# Τα δίκτυα στον 21ο αιώνα :
# Ενεργά και Προγραμματιζόμενα

**Λιόντος Ανδρέας**
**Α.Μ. : Μ6/99**

**“Τεχνολογίες Τηλεπικοινωνιών και Δικτύων”**

**Γενικά**

Ενώ οι διαθέσιμες τεχνολογίες τόσο σε επίπεδο υποδομής δικτύων και πολύ περισσότερο σε επίπεδο εφαρμογών και υπολογιστικής ισχύος αναπτύσσονται με σταθερούς και ταχύτατους ρυθμούς, η εξέλιξη σε επίπεδο υπηρεσιών δικτύου είναι πολύ πιο αργή, περιοριζόμενη από διαδικασίες τυποποίησης. Η τυποποίηση αυτή ενώ είναι απαραίτητη προκειμένου να διασφαλειστεί η διαλειτουργικότητα, ωστόσο βασίζεται σε αργές διαδικασίες γενικής ομοφωνίας. Επιπλέον η διαχείρηση και ο έλεγχος των δικτύων δεν έχουν σημειώσει την πρόοδο που θα περίμενε κανείς.

Οι νέες τάσεις στα δίκτυα στοχεύουν στην επιτάχυνση των υπηρεσιών και τη βελτίωση της διαχείρησης των δικτύων. Ένα προγραμματιζόμενο δίκτυο παρέχει μεγάλο βαθμό ελέγχου στους χρήστες προκειμένου να προσαρμόσουν τις δικτυακές υπηρεσίες δυναμικά, να τις προγραμματίσουν, ενσωματώνοντας ‘‘κινητά’ προγράμματα σε ειδικά πακέτα τα οποία εκτελούνται από τα στοιχεία του δικτύου. Τα προγράμματα αυτά μπορούν να εκτελέσουν λειτουργίες διαχείρησης και ελέγχου. Επομένως χρειάζεται κάποια συμφωνία πάνω σε κάποιο σύνολο αρχών παρά ομοφωνία σε συγκεκριμένα πρωτόκολλα και υπηρεσίες. Βέβαια το κέρδος σε ελαστικότητα δημιουργεί νέα ζητήματα ασφάλειας, διαχείρησης και καταμερισμού πόρων, τα οποία πρέπει να αντιμετωπισθούν προκειμένου να αποφευχθούν αρνητικές επιπτώσεις σε βάρος του δικτύου και των υπολοίπων χρηστών.

**Εισαγωγή**

Στα παραδοσιακά δίκτυα επικοινωνίας όλη η ευφυϊα βρίσκεται μέσα στο δίκτυο κάτω από τον έλεγχο του διαχειριστή του δικτύου. Αντίθετα στην παραδοσιακή φιλοσοφία του Internet το δίκτυο περιέχει πολύ μικρή ευφυϊα και οι εφαρμογές των τερματικών ελέγχουν την αλληλεπίδρασή τους. Με τον τρόπο που λειτουργούν σήμερα τα δίκτυα, οι επικοινωνίες μεταξύ δύο σημείων μπορούν να χρησιμοποιήσουν μόνο ένα καθορισμένο σύνολο στατικών πρωτοκόλλων που παρέχονται από τους ενδιάμεσους κόμβους. Επιπλέον οι διαδικασίες που ακολουθούνται προκειμένου να υπάρχει καθολική τυποποίηση εισάγουν υπερβολική καθυστέρηση στην εξέλιξη των δικτύων.

Μπορεί πριν 20 χρόνια που οι χρήστες ήταν ευχαριστημένοι με τις βασικές λειτουργίες μεταφοράς δεδομένων και φωνής η εξέλιξη των δικτύων να ήταν ικανοποιητική. Ωστόσο σήμερα ο χώρος των τηλεπικοινωνιών είναι πολύ πιο σύνθετος και ανταγωνιστικός, τα τερματικά γίνονται όλα πιο έξυπνα και πιο ισχυρά και η δημοτικότητα του Internet έχει δημιουργήσει ανάγκες για νέες υπηρεσίες και εφαρμογές. Μέσα σ’αυτό το ρευστό περιβάλλον οι διαχειριστές του δικτύου και οι παροχείς υπηρεσιών πρέπει να ανταποκριθούν πιο άμμεσα και ελαστικά από ότι έκαναν στο παρελθόν.

Ακόμη οι προσπάθειες για παροχή Ποιότητας Υπηρεσιών κατά περίπτωση, οι πρόσφατες εξελίξεις στα κατανεμημένα συστήματα και στο φορητό λογισμικό μαζί με την αυξανόμενη απαίτηση για καλύτερο έλεγχο των δικτυακών πόρων οδηγούν στην επανεξέταση των αρχιτεκτονικών των δικτύων.

Η συνολική προσπάθεια είναι η δημιουργία ενός περιβάλλοντος που θα κάνει τα δίκτυα δυναμικά - προγραμματιζόμενα. Ένα προγραμματιζόμενο δίκτυο θα παρέχει λειτουργίες και βελτιστοποιήσεις που με τις σημερινές δικτυακές υποδομές δεν είναι εφικτές.

Γενικά λοιπόν Ενεργά Προγραμματιζόμενα Δίκτυα είναι αυτά που α) οι δρομολογητές και οι κατανεμητές μπορούν να εκτελέσουν υπολογισμούς στα

δεδομένα των χρηστών που τους διασχίζουν και β) επιτρέπουν στους χρήστες να ελέγχουν και να διαχειρίζονται μεμονωμένα τους δικτυακούς πόρους ώστε να ικανοποιήσουν τις απαιτήσεις τους και να "προγραμματίζουν" το δίκτυο, παρέχοντας τα προγράμματα που θα εκτελέσουν τις λειτουργίες και τους υπολογισμούς. Οι εφαρμογές θα καθορίζουν και θα χρησιμοποιούν υπηρεσίες που θα τις έχουν "παραγγείλει" και θα είναι προσαρμοσμένες στις ανάγκες τους, θα προσδιορίζουν και θα απολαμβάνουν προκαθορισμένη Ποιότητα Υπηρεσιών από τους επεξεργαστικούς και αποθηκευτικούς πόρους του δικτύου. Έτσι θα είναι δυνατή η γρηγορότερη και φθηνότερη ανάπτυξη νέων υπηρεσιών και εφαρμογών. Οι κόμβοι δε θα είναι πλέον κλειστά μαύρα κουτιά με αυστηρές ενσωματωμένες λειτουργίες που υπαγορεύονται από αργές επιτροπές τυποποίησης.

Η ιδέα λοιπόν γενικά είναι να υπάρχει η δυνατότητα τα πακέτα να περιέχουν εκτός από τα δεδομένα και κώδικα. Έτσι τα δεδομένα θα επεξεργάζονται από τα πρωτόκολλα που θα αναπτύσσονται στη στιγμή και τα οποία θα παραμένουν για μικρά ή μεγάλα χρονικά διαστήματα στο δίκτυο για χρήση και από άλλες μεταδόσεις.

## Εφαρμογές

Τα ενεργά-προγραμματιζόμενα δίκτυα μπορούν να είναι ευεργετικά για ένα πλήθος εφαρμογών. Εδώ θα εξετάσουμε γιατί και με ποιο τρόπο είναι αυτό δυνατό για τη διαχείρηση του δικτύου, τον έλεγχο συμφόρησης, multicasting και caching.

**Διαχείρηση Δικτύου**

Μέχρι και σήμερα η διαχείρηση του δικτύου επιτυγχάνεται έχοντας σταθμούς διαχείρησης οι οποίοι με διαδικασίες ρουτίνας ελέγχουν τις διαχειριζόμενες συσκευές για διάφορες ανωμαλίες. Η τεχνική αυτή λειτουργούσε ικανοποιητικά στο παρελθόν. Σήμερα όμως μάλλον τίνει να γίνει προβληματική λόγω της ραγδαίας αύξησης του αριθμού και της πολυπλοκότητας των κόμβων του δικτύου. Τα διαχειρηστικά κέντρα κατακλύζονται με πληροφορίες που πολλές φορές είναι περιττές αφού απλά αναφέρουν οτι δεν υπάρχει κανένα πρόβλημα. Αν πάλι υπάρχει κάποιο πρόβλημα, ο χρόνος που μεσολαβεί μέχρι να φτάσει η πληροφορία στο κέντρο και να γυρίσει η απάντηση στον προβληματικό κόμβο είναι κρίσιμος και πολλές φορές η αντίδραση να μην είναι πλέον επαρκής.

Χρησιμοποιώντας την προσέγγιση των ενεργών-προγραμματιζόμενων δικτύων μπορούν να "μεταφερθούν" τα διαχειρηστικά κέντρα στην καρδιά του δικτύου και να αποφευχθούν οι καθυστερήσεις. Μπορούμε να εισάγουμε ειδικό κώδικα στα πακέτα ώστε να ενεργήσουν σαν "πρώτες βοήθειες" αμέσως μόλις εντοπίσουν κάποιο προβληματικό κόμβο. Άλλα πακέτα μπορούν να εκτελούν "περιπολίες" ψάχνοντας συνεχώς για ανωμαλίες στο δίκτυο. Τέλος, ακόμη και η ίδια η πολιτική διαχείρησης μπορεί να αλλάξει εύκολα.

Υπάρχουν τρία ερευνητικά προγράμματα που εφαρμόζουν την τεχνολογία των ενεργών-προγραμματιζόμενων δικτύων στη διαχείρηση δικτύων :

1. **Smart Packets , BBN Technologies.** Τα παραδοσιακά πακέτα αντικαθίστανται από "έξυπνα" πακέτα που μπορεί να μεταφέρουν προγράμματα γραμμένα σε μια ασφαλή γλώσσα ειδικά για διαχείρηση δικτύου. Υπάρχουν 4 τύποι πακέτων : α) προγράμματος , που μεταφέρουν τον κώδικα που θα εκτελεστεί στον κατάλληλο κόμβο, β) δεδομένων , που μεταφέρουν τα αποτελέσματα της εκτέλεσης στο κέντρο διαχείρησης, γ)

μηνυμάτων , που μεταφέρουν μηνύματα με πληροφορίες και δ) σφαλμάτων που μεταφέρουν πληροφορίες για τα σφάλματα.

2. **Network Management by Delegation Paradigm , Columbia University.** Η βασική αρχή είναι οτι οι λειτουργίες διαχείρησης μπορούν με κάποια "εξουσιοδότηση" να εκτελεστούν δυναμικά τοπικά στα κατάλληλα τμήματα του δικτύου και όχι κεντρικά. Αντί να μεταφερθούν τα δεδομένα από τους κόμβους στο κέντρο διαχείρησης, μπορεί να μεταφερθεί ο κώδικας της εφαρμογής διαχείρησης στα τμήματα που βρίσκονται τα δεδομένα σαν εξουσιοδοτημένοι πράκτορες. Έτσι παρέχεται μια ισχυρή αρχιτεκτονική για κλιμακωτή, αποκεντρωτική και αυτοματοποιημένη διαχείρηση.
3. **Darwin Project.** Οι εφαρμογές ή οι παροχείς υπηρεσιών στέλνουν εξουσιδοτημένους πράκτορες-κώδικα στους κόμβους του δικτύου για να εφαρμόσουν κάποια συγκεκριμένη διαχείρηση στη ροή των δεδομένων. Οι πράκτορες εκτελούνται στους κόμβους και επηρεάζουν τη διαχείρηση των πόρων μέσω ενός API ( Application Programming Interface ). Αυτό δεν πρέπει να είναι ούτε πολύ περιοριστικό αλλά ούτε και να παρέχει πολύ μεγάλη ελευθερία. Οι λειτουργίες του API χωρίζονται σε τρεις κλάσεις, αυτές που επιτρέπουν στους πράκτορες : α) να αλλάξουν τη δομή στην ιεραρχία των πόρων, β) να επηρεάσουν τη δρομολόγηση και γ) να στείλουν και να λάβουν μηνύματα.

**Έλεγχος Συμφόρησης**

Η συμφόρηση θέτει σοβαρή υποψηφιότητα για την αρχιτεκτονική των ενεργών-προγραμματιζόμενων δικτύων γιατί είναι καθαρά ενδοδικτυακό γεγονός απομακρυσμένο από την εφαρνογή. Επίσης συνήθως χρειάζεται σημαντικός χρόνος μέχρι η πληροφορία να φτάσει στο χρήστη ώστε αυτός να συμμορφωθεί και να μειωθεί η συμφόρηση. Έτσι λοιπόν ένας ενεργός κόμβος μπορεί :

1. να παρακολουθεί το διαθέσιμο εύρος και να ελέγχει το ρυθμό μια ροής δεδομένων ανάλόγως
2. αν υπάρχουν πολλές ροές δεδομένων με διαφορετικές απαιτήσεις, να ελέγχει κάθε ροή σε σχέση και με το συνολικό ρυθμό και συγχρόνως να μπορεί να προσαρμόζεται δυναμικά στις αλαγές των απαιτήσεων.
3. να μπορεί να μετασχηματίσει τα δεδομένα στα σημεία συμφόρησης και μόνο αν αυτό είναι απαραίτητο.
4. να κάνει επιλεκτική απόρριψη μονάδων, πακέτων ή κυψελίδων με αποτελεσματικό τρόπο.

Σχετική δουλειά γίνεται στο Georgia Institute of Technology. Το Δίκτου ορίζει ένα σύνολο λειτουργιών που μπορούν να εκτελεστούν σε ένα ενεργό κόμβο του δικτύου από τον ενεργό επαξεργαστή. Επίσης στην επικεφαλίδα κάθε πακέτου υπάρχει η πληροφορία για τις λειτουργίες που θα εκτελεστούν πάνω σε αυτό που ονομάζεται Πληροφορία Ελέγχου Ενεργητικής Επεξεργασίας ( Active Processing Control Information-APCI ). Η συμβατότητα με τα υπάρχοντα πρωτόκολλα επιτυγχάνεται γιατλι τα μη ενεργά δίκτυα δε χρειάζεται να αναγνωρίσουν την APCI προκειμένου να δρομολογήσουν πακέτα και η APCI δε χρειάζεται σε πακέτα που απλά δρομολογούνται από ενεργούς κόμβους.

Όταν φτάσει ένα πακέτο σε ένα κόμβο, εκτελούνται τα παρακάτω βήματα :

1. Υπολογίζεται ο προορισμός του.
2. Αν υπάρχει η APCI τότε το πακέτο στέλνεται στον ενεργό επεξεργαστή για επιπλέον επεξεργασία. Αν όχι, το πακέτα μεταδίδεται

3.Εκτελούνται οι λειτουργίες που καθορίζονται στην APCI.
4.Η επικεφαλίδα και η APCI του πακέτου ξαναϋπολογίζονται αν το αποτέλεσμα των υπολογισμών είναι μετασχηματισμένα δεδομένα. Επίσης ενημερώνεται η κατάσταση του κόμβου όπως αυτή καθορίζεται από τους υπολογισμούς.
5.Το πακέτο μεταδίδεται.

Το συγκεκριμένο μοντέλο μπορεί να χαρακτηριστεί "συντηρητικό" γιατί δε χρησιμοποιεί όλες τις δυνατότητες που προσφέρουν τα ενεργά δίκτυα. Τα πακέτα μεταφέρουν μόνο την APCI και όχι κώδικα που θα εκτελεστεί στους κόμβους. Επίσης οι κόμβοι έχουν ένα καθορισμένο σύνολο λειτουργιών και όχι απεριόριστο. Το κίνητρο για ένα τέτοιο απλό μοντέλο είναι οτι ικανοποιεί αρκετά καλά το σκοπό του χωρίς να απαιτούνται αλλαγές στο υπόλοιπο δίκτυο.

Στο μέλλον περιμένουμε να κάνουν την εμφάνισή τους πιο ριζοσπαστικά μοντέλα τα οποία θα είναι πιο ισχυρά αλλά και πιο περίπλοκα και λιγότερο συμβατά με τα μη ενεργά δίκτυα. Μία προσέγγιση είναι να χρησιμοποιήσουμε τα προγράμματα του κάθε πακέτου τα οποία θα επιτρέπουν στο ίδιο το πακέτο να πάρει αποφάσεις δρομολόγησης στη στιγμή. Με το που θα φτάνει σε ένα κόμβο το πακέτο θα εκτελεί τον κώδικα δρομολόγησης ανάλογα με τις πληροφορίες που θα υπάρχουν σε εκείνο τον κόμβο. Οι πληροφορίες αυτές μπορεί να είναι αποτέλεσμα της επεξεργασίας πληροφοριών που φτάνουν στον κόμβο από άλλα πακέτα. Είναι δηλαδή σαν να παίζουν τα πακέτα το ρόλο συσκευών παρακολούθησης του δικτύου και παροχής πληροφοριών σε κάθε κόμβο που διασχίζουν σχετικά με ότι έχουν μάθει στο "ταξίδι" τους. Με τον τρόπο αυτό ανιχνεύονται επερχόμενες συμφορήσεις και πριν αυτές συμβούν λαμβάνονται τα κατάλληλα μέτρα.

**Multicasting**

Το Intetrnet και τα δίκτυα της επόμενης γενιάς θα πρέπει να χειρίζονται μια μεγάλη ποικιλία εφαρμογών ήχου, video, τηλεδιάσκεψης κ.ά. Πολλές από αυτές απαιτούν λειτουργίες multicasting. Η εφαρμογή των τεχνικών των ενεργών δικτύων σ'αυτόν τον τομέα μπορεί να αποδειχτεί χρήσιμη : ενδιάμεσοι ενεργοί κόμβοι μπορούν να λύσουν πολλά από τα τρέχοντα προβλήματα όπως καταιγισμός από εκπεμπόμενα NACKs, υπερφόρτωση από αναμεταδόσεις, περιττές αναμεταδόσεις, διπλασιασμός των πακέτων και ανοχή σε αλλαγές των μελών των ομάδων. Τα υπάρχοντα παθητικά σχήματα παρέχουν μόνο τμηματικές λύσεις στα παραπάνω προβλήματα. Μία τέτοια προσέγγιση που εκμεταλλεύεται τα ενεργά δίκτυα έχει αναπτυχθεί στο MIT με την επωνυμία Active Reliable Multicast.

**Active Reliable Multicast (ARM).** Αξιοποιεί τους ενδιάμεσους δρομολογητές προκειμένου να προστατέψει τον αποστολέα και το εύρος του δικτύου από περιττά ACKs και αναμεταδόσεις. Οι ARM δρομολογητές παίζουν ενεργό ρόλο στην ανάκτηση λόγω απωλειών. Παρέχουν αποθήκευση για συγκεκριμένο χρονικό διάστημα κι εφόσον έχουν ελεύθερους πόρους και συγρόνως εκτελούν διαφορετικούς υπολογισμούς ανάλογα με τον τύπο του πακέτου. Να σημειωθεί οτι δε χρειάζεται να είναι ενεργοί όλοι οι κόμβοι. Το ποσοστό αυτών, το μέγεθος των αποθηκευτικών πόρων και η "ζωή" των αποθηκευμένων αντικειμένων είναι σημαντικοί παράμετροι του σχήματος. Στο ARM χρησιμοποιούνται τρεις στρατηγικές ανάκτησης από σφάλματα :

- Η πρώτη πραγματοποιείται από τους ενεργούς δρομολογητές. Αποθηκεύουν τα δεδομένα ώστε να μπορούν να τα αναμεταδόσουν αν ζητηθεί. Αυτό ονομάζεται τοπική αναμετάδοση. Έτσι καθώς τα NACKs ταξιδεύουν προς τον

αποστολέα, μπορεί να ενεργοποιήσουν την αναμετάδοση ενός χαμένου πακέτου από έναν ενδιάμεσο κόμβο. Αυτή η αποθήκευση αποτελεί μια ανταλλαγή ανάμεσα στην αποθήκευση στο δίκτυο και το εύρος του δικτύου. Επομένως έχει νόημα αυτή η αποθήκευση να γίνεται μόνο σε στρατηγικά σημεία του δικτύου

- Η δεύτερη που επίσης πραγματοποιείται από τους ενεργούς δρομολογητές είναι η επεξεργασία των NACKs με σκοπό την αποφυγή περιττής κίνησης αιτήσεων και η λήψη πληροφοριών για την αφετηρία των αιτήσεων αναμετάδοσης. Κάθε πακέτο multicast περιέχει ένα μετρητή NACK στην επικεφαλίδα. Οι ενεργοί δρομολογητές διατηρούν δύο εγγραφές : την εγγραφή NACK και την εγγραφή REPAIR. Η πρώτη περιέχει το μεγαλύτερο μετρητή NACK που έχει ληφθεί για το συγκεκριμένο πακέτο και μια ένδειξη των εξερχόμενων συνδέσμων από τους οποίους έχουν ληφθεί NACKs. Η δεύτερη περιέχει ένα διάνυσμα που δείχνει τους εξερχόμενους συνδέσμους για τους οποίους είναι καθ'οδόν μια αναμετάδοση ώστε να αγνοηθούν μελλοντικά NACKs.
- Η τρίτη είναι η έκταση της αναμετάδοσης. Όταν φτάνει ένα πακέτο REPAIR ο δρομολογητής ελέγχει την εγγραφή NACK. Αν δεν υπάρχει καμία ένδειξη τότε προωθεί το πακέτο σε όλους τους εξερχόμενους συνδέσμους αλλιώς μόνο σε εκείνους που έχουν ζητήσει την αναμετάδοση σύμφωνα με την ένδειξη.

Η προσέγγιση ARM έχει πολύ μικρότερη καθυστέρηση ανάκτησης από τα παθητικά σχήματα. Επίσης είναι ευέλικτα και ανθεκτικά γιατί δε χρειάζονται πληροφορίες για την τοπολογία των ομάδων. Όλα τα παραπάνω μπορούν να επιτευχθούν ακόμη κι αν οι ενεργοί δρομολογητές είναι λογότεροι από τους μισούς. Βέβαια δύο ανοιχτά ζητήματα είναι ο τόπος και η διάρκεια της αποθήκευσης.

**Caching**

Η προσωρινή αποθήκευση αντικειμένων κοντά στους clients μπορεί να μειώσει τόσο την κίνηση του δικτύου όσο και το χρόνο που απαιτείται για την ανάκτηση της πληροφορίας. Τα ενεργά δίκτυα μπορούν να χρησιμοποιηθούν για να παρέχουν μεθόδους "έξυπνου" caching όπου θα απαιτείται μικρότερη συνολική αποθηκευτική χωρητικότητα και θα επιτυγχάνεται μεγαλύτερη μείωση στην κίνηση και την καθυστέρηση του δικτύου.

Στις παραδοσιακές τεχνικές τοποθετούνται μεγάλοι αποθηκευτικοί πόροι σε συγκεκριμένα σημεία του δικτύου. Μια ενδιαφέρουσα τεχνική θα ήταν η επανατοποθέτηση όχι μόνο της αποθηκευμένης πληροφορίας αλλά και των ίδιων των κόμβων. Κάθε κόμβος ή μία ομάδα κόμβων αποφασίζει αν θα αποθηκεύσει ή όχι τις πληροφορίες που επιστρέφουν από το server στον client. Οι κόμβοι θα έπρεπε να είναι αρκετά έξυπνοι ώστε να αποθηκεύσουν αντικείμενα που οι κοντινοί clients θα ζητήσουν στο μέλλον και να συνεργαστούν με γειτονικούς κόμβους ώστε να μην αποθηκευουν αντικείμενα που υπάρχουν ήδη σε γειτονικούς κόμβους.

Παραδείγμα τέτοιας προσέγγισης είναι το σχήμα "**Self-Organizing Wide-Area Network Caches"** at Georgia Institute of Technology.

Οι τεχνικές γύρω από την ενεργή αποθήκευση μπορούν να εφαρμοστούν στον τομέα του αξιόπιστου multicasting προκειμένου να καθορίζεται η διάρκεια και το σημείο αποθήκευσης δυναμικά.

## Προστασία και Ασφάλεια

Μέχρι τώρα είδαμε διάφορους τομείς όπου τα ενεργά δίκτυα μπορεί να φανούν πολύ χρήσιμα. Τώρα θα δούμε τα θέματα ασφάλειας και προστασίας που προκύπτουν. Προβλήματ που μπορεί να προκύψουν τόσο από λάθη και ακούσια συμπεριφορά όσο και από εσκεμμένες επιθέσεις κατά των ιδιωτικών στοιχείων, της ακεραιότητας και της διαθεσιμότητας. Ένα πακέτο που μεταφέρει εκτελέσιμο κώδικα μπορεί να αλλάξει την κατάσταση των κόμβων οι οποίοι αποτελούν δημόσιους πόρους και είναι κρίσιμοι για τη σωστή λειτουργία σημαντικών συστημάτων. Επομένως οι απαιτήσεις για προστασία και ασφάλεια που θα τεθούν για το υπολογιστικό περιβάλλον όπου θα εκτελείται ο κώδικας των πακέτων πρέπει να είναι πολύ αυστηρές.

Με την τρέχουσα μορφή του Internet οι μόνοι πόροι που καταναλώνονται από ένα πακέτο είναι η μνήμη που απαιτείται για την προσωρινή αποθήκευσή του και οι κύκλοι της CPU μέχρι να βρεθεί η σωστή διαδρομή. Σε ένα τέτοιο περιβάλλον φυσικά ο αυστηρός έλεγχος των πόρων των ενδιάμεσων κόμβων δεν είναι κρίσιμος. Ωστόσο, ένα ενεργό πακέτο καταναλώνει όχι μόνο πολύ περισσότερους πόρους, αλλά και με γρηγορότερο ρυθμό. Επίσης μαζί με τα θέματα προστασίας και ασφάλειας τίθεται και το θέμα της δικαιοσύνης.

Σε ένα ενεργό δίκτυο, τα ενεργά πακέτα μπορεί να κάνουν κακή χρήση των ενεργών κόμβων, τους πόρους του δικτύου και τα άλλα ενεργά πακέτα με διάφορους τρόπους.. Επίσης, μπορεί οι ενεργοί κόμβοι να κάνουν κακή χρήση των ενεργών πακέτων. Μερικά από τα πιθανά προβλήματα που μπορεί να παρουσιαστούν είναι :

- Ζημιά : Ένα ενεργό πακέτο μπορεί να καταστρέψει ή να αλλάξει τους πόρους ή τις υπηρεσίες ενός κόμβου αναδιατάσσοντας, τροποποιώντας ή σβήνοντάς τους από τη μνήμη. Ένας κόμβος μπορεί να σβήσει ένα ενεργό πακέτο πριν την ολοκλήρωση της εργασίας του στον κόμβο. Τέλος, ενεργά πακέτα που μοιράζονται τους ίδιους υπολογιστικούς πόρους μπορεί να επιτεθούν το ένα στο άλλο.
- Άρνηση Υπηρεσιών : Ένα ενεργό πακέτο μπορεί να υπερφορτώσει έναν πόρο ή μία υπηρεσία επειδή διαρκώς καταναλώνει τις συνδέσεις του δικτύου ή επειδή καταναλώνει μεγάλο μέρος των κύκλων της CPU που είναι διαθέσιμοι. Κάτω από αυτές τις συνθήκες ο κόμβος δε μπορεί να λειτουργήσει σωστά και κάποιο άλλο ενεργό πακέτο δε μπορεί να εκτελεστεί ή να προωθηθεί.
- Κλοπή : Ένα ενεργό πακέτο μπορεί να παραβιάσει και να κλέψει ιδιωτικές πληροφορίες από ένα κόμβο. Από την άλλη, ένα ενεργό πακέτο είναι “τρωτό” απέναντι στον κόμβο όταν τον επισκέπτεται, γιατί αν και είναι κρυπτογραφημένο, συνήθως πρέπει να αποκρυπτογραφηθεί προκειμένου να εκτελεστεί.
- Συνδυασμένη επίθεση : Η πραγματικά μεγαλύτερη απειλή για έναν ενεργό κόμβο είναι η συνδυασμένη επίθεση με κάποιο συγκεκριμένο σκοπό. Για παράδειγμα ένας κακοπροαίρετος χρήστης μπορεί να στείλει πολλά ενεργά πακέτα σε έναν κεντρικό δρομολογητή προκειμένου να τον “ρίξει” καταναλώνοντας όλη τη χωρητικότητα του εύρους του.

Η προστασία των κόμβων και των πακέτων σε ένα ευέλικτο περιβάλλον όπως τα ενεργά δίκτυα δεν είναι κάτι εύκολο. Μερικές τεχνικές για την προστασία των ενεργών κόμβων είναι :

- Πιστοποίηση των Ενεργών Πακέτων : Κάθε ενεργό πακέτο πρέπει να έχει κάποια χαρακτηριστικά που θα το πιστοποιούν χρησιμοποιώντας κάποιο

αλγόριθμο, όπως ένας αλγόριθμος που χρησιμοποιεί κάποια υπογραφή κλειδί. Αυτό βέβαια δεν εγγυάται οτι το ενεργό πακέτο είναι αβλαβές ή χρήσιμο. Απλά παρέχει την ασφάλεια οτι κάποιος άλλος εγγυάται γι'αυτό.

- Παρακολούθηση και Έλεγχος : Μπορεί να χρησιμοποιηθεί ένας πίνακας αναφοράς ώστε να περιορίζονται οι πληροφορίες και οι πόροι και οι υπηρεσίες του συστήματος που επιτρέπεται στα ενεργά πακέτα να έχουν πρόσβαση και να χρησιμοποιήσουν. Συμβουλεύεται μία πολιτική ασφαλείας για να καθοριστεί αν παρέχεται πρόσβαση.
- Τεχνικές Περιορισμού : Χρονικοί περιορισμοί όπως ο χρόνος που παρέχεται σε ένα πακέτο για να εκτελεστεί, περιορισμοί έκτασης όπως ο συνολικός αριθμός κόμβων που επιτρέπεται να διασχίσει ένας κόμβος, καθώς και περιορισμοί αναπαραγωγής ( ο αριθμός που επιτρέπεται σε ένα πακέτο να αναπαράγει τον εαυτό του ) είναι σημαντικοί για την αποτροπή ενόα ενεργού πακέτου από το να μονοπωλήσει τους πόρους ενός κόμβου.
- Proof Carrying Code (PCC) : βασίζεται στην παρατήρηση οτι συχνά είναι πιο εύκολο να ελέγξεις μια απάντηση παρά να την παράγεις. Για ένα κινητό πρόγραμμα, αυτός που γνωρίζει τους λόγους "κλειδιά" για τους οποίους αυτό είναι σωστό, είναι ο δημιουργός του και όχι ο κόμβος που το λαμβάνει. Επομένως μπορούμε να ταιριάξουμε το πρόγραμμα μέσα σε κάθε ενεργό πακέτο με μια απόδειξη της ορθότητάς του. Ο ενεργός κόμβος μπορεί εύκολα να ελέγξει την απόδειξη και στη συνέχεια να τρέξει το πρόγραμμα. Το δύσκολο κομμάτι είναι η δημιουργία της απόδειξης, αλλά αυτό είναι δουλειά του δημιουργού του προγράμματος.

Για την προστασία των ενεργών πακέτων προτείνονται δύο μέθοδοι : τεχνικές ανοχής σφαλμάτων και κρυπτογράφηση. Η κρυπτογράφηση αναφέρεται στην κατάσταση που τα ενεργά πακέτα δεν αποτελούνται από καθαρό κείμενο κώδικα και δεδομένων. Η κρυπτογράφηση συνήθως χρησιμοποιείται για κώδικα και δεδομένα κατά τη μεταφορά. Ωστόσο, τα προγράμματα μπορεί ακόμη και να εκτελούνται σε μορφή όχι καθαρού κειμένου, οπότε ερχόμαστε στην περίπτωση της κινητής κρυπτογράφησης. Οι τεχνικές ανοχής σε σφάλματα είναι η αντιγραφή (replication), η διατήρηση (persistence) και η ανακατεύθυνση (redirection). Η αντιγραφή σημαίνει οτι τα πακέτα αντιγράφονται σε κάθε κόμβο. Η διατήρηση σημαίνει οτι τα πακέτα αποθηκεύονται προσωρινά για την αντιμετώπιση αποτυχίας του κόμβου. Η ανακατεύθυνση σημαίνει οτι τα πακέτα θα ανζητήσουν εναλλακτικές διαδρομές σε περίπτωση που η εξ'ορισμού διαδρομή τους αποτύχει. Η αντιγραφή και η διατήρηση είναι απαράδεκτες για τη μεγάλη πλειοψηφία των πακέτων γιατί καταναλώνουν μνήμη και εύρος και θα έπρεπε να επιτρέπεται μόνο στα πολύ σημαντικά ενεργά πακέτα να το κάνουν αυτό. Η ανακατεύθυνση και η κρυπτογράφηση έχουν μεγαλύτερη εφαρμογή στην προστασία των πακέτων γιατί βασικά καταναλώνουν κύκλους της CPU. Πιθανότατα ένας συνδυασμός τεχνικών ανοχής σφαλμάτων και κρυπτογράφησης ίσως να δώσει πολύ καλά αποτελέσματα στο πρόβλημα της προστασίας των ενργών πακέτων.

Συνδυάζοντας όλα τα παραπάνω, όταν ένα πακέτο που περιέχει εκτελέσιμο κώδικα φτάνει σε ένα κόμβο, το σύστημα θα πρέπει να :

1. Αποδεχτεί την αυθεντικότητα των πιστοποιητικών του πακέτου
2. Αναγνωρίσει το στοιχείο αποστολής
3. Αναγνωρίσει το χρήστη-αποστολέα

4. Να παραχωρίσει πρόσβαση στους κατάλληλους πόρους σύμφωνα με την ταυτότητα και τα πιστοποιητικά
5. Επιτρέψει την εκτέλεση σύμφωνα με την πιστοποίηση και την πολιτική ασφάλειας
6. Παρακολουθεί και να ελέγχει την πρόσβαση στους πόρους του συστήματος σε όλη τη διάρκεια της εκτέλεσης
7. Κρυπτογραφήσει αν χρειάζεται το πακέτο για να προστατέψει τον κώδικα και τα δεδομένα στη μεταφορά.

Σε περίπτωση που το πακέτο δεν αναγνωριστεί σωστά, τότε είτε θα εκτελέσει τον κώδικα σε ένα περιορισμένο περιβάλλον ή δε θα τον εκτελέσει καθόλου.

**Secure Active Network Environment**

Παρουσιάζεται εδώ μια αρχιτεκτονική με το όνομα SANE η οποία έχει αναπτυχθεί από ερευνητές του University of Pennsylvania. Η ασφάλεια πρέπει να αντιμετωπιστεί σε δύο επίπεδα : στατικό και δυναμικό. Το στατικό αναφέρεται αναφέρεται σε αυτά που πρέπει να ελέγχονται σπάνια, όπως στην περίπτωση της εκκίνησης λειτουργίας ενός ενεργού δικτύου. Το δυναμικό αναφέρεται σε εκείνους τους ελέγχους που πρέπει να γίνονται διαρκώς (ανά πακέτο). Το πρώτο μπορεί να είναι ακριβό αλλά πολύ αξιόπιστο ενώ το δεύτερο πρέπει να είναι αρκετά φθηνό ώστε να μη μειώνεται η απόδοση. Χρειάζεται ένας συνδυασμός των δύο ελέγχων, στατικών και δυναμικών) για να εξασφαλιστεί η ασφάλεια.

Η αρχιτεκτονική SANE είναι αρχιτεκτονική επιπέδων. Τα χαμηλότερα στρώματα εξασφαλίζουν οτι το σύστημα ξεκινά σε μια αναμενόμενη κατάσταση (αρχιτεκτονική AEGIS) και στη συνέχεια τα υψηλότερα στρώματα είναι υπεύθυνα για τους δυναμικούς ελέγχους που γίνονται ανά πακέτο ή ανά χρήστη. Από αυτό το σημείο και μετά το σύστημα χρησιμοποιεί διάφορες μεθόδους προκειμένου να εξασφαλίσει την ασφάλεια :

- Πρώτον, εκτελεί απομακρυσμένη πιστοποίηση, όταν χρειάζεται, για την πιστοποίησηση από κόμβο σε κόμβο
- Δεύτερον, παρέχει ένα περιορισμένο περιβάλλον εκτέλεσης για τον υπολογισμό των προγραμμάτων που λαμβάνονται από το δίκτυο
- Τέλος, χρησιμοποιεί ένα νέο σχήμα ονομάτων για να διαμοιράσει το χώρο των υπηρεσιών του κόμβου στους χρήστες.

Είναι σημαντικό να τονίσουμε οτι κάθε χρήστης και κάθε ενεργό στοιχείο κατέχουν ένα ζευγάρι δημόσιου/ιδιωτικού κλειδιού, τα οποία χρησιμοποιούνται για να πιστοποιήσουν και να εξουσιοδοτήσουν τις πράξεις αυτών των οντοτήτων.

**Προστασία και Ασφάλεια από την Προγραμματιστική πλευρά**

Είδαμε τα θέματα της ασφάλειας από την πλευρά του συστήματος. Ωστόσο, τα θέματα της ασφάλειας από την πλευρά του προγραμματισμού δημιουργούν ενδιαφέρουσες προοπτικές. Για παράδειγμα, μια καλά σχεδιασμένη γλώσσα προγραμματισμού για ενεργά δίκτυα μπορεί να λύσει πολλά από τα προβλήματα ασφάλειας, όπως ελαχιστοποιώντας τους ελέγχους την ώρα της εκτέλεσης και επομένως να βελτιώσει την απόδοση. Επίσης μια "ήπια" γλώσσα προγραμματισμού, με περιορισμένη ενδεχομένως λειτουργικότητα, μπορεί ακόμη και να εξαλείψει την την ανάγκη για την ασφάλεια της SANE ή κάποιας άλλης αρχιτεκτονικής.

Ο στόχος μιας γλώσσας προγραμματισμού για ενεργά δίκτυα είναι να παρέχει ασφάλεια και ακεραιότητα και χωρίς συμβιβασμούς από την πλευρά της απόδοσης. Από τη στιγμή που η γλώσσα προγραμματισμού καθορίζει τις λειτουργίες που μπορεί να εκτελέσει ο προγραμματιστής, μπορούμε επιλέγοντας ή σχεδιάζοντας προσεχτικά μια γλώσσα να περιορίσουμε κάποιες από τις ανεπιθύμητες ενέργειες που ένας προγραμματιστής μπορεί να κάνει, ακούσια ή εκούσια. Επίσης μπορούμε να βελτιώσουμε την απόδοση αφού η ταχύτητα της εκτέλεσης των ενεργών πακέτων εξαρτάται σε μεγάλο βαθμό από τη γλώσσα. Ενδεχόμενα προβλήματα που μπορεί να εμφανιστούν και οι πιθανές λύσεις είναι :

- Αναφορά σε αυθαίρετες περιοχές της μνήμης : για το λόγο αυτό πρέπει να χρησιμοποιούνται αυστηρά δομημένες γλώσσες όπως η Java ενώ γλώσσες όπως η C είναι ακατάλληλες γι'αυτό το σκοπό.
- Κατάληψη μεγάλου μέρους της μνήμης ή δημιουργία πολλών υπορουτίνων : η λύση είναι να μπουν όρια στη μνήμη και τον αριθμό των υποροτίνων που μπορεί να έχει μια διαδικασία.
- Χαμηλή απόδοση λόγω ελέγχων κατά την εκτέλεση : η υλοποίηση των τεχνικών προστασίας και ασφάλειας στα σημερινά συστήματα απαιτεί ελέγχους κατά την εκτέλεση. Αυτοί οι έλεγχοι επιβαρύνουν την εκτέλεση του προγράμματος και για λειτουργίες με αυστηρούς περιορισμούς απόδοσης οι καθυστερήσεις αυτές μπορε ί να είναι απράδεκτες. Ο προσδιορισμός μιας γλώσσας που αντικαθιστά τους δυναμικούς ελέγχους κατά την εκτέλεση με στατικούς είναι μια λύση. Επομένως μπορούμε να πούμε οτι μια καλά σχεδιασμένη-δομημένη γλώσσα μπορεί να εξαλείψει την ανάγκη για ελέγχους κατά την εκτέλεση.
- Χαμηλή απόδοση λόγω της εκτέλεσης κώδικα : εφόσον η προώθηση των πακέτων πρέπει να είναι γρήγορη, η εκτέλεση του κώδικα των ενεργών πακέτων πρέπει να είναι όσο το δυνατόν γρηγορότερη. Ένας τρόπος για να επιταχυνθεί η διαδικασία της προώθησης είναι να διαχωριστεί αυτή από τους υπόλοιπους υπολογισμούς.

**SafetyNet**

Είναι ένα πρόγραμμα γύρω από τη σχεδίαση μιας γλώσσας προγραμματισμού για ενεργά δίκτυα στο University of Sussex. Το αποτέλεσμα είναι ένα μοντέλο που προστατεύει τη διασύνδεση των δικτύων και συγχρόνως θέτει συγκεκριμένες απαιτήσεις για να εξασφαλίσει προστασία, ασφάλεια και δίκαιη κατανομή πόρων.

Η σχεδίαση βασίζεται σε τρεις κύριες κατευθύνσεις : επικοινωνία, προστασία-ασφάλεια και προγραμματισμό. Το μοντέλο που προκύπτει επιτρέπει τη δημιουργία νέων εφαρμογών πάνω στα ενεργά δίκτυα και συγχρόνως προστατεύει τους κοινούς πόρους από κακόβουλες ενέργειες και λανθασμένα προγράμματα. Οι τρεις κατευθύνσεις που αναφέραμε θέτουν κάποιες απαιτήσεις που ακολουθούνται από το προγραμματιστικό μοντέλο. Κάποιες από αυτές έχουν ως εξής :

**Επικοινωνιακές απαιτήσεις :** Δεδομένου ενός προορισμού, ένα πρόγραμμα θα πρέπει να μπορεί να υπολογίσει μόνο το επόμενο άλμα στο μονοπάτι προς τον προορισμό. Επίσης, το επόμενο άλμα στο ενεργό δίκτυο θα πρέπει να είναι διαφανές για όλους τους ενδιάμεσους μη ενεργούς κόμβους οι οποίοι μεσολαβούν ανάμεσα σε δύο ενεργούς ( σχηματισμός Abone όπως MBone για multicasting ).Ακόμη, η κατάσταση ενός ενεργού κόμβου θα πρέπει να διατηρείται, να επανεγκαθείσταται, να είναι σωστή και να μη χάνεται λόγω αυθαίρετων επανεκκινήσεων ή αποσυνδέσεων. Τέλος, θα πρέπει να είναι διαθέσιμη η κατάσταση του δικτύου για κάθε επόμενο

άλμα, ώστε οι αποφάσεις για πιθανούς τρόπους δράσης να βασίζονται σε εκτιμήσεις των χαρακτηριστικών του δικτύου.

**Απαιτήσεις σε προστασία και ασφάλεια :**

- Απαιτήσεις σε Εύρος : Η γέννηση πακέτων από την αποστολή ενός πακέτου στο δίκτυο πρέπει να είναι περιορισμένη. Ο αριθμός των πακέτων που δημιουργούνται ανά χρονική μονάδα από ένα μοναδικό κόμβο πρέπει επίσης να είνα περιορισμένος.
- Απαιτήσεις σε Χρόνο Επεξεργασίας : Ο κώδικας προώθησης των πακέτων δεν πρέπει να οδηγείται σε ατελείωτους κύκλους. Πρέπει να υπάρχει ένα όριο στο χρόνο επεξεργασίας που θα καταναλώνεται.
- Απαιτήσεις σε Μνήμη : Θα πρέπει να υπάρχει ένα όριο στη μνήμη που μπορεί να χρησιμοποιήσει ένα πρόγραμμα σε ένα κόμβο. Επίσης τα προγράμματα δεν πρέπει να δημιουργούν αυθαίρετες αναφορές στη μνήμη. Τέλος, ένα πρόγραμμα ενεργού δικτύου δε θα πρέπει να χειρίζεται άμμεσα τον πίνακα δρομολόγησης ενός ενεργού κόμβου, αλλά έμμεσα, με την κλήση κάποιων σχετικών ρουτινών από το πρωτόκκολο δρομολόγησης.
- Απαιτήσεις σε ασφάλεια : Το μοντέλο ασφάλειας του Safetynet βασίζεται σε ένα σύνολο "έμπιστων" κόμβων και "έμπιστου" κώδικα που προατατεύεται με τεχνικές κρυπτογράφησης. Πρέπει να είναι δυνατή η δημιουργία μιας "αλυσίδας εμπιστοσύνης" από ένα δεδομένο κόμβο σε έναν "έμπιστο" κόμβο και αυτή η αλυσίδα να μην μπορεί να πλαστογραφηθεί.

**Απαιτήσεις σε προγραμματισμό :** Η σημαντικότερη απαίτηση είναι η αντικατάσταση των ελέγχων κατά την εκτέλεση με στατικούς ελέχγους. Οι στατικοί έλεγχοι είναι συνήθως πιο πολύπλοκοι αλλά πρέπει να εκτελεστούν μόνο μια φορά. Η χρήση ενός αυστηρού περιβάλλοντος προγραμματισμού που ενσωματώνει τις πολιτικές ασφάλειας και προστασίας μέσα στο σύστημα δήλωσης και δόμησης του προγράμματος, επιτρέπει να αποδειχθεί στατικά οτι ένα πρόγραμμα είναι ασφαλές. Μία γλώσσα που συμμορφώνεται σε όλα τα παραπάνω ίσως δεν είναι αρκετή για να παρέχει ασφάλεια σε ένα περιβάλλον ενεργών δικτύων, θα εξαλείψει ωστόσο τους χρονοβόρους ελέγχους κατά την εκτέλεση, που πολλές από τις σημερινές γλώσσες έχουν. Από τη στιγμή που η απόδοση είναι ένα θέμα κλειδί κι αυτό είναι πολύ σημαντικό.

**PLAN - Programming Language for Active Networks.**

Είναι παράδειγμα μιας γλώσσας που αναπτύχθηκε ειδικά για τα ενεργά δίκτυα. Είναι μια νέα γλώσσα για τα προγράμματα που σχηματίζουν τα πακέτα ενός προγραμματιζόμενου δικτύου. Αυτά τα προγράμματα αντικαθιστούν τις επικεφαλίδες των πακέτων που χρησιμοποιούνται στα σημερινά δίκτυα. Η βασική επιλογή σχεδίασης της PLAN είναι η κατασκευή προγραμμάτων που είναι "αδύναμα" και με περιορισμένη λειτουργικότητα. Οι περιορισμοί στις δυνατότητες της PLAN μετριάζονται επιτρέποντας στον κώδικα να καλέσει άλλες ρουτίνες, που ονομάζονται ρουτίνες-υπηρεσίες και οι οποίες βρίσκονται στους κόμβους και είναι γραμμένες σε άλλες πιο ισχυρές γλώσσες προγραμματισμού. Επειδή τα προγράμματα σε PLAN είναι "αδύναμα" δε γίνεται καθόλου πιστοποίηση. Ωστόσο, για τις ρουτίνες-υπηρεσίες που βρίσκονται στους κόμβους γίνεται πιστοποίηση όπου χρειάζεται. Η γλώσσα PLAN αντιμετωπίζει τα θέματα προστασίας και ασφάλειας, απόδοσης και ευελιξίας ως εξής :

- Ασφάλεια και Προστασία. Οι απαιτήσεις σε εύρος, χρόνο επεξεργασίας και μνήμη αντιμετωπίζονται με δύο τρόπους. Πρώτον, τα προγράμματα σε PLAN

είναι εξασφαλισμένο οτι θα τερματίσουν. Κι αυτό γιατί λείπουν από τη γλώσσα οι κλήσεις συναρτήσεων και οι επαναλήψεις χωρίς όρια Δεύτερον, τα προγράμματα σε PLAN έχουν περιορισμούς στους πόρους που μπορούν να καταναλώσουν. Ο αριθμός των "μονάδων" πόρων που μπορούν να παραχθούν από ένα πακέτο και το μέγεθος κάθε μίας περιορίζονται από μετρητές. Είναι μια γλώσσα καθαρά διαδικαστική και αυστηρά δομημένη, επομένως τα προγράμματα ελέγχονται στατικά πριν εισέλθουν στο δίκτυο. Δε χρησιμοποιούνται δείκτες και προγράμματα που τρέχουν συγχρόνως δε μπορούν να επηρεαστούν μεταξύ τους.

- Απόδοση. Το μεγάλο πλεονέκτημα της απλότητας της PLAN είναι οτι η μεταγλώττισή της είναι γρήγορη διαδικασία και κάποιες απλές εργασίες γίνονται γρήγορα και εύκολα.
- Ευελιξία. Η γλώσσα PLAN δεν είναι εντελώς γενική, αλλά μπορεί να εκφράσει προγράμματα για τη σύνθεση και τα διαγνωστικά δικτύων και να παρέχει την κατανεμημένη υπολογιστική διασύνδεση ανάμεσα στις υπηρεσίες που βρίσκονται εγκατεστημένες στους κόμβους και άλλα μεγαλύτερα πρωτόκολλα.

## Αρχιτεκτονικές

Στο κομμάτι αυτό θα περιγράψουμε μερικές αρχιτεκτονικές για ενεργά δίκτυα. Αυτές ομαδοποιούνται ανάλογα με τη βασική τους προσέγγιση για την υλοποίηση του ενεργού δικτύου : σε μερικές αρχιτεκτονικές τα ενεργά πακέτα μεταφέρουν εκτελέσιμο κώδικα, συνύθως εκτελέσιμο πάνω στα δεδομένα που μεταφέρει το ίδιο το πακέτο. Σε άλλες αρχιτεκτονικές τοποθετούν τον εκτελέσιμο κώδικα στους ενεργούς κόμβους και τα πακέτα μεταφέρουν δείκτες που δηλώνουν τον κώδικα που θα εκτελεστεί γι'αυτά. Στην περίπτωση αυτή χρησιμοποιούνται τεχνικές για κατανομή και μεταφορά ώστε να μη χρειάζεται όλοι οι κόμβοι να έχουν όλους τους κώδικες που πιθανά να χρησιμοποιήσουν. Τέλος, κάποιες αρχιτεκτονικές αφήνουν το χρήστη να επιλέξει ανάμεσα σε αδύναμο κώδικα που μεταφέρεται από ενεργά πακέτα ή ισχυρό κώδικα που βρίσκεται στους κόμβους.

### Η προσέγγιση των Ενεργών Πακέτων ( Active Packets )

Ο κώδικας μεταφέρεται από τα πακέτα. Οι κόμβοι είναι επίσης ενεργοί γιατί επιτρέπουν υπολογισμούς μέχρι και το επίπεδο της εφαρμογής, αλλά δεν υπάρχει καθόλου ενεργός κώδικας σε αυτούς. Το όνομα της προσέγγισης οφείλεται στον ενεργό κώδικα που μεταφέρεται από τα πακέτα είτε για να εκτελεστεί πάνω στα δεδομένα του ίδιου πακέτου, είτε για να εκτελεστεί για να αλλάξει την κατάσταση ή τη συμπεριφορά του κόμβου. Παραδείγματα τέτοιων αρχιτεκτονικών είναι : Smart Packets Project - BBN Technologies , Active IP Option - MIT , M0 architecture - University of California (Berkeley) and University of Zurich.

**"Εξυπνα" Πακέτα (Smart Packets).** Στο πρόγραμμα αυτό πάρθηκαν δύο σημαντικές αποφάσεις σε μια προσπάθεια να δημιουργηθεί ένα πλούσιο και ευέλικτο περιβάλλον προγραμ-ματισμού χωρίς να υπερφορτωθεί η υπολογιστική δύναμη του διαχειριζόμενου κόμβου και χωρίς να γίνει το περιβάλλον αυτό τόσο πολύπλοκο ώστε να είναι δύσκολη η ασφάλειά του. Η πρώτη ήταν οτι τα προγράμματα πρέπει να είναι αυτάρκη, να χωρούν σε ένα πακέτο. Η δεύτερη ήταν οτι το λειτουργικό περιβάλλον

πρέπει να παρέχει προστασία και ασφάλεια γιατί τα πακέτα που περιέχουν εκτελέσιμο κώδικα είναι εξαιρετικά επικίνδυνα.

Ένα ειδικό πρωτόκολλο με την ονομασία Active Network Encapsulation Protocol (ANEP) δημιουργήθηκε για το πρόγραμμα ενεργών δικτύων του DARPA ώστε να υπάρχει συμβατότητα μεταξύ των διαφόρων ερευνητικών προγραμμάτων για τα ενεργά δίκτυα. Τα προγράμματα παρακολούθησης και διαχείρησης δημιουργούν τα έξυπνα πακέτα. Αυτά ενσωματώνονται στα ANEP πακέτα και αυτά με τη σειρά τους ενσωματώνονται μέσα σε ένα IP πακέτο. Τα έξυπνα πακέτα είτε στέλνονται σε κάποιο τελικό χρήστη, όπου και εκτελείται το περιεχόμενό του και τα αποτελέσματα επιστρέφουν πίσω, είτε σε κάθε δρομολογητή κατά μήκος του μονοπατιού που ακολουθεί το πακέτο μέχρι τον τελικό προορισμό και εκτελείται σε κάθε έναν από αυτούς. Ένα κομμάτι σε κάθε κόμβο είναι υπέυθυνο για να εκπέμπει και να λαμβάνει έξυπνα πακέτα και για να παρέχει ένα ασφαλές περιβάλλον, την ιδεατή μηχανή, για την εκτέλεση των προγραμμάτων.

Ο κώδικας των έξυπνων πακέτων μπορεί να γραφτεί είτε σε Sprocket, γλώσσα υψηλού επιπέδου μοιάζει με τη C, είτε σε  Spanner, μια γλώσσα assemply. Τα προγράμματα σε Sprocket μεταφράζονται σε κώδικα Spanner και αυτός με τη σειρά του μεταγλωττίζεται σε μια δυαδική κωδικοποίηση ανεξάρτητη μηχανής που τοποθετείται στα προγράμματα των πακέτων. Χρησιμοποιήθηκαν δύο νέες γλώσσες προγραμματισμού γιατί οι ήδη υπάρχουσες δε μπορούν να κωδικοποιήσουν παρά μόνο τετριμένα προγράμματα σε 1 Kb και καμιά δεν είχε κωδικοποίηση συμπικνωμένη και ανεξάρτητη από πλατφόρμα.

Σε ότι αφορά την ασφάλεια, τα έξυπνα πακέτα πετυχαίνουν την σωστή λειτουργία ενός δρομολογητή και της σύνθεσής του, κάνοντας συντηρητική εκτίμηση των προγραμμάτων ( αν μια ιδεατή μηχανή δεν ξέρει πώς να χειριστεί μια κατάσταση, σταματά την εκτέλεση και στέλνει ένα πακέτο λάθους στην πηγή του προγράμματος), ελέγχοντας αν ένα πρόγραμμα προέρχεται από κάποιο εξουσιο-δοτημένο χρήστη, ελέγχοντας την ακεραιότητα των δεδομένων του προγράμματος σε κάθε κόμβο και βάζοντας όρια στην εκτέλεση των προγραμμάτων.

**Active IP Option.** Περιγράφει μια επέκταση του IP μηχανισμού που υποστηρίζει την ενσωμάτωση τμημάτων προγράμματος σε datagrams και τον υπολογισμό αυτών των τμημάτων καθώς διασχίζουν το δίκτυο. Τα σημερινά παθητικά πακέτα αντικαθιστούνται από ενεργές κάψουλες, που είναι μικρά προγράμματα που εκτελούνται καθώς ταξιδεύουν. Αυτές οι κάψουλες μπορούν να καλέσουν προκαθορισμένες λειτουργίες που αλληλεπιδρούν με το περιβάλλον του τοπικού κόμβου ή ακόμη να αφήσουν πληροφορίες στον κόμβο που επισκέφτηκαν. Οι επόμενες κάψουλες μπορεί να μεταφέρουν κώδικα που εξαρτάται από αυτές τις πληροφορίες.

Υπάρχουν δύο προσεγγίσεις. Στην πρώτη μεταφέρουν τμήματα κώδικα κωδικοποιημένου σε διάφορες γλώσσες και στη δεύτερη ζητούν από τον κόμβο ποιες γλώσσες υποστηρίζει.

Οι Active Options μπορούν να εκτελέσουν λειτουργίες δρομολόγησης, αντιγραφής και συνένωσης. Το περιβάλον επεξεργασίας μπορεί να εξετάσει τις συνθήκες του δικτύου, να σταλεί το τρέχων datagram και να δημιουργηθούν επιπλέον datagrams και να σταλούν. Μπορεί να μεταβληθεί η κατάσταση του κόμβου. Επειδή βασίζεται στο IP οι δυνατότητες της τεχνολογίας είναι περιορισμένες. Η γλώσσα που χρησιμοποιήθηκε για την πρώτη υλοποίηση της αρχιτεκτονικής είνα η TCL. Η επεξεργασία γίνεται από έναν stripped-down TCL interpreter που οδηγεί σε ένα

περιορισμένο περιβάλλον παρόμοιο με εκείνο της safe-TCL. Αυτός είναι και ο μόνος τρόπος με τον οποίο αντιμετωπίζονται τα θέματα ασφάλειας και προστασίας.

**M0 Αρχιτεκτονική.** Ο "αγγελιοφόρος" (messenger) είναι παρόμοιος με την κάψουλα ή το έξυπνο πακέτο. Οι "αγγελιοφόροι" είναι προγράμματ που ανταλλάσονται μεταξύ των M0 κόμβων. Σε έναν M0 κόμβο υπάρχουν τέσσερα στοιχεία : συμπίπτουσες γραμμές αγγελιοφόρων, μια διαμοιραζόμενη περιοχή μνήμης, ένας απλός μηχανισμός συγχρονισμού και κανάλια προς τους γειτονικούς κόμβους.

Κάθε "αγγελιοφόρος" εκτελείται από μια ανώνυμη και ανεξάρτητη γραμμή ελέγχου. Αυτές οι γραμμές έχουν το δικό τους ιδιαίτερο χώρο μνήμης και είναι προστατευμένες η μία από την άλλη. Οι γραμμές των αγγελιοφόρων μπορούν να αφήσουν αφηρημένες δομές δεδομένων με ονόματα που έχουν επιλέξει οι ίδιες ώστε να μπορούν να είναι προσβάσιμες και από άλλες γραμμές. Οι ουρές γραμμών είναι ένας τρόπος σειριοποίησης της εκτέλεσης των γραμμών ώστε να αποφευχθούν συνθήκες ανταγωνισμού.

Ο κώδικας των αγγελιοφόρων είναι γραμμένος στην M0 γλώσσα. Είναι μια γλώσσα υψηλού επιπέδου που κληρονομεί από την PostScript τις βασικές ιδέες των τελεστέων, του λεξικού, της στοίβας εκτέλεσης καθώς και τους τελεστές χειρισμού δεδομένων και ελέγχου ροής. Ο μεταγλωττιστής M0 είναι γραμμένος σε C. Η Αρχιτεκτονική M0 δεν έχει λειτουργίες αποθήκευσης ή μεταφοράς. Ο κώδικας μεταφέρεται με κάθε αγγελιοφόρο. Αυτό λειτουργεί ικανοποιητικά για μικρά πρωτόκολλα. Για μεγάλους κώδικες οι αγγελιοφόροι εφαρμόζουν τη δική τους μέθοδο αποθήκευσης, αποθηκεύοντας τον κώδικα στη διαμοιραζόμενη μνήμη ενός κόμβου με ένα επιλεγμένο όνομα. Η επιλογή αυτή επιτρέπει την ανάπτυξη κάθε πρωτοκόλλου όσο πολύπλοκο κι αν είναι αυτό. Αυτό κάνει την M0 αρχιτεκτονική πιο ισχυρή από τις άλλες δύο.

Κάθε M0 κόμβος διαχειρίζεται τους πόρους του ανεξάρτητα από τους άλλους κόμβους. Όλοι οι πόροι έχουν κάποια "τιμή" που εξαρτάται από το φόρτο του συγκεκριμένου πόρου και από τη ζήτηση από τις γραμμές που τρέχουν. Οι γραμμές των αγγελιοφόρων "χρεώνονται" για τις ενέργειές τους. Όταν τους "τελειώσουν τα χρήματα" αποσύρονται αθόρυβα από το σύστημα. Στην αρχή κάθε γραμμη αγγελιοφόρου παίρνει έναν λογαριασμό με κάποια "χρήματα" που είναι αρκετά για κάποια αναζήτηση μέσα στον κόμβο και την αποστολή ενός νέου αγγελιοφόρου. Δεν υπάρχουν πιστοποιήσεις μεταξύ των κόμβων ούτε ταυτότητες. Τα θέματα ασφαλείας που αφορούν την κατανάλωση των πόρων χειρίζονται με τον έλεγχο της ροής των "χρημάτων". Ο έλεγχος της πρόσβασης σε πόρους του συστήματος του κόμβου γίνεται με κάποια συμφωνία ανάμεσα στον αγγελιοφόρο και το σύστημα. Η αρχιτεκτονική M0 παρέχει κάποιους κρυπτογραφικούς τελεστές που μπορούν να επικαλεστούν από έναν αγγελιοφόρο.

## Η προσέγγιση των Ενεργών Κόμβων ( Active Nodes )

Σ'αυτήν την προσέγγιση τα πακέτα δε μεταφέρουν κώδικα αλλά κάποιους δείκτες ή αναφορές σε προκαθορισμένες συναρτήσεις που βρίσκονται στους ενεργούς κόμβους. Τα πακέτα είναι ενεργά με την έννοια ότι αποφασίζουν ποιες λειτουργίες θα εκτελεστούν στα δεδομένα τους και παρέχουν τις παραμέτρους γι'αυτές τις λειτουργίες. Το κίνητρο γι'αυτές τις αρχιτεκτονικές είναι το οτι οι προσεγγίσεις των ενεργών πακέτων πάσχουν από προβλήματα που έχουν να κάνουν είτε με την απόδοση είτε με τις δυνατότητες των πακέτων.

**An Architecture for Active Networks.** Εδώ οι χρήστες ελέγχουν την ενεργοποίηση προκαθορισμένων λειτουργιών που βρίσκονται στους ενεργούς κόμβους.με τη χρληση πληροφοριών ελέγχου που βρίσκονται στις επικεφαλίδες των πακέτων. Μπορουν να επιλέξουν από ένα διαθέσιμο σύνολο συναρτήσεων για να υπολογιστούν πάνω στα δεδομένα τους και μπορούν να δώσουν και παραμέτρους σαν είσοδο για αυτούς τους υπολογισμούς. Οι διαθέσιμες συνασρτήσεις επιλέγονται και παρέχονται από τον παροχέα υπηρεσιών του δικτύου και υπηρεσίες υποστήριξης. Έτσι οι χρήστες μπορούν να επηρεάσουν τον υπολογισμό μιας επιλεγμένης λειτουργίας αλλά όχι να προσδιορίσουν αυθαίρετες λειτουργίες για να εκτελεστούν. Αυτή η προσέγγιση έχει κάποια πλεονεκτήματα σε ότι αφορά την ανάπτυξη, την ασφάλεια και την αποτελασματικότητα αλλά είναι κάπως περιοριστική γιατί μπορούν να κληθούν μόνο συναρτήσεις που έχουν φορτωθεί από πριν.

Για κάθε συνάρτηση και κάθε παράμετρο υπάρχει ένα συγκεκριμένο υποσύνολο των στοιχείων της κατάστασης του κόμβου που μπορούν αν μεταβληθούν.

Το ενεργό χαρακτηριστικό της προσέγγισης αυτής είναι η αυξητική προσθήκη συναρτήσεων ελεγχόμενων από τους χρήστες. Κάθε συνάρτηση ορίζεται πλήρως και υποστηρίζει μία συγκεκριμένη υπηρεσία. Η δημιουργά μιας νέας συνάρτησης ενεργού δικτύου απαιτεί τον προσδιορισμό του αναγνωριστικού της, των παραμέτρων της και της σημασιολογίας της. Όταν οριστεί μια συνάρτηση, ο παροχέας ή ο κατασκευαστής είναι ελεύθερος να την υλοποιήσει με τρόπο που θα ακολουθεί τις προδιαγραφές της.

Αυτή η προσέγιση έχει συμβατότητα προς τα πίσω γιατί δε χρειάζεται όλοι οι χρήστες να γνωρίζουν την ενεργή λειτουργικότητα του δικτύου και δε χρειάζεται όλοι οι κόμβοι να υποστηρίζουν τις ίδιες λειτουργίες. Μπορεί να έχει μικρή ελαστικότητα και περιορισμένες δυνατότητες, πετυχαίνει όμως υψηλή απόδοση γιατί η ασφάλεια εξασφαλίζεται εύκολα.

**Αρχιτεκτονική DAN - Distributed Code Caching for Active Networks.** Τα πακέτα σ'αυτή την αρχιτεκτονική περιέχουν μια πεπερασμένη ακολουθία δεικτών και παραμέτρων για τις συναρτήσεις. Οι συναρτήσεις είναι daisy-chained με την έννοια οτι η μία καλεί την επόμενη ανάλογα με τη σειρά των δεικτών στο πακέτο. Η πρώτη συνάρτηση δεν δηλώνεται από κάποιο δείκτη αλλά ερμηνεύεται από το περιεχόμενο στο οποίο ξεκινά η επεξεργασία του πακέτου (π.χ. ένα πακέτο που λαμβάνεται από μια κάρτα Ethernet οδηγεί στην κλήση μιας Ethernet συνάρτησης εισόδου).

Αν ο κόμβος δε μπορεί να εντοπίσει μια συνάρτηση αναβάλλει προσωρινά την επεξεργασία του πακέτου και καλεί έναν "server κώδικα" για την υλοποίηση της συνάρτησης. Ο "server κώδικα" είναι ένας γνωστός κόμβος του δικτύου που περιέχει βιβλιοθήκη με συναρτήσεις για διάφορα λειτουργικά συστήματα από διάφορους κατασκευαστές. Όταν φορτωθεί το πρόγραμμα αποθηκεύεται τοπικά στον κόμβο. Η διαδικασία αυτή δίνει μεγαλύτερη ευελιξία γιατί κάθε φορά που αναπτύσσεται μια νέα λειτουργία μπορεί να φορτωθεί μόνο στους "servers κώδικα" και όχι σε όλους τους κόμβους του δικτύου.

Οι συναρτήσεις γράφονται σε γλώσσα υψηλού επιπέδου ( όπως C ) και μεταγλωττίζονται σε object κώδικα. Όταν οι συναρτήσεις φορτώνονται στους κόμβους είναι στην ίδια μορφή που μεταγλωττίζονται και κατά τη δημιουργία τους, άρα τρέχουν αρκετά γρήγορα και η απόδοση είναι καλή. Κάποια καθυστέρηση υπάρχει όταν απαιτείται η μεταφορά οπότε μειώνεται η συνολική απόδοση οπότε μια λύση θα ήταν να φορτώνονται οι λειτουργίες πριν απαιτηθούν από τους κόμβους.

Τα θέματα της ασφάλειας αντιμετωπίζονται με τη χρήση γνωστών "server κώδικα" που πιστοποιούν τους εαυτούς τους, δίνουν τη δυνατότητα ελέγχου της

πηγής των πργραμμάτων και παρέχουν προγράμματα με ηλεκτρονική υπογραφή μόνο από γνωστούς κατασκευαστές. Έτσι το πρόβλημα είναι απλά η εφαρμογή ενός κανόνα για την επιλογή του σωστού server και μιας βάσης με κλειδιά για τον έλεγχο των υπογραφών των κατασκευαστών. Επίσης ακόμη κι αν οι πηγές και τα ίδια τα προγράμματα πιστοποιούνται οι διαχειριστές των δικτύων μπορούν να περιορίσουν το σύνολο των κατασκευαστών από τους οποίους δέχονται προγράμματα.

**ANTS -** Στην προσέγγιση αυτή, νέα αυθαίρετα πρωτόκολλα αναπτύσσονται αυτόματα και στους ενδιάνεσους κόμβους αλλά και στα τελικά συστήματα, με τη χρήση τεχνικών μεταφερόμενου κώδικα. Το δίκτυο αντιμετωπίζεται σαν ένα κατανεμημένο σύστημα προγραμματισμού. Χρησιμοποιεί τρία μέρη/σχήματα : κάψουλες, ενεργούς κόμβους και κατανομή κώδικα.

Οι κάψουλες αντικαθιστούν το πακέτο. Περιέχουν μια αναφορά στη ρουτίνα προώθησης που θα χρησιμοποιηθεί για να επεξεργαστεί την κάψουλα σε κάθε ενεργό κόμβο. Οι αναφορές και οι ρουτίνες προώθησης είναι κάτι αντίστοιχο με τους δείκτες και τις συναρτήσεις στην αρχιτεκτονική DAN. Μερικές ρουτίνες είναι διαθέσιμες σε κάθε ενεργό κόμβο ενώ άλλες είναι ειδικές για κάποια εφαρμογή. Αυτές θα πρέπει να μεταφέρονται στον κόμβο με ένα μηχανισμό κατανομής κώδικα πριν οι κάψουλες αυτού του τύπου δεχτούν επεξεργασία για πρώτη φορά. Οι σχετικοί τύποι από κάψουλες σχηματίζουν μια ομάδα κώδικα. Οι ρουτίνες προώθησης μιας ομάδας κώδικα μεταφέρονται σαν μια μονάδα από το σύστημα κατανομής κώδικα. Οι συσχετιζόμενες ομάδες κώδικα σχηματίζουν ένα πρωτόκολλο. Τα πρωτόκολλα είναι οι μονάδες μέσω των οποίων το δίκτυο σαν σύνολο διαμορφώνεται από τις εφαρμογές. Οι κάψουλες ανγνωρίζουν τον τύπο τους και το πρωτόκολλο στο οποίο ανήκουν. Όταν μια κάψουλα φτάσει σε έναν κόμβο, ελέγεται αν υπάρχει το αντίστοιχο πρωτόκολλο σαν σύνολο κώδικα. Αν όχι τότε αιτείται το κομμάτι που λείπει και η κάψουλα εισέρχεται σε μια περίοδο "ύπνου" μέχρι να ληφθεί το κομμάτι του κώδικα που λείπει. Τότε ο κόμβος το αποθηκεύει και ενεργοποιεί την κάψουλα Αν υπάρχουν αναπάντητες αιτήσεις για κώδικα οι αντίστοιχες κάψουλες πετιούνται. Η διαφορά στην φόρτωση και αποθήκευση κώδικα κατά απαίτηση με την προηγούμενη τεχνική είναι οτι εδώ γίνεται ανάμεσα σε γειτονικούς ενεργούς κόμβους.

Η κάψουλα περιέχει ένα πεδίο που δείχνει το αντίστοιχο πρωτόκολλο και τον τύπο της κάψουλας μέσα σ'αυτό. Το πεδίο αυτό βασίζεται σε ένα "αποτύπωμα" του κώδικα του πρωτοκόλλου, ώστε να μειωθεί ο κίνδυνος παραποίησης του πρωτοκόλλου και για να γίνεται η κατανομή των πρωτοκόλλων και των τύπων κάψουλας γρηγορα και αποκεντροποιημένα. Το υπόλοιπο τμήμα της κάψουλας έχει μια κοινή επικεφαλίδα που έχει πεδία κοινά για όλες τις κάψουλες, μια επικεφαλίδα ανάλογη του τύπου της και τα δεδομένα. Η κοινή επικεφαλίδα έχει τις διευθύνσεις αποστολής και προορισμού και πληροφορίες σχετικά με τους περιορισμούς πόρων που θα επιβληθούν από τους κόμβους.

Τα πρωτόκολλα πρέπει να εκτελεστούν σε ένα περιορισμένο περιβάλλον που θα οριοθετεί την πρόσβαση στους διαμοιραζόμενους πόρους. Κατά την επεξεργασία οι κόμβοι είναι υπεύθυνοι για την ακεραιότητα του δικτύου και για το χειρισμό πιθανών σφαλμάτων. Οι μικρές εργασίες δε χρειάζονται πιστοποίηση αλλά προστατεύονται από τους μηχανισμούς ασφάλειας για την μεταφορά κώδικα. Αντίθετα, η χρήση και η μεταβολή διαμοιραζόμενων πόρων πρέπει να πιστοποιείται. Κάθε κάψουλα έχει ένα όριο σε σχέση με τους πόρους, που λειτουργεί σαν ένα γενικευμένο πεδίο Χρόνου Ζωής (Time To Live-TTL). Αυτό το πεδίο μειώνεται από τους κόμβους καθώς καταναλώνονται οι πόροι και πετούν τις κάψουλες που το όριο τους έχει μηδενιστεί.

Τέλος, οι ρουτίνες προώθησης αναμένεται να ολοκληρωθούν τοπικά και σε μικρό χρονικό διάστημα και η κατανάλωσή τους σε εύρος και μνήμη είναι περιορισμένη.

## Η προσέγγιση Ενεργών Πακέτων και Κόμβων

Έχει γίνει σαφές μέχρι τώρα οτι τα ενεργά πακέτα μπορούν να μεταφέρουν κώδικα όταν ο κώδικας είναι απλός και περιορισμένος. Από την άλλη, οι ενεργοί κόμβοι μπορούν να παρέχουν κώδικα, που όμως είναι προκαθορισμένος γιατί πρέπει να βρίσκεται στον ενεργό κόμβο ή τουλάχιστον σε κάποιον από τον οποίο μπορεί να μεταφερθεί. Τα δύο αυτά πλεονεκτήματα των προηγούμενων προσεγγίσεων εδώ υπάρχουν σε ένα σ΄συτημα. Συνήθως τέτοις αρχιτεκτονικές επιτρέπουν στους χρήστες να επιλέξουν τη μία ή την άλλη ανάλογα με τη φύση της εφαρμογής τους.

**SwitchWare Architecture at University of Pennsylvania.** Χρησιμοποιεί μια αρχιτεκτονική επιπέδων προκειμένου να παρέχει μια ποικιλία διαφορετικών συμβιβασμών σε ευελιξία, ασφάλεια, απόδοση και χρησιμότητα. Τα τρία επίπεδα είναι τα Ενεργά Πακέτα (Active Packets), τα Switchlets και η Δομή των Ενεργών Δρομολογητών (Active Router Infrastructure). Το πρώτο επίπεδο υλοποιεί αυτό που έχουμε ονομάσει προσέγγιση των ενεργών πακέτων και το δέυτερο των ενεργών κόμβων.

Στην SwitchWare αρχιτεκτονική τα ενεργά πακέτα μεταφέρουν προγράμματα που αποτελούνται από κώδικα και δεδομένα και αντικαθιστούν την επικεφαλίδα και το σώμα ενός συμβατικού πακέτου. Η γλώσσα προγραμματισμού είναι η PLAN (Programming Language for Active Networks). Είναι μια αδύναμη γλώσσα. Επιτρέπει περιορισμένες λειτουργίες χωρίς πιστοποίηση, εκτελεί όμως και εξουσιοδοτημένες ενέργειες όταν απαιτείται. Τα προγράμματα σε PLAN είναι ασφαλή επειδή έχουν πολύ περιορισμένες πράξεις. Για να αντισταθμιστεί αυτός ο περιορισμός, τα προγράμματα καλούν ρουτίνες που ονομάζονται Switchlets, που μπορούν να πιστοποιηθούν ή να χρησιμοποιήσουν άλλους πιο ισχυρούς μηχανισμούς για να παρέχουν ασφάλεια κατά περίπτωση. Ένα πρόγραμμα PLAN αποτελείται από κώδικα, μια ένδειξη για τη συνάρτηση που πρέπει να εκτελεστεί πρώτα αμέσως μόλις φτάσει το πρόγραμμα σε ένα δρομολογητή και κάποια δεδομένα που σχηματίζουν τις παραμέτρους για τη συγκκριμένη συνάρτηση.

Τα Switchlets σχηματίζουν το μεσαίο στρώμα της SwitchWare αρχιτεκτονικής. Τα ενεργά πακέτα σκόπιμα έχουν περιορισμένη δύναμη για χάρη της ταχύτητας, αλλά σε συνδυασμό με τα Switchlets μπορούν να υλοποιήσουν αυθαίρετα πρωτόκολλα και λειτουργικότητα. Τα Switchlets μπορούν να φορτωθούν δυναμικά κατά μήκος του δικτύου αλλά εκτελούνται εξ'ολοκλήρου σε συγκεκριμένο δρομολογητή. Είναι επομένως μέρος της βασικής λειτουργικότητας ή δυναμικές επεκτάσεις παρά μεταφερόμενος κώδικας. Στην παρούσα υλοποίηση, τα Switchlets είναι γραμμένα σε Caml. Μπορούν να δεχθούν πιο αυστηρούς ελέγχους ασφάλειας από τα ενεργά πακέτα, ελέγχονται στατικά μόλις φτάσουν σε ένα δρομολογητή και μπορούν να περιέχουν και κρυπτογραφημένες υπογραφές. Μπορούν ακόμη να έχουν πρόσβαση σε λειτουργίες του δρομολογητή που τα ενεργά πακέτα αδυνατούν και επομένως να δημιουργήσουν ή να αλλάξουν την κατάστασή του ή ακόμη να έχουν άμεση πρόσβαση στις διεπαφές του δρομολογητή με το δίκτυο.

Η δομή του ενεργού δρομολογητή είναι η σταθερή βάση πάνω στην οποία χτίζονται τα ενεργά πακέτα και τα τα switchlets. Η ασφάλεια της αρχιτεκτονικής σαν σύνολο παρέχεται σ'αυτό το επίπεδο. Κάτω από το επίπεδο αυτό βρίσκεται η αρχιτεκτονική SANE που παρέχει ένα ελάχιστο σύνολο υποθέσεων εμπιστοσύνης,

την ικανότητα της ασφαλούς εκκίνησης του υπόλοιπου συστήματος και πιστοποίηση και υπηρεσίες ονοματολογίας για κώδικες που φορτώνονται.

Το σημείο κλειδί της SwitchWare αρχιτεκτονικής είναι τα επίπεδα και ο διαμοιρασμός της λειτουργικότητας σ'αυτά ανάλογα με τους συμβιβασμούς σε ευελιξία και ασφάλεια που απαιτούνται για κάθε επίπεδο. Τα υψηλότερα επίπεδα του συστήματος παρέχουν πιο περιορισμένη λειτουργικότητα με αντάλλαγμα τη μεγαλύτερη ασφάλεια και την πολύ καλή απόδοση. Τα χαμηλότερα στρώματα παρέχουν αυθαίρετη λειτουργικότητα, αλλά λόγω των μεγαλύτερων θεμαέτων ασφαλείας δεν είναι πολύ γρήγορα. Συνολικά υπάρχει ικανοποιητική αντιστάθμιση ανάμεσα στην ασφάλεια, την ευελιξία και την απόδοση.

**Η Netscript Αρχιτεκτονική.** Το πρόγραμμα Netscript παρέχει μια αρχιτεκτονική για προγραμματισμό δικτύων, μια αρχιτεκτονική μιας δυναμικά προγραμματιζόμενης συσκευής/κόμβου δικτύου και μια γλώσσα που ονομάζεται Netscript για τη δημιουργία λογισμικού δικτύου για ένα προγραμματιζόμενο δίκτυο. Η Netscript χρησιμοποιεί "εξουσιοδοτημένους πράκτορες" για τον προγραμματισμό και τον έλεγχο των ενδιάμεσων συσκευών/κόμβων του δικτύου.

Η Netscript αντιμετωπίζει το δίκτυο σαν μια συλλογή από Εικονικές Μηχανές Δικτύου (Virtual Network Engines,VNEs) που συνδέονται με Εικονικούς Συνδέσμους (Virtual Links,VLs). Οι VNEs μπορούν να προγραμματιστούν από Netscript πράκτορες για να επεξεργαστούν τις ροές των πακέτων και να μεταδώσουν αυτές τις ροές μέσω VLs σε άλλες VNEs. Η συλλογή των VNEs και VLs ορίζουν ένα Εικονικό Δίκτυο Netscript (Netscript Virtual Network , NVN). Η Netscript παρέχει μια γλώσσα για να προγραμματιστεί το NVN. Ένας φυσικός κόμβος μπορεί να εκτελεί πολλές VNEs και ένας VL μπορεί να αντιστοιχεί σε μια συλλογή φυσικών συνδέσμων και κόμβων που διασυνδέουν VNEs. Ένας VL μπορεί επίσης να διασυνδέει οποιοδήποτε αριθμό VNEs για να χειριστεί τους συνδέσμους εκπομπής.

Το επίπεδο Υπηρεσιών Πρακτόρων (Agent Services layer) παρέχει ένα multi-threaded περιβάλλον εκτέλεσης που υποστηρίζει την αποστολή, εκτέλεση και έλεγχο των προγραμμάτων-πρακτόρων. Επίσης υποστηρίζει υπηρεσίες επικοινωνίας μέσω μηνυμάτων μεταξύ των πρακτόρων. Το τμήμα των Υπηρεσιών Διασύνδεσης (Connectivity Services module) είναι υπεύθυνο για την αλληλεπίδραση με το υποκείμενο φυσικό περιβάλλον για την απόδοση και διατήρηση VLs στις γειτονικές VNEs. Παρέχει μια βιβλιοθήκη κανόνων που χρησιμοποιούνται από τα προγράμματα Netscript για την απόδοση των VL πόρων και τον χρονοπρογραμματισμό και τη μετάδοση των πακέτων πάνω στους VLs. Τα πακέτα περιέχουν μία ελάχιστη Netscript επικεφαλίδα που καθορίζει τη ροή στην οποία ανήκουν. Όταν φτάσει ένα πακέτο σε μία VNE, αυτή η επικεφαλίδα χρησιμοποιείται από το περιβάλλον εκτέλεσης για να περαστεί στα αντίστοιχα προγράμματα που επεξεργάζονται τη συγκεκριμένη ροή. Τα ενεργά πακέτα είναι τα Netscript πακέτα και οι ενεργοί κόμβοι είναι οι VNEs. Οι υπηρεσίες επικοινωνίας που παρέχονται από την VNE είναι τοπικές και επιτρέπουν την αλληλεπίδραση μόνο με γειτονικές VNEs.

Η Netscript είναι μια δυναμική dataflow γλώσσα που σχεδιάστηκε ειδικά για εργασίες που βασίζονται σε επικοινωνία. Μπορεί να επιδράσει σε ροές πακέτων. Βασίζεται σε απλές αρχές αντικειμενοστραφούς προγραμματισμού, έτσι ώστε οι χρήστες να μπορούν να παρακάμψουν τους εξ'ορισμού τελεστές με δικές τους προσαρμοσμένες εκδόσεις. Ένα Netscript πρόγραμμα αποτελείται από ένα σύνολο επικοινωνιακών γραμμών, οι οποίες επικοινωνούν μέσω ροών μηνυμάτων που συνδέουν τις εισόδους με τις εξόδους των προγραμμάτων που εκτελούνται. Τα προγράμματα που επικοινωνούν μπορεί να είναι γεωγραφικά κατανεμημένα. Η

Netscript παρέχει παγκόσμια αφηρημένη εικόνα μιας προγραμματιζόμενης συσκευής δικτύου.

Η κύρια διαφορά ανάμεσα στη Netscript και τις άλλες αρχιτεκτονικές είναι η έμφαση στον προγραμματισμό των δικτύων. Η υπόθεση εδώ είναι οτι μια μοναδική γλώσσα βασισμένη στο σωστό μοντέλο, μπορεί να απλοποιήσει σημαντικά την υλοποίηση πρωτοκόλλων και να επιτρέψει τον πειραματισμό με τα κατάλληλα χαρακτηριστικά προγραμματισμού. Μία άλλη διαφορά είναι οτι η Netscript αντιμετωπίζει το δίκτυο σαν ένα αφηρημένο προγραμματιστικό μοντέλο στο σύνολό του και όχι σαν μία ετερογενή συλλογή προγραμματιζόμενων ενδιάμεσων ή τελικών κόμβων.

## Συγκριτικά

Η προσέγγιση των ενεργών πακέτων μειονεκτεί σε θέματα απόδοσης γιατί προκύπτουν πολύ μεγάλες απαιτήσεις σε ασφάλεια. Σε μια προσπάθεια να μειωθεί το φορτίο της ασφάλειας και να βελτιωθεί η απόδοση, κάποιοι ερευνητές αποφάσισαν να μειώσουν τη λειτουργικότητα των προγραμμάτων που μεταφέρονται από τα ενεργά πακέτα, οδηγώντας σε αρχιτεκτονικές με μειωμένες δυνατότητες. Η M0 είναι η μόνη αρχιτεκτονική που μπορεί να παρέχει υψηλή λειτουργικότητα λόγω της τεχνικής αποθήκευσης.

Η προσέγγιση των ενεργών κόμβων έχει καλή απόδοση γιατί τα θέματα ασφάλειας είναι πολύ μικρότερα από προηγουμένως. Ωστόσο, η ευελιξία είναι περιορισμένη. Σε μια προσπάθεια αύξησης της ευελιξίας, οι αρχιτεκτονικές DAN και ANTS υιοθέτησαν ένα σχήμα όπου ο κώδικας φορτώνεται κατά απαίτηση και αποθηκεύετια για μελλοντική χρήση. Με τον τρόπο αυτό αυτές οι δύο αρχιτεκτονικές μπορούν εύκολα να αναπτύξουν κάθε νεό πρωτόκολλο. Βέβεια, η μεταφορά του κώδικα κατά απαίτηση προκαλεί κάποια καθυστέρηση που μειώνει τη συνολική απόδοση.

Ο συνδυασμός των δύο προσεγγίσεων φαίνεται να είναι αρκετά αποτελεσματικός. Η αρχιτεκτινική SwitchWare υλοποιεί αυτή την ιδέα με τη χρήση μιας αρχιτεκτονικής επιπέδων και καταφέρνει να προσφέρει μια ποικιλία διαφορετικών συμβιβασμών μεταξύ ευελιξίας, προστασίας και ασφάλειας, απόδοσης και χρησιμότητας. Τέλος, η Netscript αρχιτεκτονική προτείνει μια νέα άποψη, όπου το δίκτυο σαν σύνολο αντιμετωπίζεται σαν μία μοναδική προγραμματιζόμενη αφηρημένη οντότητα.

## Επίλογος

Είδαμε διάφορες εφαρμογές όπου τα ενεργά δίκτυα μπορούν να προσφέρουν μεγάλη βοήθεια. Η διαχείρηση του δικτύου, ο έλεγχος της συμφόρησης, αξιόπιστο και αποτελεσματικό multicasting και η αποθήκευση είναι μερικές από αυτές. Είδαμε κάποιες αρχιτεκτονικές που υλοποιούν την έννοια των ενεργών δικτύων : Network Management by Delegation, Darwin Project, Secure Active Network Environment, SafetyNet, PLAN, Smart Packets, Active IP Option, M0, Distributed Code Caching for Active Networks, ANTS, SwitchWare, Netscript. Τέλος, είδαμε το θέμα της ασφάλειας σε μια τέτοια ευέλικτη δομή, όπως και θέματα προγραμματισμού που προκύπτουν από την ανάγκη για ασφάλεια χωρίς την μείωση της απόδοσης.

Τα Ενεργά-Προγραμματιζόμενα δίκτυα είναι σίγουρα ένα μεγάλο βήμα στη σχεδίαση των δικτύων. Φαίνεται να παρέχουν λύσεις σε πολλά από τα προβλήματα των σημερινών "παθητικών" δικτύων και έχουν ένα πολύ ευρύ πεδίο εφαρμογών. Ωστόσο, οι υλοποιήσεις που έχουν γίνει σε διάφορα ερευνητικά προγράμματα δεν έχουν ακόμη δοκιμαστεί σε ευρείας κλίμακας δίκτυα. Για να υπάρξουν πειστικά αποτελέσματα θα έπρεπε να δημιουργηθέι ένα Abone (ανάλογο του Mbone που αναπτύχθηκε για το multicasting) και να δοκιμαστούν οι διάφορες τεχνολογίες των ενεργών και προγραμματιζόμενων δικτύων σε πραγματικές συνθήκες. Τέλος υπάρχουν δύο αντισταθμίσεις στα ενεργά δίκτυα : μία ανάμεσα στα επίπεδα ασφαλείας και την απόδοση και μία ανάμεσα στην ευελιξία/χρησιμότητα και την πολυπλοκότητα. Υπάρχουν πολλά ακόμη που μπορούν να γίνουν σε θέματα ασφάλειας και προγραμματισμού ώστε να βρεθούν οι βέλτιστες τομές.

Άσχετα από τις πεποιθήσεις και τις αμφιβολίες του καθένα, η έρευνα για τα ενεργά δίκτυα βρίσκεται ήδη καθ'οδόν. Η επιτυχία της θα σημαίνει οτι οι λειτουργίες των ενδιάμεσων κόμβων θα προγραμματίζονται και θα αναπτύσσονται με απλές, ανποιχτές και ταχύτατες διαδικασίες, χωρίς να απαιτούνται τυποποιήσεις από επιτροπές ή κατασκευαστές. Θα σημαίνει ακόμη, την αυτόματη αναβάθμιση των πρωτοκόλλων δικτύου. Συλλογιζόμενοι αυτές τις δύο προοπτικές, φαίνεται οτι οι έρευνες αξίζουν την προσπάθεια και οτι πλέον το ερώτημα δεν είναι αν τα δίκτυα θα έπρεπε να είναι προγραμματιζόμενα, αλλά ποιος είναι ο πιο αποτελεσματικός τρόπος για να γίνει αυτό.

## Αναφορές


1. **Programmable Network Framework.**
   http://www.comet.columbia.edu/distributed/lectures/lecture5
2. **WIRED : Archive, Oct 1997.**
   Ενεργοί Κόμβοι και Δίκτυα.
3. **Tradeoffs In Active Networks.**
   Seth Proctor {stp@cs.brown.edu }, Department of Computer Science , Brown University , Undergraduate Honors Thesis , May 6 1999.
   Συνολική θεώρηση των Ενεργών Δικτύων,πλεονεκτήματα και προβλήματα, PANTS.
4. **Towards Active Programmable Networks.**
   Yechiam Yemini & Sushil Da Silva, DCC Lab, Columbia University.
   Ενεργά Δίκτυα, Δίκτυα Προγραμματιζόμενα στη στιγμή, NetScript.
   http://www.cs.columbia.edu/dcc/act…bs/dsom96-slides/dsom96-notes.html
5. **NetScript : A Language and Environment for Programmable Networks.**
   Εισαγωγή και σύντομη περιγραφή της αρχιτεκτονικής NetScript.
   http://www.cs.columbia.edu/dcc/active/
   http://www.cs.columbia.edu/dcc/netscript
4. **Towards Programmable Networks.**
   Yechiam Yemini & Sushil Da Silva, Department of Computer Science, Columbia University, April 15 1996.
   NetScript Project : Η αρχιτεκτονική, η γλώσσα προγραμματισμού.
5. **NetScript : A Language-Based Active Network Architecture.**
   Σύντομη περιγραφή του ερευνητικού προγράμματος.
   http://www.darpa.mil/ito/psum1998/F3 15-0.html
6. **The NetScript Project website : http://www.cs.columbia.edu/dcc/netscript**

7. **SwitchWare : Towards a 21st Century Network Infrastracture.**
J.M.Smith , D.J.Fabert , C.A.Gunter , S.M.Nettles CIS Deprtment, University of Penssylvania
Mark E.Segal , W.D.Sincodkie Bellcore
D.C.Feldmeier , D.Scott Alexander MUSIC Semiconductors Inc.
Η αρχιτεκτονική SwitchWare , εφαρμογές και θέματα ασφάλειας..
7. **The SwitchWare Active Network Architecture.**
D.Scott Alexander, William A.Arbaugh , Michael W.Hicks, Pankaj Kakkar , Angelos D.Keromytis , Jonathan T.Moore , Carl A.Gunter , Scott M.Nettles and Jonathan M.Smith.
University of Pennsylvania , July 7 1998.
Περιγραφή , γλώσσα προγραμματισμού , Ενεργά Πακέτα , PLAN , SANE.
8. **The SwitchWare Project website : http://www.cis.upenn.edu/~switchware/**

9. **PLAN : A Packet Language for Active Networks.**
Michael Hicks , Pankaj Kakkar , Jonathan T.Moore , Carl A.Gunter and Scott M.Nettles.
Department of Computer and Information Science, University of Penssylvania.
Εισαγωγή και περιγραφή της γλώσσας PLAN , απαιτήσεις και σχεδίαση.

10. **From Internet to ActiveNet.**
http://www.tns.lcs.mit.edu/publications/rfc96/
11. **From Internet to ActiveNet , 1998 Project Summary , MIT - DARPA ITO Sponsored Research.**
http://www.darpa.mil/ito/psum1998/E242-0.html
12. **The ActiveNet Project website :** http://www.sds.lcs.mit.edu/activeware/

13. **Active Networks : Applications , Security, Safety and Architectures.**
**Konstantinos Psounis , Stanford University.**
Παρουσίαση διαφόρων αρχιτεκτονικών για Ενεργά και Προγραμματιζόμενα Δίκτυα , Εφαρμογές, Προβλήματα και Αντιπαράθεση.
http://happy.comsoc.org/pubs/surveys/1q99issue/psounis.html
14. **Active Networks : Programmed on the Fly** by **Williams S.Marcus and Mark E.Segal**
http://www.telcordia.com/newsroom/…change/winter1999/w99feature1.html
15. **Towards an Active Network Architecture.**
David L.Tennenhouse and David J.Wetherall , Laboratory for Computer Science, MIT
http://www.sce.carleton.ca/netmanage/activeNetworks/mmcn96.html
16. **Active Nodes and Networks** by **Kenji Morrow.**
Ενεργοί Κόμβοι , Κάψουλες
17. **Smart Packets Project website :**
http://www.net-tech.bbn.com/smtpkts/smtpkts-index.html
18. **Report to the EPSRC on th First UK Programmable Networks and Telecommunications Workshop.**
Ian Wakeman , Morris Sloman , Doug Shepherd , David Duce , Deborah Miller and Rob Cole.
http://www.cogs.susx.ac.uk/projects/safetynet/prognet/epsrc.html
Στόχοι , Τεχνολογίες , Προβλήματα και πιθανές Εφαρμογές