definitions of
contextEngineID
contextName
scopedPDU
maxSizeResponseScopedPDU
securityModel
securityName
securityLevel
messageProcessingModel

For incoming messages, a version-specific message processing module
provides these values to the Dispatcher. For outgoing messages, an

application provides these values to the Dispatcher.

For some version-specific processing, the values may be extracted from received messages; for other versions, the values may be determined by algorithm, or by an implementation-defined mechanism.
The mechanism by which the value is determined is irrelevant to the Dispatcher.

The following additional or expanded definitions are for use within the Dispatcher.

messageProcessingModel

The value of messageProcessingModel identifies a Message Processing Model. A Message Processing Model describes the version-specific procedures for extracting data from messages, generating messages, calling upon a securityModel to apply its security services to messages, for converting data from a version-specific message format into a generic format usable by the Dispatcher, and for converting data from Dispatcher format into a version-specific message format.

pduVersion

The value of pduVersion represents a specific version of protocol operation and its associated PDU formats, such as SNMPv1 or SNMPv2 [RFC1905]. The values of pduVersion are specific to the version of the PDU contained in a message, and the PDUs processed by applications. The Dispatcher does not use the value of pduVersion directly.

For incoming messages, pduVersion is provided to the Dispatcher by a version-specific Message Processing module. The PDU Dispatcher passes the pduVersion to the application so it knows how to handle the PDU properly. For example, a command responder application needs to know whether to use [RFC1905] elements of procedure and syntax instead of those specified for SNMPv1.

pduType

A value of pduType represents a specific type of protocol operation.
The values of pduType are specific to the version of the PDU contained in a message.

Applications register to support particular pduTypes for particular contextEngineIDs.

```
   For incoming messages, pduType is provided to the Dispatcher by a
   version-specific Message Processing module. It is subsequently
used
   to dispatch the PDU to the application which registered for the
   pduType for the contextEngineID of the associated scopedPDU
```

sendPduHandle

```
   This handle is generated for coordinating the processing of
requests
   and responses between the SNMP engine and an application. The
handle
   must be unique across all version-specific Message Processing
Models,
   and is of local significance only.
```

The application requests this using the abstract service primitive:

```
       statusInformation =              -- sendPduHandle if success
                                        -- errorIndication if failure
         sendPdu(
         IN   transportDomain           -- transport domain to be
used
         IN   transportAddress          -- destination network
address
         IN   messageProcessingModel    -- typically, SNMP version
         IN   securityModel             -- Security Model to use
         IN   securityName              -- on behalf of this
principal
         IN   securityLevel             -- Level of Security
requested
         IN   contextEngineID           -- data from/at this entity
         IN   contextName               -- data from/in this context
         IN   pduVersion                -- the version of the PDU
         IN   PDU                       -- SNMP Protocol Data Unit
         IN   expectResponse            -- TRUE or FALSE
              )
returnResponsePdu(
       IN   messageProcessingModel   -- typically, SNMP version
       IN   securityModel            -- Security Model in use
       IN   securityName             -- on behalf of this principal
       IN   securityLevel            -- same as on incoming request
       IN   contextEngineID          -- data from/at this SNMP entity
       IN   contextName              -- data from/in this context
       IN   pduVersion               -- the version of the PDU
       IN   PDU                      -- SNMP Protocol Data Unit
       IN   maxSizeResponseScopedPDU -- maximum size of Response PDU
       IN   stateReference           -- reference to state
information
                                     -- as presented with the request
       IN   statusInformation        -- success or errorIndication
       )                             -- (error counter OID and value
                                     -- when errorIndication)
```

The Message Dispatcher sends the request to the appropriate Message Processing Model indicated by the received value of messageProcessingModel using the abstract service primitive:

```
      result =                        -- SUCCESS or errorIndication
       prepareResponseMessage(
```

```
    IN   messageProcessingModel   -- specified by application
    IN   securityModel            -- specified by application
    IN   securityName             -- specified by application
    IN   securityLevel            -- specified by application
    IN   contextEngineID          -- specified by application
    IN   contextName              -- specified by application
    IN   pduVersion               -- specified by application
    IN   PDU                      -- specified by application
    IN   maxSizeResponseScopedPDU -- specified by application
    IN   stateReference           -- specified by application
    IN   statusInformation        -- specified by application
    OUT  destTransportDomain      -- destination transport domain
    OUT  destTransportAddress     -- destination transport address
    OUT  outgoingMessage          -- the message to send
    OUT  outgoingMessageLength    -- the message length
         )
```

If the result is an errorIndication, the errorIndication is returned to the calling application. No further processing is performed.

If the result is success, the outgoingMessage is sent over the transport specified by the transportDomain to the address specified by the transportAddress.

Περιγραφή "Το MIB για τα Message Processing and Dispatching"

```
        ::= { snmpModules 11 }

    -- Administrative assignments
 ****************************************

    snmpMPDAdmin           OBJECT IDENTIFIER ::= { snmpMPDMIB 1
 }
    snmpMPDMIBObjects      OBJECT IDENTIFIER ::= { snmpMPDMIB 2
 }
    snmpMPDMIBConformance  OBJECT IDENTIFIER ::= { snmpMPDMIB 3
 }

    -- Statistics for SNMP Messages
 ************************************

    snmpMPDStats           OBJECT IDENTIFIER ::= {
 snmpMPDMIBObjects 1 }
snmpUnknownSecurityModels OBJECT-TYPE
    SYNTAX       Counter32
    MAX-ACCESS   read-only
    STATUS       current
```

Περιγραφή ΄΄ο συνολικός αριθμός των πακέτων που παρέλαβε ο SNMP engine τα οποία απορρίφθηκαν επειδή συσχετίστηκαν από το securityModel το οποίο δεν ήξερε και υποστήριζε την SNMP engine’’

```
    ::= { snmpMPDStats 1 }

snmpInvalidMsgs OBJECT-TYPE
    SYNTAX       Counter32
```

```
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The total number of packets received by the SNMP
                    engine which were dropped because there were
invalid
                    or inconsistent components in the SNMP message.
                   "
       ::= { snmpMPDStats 2 }

   snmpUnknownPDUHandlers OBJECT-TYPE
       SYNTAX       Counter32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The total number of packets received by the SNMP
                    engine which were dropped because the PDU
contained
                    in the packet could not be passed to an
application
                    responsible for handling the pduType, e.g. no
SNMP
                    application had registered for the proper
                    combination of the contextEngineID and the
pduType.
                   "
       ::= { snmpMPDStats 3 }
```

The SNMPv3 Message Format

SNMP version 3 Message Processing Model (v3MP).

```
SNMPv3MessageSyntax DEFINITIONS IMPLICIT TAGS ::= BEGIN

    SNMPv3Message ::= SEQUENCE {
        -- identify the layout of the SNMPv3Message
        -- this element is in same position as in SNMPv1
        -- and SNMPv2c, allowing recognition
        msgVersion INTEGER { snmpv3 (3) },
        -- administrative parameters
        msgGlobalData HeaderData,
        -- security model-specific parameters
        -- format defined by Security Model
        msgSecurityParameters OCTET STRING,
        msgData  ScopedPduData
    }

    HeaderData ::= SEQUENCE {
        msgID      INTEGER (0..2147483647),
        msgMaxSize INTEGER (484..2147483647),

        msgFlags   OCTET STRING (SIZE(1)),
                   --  .... ...1   authFlag
                   --  .... ..1.   privFlag
                   --  .... .1..   reportableFlag
                   --              Please observe:
                   --  .... ..00   is OK, means
       noAuthNoPriv
                   --  .... ..01   is OK, means authNoPriv
                   --  .... ..10   reserved, must NOT be
       used.
                   --  .... ..11   is OK, means authPriv
```

```
                msgSecurityModel INTEGER (0..2147483647)
            }

            ScopedPduData ::= CHOICE {
                plaintext    ScopedPDU,
                encryptedPDU OCTET STRING  -- encrypted scopedPDU
      value
            }

            ScopedPDU ::= SEQUENCE {
                contextEngineID  OCTET STRING,
                contextName      OCTET STRING,
                data             ANY -- e.g., PDUs as defined in
      RFC1905
            }
         END
```

Authoritative SNMP engine

In order to protect against message replay, delay and redirection, one of the SNMP engines involved in each communication is designated to be the authoritative SNMP engine. When an SNMP message contains a payload which expects a response (those messages that contain a + Confirmed Class PDU [RFC-ARCH]), then the receiver of such messages is authoritative. When an SNMP message contains a payload which does not expect a response (those messages that contain an Unconfirmed + Class PDU [RFC-ARCH]), then the sender of such a message is authoritative.

Mechanisms

The following mechanisms are used:

1) To protect against the threat of message delay or replay (to an extent greater than can occur through normal operation), a set of timeliness indicators (for the authoritative SNMP engine) are included in each message generated. An SNMP engine evaluates the timeliness indicators to determine if a received message is recent. An SNMP engine may evaluate the timeliness indicators to ensure that a received message is at least as recent as the last message it received from the same source. A non-authoritative SNMP engine uses received authentic messages to advance its notion

of the timeliness indicators at the remote authoritative source.

An SNMP engine MUST also use a mechanism to match incoming Responses to outstanding Requests and it MUST drop any Responses that do not match an outstanding request. For example, a msgID can be inserted in every message to cater for this functionality.

These mechanisms provide for the detection of authenticated messages whose time of generation was not recent.

This protection against the threat of message delay or replay does not imply nor provide any protection against unauthorized deletion or suppression of messages. Also, an SNMP engine may not be able to detect message reordering if all the messages involved are sent within the Time Window interval. Other mechanisms defined independently of the security protocol can also be used to detect the re-ordering replay, deletion, or suppression of messages containing Set operations (e.g., the MIB variable snmpSetSerialNo [RFC1907]).

2) Verification that a message sent to/from one authoritative SNMP engine cannot be replayed to/as-if-from another authoritative SNMP engine.

Included in each message is an identifier unique to the authoritative SNMP engine associated with the sender or intended recipient of the message.

A message containing an Unconfirmed Class PDU sent by an + authoritative SNMP engine to one non-authoritative SNMP engine can potentially be replayed to another non-authoritative SNMP engine. The latter non-authoritative SNMP engine might (if it knows about the same userName with the same secrets at the authoritative SNMP engine) as a result update its notion of timeliness indicators of the authoritative SNMP engine, but that is not considered a threat. In this case, A Report or Response message will be discarded by the Message Processing Model, because there should not be an outstanding Request message. A Trap will possibly be accepted. Again, that is not considered a threat, because the communication was authenticated and timely. It is as if the authoritative SNMP engine was configured to start sending Traps to the second SNMP engine, which theoretically can happen without the knowledge of the second SNMP engine anyway. Anyway, the second

SNMP engine may not expect to receive this Trap, but is allowed to see the management information contained in it.

3) Detection of messages which were not recently generated.

A set of time indicators are included in the message, indicating the time of generation. Messages without recent time indicators are not considered authentic. In addition, an SNMP engine MUST drop any Responses that do not match an outstanding request. This however is the responsibility of the Message Processing Model.

This memo allows the same user to be defined on multiple SNMP engines. Each SNMP engine maintains a value, snmpEngineID, which uniquely identifies the SNMP engine. This value is included in each message sent to/from the SNMP engine that is authoritative (see section 1.5.1). On receipt of a message, an authoritative SNMP engine checks the value to ensure that it is the intended recipient, and a non-authoritative SNMP engine uses the value to ensure that the message is processed using the correct state information.

Each SNMP engine maintains two values, snmpEngineBoots and snmpEngineTime, which taken together provide an indication of time at that SNMP engine. Both of these values are included in an authenticated message sent to/received from that SNMP engine. On receipt, the values are checked to ensure that the indicated timeliness value is within a Time Window of the current time. The Time Window represents an administrative upper bound on acceptable delivery delay for protocol messages.

For an SNMP engine to generate a message which an authoritative SNMP engine will accept as authentic, and to verify that a message received from that authoritative SNMP engine is authentic, such an SNMP engine must first achieve timeliness synchronization with the authoritative SNMP engine.

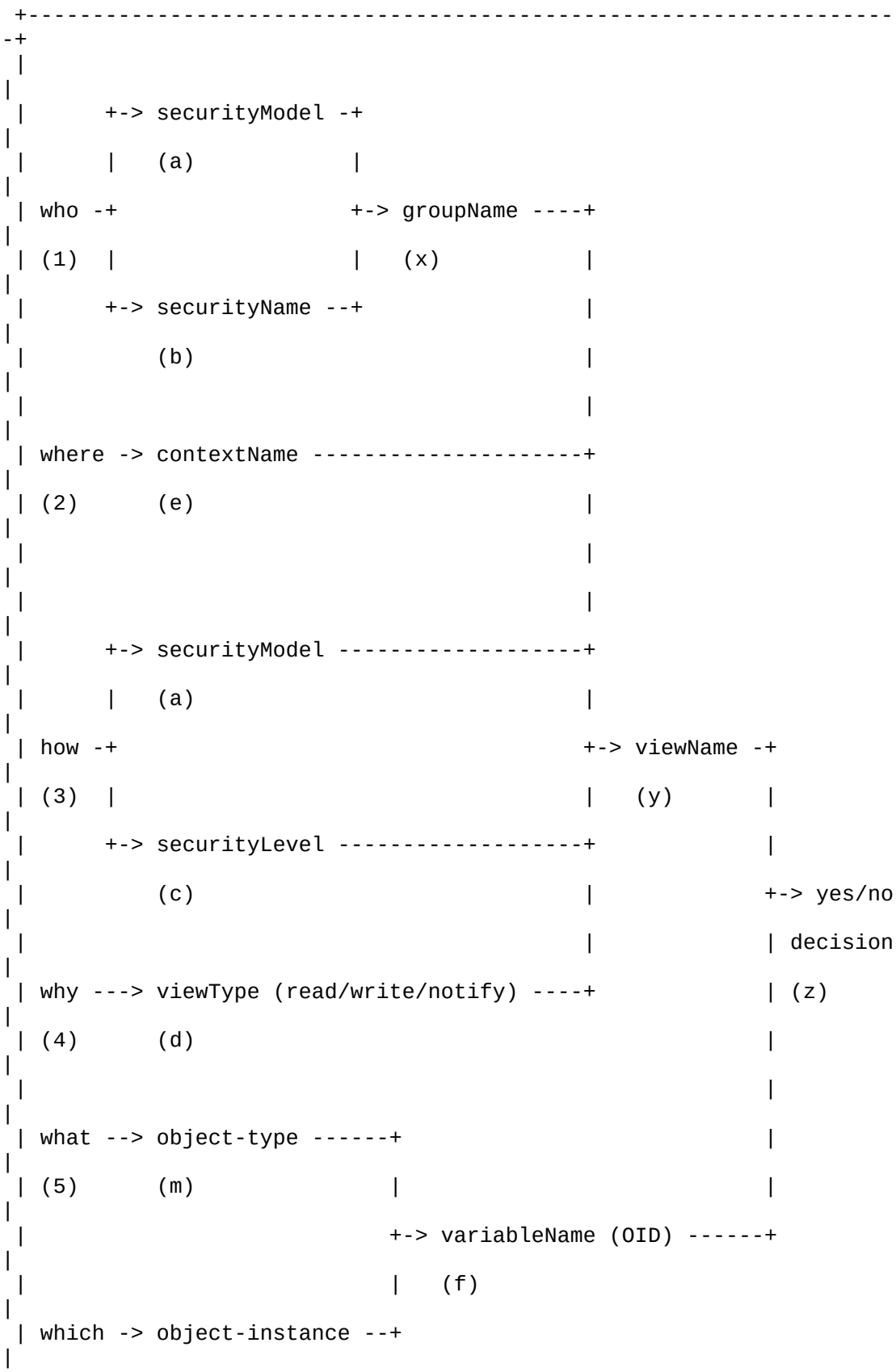
Each SNMP engine maintains three objects:

- **snmpEngineID,** which (at least within an administrative domain) uniquely and unambiguously identifies an SNMP engine.

- **snmpEngineBoots,** which is a count of the number of times the SNMP engine has re-booted/re-initialized since snmpEngineID was last configured; and,

- **snmpEngineTime,** which is the number of seconds since the snmpEngineBoots counter was last incremented.

Η παρακάτω εικόνα δείχνει πως η αποδοχή για έλεγχο φτιάχνεται από τη
View-based Access Control Model



```
 | (6)     (n)
|
 |
|
 +-----------------------------------------------------------------
-+
```

πως η απόφαση για την AccessAllowed φτιάχνεται

```
 1) Inputs to the isAccessAllowed service are:

    (a)       securityModel    -- Security Model in use
    (b)       securityName     -- principal who wants to
access
    (c)       securityLevel    -- Level of Security
    (d)       viewType         -- read, write, or notify
view
    (e)       contextName      -- context containing
variableName
    (f)       variableName     -- OID for the managed
object
                               -- this is made up of:

                            - object-type (m)
                                - object-instance (n)

 2) The partial "who" (1), represented by the
securityModel (a) and
    the securityName (b), are used as the indices (a,b)
into the
    vacmSecurityToGroupTable to find a single entry that
produces a
    group, represented by groupName (x).

 3) The "where" (2), represented by the contextName (e),
the "who",
    represented by the groupName (x) from the previous
step, and the
    "how" (3), represented by securityModel (a) and
securityLevel (c),
    are used as indices (e,x,a,c) into the
vacmAccessTable to find a
    single entry that contains three MIB views.

 4) The "why" (4), represented by the viewType (d), is
used to select
    the proper MIB view, represented by a viewName (y),
from the
    vacmAccessEntry selected in the previous step. This
viewName (y) is
    an index into the vacmViewTreeFamilyTable and selects
the set of
```

entries that define the variableNames which are included in or
excluded from the MIB view identified by the viewName (y).

5) The "what" (5) type of management data and "which" (6) particular
instance, represented by the variableName (f), is then checked to be
in the MIB view or not, e.g., the yes/no decision (z).

Definitions

```
SNMP-VIEW-BASED-ACM-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-COMPLIANCE, OBJECT-GROUP       FROM SNMPv2-CONF
    MODULE-IDENTITY, OBJECT-TYPE,
    snmpModules                           FROM SNMPv2-SMI
    TestAndIncr,
    RowStatus, StorageType                FROM SNMPv2-TC
    SnmpAdminString,
    SnmpSecurityLevel,
    SnmpSecurityModel                     FROM SNMP-FRAMEWORK-MIB;

snmpVacmMIB      MODULE-IDENTITY
```

```
DESCRIPTION "The table of locally available contexts.

            This table provides information to SNMP Command
            Generator applications so that they can properly
            configure the vacmAccessTable to control access to
            all contexts at the SNMP entity.

            This table may change dynamically if the SNMP entity
            allows that contexts are added/deleted dynamically
            (for instance when its configuration changes). Such
            changes would happen only if the management
            instrumentation at that SNMP entity recognizes more
            (or fewer) contexts.

            The presence of entries in this table and of entries
            in the vacmAccessTable are independent.  That is, a
            context identified by an entry in this table is not
            necessarily referenced by any entries in the
            vacmAccessTable; and the context(s) referenced by an
            entry in the vacmAccessTable does not necessarily
```

```
                currently exist and thus need not be identified by
an
                entry in this table.

                This table must be made accessible via the default
                context so that Command Responder applications have
                a standard way of retrieving the information.

                This table is read-only. It cannot be configured via
                SNMP.
               "
    ::= { vacmMIBObjects 1 }

vacmContextEntry OBJECT-TYPE
    SYNTAX       VacmContextEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "Information about a particular context."
    INDEX       {
                  vacmContextName
                }
    ::= { vacmContextTable 1 }

VacmContextEntry ::= SEQUENCE
    {
        vacmContextName SnmpAdminString
    }

vacmContextName  OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(0..32))
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "A human readable name identifying a particular
                 context at a particular SNMP entity.

                 The empty contextName (zero length) represents the
                 default context.
                "
    ::= { vacmContextEntry 1 }

-- Information about Groups
****************************************

vacmSecurityToGroupTable OBJECT-TYPE
    SYNTAX       SEQUENCE OF VacmSecurityToGroupEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "This table maps a combination of securityModel and
                 securityName into a groupName which is used to
define
                 an access control policy for a group of principals.
                "
    ::= { vacmMIBObjects 2 }

vacmSecurityToGroupEntry OBJECT-TYPE
    SYNTAX       VacmSecurityToGroupEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "An entry in this table maps the combination of a
                 securityModel and securityName into a groupName.
                "
```

```
    INDEX       {
                  vacmSecurityModel,
                  vacmSecurityName
                }
    ::= { vacmSecurityToGroupTable 1 }

VacmSecurityToGroupEntry ::= SEQUENCE
    {
        vacmSecurityModel               SnmpSecurityModel,
        vacmSecurityName                SnmpAdminString,
        vacmGroupName                   SnmpAdminString,
        vacmSecurityToGroupStorageType  StorageType,
        vacmSecurityToGroupStatus       RowStatus
    }

vacmSecurityModel OBJECT-TYPE
    SYNTAX       SnmpSecurityModel(1..2147483647)
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "The Security Model, by which the vacmSecurityName
                 referenced by this entry is provided.

                 Note, this object may not take the 'any' (0) value.
                "
    ::= { vacmSecurityToGroupEntry 1 }

vacmSecurityName OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(1..32))
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "The securityName for the principal, represented in a
                 Security Model independent format, which is mapped
by
                 this entry to a groupName.

                 The securityName for a principal represented in a
                 Security Model independent format.
                "
    ::= { vacmSecurityToGroupEntry 2 }

vacmGroupName    OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(1..32))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The name of the group to which this entry (e.g., the
                 combination of securityModel and securityName)
                 belongs.

                 This groupName is used as index into the
                 vacmAccessTable to select an access control policy.
                "
    ::= { vacmSecurityToGroupEntry 3 }

vacmSecurityToGroupStorageType OBJECT-TYPE
    SYNTAX       StorageType
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The storage type for this conceptual row.
```

```
                Conceptual rows having the value 'permanent' need
not
                allow write-access to any columnar objects in the
row.
               "
    DEFVAL      { nonVolatile }
    ::= { vacmSecurityToGroupEntry 4 }

vacmSecurityToGroupStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION "The status of this conceptual row.

                Until instances of all corresponding columns are
                appropriately configured, the value of the
                corresponding instance of the
vacmSecurityToGroupStatus
                column is 'notReady'.

                In particular, a newly created row cannot be made
                active until a value has been set for vacmGroupName.

                The  RowStatus TC [RFC1903] requires that this
                DESCRIPTION clause states under which circumstances
                other objects in this row can be modified:

                The value of this object has no effect on whether
                other objects in this conceptual row can be
modified.
               "
    ::= { vacmSecurityToGroupEntry 5 }

-- Information about Access Rights
**********************************

vacmAccessTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF VacmAccessEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION "The table of access rights for groups.

                Each entry is indexed by a contextPrefix, a
groupName
                a securityModel and a securityLevel.  To determine
                whether access is allowed, one entry from this table
                needs to be selected and the proper viewName from
that
                entry must be used for access control checking.

                To select the proper entry, follow these steps:

                1) the set of possible matches is formed by the
                   intersection of the following sets of entries:
                     the set of entries with identical vacmGroupName
                     the union of these two sets:
                      - the set with identical
vacmAccessContextPrefix
                      - the set of entries with
vacmAccessContextMatch
                        value of 'prefix' and matching
```

```
                        vacmAccessContextPrefix
                      intersected with the union of these two sets:
                       - the set of entries with identical
                         vacmSecurityModel
                       - the set of entries with vacmSecurityModel
                         value of 'any'
                      intersected with the set of entries with
                      vacmAccessSecurityLevel value less than or
equal
                      to the requested securityLevel

                  2) if this set has only one member, we're done
                     otherwise, it comes down to deciding how to
weight
                     the preferences between ContextPrefixes,
                     SecurityModels, and SecurityLevels as follows:
                     a) if the subset of entries with securityModel
                        matching the securityModel in the message is
                        not empty, then discard the rest.
                     b) if the subset of entries with
                        vacmAccessContextPrefix matching the
contextName
                        in the message is not empty,
                        then discard the rest
                     c) discard all entries with ContextPrefixes
shorter
                        than the longest one remaining in the set
                     d) select the entry with the highest
securityLevel

                   Please note that for securityLevel noAuthNoPriv, all
                   groups are really equivalent since the assumption
that
                   the securityName has been authenticated does not
hold.
                  "
    ::= { vacmMIBObjects 4 }

vacmAccessEntry  OBJECT-TYPE
    SYNTAX       VacmAccessEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "An access right configured in the Local
Configuration
                 Datastore (LCD) authorizing access to an SNMP
context.
                "
    INDEX       { vacmGroupName,
                  vacmAccessContextPrefix,
                  vacmAccessSecurityModel,
                  vacmAccessSecurityLevel
                }
    ::= { vacmAccessTable 1 }
VacmAccessEntry ::= SEQUENCE
    {
        vacmAccessContextPrefix    SnmpAdminString,
        vacmAccessSecurityModel    SnmpSecurityModel,
        vacmAccessSecurityLevel    SnmpSecurityLevel,
        vacmAccessContextMatch     INTEGER,
        vacmAccessReadViewName     SnmpAdminString,
        vacmAccessWriteViewName    SnmpAdminString,
```

```
        vacmAccessNotifyViewName   SnmpAdminString,
        vacmAccessStorageType      StorageType,
        vacmAccessStatus           RowStatus
    }

vacmAccessContextPrefix OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(0..32))
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "In order to gain the access rights allowed by this
                 conceptual row, a contextName must match exactly
                 (if the value of vacmAccessContextMatch is 'exact')
                 or partially (if the value of vacmAccessContextMatch
                 is 'prefix') to the value of the instance of this
                 object.
                "
    ::= { vacmAccessEntry 1 }

vacmAccessSecurityModel OBJECT-TYPE
    SYNTAX       SnmpSecurityModel
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "In order to gain the access rights allowed by this
                 conceptual row, this securityModel must be in use.
                "
    ::= { vacmAccessEntry 2 }

vacmAccessSecurityLevel OBJECT-TYPE
    SYNTAX       SnmpSecurityLevel
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "The minimum level of security required in order to
                 gain the access rights allowed by this conceptual
                 row.  A securityLevel of noAuthNoPriv is less than
                 authNoPriv which in turn is less than authPriv.

                 If multiple entries are equally indexed except for
                 this vacmAccessSecurityLevel index, then the entry
                 which has the highest value for
                 vacmAccessSecurityLevel wins.
                "
    ::= { vacmAccessEntry 3 }

vacmAccessContextMatch OBJECT-TYPE
    SYNTAX       INTEGER
                {  exact (1), -- exact match of prefix and contextName
                   prefix (2) -- Only match to the prefix
                }
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "If the value of this object is exact(1), then all
                 rows where the contextName exactly matches
                 vacmAccessContextPrefix are selected.

                 If the value of this object is prefix(2), then all
                 rows where the contextName whose starting octets
                 exactly match vacmAccessContextPrefix are selected.
                 This allows for a simple form of wildcarding.
                 See also the example in the DESCRIPTION clause of
                 the vacmAccessTable above.
                "
```

```
    DEFVAL      { exact }
    ::= { vacmAccessEntry 4 }

vacmAccessReadViewName OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(0..32))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The value of an instance of this object identifies
                 the MIB view of the SNMP context to which this
                 conceptual row authorizes read access.

                 The identified MIB view is that one for which the
                 vacmViewTreeFamilyViewName has the same value as the
                 instance of this object; if the value is the empty
                 string or if there is no active MIB view having this
                 value of vacmViewTreeFamilyViewName, then no access
                 is granted.
                "
    DEFVAL      { ''H }   -- the empty string
    ::= { vacmAccessEntry 5 }

vacmAccessWriteViewName OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(0..32))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The value of an instance of this object identifies
the MIB view of the SNMP context to which this
                 conceptual row authorizes write access.

                 The identified MIB view is that one for which the
                 vacmViewTreeFamilyViewName has the same value as the
                 instance of this object; if the value is the empty
                 string or if there is no active MIB view having this
                 value of vacmViewTreeFamilyViewName, then no access
                 is granted.
                "
    DEFVAL      { ''H }   -- the empty string
    ::= { vacmAccessEntry 6 }

vacmAccessNotifyViewName OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(0..32))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The value of an instance of this object identifies
                 the MIB view of the SNMP context to which this
                 conceptual row authorizes access for notifications.

                 The identified MIB view is that one for which the
                 vacmViewTreeFamilyViewName has the same value as the
                 instance of this object; if the value is the empty
                 string or if there is no active MIB view having this
                 value of vacmViewTreeFamilyViewName, then no access
                 is granted.
                "
    DEFVAL      { ''H }   -- the empty string
    ::= { vacmAccessEntry 7 }

vacmAccessStorageType OBJECT-TYPE
    SYNTAX       StorageType
    MAX-ACCESS   read-create
    STATUS       current
```

```
    DESCRIPTION "The storage type for this conceptual row.

                 Conceptual rows having the value 'permanent' need
not
                 allow write-access to any columnar objects in the
row.
                "
    DEFVAL      { nonVolatile }
    ::= { vacmAccessEntry 8 }

vacmAccessStatus OBJECT-TYPE
    SYNTAX       RowStatus
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The status of this conceptual row.
                 The  RowStatus TC [RFC1903] requires that this
                 DESCRIPTION clause states under which circumstances
                 other objects in this row can be modified:

                 The value of this object has no effect on whether
                 other objects in this conceptual row can be
modified.
                "
    ::= { vacmAccessEntry 9 }

-- Information about MIB views
**************************************

-- Support for instance-level granularity is optional.
--
-- In some implementations, instance-level access control
-- granularity may come at a high performance cost.  Managers
-- should avoid requesting such configurations unnecessarily.

vacmMIBViews     OBJECT IDENTIFIER ::= { vacmMIBObjects 5 }

vacmViewSpinLock OBJECT-TYPE
    SYNTAX       TestAndIncr
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION "An advisory lock used to allow cooperating SNMP
                 Command Generator applications to coordinate their
                 use of the Set operation in creating or modifying
                 views.

                 When creating a new view or altering an existing
                 view, it is important to understand the potential
                 interactions with other uses of the view.  The
                 vacmViewSpinLock should be retrieved.  The name of
                 the view to be created should be determined to be
                 unique by the SNMP Command Generator application by
                 consulting the vacmViewTreeFamilyTable.  Finally,
                 the named view may be created (Set), including the
                 advisory lock.
                 If another SNMP Command Generator application has
                 altered the views in the meantime, then the spin
                 lock's value will have changed, and so this creation
                 will fail because it will specify the wrong value
for
                 the spin lock.
```

```
                Since this is an advisory lock, the use of this lock
                is not enforced.
              "
    ::= { vacmMIBViews 1 }
vacmViewTreeFamilyTable OBJECT-TYPE
    SYNTAX       SEQUENCE OF VacmViewTreeFamilyEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "Locally held information about families of subtrees
                within MIB views.

                Each MIB view is defined by two sets of view
subtrees:
                  - the included view subtrees, and
                  - the excluded view subtrees.
                Every such view subtree, both the included and the
                excluded ones, is defined in this table.

                To determine if a particular object instance is in
                a particular MIB view, compare the object instance's
                OBJECT IDENTIFIER with each of the MIB view's active
                entries in this table.  If none match, then the
                object instance is not in the MIB view.  If one or
                more match, then the object instance is included in,
                or excluded from, the MIB view according to the
                value of vacmViewTreeFamilyType in the entry whose
                value of vacmViewTreeFamilySubtree has the most
                sub-identifiers.  If multiple entries match and have
                the same number of sub-identifiers, then the
                lexicographically greatest instance of
                vacmViewTreeFamilyType determines the inclusion or
                exclusion.

                An object instance's OBJECT IDENTIFIER X matches an
                active entry in this table when the number of
                sub-identifiers in X is at least as many as in the
                value of vacmViewTreeFamilySubtree for the entry,
                and each sub-identifier in the value of
                vacmViewTreeFamilySubtree matches its corresponding
                sub-identifier in X.  Two sub-identifiers match
                either if the corresponding bit of the value of
                vacmViewTreeFamilyMask for the entry is zero (the
                'wild card' value), or if they are equal.

                A 'family' of subtrees is the set of subtrees
defined
                by a particular combination of values of
                vacmViewTreeFamilySubtree and
vacmViewTreeFamilyMask.
                In the case where no 'wild card' is defined in the
                vacmViewTreeFamilyMask, the family of subtrees
reduces
                to a single subtree.

                When creating or changing MIB views, an SNMP Command
                Generator application should utilize the
                vacmViewSpinLock to try to avoid collisions. See
                DESCRIPTION clause of vacmViewSpinLock.

                When creating MIB views, it is strongly advised that
                first the 'excluded' vacmViewTreeFamilyEntries are
```

```
                created and then the 'included' entries.

                When deleting MIB views, it is strongly advised that
                first the 'included' vacmViewTreeFamilyEntries are
                deleted and then the 'excluded' entries.

                If a create for an entry for instance-level access
                control is received and the implementation does not
                support instance-level granularity, then an
                inconsistentName error must be returned.
               "
    ::= { vacmMIBViews 2 }

vacmViewTreeFamilyEntry OBJECT-TYPE
    SYNTAX       VacmViewTreeFamilyEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "Information on a particular family of view subtrees
                included in or excluded from a particular SNMP
                context's MIB view.

                Implementations must not restrict the number of
                families of view subtrees for a given MIB view,
                except as dictated by resource constraints on the
                overall number of entries in the
                vacmViewTreeFamilyTable.

                If no conceptual rows exist in this table for a
given
                MIB view (viewName), that view may be thought of as
                consisting of the empty set of view subtrees.
               "
    INDEX       { vacmViewTreeFamilyViewName,
                  vacmViewTreeFamilySubtree
                }
    ::= { vacmViewTreeFamilyTable 1 }

VacmViewTreeFamilyEntry ::= SEQUENCE
    {
        vacmViewTreeFamilyViewName     SnmpAdminString,
        vacmViewTreeFamilySubtree      OBJECT IDENTIFIER,
        vacmViewTreeFamilyMask         OCTET STRING,
        vacmViewTreeFamilyType         INTEGER,
        vacmViewTreeFamilyStorageType  StorageType,
        vacmViewTreeFamilyStatus       RowStatus
    }

vacmViewTreeFamilyViewName OBJECT-TYPE
    SYNTAX       SnmpAdminString (SIZE(1..32))
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION "The human readable name for a family of view
subtrees.
               "
    ::= { vacmViewTreeFamilyEntry 1 }

vacmViewTreeFamilySubtree OBJECT-TYPE
    SYNTAX       OBJECT IDENTIFIER
    MAX-ACCESS   not-accessible
    STATUS       current
```

```
    DEFVAL      { included }
    ::= { vacmViewTreeFamilyEntry 4 }

vacmViewTreeFamilyStorageType OBJECT-TYPE
    SYNTAX       StorageType
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The storage type for this conceptual row.

                 Conceptual rows having the value 'permanent' need
not                 allow write-access to any columnar objects in the
row.
                "
    DEFVAL      { nonVolatile }
    ::= { vacmViewTreeFamilyEntry 5 }

vacmViewTreeFamilyStatus OBJECT-TYPE
    SYNTAX       RowStatus
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION "The status of this conceptual row.

                 The  RowStatus TC [RFC1903] requires that this
                 DESCRIPTION clause states under which circumstances
                 other objects in this row can be modified:

                 The value of this object has no effect on whether
                 other objects in this conceptual row can be
modified.
                "
    ::= { vacmViewTreeFamilyEntry 6 }

-- Conformance information
****************************************

vacmMIBCompliances  OBJECT IDENTIFIER ::= { vacmMIBConformance 1 }
vacmMIBGroups       OBJECT IDENTIFIER ::= { vacmMIBConformance 2 }

-- Compliance statements
****************************************

vacmMIBCompliance MODULE-COMPLIANCE
    STATUS       current
    DESCRIPTION "The compliance statement for SNMP engines which
                 implement the SNMP View-based Access Control Model
                 configuration MIB.
                "
    MODULE -- this module
        MANDATORY-GROUPS { vacmBasicGroup }

        OBJECT        vacmAccessContextMatch
        MIN-ACCESS    read-only
        DESCRIPTION  "Write access is not required."
        OBJECT        vacmAccessReadViewName
        MIN-ACCESS    read-only
        DESCRIPTION  "Write access is not required."

        OBJECT        vacmAccessWriteViewName
        MIN-ACCESS    read-only
        DESCRIPTION  "Write access is not required."
```

View-Based Access Control

Ο έλεγχος είναι μια συνάρτηση ασφάλειας που παρουσιάζεται στο επίπεδο PDU. Ορίζει μηχανισμούς για κρίση . το πρωτόκολλο μας ορίζει το μοντέλο VACM. Τα σημαντικά χαρακτηριστικά του είναι τα εξής:
:

VACM determines whether access to a managed object in a local MIB by a remote principal
should be allowed.
VACM makes use of a MIB that:

Defines the access control policy for this agent
Makes it possible for remote configuration to be used.

Στοιχεία του μοντέλου αυτού είναι:

- groups
- security level
- contexts
- MIB views
- access policy.

Groups -- A group is defined as a set of zero or more <securityModel, securityName> tupleson whose behalf SNMP management objects can be accessed. A securityName refers to aprincipal, and access rights for all principals in a given group are identical. A unique groupName isassociated with each group. The group concept is a useful tool for categorizing managers with respect to access rights. For example, all top-level managers may have one set of access rights, while intermediate-level managers may have a different set of access rights. Any given combination of securityModel and securityName can belong to at most one group.That is, for this agent, a given principal whose communications are protected by a givensecurityModel can only be included in one group.

Security Level -- The access rights for a group may differ depending on the security level of the message that contains the request. For example, an agent may allow read-only access for a request communicated in an unauthenticated message but may require authentication for write access. Further, for certain sensitive objects, the agent may require that the request and its response be communicated using the privacy service.

Contexts -- A MIB context is a named subset of the object instances in the local MIB.Contexts provide a useful way of aggregating objects into collections with different access policies. The context is a concept that relates to access control. When a management station interacts with an agent to access management information at the agent, then the interaction is between a

management principal and the agent's SNMP engine, and the access control privileges are expressed in a MIB view that applies to this principal and this context. Contexts have the following key characteristics:

An SNMP entity, uniquely identified by a contextEngineID, may maintain more than one context. An object or an object instance may appear in more than one context. When multiple contexts exist, to identify an individual object instance, its contextName and contextEngineID must be identified in addition to its object type and its instance.

MIB Views -- It is often the case that we would like to restrict the access of a particulargroup to a subset of the managed objects at an agent. To achieve this objective, access to a context is by means of a MIB view, which defines a specific set of managed objects (and optionally specific object instances). VACM makes use of a powerful and flexible technique for defining MIB views, based on the concepts of view subtrees and view families. The MIB view is defined in terms of a collection, or family, of subtrees, with each subtree being included in or excluded from the view. The managed objects in a local database are organized into a hierarchy, or tree, based on the object identifiers of the objects. This local database comprises a subset of all object types defined according to the Internet-standard Structure of Management Information (SMI) and includes object instances whose identifiers conform to the SMI conventions. SNMPv3 includes the concept of a subtree. A subtree is simply a node in the MIB's naminghierarchy plus all of its subordinate elements. More formally, a subtree may be defined as the set of all objects and object instances that have a common ASN.1 OBJECT IDENTIFIER prefix to their names. The longest common prefix of all of the instances in the subtree is the object identifier of the parent node of that subtree. Associated with each entry in vacmAccessTable are three MIB views, one each for read, write, and notify access. Each MIB view consists of a set of view subtrees. Each view subtree in theMIB view is specified as being included or excluded. That is, the MIB view either includes or excludes all object instances contained in that subtree. In addition, a view mask is defined in order to reduce the amount of configuration information required when fine-grained access control is required(e.g., access control at the object instance level).

Access Policy -- VACM enables an SNMP engine to be configured to enforce a particular set of access rights. Access determination depends on the following factors:

The principal making the access request. The VACM makes it possible for an agent to allow different access privileges for different users. For example, a manager system responsible for network-wide configuration may have broad authority to alter items in the local MIB, while an intermediate level manager with monitoring responsibility may have read-only access and may further be limited to accessing only a subset of the local MIB. As was discussed, principals are assigned to groups and access policy is specified with respect to groups. The security level by which the request was communicated in an SNMP message.

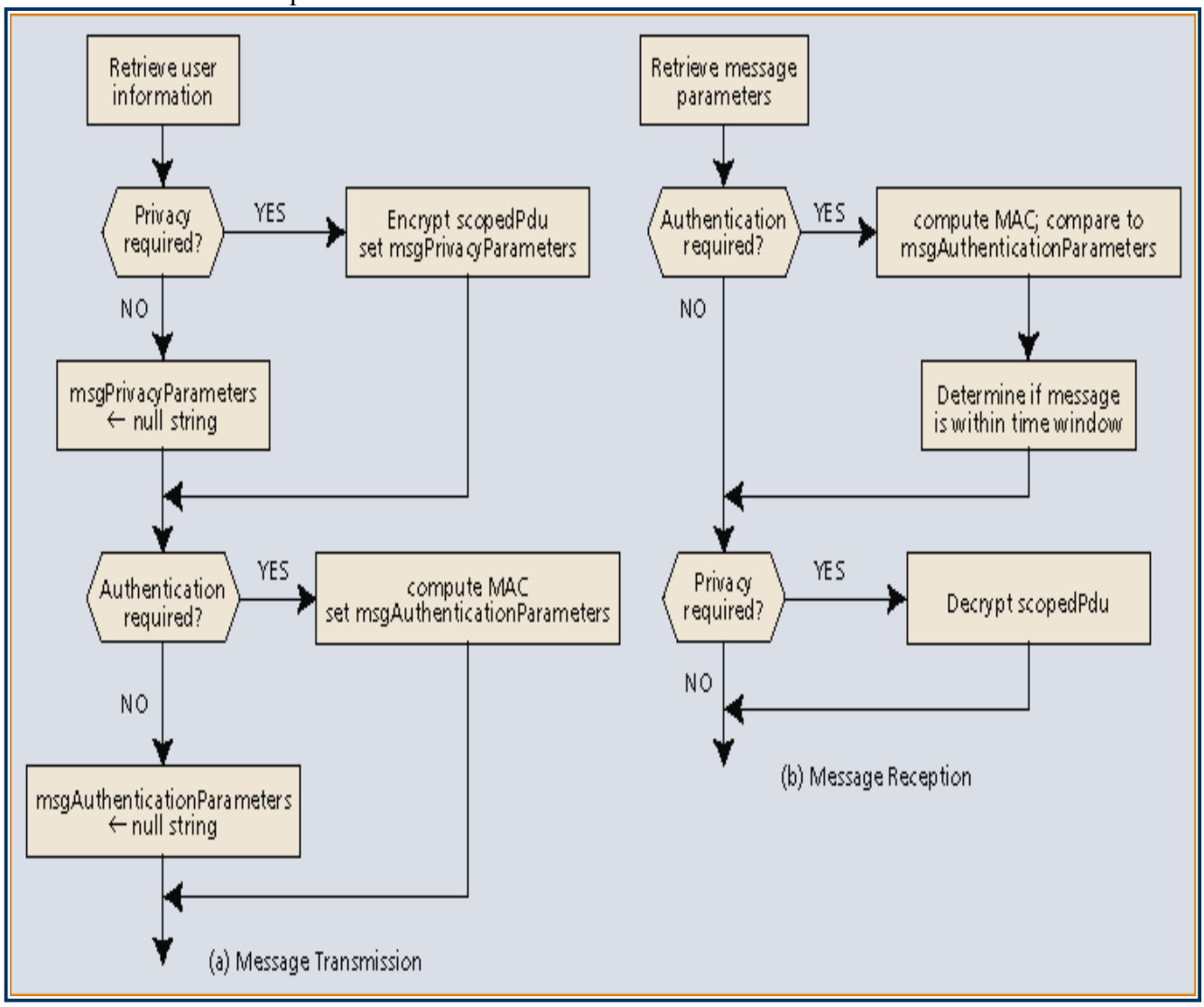
Typically,

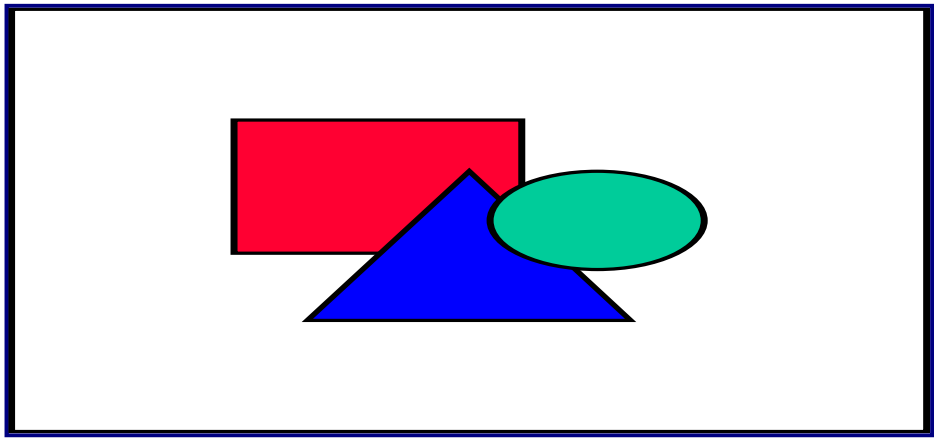
an agent will require the use of authentication for messages containing a set request (write operation). The security model used for processing the request message. If multiple security models are implemented at an agent, the agent may be configured to provide different levels of access to requests communicated by messages processed by different security models. For example, certain items may be accessible if the request message comes through USM, but not accessible if the Security Model is SNMPv1.

The MIB context for the request. The specific object instance for which access is requested. Some objects hold more critical or sensitive information than others, and therefore the access policy must depend on the

specific object instance requested. The type of access requested (read, write, notify). Read, write, and notify are distinct management operations, and different access control policies may apply for each of these

operations.





Encoding

.

Tag field

The tag field consists of one octet containing three subfields: class (2 bits), form (1 bit), number (5 bits).

Class subfield

There are four classes of tags in ASN.1: universal (00), application-wide (01), context-specific (10) and private-use (11).

Form subfield

The meaning of the form field is simple, it specifies whether the data that follows is just a single data value (0), or that it consists of multiple BER-encoded data values (1). Examples of form types are a Sequence (a concatination of an arbitrary number of ASN.1 values) and a PDU.

Number subfield

The number field is a non-negative number (0 - 32), which specifies the type of the encoded data. The following types are used in SNMPv3:

universal types		application-wide types		context-specific types	
nr	**type**	**nr**	**type**	**nr**	**type**
2	32-bit signed integer	0	IP address	0	get request PDU
4	octetstring	1	32-bit counter	1	get next request PDU
5	null	2	gauge	2	response PDU
6	object identifier	3	timeticks	3	set request PDU
16	sequence	6	64-bit counter	5	get bulk request PDU
				6	inform PDU
				7	trap PDU
				8	report PDU

An octetstring is an arbitrary number of octets. Usually this means a byte array. The 32-bit counter, gauge and timeticks types are all 32-bit unsigned integers. The 64-bit counter is a 64-bit unsigned integer. These different types all have a different pre-defined behavior, but as this is not within the scope of this project (it's only of importance to manager applications and Local Processing), those behaviors will not be discussed.

Value field

The value field holds the actual value of an encoded variable. The encoding of the value differs per ASN.1 type. Below is a table with the encoding of the types.

type	encoding
signed/unsigned integer, gauge, 32/64-bit counter, timeticks	These values are encoded in network byte order, which means the most significant byte first. Heading null bytes are not encoded.
octetstring	The encoding of this type is just a serialization of the octets.
null	A null variable has no data, and thus the value field is empty.
object identifier	For an object identifier the sub-identifiers are encoded like integers.
sequence	A sequence is not a data value itself, but consists of a concatination of multiple variables. Thus the value field of a sequence contains the BER-encoded variables.
IP address	The encoding of an IP address consist of the four numbers in network byte order.
PDUs	A PDU is basically a pre-defined sequence of variables, so the value field of a PDU also contains a series of BER-encoded variables.

Encoding example

The following example (please disregard the syntax) will be used to show BER at work:

```
Sequence {integer: 3, octetstring: "SNMP"}
```

Encoding is started with the integer. First the value will be encoded. From the table the conclusion can be made that only the lower byte will be encoded (as null bytes are left out of it), so the value will be encoded as 00000011 (binary). Then a tag is added. As an integer is a universal type, the class of the tag is 00 (binary). An integer isn't a multiple of ASN.1 values, so the form of the tag is 0 (binary). Finally, the number of an integer is 2 (decimal), so the resulting tag will be 00000010 (binary). After that the length will be added, as the integer is only one byte long in its encoded form, this will be 1 (decimal). The result of the encoding is therefore:

tag	length	value
00000010	00000001	00000011

The next step is the encoding of the octetstring. For the value the bytes of the string itself will be used, therefore the value will be encoded as 01010011 01001110 01001101 01010000 (binary). It's obvious of course that the length of this value is 4 bytes. The class and form part of the tag field are the same as those of an integer. The number of an octetstring is 4 (decimal), so the tag will be encoded as 00000100. The encoding will result in the following:

tag	length	value			
00000100	00000100	01010011	01001110	01001101	01010000

The final part of the encoding consist of adding a header for the sequence. A sequence is also a universal type, so the class is 00 here also. As a sequence actually consists of multiple ASN.1 values, the form field is 1 here. The number of a sequence is 16, so the resulting tag is 00110000 (binary). The length of the sequence is the length of its values combined, which is 5. So the following header will be prepended: 00110000 00000101.

The final result of the encoding:

sequence		integer		octetstring					
00110000	00000101	00000010	00000001	00000011	00000100	00000100	01010011	01001110	01001101	01010000

Βιβλιογραφία

sites of internet

- http://www.tis.com/docs/research/network/snmp-ng.html
- http://www.tis.com/docs/research/network/snmp-ng_slide.html
- http://rosegarden.external.hp.com/snmp++
- http://www.adventnet.com/snmpapi/snmpv2c/docs/docs/packages.html
- http://ietf.org/html.charters/snmpv3-charter.html
- http://www.ibr.cs.tu-bs.de/projects/snmpv3
- http://www.simple-times.org
- http://wwwsnmp.cs.utwente.nl
- http://www.infosabah.com.mg/znma/train/snmp.html
- http://info.internet.isi.edu:80/R1064178-1069105-1m/in-drafts/id-abstracts.html
- http://www.busn.ucok.edu/tips/info-int/snmp.htm
- http://www.noinfo.com/technology/snmp/index.html
- http://wwwsnmp.cs.utwente/research/projects/laforge/assignement/index.htm
- http://www.snmp.com/v3hotspot
- http://www.snmp.com/comnet98.html
- http://www.shore.net/~ws
- http://info.internet.isi.edu:70/in-draft…s/draft-ietf-snmpv.3-next-gen-arch-07.tx
- http://ietf.org/internet-drafts/draft-ietf-snmpv3-usm-v2-03.txt
- http://www.snmp.com/whyus.html
- http://www.snmp.com/snmppages.html
- http://networkcomputing.com/915/915f14.html
- http://comsoc.org/pubs/surveys/4q98issue/stallings.html
- http://www.iee.com
- http://www.altavista.com
- http://www.yahoo.com

Διεθνή επιστημονικά περιοδικά

- ***IEEE Communications Magazine –March1998***
- ***Network Computing- August 1998***

Βιβλία

- ***Πανεπιστημιακές παραδόσεις κ .Α. Πομπόρτση Ιανουάριος 99***
- ***Πομπόρτσης, Ανδρέας, Εισαγωγή στις νέες τεχνολογίες τηλεπικοινωνιών***
- ***Stallings , William, Data and Computer Communications, Prentice-Hall, 1997***
- ***Sheldon, Tom, Encycopedia of Networking, Osborne/McGraw-Hill,1998***