ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΣΗΜΕΙΩΣΕΙΣ ΜΑΘΗΜΑΤΟΣ

«ΠΡΟΣΤΑΣΙΑ ΚΑΙ ΑΣΦΑΛΕΙΑ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ»

Κ. Βασιλάκης

Ακαδημαϊκό έτος 2004-2005

Περιεχόμενα

ΠΡΟΣΤΑΣΙΑ ΚΑΙ ΑΣΦΑΛΕΙΑ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

1 Βασικοί ορισμοί

Με τον όρο «ασφάλεια πληροφοριακών συστημάτων» εννοούμε την προστασία *πόρων* (δεδομένων και προγραμμάτων) από συμπτωματική ή κακόβουλη τροποποίηση, καταστροφή ή διαρροή.

Για την πληρέστερη παρουσίαση του αντικειμένου θα παραθέσουμε στη συνέχεια μερικούς ορισμούς βασικών εννοιών που αφορούν το αντικείμενο.

Έννοια	***Ορισμός***
Αγαθό	Ο όρος αυτός περιλαμβάνει οτιδήποτε χρήζει προστασίας, είτε πρόκειται για υλικό (υπολογιστής, καλώδιο επικοινωνίας) είτε άυλο (π.χ. δεδομένα, υπηρεσία).
Ιδιοκτήτης	Το φυσικό ή νομικό πρόσωπο που κατέχει το αγαθό
Εξουσιοδότηση	Η παροχή από τον ιδιοκτήτη δικαιώματος χρήσης πάνω σε ένα συγκεκριμένο αγαθό. Η εξουσιοδότηση μπορεί να παρέχεται είτε σε κάποιο φυσικό πρόσωπο είτε σε κάποια διαδικασία (π.χ. η διαδικασία τήρησης εφεδρικών αντιγράφων)
Χρήστης	Ονομάζεται το φυσικό ή νομικό πρόσωπο ή η διαδικασία που χρησιμοποιεί ένα συγκεκριμένο αγαθό. Ένας χρήστης μπορεί να είναι εξουσιοδοτημένος, να χρησιμοποιεί δηλαδή το αγαθό κατόπιν εξουσιοδότησης του ιδιοκτήτη, ή όχι.
Αξία	Πρόκειται για ένα μέτρο έκφρασης της σπουδαιότητας του αγαθού. Η αξία μπορεί να εκφράζεται ως οικονομικό μέγεθος ή με οποιοδήποτε άλλο πρόσφορο τρόπο.
Ζημιά	Η υποβάθμιση της αξίας ενός αγαθού.
Επίπτωση	Οι συνέπειες που μπορεί να έχει μία ζημιά σε ένα αγαθό. Οι επιπτώσεις μπορεί να είναι οικονομικές ή και άλλης φύσεως π.χ. αρνητική δημοσιότητα.
Κίνδυνος	Η πιθανότητα να υποστεί ζημιά κάποιο αγαθό
Παραβίαση	Ένα συμβάν κατά το οποίο κάποιο αγαθό υπόκειται ζημιά
Αδυναμία	Ένα χαρακτηριστικό το συστήματος που είναι δυνατόν να επιτρέψει την εμφάνιση κάποιας παραβίασης
Απειλή	Ένας παράγοντας που μπορεί να προξενήσει ζημιά σε ένα αγαθό. Μία *απειλή* μπορεί να αξιοποιήσει μία *αδυναμία* οδηγώντας σε κάποια *παραβίαση*.
Μέσο προστασίας	Οι ενέργειες που γίνονται και οι μηχανισμοί που χρησιμοποιούνται από τον ιδιοκτήτη προκειμένου να περιορισθούν οι κίνδυνοι για τα αγαθά.

Έννοια	*Ορισμός*
Πρόληψη	Η διαδικασία εφαρμογής μέσων προστασίας με στόχο την παρεμπόδιση της εμφάνισης παραβιάσεων
Ανίχνευση	Ο εντοπισμός παραβιάσεων και των επιπτώσεών τους
Επανόρθωση	Η αποκατάσταση των επιπτώσεων μιας παραβίασης
Κόστος μέτρου	Το αντίκτυπο που έχει η χρήση ενός μέσου προστασίας. Μπορεί να είναι οικονομικό, συστημικό (π.χ. υποβάθμιση της απόδοσης του συστήματος), ψυχολογικό (π.χ. δυσαρέσκεια από τη χρήση του μέτρου) κ.λπ.

Από τους παραπάνω ορισμούς μπορούμε να δούμε ότι για να προστατεύσουμε κάποια **αγαθά** που έχουν συγκεκριμένη **αξία** και να μην υποστούμε τις **επιπτώσεις** που θα έχει μία **παραβίαση** που θα τα αφορά, θα πρέπει να χρησιμοποιήσουμε κάποια **μέσα προστασίας** που έχουν κάποιο **κόστος**. Είναι σαφές ότι το κόστος που συνεπάγονται μέσα προστασίας δεν θα πρέπει να είναι δυσανάλογο της αξίας των αγαθών ή των επιπτώσεων που σχετίζονται με αυτά. Είναι έτσι πιθανόν να **επιλέξει** ο ιδιοκτήτης ενός πληροφοριακού συστήματος να μην εφαρμόσει κάποια μέσα προστασίας, διότι εκτιμά ότι το κόστος τους είναι υπερβολικό, σε σχέση με την αξία τους και τους κινδύνους που διατρέχουν. Στις περιπτώσεις αυτές προφανώς ο βαθμός ασφάλειας του συστήματος είναι ελαττωμένος. Δεδομένου, πάντως, ότι ο ιδιοκτήτης έχει στη διάθεσή του τα στοιχεία για τα αγαθά, τους κινδύνους και τα μέσα προστασίας και αποφασίζει ποια θα εφαρμόσει και ποια όχι, είναι προτιμότερο να μιλάμε για *διαχείριση κινδύνων* παρά για *ασφάλεια*.

Μια εναλλακτική (ή συμπληρωματική) προσέγγιση στην απόφαση για την εφαρμογή μέσων προστασίας είναι η αποτίμηση του οφέλους που θα αποκομίσει κάποιος που θα επιτύχει την παραβίαση του πληροφοριακού συστήματος: προκειμένου να επιτύχει την **παραβίαση**, ο εισβολέας θα επωμισθεί κάποιο **κόστος**, το οποίο αν είναι μεγαλύτερο από το όφελος που θα απολάβει θα καταστήσει ασύμφορο το όλο εγχείρημα. Έτσι, από τα μέσα ασφάλειας μπορεί να επιλεχθούν προς εφαρμογή αυτά που αυξάνουν το κόστος παραβίασης του συστήματος. Η προσέγγιση αυτή λαμβάνει υπόψη και το είδος του επίδοξου εισβολέα: ένας περιστασιακός hacker που απλά θέλει να νοιώσει την ικανοποίηση ότι «έσπασε» ένα ακόμη σύστημα, πιθανώς να εγκαταλείψει μετά τις πρώτες δυσκολίες, οπότε οι βασικές τεχνικές προστασίας είναι γενικώς επαρκείς. Αντίθετα, αν η επίθεση προέρχεται από ανταγωνιστικές εταιρίες ή μυστικές υπηρεσίες (κάτι που προφανώς δεν θα συμβεί στον μέσο οικιακό υπολογιστή), υπάρχει αυξημένη πιθανότητα οι εισβολείς να επιμείνουν λίγο παραπάνω, άρα χρειάζονται πιο αυστηρά και αποτελεσματικά μέτρα προστασίας.

Έχοντας στη διάθεσή μας τους ορισμούς που παραθέσαμε ανωτέρω, μπορούμε πλέον να δώσουμε τις βασικές διαστάσεις της ασφάλειας των πληροφοριακών συστημάτων:

- *Εμπιστευτικότητα*: οι πληροφορίες είναι προσπελάσιμες μόνο από εξουσιοδοτημένους χρήστες
- *Ακεραιότητα*: τα δεδομένα και τα προγράμματα τροποποιούνται και καταστρέφονται μόνο με καλά καθορισμένους τρόπους και με κατάλληλη εξουσιοδότηση

- *Διαθεσιμότητα*: Οι εξουσιοδοτημένοι χρήστες θα μπορούν να χρησιμοποιήσουν δεδομένα, προγράμματα και υπηρεσίες όταν το επιθυμήσουν
- *Αυθεντικότητα*: εξασφάλιση ότι τα δεδομένα είναι απαλλαγμένα ατελειών και ανακριβειών κατά τις εξουσιοδοτημένες τροποποιήσεις
- *Εγκυρότητα*: εξασφάλιση ότι τα δεδομένα είναι ακριβή και πλήρη

2 Συνηθισμένες απειλές στην ασφάλεια

2.1 Αποκάλυψη συνθηματικών

Τα συνθηματικά είναι ένας από τους πιο διαδεδομένους τρόπους για να «αναγνωρίζεται» ένας χρήστης από το σύστημα. Παρά την ευρεία τους διάδοση και πολύχρονη χρήση ωστόσο υπάρχει μία σειρά από ζητήματα που σχετίζεται με τη χρήση και αποτελεσματικότητά τους. Τα συνθηματικά μπορεί να αποκαλυφθούν είτε μέσω εξαντλητικής αναζήτησης (δοκιμή όλων των δυνατών συνθηματικών), είτε με χρήση λιστών με συχνά χρησιμοποιούμενα συνθηματικά είτε με αξιοποίηση «προκαθορισμένων» συνθηματικών, καθώς και με πληθώρα άλλων μεθόδων. Οι επιθέσεις που αφορούν αποκάλυψη συνθηματικών διευκολύνονται από τους γρήγορους υπολογιστές (δυνατότητα εξέτασης περισσότερων συνθηματικών στη μονάδα του χρόνου) και τα γρήγορα δίκτυα.

2.2 Πλοήγηση

Ένας «νόμιμος» χρήστης ενός συστήματος (ή ένας εισβολέας που έχει αποκτήσει περιορισμένη πρόσβαση) ψάχνει στο σύστημα για να βρει πληροφορίες που θα του δώσουν περισσότερα προνόμια. Η αναζήτηση μπορεί να γίνεται σε μπλοκ δίσκου, σε σελίδες μνήμης, σε απροστάτευτα αρχεία κ.λπ.

2.3 Αντιποίηση ή μεταμφίεση

Στην περίπτωση αυτή ο χρήστης πιστεύει ότι αλληλεπιδρά με μία οντότητα (πρόγραμμα, υπηρεσία, χρήστη) ενώ στην πραγματικότητα αλληλεπιδρά με κάποιον άλλο που έχει έντεχνα μεταμφιεσθεί. Από τις πρώτες (ιδιαίτερα επιτυχημένες) προσπάθειες αντιποίησης ήταν η συγγραφή προγραμμάτων που προσομοίωναν τη λειτουργικότητα του προγράμματος σύνδεσης (login) του συστήματος. Οι χρήστες ανυποψίαστοι εισήγαγαν τα διαπιστευτήρια σύνδεσής τους στο πρόγραμμα αυτό, το οποίο αντί να τους παρέχει σύνδεση με το σύστημα τα καταχωρούσε για να τα βρει ο δόλιος συγγραφέας του. Στο περιβάλλον του διαδικτύου είναι δυνατόν να αντιποιηθούν ιστοχώροι, π.χ. είναι δυνατόν να δημιουργηθεί ο ιστοχώρος **www.amason.com** ο οποίος θα μοιάζει καθ' όλα με τον «κανονικό» ιστοχώρο **www.amazon.com**, εκτός από το ότι τα στοιχεία της πιστωτικής κάρτας που θα εισάγει ένας «πελάτης» δεν θα χρησιμοποιηθούν για την πληρωμή των βιβλίων. Τέλος, σε περιβάλλον δικτύου η αντιποίηση εμφανίζεται με την αποστολή δικτυακών πακέτων που φαίνονται να προέρχονται από διαφορετικούς υπολογιστές από αυτούς όπου πραγματικά εκπέμφθηκαν.

2.4 Δούρειοι ίπποι

Πρόκειται για προγράμματα που «διαφημίζονται» ότι προσφέρουν κάποια χρήσιμη υπηρεσία ή δυνατότητα στους χρήστες τους, π.χ. ένας κειμενογράφος για το Unix που

να είναι καλύτερος από τον vi[1]. Ο «δούρειος ίππος» συνήθως εκπληρώνει τις υποσχέσεις του, *αλλά* εμπεριέχει και κάποια λειτουργικότητα που δεν αναφέρεται στη διαφήμιση και που συνήθως είναι ανεπιθύμητη. Έτσι, ο ως άνω κειμενογράφος θα μπορούσε να καθιστά τα αρχεία μας εγγράψιμα από τον συγγραφέα του ή να του στέλνει συνθηματικά και διευθύνσεις που έχουμε αποθηκευμένα σε προκαθορισμένες θέσεις. Οι δούρειοι ίπποι βασίζονται στην ιδιότητα ότι μια και το πρόγραμμα εκτελείται από τον χρήστη Α, θα έχει πρόσβαση σε όλα τα δεδομένα και τις υπηρεσίες που ο Α είναι εξουσιοδοτημένος να προσπελάσει, αλλά φυσικά οι προσβάσεις αυτές δεν θα γίνονται πια για νομότυπους σκοπούς. Τα αποτελέσματα των προσπελάσεων μπορούν να σταλούν στον συγγραφέα του δούρειου ίππου ή να του ανοίξουν τον δρόμο για απ' ευθείας πρόσβαση.

Κάθε πρόγραμμα είναι ύποπτο για «δούρειος ίππος» αν δεν εμπιστευόμαστε τόσο τον συγγραφέα του όσο και το δίκτυο διανομής μέσω του οποίου το πρόγραμμα έφτασε σε εμάς.

2.5 Αξιοποίηση προγραμματιστικών σφαλμάτων

Σε πολλές περιπτώσεις, προγραμματιστικά σφάλματα σε εφαρμογές ή σε λειτουργικά συστήματα επιτρέπουν σε επίδοξους εισβολείς να υποβαθμίσουν την ασφάλεια των υπολογιστικών συστημάτων. Με τον όρο *προγραμματιστικά σφάλματα*

2.6 Παραπόρτια (trapdoors)

Πρόκειται για τροποποιήσεις συστημάτων που παρέχουν πρόσβαση στα συστήματα, χωρίς ιδιαίτερες διατυπώσεις. Μολονότι συνήθως εγκαθίστανται από τους εισβολείς μετά από μία επιτυχημένη επίθεση και για μελλοντική χρήση, δεν είναι σπάνια η περίπτωση να εγκατασταθούν από τους κατασκευαστές ως «δίοδοι ταχείας πρόσβασης» για την περίπτωση που «κάτι πάει στραβά». Διάσημα προγράμματα αυτής της κατηγορίας είναι οι παρεφθαρμένες εκδόσεις του login που επιτρέπουν είσοδο με δικαιώματα υπερχρήστη σε συγκεκριμένα usernames και το Back Orifice σε περιβάλλον PC που δίνει σε απομακρυσμένους χρήστες δικαιώματα διαχείρισης στον δικό μας υπολογιστή.

2.7 Ιοί

Πρόκειται για προγράμματα που «μολύνουν» άλλα προγράμματα, ενσωματώνοντας σ' αυτά αντίγραφα του εαυτού τους -πιθανώς εξελιγμένα. Οι βλάβες που προκαλούν κυμαίνονται από διαγραφή/αλλοίωση αρχείων, υποβάθμιση απόδοσης, κατανάλωση χώρου ή απλά ενόχληση. Οι βασικοί τρόποι διάδοσης των ιών περιλαμβάνουν τους τομείς εκκίνησης, την εκτέλεση μολυσμένων προγραμμάτων, την εκτέλεση κάποιας δικτυακής εφαρμογής, το άνοιγμα «παραλλαγμένου» συνημμένου αρχείου, τη χρήση διαμοιρασμένου πόρου κ.λπ.

2.8 Διαρροή δεδομένων

Διεργασίες που είναι εξουσιοδοτημένες να προσπελαύνουν δεδομένα τα αποκαλύπτουν σε χρήστες που δεν είναι εξουσιοδοτημένοι. Για παράδειγμα, το πρόγραμμα διακίνησης ηλεκτρονικής αλληλογραφίας *sendmail* ήταν δυνατόν να κληθεί με την ένδειξη *–bp* με αποτέλεσμα να αναφέρει τα μηνύματα των οποίων η παράδοση εκκρεμούσε κατά τη στιγμή της εκτέλεσης, αναφέροντας μάλιστα και τον

[1] Κάτι τέτοιο είναι προφανώς αδύνατον, αλλά μερικοί χρήστες είναι εύπιστοι

αποστολέα και τον παραλήπτη. Με συνεχή χρήση αυτής της εντολής ήταν δυνατόν να πληροφορηθεί κανείς ποιος επικοινωνεί με ποιον.

2.9 Συμπερασμός πληροφοριών

Ο συμπερασμός πληροφοριών αναφέρεται στη συσχέτιση φαινομενικά άσχετων δεδομένων για εξαγωγή χρήσιμων πληροφοριών. Πολλά τέτοια προβλήματα εμφανίζονται στις *στατιστικές βάσεις δεδομένων*, οι οποίες οφείλουν να δίνουν πληροφορίες για ομάδες πληθυσμού, αλλά όχι για μεμονωμένα άτομα. Με κατάλληλες ωστόσο ερωτήσεις και ελλείψει μηχανισμών ασφαλείας είναι δυνατόν να εξαχθούν ατομικές πληροφορίες: Για παράδειγμα, έστω η ερώτηση

> *«ποιο είναι το πλήθος και μέσος μισθός των ανδρών με ηλικία μικρότερη των 35 ετών»*

η οποία απαντάται με τα στοιχεία

> *(10, 2000)*

Αν η ερώτηση

> *«ποιο είναι το πλήθος και μέσος μισθός των ανδρών με ηλικία μικρότερη των 34 ετών»*

απαντάται με τα στοιχεία

> *(9,1800)*

τότε συνάγεται ότι ο μοναδικός άνδρας ηλικίας 34 ετών έχει μισθό 3800.

2.10 Πλαστογράφηση

Πρόκειται για τη μη εξουσιοδοτημένη τροποποίηση δεδομένων με αποτέλεσμα τη δημιουργία πλαστογραφημένων εκδόσεών τους. Η τροποποίηση μπορεί να γίνει είτε στα αποθηκευμένα δεδομένα (με αποτέλεσμα τη μόνιμη παραποίησή τους), είτε στα δεδομένα όταν αυτά μεταδίδονται μέσω δικτύου.

2.11 Κανάλια διαρροής

Τα κανάλια διαρροής είναι ένας μηχανισμός σύμφωνα με τον οποίο μία διεργασία που έχει δικαίωμα να προσπελάσει κάποια δεδομένα τα μεταδίδει σε μία διεργασία που δεν θα είχε κανονικά δικαίωμα να τα προσπελάσει, χρησιμοποιώντας όχι τυποποιημένους μηχανισμούς διαδιεργασιακής επικοινωνίας αλλά τεχνικές φαινομενικά άσχετες προς τη μετάδοση δεδομένων. Για παράδειγμα μία διεργασία μπορεί να αυξάνει ή να μειώνει τη μνήμη που έχει δεσμεύσει, μεταδίδοντας έτσι τα bits 1 και 0 αντίστοιχα. Μία άλλη διεργασία μπορεί παρακολουθώντας το μέγεθος της μνήμης που καταλαμβάνει η πρώτη (που συνήθως είναι αδιαβάθμητη και προσπελάσιμη σε όλους πληροφορία) να «λάβει» τα δεδομένα που «μεταδίδει» η πρώτη. Το ίδιο μπορεί να επιτευχθεί με άλλους τρόπους π.χ. κλείδωμα και ξεκλείδωμα πόρων, τον χρόνο διεκπεραίωσης μιας εργασίας, την αυξομείωση μεγέθους αρχείων, τη χρήση της ΚΜΕ κ.λπ.

2.12 Παρεμπόδιση παροχής υπηρεσιών

Η υποβάθμιση της αξίας ενός υπολογιστικού συστήματος μπορεί να επέλθει χωρίς κάποια φυσική καταστροφή ή φθορά δεδομένων, αλλά επίσης και με την ανάθεση σ' αυτόν ενός ιδιαίτερα επαχθούς έργου που να εξαντλεί τους πόρους του καθιστώντας το ανίκανο να προσφέρει το έργο που του έχει ανατεθεί. Έτσι ένας εξυπηρέτης

ηλεκτρονικού ταχυδρομείου μπορεί να καταστεί «άχρηστος» αν του ανατεθεί να διακινήσει 5000 μηνύματα των 200 Mbytes έκαστο, καθώς σίγουρα θα εξαντληθεί ο αποθηκευτικός του χώρος. Επίσης ένας εξυπηρέτης WWW θα είναι επίσης «άχρηστος» αν «βομβαρδισθεί» με δυσανάλογο προς τις προδιαγραφές του αριθμό αιτήσεων. Η παρεμπόδιση παροχής υπηρεσιών συνίσταται, συνήθως, στην υποβολή πολλών αιτήσεων που η κάθε μία είναι μεμονωμένα «νομότυπη», αλλά συνδυαστικά έχουν άσχημα αποτελέσματα. Η παρεμπόδιση παροχής υπηρεσιών αποσκοπεί στη στέρηση από τους νόμιμους χρήστες της δυνατότητάς τους να εξυπηρετηθούν από το υπολογιστικό σύστημα.

2.13 Μη ηθελημένη καταστροφή

Ένας χρήστης μπορεί να πραγματοποιήσει ατυχείς ενέργειες π.χ. να διαγράψει ένα (χρήσιμο) αρχείο ή να σβήσει ένα σύνολο εγγραφών από μια βάση δεδομένων. Ως ενέργειες που υποβαθμίζουν την αξία του συστήματος τα περιστατικά αυτά πρέπει να καλύπτονται από τους μηχανισμούς ασφάλειας. Μολονότι προφανώς δεν είναι δυνατόν να στερήσουμε από τους χρήστες τα βασικά τους προνόμια για να αποτραπούν οι ατυχείς ενέργειες, θα πρέπει στο σχέδιο ασφάλειας να μεριμνούμε για μεθόδους αντιμετώπισης των περιστατικών αυτών.

Παρά την πληθώρα δυνατών επιθέσεων στην ασφάλεια και τις σημαντικές συνέπειες που μπορεί αυτές να έχουν, πολλές φορές οι επιθέσεις αυτές δεν αναφέρονται στους υπεύθυνους, στη διοίκηση ή σε κατάλληλους φορείς στο Internet. Οι λόγοι μη αναφοράς είναι κυρίως οι ακόλουθοι:

- Η αναφορά ενός προβλήματος δίνει ιδέες σε άλλους επίδοξους εισβολείς. Έτσι αν διαρρεύσει μία πληροφορία ότι «ο τάδε υπολογιστής έχει μία αδυναμία σ' αυτή την υπηρεσία», αρκετοί εισβολείς μπορεί να προσπαθήσουν να εκμεταλλευτούν το συγκεκριμένο κενό ή να εντοπίσουν και άλλα.
- Η αρνητική δημοσιότητα διώχνει πελάτες και δυσαρεστεί τους μετόχους. Για παράδειγμα, αν μία τράπεζα ανακοινώσει ότι κάποιος «έσπασε» το διαδυκτιακό σύστημα εξυπηρέτησης πελατών, οι καταθέτες της τράπεζας θα είναι πολύ διστακτικοί στο να αξιοποιήσουν την υπηρεσία αυτή, ενώ και η μετοχή στη Σοφοκλέους πιθανόν να μπει στην κόκκινη ζώνη[2].
- Πολλές φορές η σημασία ενός συμβάντος υποβαθμίζεται και δεν τίθεται στις πραγματικές της διαστάσεις, πιθανώς λόγω άγνοιας των ενδεχόμενων συνεπειών.

Η μη αναφορά των περιστατικών πάντως δίνει την ψευδαίσθηση ότι «όλα πάνε καλά» και έτσι δεν βοηθά στην δημιουργία (ή αναμόρφωση) και εφαρμογή ενός καλύτερου σχεδίου ασφάλειας.

3 Ασφάλεια σε δικτυακό περιβάλλον

Με την πρόοδο των δικτυακών τεχνολογιών και την παροχή δικτυακών υπηρεσιών, έχουν δημιουργηθεί νέα δεδομένα που πρέπει να ληφθούν υπόψη στον σχεδιασμό ασφάλειας. Για τους σκοπούς του παρόντος μαθήματος, το δίκτυο είναι ένα σύνολο

[2] Η είσοδος της μετοχής στην κόκκινη ζώνη δεν χρειάζεται απαραίτητα τη βοήθεια τέτοιων ανακοινώσεων.

διασυνδεδεμένων υπολογιστών, οι οποίοι παρέχουν υπηρεσίες και αποθηκεύουν πληροφορίες. Οι χρήστες στο περιβάλλον αυτό προσπελαύνουν υπηρεσίες και ανταλλάσσουν ή/και αποθηκεύουν πληροφορίες, και η απαίτηση από πλευράς ασφάλειας είναι να εξασφαλιστεί η εμπιστευτικότητα, η ακεραιότητα και η διαθεσιμότητα δεδομένων και υπηρεσιών. Για να είναι δυνατή η επίτευξη αυτών των στόχων, είναι απαραίτητο να εξασφαλισθούν οι κάτωθι συνιστώσες:

- *Ισχυρή διακρίβωση ταυτότητας των ενεχόμενων μερών* (χρηστών-συστημάτων), έτσι ώστε αφ' ενός το σύστημα να είναι βέβαιο για την ταυτότητα του χρήστη με το οποίο συνδιαλλάσσεται, αφ' ετέρου δε ο χρήστης να είναι βέβαιος ότι έχει συνδεθεί με το σύστημα που επιθυμεί.
- *Αξιόπιστοι μηχανισμοί ελέγχου εξουσιοδότησης/προσπέλασης*, έτσι ώστε να εξασφαλίζεται ότι τα κατάλληλα δικαιώματα έχουν αποδοθεί στους κατάλληλους χρήστες και ότι οι κανένας χρήστης δεν μπορεί να υπερβεί τα προνόμιά του.
- *Αποτελεσματικοί έλεγχοι κατάχρησης δικαιωμάτων*, προκειμένου να εντοπίζονται οι περιπτώσεις όπου δικαιώματα που έχουν παραχωρηθεί για συγκεκριμένους λόγους χρησιμοποιούνται για άλλους σκοπούς.
- *Τέλειες πολιτικές και απαράβατη εφαρμογή τους*, δηλαδή επακριβής ορισμός των δικαιοδοσιών του κάθε χρήστη και επιβολή αυτών των περιορισμών και στην πράξη.
- *Άψογα πρωτόκολλα, λειτουργικά συστήματα και εφαρμογές*, τα οποία είναι τα εργαλεία που θα επιτρέψουν την εφαρμογή των πολιτικών.
- *Κάθε χρήστης είναι ειδικός στην ασφάλεια*, έτσι ώστε να είναι σε θέση να αξιολογήσει τους κινδύνους που μπορεί να επιφέρει οποιαδήποτε ενέργειά του και να πράξει αναλόγως.

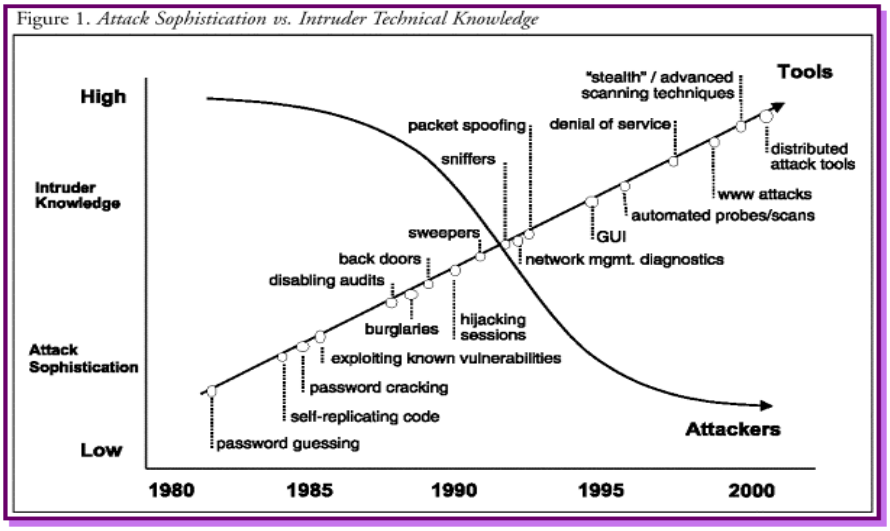
Η πραγματικότητα ωστόσο είναι αρκετά διαφορετική από την πιο πάνω «λίστα επιθυμιών» και έχει ως ακολούθως:

- Δεν εφαρμόζονται αποτελεσματικοί μέθοδοι προστασίας.
- Δεν εγκαθίστανται οι επιδιορθώσεις που παρέχονται από τους κατασκευαστές λογισμικού, προκειμένου να αντιμετωπισθούν προβλήματα ασφάλειας στα λειτουργικά συστήματα, στα πρωτόκολλα ή τις εφαρμογές.
- Η δικτυακή πρόσβαση δεν απαιτεί επαρκή πιστοποίηση, η πρόσβαση στους «εσωτερικούς» υπολογιστές (τους υπολογιστές δηλαδή ενός εταιρικού δικτύου, σε αντιδιαστολή με τους υπολογιστές εκτός αυτού) δεν παρακολουθείται και δεν ελέγχεται.
- Δεν διατίθεται προσωπικό για ζητήματα ασφάλειας. Τα ζητήματα αυτά επαφίενται στους διαχειριστές, οι οποίοι δεν είναι κατ' ανάγκην ειδικοί ή καλά καταρτισμένοι στα ζητήματα ασφάλειας, και σε κάθε περίπτωση έχουν (και) άλλα καθήκοντα να επιτελέσουν.
- Δεν εφαρμόζονται πολιτικές, δεν συντάσσεται δηλαδή μία γενική «χάρτα» δικαιωμάτων και υποχρεώσεων των χρηστών, αναφορικά με την ασφάλεια.
- Σε πολλές περιπτώσεις η πεποίθηση ότι το σύστημα είναι ασφαλές στηρίζεται στην άποψη ότι «δεν είναι γνωστό τι υπηρεσίες παρέχω, άρα δεν είναι δυνατόν να επιτεθεί κανείς σ' αυτές». Η πεποίθηση αυτή είναι απόλυτα

εσφαλμένη, καθώς οι επιθέσεις δεν γίνονται (πλέον) χειρωνακτικά από τους χρήστες αλλά με αυτοματοποιημένα εργαλεία που μπορούν να αναλύσουν σε λίγα δευτερόλεπτα όλες τις υπηρεσίες που προσφέρει ένας μη επαρκώς προστατευμένος υπολογιστής, υποδεικνύοντας έτσι τις πιθανές κερκόπορτες στην ασφάλεια του συστήματος.

Η χρονική εξέλιξη των επιθέσεων, όπως απεικονίζεται στο ακόλουθο σχήμα, έρχεται προς επίρρωση της τελευταίας παραγράφου. Από το 1990 και μετά, υπάρχει μία εντυπωσιακή μεταστροφή σε επιθέσεις με αυτοματοποιημένα εργαλεία, τα οποία είναι διαθέσιμα ευρέως στην κοινότητα του διαδικτύου. Η χρήση εργαλείων έχει δύο παρεπόμενα: πρώτον, μία επίθεση είναι ιδιαίτερα εξαντλητική, εξετάζοντας ένα ιδιαίτερα μεγάλο πλήθος μεθόδους παραβίασης της ασφάλειας, άρα και πιο αποτελεσματική σε σχέση με μία χειρονακτική επίθεση. Δεύτερον, δεν χρειάζεται να είναι πια κανείς ειδικός σε θέματα παραβίασης ασφάλειας για να ξεπεράσει τις όποιες προστασίες έχει ένα υπολογιστικό σύστημα: οποιοσδήποτε χρήστης του Internet μπορεί να εγκαταστήσει στον υπολογιστή του ένα τέτοιο εργαλείο, να διαλέξει τον στόχο του και να ξεκινήσει την επίθεση.

Figure 1. *Attack Sophistication vs. Intruder Technical Knowledge*



4 Προσεγγίσεις στην επίτευξη ασφάλειας

Προκειμένου να επιτευχθεί υψηλό επίπεδο ασφάλειας σε κάποιο σύστημα μπορούν να ακολουθηθούν οι εξής κατευθύνσεις.

- *Ορισμός ασφαλών διαδικασιών*. Η κατεύθυνση αυτή αφορά περισσότερο εξωσυστημικά ζητήματα, όπως π.χ. ποιος θα κάνει κάποια εργασία, αν η εργασία θα πρέπει να γίνεται παρουσία άνω του ενός ατόμων, ζητήματα καταγραφής κ.λπ., και δεν θα μας απασχολήσει στη συνέχεια.
- *Εφαρμογή μηχανισμών για την επιβολή μέτρων ασφάλειας*.
- *Διασφάλιση μέσω ανάλυσης-επαλήθευσης*.

Οι δύο τελευταίες κατευθύνσεις θα αναλυθούν στις επόμενες παραγράφους.

4.1 Μηχανισμοί προστασίας

Οι δύο βασικοί μηχανισμοί προστασίας στα συστήματα είναι ο *έλεγχος ταυτότητας*, προκειμένου να εξασφαλίζεται ότι η οντότητα που παρουσιάζεται με κάποια ταυτότητα όντως είναι αυτή που ισχυρίζεται, και ο *έλεγχος προσπέλασης* που

χρησιμοποιείται για να επιτρέψει σε κάποια (διακριβωμένη πια) οντότητα να προσπελάσει μόνο τα αντικείμενα και τις υπηρεσίες για τα οποία είναι εξουσιοδοτημένη.

4.1.1 Διακρίβωση ταυτότητας

Η διακρίβωση ταυτότητας είναι ένα βασικό δομικό στοιχείο της ασφάλειας συστημάτων, καθώς αποτελεί τη βάση για τους περισσότερους τύπους ελέγχου πρόσβασης και καταλογισμού ευθυνών. Το σύστημα θα πρέπει να έχει τη δυνατότητα να ταυτοποιεί τους χρήστες και να μπορεί να τους ξεχωρίζει. Επί παραδείγματι, ο έλεγχος πρόσβασης συχνά βασίζεται στην αρχή των *ελάχιστων προνομίων*, δίνοντας στους χρήστες μόνο τα δικαιώματα που τους είναι απολύτως απαραίτητα για την επιτέλεση των εργασιών τους. Ο καταλογισμός ευθυνών απαιτεί τη σύνδεση των δραστηριοτήτων σε ένα υπολογιστικό σύστημα με συγκεκριμένα άτομα, συνεπώς το σύστημα πρέπει να γνωρίζει την ταυτότητα των χρηστών.

Κατά τη διακρίβωση ταυτότητας, η οντότητα αρχικά παρουσιάζει στο σύστημα έναν ισχυρισμό περί της ταυτότητας της και ακολούθως το σύστημα εξετάζει αν αυτός ο ισχυρισμός είναι αληθής. Στη διαδικασία αυτή υπάρχουν τα εξής βήματα: η συλλογή των πληροφοριών που δίνει ο χρήστης, η ασφαλής μετάδοσή τους και ο προσδιορισμός του αν ο χρήστης που αρχικά διακριβώθηκε εξακολουθεί να είναι ο ίδιος που τώρα χρησιμοποιεί το σύστημα. Για παράδειγμα, αν ένας χρήστης συνδεθεί σε κάποιο τερματικό και στη συνέχεια το εγκαταλείψει προσωρινά, είναι δυνατόν κάποιος άλλος χρήστης να το χρησιμοποιήσει υπό την ταυτότητα του πρώτου.

Υπάρχουν τρεις βασικοί τρόποι διακρίβωσης της ταυτότητας, που μπορούν να χρησιμοποιηθούν μεμονωμένα ή συνδυαστικά:

1. να ζητάται κάτι που ο χρήστης *γνωρίζει* (ένα μυστικό, π.χ. ένα συνθηματικό, ένας προσωπικός αριθμός αναγνώρισης ή ένα κρυπτογραφικό κλειδί)
2. να ζητάται κάτι που βρίσκεται *υπό την κατοχή του χρήστη*, όπως μία έξυπνη κάρτα, μία κάρτα αυτόματων ταμειακών συναλλαγών κ.λπ.
3. να εξετάζεται κάποιο βιομετρικό χαρακτηριστικό του χρήστη, όπως π.χ. δακτυλικά αποτυπώματα, σχήμα ίριδας, τρόπος γραφής κ.τ.λ.

Μολονότι φαίνεται ότι οποιοδήποτε από αυτά τα μέσα μπορεί να παρέχει ισχυρή διακρίβωση ταυτότητας, υπάρχουν προβλήματα μ' αυτές: τα συνθηματικά μπορεί να διαρρεύσουν ή να μαντευθούν· οι έξυπνες κάρτες μπορεί να κλαπούν ή να κατασκευαστούν πλαστές· ακόμη και τα βιομετρικά συστήματα μπορούν να ξεγελασθούν. Κάθε μέθοδος έχει επίσης μειονεκτήματα για τους διαχειριστές και τους «νόμιμους» χρήστες: οι χρήστες ξεχνάνε τα συνθηματικά ή χάνουν τις έξυπνες κάρτες, και η διαχειριστική επιβάρυνση για την αντιμετώπιση αυτών των ζητημάτων μπορεί να είναι σημαντική. Επίσης, τα βιομετρικά συστήματα συναντούν προβλήματα αποδοχής από πλευράς χρηστών, έχουν υψηλό κόστος και τεχνικές δυσκολίες.

4.1.1.1 Τεχνικές όπου ζητάται κάτι που ο χρήστης γνωρίζει

Η πιο συνηθισμένη τεχνική διακρίβωσης ταυτότητας συσχετίζει κάθε ταυτότητα χρήστη με ένα συνθηματικό. Η τεχνική αυτή βασίζεται αποκλειστικά σε κάτι που ο χρήστης γνωρίζει. Υπάρχουν και άλλες τεχνικές που ζητούν κάτι που γνωρίζει ο χρήστης, όπως π.χ. ένα κρυπτογραφικό κλειδί.

Συνθηματικά

Γενικώς, τα συστήματα συνθηματικών λειτουργούν απαιτώντας από τον χρήστη να εισάγει την ταυτότητά του μαζί με ένα συνθηματικό (ή συνθηματική φράση ή προσωπικό αριθμό ταυτότητας κ.λπ.). Το σύστημα συγκρίνει το συνθηματικό με αυτό που είναι αποθηκευμένο στο *αρχείο συνθηματικών* για τον συγκεκριμένο χρήστη. Αν είναι ίδια, η ταυτότητα έχει διακριβωθεί επιτυχώς.

Η χρήση συνθηματικών έχει παράσχει ασφάλεια σε υπολογιστικά συστήματα για μεγάλο χρονικό διάστημα. Οι σχετικοί μηχανισμοί είναι ενσωματωμένοι στα λειτουργικά συστήματα και οι χρήστες, αλλά και οι διαχειριστές συστημάτων είναι εξοικειωμένοι με αυτά. Με κατάλληλη διαχείριση σε ένα ελεγχόμενο περιβάλλον μπορούν να αποτελέσουν αποτελεσματικό μηχανισμό διακρίβωσης ταυτότητας.

Από την άλλη πλευρά, η λειτουργία του όλου σχήματος βασίζεται στο ότι δεν θα διαρρεύσουν τα συνθηματικά. Δυστυχώς υπάρχουν πολλοί τρόποι με τους οποίους είναι δυνατόν να αποκαλυφθούν:

1. *«Μάντεμα» ή εύρεση συνθηματικών.* Αν οι χρήστες διαλέγουν μόνοι τους τα συνθηματικά, τείνουν να επιλέγουν κάποια εύκολα για απομνημόνευση. Αυτό συνήθως συνεπάγεται και ευκολία στο μάντεμα, καθώς οι συνήθεις επιλογές είναι ονόματα συζύγων, παιδιών, ομάδων ή κατοικίδιων, τηλέφωνα κ.λπ., τα οποία όμως είναι γνωστά και σε άλλους. Από την άλλη πλευρά, αν τα συνθηματικά δεν είναι εύκολο να απομνημονευθούν, οι χρήστες πιθανότατα θα τα σημειώσουν, ενδεχομένως σε σημεία που και άλλοι έχουν πρόσβαση. Πολλά συστήματα με προεγκατεστημένα λειτουργικά έχουν τυποποιημένα συνθηματικά για τους διαχειριστές ή άλλους χρήστες, που κανείς δεν μεριμνά να τροποποιήσει. Μολονότι οι ειδικοί συνεχώς ειδοποιούν για τα ζητήματα αυτά για πολλά χρόνια τώρα, οι χρήστες και οι διαχειριστές δεν έχουν συμμορφωθεί. Ένας εναλλακτικός τρόπος να μάθει κανείς το συνθηματικό κάποιου άλλου είναι να τον παρατηρεί κατά την ώρα που ο τελευταίος εισάγει το συνθηματικό.

2. *«Διαμοιρασμός» των συνθηματικών.* Σε πολλές περιπτώσεις οι χρήστες κοινοποιούν τα συνθηματικά τους σε τρίτους, πιθανώς σε συναδέλφους προκειμένου για διαμοιρασμό αρχείων. Σε μερικές περιπτώσεις οι χρήστες δίνουν τα συνθηματικά τους σε άτομα που δηλώνουν «διαχειριστές» ή «υπεύθυνοι ασφάλειας» ή ακόμη και σε προγράμματα που προσομοιάζουν τη λειτουργία σύνδεσης.

3. *Ηλεκτρονική παρακολούθηση.* Κατά τη μετάδοση των συνθηματικών προς ένα υπολογιστικό σύστημα (κυρίως σε κατανεμημένα περιβάλλοντα) είναι δυνατόν αυτά να υποκλαπούν. Η κρυπτογράφηση εδώ δεν λύνει το πρόβλημα, καθώς η επανακρυπτογράφηση του ίδιου συνθηματικού θα δώσει το ίδιο κρυπτογραφημένο κείμενο. Συνεπώς, σε ό,τι αφορά το σύστημα που λαμβάνει το συνθηματικό, αν του σταλεί ξανά το κρυπτογραφημένο κείμενο που υπεκλάπη θα το θεωρήσει ως σωστό συνθηματικό.

4. *Πρόσβαση στο αρχείο συνθηματικών.* Αν το αρχείο συνθηματικών δεν είναι επαρκώς προστατευμένο με μηχανισμούς ελέγχου πρόσβασης, μπορεί να προσπελασθεί από οποιονδήποτε χρήστη. Τα περισσότερα αρχεία συνθηματικών συνήθως προστατεύονται με μονόδρομη κρυπτογράφηση (δηλαδή κωδικοποίηση της οποίας το αποτέλεσμα δεν μπορεί να χρησιμοποιηθεί για να βρεθεί το αρχικό κείμενο με υπολογιστικά εφικτό

τρόπο), έτσι ώστε τα συνθηματικά να μην είναι διαθέσιμα στους διαχειριστές ή τους εισβολείς. Ακόμη και σ' αυτήν την περίπτωση όμως, είναι δυνατή η προσπάθεια εύρεσης των συνθηματικών με εξαντλητική αναζήτηση, δεδομένης μάλιστα της αυξημένης υπολογιστικής ισχύος των σύγχρονων υπολογιστών.

Πολλές φορές τα συνθηματικά χρησιμοποιούνται από τα λειτουργικά συστήματα και για έλεγχο πρόσβασης σε πόρους. Η πολλαπλή αυτή χρήση των συνθηματικών δεν είναι καλή ιδέα, καθώς μειώνει τη συνολική ασφάλεια του συστήματος.

Προκειμένου να αντιμετωπισθούν τα ανωτέρω ζητήματα σχετικά με την ασφάλεια των συνθηματικών, είναι δυνατόν να ληφθούν τα κάτωθι μέτρα:

1. *Γεννήτριες συνθηματικών*. Αν δεν επιτρέπεται στους χρήστες να δώσουν τα συνθηματικά που οι ίδιοι επιθυμούν, προφανώς είναι αδύνατη η επιλογή από μέρους τους συνθηματικών που μπορούν να μαντευθούν εύκολα. Μερικές γεννήτριες παράγουν συνθηματικά που είναι εύκολο να προφερθούν, προκειμένου να μειωθεί η πιθανότητα να τα σημειώσει ο χρήστης σε κάποιο ευπρόσιτο σε άλλους μέρος.
2. *Περιορισμός στις αποτυχημένες προσπάθειες σύνδεσης*. Πολλά λειτουργικά συστήματα παρέχουν τη δυνατότητα κλειδώματος ενός κωδικού μετά από ένα συγκεκριμένο πλήθος διαδοχικών αποτυχημένων προσπαθειών σύνδεσης, αποτρέποντας έτσι την αποκάλυψη συνθηματικών με τη μέθοδο «δοκιμή και λάθος».
3. *Χαρακτηριστικά συνθηματικών*. Το σύστημα μπορεί να επιβάλλει στους χρήστες να επιλέγουν συνθηματικά που (α) έχουν ένα ελάχιστο μήκος (β) περιέχουν ειδικούς χαρακτήρες (γ) δεν «μοιάζουν» με την ταυτότητα χρήστη (δ) δεν περιέχονται σε κάποιο λεξικό. Έτσι τα συνθηματικά γίνονται δύσκολο να μαντευθούν (αν και αυξάνεται η πιθανότητα να τα σημειώσει ο χρήστης σε κάποιο χαρτί).
4. *Αλλαγή συνθηματικών*. Η περιοδική αλλαγή των συνθηματικών μπορεί να μειώσει τις επιπτώσεις από μία ενδεχόμενη διαρροή του αρχείου κρυπτογραφημένων συνθηματικών και να καταστήσει δυσχερέστερες τις επιθέσεις που βασίζονται σε εξαντλητική δοκιμή όλων των δυνατών συνθηματικών. Οι πολύ συχνές αλλαγές ωστόσο δεν γίνονται ευμενώς δεκτές από τους χρήστες.
5. *Προστασία του αρχείου συνθηματικών*. Τα συνθηματικά πρέπει να αποθηκεύονται μετά από τη μονόδρομη κρυπτογράφησή τους, ενώ το ίδιο το αρχείο που περιέχει τα συνθηματικά πρέπει κανονικά να είναι απροσπέλαστο για τους κοινούς χρήστες.
6. *Χρήση πλήκτρων ενεργοποίησης της διαδικασίας σύνδεσης*. Μια σημαντική απειλή για την αποκάλυψη συνθηματικών σε τρίτους είναι η χρήση προγραμμάτων που προσομοιώνουν τη διαδικασία σύνδεσης και προτρέπουν τους χρήστες να εισάγουν τα διαπιστευτήρια σύνδεσής τους, τα οποία αποστέλλονται στον συγγραφέα του προγράμματος. Το πρόβλημα αυτό αντιμετωπίζεται με την εισαγωγή ενός επιπλέον βήματος στη διαδικασία σύνδεσης, στο οποίο ο χρήστης πατάει έναν συγκεκριμένο συνδυασμό πλήκτρων, για τον οποίο το λειτουργικό σύστημα εγγυάται ότι δεν είναι δυνατόν να παγιδευτεί ή να χρησιμοποιηθεί για άλλο σκοπό (π.χ. CTRL-ALT-DEL στα Windows). Μετά το πάτημα του συγκεκριμένου συνδυασμού

πλήκτρων, ο χρήστης είναι βέβαιος ότι η οθόνη που του παρουσιάζεται είναι η κανονική οθόνη σύνδεσης του συστήματος.

Κρυπτογραφικά κλειδιά

Αν και η δυνατότητα διακρίβωσης της ταυτότητας μέσω κρυπτογραφικού κλειδιού βασίζεται σε κάτι που γνωρίζει ο χρήστης, αυτός πρέπει συνήθως να έχει στη διάθεσή του κάποια συσκευή (π.χ. έξυπνη κάρτα ή PC), η οποία θα εκτελέσει τους κρυπτογραφικούς υπολογισμούς. Για τον λόγο αυτό, η συγκεκριμένη προσέγγιση περιγράφεται στο εδάφιο που πραγματεύεται τη διακρίβωση ταυτότητας βάσει αντικειμένων που έχει στην κατοχή του ο χρήστης.

4.1.1.2 Τεχνικές όπου ζητάται κάτι που ο χρήστης κατέχει

Αν και αρκετές τεχνικές βασίζονται αποκλειστικά σε κάτι που ο χρήστης κατέχει, συνήθως ζητάται παράλληλα και κάτι που ο χρήστης γνωρίζει. Ο συνδυασμός αυτός συνήθως αποφέρει υψηλότερα επίπεδα ασφάλειας από προσεγγίσεις όπου απαιτείται *μόνον* κάτι που γνωρίζει ο χρήστης ή *μόνον* κάτι που κατέχει.

Τα αντικείμενα που κατέχει ο χρήστης για σκοπούς διακρίβωσης ταυτότητας καλούνται *διακριτικά (tokens)*, τα οποία διαχωρίζονται σε *διακριτικά μνήμης* και σε *έξυπνα διακριτικά.*

Διακριτικά μνήμης

Τα διακριτικά μνήμης αποθηκεύουν αλλά δεν επεξεργάζονται πληροφορίες. Η εγγραφή και η ανάγνωση δεδομένων σε/από αυτά διενεργείται μέσω ειδικών συσκευών. Ο πιο διαδεδομένος τύπος διακριτικών μνήμης είναι οι κάρτες που είναι εφοδιασμένες με μία μαγνητική ταινία (π.χ. κάρτες τραπεζών για ανάληψη μετρητών), οι οποίες διαβάζονται από ειδικούς αναγνώστες (αυτόματες ταμειακές μηχανές – ATM). Οι χρήστες απαιτείται, πέρα από το ίδιο το διακριτικό (την κάρτα), να εισάγουν και έναν προσωπικό αριθμό αναγνώρισης. Σε μερικά συστήματα η διακρίβωση ταυτότητας γίνεται αποκλειστικά μέσω ενός διακριτικού, χωρίς να ζητάται κάτι που ο χρήστης γνωρίζει. Τα συστήματα αυτά είναι σχετικά λίγα και κυρίως αφορούν τον έλεγχο φυσικής πρόσβασης σε χώρους.

Τα διακριτικά μνήμης που συνδυάζονται με προσωπικούς αριθμούς αναγνώρισης είναι πολύ πιο ασφαλή από τα συνθηματικά. Επιπρόσθετα, τα διακριτικά μνήμης είναι ιδιαίτερα φθηνά, ενώ για καταφέρει ένας μη εξουσιοδοτημένος χρήστης να αποκτήσει πρόσβαση πρέπει *και* να έχει στην κατοχή του το διακριτικό *και* να γνωρίζει και τον αριθμό. Ο συνδυασμός αυτός είναι πιο δύσκολο να αποκτηθεί απ' ό,τι ένα ζεύγος (ταυτότητα χρήστη, συνθηματικό), ειδικότερα αν λάβουμε υπόψη ότι οι ταυτότητες χρήστη δεν είναι μυστικές. Ένα περαιτέρω πλεονέκτημα των διακριτικών είναι ότι μπορούν να χρησιμοποιηθούν για παραγωγή αρχείων καταγραφής, χωρίς να είναι απαραίτητο να εισάγει ο χρήστης την ταυτότητά του για κάθε δοσοληψία ή συμβάν που πρέπει να καταγραφεί, καθώς το σύστημα αντλεί τη σχετική πληροφορία από το διακριτικό. Αν το διακριτικό χρησιμοποιείται –εκτός από την διακρίβωση ταυτότητας στον υπολογιστή– και για είσοδο και έξοδο από τον φυσικό χώρο, τότε οι χρήστες αναγκαστικά το αφαιρούν από τον υπολογιστή όταν απομακρύνονται από τον χώρο. Με τον τρόπο αυτό μηδενίζεται η πιθανότητα να χρησιμοποιήσει κανείς κάποιο τερματικό που άφησε ανεπιτήρητο ένας χρήστης.

Η χρήση διακριτικών μνήμης όμως έχει και κάποια μειονεκτήματα. Αν και είναι τελικά δυνατή η πραγματοποίηση πολύ καλά προετοιμασμένων επιθέσεων ενάντια σε

συστήματα που χρησιμοποιούν αυτή τη μέθοδο, τα περισσότερα προβλήματα ανάγονται στο κόστος, τη διαχείριση, την απώλεια των διακριτικών, τη δυσαρέσκεια των χρηστών και τη διαρροή των προσωπικών αριθμών αναγνώρισης.

1. *Απαίτηση για εξειδικευμένες συσκευές ανάγνωσης.* Για να είναι δυνατόν να χρησιμοποιηθεί το διακριτικό μνήμης σε ένα σύστημα, απαιτείται μία διάταξη που θα διαβάζει το διακριτικό και τον προσωπικό αριθμό του χρήστη. Κατόπιν η συσκευή είτε θα στέλνει τα δεδομένα στο σύστημα προς διακρίβωση της ορθότητας, είτε θα διενεργεί η ίδια τη διακρίβωση. Στην πρώτη περίπτωση απαιτείται η χρήση κρυπτογραφίας κατά τη μετάδοση για να μην είναι δυνατόν να υποκλαπούν τα δεδομένα.

2. *Απώλεια διακριτικού.* Αν ένας χρήστης χάσει το διακριτικό του, δεν θα μπορεί να συνδεθεί στο σύστημα μέχρι να αντικατασταθεί το διακριτικό. Με τον τρόπο αυτό αυξάνεται το διαχειριστικό κόστος και η επιβάρυνση. Το απολεσθέν διακριτικό μπορεί να έχει κλαπεί ή μπορεί να βρεθεί από κάποιον και ο νέος κάτοχός του μπορεί να επιχειρήσει να εισέλθει στο σύστημα με την ταυτότητα του χρήστη στον οποίο κανονικά ανήκει το διακριτικό. Αν το σύστημα απαιτεί πέραν του διακριτικού και κάποιον προσωπικό αριθμό αναγνώρισης, οποιαδήποτε μέθοδος από αυτές που περιγράφηκαν ανωτέρω για την κλοπή συνθηματικών είναι δυνατόν να χρησιμοποιηθεί για κλοπή του προσωπικού αριθμού.

3. *Δυσαρέσκεια χρηστών.* Γενικά, οι χρήστες επιθυμούν υπολογιστές εύκολους στη χρήση. Πολλοί χρήστες το βρίσκουν άβολο να κουβαλάνε και να χρησιμοποιούν ένα διακριτικό. Οι αντιδράσεις ωστόσο περιορίζονται αν είναι προφανής η αναγκαιότητα για αυξημένη ασφάλεια.

Έξυπνα διακριτικά

Ένα έξυπνο διακριτικό επεκτείνει τη λειτουργικότητα ενός διακριτικού μνήμης, ενσωματώνοντας ένα ή περισσότερα ολοκληρωμένα κυκλώματα. Όταν χρησιμοποιείται για διακρίβωση ταυτότητας, ένα έξυπνο διακριτικό εμπίπτει στην κατηγορία τεχνικών όπου ζητάται κάτι που ο χρήστης κατέχει, ενώ είναι δυνατόν παράλληλα να ζητάται κάτι που ο χρήστης γνωρίζει (όπως π.χ. ένας προσωπικός αριθμός αναγνώρισης).

Υπάρχουν πολλά διαφορετικά είδη έξυπνων διακριτικών. Γενικά, τα έξυπνα διακριτικά μπορούν να καταταχθούν σε κατηγορίες βάσει των φυσικών χαρακτηριστικών τους, της διεπαφής τους και των πρωτοκόλλων που χρησιμοποιούν. Οι κατηγοριοποιήσεις αυτές δεν είναι αμοιβαία αποκλειόμενες.

1. *Φυσικά χαρακτηριστικά.* Τα έξυπνα διακριτικά μπορεί να είναι «έξυπνες κάρτες», οι οποίες μοιάζουν με πιστωτικές κάρτες αλλά περιλαμβάνουν επίσης και κάποιον μικροεπεξεργαστή. Οι έξυπνες κάρτες περιγράφονται από ένα πρότυπο του διεθνούς οργανισμού προτύπων (ISO). Τα έξυπνα διακριτικά που δεν είναι «έξυπνες κάρτες» μοιάζουν συνήθως με μικρές αριθμομηχανές.

2. *Διεπαφή.* Τα έξυπνα διακριτικά έχουν μία διεπαφή που μπορεί να τους επιτρέπει να επικοινωνούν είτε με ανθρώπους είτε με ηλεκτρονικά συστήματα. Τα διακριτικά που έχουν διεπαφή για επικοινωνία με ανθρώπους ενσωματώνουν οθόνες ή/και πληκτρολόγια για να επιτρέπουν την εισαγωγή και την προβολή στοιχείων. Τα διακριτικά με διεπαφές για επικοινωνία με ηλεκτρονικά συστήματα ανταλλάσσουν δεδομένα με ειδικές διατάξεις

ανάγνωσης/εγγραφής. Τα διακριτικά που έχουν τη μορφή αριθμομηχανών συνήθως διαθέτουν διεπαφή για επικοινωνία με ανθρώπους.

3. *Πρωτόκολλο.* Υπάρχουν πολλά πρωτόκολλα που ένα έξυπνο διακριτικό μπορεί να χρησιμοποιήσει για διακρίβωση ταυτότητας. Γενικά μπορούν να διακριθούν σε τρεις κατηγορίες:

    a. *Στατική ανταλλαγή συνθηματικών.* Βάσει του πρωτοκόλλου αυτού οι χρήστες εισάγουν το συνθηματικό τους στο έξυπνο διακριτικό, το οποίο κατόπιν συνεργάζεται με τον υπολογιστή για τη διακρίβωση της ταυτότητας του χρήστη.

    b. *Δυναμική γέννηση συνθηματικών.* Βάσει του πρωτοκόλλου αυτού, το έξυπνο διακριτικό δημιουργεί μία μοναδική τιμή, π.χ. έναν οκταψήφιο αριθμό, ο οποίος αλλάζει περιοδικά (π.χ. κάθε λεπτό). Αν το διακριτικό έχει διεπαφή προσανατολισμένη σε επικοινωνία με ανθρώπους, ο χρήστης απλά διαβάζει τον αριθμό από την οθόνη του διακριτικού και το εισάγει στον υπολογιστή για διακρίβωση της ταυτότητάς του. Αν το διακριτικό έχει διεπαφή προσανατολισμένη σε επικοινωνία με ηλεκτρονικές διατάξεις, ο αριθμός αποστέλλεται αυτομάτως. Αν η εισαχθείσα τιμή είναι σωστή (δηλαδή είναι ανάμεσα στις παραδεκτές τιμές που ο υπολογιστής γνωρίζει ότι μπορεί να παράγει το συγκεκριμένο διακριτικό για τη δεδομένη χρονική περίοδο), θεωρείται ότι η ταυτότητα του χρήστη έχει διακριβωθεί.

    c. *Πρωτόκολλα ερωταποκρίσεων.* Βάσει του πρωτοκόλλου αυτού ο υπολογιστής δημιουργεί μία *ερώτηση* π.χ. μία τυχαία ακολουθία από αριθμούς. Το έξυπνο διακριτικό παράγει μία *απάντηση*, ως συνάρτηση της ερώτησης, η οποία αποστέλλεται στον υπολογιστή, και ο υπολογιστής διακριβώνει την ταυτότητα του χρήστη βάσει της απάντησης. Οι αλγόριθμοι υπολογισμού της απάντησης από την ερώτηση στηρίζονται σε κρυπτογραφικές μεθόδους. Τα πρωτόκολλα ερωταποκρίσεων μπορούν να χρησιμοποιηθούν είτε με διεπαφές προσανατολισμένες σε επικοινωνία με ανθρώπους είτε με διεπαφές για επικοινωνία με ηλεκτρονικές διατάξεις.

Τα έξυπνα διακριτικά παρέχουν μεγάλη ευελιξία και μπορούν να λύσουν πολλά προβλήματα διακρίβωσης ταυτότητας. Τα πλεονεκτήματα που αποκομίζουμε από τη χρήση τους ποικίλλουν, ανάλογα με το είδος των διακριτικών που χρησιμοποιούνται, στη γενική περίπτωση πάντως προσφέρουν μεγαλύτερη ασφάλεια από τα διακριτικά μνήμης. Τα έξυπνα διακριτικά μπορούν να λύσουν και το πρόβλημα της υποκλοπής των συνθηματικών κατά τη δικτυακή επικοινωνία, ακόμη και αν αυτή πραγματοποιείται μέσα από ανοικτά δημόσια δίκτυα, καθώς μπορούν να εφαρμόσουν τεχνικές *συνθηματικών μίας χρήσης* (π.χ. στην περίπτωση του πρωτοκόλλου ερωταποκρίσεων).

1. *Συνθηματικά μιας χρήσης.* Τα έξυπνα διακριτικά μπορούν να χρησιμοποιούν είτε δυναμική γέννηση συνθηματικών είτε πρωτόκολλα ερωταποκρίσεων για να παράξουν συνθηματικά μίας χρήσης. Η υποκλοπή του συνθηματικού σε αυτή την περίπτωση δεν συνεπάγεται κάποιο πρόβλημα στην ασφάλεια, καθώς σε κάθε διαδικασία διακρίβωσης ταυτότητας χρησιμοποιείται διαφορετικό συνθηματικό.

2. *Ελαττωμένος κίνδυνος παραχάραξης*. Γενικά, η μνήμη ενός έξυπνου διακριτικού δεν είναι αναγνώσιμη αν δεν εισαχθεί ο προσωπικός αριθμός αναγνώρισης. Επιπρόσθετα, τα έξυπνα διακριτικά είναι πιο πολύπλοκα και πιο δύσκολο να δημιουργηθούν παραχαραγμένα αντίγραφά τους.
3. *Χρήση με πολλές εφαρμογές*. Τα έξυπνα διακριτικά με διασύνδεση επικοινωνίας με ηλεκτρονικές συσκευές επιτρέπουν στους χρήστες τους να προσπελαύνουν πολλούς υπολογιστές και υπηρεσίες με μία μόνο διαδικασία σύνδεσης. Οι χρήστες πιστοποιούν τον εαυτό τους στο διακριτικό και στη συνέχεια αυτό μπορεί να εμπεριέχει όλη την απαραίτητη πληροφορία για να πιστοποιήσει τον χρήστη στις υπηρεσίες ή υπολογιστές που αυτός προσπελαύνει.

Η χρήση των έξυπνων διακριτικών έχει όμως και προβλήματα. Όπως και με τα διακριτικά μνήμης, αυτά εντοπίζονται στο κόστος, τη διαχείριση, την απώλεια των διακριτικών και τη δυσαρέσκεια των χρηστών. Η πιθανότητα διαρροής των προσωπικών αριθμών αναγνώρισης είναι ελαττωμένη, καθώς αυτοί δεν μεταδίδονται προς τους υπολογιστές αλλά ελέγχονται από το ίδιο το διακριτικό. Από την άλλη πλευρά βέβαια, τα έξυπνα διακριτικά κοστίζουν περισσότερο από τα διακριτικά μνήμης. Υπάρχει βέβαια και πάλι η αναγκαιότητα για διατάξεις ανάγνωσης-εγγραφής, ενώ όταν χρησιμοποιούνται διακριτικά με διεπαφές για επικοινωνία με ανθρώπους, οι χρήστες πρέπει να πληκτρολογούν μακροσκελείς συμβολοσειρές, κάτι που αυξάνει τη δυσαρέσκειά τους. Τέλος, υπάρχει υψηλό διαχειριστικό κόστος, ειδικότερα όταν χρησιμοποιούνται τεχνικές κρυπτογραφίας.

4.1.1.3 Πιστοποίηση βασισμένη σε βιομετρικά χαρακτηριστικά

Τα συστήματα διακρίβωσης ταυτότητας βάσει βιομετρικών χαρακτηριστικών αξιοποιούν τη μοναδικότητα ορισμένων χαρακτηριστικών των ανθρώπων για να τους αναγνωρίσουν. Μπορεί να εξετάζουν φυσικά χαρακτηριστικά (π.χ. δακτυλικά αποτυπώματα, γεωμετρία χειρός κ.τ.λ.) ή χαρακτηριστικά συμπεριφοράς (π.χ. τρόπος υπογραφής ή χροιά φωνής). Υπάρχουν ήδη υλοποιημένες τεχνικές διακρίβωσης ταυτότητας που βασίζονται σε βιομετρικά μεγέθη και που έχουν ολοκληρωθεί σε υπολογιστικά συστήματα.

Η διακρίβωση ταυτότητας βάσει βιομετρικών μεγεθών είναι δαπανηρή και τεχνικά περίπλοκη, ενώ οι χρήστες είναι δυνατόν να μην την αποδεχθούν εύκολα. Μπορούν να παράσχουν υψηλά επίπεδα ασφάλειας, αλλά η τεχνολογία τους δεν είναι ακόμη το ίδιο ώριμη όπως αυτή των διακριτικών. Σ' αυτά πρέπει να συνυπολογίσουμε ότι μερικά βιομετρικά χαρακτηριστικά μπορεί να μεταβάλλονται, π.χ. η χροιά της φωνής κάποιου μπορεί να αλλάξει σε συνθήκες υπερβολικής έντασης ή ενός καλού κρυολογήματος.

Ο γενικός τρόπος λειτουργίας των συστημάτων διακρίβωσης ταυτότητας βάσει βιομετρικών χαρακτηριστικών είναι ο ακόλουθος: πριν από οποιαδήποτε προσπάθεια διακρίβωσης ταυτότητας για έναν συγκεκριμένο χρήστη μετρώνται τα σχετικά βιομετρικά μεγέθη του και οι μετρήσεις αυτές συσχετίζονται με το συγκεκριμένο φυσικό πρόσωπο. Κατόπιν, όταν ένας χρήστης επιχειρεί να συνδεθεί με το σύστημα, μετρώνται τα ίδια βιομετρικά χαρακτηριστικά του και συγκρίνονται με αυτά που έχουν αποθηκευθεί. Τα αποτελέσματα της σύγκρισης καθορίζουν αν η σύνδεση του χρήστη θα γίνει αποδεκτή ή όχι.

4.1.2 Έλεγχος προσπέλασης

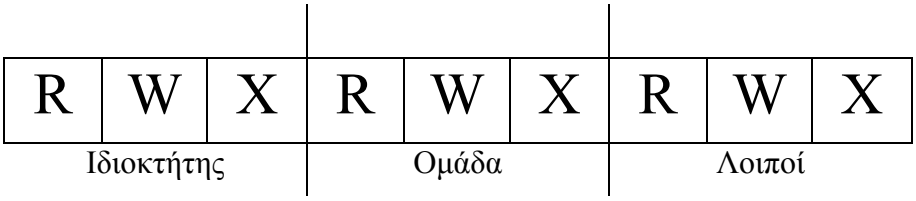
Από τη στιγμή που έχει διακριβωθεί η ταυτότητα ενός χρήστη (ή, γενικότερα, μιας οντότητας), το σύστημα θα πρέπει να φροντίζει έτσι ώστε η οντότητα αυτή να μπορεί να ενεργήσει μόνο στα πλαίσια των κανόνων που καθορίζονται από την πολιτική ασφάλειας. Αυτό μπορεί να επιτευχθεί εφαρμόζοντας *ελέγχους προσπέλασης*. Σχετικά με τους ελέγχους προσπέλασης ισχύουν οι ακόλουθες έννοιες:

- *Υποκείμενα*. Πρόκειται για τις ενεργές οντότητες στο σύστημα (χρήστες, διεργασίες, υπηρεσίες)
- *Αντικείμενα*. Με τον όρο αυτό περιγράφονται οι πόροι ή οι παθητικές οντότητες στο σύστημα (αρχεία, συσκευές, προγράμματα)
- *Τρόπος προσπέλασης*. Ο όρος αυτός αναφέρεται στην ενέργεια που πραγματοποιεί ένα υποκείμενο σε ένα αντικείμενο π.χ. ανάγνωση, εγγραφή, εκτέλεση, αναφορά ιδιοχαρακτηριστικών.

Ο έλεγχος προσπέλασης συνίσταται στην εξέταση αν το υποκείμενο έχει δικαίωμα για τον συγκεκριμένο τρόπο προσπέλασης στο αντικείμενο, και στην απαγόρευση της ενέργειας, αν τελικά δεν υπάρχει το σχετικό δικαίωμα.

Υπάρχουν δύο γενικές κατηγορίες σχημάτων προσπέλασης, τα *κατ' επιλογήν* και τα *υποχρεωτικά*. Στα *κατ' επιλογήν* σχήματα, ο ιδιοκτήτης του αντικειμένου αποφασίζει τα δικαιώματα που θα εκχωρήσει σε διάφορες οντότητες πάνω στο αντικείμενο. Στα κατ΄ επιλογήν σχήματα ελέγχου προσπέλασης μπορούμε να κατατάξουμε τα bits RWX του UNIX, καθώς και τους *πίνακες ελέγχου προσπέλασης* που λαμβάνουν τη μορφή *λιστών ελέγχου προσπέλασης* ή *λιστών προσδιοριστών δικαιωμάτων*.

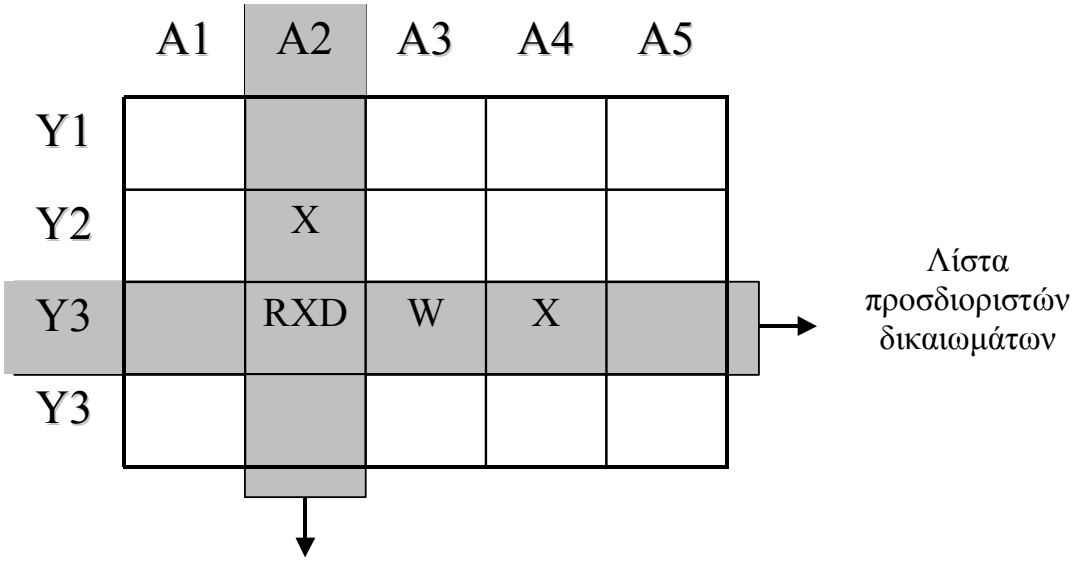
Βάσει του σχήματος των bits RWX του UNIX, σε κάθε *αντικείμενο* αντιστοιχίζονται εννέα bits, οργανωμένα σε τρεις τριάδες. Η πρώτη από αυτές αφορά τον *ιδιοκτήτη* του αρχείου, η δεύτερη την *ομάδα* στην οποία ανήκει το αρχείο και η τρίτη όλους τους υπόλοιπους χρήστες. Εντός κάθε τριάδας bits, το πρώτο από αυτά προσδιορίζει το δικαίωμα ανάγνωσης, το δεύτερο το δικαίωμα εγγραφής και το τρίτο το δικαίωμα εκτέλεσης πάνω στο αρχείο, για την οντότητα (ή τις οντότητες) που αφορά η ομάδα bits. Το σχήμα των bits RWX απεικονίζεται στο σχήμα που ακολουθεί.



Το σχήμα των bits RWX είναι εύκολο στην υλοποίησή του, είναι σχετικά κατανοητό από τους χρήστες, αλλά είναι ιδιαίτερα δύσκαμπτο στη λεπτομερή ανάθεση προνομίων.

Στους πίνακες ελέγχου προσπέλασης, ουσιαστικά καταρτίζεται ένας πίνακας του οποίου οι γραμμές αντιστοιχούν στα *υποκείμενα*, οι στήλες στα αντικείμενα, ενώ οι τιμές των κελιών του πίνακα προσδιορίζουν τα δικαιώματα που το υποκείμενο που αντιστοιχεί στη συγκεκριμένη γραμμή έχει πάνω στο αντικείμενο που αντιστοιχεί στη συγκεκριμένη στήλη, όπως φαίνεται στο σχήμα που ακολουθεί. Διαβάζοντας τον πίνακα ελέγχου προσπέλασης *κατά στήλες* έχουμε διανύσματα που καθορίζουν όλες τις δυνατές προσβάσεις σε κάθε ένα αντικείμενο ή αλλιώς *λίστες ελέγχου προσπέλασης*. Διαβάζοντας τον πίνακα ελέγχου κατά γραμμές έχουμε διανύσματα που

καθορίζουν όλες τις δυνατές προσβάσεις για κάθε υποκείμενο, ή αλλιώς *λίστες προσδιοριστών δικαιωμάτων*.



Στα *υποχρεωτικά* σχήματα το σύστημα είναι αυτό που αποφασίζει ποιο υποκείμενο μπορεί να προσπελάσει κάθε αντικείμενο και τους επιτρεπτούς τρόπους. Η απόφαση βασίζεται στα ιδιοχαρακτηριστικά του υποκειμένου που ζητά πρόσβαση, και του αντικειμένου που επιχειρείται να προσπελασθεί.

Σε ένα υποχρεωτικό σχήμα ελέγχου προσπέλασης, κάθε υποκείμενο και κάθε αντικείμενο έχει ένα *επίπεδο ασφάλειας*, με κάθε επίπεδο ασφάλειας να αποτελείται από μια *διαβάθμιση* και ένα *σύνολο κατηγοριών*. Όταν ένα υποκείμενο επιχειρεί πρόσβαση σε ένα αντικείμενο, τα επίπεδα ασφαλείας τους συγκρίνονται, και η σύγκριση δίνει ένα από τα αποτελέσματα *ίσο*, *μεγαλύτερο*, *μικρότερο* και *μη συγκρίσιμο*. Για να γίνει πιο κατανοητό το σχήμα αυτό, ας θεωρήσουμε το ακόλουθο παράδειγμα:

- Τρία επίπεδα ασφάλειας, *αδιαβάθμητο*, *εμπιστευτικό*, *απόρρητο*
- Τρεις κατηγορίες ασφάλειας, *προμήθειες*, *λογιστήριο*, *διοίκηση*
- Συγκρίσεις:
    - εμπιστευτικό/(προμήθειες) = εμπιστευτικό/(προμήθειες)
    - απόρρητο/(προμήθειες) > εμπιστευτικό/(προμήθειες)
    - εμπιστευτικό/(προμήθειες) < εμπιστευτικό(προμήθειες, λογιστήριο)
    - απόρρητο(προμήθειες) *μη συγκρίσιμο* απόρρητο(λογιστήριο)

Έχοντας πραγματοποιήσει τη σύγκριση, η πρόσβαση επιτρέπεται ή απαγορεύεται βάσει των εξής κανόνων:

- Η ανάγνωση επιτρέπεται αν το υποκείμενο έχει μεγαλύτερο ή ίσο επίπεδο ασφάλειας από το αντικείμενο (no read-up)
- Η εγγραφή επιτρέπεται αν το υποκείμενο έχει μικρότερο ή ίσο επίπεδο ασφάλειας από το αντικείμενο (no write-down)

Η χρησιμότητα του πρώτου κανόνα είναι προφανής· ο δεύτερος κανόνας αποτρέπει την μεταφορά πληροφορίας από ανώτερο επίπεδο σε κατώτερο, απ' όπου μπορεί να διαβαστεί από υποκείμενα κατώτερης εξουσιοδότησης.

Προκειμένου να εφαρμόζονται οι κανόνες που ορίζονται για τον έλεγχο πρόσβασης, είναι απαραίτητο να εξετάζονται όλες οι απόπειρες προσπέλασης αντικειμένων και ανάλογα να επιτρέπονται ή να απαγορεύονται. Ένας τρόπος εφαρμογής των κανόνων αυτών είναι η εισαγωγή μιας ενότητας λογισμικού στο λειτουργικό σύστημα που ονομάζεται *επόπτης αναφορών*, δια μέσω του οποίου διέρχονται όλες οι αιτήσεις προσπέλασης σε αντικείμενα. Για κάθε πρόσβαση, ο επόπτης αναφορών συμβουλεύεται τα στοιχεία εξουσιοδοτήσεων, ταυτοτήτων και δικαιωμάτων και αποφασίζει ανάλογα. Ο επόπτης αναφορών πρέπει να μην είναι δυνατόν να παρακαμφθεί (δηλαδή κάποιες αιτήσεις προσπέλασης να μην διέλθουν μέσω αυτού), να μην είναι δυνατόν να ξεγελασθεί (δηλαδή να αποφασίζει πάντα χρησιμοποιώντας τις σωστές ταυτότητες αντικειμένων και υποκειμένων και τα σωστά στοιχεία εξουσιοδότησης), να έχει τη δυνατότητα καταγραφής και θα πρέπει να είναι δυνατόν να ελεγχθεί η ορθότητά του.

Μια εναλλακτική προσέγγιση είναι η συγκέντρωση όλων των ελέγχων και μηχανισμών που σχετίζονται με την ασφάλεια σε μία ειδική ενότητα του λειτουργικού συστήματος, τον *πυρήνα ασφάλειας*. Ο πυρήνας ασφάλειας είναι υπεύθυνος για την υλοποίηση όλων των μηχανισμών ασφάλειας του Λ.Σ. και σε όλα τα επίπεδα (υλικό, εφαρμογές, χρήστες κ.λπ.), ενώ τα υπόλοιπα τμήματα του Λ.Σ. απλά καλούν λειτουργίες που παρέχονται από τον πυρήνα ασφάλειας. Στα υπέρ της προσέγγισης αυτής είναι ο διαχωρισμός των λειτουργιών ασφάλειας από τις λοιπές λειτουργίες του Λ.Σ., η ενοποίησή τους σε ένα ξεχωριστό τμήμα, το οποίο μπορεί να συντηρηθεί και να επαληθευτεί ξεχωριστά από το υπόλοιπο Λ.Σ. Από την άλλη πλευρά, ένας διαχωρισμός τέτοιου είδους καταργεί πολλές βελτιστοποιήσεις οδηγώντας σε υποβάθμιση της απόδοσης, αυξάνει το μέγεθος του συστήματος, ενώ είναι πρακτικά αδύνατο να ενσωματωθεί σε υπάρχοντα λειτουργικά συστήματα, καθ' όσον απαιτεί αλλαγή της δομής τους.

4.2 Τεχνικές διασφάλισης

Οι τεχνικές διασφάλισης αποσκοπούν στο να εξασφαλίσουν ότι ένα σύστημα είναι ασφαλές ή/και στην ανάδειξη και έγκαιρη διόρθωση των τρωτών σημείων του συστήματος, πριν αυτά γίνουν αντικείμενο εκμετάλλευσης από εισβολείς.

Η πρώτη προσέγγιση, η οποία συνίσταται από τους μηχανικούς λογισμικού, είναι η υιοθέτηση καλών προγραμματιστικών τεχνικών και πρακτικών, καθώς και εκτεταμένες δοκιμές του προϊόντος λογισμικού. Μολονότι ακούγεται πολύ δελεαστική προσέγγιση, στην πράξη δεν δίνει αποτελέσματα καθώς τα προϊόντα λογισμικού που κυκλοφορούν στην αγορά *θεωρητικά τουλάχιστον* έχουν αναπτυχθεί σωστά και ελεγχθεί διεξοδικά, χωρίς αυτό να αποτρέπει την ύπαρξη προβλημάτων ασφάλειας.

Μία δεύτερη προσέγγιση είναι η ανάλυση του συστήματος για εντοπισμό πιθανών αδυναμιών. Σύμφωνα με την πρακτική αυτή δημιουργείται μία «ομάδα τίγρης» (tiger team), η οποία συλλέγει γνωστές αδυναμίες και τεχνικές εκμετάλλευσής τους, επιχειρεί μία γενίκευση και κατόπιν τις εφαρμόζει στο υπό διερεύνηση σύστημα. Το σκεπτικό της διενέργειας μίας τέτοιας ελεγχόμενης επίθεσης είναι να εντοπισθούν έγκαιρα και να διορθωθούν οι αδυναμίες από το προσωπικό ασφάλειας πριν τις εντοπίσουν και τις εκμεταλλευτούν οι –λιγότερο φιλικοί– εισβολείς. Μία τέτοια

δοκιμή ωστόσο θα καταδείξει ενδεχομένως την *ύπαρξη* αδυναμιών, δεν εγγυάται όμως την απουσία τους.

Η τρίτη προσέγγιση είναι η προσπάθεια πρόληψης ή ανίχνευσης των προσπαθειών για εκμετάλλευση των αδυναμιών των συστημάτων. Συνολικά, η ασφάλεια των συστημάτων διακυβεύεται από:

- Προγραμματιστικά σφάλματα σε μεμονωμένες διεργασίες, όπως π.χ. υπερχείλιση ενδιάμεσης μνήμης, συνθήκες ανταγωνισμού, δούρειοι ίπποι κ.λπ.
- Απρόσμενες αλληλεπιδράσεις μεταξύ προγραμμάτων, όπως εσφαλμένη ανάθεση δικαιωμάτων σε αρχεία, σφάλματα στη δομή και το περιεχόμενο αρχείων διαμόρφωσης κ.ά.

Προκειμένου να αντιμετωπισθούν τα ζητήματα αυτά είναι δυνατόν να υιοθετηθούν *στατικές* ή δυναμικές μέθοδοι, όπως συνοψίζεται στον πίνακα που ακολουθεί:

	Ανάλυση (στατική μέθοδος)	**Ανίχνευση επιβολή πολιτικών (δυναμική μέθοδος)**
Προγραμματιστικά σφάλματα	Ανάλυση πρωτογενούς κώδικα για ανίχνευση σφαλμάτων	Παρακολούθηση συμπεριφοράς και επιβολή ασφαλών προτύπων για τα προγράμματα
Σφάλματα ολοκλήρωσης	Ανάλυση διαμόρφωσης συστήματος για εντοπισμό αδυναμιών	Ανίχνευση προσπαθειών για εκμετάλλευση αδυναμιών

Η *ανάλυση του πρωτογενούς κώδικα για ανίχνευση σφαλμάτων* (security audit) είναι μία διαδικασία όπου ειδικοί περί την ασφάλεια αναλύουν γραμμή προς γραμμή τον πρωτογενή κώδικα των συστημάτων για εντοπισμό πιθανών ευπαθειών. Δεδομένου όμως ότι η ανάλυση γίνεται από ανθρώπους πάντα ενυπάρχει ο κίνδυνος σφάλματα να «ξεφύγουν» και έτσι προϊόντα που έχουν περάσει από πολλαπλές αναλύσεις ασφάλειας κώδικα διαπιστώθηκε στο τέλος ότι έχουν προβλήματα ασφάλειας. Σχετικά με τις δυναμικές μεθόδους, υπάρχουν οι ακόλουθες διαστάσεις:

- *Ανίχνευση αδόκιμων τρόπων χρήσης*. Με την προσέγγιση αυτή κωδικοποιούνται οι αδόκιμοι τρόποι χρήσης του συστήματος και επιχειρείται η ανίχνευση εμφανίσεών των. Κάθε τέτοια εμφάνιση είναι και μία πιθανή επίθεση.
- *Ανίχνευση μη φυσιολογικών συμπεριφορών*. Βάσει αυτής της τεχνικής, το σύστημα έχει κάποια *«φυσιολογική συμπεριφορά»*, ποσοτικοποιούμενη από κάποια μεγέθη. Απόκλιση από αυτά τα μεγέθη σηματοδοτεί κάποια ενδεχόμενη επίθεση.
- *Ανίχνευση βάσει προδιαγραφών*. Με βάση την προσέγγιση αυτή καθορίζεται η προτιθέμενη συμπεριφορά των προγραμμάτων έναντι της κωδικοποίησης των αδόκιμων τρόπων χρήσης. Για παράδειγμα, μπορεί να κωδικοποιηθεί ότι η προτιθέμενη συμπεριφορά ενός κειμενογράφου είναι το άνοιγμα ενός αρχείου ακολουθούμενο από την εγγραφή του. Αν ένας κειμενογράφος διαπιστωθεί ότι προσπαθεί να τροποποιήσει ένα αρχείο συστήματος που δεν έχει ανοιχθεί προηγουμένως ρητώς από τον χρήστη, τότε μάλλον υφίσταται πρόβλημα στην ασφάλεια.

4.3 Ασφάλεια στον προγραμματισμό

Στον τομέα της ανάπτυξης λογισμικού είναι αρκετά διαδεδομένη η αντίληψη ότι ο κώδικας των προγραμμάτων δεν σχετίζεται με την ασφάλεια, η οποία θα πρέπει να είναι μέλημα πρόσθετων εξειδικευμένων μηχανισμών που συμπληρώνουν το λογισμικό. Η αντίληψη αυτή είναι εσφαλμένη καθώς συγκεκριμένοι τρόποι συγγραφής κώδικα δίνουν έδαφος σε επίδοξους εισβολείς να υποβαθμίσουν την ασφάλεια του συστήματος. Πέραν της άγνοιας, η παράβλεψη των πρακτικών ασφαλείας κατά τη συγγραφή κώδικα μπορεί να οφείλεται και σε άλλους λόγους, π.χ. η ασφάλεια να θεωρείται πάρεργο του προγραμματιστή, σε σχέση με την «πιο ενδιαφέρουσα» αντιμετώπιση των λειτουργικών προδιαγραφών ή η έμφαση σε χρονικούς περιορισμούς παράδοσης του λογισμικού που δημιουργεί πίεση από υψηλότερα εταιρικά κλιμάκια. Η μειωμένη ασφάλεια όμως ενός λογισμικού δημιουργεί σοβαρές παρενέργειες στους οίκους ανάπτυξης λογισμικού καθώς, όταν ανακαλυφθεί το κενό στην ασφάλεια, θα πρέπει:

- Οι προγραμματιστές να σταματήσουν την τρέχουσα εργασία τους και να ασχοληθούν με την επιδιόρθωση του προβλήματος
- Να ειδοποιηθούν οι πελάτες/χρήστες
- Να αποσταλεί η νέα έκδοση
- Να εγκατασταθεί η νέα έκδοση (με πιθανή συνδρομή τεχνικών της εταιρείας)

Όλες οι ανωτέρω διαδικασίες ενέχουν σημαντικό οικονομικό και διαχειριστικό κόστος για την εταιρεία, πέραν βέβαια της βλάβης που προκαλείται στην εικόνα της, ειδικότερα αν η ασφάλεια πρόκειται για κρίσιμο παράγοντα στις υπηρεσίες που παρέχει (π.χ. χρηματοπιστωτικοί οργανισμοί, διαχείριση ευαίσθητων δεδομένων κ.λπ.).

Βάσει των ανωτέρω, θα πρέπει η ασφάλεια να ενταχθεί ως απαραίτητο συστατικό της διαδικασίας διασφάλισης ποιότητας του λογισμικού. Δύο κρίσιμες διαστάσεις του θέματος είναι οι ακόλουθες:

- Γνώση από τους προγραμματιστές όλων των πιθανών ευπαθειών ασφάλειας της γλώσσας και του περιβάλλοντος προγραμματισμού, ώστε αυτές να αποφεύγονται ή να αντιμετωπίζονται κατάλληλα
- Επιθεωρήσεις ασφαλείας για τον κώδικα, δηλ. εντοπισμός των πιθανών σημείων όπου διακυβεύεται η ασφάλεια του λογισμικού. Οι επιθεωρήσεις μπορούν να γίνονται:
    - Από προγράμματα ανίχνευσης ευπαθειών
    - Από εξειδικευμένους προγραμματιστές (security audits).

Στο σχήμα που ακολουθεί φαίνεται ενδεικτικά πώς σε ένα τετριμμένο πρόγραμμα γλώσσας C μπορούν να εντοπισθούν αρκετά σημεία με προβληματική ασφάλεια από ένα εργαλείο επιθεώρησης .ασφάλειας. Τα αναφερόμενα προβλήματα επεξηγούνται στη συνέχεια του εδαφίου.

```
#include <stdio.h>
#include <string.h>

int main(void) {
char s1[100], s2[80];

puts("Enter a string: ");
gets(s1);
strcpy(s2, s1);
printf(s2);
return 0;
}
```

(α) Πρόγραμμα probs.c

```
its4 probs.c
probs.c:8:(Urgent) gets
The input buffer can almost always be
overflowed.
Use fgets(buf,size,stdin) instead.
---------------
probs.c:10:(Urgent) printf
Non-constant format strings can often be
attacked.
Use a constant format string.
---------------
probs.c:9:(Very Risky) strcpy
This function is high risk for buffer
overflows
Use strncpy instead.
---------------
```

(β) Ανάλυση ευπαθειών

4.3.1 Ευπάθειες στις γλώσσες C/C++

Οι γλώσσες προγραμματισμού C και C++ παρουσιάζουν το πλουσιότερο ιστορικό αναφερόμενων ευπαθειών σε προγράμματα που έχουν αναπτυχθεί μ' αυτές για δύο κυρίως λόγους:

1. πρόκειται για τις γλώσσες που κατά κύριο λόγο χρησιμοποιούνται στην ανάπτυξη λογισμικού συστήματος και λογισμικού εφαρμογών, συμπεριλαμβάνοντας λειτουργικά συστήματα, βάσεις δεδομένων, παραθυρικά περιβάλλοντα, εξυπηρέτες διαδικτύου, κ.ο.κ.
2. οι γλώσσες αυτές έχουν σχεδιαστεί με τη φιλοσοφία ότι ο προγραμματιστής *γνωρίζει καλά τι κάνει*, επιτρέποντας λειτουργίες που σε άλλες γλώσσες θα απαγορευόταν συνολικά – π.χ. η αυθαίρετη μετατροπή δεικτών από έναν τύπο δεδομένων σε έναν άλλο, η χρήση αρνητικών δεικτών σε έναν πίνακα, η κλήση συναρτήσεων με μεταβλητό αριθμό παραμέτρων κ.ο.κ. Η κακή ή απρόσεκτη χρήση των δυνατοτήτων αυτών οδηγεί πολλές φορές σε προβλήματα ασφάλειας.

4.3.1.1 Υπερχείλιση μνήμης

Ένα μεγάλο σύνολο από προβλήματα ασφάλειας συσχετίζεται με την υπερχείλιση μνήμης. Με τον όρο αυτό αναφερόμαστε στις περιπτώσεις όπου δεσμεύεται ένα ποσό μνήμης για κάποια λειτουργία, αλλά κατά την εκτέλεση της λειτουργίας απαιτείται μεγαλύτερο ποσό από αυτό που δεσμεύτηκε. Οι αναγκαίες συνθήκες για να έχουμε υπερχείλιση ενδιάμεσης μνήμης είναι:

1. Λήψη δεδομένων από μη αξιόπιστη πηγή (πληκτρολόγιο, δίκτυο, αρχείο, διαδιεργασιακή επικοινωνία κ.λπ.)
2. Αποθήκευση των δεδομένων σε ενδιάμεση μνήμη περιορισμένου μεγέθους χωρίς κατάλληλο έλεγχο αν τα δεδομένα όντως χωράνε στη μνήμη

Το αποτέλεσμα είναι να επικαλυφθούν τιμές από γειτονικές μεταβλητές ή/και διευθύνσεις επιστροφής από συναρτήσεις, τιμές αποθηκευμένων καταχωρητών κ.ο.κ.

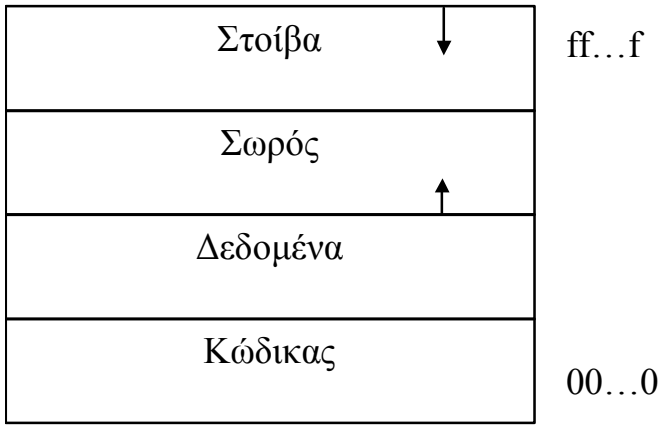
Στις επόμενες παραγράφους αναλύονται οι διάταξη της μνήμης και οι επιπτώσεις στην ασφάλεια που έχουν τα συνηθέστερα λάθη.

Δομή μνήμης διεργασίας

Στα περισσότερα λειτουργικά συστήματα η μνήμη της διεργασίας οργανώνεται σε τμήματα με τα κάτωθι χαρακτηριστικά:

1. *Τμήμα κώδικα*. Περιέχει τον εκτελέσιμο κώδικα της διεργασία, δηλ. τις εντολές που εκτελεί ο επεξεργαστής. Τις περισσότερες φορές είναι ανάγνωσης μόνο.
2. *Τμήμα δεδομένων*. Περιέχει τα *καθολικά δεδομένα* της διεργασίας, καθώς και στατικές μεταβλητές συναρτήσεων (και κλάσεων στη C++). Μερικές φορές διαχωρίζεται σε *τμήμα αρχικοποιημένων δεδομένων* και *τμήμα μη αρχικοποιημένων δεδομένων*, ενώ μπορεί να περιλαμβάνει και *τμήμα σταθερών* που μπορεί να χαρακτηρίζεται ως *ανάγνωσης μόνο*.
3. *Τμήμα στοίβας*. Το τμήμα στοίβας περιέχει τα δεδομένα που δημιουργούνται και καταστρέφονται δυναμικά κατά την εκτέλεση της διεργασίας. Τέτοια δεδομένα είναι οι χώροι που δεσμεύονται με τη malloc, καθώς και οι τιμές παραμέτρων συναρτήσεων και οι τοπικές τους μεταβλητές (που δεν είναι static). Στη στοίβα επίσης αποθηκεύονται και οι *διευθύνσεις επιστροφής* από συναρτήσεις, π.χ. όταν η main() καλεί μία συνάρτηση f(), στη στοίβα αποθηκεύεται η διεύθυνση της εντολής της main() στην οποία θα συνεχιστεί η εκτέλεση μετά το πέρας της f(). Το τμήμα στοίβας διαχωρίζεται συνήθως στον *σωρό*, που περιέχει τα δυναμικά δεσμευόμενα τμήματα και τη *στοίβα* που περιέχει τα σχετιζόμενα με κλήση συναρτήσεων δεδομένα. Η στοίβα επεκτείνεται από τις υψηλότερες διευθύνσεις προς τις χαμηλότερες, ενώ ο σωρός αντιστρόφως.

Το σχήμα που ακολουθεί απεικονίζει τη διάταξη μνήμης μιας τυπικής διεργασίας.



Μηχανισμός κλήσης συναρτήσεων

Όταν μία συνάρτηση f1() καλεί κάποια άλλη f2(), ο μηχανισμός κλήσης των γλωσσών C/C++ προβαίνει στις ακόλουθες ενέργειες:

1. Τοποθετεί στη στοίβα τις τιμές των παραμέτρων που μεταβιβάζονται από την f1() στην f2(). Η διάταξη τοποθέτησης ορίζει ότι πρώτα θα τοποθετηθεί η *τελευταία παράμετρος* και *τελευταία η πρώτη*, έτσι ώστε στο άκρο της στοίβας να βρίσκεται πάντα η πρώτη παράμετρος. Αυτό είναι ουσιώδες για να υποστηρίζονται παράμετροι με μεταβλητό πλήθος παραμέτρων, όπως η

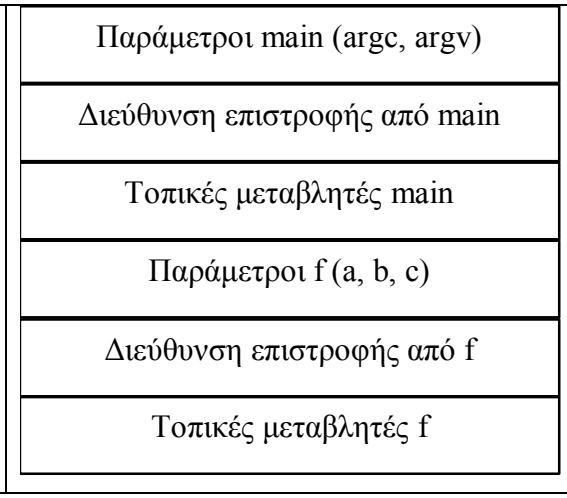
printf(). Ειδικότερα αν ο τύπος επιστροφής της f2 δεν χωράει σε έναν καταχωρητή (ή σε ζεύγος καταχωρητών, κατά περίπτωση) στη στοίβα δεσμεύεται επαρκής χώρος για την τιμή του αποτελέσματος.

2. Εκτελείται μία εντολή γλώσσας μηχανής «CALL f2» με αποτέλεσμα στη στοίβα να τοποθετούνται τα περιεχόμενα του δείκτη εντολών προγράμματος και –πιθανώς- άλλων καταχωρητών, όπως ορίζεται από την αρχιτεκτονική του επεξεργαστή.
3. Οι πρώτες εντολές της συνάρτησης f2() (εισάγονται από τον μεταγλωττιστή κατά την παραγωγή του κώδικα μηχανής) φροντίζουν να δεσμευτεί στη στοίβα επαρκής χώρος για τις τοπικές μεταβλητές.

Κατά την επιστροφή από την f2(), αυτή πρώτα θα καθαρίσει τη στοίβα από τις τοπικές της μεταβλητές και εν συνεχεία θα εκτελέσει μία εντολή γλώσσας μηχανής RETURN, προκειμένου να φορτώσει στον δείκτη εντολών προγράμματος την αποθηκευμένη τιμή (πιθανώς να φορτώνεται και η αποθηκευμένη τιμή άλλων καταχωρητών, αν αυτό προβλέπεται από το υλικό). Τέλος η f1() θα καθαρίσει τη στοίβα από τις τιμές των παραμέτρων που είχε τοποθετήσει εκεί.

Στο ακόλουθο σχήμα απεικονίζεται ένα απόσπασμα προγράμματος C καθώς και η κατάσταση της στοίβας όταν εκτελείται το σώμα της συνάρτησης f(), η οποία συνάρτηση έχει κληθεί άμεσα από τη main.

```
int status = 0; int tbl[5000];
void f(int a, char *b, int c[1000]) {
    int i, j; double sum; …; return; }
int main(int argc, char *argv[]) {
    int x = 7, y = 2; int t[2000];
    …
    f(x + y, "str", &t[999]);
}
```



Υπερχείλιση μνήμης για τοπικές μεταβλητές

Έχοντας υπόψη τη διάταξη της μνήμης και τον μηχανισμό κλήσεων συναρτήσεων, ας θεωρήσουμε την περίπτωση του κώδικα που απεικονίζεται στο ακόλουθο σχήμα:

```
int testPassword(void) {
int password_correct = 0, attempts = 0;
char buf[16];

while ((attempts < 3) && (password_correct == 0)) {
        printf("Enter password: ");
        gets(buf);
        if (strcmp(buf, "secret") == 0)
                password_correct = 1;
        else
                attempts++;
}
if (password_correct) return 1;
else return 0;
}
```

Ο κώδικας ζητάει από τον χρήστη ένα συνθηματικό και αποθηκεύει την απάντησή του στον πίνακα buff μεγέθους 16 bytes. Το βασικό ερώτημα είναι τι θα συμβεί αν ο χρήστης δώσει *πάνω από 16 χαρακτήρες* ως συνθηματικό. Βάσει του μηχανισμού κλήσης συναρτήσεων, οι τοπικές μεταβλητές password_correct, attempts και buf αποθηκεύονται στη στοίβα, ας υποθέσουμε στις εξής διευθύνσεις (σημειώστε ότι η στοίβα μεγαλώνει *προς τα κάτω*):

Μεταβλητή	Διεύθυνση
password_correct	1000-1003
attempts	996-999
buff	980-995

Αν τώρα ο χρήστης εισάγει ένα συνθηματικό μεγέθους 24 χαρακτήρων, έστω 24 επαναλήψεις του κεφαλαίου λατινικού Α, τα 16 πρώτα Α θα αποθηκευθούν στα όρια του buff, τα 4 επόμενα στη μεταβλητή attempts και τα 4 επόμενα στη μεταβλητή password_correct (επίσης θα αποθηκευθεί και ένα μηδενικό στη θέση 1004 – σε αυτό θα επανέλθουμε στη συνέχεια). Οι δύο ακέραιες μεταβλητές έχοντας κάθε ένα από τα 4 bytes τους να ισούνται με 65 (ο ASCII κωδικός του Α) διερμηνεύονται από τη C ως έχουσες την τιμή 1094795585 – έτσι η συνθήκη της εντολής "if (password_correct)" είναι αληθής και η συνάρτηση επιστρέφει 1!

Αντίστοιχο πρόβλημα θα αντιμετωπίσουμε και με τον κώδικα του ακόλουθου σχήματος:

```
int testPassword(void) {
        char thePass[] = "secret";
        char userPass[7];
        int passlen = strlen(thePass);

        printf("Enter password:");
        gets(userPass);
        if (strncmp(thePass, userPass, passlen) == 0)
                return 1;
        else
                return 0;
}
```

Εδώ αν ο χρήστης εισάγει «aaaaaaaaaaaaaaaa» (πάνω από 14 επαναλήψεις του a) η συνάρτηση και πάλι επιστρέφει 1.

Και στις δύο περιπτώσεις το πρόβλημα δημιουργήθηκε γιατί έχουμε δεσμεύσει κάποιο χώρο για την αποθήκευση της εισόδου τους χρήστη, αλλά η συνάρτηση gets() δεν κάνει οποιονδήποτε έλεγχο για το αν οι περιορισμοί χώρου πληρούνται κατά την αποθήκευση της εισόδου στη μεταβλητή.

Υπερχείλιση μνήμης για τοπικές μεταβλητές

Στα προηγούμενα παραδείγματα η επικάλυψη των δεδομένων της στοίβας σταμάτησε στα όρια των τοπικών μεταβλητών της συνάρτησης. Αυτό δεν είναι πάντα απαραίτητο: αν η υπερχείλιση είναι αρκετά μεγάλη, τότε η επικάλυψη των δεδομένων θα συνεχιστεί και πέρα από αυτά, στον χώρο όπου βρίσκεται η *διεύθυνση επιστροφής* από τη συνάρτηση και τυχόν άλλοι καταχωρητές. Κατά συνέπεια, όταν εκτελεστεί η εντολή *return* της συνάρτησης ο έλεγχος θα μεταφερθεί όχι στη συνάρτηση που την κάλεσε, αλλά όπου υποδεικνύουν τα bytes που έχουν αντικαταστήσει την αρχική διεύθυνση επιστροφής.

Στην «καλή» περίπτωση, τα bytes αυτά θα σχηματίζουν μία άκυρη διεύθυνση, οπότε το πρόγραμμα θα τερματιστεί άμεσα από το λειτουργικό σύστημα, ή κάποια έγκυρη διεύθυνση με τυχαία bytes, οπότε το συνηθέστερο είναι να εκτελούνται λίγες εντολές πριν την οριστική κατάρρευση.. Στην «κακή» περίπτωση τα bytes είναι κατάλληλα επιλεγμένα από τον εισβολέα ώστε:

- είτε να δείχνουν σε κώδικα μηχανής που ο ίδιος έδωσε ως δεδομένα (και έχει αποθηκευθεί στη στοίβα!)
- είτε να καλούν επιλεγμένες κλήσεις συστήματος εφοδιασμένες με κατάλληλες παραμέτρους (τις οποίες ο εισβολέας παρέχει στα *επόμενα bytes* από τη διεύθυνση επιστροφής) – π.χ. exec("sh", "/bin/sh", NULL);

Οι συνηθέστερες αιτίες υπερχείλισης μνήμης

Η γλώσσα C παρέχει αρκετές συναρτήσεις βιβλιοθήκης που είναι επιρρεπείς σε εμφάνιση υπερχείλιση μνήμης. Μερικές από αυτές φαίνονται στον ακόλουθο πίνακα μαζί με σχετικά σχόλια και τους επικρατέστερους «ασφαλείς αντικαταστάτες».

Συνάρτηση	Σχόλια	Αντικαταστάτης
gets(myStr);	Κανένας έλεγχος στο πλήθος των αποθηκευόμενων bytes	fgets(myStr, 80, stdin);
scanf("%s", mystr);	Κανένας έλεγχος στο πλήθος των αποθηκευόμενων bytes	scanf("%80s", mystr);
sprintf(myStr, "%d%s%f", …);	Δεν διασφαλίζεται ότι τα bytes που θα αποθηκευθούν χωράνε στο mystr	snprintf(myStr, 80, "%d%s%f", …); sprintf(myStr, "%5d%.65s%8.2f", …);
strcpy(prayItFits, bigOne);	Δεν διασφαλίζεται ότι το μήκος της bigOne είναι μικρότερο από τη χωρητικότητα του prayItFits	strncpy(prayItFits, bigOne, 80);
strcat(smallStr, possiblyBig);	Δεν διασφαλίζεται ότι το μήκος της bygone *συν το τρέχον μήκος του* prayItFits είναι μικρότερο από τη χωρητικότητα του prayItFits	strncat(smallStr, possiblyBig, 80 – strlen(smallStr));
while ((c = getchar()) != '\n') str[idx++] = c;	Η getchar είναι ασφαλής, αλλά στον βρόχο δεν γίνεται ο κατάλληλος έλεγχος χωρητικότητας	while ((idx < 80) && ((c = getchar()) != '\n')) str[idx++] = c;
char s[256]; getcwd(s, PATH_MAX);	Η getcwd είναι ασφαλής, αλλά έχουμε δεσμεύσει (δυνητικά) λίγο χώρο.	char s[PATH_MAX + 1]; getcwd(s, PATH_MAX);

Υπερχείλιση μνήμης στον σωρό

Η υπερχείλιση μνήμης στον σωρό είναι συνήθως συνυφασμένη με αναποτελεσματικό έλεγχο της υπερχείλισης των πράξεων ακεραίων. Θεωρήστε το ακόλουθο πρόγραμμα:

```
printf("Enter array size: ");
scanf("%d", &numElements);
numBytes = numElements * sizeof(int);
if ((myArray = malloc(numBytes)) == NULL) {
        perror("Out of memory");
        exit(1);
}
for (i = 0; i < numElements; i++)
        myArray[numElements – 1] = 0;
```

που δεσμεύει και αρχικοποιεί έναν πίνακα ακεραίων. Το πρόβλημα με το πρόγραμμα είναι ότι δεν ελέγχει αποτελεσματικά την περίπτωση που το εισαχθέν από το χρήστη πλήθος στοιχείων πολλαπλασιαζόμενο με το μέγεθος του ακεραίου ξεπερνά την μέγιστη τιμή του μη προσημασμένου ακεραίου, οπότε έχουμε ή αναδίπλωση, ή επιστροφή του μέγιστου μη προσημασμένου ακεραίου, ανάλογα με την υλοποίηση. Ο σωστός κώδικας απεικονίζεται στο σχήμα που ακολουθεί:

```
printf("Enter array size: ");
scanf("%d", &numElements);
if (numElements > UINT_MAX / 4) {
        fprintf(stderr, "Array too large!\n");
        exit(1);
}
numBytes = numElements * sizeof(int);
if ((myArray = malloc(numBytes)) == NULL) {
        perror("Out of memory");
        exit(1);
}
for (i = 0; i < numElements; i++)
        myArray[numElements – 1] = 0;
```

Το τελικό αποτέλεσμα από τον πλημμελή έλεγχο ορίων είναι ότι και πάλι μπορεί να επικαλυφθεί η στοίβα ή άλλα δεδομένα στον σωρό.

4.3.1.2 Παράλειψη προσδιοριστή μορφής στην printf

Όταν η printf/fprintf κ.λπ. τυπώνει μία συμβολοσειρά και μόνο, πολλοί προγραμματιστές παραλείπουν το *"%s"* ως προσδιοριστή μορφής και παραθέτουν μόνο τη συμβολοσειρά. Η συμβολοσειρά τυπώνεται στη γενική περίπτωση, αλλά αν περιέχει ενδείξεις εκτύπωσης παραμέτρων (π.χ. %s, %d κ.λπ), η printf θα αναζητήσει τις τιμές τους στη στοίβα. Αντί για τιμές παραμέτρων της printf (που δεν υπάρχουν) εκεί βρίσκονται άλλα δεδομένα, όπως διευθύνσεις επιστροφής, τοπικές μεταβλητές, παράμετροι προς τη συνάρτηση που καλεί την printf κ.ο.κ., με αποτέλεσμα να έχουμε πιθανή διαρροή πληροφοριών. Έτσι, για να τυπώσουμε μία συμβολοσειρά s θα πρέπει *πάντα* να γράφουμε printf("%s", s); (ή να την τυπώνουμε με εναλλακτικό τρόπο, π.χ. puts(s)).

4.3.2 Συνθήκες ανταγωνισμού

Ο όρος *συνθήκες ανταγωνισμού* αναφέρεται στις περιπτώσεις όπου δύο διεργασίες αλληλεπιδρούν με μη προβλεπόμενο τρόπο, κυρίως λόγω χρονισμού ενεργειών που

αφορούν το ίδιο αντικείμενο. Η βασική αιτία πίσω από τις συνθήκες ανταγωνισμού είναι είτε το γεγονός ότι ένα αντικείμενο ελέγχεται σε κάποια στιγμή και χρησιμοποιείται αργότερα ενώ στο μεσοδιάστημα έχει αλλάξει, είτε ότι μία εννοιολογικά αδιαίρετη λειτουργία διασπάται σε μικρότερα κομμάτια, δίνοντας έδαφος για εξωτερικές παρεμβάσεις. Συγκεκριμένες περιπτώσεις συνθηκών ανταγωνισμού μπορεί να έχουν σοβαρές επιπτώσεις στην ασφάλεια του συστήματος.

Μία από τις πρώτες συνθήκες ανταγωνισμού που ανακαλύφθηκαν (και αξιοποιήθηκαν από τους εισβολείς) ήταν η πρώτη υλοποίηση της δημιουργίας καταλόγων στο Unix που γινόταν με τον ακόλουθο κώδικα:

```
mknod(path, S_IFDIR, dev);
/* Υπάρχει ο κατάλογος, ιδιοκτησία του root και είναι κενός */
chown(path, uid, gid); /* Αλλαγή ιδιοκτήτη*/
chdir(path);
link(path, "."); /* Δημιουργία καταχώρησης. */
link(parent, ".."); /* Δημιουργία καταχώρησης .. */}
```

Η αξιοποίηση του κενού ασφάλειας στην υλοποίηση αυτή (που εκτελείται με τα προνόμια του υπερχρήστη) συνίσταται στην εκτέλεση του κάτωθι κώδικα:

```
while true
do
nice -39 mkdir foo &
rm -rf foo; ln /etc/passwd foo
rm -fr foo &
ls -l /etc/passwd
done
```

Ο στόχος του κακόβουλου κώδικα είναι η επίτευξη της εκτέλεσης της γραμμής

```
rm -rf foo; ln /etc/passwd foo
```

μετά την mknod και *πριν* την chown. Το αποτέλεσμα είναι να αλλάξει η ιδιοκτησία του αρχείου /etc/passwd και να περιέλθει στον κακόβουλο χρήστη.

Μια πιο γενική περίπτωση συνθήκης ανταγωνισμού είναι η χρήση της συνάρτησης access για έλεγχο του αν σε προγράμματα παραχώρησης ταυτότητας χρήστη ο *πραγματικός χρήστης* έχει δικαιώματα πρόσβασης σε ένα αρχείο ή όχι· αν διαπιστωθεί ότι ο πραγματικός χρήστης έχει το σχετικό δικαίωμα τότε το πρόγραμμα το ανοίγει. Ο έλεγχος αυτός συνήθως κωδικοποιείται ως ακολούθως:

```
if(access(theFile, R_OK) == 0) {
        /* User has read permission */
        if((fd = open(theFile, O_RDONLY)) < 0){
        perror(theFile);
                exit(-1);
        }
/* print the file */
}
else perror(theFile);
```

Το πρόβλημα με τον κώδικα αυτό συνίσταται ότι τα δικαιώματα *του πραγματικού χρήστη* ελέγχονται με την *access*, ενώ το άνοιγμα πραγματοποιείται με την *open* (με δικαιώματα υπερχρήστη) σε μεταγενέστερο χρονικό σημείο, και είναι έτσι πιθανό στο μεσοδιάστημα το αρχείο να έχει αλλάξει. Ως παράδειγμα, ας θεωρήσουμε την

περίπτωση όπου ο ανωτέρω κώδικας ενσωματώνεται στην εντολή εκτύπωσης lpr, και ο χρήστης εκτελεί σε ένα παράθυρο ένα πρόγραμμα C με το εξής περιεχόμενο:

```
while(1) {
        creat("harmless", 0644);
        unlink("harmless");
        symlink("/etc/shadow", "harmless");
}
```

και σε ένα άλλο παράθυρο την εντολή

```
repeat 10000 lpr harmless
```

με στόχο να επιτευχθεί η εκτέλεση των «unlink("harmless"); symlink("/etc/shadow", "harmless");» αμέσως μετά την access αλλά πριν την open. Λόγω του μεγάλου πλήθους των επαναλήψεων είναι βέβαιο ότι κάποτε το αποτέλεσμα θα επιτευχθεί, και έτσι θα διαβαστεί το αρχείο που κανονικά είναι προστατευμένο. Αντίστοιχες περιστάσεις μπορούν να προκύψουν και με εγγραφή, οπότε έχουμε αλλοίωση αρχείων.

Η αντιμετώπιση των συνθηκών ανταγωνισμού είναι ιδιαίτερα δύσκολη, διότι δεν είναι εύκολο να αναπαραχθούν. Ως γενικά μέτρα αντιμετώπισης ενδείκνυνται:

1. Η κωδικοποίηση των ατομικών λειτουργιών με κατάλληλους μηχανισμούς που εξασφαλίζουν τη μη διαιρετότητά τους π.χ. η δημιουργία καταλόγων έχει πλέον υλοποιηθεί ως κλήση συστήματος (mkdir()).
2. Να προσπελαύνεται το αντικείμενο και να γίνεται ο έλεγχος πάνω στο αντικείμενο που έχει αποκτηθεί, όταν αυτό είναι δυνατόν. Επί παραδείγματι, πολλά συστήματα υποστηρίζουν τη συνάρτηση faccess() που ελέγχει αν ο *πραγματικός χρήστης* έχει δικαιώματα πάνω στο *ανοικτό αρχείο*. Έτσι μία ασφαλής κωδικοποίηση του ελέγχου θα ήταν:

   ```
   if((fd = open(theFile, O_RDONLY)) < 0){
           perror(theFile);
           exit(1);
   }
   if(faccess(fd, R_OK) == 0) {
           /* User has read permission */
   /* print the file */
   }
   else {
           perror(theFile);
           exit(1);
   }
   ```

   Στο απόσπασμα αυτό κώδικα, ακόμη και αν μετακινηθεί το αρχείο δεν αντιμετωπίζεται πρόβλημα διότι ο περιγραφέας αρχείου fd αναφέρεται στο αρχείο που έχει ανοιχθεί.
3. Αποκλεισμός των συμβολικών συνδέσμων, έλεγχος δηλαδή ότι τα αρχεία που δίνονται ως παράμετροι στο πρόγραμμα δεν είναι συμβολικοί σύνδεσμοι. Η λύση αυτή δεν είναι ιδιαίτερα αποτελεσματική διότι (α) καταργεί τη λειτουργικότητα των συμβολικών συνδέσμων για συγκεκριμένες λειτουργίες αίροντας την ομοιομορφία χειρισμού τους (β) μπορεί να ξεπεραστεί χρησιμοποιώντας «κανονικούς» συνδέσμους και (γ) δεν υποστηρίζεται από όλα τα συστήματα.

4. Μετάβαση προσωρινά σε κατάσταση ελαττωμένων δικαιωμάτων για το άνοιγμα του αρχείου. Το χαρακτηριστικό αυτό εισήχθη στο πρότυπο POSIX μετά την πρώτη του έκδοση, οπότε ΄δεν υποστηρίζεται από κάποια συστήματα. Βάσει αυτής της λύσης, η εφαρμογή θέτει ως ενεργό ταυτότητα χρήστη την *πραγματική ταυτότητα χρήστη* προκειμένου για να ανοίξει το αρχείο και στη συνέχεια επανέρχεται σε καθεστώς αυξημένων προνομίων. Η λύση σκιαγραφείται στο ακόλουθο απόσπασμα κώδικα. Τα συστήματα που υποστηρίζουν τη λειτουργικότητα αυτή έχουν ορισμένη τη σταθερά _POSIX_SAVED_IDS κατά τη μεταγλώττιση.

```
uid_t euid, ruid;
gid_t egid, rgid;

euid = geteuid(); /* Αποθήκευση τρέχουσας ενεργού ταυτότητας χρήστη */
egid = getegid(); /* Αποθήκευση τρέχουσας ενεργού ταυτότητας ομάδας */
ruid = getuid(); /* Αποθήκευση τρέχουσας πραγματικής ταυτότητας χρήστη */
rgid = getgid(); /* Αποθήκευση τρέχουσας πραγματικής ταυτότητας ομάδας */

if(setegid(rgid) < 0) /* Η πραγματική ταυτότητα χρήστη τίθεται ως ενεργός */
        exit(1);
if(seteuid(ruid) < 0) /* Η πραγματική ταυτότητα χρήστη τίθεται ως ενεργός */
        exit(1);

open("...", ...);

if(setegid(egid) < 0) /* Αποκατάσταση αρχικής ενεργού ταυτότητας ομάδας */
        exit(1);
if(seteuid(euid) < 0) /* Αποκατάσταση αρχικής ενεργού ταυτότητας χρήστη */
        exit(1);
```

5. Δημιουργία μιας θυγατρικής διεργασίας η οποία απεκδύεται τα πρόσθετα προνόμια και ανοίγει το αρχείο με την ταυτότητα του χρήστη. Στη συνέχεια η διεργασία αυτή μεταβιβάζει στη γονική της τον *προσδιοριστή αρχείου* που έχει ανοίξει.

4.3.3 Προσωρινά αρχεία

Τα προσωρινά αρχεία είναι εξαιρετικά ύποπτα για διάφορα προβλήματα, συμπεριλαμβάνοντας δυσλειτουργία προγραμμάτων, διαρροή πληροφοριών κ.ο.κ.

Το πρώτο πρόβλημα που αρχικά αντιμετωπίσθηκε ήταν ότι τα προσωρινά αρχεία δημιουργούνταν σε καταλόγους που ήταν εγγράψιμοι από όλους (/tmp, /var/tmp). Βάσει της αρχικής σημασιολογίας του Unix, αν ένας χρήστης είχε δικαίωμα εγγραφής σε έναν κατάλογο, μπορούσε να διαγράψει οποιοδήποτε αρχείο εντός αυτού, ανεξαρτήτως δικαιωμάτων και ιδιοκτησίας του αρχείου. Έτσι, οποιοσδήποτε χρήστης θα μπορούσε να διαγράψει το προσωρινό αρχείο μίας διεργασίας δημιουργώντας της προβλήματα, ή να το αντικαταστήσει με κάποιο που θα μπορούσε ο ίδιος να διαβάσει, προκειμένου να υποκλέψει δεδομένα. Το ζήτημα αυτό αντιμετωπίσθηκε με τον εμπλουτισμό της σημασιολογίας των δικαιωμάτων καταλόγων, στον οποίο προστέθηκε ο χειρισμός του «save text» bit. Όταν το bit αυτό είναι ενεργό *για κατάλογο* και ο χρήστης έχει δικαίωμα εγγραφής σ' αυτόν, επιτρέπεται η δημιουργία νέων αρχείων στον κατάλογο, αλλά η διαγραφή περιορίζεται στα αρχεία των οποίων έχει την ιδιοκτησία (εκτός φυσικά αν είναι ο υπερχρήστης).

Ένα δεύτερο ζήτημα με τα προσωρινά αρχεία είναι ότι η διαδικασία δημιουργίας τους συνήθως περιλαμβάνει τη γέννηση ενός μοναδικού ονόματος αρχείου και εν συνεχεία τη δημιουργία του ίδιου του αρχείου:

```
char *tmp_name;
int tmpfd;
tmp_name = tmpnam(NULL);
if( (tmpfd = open(tmp_name, O_RDWR | O_CREAT | 0600)) < 0)
        exit(1);
unlink(tmp_name); /* Αδύνατη η αναφορά στο αρχείο, αυτόματη
διαγραφή του όταν τερματίσει το πρόγραμμα */
```

Εδώ βλέπουμε ότι τίθεται ξανά ένα θέμα συνθήκης ανταγωνισμού, καθώς το όνομα tmp_nam είναι *σίγουρα μοναδικό* κατά την παραγωγή του (όταν δηλ. εκτελείται η συνάρτηση tmpnam), αλλά δεν υπάρχει η εγγύηση ότι δεν έχει δημιουργηθεί μέχρι να εκτελεσθεί η open().

Μία λύση στο ζήτημα αυτό θα ήταν η χρήση της mkstemp, η οποία δημιουργεί *και ανοίγει* ένα προσωρινό αρχείο ως ατομική ενέργεια.

```
fd = mkstemp("/var/tmp/atempFile.XXXXXX");
fchmod(fd, 0600);
```

Η εντολή fchmod επιχειρεί να εξασφαλίσει ότι κανείς άλλος δεν έχει δικαίωμα πρόσβασης στο αρχείο, αλλά δυστυχώς ελλοχεύει ο κίνδυνος να προλάβει ο επιτιθέμενος να ανοίξει το αρχείο *αμέσως μετά* την mkstemp αλλά *πριν* την fchmod (οι περισσότερες υλοποιήσεις της mkstemp δημιουργούσαν αρχείο με δικαιώματα 0666 – οι νεώτερες λαμβάνουν υπόψη το umask που σπανίως όμως έχει κατάλληλη τιμή π.χ. 077). Στην αξιοποίηση του umask στηρίζεται έτσι η τρίτη λύση:

```
umask(066);
fd = mkstemp("/var/tmp/atempFile.XXXXXX");
```

Με τη ρύθμιση αυτή το νέο αρχείο θα έχει εξ αρχής δικαιώματα πρόσβασης που θα απαγορεύουν στους χρήστες της ομάδας και τους λοιπούς χρήστες οποιαδήποτε πρόσβαση. Μία εναλλακτική λύση είναι η αξιοποίηση της ένδειξης O_EXCL στην κλήση συστήματος open για τη δημιουργία αρχείου, η οποία υποδεικνύει ότι το άνοιγμα του αρχείου πρέπει να αποτύχει *αν το αρχείο ήδη υπάρχει*. Ο κώδικας έτσι διαμορφώνεται ως ακολούθως:

```
char *tmp_name;
int tmpfd;
tmp_name = tmpnam(NULL);
if( (tmpfd = open(tmp_name, O_RDWR | O_CREAT | O_EXCL | 0600)) < 0)
        exit(1);
unlink(tmp_name); /* Αδύνατη η αναφορά στο αρχείο, αυτόματη διαγραφή του
όταν τερματίσει το πρόγραμμα */
```

4.3.4 Μεταβλητές περιβάλλοντος

Κάθε πρόγραμμα έχει μεταβλητές περιβάλλοντος που περιέχουν συγκεκριμένες πληροφορίες, όπως π.χ. το όνομα του χρήστη, τη διαδρομή αναζήτησης εντολών, διαδρομή αναζήτησης βιβλιοθηκών κ.λπ. Οι μεταβλητές είναι προσπελάσιμες μέσω της μεταβλητής environ, η οποία είναι ένας πίνακας συμβολοσειρών με την ακόλουθη μορφή:

environ[0] = "USER=thomas";
environ[1] = "PATH=/usr/bin:/sbin:/usr/local/bin";
environ[2] = "LD_LIBRARY_PATH=/oracle/lib:/usr/lib";

Τα προγράμματα (ειδικότερα τα πιο προνομιούχα) ΔΕΝ πρέπει να βασίζονται στις τιμές αυτές, καθώς μπορούν να τροποποιηθούν από τους χρήστες π.χ.

USER=root; export USER

Αντί για τις μεταβλητές περιβάλλοντος θα πρέπει τα προγράμματα να βασίζονται σε *κλήσεις συστήματος* οι οποίες είναι αξιόπιστες. Έτσι η ακολουθία κλήσεων συστήματος

getpwuid(getuid())->pw_name

θα μας δώσει εγγυημένα αξιόπιστα αποτελέσματα, σε αντίθεση με την επισφαλή

getenv("USER");

Επίσης πρέπει να δίνεται ιδιαίτερη μέριμνα σε ειδικές μεταβλητές περιβάλλοντος όπως η PATH που ορίζει τους καταλόγους όπου αναζητούνται εντολές και η LD_LIBRARY_PATH που ορίζει τους καταλόγους όπου αναζητούνται βιβλιοθήκες, ειδικότερα κατά την εκτέλεση προνομιούχων προγραμμάτων. Για τον σκοπό αυτό τα περισσότερα συστήματα παρέχουν τη ρύθμιση SUPATH η οποία θέτει αυτόματα τη μεταβλητή PATH για προνομιούχα προγράμματα. Ο διαχειριστής του συστήματος οφείλει να εξασφαλίσει ότι μόνο αξιόπιστοι κατάλογοι μνημονεύονται στη μεταβλητή αυτή – ειδικότερα ο τρέχον κατάλογος (.) *δεν* πρέπει να μνημονεύεται. Επίσης, ο *φορτωτής δυναμικών βιβλιοθηκών* του συστήματος πρέπει να *αγνοεί* επανορισμούς κλήσεων συστήματος ή βασικών διαδικασιών βιβλιοθήκης που βρίσκονται σε μη «εγκεκριμένους» καταλόγους, ανεξάρτητα με τη διάταξη των καταλόγων στη μεταβλητή LD_LIBRARY_PATH. Για παράδειγμα, αν ο χρήστης δώσει την τιμή

LD_LIBRARY_PATH=/home/pika/libs

όπου υπάρχει η βιβλιοθήκη *mylib.so* που περιέχει την υλοποίηση συνάρτησης

uid_t geteuid(void) {return 0;}

ο φορτωτής πρέπει να την αγνοήσει και να χρησιμοποιήσει κανονικά την υλοποίηση που παρέχεται στη στάνταρ βιβλιοθήκη της C.

4.3.5 Περιορισμός πόρων

Η C και η C++ μας δίνουν τη δυνατότητα να περιορίσουμε ποσοτικά τους πόρους στους οποίους έχει πρόσβαση το πρόγραμμά μας. Οι περιορισμοί αυτοί μπορούν να συμβάλλουν σε δύο κατευθύνσεις αναφορικά με την επαύξηση της ασφάλειας:

1. *αποτροπή ή περιορισμός των επιπτώσεων επιθέσεων τύπου άρνησης παροχής υπηρεσιών*. Μία τέτοια επίθεση έχει ως στόχο συνήθως μία διεργασία, η οποία καταλαμβάνει όλους τους πόρους του συστήματος (μνήμη, πίνακας διεργασιών, πίνακας ανοικτών αρχείων κ.λπ.) για την εξυπηρέτηση ανούσιων αιτημάτων, καθιστώντας έτσι το σύστημα μη λειτουργικό. Περιορίζοντας τους πόρους στους οποίους έχει πρόσβαση η κάθε διεργασία, η οποιαδήποτε τέτοια επίθεση θα έχει περιορισμένα αποτελέσματα, μια και το λειτουργικό σύστημα δεν θα επιτρέψει στη διεργασία να καταλάβει όλους τους πόρους του συστήματος.

2. *αποτροπή διαρροής πληροφοριών*. Σε πολλά συστήματα όταν τερματίζεται κατά «μη κανονικό τρόπο» μία διεργασία δημιουργείται ένα *αρχείο αποτύπωσης μνήμης*, που στόχος του είναι να διευκολύνει τον εντοπισμό σφαλμάτων. Το αρχείο όμως αυτό δυνητικά περιέχει ευαίσθητες πληροφορίες π.χ. συνθηματικά, που δεν πρέπει να διαρρεύσουν. Ένας κακόβουλος χρήστης θα μπορούσε για παράδειγμα να προσπαθήσει να υποκλέψει το αρχείο κρυπτογραφημένων συνθηματικών προκαλώντας τον απότομο τερματισμό της εντολής passwd:

   passwd → pid = 2078
   kill –ABRT 2078 → stop and core dump
   strings core | grep root → extract root entry

   Η πιο αποτελεσματική άμυνα είναι να μην δημιουργείται καθόλου το αρχείο αυτό, περιορίζοντας το μέγιστο μέγεθός του στο μηδέν.

5 Προστασία πόρων στο διαδίκτυο

Οι περισσότεροι υπολογιστές στην σημερινή εποχή είναι συνδεδεμένοι στο διαδίκτυο, προκειμένου να παράσχουν στους χρήστες τους τη δυνατότητα να προσπελαύνουν πληροφορίες ή να επικοινωνούν με άλλους χρήστες του διαδικτύου. Η σύνδεσή αυτή όμως παρέχει σε κακόβουλους χρήστες του διαδικτύου τη δυνατότητα να επιτεθούν στους υπολογιστές αυτούς, προσπαθώντας να παραβιάσουν την ασφάλειά τους και ως εκ τούτου πρέπει να ληφθούν μέτρα για την προστασία τους.

Μία προσέγγιση σ' αυτό το ζήτημα είναι να εγκατασταθούν *σε κάθε έναν από τους υπολογιστές που μας αφορούν* (π.χ. τους υπολογιστές ενός εταιρικού δικτύου) μέτρα προστασίας, συμπεριλαμβανομένων των επιδιορθωτικών προγραμμάτων που παρέχουν οι κατασκευαστές και μηχανισμών περιορισμού της πρόσβασης για χρήστες που δεν βρίσκονται εντός του εταιρικού δικτύου (οι χρήστες εντός του εταιρικού δικτύου θεωρούνται έμπιστοι και επίσης θα πρέπει να έχουν πρόσβαση σε συγκεκριμένες υπηρεσίες που παρέχονται από τους εξυπηρέτες του). Η προσέγγιση αυτή ωστόσο έχει δύο σημαντικά μειονεκτήματα:

1. Η εγκατάσταση και συντήρηση μηχανισμών προστασίας σε ένα μεγάλο δίκτυο είναι μία ανιαρή, χρονοβόρα και επιρρεπής σε λάθη διαδικασία, ειδικότερα όταν έχουμε να κάνουμε με ετερογενή συστήματα όπου διαφορετικοί μηχανισμοί πρέπει να εγκατασταθούν σε διαφορετικούς υπολογιστές. Με τον τρόπο αυτό αυξάνεται η πιθανότητα κάποιος ή κάποιοι υπολογιστές του εταιρικού δικτύου να έχουν ασφάλεια υποδεέστερη των προδιαγραφών.
2. Αν οι εισβολείς αποκτήσουν πρόσβαση σε κάποιον υπολογιστή του εταιρικού δικτύου (κάποιον που έχει προστατευθεί λιγότερο καλά από τους άλλους), μπορούν να τον χρησιμοποιήσουν ως ορμητήριο για νέες επιθέσεις προς άλλους υπολογιστές του εταιρικού δικτύου. Ο υπολογιστής αυτός, ως μέλος του εταιρικού δικτύου θεωρείται αξιόπιστος και έτσι η πρόσβασή του σε άλλους υπολογιστές του εταιρικού δικτύου δεν παρεμποδίζεται.

Μια διαφορετική προσέγγιση είναι αντί να προστατεύεται ο κάθε υπολογιστής ξεχωριστά να προστατεύεται το σύνολο του εταιρικού δικτύου με την τοποθέτηση συστημάτων ταυτοποίησης-εξουσιοδότησης στα όρια του εταιρικού δικτύου, οι οποίοι ονομάζονται *firewalls*. Ο στόχος του firewall είναι να προστατεύσει το δίκτυο που βρίσκεται «πίσω» από αυτό (το εταιρικό δίκτυο δηλαδή), ελέγχοντας τη δικτυακή

κυκλοφορία και περιορίζοντας τις επιτρεπόμενες ροές κυκλοφορίας. Τα firewalls επιτυγχάνουν αποτελεσματικούς ελέγχους πρόσβασης, «φιλτράροντας» τις ανεπιθύμητες ροές πληροφορίας, περιορίζοντας τη χρήση ευάλωτων πρωτοκόλλων, παρακολουθώντας συνολικά τη λειτουργία του δικτύου και κεντρικοποιώντας τον έλεγχο πρόσβασης στο εσωτερικό δίκτυο από τους υπολογιστές του διαδικτύου. Το firewall γενικώς συντίθεται από τα ακόλουθα τμήματα υλικού και λογισμικού:

- *Δρομολογητές με μηχανισμούς φιλτραρίσματος πακέτων*. Οι δρομολογητές αυτοί τοποθετούνται στα όρια του εταιρικού δικτύου και έχουν ως στόχο να αποτρέπουν την επικοινωνία των χρηστών του διαδικτύου με συγκεκριμένους υπολογιστές ή υπηρεσίες εντός του εταιρικού δικτύου.
- *Εξυπηρέτες με ρόλο αντιπροσώπευσης υπηρεσιών*. Οι εξυπηρέτες αυτοί εμφανίζονται ως μία γραμμή άμυνας πριν τους εξυπηρέτες του εταιρικού δικτύου που προσφέρουν μία υπηρεσία που πρέπει να είναι προσπελάσιμη στους χρήστες του διαδικτύου.
- *Εξυπηρέτες με ρόλο προμαχώνα*. Οι εξυπηρέτες αυτοί τοποθετούνται μεταξύ του διαδικτύου και του εταιρικού δικτύου προκειμένου να ελέγχουν την πρόσβαση σε επισφαλείς υπηρεσίες, είτε από τους εσωτερικούς χρήστες προς το διαδίκτυο είτε από χρήστες του διαδικτύου προς το εταιρικό δίκτυο. Οι εξυπηρέτες αυτοί είναι «θωρακισμένοι» από άποψη ασφάλειας (εξ ου και το όνομά τους).

Προκειμένου να είναι αποτελεσματικός ένας μηχανισμός firewall πρέπει να συντρέχουν οι κάτωθι προϋποθέσεις:

1. *Όλη η δικτυακή κυκλοφορία διέρχεται μέσω του μέσω του firewall*. Αν υπάρχουν εναλλακτικές διοδεύσεις πληροφορίας μεταξύ του διαδικτύου και του εταιρικού δικτύου με ελαττωμένη ασφάλεια, όσο αποτελεσματικό και αν είναι το firewall δεν θα προσφέρει καμία ασφάλεια, καθώς οι εισβολείς θα προτιμήσουν την πιο εύκολη οδό.
2. *Μόνο η κυκλοφορία που επιτρέπεται από την πολιτική ασφάλειας διεκπεραιώνεται τελικά*. Με άλλα λόγια το firewall θα πρέπει να είναι σωστά ρυθμισμένο και να ανταποκρίνεται στις προδιαγραφές του.
3. *Το ίδιο το firewall είναι άτρωτο*. Αν οι εισβολείς «καταλάβουν» το firewall έχουν ανοικτό το δρόμο προς το εταιρικό δίκτυο.

Για τα firewalls υπάρχουν δύο γενικοί σχεδιασμοί:

- επιτρέπονται όσα δεν απαγορεύονται ρητώς
- απαγορεύονται όσα δεν επιτρέπονται ρητώς

Η πρώτη φιλοσοφία είναι η πιο ελαστική (και επισφαλής), καθώς οι διαχειριστές πρέπει να ρυθμίζουν το firewall ώστε να απαγορεύει τις προσβάσεις που είναι εγνωσμένα επικίνδυνες. Καθώς ο κατάλογος αυτός είναι μακροσκελής, υπάρχει σημαντική πιθανότητα κάποια πρόσβαση να μην περιληφθεί στη λίστα απαγορεύσεων και έτσι να μην επιτυγχάνεται το επιθυμητό επίπεδο ασφάλειας.

Η δεύτερη φιλοσοφία είναι πιο αυστηρή και ασφαλής, μια και οι διαχειριστές πρέπει να ορίζουν μόνο τις ροές πληροφορίας που τελικά επιτρέπονται. Ο κατάλογος των ροών αυτών είναι συνήθως μικρότερος, άρα πιο διαχειρίσιμος, ενώ αν παραληφθεί μία επιτρεπτή πρόσβαση από τις ρυθμίσεις κάποιος από τους νόμιμους χρήστες θα το επισημάνει στους διαχειριστές, καθώς δεν θα μπορεί να

επιτελέσει κάποια εργασία. Αντίθετα, στην πρώτη φιλοσοφία ο μέσος χρήστης του διαδικτύου δεν θα ειδοποιήσει τους διαχειριστές ότι έχουν επιτρέψει κάποιες προσβάσεις παραπάνω απ' όσες θα έπρεπε.

Φυσικά η εγκατάσταση ενός firewall *ΔΕΝ* λύνει συνολικά το πρόβλημα ασφάλειας ενός εταιρικού δικτύου. Πρώτα απ' όλα δεν αντιμετωπίζει το ζήτημα του *περιεχομένου* της επικοινωνίας και έτσι π.χ. ένα μήνυμα ηλεκτρονικού ταχυδρομείου που διέρχεται νομότυπα από το firewall μπορεί να περιέχει ιούς. Δεύτερον, δεν παρέχουν καμία προστασία για επιθέσεις εκ των έσω ή από κανάλια διαρροής, καθώς ο στόχος τους είναι εντελώς διαφορετικός. Τρίτον, σε μεγάλα δίκτυα με μεγάλη εισερχόμενη και εξερχόμενη κυκλοφορία μπορεί να αποτελέσουν σημεία συμφόρησης, καθώς θα πρέπει να ελέγχουν χιλιάδες πακέτα δικτύου στη μονάδα του χρόνου. Τέλος, αν χρησιμοποιούνται ως μοναδικός μηχανισμός άμυνας, τότε η κατάρρευσή τους θα αφήσει όλο το εταιρικό δίκτυο απόλυτα εκτεθειμένο.

5.1 Φιλτράρισμα πακέτων

Η πρώτη βασική λειτουργία των firewalls είναι να «φιλτράρουν» τα δικτυακά πακέτα, επιτρέποντας επιλεκτικά την πρόσβαση σε συγκεκριμένους μόνο υπολογιστές και υπηρεσίες. Ας θεωρήσουμε ένα εταιρικό δίκτυο που διαθέτει τους ακόλουθους εξυπηρέτες: WWW, ηλεκτρονικού ταχυδρομείου, μεταφοράς αρχείων (FTP), εκτυπώσεων, αρχείων για Windows (CIFS) και αρχείων για UNIX (NFS). Πέρα από τους εξυπηρέτες, υπάρχουν και αρκετοί εξυπηρετούμενοι για χρήση του προσωπικού της εταιρίας. Σε ένα τέτοιο περιβάλλον θα μπορούσαμε να περιμένουμε ότι ισχύουν οι κάτωθι κανόνες πρόσβασης:

1. Οι εξωτερικοί ως προς το εταιρικό δίκτυο χρήστες μπορούν να προσπελάσουν:
    a. Τον εξυπηρέτη WWW στη θύρα 80, όπου προσφέρεται η υπηρεσία WWW
    b. Τον εξυπηρέτη ηλεκτρονικού ταχυδρομείου στη θύρα 25, όπου παρέχεται η υπηρεσία διακίνησης μηνυμάτων.
    c. Τον εξυπηρέτη μεταφοράς αρχείων στις θύρες 20 και 21.

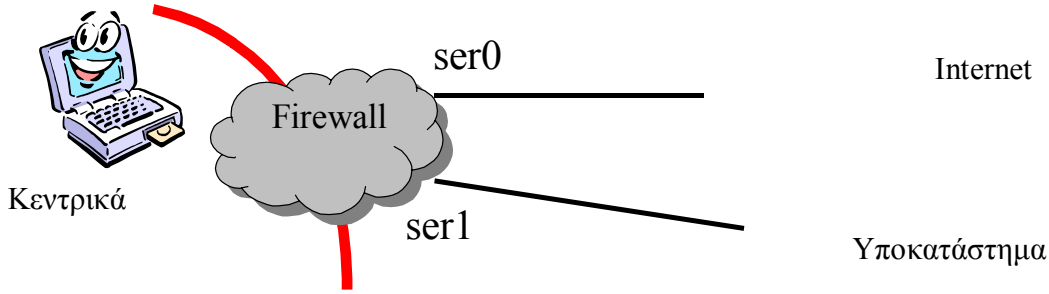
    Οι εξωτερικοί ως προς το εταιρικό δίκτυο χρήστες δεν μπορούν να προσπελάσουν *καθόλου* τους υπόλοιπους υπολογιστές (συμπεριλαμβανομένων των εξυπηρετών εκτυπώσεων, αρχείων για Windows και αρχείων για UNIX) ούτε τις υπόλοιπες θύρες εξυπηρέτησης στους τρεις ανωτέρω εξυπηρέτες.
2. Οι εσωτερικοί χρήστες μπορούν απρόσκοπτα να προσπελάσουν όλες τις υπηρεσίες σε όλους τους υπολογιστές του εταιρικού δικτύου και όλες τις υπηρεσίες σε υπολογιστές του διαδικτύου. (Κατά περίπτωση βέβαια, είναι δυνατόν να περιορισθεί και η πρόσβαση των εσωτερικών χρηστών στο διαδίκτυο είτε για λόγους ασφάλειας είτε για λόγους παραγωγικότητας.)

Προκειμένου να μπορέσει το firewall να περιορίσει την κυκλοφορία των πακέτων στο δίκτυο έχει στη διάθεσή του *κανόνες* οι οποίοι εξετάζουν διάφορα χαρακτηριστικά των δικτυακών πακέτων και υποδεικνύουν αν το πακέτο πρέπει να προωθηθεί στον προορισμό του ή να απορριφθεί. Οι κανόνες των firewalls εξετάζουν συνήθως τα ακόλουθα χαρακτηριστικά των πακέτων δικτύου:

- *Διεύθυνση αφετηρίας (IP, Port)*. Η διεύθυνση IP μπορεί να αντιστοιχεί σε ένα σύνολο υπολογιστών, καθορίζοντας έτσι ένα ολόκληρο *υποδίκτυο* αντί για κάποιον μεμονωμένο υπολογιστή. Επίσης, το Port μπορεί να καθορίζει μία περιοχή θυρών σύνδεσης αντί για μία συγκεκριμένη.
- *Διεύθυνση προορισμού (IP, Port)*. Όμοια με τη διεύθυνση αφετηρίας τα τμήματα IP και Port μπορεί να καθορίζουν ένα υποδίκτυο ή μια περιοχή θυρών σύνδεσης, αντίστοιχα.
- *Φυσική συσκευή λήψης του πακέτου*. Σε δρομολογητές που έχουν πάνω από μία δικτυακές διασυνδέσεις, η φυσική συσκευή λήψης του πακέτου εξετάζεται σε συνδυασμό με την αναγραφόμενη στο πακέτο διεύθυνση αφετηρίας, προκειμένου να εντοπισθούν πακέτα με *πλαστογραφημένη διεύθυνση αφετηρίας*. Η πιο συνηθισμένη περίπτωση τέτοιων πακέτων αφορά πακέτα που προέρχονται στην πραγματικότητα από το διαδίκτυο (συνεπώς λαμβάνονται από τη φυσική συσκευή που συνδέει το εταιρικό δίκτυο με το διαδίκτυο) αλλά αναφέρουν ψευδώς ως διεύθυνση αφετηρίας κάποια εσωτερική διεύθυνση του διαδικτύου, προκειμένου να τύχουν καλύτερης μεταχείρισης.
- *Πρωτόκολλο ανώτερου επιπέδου (TCP/UDP)*.
- *Δικτυακές ενδείξεις* (π.χ. αίτηση εγκαθίδρυσης σύνδεσης)

Αν τα χαρακτηριστικά του πακέτου του δικτύου ταιριάζουν με αυτά που περιγράφονται στον κανόνα, τότε το πακέτο τυγχάνει της μεταχείρισης που προσδιορίζει ο κανόνας (προώθηση ή απόρριψη).

Οι (απλοποιημένοι) κανόνες που θα μπορούσαν να ισχύουν για ένα firewall μιας τράπεζας (βλ. σχήμα) όπου το κεντρικό κατάστημα διαθέτει έναν δρομολογητή με δύο εξωτερικές συνδέσεις, μία προς το Internet και μία άλλη προς ένα υποκατάστημα έχουν ως εξής:



Φυσική συσκευή	Αφετηρία	Προορισμός	Πρωτόκολλο	Ενδείξεις	Ενέργεια
ser0	branch1/*	*/*	*	*	Deny
*	*/*	mailsrv/25	TCP	*	Allow
*	*	namesrv/143	TCP/UDP	*	Allow
ser1	branch1/*	oraclesrv/1525	TCP	*	Allow
*	*	*	*	*	Deny

Ο πρώτος κανόνας παρέχει προστασία έναντι πλαστογραφημένων πακέτων που έρχονται στην πραγματικότητα από το Internet αλλά εμφανίζονται να έχουν

διεύθυνση που αντιστοιχεί στο υποκατάστημα. Οι δύο επόμενες γραμμές επιτρέπουν σε όλους την πρόσβαση στον εξυπηρέτη ηλεκτρονικού ταχυδρομείου και στον εξυπηρέτη ονοματολογίας, ενώ η τέταρτη γραμμή δίνει στους υπολογιστές του υποκαταστήματος δικαίωμα πρόσβασης στον εξυπηρέτη βάσεων δεδομένων των κεντρικών. Η τελευταία γραμμή φροντίζει για την απόρριψη όλων των πακέτων που δεν ταιριάζουν με κάποιον από τους προηγούμενους κανόνες, υλοποιώντας ουσιαστικά την πολιτική «ό,τι δεν επιτρέπεται ρητώς, απαγορεύεται».

Το φιλτράρισμα πακέτων παρουσιάζει τα εξής σημαντικά πλεονεκτήματα:

1. Είναι πολύ απλό στην υλοποίηση
2. Ενσωματωμένο σε δικτυακές διατάξεις, όπως δρομολογητές, παρέχει εξαιρετικές επιδόσεις.

Από την άλλη πλευρά ωστόσο, το φιλτράρισμα πακέτων δεν δίνει τη δυνατότητα τήρησης αρχείων καταγραφής (τουλάχιστον όχι χωρίς σημαντικές επιπτώσεις στην απόδοση), είναι δύσκολο να ρυθμιστεί και να ελεγχθεί, –ιδίως για πολύπλοκες περιπτώσεις, ενώ δεν είναι ιδιαίτερα ευέλικτο ή άμεσα επεκτάσιμο. Τέλος μπορεί να παρακαμφθεί μέσω της τεχνικής που ονομάζεται «tunneling», δηλαδή της μεταφοράς πακέτων μιας απαγορευμένης υπηρεσίας (π.χ. *telnet*) μέσα από πακέτα μιας επιτρεπόμενης υπηρεσίας (π.χ. WWW). Η τεχνική αυτή απαιτεί και τη χρήση ειδικού λογισμικού στα δύο άκρα της επικοινωνίας, το οποίο όμως είναι εύκολο να βρεθεί και να εγκατασταθεί.

Στο σχήμα που ακολουθεί παρουσιάζεται ένα παράδειγμα κανόνων φιλτραρίσματος, βασισμένο στο ευρέως διαδεδομένο πακέτο του Linux, iptables

```
iptables -s 10.30.19.0/24 –i ! eth0 –j DENY –l
iptables -s 195.134.65.128/26 –i ! eth2 –j DENY –l

iptables -A forward -s 10.30.19.0/24 -i eth0 –o eth2 -j good-dmz
iptables -A forward -s 10.30.19.0/24 -i eth0 –o eth1 -j good-bad
iptables -A forward -s 195.134.65.128/26 -i eth2 –o eth1 -j dmz-bad
iptables -A forward -s 195.134.65.128/26 -i eth2 –o eth0 -j dmz-good
iptables -A forward -i eth0 -j bad-dmz
iptables -A forward -i eth2 -j bad-good
iptables -A forward -j DENY -l

iptables -A good-dmz -p tcp -d 195.134.65.183 smtp -j ACCEPT
iptables -A good-dmz -p tcp -d 195.134.65.183 pop3 -j ACCEPT
iptables -A good-dmz -j DENY -l

iptables -A dmz-good -p tcp ! -y -s 195.134.65.183 smtp -j ACCEPT
iptables -A dmz-good -p tcp ! -y -s 195.134.65.183 pop3 -j ACCEPT
iptables -A dmz-good -j DENY –l

iptables -A bad-dmz -p tcp -d 195.134.65.183 smtp -j ACCEPT
iptables -A bad-dmz -p tcp ! -y --sport 25 -d 195.134.65.183 -j ACCEPT
iptables -A bad-dmz -j DENY

iptables -A dmz-bad -p tcp -s 195.134.65.183 --dport smtp -j ACCEPT
iptables -A dmz-bad -p tcp ! –y -s 195.134.65.183 smtp -j ACCEPT
iptables -A dmz-bad -j DENY –l

iptables -A good-bad -p tcp --dport www -j MASQ
iptables –A good-bad -j DENY –l

iptables –A bad-good -j DENY
```

Η πρώτη ομάδα εντολών ασχολείται με την προστασία από πακέτα με παραποιημένες διευθύνσεις αφετηρίας: διευθύνσεις αφετηρίας της μορφής 10.30.19.0/24 μπορούν να

εμφανίζονται *μόνο* στη συσκευή *eth0*, ενώ διευθύνσεις αφετηρίας της μορφής 195.134.65.128/26 μπορούν να εμφανίζονται *μόνο* στη συσκευή *eth2*.

Οι έξι πρώτες εντολές της δεύτερης ομάδας διαχωρίζουν τα πακέτα που πρόκειται να δρομολογηθούν σε έξι κατηγορίες, ανάλογα με (α) τη διεύθυνση αφετηρίας (-s) και τη φυσική συσκευή προς την οποία θα προωθηθούν (-i). Για κάθε μία από αυτές, ορίζεται το σύνολο κανόνων που θα εφαρμοσθεί (good-dmz, good-bad κ.λπ.). Η τελευταία εντολή της δεύτερης ομάδας ορίζει ότι απορρίπτονται όλα τα υπόλοιπα υπό προώθηση πακέτα.

Η τρίτη ομάδα εντολών ρυθμίζει την κυκλοφορία που δρομολογείται από τη φυσική συσκευή eth0 προς τη φυσική συσκευή eth2 ($1^{η}$ εντολή $2^{ης}$ ομάδας). Ο πρώτος κανόνας είναι ότι επιτρέπεται η δρομολόγηση αν η διεύθυνση προορισμού είναι η 195.134.65.183 και η θύρα προορισμού είναι η smtp (25 - αποστολή μηνυμάτων ηλ. ταχυδρομείου). Ο δεύτερος κανόνας είναι ότι επιτρέπεται η δρομολόγηση αν η διεύθυνση προορισμού είναι η 195.134.65.183 και η θύρα προορισμού είναι η pop3 (110 – ανάγνωση μηνυμάτων ηλεκτρονικού ταχυδρομείου με το πρωτόκολλο Post Office Protocol έκδοση 3). Ο τρίτος κανόνας της ομάδας υποδεικνύει ότι οποιαδήποτε άλλη κυκλοφορία απαγορεύεται, υλοποιώντας έτσι την πολιτική «ό,τι δεν επιτρέπεται ρητώς, απαγορεύεται».

Η τέταρτη ομάδα εντολών ρυθμίζει την κυκλοφορία που δρομολογείται από τη φυσική συσκευή eth2 προς τη φυσική συσκευή eth0. Ο πρώτος κανόνας ορίζει ότι επιτρέπεται η δρομολόγηση πακέτων που αποστέλλονται από τη θύρα smtp του υπολογιστή 195.134.65.183 *μόνο εφ' όσον δεν πρόκειται για πακέτα εγκαθίδρυσης σύνδεσης* (! –y), επιτρέποντας έτσι ουσιαστικά μόνο στον εξυπηρέτη ηλεκτρονικού ταχυδρομείου να απαντήσει σε πακέτα που έχει δεχθεί λόγω του $1^{ου}$ κανόνα της δεύτερης ομάδας. Ο ίδιος ο εξυπηρέτης ηλεκτρονικού ταχυδρομείου, δεν μπορεί να αιτηθεί σύνδεση προς υπολογιστή πίσω από τη δικτυακή συσκευή eth0 (αυτό δεν περιλαμβάνεται στην κανονική λειτουργία του).

Η έβδομη ομάδα εντολών ρυθμίζει την κυκλοφορία που δρομολογείται από τη φυσική συσκευή eth0 προς τη φυσική συσκευή eth1. Η πρώτη εντολή της ομάδας αυτής ορίζει ότι αιτήσεις για τέτοια δρομολόγηση που προορίζονται για θύρα www (80) θα *μεταμφιέζονται*, δηλ. θα αποκτούν ως διεύθυνση αφετηρίας τη διεύθυνση του firewall και θα προωθούνται, ενώ ο επόμενος κανόνας ορίζει ότι υπόλοιπες αιτήσεις θα απορρίπτονται.

Τέλος, η όγδοη ομάδα εντολών αποτελείται από έναν μόνο κανόνα που ορίζει ότι όλες οι δρομολογήσεις από τη φυσική συσκευή eth1 προς τη φυσική συσκευή eth0 απορρίπτονται. Σημειώνουμε ότι αυτό δεν επηρεάζει τις αιτήσεις που έχουν μεταμφιεστεί δυνάμει της $7^{ης}$ ομάδας εντολών, καθώς αυτές τυγχάνουν επεξεργασίας απ' ευθείας από το υποσύστημα μεταμφίεσης.

Μία επέκταση της τεχνικής φιλτραρίσματος είναι το *δυναμικό φιλτράρισμα πακέτων*, το οποίο παράλληλα με τους κανόνες πρόσβασης που έχουν τεθεί στο firewall εξετάζονται και οι κανόνες του *δικτυακού πρωτοκόλλου* (π.χ. TCP) που χρησιμοποιείται στην επικοινωνία. Για παράδειγμα, θα μπορούσαν να απορρίπτονται πακέτα που καταφθάνουν χωρίς να έχει εγκαθιδρυθεί σχετική σύνδεση, καθώς και πακέτα που καταφθάνουν μετά την καταστροφή της σχετικής σύνδεσης ή πακέτα που απαντούν σε ερωτήσεις που ποτέ δεν έγιναν. Το μοναδικό μειονέκτημα αυτής της πρακτικής είναι ότι πρέπει να παρακολουθεί την κατάσταση όλων των συνδέσεων μεταξύ του διαδικτύου και του εταιρικού δικτύου, κάτι που απαιτεί πολλή μνήμη. Η

απαίτηση αυτή αξιοποιείται μερικές φορές από τους επιτιθέμενους, οι οποίοι εξαπολύουν επιθέσεις που εγκαθιδρύουν και διατηρούν ενεργές πολλές συνδέσεις, με αποτέλεσμα να εξαντλούνται οι πόροι του μηχανήματος που υλοποιεί το φιλτράρισμα και έτσι αυτό να καθίσταται ανενεργό.

5.2 Αντιπροσώπευση υπηρεσιών

Η τεχνική της αντιπροσώπευσης υπηρεσιών χρησιμοποιείται σε συνδυασμό συνήθως με το φιλτράρισμα πακέτων. Βάσει της τεχνικής αυτής, για κάθε υπηρεσία που παρέχεται από το εταιρικό δίκτυο και που πρέπει να είναι προσπελάσιμη από το διαδίκτυο τοποθετείται ένας *εξυπηρέτης αντιπροσώπευσης* για την υπηρεσία αυτή, ο οποίος είναι προσπελάσιμος από το διαδίκτυο, τουλάχιστον αναφορικά με τη συγκεκριμένη υπηρεσία. Οι εξυπηρετούμενοι του διαδικτύου που επιθυμούν να προσπελάσουν την υπηρεσία θα παραδώσουν την αίτησή τους στον εξυπηρέτη αντιπροσώπευσης, ο οποίος θα την προωθήσει στον *κανονικό εξυπηρέτη* που είναι τοποθετημένος εντός του εταιρικού δικτύου και απροσπέλαστος από όλους τους εξωτερικούς προς το εταιρικό δίκτυο υπολογιστές *εκτός από τον εξυπηρέτη αντιπροσώπευσης*. Ο εξυπηρέτης αντιπροσώπευσης θα συλλέξει την απάντηση από τον κανονικό εξυπηρέτη και θα την προωθήσει στον εξυπηρετούμενο.

Μέσω αυτής της τεχνικής επιτυγχάνεται η πλήρης απομόνωση του κανονικού εξυπηρέτη από το «επικίνδυνο» διαδίκτυο, μια και αυτός επικοινωνεί άμεσα μόνο με τον εξυπηρέτη αντιπροσώπευσης. Η απομόνωση είναι χρήσιμη διότι ο κανονικός εξυπηρέτης συνήθως έχει πολύτιμο περιεχόμενο (σελίδες WWW, βάσεις δεδομένων κ.λπ.) για τα οποία δεν θα θέλαμε να διακινδυνεύσουμε την ανεξέλεγκτη διαρροή, παραφθορά ή καταστροφή. Αν οι επιτιθέμενοι καταφέρουν να παραβιάσουν την ασφάλεια του εξυπηρέτη αντιπροσώπευσης η ζημιά είναι σχετικά μικρή, καθώς ο εξυπηρέτης αυτός δεν περιέχει πληροφορίες, και το μόνο που πρέπει να γίνει είναι να εγκατασταθούν εκ νέου οι υπηρεσίες ή/και το λειτουργικό σύστημα.

Συνολικά, η τεχνική της χρήσης εξυπηρετών αντιπροσώπευσης είναι απλή στην υλοποίηση, χαμηλού κόστους και παρέχει τη δυνατότητα καταγραφής των «ύποπτων» αιτήσεων (μια και εκτελείται σε υπολογιστή και όχι σε δικτυακή διάταξη). Επίσης, το εσωτερικό δίκτυο είναι απόλυτα «αόρατο» στους εξωτερικούς χρήστες, καθώς αυτοί επικοινωνούν μόνο με τους εξυπηρέτες αντιπροσώπευσης.

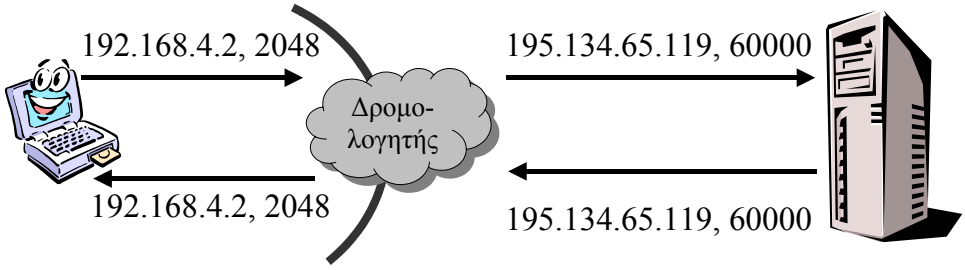
Από την άλλη πλευρά, είναι δυνατόν οι εξυπηρέτες αυτοί να αποτελέσουν σημεία συμφόρησης, καθώς –συνήθως– οι εξυπηρέτες αντιπροσώπευσης φιλοξενούν πάνω από μία υπηρεσίες, ενώ το υλικό που χρησιμοποιείται γι' αυτούς δεν είναι ιδιαίτερα ισχυρό. Για να αντιμετωπισθεί το ζήτημα αυτό, σε μερικές περιπτώσεις «ακίνδυνων» υπηρεσιών είναι δυνατόν να επιτραπεί η απ' ευθείας πρόσβαση από τους χρήστες του διαδικτύου στον κανονικό εξυπηρέτη. Το σχήμα επίσης δεν είναι ιδιαίτερα ευέλικτο, καθώς η προσθήκη μιας νέας υπηρεσίας απαιτεί πλέον και την εγκατάσταση και ρύθμιση του εξυπηρέτη αντιπροσώπευσης. Η αναγκαιότητα για ύπαρξη «διπλών» εξυπηρετών για κάθε υπηρεσία συνεπάγεται επίσης οικονομικό και διαχειριστικό κόστος, ενώ, τέλος, μερικές υπηρεσίες δεν είναι εύκολο να αντιπροσωπευθούν, λόγω ιδιαιτεροτήτων στα πρωτόκολλα εφαρμογής.

5.3 Πρόσβαση από εσωτερικούς χρήστες σε εξωτερικές υπηρεσίες

Σε πολλές περιπτώσεις είναι επιθυμητό να υπάρχει προστασία και κατά την πρόσβαση από τους *εσωτερικούς χρήστες στο διαδίκτυο*. Ένα επίπεδο προστασίας είναι η απόκρυψη της πραγματικής διεύθυνσης IP του υπολογιστή, οπότε να μην είναι

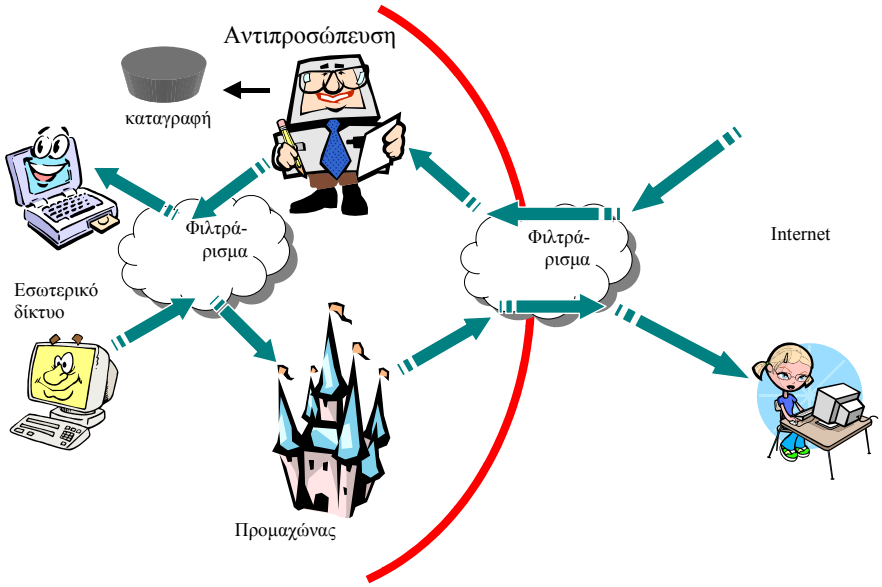
δυνατόν να αποτελέσει ο υπολογιστής στόχο για επιθέσεις. Επίσης, η χρήση και μόνο μερικών υπηρεσιών του διαδικτύου είναι επισφαλής (π.χ. ICQ), οπότε η απ' ευθείας πρόσβαση στις υπηρεσίες αυτές από τους εσωτερικούς υπολογιστές του εταιρικού δικτύου τους εκθέτει σε κίνδυνο και κατ' επέκταση διακυβεύει όλη την ασφάλεια του εταιρικού δικτύου. Για την αντιμετώπιση των ζητημάτων αυτών χρησιμοποιούνται δύο κυρίως τεχνικές, *η μετάφραση διευθύνσεων δικτύου* (network address translation), και η εισαγωγή *υπολογιστών-προμαχώνων*.

Στη μετάφραση διευθύνσεων, η οποία συνήθως πραγματοποιείται από δρομολογητές, όταν ένας υπολογιστής του εσωτερικού δικτύου προσπαθεί να επικοινωνήσει με έναν υπολογιστή του διαδικτύου, ο δρομολογητής αλλάζει το ζεύγος (διεύθυνση αφετηρίας, θύρα αφετηρίας) που αναγράφεται στα δικτυακά πακέτα σε ένα άλλο ζεύγος όπου ως διεύθυνση αφετηρίας εμφανίζεται η δική του διεύθυνση IP και ως θύρα αφετηρίας μία τυχαία θύρα, συνήθως στην περιοχή 60000 έως 65000. Ο δρομολογητής σημειώνει την αντιστοιχία του ζεύγους (διεύθυνση αφετηρίας, θύρα αφετηρίας) με την μεταφρασμένη θύρα σε έναν πίνακα αντιστοιχιών, προκειμένου (α) να τη χρησιμοποιήσει σε μεταγενέστερα πακέτα της ίδιας επικοινωνίας (β) να τη χρησιμοποιήσει στην προώθηση της απάντησης. Όταν επιστρέφει η απάντηση σε κάποιο πακέτο, αυτή παραλαμβάνεται από τον δρομολογητή σε κάποια θύρα. Από τον πίνακα αντιστοιχιών εξάγεται το ζεύγος (διεύθυνση αφετηρίας, θύρα αφετηρίας) για τον υπολογιστή που υπέβαλε την αρχική αίτηση, και η απάντηση προωθείται στη διεύθυνση που ορίζεται από το ζεύγος αυτό. Η λειτουργία της μετάφρασης διευθύνσεων απεικονίζεται στο σχήμα που ακολουθεί.



Η χρήση *υπολογιστών-προμαχώνων* συνίσταται στην εγκατάσταση και λειτουργία υπολογιστών οι οποίοι είναι «θωρακισμένοι» από άποψης ασφάλειας (δηλαδή έχουν *πάντα* εγκαταστημένα όλα τα διαθέσιμα επιδιορθωτικά προγράμματα, οι ρυθμίσεις ασφάλειας είναι ελεγμένες διεξοδικά κ.τ.λ.). Οι επικοινωνία των υπολογιστών αυτών με το εσωτερικό δίκτυο περιορίζεται στο να επιτρέπει στους εσωτερικούς χρήστες τη σύνδεση και τη μεταφορά αρχείων. Όταν ένας εσωτερικός χρήστης επιθυμεί να χρησιμοποιήσει μία ανασφαλή υπηρεσία, συνδέεται στον υπολογιστή-προμαχώνα και εκτελεί εκεί την υπηρεσία, ενώ μετά τη χρήση της μπορεί να μεταφέρει αρχεία που τυχόν έλαβε στον δικό του, εσωτερικό υπολογιστή. Η εκτέλεση της υπηρεσίας στον υπολογιστή-προμαχώνα δεν είναι τόσο επικίνδυνη όσο η εκτέλεσή της σε έναν εσωτερικό υπολογιστή, αφ' ενός μεν διότι ο υπολογιστής-προμαχώνας διαθέτει πιο ασφαλείς ρυθμίσεις, αφ' ετέρου δε διότι ακόμη και αν παραβιασθεί η ασφάλεια του υπολογιστή-προμαχώνα αυτό δεν έχει ιδιαίτερες συνέπειες για το εσωτερικό δίκτυο.

Το σχήμα που ακολουθεί παρουσιάζει τη γενική αρχιτεκτονική των firewalls με όλα τα δυνατά συστατικά.



5.4 Χρήση «περιτυλιγμάτων» υπηρεσιών

Τα περιτυλίγματα υπηρεσιών είναι ανεξάρτητα προγράμματα που χρησιμοποιούνται για να ελέγχουν την πρόσβαση σε άλλα προγράμματα ή υπηρεσίες. Η πραγματοποίηση των σχετικών ελέγχων με την τεχνική του περιτυλίγματος αποσκοπεί στον διαχωρισμό των λειτουργιών ελέγχου πρόσβασης από τις καθ' εαυτό λειτουργίες των προγραμμάτων ή υπηρεσιών που επιθυμούμε να προστατευθούν, ενώ επίσης μπορούμε να προστατεύσουμε και προγράμματα ή υπηρεσίες που είχαν αναπτυχθεί χωρίς να έχουν κατά νου την ασφάλεια και έτσι δεν ενσωματώνουν χαρακτηριστικά ελέγχου πρόσβασης. Ο διαχωρισμός επιτρέπει επίσης την ανεξάρτητη ανάπτυξη, έλεγχο και συντήρηση του περιτυλίγματος, ενώ τέλος ένα μόνο περιτύλιγμα μπορεί να χρησιμοποιηθεί για να ελέγχει την πρόσβαση σε πολλές υπηρεσίες, εξοικονομώντας έτσι πόρους συστήματος και μειώνοντας τη διαχειριστική επιβάρυνση.

Τα περιτυλίγματα μπορούν να υλοποιούν καταγραφή συνδέσεων, ενώ παράλληλα να εξετάζουν διάφορες παραμέτρους πριν επιτρέψουν κάποια χρήση προγράμματος ή υπηρεσίας. Οι έλεγχοι μπορούν να περιλαμβάνουν διάφορα χαρακτηριστικά, όπως η διεύθυνση και το όνομα του υπολογιστή απ' όπου προέρχεται η αίτηση, η ταυτότητα του χρήστη που επιθυμεί να χρησιμοποιήσει το πρόγραμμα ή την υπηρεσία κ.λπ. Το αν η χρήση της υπηρεσίας επιτρέπεται ή απαγορεύεται τελικά αποφασίζεται βάσει ενός συνόλου κανόνων.

Το πιο διαδεδομένο πρόγραμμα περιτύλιξης υπηρεσιών είναι το πακέτο *tcpd*, το οποίο χρησιμοποιείται συνήθως σε υπολογιστές τύπου Unix. Το πακέτο tcpd υποδέχεται όλες τις αιτήσεις για χρήση υπηρεσιών στον υπολογιστή (π.χ. telnet, ftp, printer κ.λπ.) και διενεργεί τους σχετικούς ελέγχους βάσει των κανόνων που περιγράφονται σε δύο αρχεία, τα */etc/hosts.allow* και */etc/hosts.deny*. Αν οι έλεγχοι είναι επιτυχής, ο έλεγχος μεταβιβάζεται στο σχετικό πρόγραμμα παροχής υπηρεσίας, αν όμως όχι η σύνδεση κλείνεται άμεσα και το πρόγραμμα παροχής υπηρεσίας δεν καλείται καθόλου.

Ένα παράδειγμα αρχείων hosts.allow και hosts.deny παρατίθεται στα σχήματα :

```
imapd: .mycom.com .affliatecom.com LOCAL
in.telnetd: .mycom.com .affliatecom.com
```

```
in.telnetd: 195.170.21.138 192.168.3.9/24
sshd: 60.70.80. 10.0.0.0/255.0.0.0
ALL: .mycom.com .affliatecom.com goodgye.somewhere.org
```

Παράδειγμα αρχείου hosts.allow

```
ALL: UNKNOWN: (/usr/sbin/safe_finger -l @%a | \
     /usr/ucb/mail -s %H-%d-%h root)&
ALL: 143.233.160.99: (/usr/sbin/safe_finger -l @%a | \
     /usr/ucb/mail -s %H-%d-%h root)&
ALL: PARANOID
ALL: .hackers.org .rivals.com 195.134.79.73
imapd: ALL
```

Παράδειγμα αρχείου hosts.deny

Παρατηρούμε ότι το αρχείο hosts.allow περιέχει γραμμές της ακόλουθης μορφής:

Όνομα_υπηρεσίας: λίστα_επιτρεπόμενων_διευθύνσεων

Το *Όνομα_υπηρεσίας* είναι στην ουσία το όνομα του προγράμματος παροχής της συγκεκριμένης υπηρεσίας, ενώ η *λίστα_επιτρεπόμενων_διευθύνσεων* παραθέτει δικτυακές ή συμβολικές διευθύνσεις υπολογιστών που επιτρέπεται να χρησιμοποιήσουν την συγκεκριμένη υπηρεσία. Η κάθε διεύθυνση μπορεί να καθορίζεται ως:

Τρόπος	***Παράδειγμα***	***Σχόλια***
Μοναδική συμβολική διεύθυνση	`goodgye.somewhere.org`	Ο υπολογιστής με το συγκεκριμένο συμβολικό όνομα
Περιοχή συμβολικών ονομάτων	`.mycom.com`	Όλοι οι υπολογιστές των οποίων το συμβολικό όνομα τελειώνει με την παρατιθέμενη συμβολοσειρά
Μοναδική δικτυακή διεύθυνση	`195.170.21.138`	Ο υπολογιστής με τη συγκεκριμένη δικτυακή διεύθυνση
Περιοχή δικτυακών διευθύνσεων	`60.70.80.`	Οι υπολογιστές που τα τρία πρώτα ψηφία της δικτυακής τους διεύθυνσης είναι `60.70.80.`
Περιοχή δικτυακών διευθύνσεων	`192.168.3.9/24`	Οι υπολογιστές που τα 24 πρώτα bits της δικτυακής τους διεύθυνσης είναι ίσα με τα 24 πρώτα bits της διεύθυνσης `192.168.3.9`
Περιοχή δικτυακών διευθύνσεων	`10.0.0.0/255.0.0.0`	Οι υπολογιστές με διεύθυνση a.b τέτοια ώστε (x & `255.0.0.0`) ==

Τρόπος	*Παράδειγμα*	*Σχόλια*
		`10.0.0.0 & 255.0.0.0)`

Τόσο το όνομα υπηρεσίας όσο και οποιοδήποτε μέλος της λίστας διευθύνσεων μπορεί να είναι η λέξη `ALL` που δηλώνει *όλες τις υπηρεσίες* ή *όλες τις διευθύνσεις* αντίστοιχα.

Ειδικότερα για τα μέλη της λίστας διευθύνσεων μπορούν να χρησιμοποιηθούν οι κάτωθι συμβολισμοί:

Συμβολισμός	*Σημασία*
`UNKNOWN`	Οι υπολογιστές των οποίων η συμβολική διεύθυνση δεν είναι γνωστή. Όταν γίνεται η σύνδεση, το περιτύλιγμα έχει στη διάθεσή του τη δικτυακή διεύθυνση του απομακρυσμένου υπολογιστή και προσπαθεί *πάντα* να βρει από αυτή τη συμβολική διεύθυνση, συνήθως με ένα ερώτημα αντίστροφης επίλυσης (reverse resolution). Αν αυτό δεν δώσει αποτέλεσμα, ο υπολογιστής χαρακτηρίζεται *UNKNOWN*.
`PARANOID`	Αν για μία σύνδεση βρεθεί το συμβολικό όνομα ενός υπολογιστή, τότε επιχειρείται και ένα ερώτημα ευθείας επίλυσης για τη συμβολική διεύθυνση για την εύρεση της δικτυακής. Υπό κανονικές συνθήκες, το ερώτημα αυτό θα πρέπει να έχει ως απάντηση τη δικτυακή διεύθυνση του υπολογιστή από τον οποίο έγινε η σύνδεση. Αν το αποτέλεσμα είναι διαφορετικό (κάτι που πολλές φορές οφείλεται είτε σε κακή ρύθμιση ή σε σκόπιμη προσπάθεια εξαπάτησης), ο υπολογιστής χαρακτηρίζεται ως *PARANOID*.

Οι συμβολισμοί UNKNOWN και PARANOID χρησιμοποιούνται συνηθέστερα στο hosts.deny.

Η σειρά επεξεργασίας των κανόνων στα δύο αρχεία, τέλος, είναι:

1. Πρώτα εξετάζεται το αρχείο *hosts.allow* και, αν βρεθεί ταίριασμα, η πρόσβαση επιτρέπεται.
2. Ακολούθως εξετάζεται το αρχείο *hosts.deny* και, αν βρεθεί ταίριασμα, η πρόσβαση απαγορεύεται.
3. Αν δεν βρεθεί ταίριασμα σε κανένα αρχείο, η πρόσβαση επιτρέπεται.

5.5 Σχεδιασμός τοπολογίας δικτύου

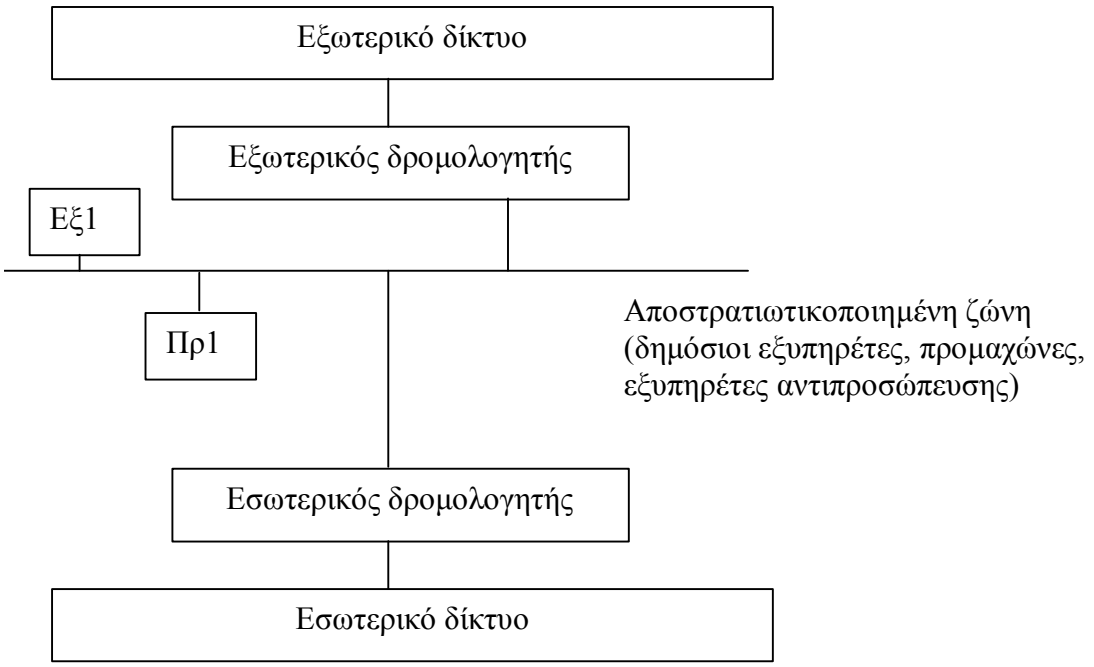
Έχοντας στη διάθεσή μας το φιλτράρισμα πακέτων, ένα ζήτημα που συνήθως τίθεται είναι η διαμόρφωση της τοπολογίας του δικτύου προκειμένου να μεγιστοποιείται η ασφάλεια. Η πιο διαδεδομένη προσέγγιση προς το ζήτημα αυτό είναι ο διαχωρισμός των υπολογιστών/ενεργών συσκευών σε τρεις κατηγορίες:

1. *Εξωτερικό δίκτυο*. Είναι οι υπολογιστές/συσκευές που βρίσκονται εκτός του εταιρικού δικτύου, οι οποίοι θεωρούνται και οι πιο επικίνδυνοι.
2. *Εσωτερικό δίκτυο*. Είναι οι υπολογιστές/συσκευές που ανήκουν στο εταιρικό δίκτυο και δεν παρέχουν καμία υπηρεσία προς τους εξωτερικούς χρήστες – με άλλα λόγια, κανένας υπολογιστής του εξωτερικού δικτύου δεν μπορεί να συνδεθεί προς οποιονδήποτε υπολογιστή του εσωτερικού δικτύου. Ανάλογα

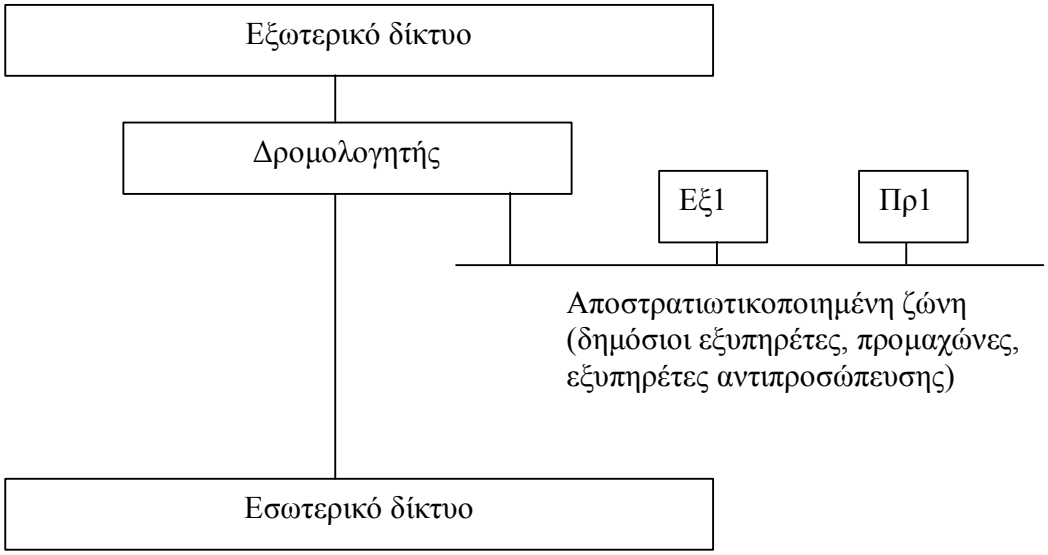
με την πολιτική ασφάλειας *είναι πιθανό* να επιτρέπεται στους εσωτερικούς χρήστες να προσπελαύνουν υπηρεσίες σε εξωτερικούς εξυπηρέτες, π.χ. να προσπελαύνουν εξυπηρέτες ΠΠΠ (WWW). Στην περίπτωση αυτή επιτρέπεται στους εξωτερικούς εξυπηρέτες να επιστρέψουν απαντήσεις στα αιτήματα που τους έχουν γίνει.

3. *Αποστρατιωτικοποιημένη ζώνη*. Είναι οι υπολογιστές /συσκευές που ανήκουν στο εταιρικό δίκτυο και παρέχουν κάποια υπηρεσία προς τους εξωτερικούς χρήστες – π.χ. οι εξυπηρέτες WWW, DNS, ηλεκτρονικού ταχυδρομείου της εταιρείας, οι οποίοι προφανώς πρέπει να είναι προσπελάσιμοι από τους υπολογιστές του εξωτερικού δικτύου. Σημειώνεται ότι εξυπηρέτες που έμμεσα υποστηρίζουν τις υπηρεσίες αυτές, π.χ. ένας εξυπηρέτης βάσεων δεδομένων που έχει υλικό για διαμόρφωση σελίδων www, *δεν* τοποθετούνται στην αποστρατιωτικοποιημένη ζώνη, αλλά στο *εσωτερικό δίκτυο*. Το ίδιο ισχύει και στην περίπτωση που έχουμε αντιπροσώπευση υπηρεσιών, όπου ο πραγματικός εξυπηρέτης τοποθετείται στο εσωτερικό δίκτυο και ο αντιπρόσωπός του στην αποστρατιωτικοποιημένη ζώνη.

Η δικτυακή διασύνδεση των τριών αυτών κατηγοριών υλοποιείται με τη χρήση *δύο δρομολογητών*, όπως φαίνεται στο σχήμα που ακολουθεί:



Επιτρέπεται σε μία τέτοια διάταξη να συγχωνεύονται οι εξωτερικός και ο εσωτερικός δρομολογητής, υπό την προϋπόθεση ότι ο δρομολογητής που θα χρησιμοποιηθεί (α) έχει διαφορετικές φυσικές συσκευές για κάθε υποδίκτυο και (β) μπορεί να εκτελεί φιλτράρισμα λαμβάνοντας υπόψη τη φυσική συσκευή εισόδου και εξόδου για κάθε πακέτο. Η εναλλακτική προσέγγιση φαίνεται στο ακόλουθο σχήμα:



6 Εισαγωγή στην κρυπτογραφία και τη διαχείριση κλειδιών

Δεδομένου ότι οι υπολογιστές και η επικοινωνία μεταξύ τους χρησιμοποιούνται όλο και περισσότερο για αποθήκευση και ανταλλαγή πολύτιμων στοιχείων, πρέπει να λαμβάνονται μέτρα για να προστατεύονται τα στοιχεία αυτά από εσκεμμένη ή συμπτωματική τροποποίηση ή κακή χρήση. Τα στοιχεία που πρέπει να προστατευθούν ποικίλλουν, αναφορικά με τη φύση τους. Μπορεί να συνίστανται σε πόρους συστήματος που ανήκουν σε παροχείς υπηρεσιών, οι οποίοι προσφέρουν πρόσβαση σε βάσεις δεδομένων, πληροφοριακά συστήματα ή, γενικώς, υπολογιστικές υπηρεσίες που περιλαμβάνουν πληροφορίες που αποθηκεύονται και ανταλλάσσονται με τη βοήθεια υπολογιστών. Μπορούν ακόμη να περιλαμβάνουν ηλεκτρονικά υπογεγραμμένα και αποθηκευμένα συμβόλαια, τα οποία αφορούν τρίτους ή ψηφιακή πληροφορία για νομικές δεσμεύσεις σε επιχειρησιακά πλαίσια.

Τα τελευταία παραδείγματα καταδεικνύουν ότι δεν χρειάζονται απλά νέες τεχνικές ασφάλειας για τις περιοχές όπου οι κλασικές προσεγγίσεις, όπως η χρήση συνθηματικών, έχουν αποδειχθεί ανεπαρκείς. Απαιτείται ασφάλεια για τις περιοχές όπου η οδηγείται η χρήση της επικοινωνίας μεταξύ υπολογιστικών συστημάτων: ως μέσο συλλογικής εργασίας που μπορεί να ανήκουν σε διαφορετικές εταιρίες, να αφορούν οποιοδήποτε αριθμό ατόμων με ενδεχομένως συγκρουόμενες επιδιώξεις και συμφέροντα, και που πιθανώς δεν γνωρίζονται μεταξύ τους πριν την εγκαθίδρυση της επικοινωνιακής οδού. Για την περιοχή αυτή, γνωστή ως «τηλεσυνεργασία», οι προηγμένες τεχνικές ασφάλειας είναι μια αναγκαιότητα.

Η ασφάλεια είναι χρήσιμη τόσο για τους χρήστες των υπολογιστικών συστημάτων όσο και για τους παροχείς υπολογιστικών υπηρεσιών, Η κλασική μέθοδος επίτευξης της ασφάλειας εξυπηρετεί καλύτερα τα συμφέροντα των παροχέων συστημάτων παρά αυτά των χρηστών. Πρώτον, οι χρήστες είναι συνήθως αυστηρά περιορισμένοι σε υπηρεσίες και πόρους συστήματος για τους οποίους έχουν πληρώσει ή για τους οποίους πιστεύεται από τους διαχειριστές των συστημάτων ότι είναι επαρκείς για τις ανάγκες των χρηστών. Δεύτερον, η πρόσβαση σε πόρους και πληροφορίες προστατεύεται με μηχανισμούς που βασίζονται σε συνθηματικά, οι οποίοι προστατεύουν τα συστήματα από μη εξουσιοδοτημένους χρήστες και κάθε χρήστη από τους υπόλοιπους, αλλά δεν παρέχουν καμία προστασία στους χρήστες απέναντι

στους διαχειριστές των συστημάτων. Τρίτον, τα συστήματα δίνουν τη δυνατότητα στους διαχειριστές να καταγράψουν λεπτομερώς τις κινήσεις των χρηστών, κάτι που δίνει τη δυνατότητα προστασίας των πόρων και επακριβούς χρέωσης για τους πόρους που χρησιμοποιήθηκαν, ενέχει όμως τον κίνδυνο της παραβίασης της ιδιωτικότητας των χρηστών. Όλα αυτά οδηγούν σε *κλειστά συστήματα*. Από την άλλη πλευρά, τα ανοικτά συστήματα που δεν παρέχουν στους χρήστες τη δυνατότητα να δημιουργήσουν κλειστά περιβάλλοντα, δεν μπορούν να φιλοξενήσουν τις περισσότερες εφαρμογές. Τα ανοικτά συστήματα στα οποία οι χρήστες έχουν τη δυνατότητα να απαιτούν υπηρεσίες, χωρίς χρονικούς ή άλλους περιορισμούς, πιθανώς με ανώνυμο τρόπο αν αυτό επιθυμείται, χωρίς προϋποθέσεις για εκ των προτέρων εγγραφή και χωρίς να είναι δυνατόν να καταγραφούν οι ενέργειές τους δεν είναι δυνατόν να στηριχθούν σε μηχανισμούς που βασίζονται στα συνθηματικά.

6.1.1 Κύρια ζητήματα για την ασφάλεια

Μπορούμε να ορίσουμε τέσσερις κύριες κατηγορίες απειλών για την ασφάλεια των επικοινωνιών στα ανοικτά συστήματα:

1. μη εξουσιοδοτημένη απόκτηση της πληροφορίας μέσω παθητικής παρακολούθησης
2. μη εξουσιοδοτημένη τροποποίηση της πληροφορίας, π.χ. αλλαγή, αναπαραγωγή ή επαναποστολή της πληροφορίας που ανταλλάσσεται μεταξύ δύο οντοτήτων.
3. *Μεταμφίεση*, δηλ. διενέργεια πράξεων υπό ταυτότητα διαφορετική από την πραγματική. Η μεταμφίεση μπορεί να λάβει διάφορες μορφές.
4. Αποκήρυξη της επικοινωνίας (δηλ. άρνηση συμμετοχής σ' αυτή), από οποιοδήποτε από τα ενεχόμενα μέρη.

Η *αυθεντικότητα* είναι ένα από τα σημεία-κλειδιά της ασφάλειας. Η αυθεντικότητα των υποκειμένων (προσώπων, οργανισμών, τμημάτων υλικού) και των αντικειμένων (αρχείων, πληροφορίας, προγραμμάτων, κλειδιών) στα συστήματα διαχείρισης πληροφοριών είναι η βάση στην οποία στηρίζεται η υπευθυνότητα, που με τη σειρά της είναι η βάση για τη συλλογική εργασία. Οι απαιτήσεις που σχετίζονται με την ασφάλεια, όπως η ακεραιότητα των δεδομένων, ο έλεγχος πρόσβασης, η απουσία δυνατότητας αποκήρυξης της επικοινωνίας και η παρεμπόδιση της μεταμφίεσης μπορούν να καλυφθούν μόνο αν υπάρχει αξιόπιστη διακρίβωση της ταυτότητας των εταίρων.

Η *εμπιστευτικότητα* είναι επίσης σημαντικό ζήτημα για πολλές εφαρμογές. Πληροφορίες που είναι απόρρητες, όπως προσωπικά δεδομένα, ιατρικά στοιχεία κ.τ.λ. δεν πρέπει να μεταδίδονται χωρίς κρυπτογράφηση μέσα από δημόσια δίκτυα, αν η διαρροή αυτών των πληροφοριών είναι δυνατόν να οδηγήσει σε οικονομικές ή άλλου τύπου ζημιές. Σε πολλές περιπτώσεις η επικοινωνία μεταξύ δύο συστημάτων διέρχεται από πολλαπλά επικοινωνιακά κανάλια, με κυμαινόμενες δυνατότητες για επίδοξους υποκλοπείς να αντλήσουν την πληροφορία που επιθυμούν. Με την πρόοδο των δικτυακών συστημάτων αρχείων, όπως π.χ. τα NFS και CIFS, οι χρήστες πολλές φορές αγνοούν ότι μία ενέργεια που φαίνεται να εκτελείται «τοπικά» στον υπολογιστή τους, στην πραγματικότητα μετακινεί πολλά δεδομένα μέσα από το δίκτυο, από ή προς τον υπολογιστή όπου πραγματικά αποθηκεύεται το αρχείο. Η κρυπτογράφηση από άκρου εις άκρον είναι απαραίτητη στις περισσότερες περιπτώσεις, ενώ και μηχανισμοί κρυπτογράφησης μεταξύ δικτυακών στοιχείων

επικοινωνίας μπορεί να είναι καλό να χρησιμοποιηθούν για προστασία από τεχνικές όπως η ανάλυση ροής πληροφορίας.

Θα πρέπει ωστόσο να σημειώσουμε ότι η εμπιστευτικότητα προϋποθέτει την αυθεντικότητα. Ο κάθε ένας πρέπει να είναι σίγουρος για την ταυτότητα αυτού με τον οποίο ανταλλάσσει ή μοιράζεται κλειδιά για την επίτευξη της εμπιστευτικότητας. Η αυθεντικότητα είναι προαπαιτούμενο για άλλες υπηρεσίες ασφάλειας και ως εκ τούτου το κεντρικό ζήτημα στην ασφάλεια.

Συνοψίζοντας μπορούμε να εντοπίσουμε την αναγκαιότητα για:

- Δυνατότητα ισχυρής διακρίβωσης της ταυτότητας των χρηστών ή άλλων ενεργών οντοτήτων (π.χ. εκτελούμενων προγραμμάτων) με αποκεντρωμένο τρόπο και υπό τον έλεγχο του χρήστη. Κατά τη διακρίβωση της ταυτότητας δεν πρέπει να αποκαλύπτονται περισσότερα στοιχεία απ' ότι είναι απαραίτητο για να εξυπηρετηθούν τα έννομα συμφέροντα των ενεχομένων μερών, και δεν πρέπει να παρεμβαίνουν ασκόπως, ή να καταγράφουν τα τεκταινόμενα κεντρικές αρχές.
- Δυνατότητα παροχής και επαλήθευσης αποδείξεων αυθεντικότητας και ακεραιότητας της πληροφορίας.
- Δυνατότητα εγγύησης της εμπιστευτικότητας και του απορρήτου σε ένα πολυχρηστικό περιβάλλον.

6.2 Κρυπτογραφία

Ο βασικός μηχανισμός για την παροχή ασφάλειας με τα χαρακτηριστικά που περιγράφηκαν πιο πάνω είναι η κρυπτογραφία. Οι αλγόριθμοι κρυπτογραφίας χρησιμοποιούνται για δύο σκοπούς. Ο πρώτος είναι για να αποδείξει κάποιος στους υπόλοιπους ότι είναι κάτοχος κάποιου συγκεκριμένου κλειδιού. Αν ο εταίρος ή ο επαληθευτής είναι σε θέση να αποκρυπτογραφήσει δεδομένα που έχουν κρυπτογραφηθεί με χρήση του συγκεκριμένου κλειδιού, τότε θεωρείται ότι η κατοχή του κλειδιού έχει αποδειχθεί. Αν επιπρόσθετα μπορεί να εξασφαλισθεί, με άλλα μέσα, ότι κανείς άλλος χρήστης ή οντότητα δεν μπορεί να έχει στη διάθεσή του το ίδιο κλειδί, η χρήση αυτού του κλειδιού συνιστά ταυτόχρονα και σύνδεσμο προς τον χρήστη του κλειδιού. Με τον τρόπο αυτό επιτυγχάνεται η αυθεντικοποίηση. Ο δεύτερος σκοπός για τον οποίο χρησιμοποιείται η κρυπτογραφία είναι η απόκρυψη της πληροφορίας, δηλαδή η προσπάθεια να αποφευχθεί η αποκάλυψή της σε μη εξουσιοδοτημένες οντότητες.

Στην κρυπτογραφία γενικά χρησιμοποιούνται οι εξής όροι:

1. *απλό* ή *μη κρυπτογραφημένο κείμενο* (plaintext). Τα δεδομένα όπως χρησιμοποιούνται από τους ανθρώπους ή τις εφαρμογές.
2. *κρυπτογραφημένο κείμενο* (cipher text). Τα δεδομένα σε ακατάληπτη για τους ανθρώπους ή τις εφαρμογές μορφή.
3. *Κρυπτογράφηση*. Ο μετασχηματισμός του απλού κειμένου σε κρυπτογραφημένο κείμενο.
4. *Αποκρυπτογράφηση*. Ο μετασχηματισμός του κρυπτογραφημένου κειμένου σε απλό.

5. *Κλειδί.* Μια ποσότητα πληροφορίας (σύνολο bytes) που καθορίζει τους μετασχηματισμούς που θα πραγματοποιηθούν κατά τη διαδικασία της κρυπτογράφησης ή αποκρυπτογράφησης.
6. *Χώρος μη κρυπτογραφημένων μηνυμάτων Μ.* Όλα τα δυνατά μηνύματα απλού κειμένου.
7. *Χώρος κρυπτογραφημένων μηνυμάτων C.* Όλα τα δυνατά μηνύματα κρυπτογραφημένου κειμένου.
8. *Χώρος κλειδιών Κ.* Όλα τα δυνατά κλειδιά.
9. *Οικογένεια μετασχηματισμών κρυπτογράφησης.* Μια ομάδα συναρτήσεων E_k με πεδίο ορισμού το Μ και πεδίο τιμών το C. Υπάρχει μία συνάρτηση για κάθε κλειδί. Η κρυπτογράφηση ενός απλού κειμένου μπορεί να γραφεί ως $E_k(m) = c$, όπου $m \in M$, $c \in C$.
10. *Οικογένεια μετασχηματισμών αποκρυπτογράφησης.* Μια ομάδα συναρτήσεων D_k με πεδίο ορισμού το C και πεδίο τιμών το Μ. Υπάρχει μία συνάρτηση για κάθε κλειδί. Η αποκρυπτογράφηση ενός κρυπτογραφημένου κειμένου μπορεί να γραφεί ως $D_k(c) = m$, όπου $m \in M$, $c \in C$

Έχοντας στη διάθεσή μας τους ανωτέρω ορισμούς μπορούμε να διατυπώσουμε τους στόχους της κρυπτογραφίας με πιο τεχνικό τρόπο ως ακολούθως:

1. Εχεμύθεια
    i. Πρέπει να είναι ανέφικτο να υπολογιστεί το D_k από το c, ακόμη και αν είναι γνωστό το m
    ii. Πρέπει να είναι ανέφικτος ο υπολογισμός του m από ένα c
2. Αυθεντικότητα
    i. Πρέπει να είναι υπολογιστικά ανέφικτο να προσδιοριστεί το Ek από το c, ακόμη και αν είναι γνωστό το m
    ii. Πρέπει να είναι υπολογιστικά ανέφικτο να βρεθεί ένα c', τέτοιο ώστε το Dk(c') να είναι παραδεκτό μη κρυπτογραφημένο μήνυμα του συνόλου Μ

Για ένα σύστημα κρυπτογραφίας είναι επίσης επιθυμητές η κάτωθι ιδιότητες:

- Πρέπει να υπάρχουν αποδοτικοί αλγόριθμοι για τις λειτουργίες της κωδικοποίησης και της αποκωδικοποίησης
- Το σύστημα πρέπει να είναι εύχρηστο.
- Η προστασία που παρέχει το σύστημα πρέπει να προϋποθέτει μόνο τη μυστικότητα των κλειδιών, όχι του αλγόριθμου

Αναφορικά με τις κρυπτογραφικές μεθόδους μπορούμε να διακρίνουμε δύο μείζονες κατηγορίες: η πρώτη είναι οι *συμμετρικοί αλγόριθμοι,* όπου η κρυπτογράφηση και η αποκρυπτογράφηση γίνεται χρησιμοποιώντας το ίδιο κλειδί αλλά αντίστροφες λειτουργίες. Ο πιο διαδεδομένος αλγόριθμος συμμετρικής κρυπτογραφίας είναι ο DES που επινοήθηκε το 1977. Η δεύτερη κατηγορία είναι οι *ασύμμετροι αλγόριθμοι*. Ο πρώτος επινοήθηκε από τους Diffie και Hellmann το 1976. Οι αλγόριθμοι αυτοί χρησιμοποιούν διαφορετικά κλειδιά για τις λειτουργίες της κρυπτογράφησης και της αποκρυπτογράφησης. Τα κλειδιά είναι μαθηματικώς συναρτώμενα μεταξύ τους, αλλά

υπάρχει η πρόσθετη απαίτηση να *το πολύ ένα εξ αυτών να είναι δυνατόν να υπολογιστεί από το άλλο με «υπολογιστικά εφικτό» τρόπο, ενώ η άλλη κατεύθυνση υπολογισμού θα πρέπει να είναι υπολογιστικά ανέφικτη.* Η ιδιότητα αυτή επιτρέπει να δημοσιοποιηθεί το ένα κλειδί (αυτό που υπολογίζεται βάσει του άλλου), ενώ το άλλο κλειδί τηρείται μυστικό και συνδέεται άρρηκτα με την οντότητα που προσδιορίζει. Λόγω του σχήματος λειτουργίας αυτού, οι αλγόριθμοι αυτοί πολλές φορές ονομάζονται *αλγόριθμοι δημόσιου κλειδιού*.

Ένας ακόμη διαχωρισμός που μπορεί να γίνει στους αλγόριθμους κρυπτογραφίας είναι μεταξύ των αλγορίθμων *κρυπτογράφησης κατά μπλοκ* και των αλγορίθμων *κρυπτογράφησης αλυσιδωτών μπλοκ*. Στους αλγόριθμους κρυπτογράφησης κατά μπλοκ το αρχικό μήνυμα Μ διασπάται σε διαδοχικά μπλοκ Μ1, Μ2, ... Κατόπιν το κάθε τμήμα κρυπτογραφείται με το ίδιο κλειδί Κ και το τελικό κρυπτογραφημένο μήνυμα είναι η ακολουθία $E_k(M1)E_k(M2)$. Η μέθοδος αυτή έχει τα πλεονεκτήματα ότι γίνεται μόνο μία εκτέλεση του κρυπταλγόριθμου ανά μπλοκ και ότι σφάλματα κατά μετάδοση επηρεάζουν μόνο το συγκεκριμένο μπλοκ όπου εμφανίζονται, από την άλλη πλευρά όμως είναι πιο ευάλωτα σε επιθέσεις κρυπτανάλυσης, καθώς ίδια τμήματα μη κρυπτογραφημένου κειμένου παράγουν ίδια τμήματα κρυπτογραφημένου κειμένου. Σε αντιδιαστολή με τους αλγόριθμους *κρυπτογράφησης κατά μπλοκ*, στους αλγόριθμους *κρυπτογράφησης αλυσιδωτών μπλοκ* το κάθε μπλοκ δεν είναι αυτόνομο, αλλά περιλαμβάνει bits από τα προηγούμενα (κρυπτογραφημένα ή μη). Με τον τρόπο αυτό από τη μία πλευρά μειώνονται οι διαθέσιμες θέσεις πληροφορίας σε κάθε μπλοκ και αναιρείται το πλεονέκτημα της ανοχής σε σφάλματα, ενώ επίσης για να επεξεργασθούμε ένα μόνο τμήμα του μηνύματος είναι απαραίτητο να αποκρυπτογραφηθεί όλο το μήνυμα, προκαλώντας έτσι απώλειες στην απόδοση. Από την άλλη πλευρά όμως, αυξάνεται σημαντικά η ασφάλεια.

6.2.1 Συμμετρικοί αλγόριθμοι κρυπτογραφίας

Στα επόμενα εδάφια παρουσιάζονται μερικοί από τους συμμετρικούς αλγόριθμους κρυπτογραφίας που έχουν κατά καιρούς χρησιμοποιηθεί.

6.2.1.1 Κρυπτογράφηση με μεταθέσεις

Η κρυπτογράφηση με μεταθέσεις βασίζεται στη γενική ιδέα ότι τα bytes του αρχικού μηνύματος αναδιατάσσονται με κάποιον τρόπο που προσδιορίζει ο αλγόριθμος και το κλειδί. Οι πιο γνωστοί αλγόριθμοι είναι η απλή μετάθεση, το «συρματόπλεγμα» και η μετάθεση κατά στήλες.

Απλή μετάθεση

Στη διαδικασία της απλής μετάθεσης χρησιμοποιείται ως κλειδί ένα διάνυσμα ν-θέσεων, όπου σε κάθε θέση περιέχεται ένας αριθμός από το ένα έως το ν. Το διάνυσμα πρέπει να περιέχει όλους τους αριθμούς από το ένα έως το ν, και έτσι κάθε αριθμός εμφανίζεται μία μόνο φορά.

Στη διαδικασία κρυπτογράφησης το μήνυμα αρχικά κατατμείται σε μπλοκ μεγέθους *ν*, και εντός κάθε τμήματος τα bytes αναδιατάσσονται όπως ορίζει το κλειδί. Αν, για παράδειγμα το πρώτο στοιχείο του κλειδιού είναι ίσο με 2, το πρώτο byte στο κρυπτογραφημένο τμήμα θα ισούται με το δεύτερο byte του μη κρυπτογραφημένου τμήματος.

Παράδειγμα:

Έστω ότι το κλειδί είναι ίσο με *(2 5 4 1 3)* και το μη κρυπτογραφημένο κείμενο είναι ίσο με *ΜΥΣΤΙΚΟ ΜΗΝΥΜΑ*. Αρχικά πρέπει να κατατμηθεί το μη κρυπτογραφημένο κείμενο σε τεμάχια των 5 bytes, όσο δηλαδή και το μήκος του κλειδιού. Επειδή το μήνυμα έχει μήκος 14 bytes, το οποίο δεν είναι ακέραιο πολλαπλάσιο του 5, θα συμπληρωθεί με τον ειδικό χαρακτήρα ∅, ο οποίος κανονικά δεν είναι δυνατόν να εμφανισθεί ανάμεσα στους παραδεκτούς χαρακτήρες του μη κρυπτογραφημένου μηνύματος. Έτσι τα τεμάχια του μη κρυπτογραφημένου μηνύματος θα έχουν ως ακολούθως:

ΜΥΣΤΙ	ΚΟ ΜΗ	ΝΥΜΑ∅

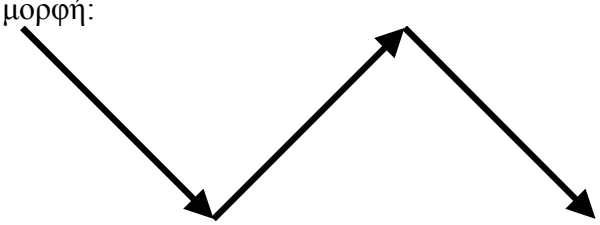
Το κάθε τμήμα πλέον αναδιατάσσεται όπως ορίζει το κλειδί, καταλήγοντας στο κρυπτογραφημένο μήνυμα

ΥΙΤΜΣ	ΟΗΜΚ	Υ∅ΑΝΜ

Η διαδικασία της αποκρυπτογράφησης ακολουθεί ακριβώς την αντίστροφη διαδικασία, ακολουθούμενη από την εξάλειψη των χαρακτήρων «∅» που υπάρχουν στο τέλος του μηνύματος.

«Συρματόπλεγμα»

Στην κρυπτογράφηση βάσει της μεθόδου του «συρματοπλέγματος» το μη κρυπτογραφημένο μήνυμα αρχικά γράφεται κατά μήκος ενός νοητού «σύρματος» που έχει την ακόλουθη μορφή:



Το πλήθος των χαρακτήρων που γράφονται σε κάθε τμήμα του «σύρματος» καθορίζεται από το κλειδί. Αφού το απλό κείμενο γραφεί κατ' αυτή την έννοια, διαβάζεται ακολούθως *κατά γραμμές*, διαμορφώνοντας έτσι το κρυπτογραφημένο κείμενο.

Παράδειγμα:

Έστω ότι το κλειδί μας υποδεικνύει να γράφονται τρία bytes σε κάθε τμήμα του «συρματοπλέγματος» και ότι το απλό κείμενο είναι *ΜΥΣΤΙΚΟ ΜΗΝΥΜΑ*. Η γραφή στο «συρματόπλεγμα» θα έχει ως εξής:

Μ				Ι				Μ				Μ	
	Υ		Τ		Κ				Η		Υ		Α
		Σ				Ο				Ν			

Στη συνέχεια, διαβάζοντας τον ανωτέρω πίνακα κατά γραμμές (αρχικά η πρώτη γραμμή, ακολούθως η δεύτερη κ.λπ.) καταλήγουμε στο ακόλουθο κρυπτογραφημένο μήνυμα:

ΜΙΜΜΥΤΚ ΗΥΑΣΟΝ

Το κλειδί σε μία τέτοια κρυπτογράφηση αποτελείται από δύο ακεραίους, ο πρώτος από τους οποίους προσδιορίζει το πλήθος των bytes που γράφουμε σε κάθε «τμήμα σύρματος» (ισοδύναμο με το πλήθος των γραμμών του ανωτέρω πίνακα) και ο δεύτερος τη *μετατόπιση έναρξης*, που ουσιαστικά υποδεικνύει πόσες στήλες στον ως άνω πίνακα θα αφήσουμε κενές πριν αρχίσουμε να γράφουμε τα bytes του απλού κειμένου.

Μετάθεση κατά στήλες

Στη μετάθεση κατά στήλες ως κλειδί χρησιμοποιείται μία λέξη, της οποίας τα γράμματα αντιστοιχίζονται σε αριθμούς, ανάλογα με τη σειρά εμφάνισής τους στο αλφάβητο. Για παράδειγμα, αν η λέξη-κλειδί είναι

ΠΟΛΥΜΕΡΕΣ

η αντιστοιχία των γραμμάτων του κλειδιού σε αριθμούς εκφράζεται από το διάνυσμα

(6 5 3 9 4 1 7 2 8)

(Το Ε αντιστοιχίζεται στο 1 γιατί είναι το μικρότερο λεξικογραφικά γράμμα της λέξης, ενώ η δεύτερη εμφάνιση του Ε αντιστοιχίζεται με το 2. Το επόμενο λεξικογραφικά γράμμα είναι το Λ, που αντιστοιχίζεται στον αριθμό 3 κ.ο.κ.). Στη συνέχεια, το μη κρυπτογραφημένο κείμενο γράφεται σε έναν πίνακα που έχει τόσες στήλες όσες τα γράμματα του κλειδιού, ενώ το πλήθος γραμμών καθορίζεται από το μήκος του μη κρυπτογραφημένου κειμένου. Τέλος, το κρυπτογραφημένο κείμενο παράγεται με ανάγνωση του πίνακα κατά στήλες, με τη σειρά που ορίζεται από την απεικόνιση του κλειδιού. Για παράδειγμα, αν το μη κρυπτογραφημένο κείμενο είναι

ΑΣΠΡΗ ΠΕΤΡΑ ΞΕΞΑΣΠΡΗ

για να κρυπτογραφηθεί με το κλειδί ΠΟΛΥΜΕΡΕΣ θα γραφεί σε έναν πίνακα ως εξής:

Π	Ο	Λ	Υ	Μ	Ε	Ρ	Ε	Σ
6	5	3	9	4	1	7	2	8
Α	Σ	Π	Ρ	Η		Π	Ε	Τ
Ρ	Α		Ξ	Ε	Ξ	Α	Σ	Π
Ρ	Η	∅	∅	∅	∅	∅	∅	∅

και το κρυπτογραφημένο κείμενο που θα παραχθεί θα είναι

Ξ∅Ε Σ∅Π ∅ΗΕ∅ΣΑΗΑΡΡΠΑ∅ΤΠ∅ΡΞ∅

Βλέπουμε ότι το μη κρυπτογραφημένο κείμενο συμπληρώνεται με τον ειδικό χαρακτήρα ∅ προκειμένου να αποκτήσει μήκος πολλαπλάσιο του κλειδιού.

6.2.1.2 Κρυπτογράφηση με αντικατάσταση

Η κρυπτογράφηση με αντικατάσταση βασίζεται στη γενική ιδέα ότι η παραγωγή του κρυπτογραφημένου κειμένου γίνεται αντικαθιστώντας κάθε ένα byte του μη κρυπτογραφημένου κειμένου με κάποιο άλλο byte, όπως προκύπτει από μία συνάρτηση αντικατάστασης. Στη συνέχεια παρουσιάζονται τα πιο χαρακτηριστικά παραδείγματα.

Απλή αντικατάσταση

Στην απλή αντικατάσταση, για κάθε γράμμα του αλφαβήτου των μηνυμάτων ορίζουμε την απεικόνισή του, συνήθως μέσω ενός πίνακα. Αν, για παράδειγμα, το αλφάβητο των μηνυμάτων είναι το Ελληνικό αλφάβητο, ο πίνακας απεικόνισης θα μπορούσε να έχει την εξής μορφή:

Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω
Δ	Κ	Ρ	Ο	Υ	Λ	Ω	Ε	Π	Η	Α	Φ	Ζ	Μ	Τ	Β	Χ	Θ	Ν	Ψ	Γ	Ι	Σ	Ξ

Με βάση τον πίνακα αυτόν, η κρυπτογράφηση του απλού κειμένου

ΑΣΠΡΗ ΠΕΤΡΑ ΞΕΞΑΣΠΡΗ

παράγει το κρυπτογραφημένο κείμενο

ΔΘΒΧΩ ΒΥΝΧΔ ΜΥΜΔΘΒΧΩ

Η συγκεκριμένη μέθοδος κρυπτογράφησης έχει το μειονέκτημα να είναι ιδιαίτερα ευάλωτη σε επιθέσεις βασισμένες σε στατιστικές αναλύσεις εμφάνισης μεμονωμένων χαρακτήρων, ζευγών, τριάδων, κ.λπ.

Πολυαλφαβητική αντικατάσταση

Η πολυαλφαβητική αντικατάσταση είναι μία φυσική επέκταση της απλής αντικατάστασης, όπου πέρα από τον πίνακα αντικατάστασης χρησιμοποιεί και ένα επιπλέον κλειδί *K*, του οποίου τα στοιχεία ανήκουν σε ένα αλφάβητο A_k. Ο πίνακας αντικαταστάσεων έχει τόσες στήλες όσα τα στοιχεία του αλφαβήτου μηνυμάτων και τόσες γραμμές όσα τα στοιχεία του A_k, το κάθε δε στοιχείο του πίνακα περιέχει τον χαρακτήρα του αλφαβήτου κρυπτογραφημένων μηνυμάτων που πρέπει να χρησιμοποιηθεί όταν κρυπτογραφείται το στοιχείο του αλφαβήτου μηνυμάτων που αντιστοιχεί στη στήλη με το στοιχείο του αλφάβητου κλειδιών που αντιστοιχεί στη γραμμή. Αν $M_ι$ είναι ο υπ' αριθμόν *ι* χαρακτήρας του μη κρυπτογραφημένου μηνύματος και $K_ι$ ο υπ' αριθμόν *ι* χαρακτήρας του κλειδιού, ο υπ' αριθμόν *ι* χαρακτήρας του κρυπτογραφημένου μηνύματος είναι η καταχώρηση στη θέση ($K_ι$, $M_ι$) του πίνακα.

Παράδειγμα:

Έστω ότι το αλφάβητο των μη κρυπτογραφημένων μηνυμάτων και των κρυπτογραφημένων μηνυμάτων είναι το ελληνικό αλφάβητο, το αλφάβητο των κλειδιών είναι το `{Α, Β}`, και ο πίνακας αντικατάστασης είναι ο εξής:

	Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω
Α	Δ	Κ	Ρ	Ο	Υ	Λ	Ω	Ε	Π	Η	Α	Φ	Ζ	Μ	Τ	Β	Χ	Θ	Ν	Ψ	Γ	Ι	Σ	Ξ
Β	Η	Λ	Θ	Ρ	Δ	Ξ	Κ	Α	Φ	Ο	Γ	Ψ	Π	Ι	Υ	Χ	Μ	Β	Σ	Ω	Ε	Ν	Ζ	Τ

τότε η κρυπτογράφηση του απλού κειμένου

```
ΑΣΠΡΗ ΠΕΤΡΑ ΞΕΞΑΣΠΡΗ
```

με το κλειδί

```
ΑΒΒΑΑ ΒΒΑΑΒ ΒΑΑΒΒΑΑΒ
```

παράγει το κρυπτογραφημένο κείμενο

```
ΔΒΧΧΩ ΧΔΝΧΗ ΙΘΜΗΒΒΧΚ
```

Ένα ζήτημα που τίθεται στην πολυαλφαβητική αντικατάσταση είναι τι συμβαίνει αν το μήκος του μη κρυπτογραφημένου κειμένου είναι μεγαλύτερο από αυτό του κλειδιού. Στην περίπτωση αυτή υπάρχουν οι κάτωθι επιλογές:

1. *το κλειδί χρησιμοποιείται εξ αρχής*, π.χ. αν το κλειδί είναι *ΑΒΒΑ* μπορεί να επεκταθεί σε *ΑΒΒΑΑΒΒΑΑΒΒΑΑΒΒΑ...*
2. *το κλειδί χρησιμοποιείται μετασχηματισμένο, π.χ. αυξάνοντας όλους τους χαρακτήρες του κατά ένα,* π.χ. το κλειδί *HAL* θα γίνει *HALIBMJCN...*
3. *χρησιμοποιούνται ως κλειδί οι αρχικοί χαρακτήρες του μη κρυπτογραφημένου κειμένου*, π.χ. αν το κείμενο είναι *ΑΣΠΡΗ ΠΕΤΡΑ ΞΕΞΑΣΠΡΗ* και το κλειδί είναι *ΑΒΒΑ*, το τελικό κλειδί που θα χρησιμοποιηθεί θα είναι *ΑΒΒΑΑΣΠΡΗ ΠΕΤΡΑ ΞΕΞΑ*. Για να είναι εφικτή αυτή η μέθοδος θα πρέπει το αλφάβητο των κλειδιών και το αλφάβητο των μη κρυπτογραφημένων μηνυμάτων να ταυτίζονται.

Κρυπτογράφηση τρέχοντος κλειδιού

Η κρυπτογράφηση τρέχοντος κλειδιού είναι παρόμοια με την πολυαλφαβητική αντικατάσταση με τον περιορισμό ότι το κλειδί δεν είναι δυνατόν να εξαντληθεί. Αυτό εξασφαλίζεται αν χρησιμοποιηθεί ως κλειδί το κείμενο κάποιου βιβλίου, ψευδοτυχαία δεδομένα (δεδομένα που έχουν έναν βαθμό τυχαιότητας αλλά παράγονται αλγοριθμικά) ή τυχαία δεδομένα. Η χρήση των τυχαίων δεδομένων είναι προτιμότερη, ως λιγότερο ευάλωτη σε στατιστικές αναλύσεις.

Μέθοδος Vernam

Η μέθοδος Vernam έχει τα εξής χαρακτηριστικά:

1. Τα κλειδιά ανταλλάσσονται εκ των προτέρων εξωσυστημικά
2. Το κάθε κλειδί χρησιμοποιείται μόνο μία φορά
3. Το κάθε κλειδί έχει μήκος τουλάχιστον ίσο με το μήκος του κρυπτογραφούμενου μηνύματος

Η κρυπτογράφηση γίνεται εκτελώντας την πράξη της αποκλειστικής διάζευξης ανάμεσα στα αντίστοιχα bytes του μη κρυπτογραφημένου μηνύματος και του κλειδιού. Η μέθοδος αυτή είναι εξαιρετικά ασφαλής (θεωρείται αδύνατο να «σπάσει»), αλλά ο περιορισμός της εκ των προτέρων ανταλλαγής των κλειδιών με εξωσυστημικό τρόπο περιορίζει τη δυνατότητα εφαρμογής της σε συστήματα όπου οι εταίροι της επικοινωνίας είναι εκ των προτέρων γνωστοί.

Αλγόριθμος DES

Ο αλγόριθμος DES ανήκει στην κατηγορία των αλγορίθμων *κρυπτογράφησης κατά μπλοκ* και οι κρυπτογράφηση γίνεται σε μπλοκ των 64 bits με κλειδιά μεγέθους 56 bits. Ο αλγόριθμος έχει επινοηθεί το 1977 και τα μεγέθη των κλειδιών ήταν βασισμένα στις δυνατότητες των υπολογιστών της εποχής, οι οποίες όμως έχουν αλλάξει δραστικά μέσα σε 25 χρόνια. Στη βασική του διαμόρφωση ο DES είναι πλέον ανασφαλής, καθώς σε διαγωνισμό του ’98 ο κώδικας έσπασε σε 56 ώρες από μηχανή που κόστισε λιγότερο από 300.000 €. Μία προσπάθεια για αύξηση της ασφάλειας του αλγόριθμου DES χωρίς να εισαχθεί νέος αλγόριθμος ήταν να εφαρμόζεται ο αλγόριθμος δύο φορές, με κλειδιά K_1 και K_2. Με άλλα λόγια, το κρυπτογραφημένο κείμενο C παράγεται από το μη κρυπτογραφημένο μήνυμα M με τον τύπο $E(E(M, K_1), K_2)$. Η εφαρμογή όμως του αλγόριθμου δύο φορές έχει τα εξής ζητήματα:

- Υπάρχουν τέσσερα ασθενή κλειδιά K, τέτοια ώστε $E(E(M, K), K) = M$
- Υπάρχουν τέσσερα *ημιασθενή* ζεύγη κλειδιών (K_1, K_2), τέτοια ώστε $E(E(M, K_1), K_2) = M$
- Ακόμη χειρότερα όμως, η διπλή εφαρμογή του αλγόριθμου είναι ευάλωτη σε επιθέσεις τύπου *«συνάντησης στο μέσον»*, όπου η πολυπλοκότητα του αλγορίθμου της αποκρυπτογράφησης είναι ίση με αυτή του απλού αλγόριθμου.

Αν και η διπλή εφαρμογή του αλγόριθμου δεν επιφέρει ιδιαίτερη αύξηση της ασφάλειας, η *τριπλή εφαρμογή* του καταλήγει σε ένα αρκετά πιο ασφαλές σύστημα. Η τριπλή εφαρμογή έχει δύο παραλλαγές:

1. Το αρχικό μήνυμα κρυπτογραφείται με το κλειδί K_1, κατόπιν το αποτέλεσμα κρυπτογραφείται με το κλειδί K_2 και το εξαγόμενο του δεύτερου βήματος κρυπτογραφείται με το κλειδί K_3.
2. Όμοια με την προηγούμενη περίπτωση, αλλά στο δεύτερο βήμα διενεργείται η πράξη της *αποκρυπτογράφησης* αντί της πράξης της κρυπτογράφησης.

Οι δύο παραλλαγές ονομάζονται εν συντομία 3DES-EEE και 3DES-EDE.

6.2.2 Ασύμμετροι αλγόριθμοι κρυπτογραφίας

Οι ασύμμετροι αλγόριθμοι κρυπτογραφίας, σε αντίθεση με τους συμμετρικούς, χρησιμοποιούν *δύο διαφορετικά κλειδιά* για τις λειτουργίες κρυπτογράφησης και αποκρυπτογράφησης. Τα κλειδιά αυτά χρησιμοποιούνται *κατά ζεύγη* με το ένα κλειδί συνήθως να τηρείται μυστικό (στην κατοχή του ιδιοκτήτη) και το άλλο να δημοσιοποιείται.

Οι ασύμμετροι αλγόριθμοι μπορούν περαιτέρω να διαχωρισθούν σε δύο υποκατηγορίες, τους *αντιστρέψιμους* και τους *μη αντιστρέψιμους*. Αν E είναι η συνάρτηση κρυπτογράφησης και D είναι η συνάρτηση αποκρυπτογράφησης, και E_k και D_k τα αντίστοιχα κλειδιά, για έναν αντιστρέψιμο αλγόριθμο ισχύει:

$$D(E(data, E_k), D_k) = E(D(data, D_k), E_k) = data$$

Με άλλα λόγια δηλαδή, η σειρά εφαρμογής των πράξεων κρυπτογράφησης και αποκρυπτογράφησης δεν επηρεάζει το αποτέλεσμα. Η ιδιότητα αυτή επιτρέπει σε έναν μοναδικό αλγόριθμο –και ενδεχομένως ένα ζεύγος κλειδιών– να χρησιμοποιηθεί τόσο για στόχους αυθεντικότητας όσο και για εμπιστευτικότητας, υπό το σχήμα ότι χρησιμοποιεί κανείς το δικό του μυστικό κλειδί για να δημιουργήσει μια δική του *ψηφιακή υπογραφή* η οποία μπορεί να επαληθευτεί από οποιονδήποτε βάσει του δημόσιου κλειδιού, ενώ παράλληλα οποιοσδήποτε μπορεί να κρυπτογραφήσει πληροφορία με το δημόσιο κλειδί και να την αποστείλει στον κάτοχο του ιδιωτικού κλειδιού, ο οποίος είναι ο μόνος που μπορεί να την αποκρυπτογραφήσει. Ο πιο γνωστός αντιστρέψιμος ασύμμετρος αλγόριθμος είναι ο RSA.

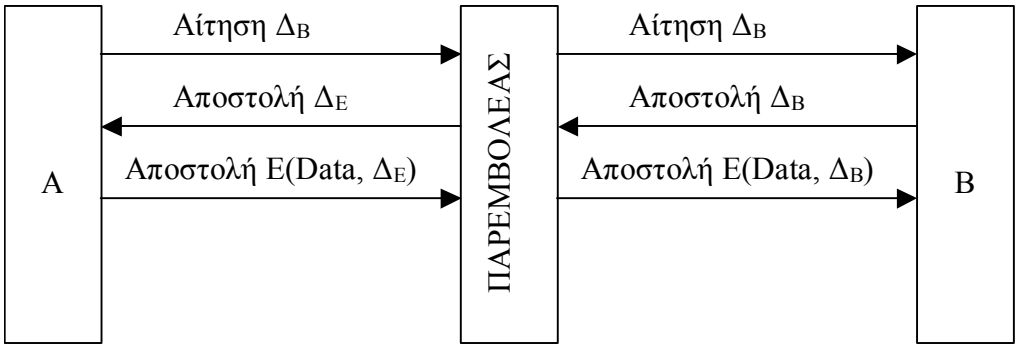
Οι μη αντιστρέψιμοι ασύμμετροι αλγόριθμοι δεν έχουν αυτή την ιδιότητα, δηλ. δεν είναι δυνατόν να ανακτηθούν τα αρχικώς κρυπτογραφημένα δεδομένα από τα κρυπτογραφημένα, και συνεπώς δεν υπάρχουν συναρτήσεις κρυπτογράφησης-αποκρυπτογράφησης με την ανωτέρω έννοια. Οι αλγόριθμοι αυτοί επιτρέπουν την διακρίβωση ότι μία ψηφιακή υπογραφή δημιουργήθηκε με ένα συγκεκριμένο μυστικό κλειδί, χρησιμοποιώντας το αντίστοιχο δημόσιο κλειδί. Για τον λόγο αυτό η συγκεκριμένη κατηγορία αλγορίθμων καλείται *αλγόριθμοι υπογραφής μόνο*.

6.2.2.1 Γενική λειτουργία ασύμμετρης κρυπτογραφίας

Υποθέτοντας ότι τα δύο μέρη που θα επικοινωνήσουν είναι τα A και B και ότι τα δημόσια και ιδιωτικά κλειδιά τους είναι Δ_A, Δ_B, I_A και I_B, θα πρέπει ο A να έχει στην κατοχή του τα I_A και Δ_B, ενώ ο B να έχει στην κατοχή του τα I_B και Δ_A. Για να στείλει ο A στον B κάποια δεδομένα τα κρυπτογραφεί με το Δ_B και τα αποστέλλει. Το κρυπτογραφημένο μήνυμα που αποστέλλεται μπορεί να αποκρυπτογραφηθεί μόνο με το ιδιωτικό κλειδί του B, το οποίο είναι γνωστό μόνον στον B – συνεπώς τα κρυπτογραφημένα δεδομένα δεν είναι χρήσιμα σε κάποιον υποκλοπέα που πιθανώς θα τα καταγράψει από τη γραμμή επικοινωνίας. Αντιστρόφως, για να στείλει δεδομένα ο B στον A τα κρυπτογραφεί πρώτα με το Δ_A και ο A τα αποκρυπτογραφεί με το I_A.

Στο σημείο αυτό τίθεται το ζήτημα πώς ο Α μαθαίνει το δημόσιο κλειδί του Β και πώς ο Β το δημόσιο κλειδί του Α. Αν ο κάθε εταίρος της επικοινωνίας απλά το ζητάει από τον άλλο, θα λάβει ένα κλειδί αλλά δεν είναι καθόλου βέβαιο ότι θα είναι το σωστό. Υπάρχει το ενδεχόμενο της επίθεσης που είναι γνωστή ως «παρεμβολέας» ή "man in the middle". Στον τύπο αυτό της επίθεσης, που απεικονίζεται στο σχήμα που ακολουθεί, ο εισβολέας παρεμβαίνει στην επικοινωνία, υποδεχόμενος όλα τα δικτυακά πακέτα που μετακινούνται προς τις δύο κατευθύνσεις. Έχοντας υποδεχθεί το κάθε πακέτο το επεξεργάζεται, το καταγράφει και το προωθεί στον προορισμό του, πιθανώς αλλοιωμένο. Η επίθεση συνίσταται στα εξής βήματα:

1. Όταν ο Α ζητάει το δημόσιο κλειδί του Β, ο εισβολέας προωθεί κανονικά την αίτηση στον Β.
2. Ο εισβολέας «συλλαμβάνει» την απάντηση του Β, μαθαίνοντας έτσι το δημόσιο κλειδί του Β, και αποστέλλει στον Α το δικό του δημόσιο κλειδί.
3. Ο Α παραλαμβάνει το δημόσιο κλειδί του εισβολέα, πιστεύοντας ότι είναι του Β, κρυπτογραφεί με αυτό τα δεδομένα και τα στέλνει.
4. Ο εισβολέας και πάλι «συλλαμβάνει» την κρυπτογραφημένη επικοινωνία, η οποία όμως έχει κρυπτογραφηθεί με το δικό του δημόσιο κλειδί, άρα μπορεί να την αποκρυπτογραφήσει με το δικό του ιδιωτικό κλειδί (το οποίο φυσικά κατέχει). Έτσι έχει στη διάθεσή του τα δεδομένα για καταγραφή ή αλλοίωση. Τέλος, ο εισβολέας κρυπτογραφεί τα δεδομένα με το δημόσιο κλειδί του Β (το οποίο και γνωρίζει από το βήμα 2) και τα αποστέλλει στον Β.



Το πρόβλημα ανακύπτει ουσιαστικά διότι οι εταίροι Α και Β δεν έχουν καμία εγγύηση ότι τα κλειδιά που λαμβάνουν είναι αυθεντικά. Το ζήτημα της αυθεντικότητας των κλειδιών αναπτύσσεται στην παράγραφο 6.2.4.

6.2.2.2 Παραδείγματα αλγορίθμων ασύμμετρης κρυπτογραφίας

Στις ακόλουθες παραγράφους παρουσιάζονται μερικά παραδείγματα αλγορίθμων ασύμμετρης κρυπτογραφίας.

Ο αλγόριθμος RSA

Ο αλγόριθμος RSA βασίζει την ασφάλειά του στη δυσκολία παραγοντοποίησης μεγάλων ακέραιων αριθμών. Για την παραγωγή των κλειδιών χρησιμοποιείται ο πολλαπλασιασμός πρώτων αριθμών ως ακολούθως:

1. Επιλέγουμε δύο μεγάλους πρώτους αριθμούς p και q
2. Υπολογίζουμε το $n = p * q$
3. Υπολογίζουμε το $\varphi(n) = (p - 1) * (q - 1)$

4. Επιλέγουμε έναν ακέραιο e με $3 \leq e \leq \varphi(n)$ τέτοιο ώστε να μην έχει κοινό παράγοντα με το $\varphi(n)$
5. Επιλέγουμε έναν ακέραιο d τέτοιο ώστε
6. $d * e \bmod \varphi(n) = 1$
7. Τα e, n δημοσιοποιούνται
8. Τα p, q, d, $\varphi(n)$ φυλάσσονται μυστικά

Για την παραγωγή του κρυπτογραφημένου κειμένου C από ένα μη κρυπτογραφημένο κείμενο M χρησιμοποιείται ο τύπος $C = M^e \bmod n$, ενώ για την αποκρυπτογράφηση του κρυπτογραφημένου μηνύματος χρησιμοποιείται ο τύπος $M = C^d \bmod n$.

6.2.3 Συμμετρικοί έναντι ασύμμετρων αλγορίθμων

Οι συμμετρικοί αλγόριθμοι μπορούν να υλοποιηθούν ιδιαίτερα αποτελεσματικά, υπάρχει όμως το πρόβλημα ότι πρέπει να υπάρχει ένα *διαμοιραζόμενο μυστικό*. Δύο εταίροι που ενέχονται σε επικοινωνία με συμμετρική κρυπτογράφηση πρέπει να έχουν το ίδιο μυστικό κλειδί. Για τον λόγο αυτό, η συμμετρική κρυπτογραφία είναι ακατάλληλη για να αποδεικνύεται η ταυτότητα του κάθε μέρους σε τρίτους, καθώς τουλάχιστον δύο οντότητες μοιράζονται το ίδιο κλειδί, και έτσι το κλειδί δεν είναι μονοσήμαντος σύνδεσμος προς συγκεκριμένο χρήστη. Ένα ακόμη πρόβλημα είναι η αναγκαιότητα μετάδοσης του κλειδιού διαμέσου του δικτύου αν οι εταίροι της επικοινωνίας βρίσκονται σε διαφορετικές τοποθεσίες. Η συμμετρική κρυπτογραφία μπορεί να χρησιμοποιηθεί για επίτευξη της αυθεντικότητας, σε συνδυασμό με έναν κεντρικό εξυπηρέτη αυθεντικοποίησης, ο οποίος φυλάσσει τα μυστικά για όλους τους εταίρους.

Το πρόβλημα του κοινού μυστικού δεν υφίσταται στην ασύμμετρη κρυπτογραφία. Κάθε οντότητα κατέχει ένα μοναδικό *ζεύγος κλειδιών*. Ένα από αυτά δημοσιοποιείται, ενώ το άλλο παραμένει στην αποκλειστική κατοχή και χρήση της οντότητας, πιθανώς αποθηκευόμενο σε κάποια έξυπνη κάρτα. Με το ιδιωτικό κλειδί παράγονται ψηφιακές υπογραφές, οι οποίες είναι δυνατόν να επαληθευτούν με το δημόσιο κλειδί, επιτυγχάνοντας έτσι την αυθεντικότητα. Χρησιμοποιώντας αντιστρέψιμους αλγόριθμους είναι δυνατόν να επιτευχθεί και η εμπιστευτικότητα, καθώς ακόμη και αν υποκλαπεί η επικοινωνία, μόνο ο κάτοχος του μυστικού κλειδιού μπορεί να αποκρυπτογραφήσει το περιεχόμενό της.

Η εφαρμογή ωστόσο της ασύμμετρης κρυπτογραφίας οδηγεί σε ένα σύνολο πρακτικών ζητημάτων. Ένα από αυτά είναι ότι βασίζονται σε περίπλοκες μαθηματικές θεωρίες και περιλαμβάνουν τη χρήση μεγάλων αριθμών, με αποτέλεσμα να είναι πιο αργοί από τους συμμετρικούς και συνήθως ακατάλληλοι για κρυπτογράφηση δεδομένων μεγάλου όγκου. Μια γενικώς παραδεκτή λύση είναι να κρυπτογραφούνται τα δεδομένα με συμμετρικούς αλγόριθμους και να ανταλλάσσονται τα σχετικά κλειδιά με ασύμμετρους.

6.2.4 Διακρίβωση δημόσιων κλειδιών

Ακόμη ένα θεμελιώδες πρόβλημα που ανακύπτει σε ανοικτά περιβάλλοντα με μεγάλο πλήθος εταίρων που δεν γνωρίζονται μεταξύ τους είναι η αυθεντικότητα των δημόσιων κλειδιών. Κατά την επαλήθευση μιας ψηφιακής υπογραφής, ο επαληθευτής πρέπει να είναι βέβαιος ότι το δημόσιο κλειδί που χρησιμοποιεί για την επαλήθευση της υπογραφής είναι πραγματικά το δημόσιο κλειδί του υποτιθέμενα υπογράφοντος. Με άλλα λόγια πρέπει να υπάρχει εμπιστοσύνη στην αντιστοιχία μεταξύ δημόσιου

κλειδιού και οντότητας. Χωρίς πρόσθετα μέτρα, θα πρέπει κάθε χρήστης να διακριβώνει εξωσυστημικά την αυθεντικότητα κάθε δημόσιου κλειδιού πριν επιλέξει να το εμπιστευθεί. Η πολυπλοκότητα του ζητήματος μπορεί να μειωθεί εισάγοντας τη δυνατότητα διακρίβωσης για τα δημόσια κλειδιά μέσω μιας τρίτης οντότητας την οποία εμπιστεύονται και τα δύο μέρη. Η τρίτη οντότητα, που καλείται επίσης *αρχή πιστοποίησης*, υπογράφει με το δικό της ιδιωτικό κλειδί τα δημόσια κλειδιά και τα αντίστοιχα ονόματα, προσθέτοντας επίσης κάποια επιπλέον στοιχεία, π.χ. περίοδο εγκυρότητας. Το κομμάτι αυτό δεδομένων που έχει υπογραφεί από την αρχή πιστοποίησης καλείται *πιστοποιητικό*. Το πιστοποιητικό μπορεί να επαληθευτεί χρησιμοποιώντας το δημόσιο κλειδί της αρχής πιστοποίησης. Με τον τρόπο αυτό ο κάθε εταίρος μπορεί να διακριβώσει την ταυτότητα του άλλου, επαληθεύοντας πρώτα την ψηφιακή υπογραφή του εταίρου με το δημόσιο κλειδί του εταίρου και κατόπιν επαληθεύοντας την αυθεντικότητα του δημόσιου κλειδιού του εταίρου μέσα από την ψηφιακή υπογραφή του πιστοποιητικού, χρησιμοποιώντας το δημόσιο κλειδί της αρχής πιστοποίησης. Ο κάθε ένας έτσι αρκεί να έχει στην κατοχή του και να εμπιστεύεται μόνο το δημόσιο κλειδί της αρχής πιστοποίησης (και φυσικά το δικό του ιδιωτικό κλειδί), μειώνοντας έτσι δραστικά το πλήθος των κλειδιών που πρέπει κανείς να εμπιστεύεται. Στη γενική περίπτωση ένα πιστοποιητικό περιέχει τις ακόλουθες πληροφορίες:

- Έκδοση και αριθμό σειράς
- Το όνομα του εκδότη
- Το όνομα του υποκειμένου και άλλες τυχόν επεκτάσεις (διεύθυνση οικίας, εργασίας, αριθμό ταυτότητας κ.λπ.)
- Το σκοπό του πιστοποιητικού (ένδειξη αν το υποκείμενο δρα και ως αρχή πιστοποίησης)
- Το δημόσιο κλειδί του υποκειμένου
- Την περίοδο εγκυρότητας του πιστοποιητικού
- Την υπογραφή της αρχής διαχείρισης πιστοποιητικών

Σε πολυπληθείς κοινότητες οντοτήτων, μία και μόνη αρχή πιστοποίησης δεν είναι επαρκής, καθώς θα αποτελούσε σημείο συμφόρησης για όλο το δίκτυο και σημείο συνολικής αποτυχίας. Επίσης, βάσει των κανόνων της «ελεύθερης αγοράς» η κάθε εταιρεία ή άτομο θα πρέπει να μπορεί να επιλέγει ποιος θα είναι ο υπεύθυνος για τις δικές του λειτουργίες σε σχέση με τα δημόσια κλειδιά δηλ. (α) ποιος θα εκδώσει το δικό του πιστοποιητικό και (β) σε ποιον θα απευθύνει τις ερωτήσεις του σχετικά με το αν κάποιο πιστοποιητικό είναι έγκυρο. Σε ένα σχήμα με πολλαπλές αρχές πιστοποίησης, αν θεωρήσουμε την περίπτωση που το υποκείμενο Χ πιστοποιείται από και εμπιστεύεται για πιστοποίηση την αρχή Α και το υποκείμενο Υ πιστοποιείται από και εμπιστεύεται για πιστοποίηση την αρχή Β, τίθεται το θέμα του ποια αξία έχουν για τον Υ τα πιστοποιητικά του Α (δηλ. πιστοποιητικά που έχουν εκδοθεί από αρχή *διαφορετική* από αυτή που έχει επιλέξει ο Υ).

Η πρώτη προσέγγιση πάνω στο ζήτημα αυτό είναι να πρέπει κάθε υποκείμενο να δηλώνει ρητώς ποιες αρχές πιστοποίησης εμπιστεύεται ως προς την έκδοση πιστοποιητικών. Η προσέγγιση αυτή είναι μη διαχειρίσιμη, λόγω του μεγάλου πλήθους των αρχών πιστοποίησης (θεωρητικά μπορούν να υπάρχουν μερικές δεκάδες έως εκατοντάδες σε κάθε χώρα) καθώς και της ανεπάρκειας γνώσεων των χρηστών (ο

κάθε χρήστης δεν μπορεί να γνωρίζει όλες τις παραμέτρους αδειοδότησης και λειτουργίας όλων των αρχών).

Η δεύτερη προσέγγιση είναι αυτή της *ομότιμης διασταυρούμενης πιστοποίησης*. Βάσει της προσέγγισης αυτής οι αρχές πιστοποίησης εγκαθιστούν μεταξύ τους μονόδρομες ή αμφίδρομες σχέσεις εμπιστοσύνης σε ομότιμη βάση. Η μονόδρομη σχέση εμπιστοσύνης ορίζει ότι η αρχή πιστοποίησης Α πιστοποιεί τη Β ως έγκυρη αρχή πιστοποίησης, ενώ η αμφίδρομη καλύπτει και την αντίστροφη κατεύθυνση (και η Β πιστοποιεί την Α). Οι χρήστες και πάλι εμπιστεύονται τις επιμέρους αρχές πιστοποίησης, ωστόσο κατά τη διακρίβωση εγκυρότητας των πιστοποιητικών αξιοποιούνται οι σχέσεις εμπιστοσύνης. Για να θεωρηθεί από τον χρήστη Χ που εμπιστεύεται την αρχή Α έγκυρο ένα πιστοποιητικό που έχει εκδοθεί από την αρχή Β θα πρέπει να υπάρχει μία *αλυσίδα εμπιστοσύνης που να ξεκινάει από την Α και να καταλήγει στη Β*. Με άλλα λόγια θα πρέπει η αρχή Α να πιστοποιεί άμεσα τη Β ως «έγκυρη αρχή πιστοποίησης» ή να πιστοποιεί την αρχή Γ που με τη σειρά της πιστοποιεί τη Β κ.ο.κ. Κατά περίπτωση είναι δυνατόν να τίθενται και όρια στο μήκος της αλυσίδας πιστοποίησης, π.χ. να μην περιλαμβάνει πάνω από τρεις ενδιάμεσους κόμβους.

Μία τρίτη προσέγγιση στο ζήτημα είναι αυτή της *ιεραρχικής διασταυρούμενη πιστοποίησης*. Βάσει της προσέγγισης αυτής, οι αρχές πιστοποίησης οργανώνονται σε ιεραρχίες, με κάθε μία να πιστοποιεί τις υφιστάμενές της ως αρχές πιστοποίησης και ο κάθε χρήστης εμπιστεύεται την *πρωταρχική αρχή πιστοποίησης*, στη ρίζα της ιεραρχίας. Τα ίδια τα πιστοποιητικά εκδίδονται όχι (κατ' ανάγκην) από την πρωταρχική αρχή πιστοποίησης, αλλά από (κυρίως) από τις αρχές πιστοποίησης χαμηλότερα στην ιεραρχία. Οι χρήστες εμπιστεύονται ένα πιστοποιητικό διότι έχει εκδοθεί από μία αρχή για την οποία εγγυάται (άμεσα ή έμμεσα) η πρωταρχική αρχή πιστοποίησης (την οποία και εμπιστεύονται). Η διακρίβωση ενός πιστοποιητικού πρακτικά συνίσταται στην προσπάθεια να βρούμε αν υπάρχει μονοπάτι πιστοποίησης που να ξεκινάει από τη ρίζα και να φτάνει ως την αρχή που εξέδωσε το πιστοποιητικό.

Μία τελική προσέγγιση είναι το υβριδικό μοντέλο, όπου έχουμε στην ένα πλήθος *πρωταρχικών αρχών πιστοποίησης* που σχηματίζουν ανεξάρτητα μεταξύ τους δένδρα αρχών πιστοποίησης., παράλληλα όμως οι πρωταρχικές αρχές πιστοποίησης εγκαθιδρύουν μεταξύ τους σχέσεις εμπιστοσύνης.

Ανεξάρτητα από την προσέγγιση, είναι απαραίτητο ο κάθε χρήστης να εμπιστεύεται την αρχή πιστοποίησης που έχει επιλέξει και για να είναι αποτελεσματικό αυτό είναι απαραίτητο να έχει το δημόσιο κλειδί της. Η συνήθης προσέγγιση είναι να *εγκαθίσταται* αυτό το κλειδί στον υπολογιστή είτε με εξωσυστημική μεταφορά είτε από ασφαλές κανάλι επικοινωνίας, που συνήθως συμπληρώνεται και από τον οπτικό έλεγχο του χειριστή σε διάφορα στοιχεία του κλειδιού (άθροισμα ελέγχου, ονόματα, διευθύνσεις κ.λπ.). Παράλληλα, στον υπολογιστή εγκαθίσταται και το *ιδιωτικό κλειδί της οντότητας*, εφ' όσον αυτό υπάρχει. Το ιδιωτικό κλειδί πρέπει να είναι προστατευμένο όχι μόνο από τροποποίηση αλλά και από διαρροή, καθ' όσον αν διαρρεύσει οποιοσδήποτε κάτοχός του μπορεί να αντιποιηθεί την οντότητα.

6.2.5 Ανάκληση κλειδιών

Μολονότι τα πιστοποιητικά που εκδίδονται από τις αρχές πιστοποίησης εμπεριέχουν περίοδο ισχύος, είναι δυνατόν να χρειασθεί κάποιο πιστοποιητικό να *ανακληθεί* προ της λήξεως της ισχύος του, για ένα πλήθος λόγων:

1. αν το ιδιωτικό κλειδί του χρήστη έχει διαρρεύσει, το δημόσιο πρέπει να ανακληθεί διότι μπορούν να κυκλοφορούν έγγραφα με πλαστή υπογραφή.
2. κάποιο από τα στοιχεία που περιέχονται στο πιστοποιητικό, έχει αλλάξει, καθιστώντας άκυρο το πιστοποιητικό συνολικά.
3. το πιστοποιητικό της αρχής πιστοποίησης θεωρείται ελαττωμένης ασφάλειας.
4. ο χρήστης παραβίασε τους κανόνες της αρχής πιστοποίησης.

Η αρχή πιστοποίησης ακυρώνει ένα πιστοποιητικό τοποθετώντας το σε μία λίστα *ανακληθέντων πιστοποιητικών*, η οποία δημοσιοποιείται για να είναι προσπελάσιμη στους ενδιαφερόμενους. Για να έχει αποτέλεσμα η λίστα ανάκλησης κλειδιών θα πρέπει το λογισμικό που χρησιμοποιείται να *μην* χρησιμοποιεί αποθηκευμένα πιστοποιητικά (cached certificates) χωρίς να επιβεβαιώνει ότι τα πιστοποιητικά αυτά δεν έχουν ήδη ανακληθεί.

6.2.6 Διαχείριση κλειδιών

Η διαχείριση κλειδιών είναι η διαδικασία παραγωγής κατάλληλων κλειδιών για ίδια χρήση και διάθεσής τους στους εταίρους με τρόπο ώστε να είναι δυνατόν να επιβεβαιωθούν και να χρησιμοποιηθούν με τον προτιθέμενο τρόπο, ενώ παράλληλα κανείς δεν μπορεί να τα καταχρασθεί. Αυτό σημαίνει ότι τα συμμετρικά κλειδιά, καθώς και τα μυστικά κλειδιά της ασύμμετρης κρυπτογραφίας πρέπει να μεταδίδονται εμπιστευτικά (με κρυπτογράφηση ή εξωσυστημική μετάδοση). Για τα δημόσια κλειδιά της ασύμμετρης κρυπτογραφίας πρέπει να είναι δυνατόν να επαληθευτεί η αυθεντικότητα του κλειδιού.

Οι ίδιες απαιτήσεις ισχύουν για την επικοινωνία μεταξύ μιας αρχής πιστοποίησης και ενός χρήστη για τη διακρίβωση ενός δημόσιου κλειδιού. Είναι απαραίτητη η ανταλλαγή κλειδιών μεταξύ του χρήστη και της αρχής πιστοποίησης. Υπάρχουν δύο κύριες εναλλακτικές προσεγγίσεις για τη λειτουργία της διακρίβωσης δημόσιων κλειδιών, οι οποίες έχουν διαφορετικά χαρακτηριστικά ασφάλειας:

1. Η αρχή πιστοποίησης δημιουργεί το ζεύγος κλειδιών ασύμμετρης κρυπτογραφίας, πιστοποιεί το δημόσιο κλειδί και δίνει στον χρήστη το μυστικό κλειδί, το πιστοποιητικό (που περιλαμβάνει το δημόσιο κλειδί) και ενδεχομένως άλλα πιστοποιητικά που συνδέουν το σύνολο των κλειδιών με το κλειδί ρίζας.
2. Ο χρήστης δημιουργεί μόνος του το ζεύγος των κλειδιών και αποστέλλει το δημόσιο κλειδί στην αρχή πιστοποίησης, η οποία το πιστοποιεί και δίνει στον χρήστη το πιστοποιητικό (που περιλαμβάνει το δημόσιο κλειδί) και ενδεχομένως άλλα πιστοποιητικά που συνδέουν το σύνολο των κλειδιών με το κλειδί ρίζας.

Στην πρώτη περίπτωση υφίσταται η ανάγκη για εμπιστευτική επικοινωνία, στη δεύτερη όχι. Η εμπιστευτική επικοινωνία από την αρχή πιστοποίησης προς τον χρήστη είναι εύκολη, με τη δημιουργία μιας έξυπνης κάρτας που περιέχει τα απαραίτητα στοιχεία και που παραδίδεται αυτοπροσώπως στον χρήστη. Αν χρησιμοποιείται ηλεκτρονική μετάδοση, πρέπει να ανταλλαχθεί μεταξύ χρήστη και αρχής πιστοποίησης ένα συμμετρικό κλειδί κρυπτογράφησης με εξωσυστημικό τρόπο. Στην πρώτη περίπτωση η αρχή πιστοποίησης έχει πλήρη έλεγχο για την ποιότητα των παραγόμενων κλειδιών, στη δεύτερη ο κάθε χρήστης είναι υπεύθυνος για το δικό του ζεύγος κλειδιών. Σε κάθε περίπτωση, είναι απαραίτητη μια

εξωσυστημική ανταλλαγή πληροφορίας, διαμέσου της οποίας η αρχή πιστοποίησης θα βεβαιωθεί ότι η πιστοποιούμενη οντότητα είναι πραγματικά αυτή που ισχυρίζεται ότι είναι.

6.2.7 Δημόσιοι κατάλογοι

Η χρήση της κρυπτογραφίας δημόσιου κλειδιού καθιστά αναγκαία τη γνώση των δημόσιων κλειδιών των άλλων και την ύπαρξη εμπιστοσύνης προς αυτά. Οι δημόσιοι κατάλογοι, όπως οι κατάλογοι τύπου X500, μπορούν να υποστηρίξουν αυτόν τον στόχο. Οι χρήστες και οι αρχές πιστοποίησης, όταν ενέχονται, έχουν τις ακόλουθες απαιτήσεις όταν χρησιμοποιούν υπηρεσίες ασφάλειας με βάση την κρυπτογραφία δημόσιου κλειδιού:

- Οι χρήστες θέλουν να δημοσιοποιούν τα δικά τους δημόσια κλειδιά, να μπορούν να προσπελάσουν τα πιστοποιητικά των άλλων και τις λίστες ανάκλησης πιστοποιητικών, προκειμένου να διακριβώνουν ότι κάποιο πιστοποιητικό είναι έγκυρο και δεν έχει ανακληθεί.
- Οι αρχές πιστοποίησης επιθυμούν να δημοσιοποιούν τα δημόσια κλειδιά και τις λίστες ανάκλησης με τρόπο που να τα καθιστά διαθέσιμα στην ενδιαφερόμενη κοινότητα.
- Οι αρχές πιστοποίησης επιθυμούν να ανταλλάσσουν κλειδιά αμοιβαίας πιστοποίησης με άλλες αρχές πιστοποίησης και να δημοσιοποιούν τα δικά τους δημόσια κλειδιά.

Οι δημόσιοι κατάλογοι βοηθούν σ' αυτή την κατεύθυνση ως παροχείς πληροφορίας στη βασική υποδομή ασφάλειας, αποθηκεύοντας δηλαδή τη σχετική πληροφορία και επιτρέποντας την ευέλικτη αναζήτησή της. Παράλληλα δε, οι κατάλογοι είναι και *χρήστες μηχανισμών ασφάλειας,* καθώς ενσωματώνουν τεχνικές για την προστασία της πληροφορίας του κατάλογου, των επικοινωνιών με άλλες οντότητες καθώς και των πόρων των υπολογιστικών συστημάτων που τους φιλοξενούν. Για τους λόγους αυτούς, οι προδιαγραφές του δημόσιου κατάλογου τύπου X500 έχουν πλαισιωθεί από το *Πλαίσιο Αυθεντικότητας X509*, το οποίο συμπεριλαμβάνει ισχυρούς μηχανισμούς διακρίβωσης ταυτότητας.

6.2.8 Ψηφιακές υπογραφές

Οι ψηφιακές υπογραφές είναι μία τεχνική που επιτρέπει την εξακρίβωση ότι ένα συγκεκριμένο κείμενο προέρχεται από έναν συγκεκριμένο αποστολέα, υποστηρίζοντας έτσι τη διάσταση της *αυθεντικότητας*. Προκειμένου να εξακριβωθεί η αυθεντικότητα ενός εγγράφου λαμβάνουν χώρα τα κάτωθι βήματα:

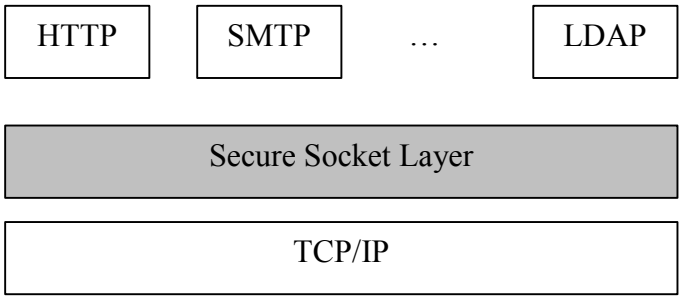
1. Αρχικά στο μήνυμα εφαρμόζεται μία *συνάρτηση κερματισμού*, η οποία υπολογίζει ένα πλήθος από bits βάσει του περιεχομένου του μηνύματος. Τα bits αυτά ονομάζονται digest του μηνύματος.
2. Το digest του μηνύματος κρυπτογραφείται με το ιδιωτικό κλειδί του αποστολέα παράγοντας την *ψηφιακή υπογραφή του μηνύματος.*
3. Η ψηφιακή υπογραφή επισυνάπτεται στο μήνυμα και αποστέλλεται μαζί με αυτό.
4. Ο παραλήπτης του μηνύματος αρχικά διαχωρίζει την ψηφιακή υπογραφή από το καθ' εαυτό μήνυμα και εφαρμόζει στο καθ' εαυτό μήνυμα την ίδια συνάρτηση κερματισμού, παράγοντας και αυτός ένα digest.

5. Ακολούθως αποκρυπτογραφεί την ψηφιακή υπογραφή με το δημόσιο κλειδί του υπογράψαντος παράγοντας ένα digest'.
6. Τέλος, τα digest και digest' των βημάτων 5 και 6 συγκρίνονται. Μόνο αν αυτά είναι ίσα το μήνυμα θεωρείται αυθεντικό.

Σημειώνεται ότι καθώς η ανωτέρω διαδικασία περιλαμβάνει τόσο τα κλειδιά (δημόσιο και ιδιωτικό) του αποστολέα όσο και το ίδιο το περιεχόμενο του μηνύματος (βάσει του οποίου υπολογίζεται το digest), η τεχνική αυτή διακριβώνει τόσο το γεγονός ότι το μήνυμα εστάλη από τον συγκεκριμένο αποστολέα όσο και ότι το περιεχόμενό του παραμένει αναλλοίωτο από τη στιγμή που υπεγράφη ψηφιακά.

6.2.9 Το πρωτόκολλο SSL

Το πρωτόκολλο TCP/IP είναι το κυρίαρχο για μετάδοση και δρομολόγηση πληροφοριών στα πλαίσια του διαδικτύου. Άλλα πρωτόκολλα, όπως π.χ. το SMTP (διακίνησης αλληλογραφίας), το HTTP (διακίνησης υπερκειμένου), LDAP (υπηρεσίες καταλόγου) κ.λπ. εκτελούνται «πάνω» από το TCP/IP, με την έννοια ότι χρησιμοποιούν το TCP/IP για τη διακίνηση δεδομένων. Βάσει της στρωματοποιημένης αρχιτεκτονικής ISO/OSI για τα δίκτυα υπολογιστών, το πρωτόκολλο TCP/IP καλύπτει τα επίπεδα 3 και 4 (δρομολόγησης και μεταφοράς), ενώ τα SMTP, HTTP, LDAP κ.λπ. λειτουργούν στο επίπεδο 7 (εφαρμογής). Το πρωτόκολλο SSL παρεμβάλλεται μεταξύ των πρωτοκόλλων εφαρμογής και του TCP/IP (όπως φαίνεται στο σχήμα που ακολουθεί), χρησιμοποιώντας το πρωτόκολλο TCP/IP για λογαριασμό των πρωτοκόλλων υψηλότερου επιπέδου και παρέχοντας την εξής επιπρόσθετη λειτουργικότητα:



- Πιστοποίηση του εξυπηρέτη προς τον εξυπηρετούμενο
- Πιστοποίηση του εξυπηρετούμενου προς τον εξυπηρέτη
- Κρυπτογράφηση της επικοινωνίας

Η λειτουργικότητα αυτή είναι θεμελιώδης για ασφαλή επικοινωνία στο διαδίκτυο για τους κάτωθι λόγους:

- *Πιστοποίηση του εξυπηρέτη προς τον εξυπηρετούμενο*. Ο εξυπηρετούμενος μπορεί να διακριβώσει την ταυτότητα του εξυπηρέτη. Το λογισμικό του εξυπηρετούμενου μπορεί να χρησιμοποιήσει ένα σύνολο από τεχνικές κρυπτογραφίας δημόσιου κλειδιού για να ελέγξει ότι το πιστοποιητικό και η δημόσια ταυτότητα του εξυπηρέτη είναι έγκυρα και έχουν εκδοθεί από μία αρχή πιστοποίησης την οποία ο εξυπηρετούμενος εμπιστεύεται. Η σημασία του ελέγχου μπορεί να είναι μεγάλη, αν π.χ. αποστέλλονται αριθμοί πιστωτικών καρτών ή απόρρητα δεδομένα και πρέπει να εξασφαλισθεί ότι μόνο ο προτιθέμενος εξυπηρέτης τα λαμβάνει.

- *Πιστοποίηση του εξυπηρετούμενου προς τον εξυπηρέτη*. Ο εξυπηρέτης διακριβώνει την ταυτότητα του χρήστη, με τις ίδιες τεχνικές που χρησιμοποιούνται για την πιστοποίηση του εξυπηρέτη προς τον εξυπηρετούμενο. Έτσι ελέγχεται ότι το πιστοποιητικό και η δημόσια ταυτότητα του εξυπηρετούμενου είναι έγκυρα και έχουν εκδοθεί από μία αρχή πιστοποίησης την οποία ο εξυπηρέτης εμπιστεύεται. Η σημασία του ελέγχου μπορεί να είναι μεγάλη, αν π.χ. αποστέλλονται εμπιστευτικά δεδομένα και ο εξυπηρέτης θέλει να εξασφαλίσει ότι μόνο ο προτιθέμενος παραλήπτης τα λαμβάνει.

- *Κρυπτογράφηση της επικοινωνίας*. Το πρωτόκολλο SSL κρυπτογραφεί όλη την επικοινωνία μεταξύ εξυπηρέτη και εξυπηρετούμενου. Τα δεδομένα κρυπτογραφούνται από τον αποστολέα και αποκρυπτογραφούνται από τον παραλήπτη, επιτυγχάνοντας έτσι υψηλό βαθμό εμπιστευτικότητας. Η εμπιστευτικότητα είναι σημαντική και για τους δύο ενεχόμενους σε οποιαδήποτε ιδιωτική συναλλαγή. Επιπρόσθετα, όλα τα δεδομένα που αποστέλλονται μέσω μιας κρυπτογραφημένης σύνδεσης SSL προστατεύονται με ένα μηχανισμό για ανίχνευση παρεμβάσεων, εντοπισμό δηλαδή προσπαθειών για αλλοίωσή τους κατά τη μεταφορά.

Το πρωτόκολλο SSL περιλαμβάνει δύο επί μέρους πρωτόκολλα: το *πρωτόκολλο εγγραφών SSL* και το *πρωτόκολλο χειραψίας SSL*. Το πρωτόκολλο εγγραφών SSL καθορίζει τη μορφή που χρησιμοποιείται για τη μετάδοση των δεδομένων. Το πρωτόκολλο χειραψίας SSL ορίζει μία ακολουθία μηνυμάτων που πρέπει να ανταλλαχθούν μεταξύ εξυπηρέτη και εξυπηρετούμενου προκειμένου να εγκαθιδρυθεί μία σύνδεση μεταξύ τους. Τα μηνύματα αυτά ανταλλάσσονται με στόχο:

- Να πιστοποιηθεί ο εξυπηρέτης στον εξυπηρετούμενο
- Να επιτραπεί στον εξυπηρέτη και στον εξυπηρετούμενο να συμφωνήσουν πάνω στους αλγόριθμους κρυπτογραφίας που θα χρησιμοποιηθούν για την επικοινωνία.
- Προαιρετικά, να πιστοποιηθεί ο εξυπηρετούμενος στον εξυπηρέτη
- Να δημιουργηθούν «διαμοιραζόμενα μυστικά» μέσω τεχνικών κρυπτογραφίας δημόσιου κλειδιού. Τα «διαμοιραζόμενα μυστικά» θα χρησιμοποιηθούν για την κρυπτογράφηση της επικοινωνίας.
- Εγκαθίδρυση του κρυπτογραφημένου διαύλου επικοινωνίας

6.2.9.1 Η χειραψία του πρωτοκόλλου SSL

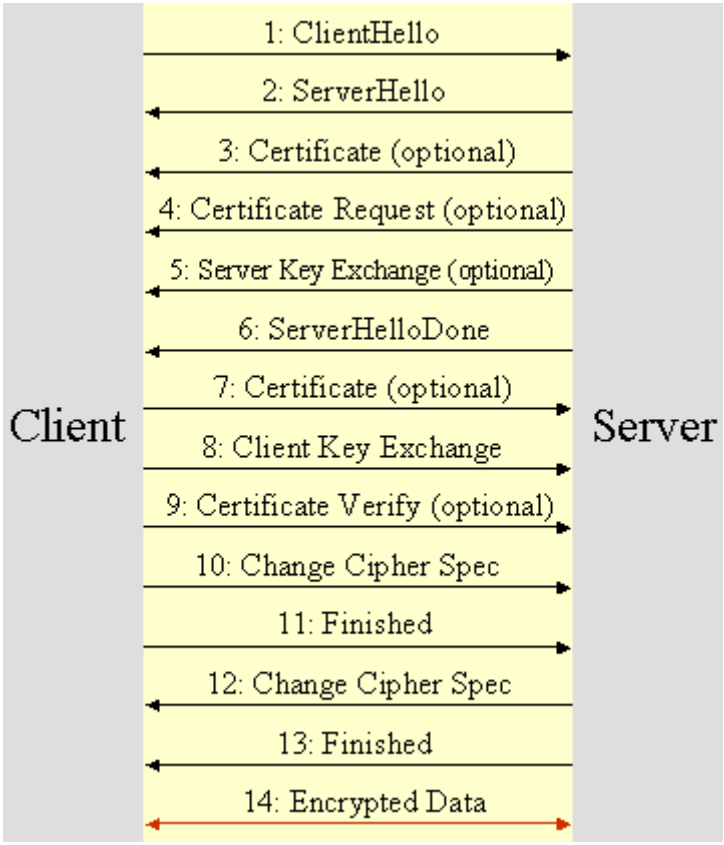
Το πρωτόκολλο SSL χρησιμοποιεί έναν συνδυασμό κρυπτογραφίας δημοσίου κλειδιού και συμμετρικής κρυπτογραφίας. Η συμμετρική κρυπτογραφία είναι ταχύτερη, αλλά η κρυπτογραφία δημοσίου κλειδιού παρέχει καλύτερες τεχνικές διακρίβωσης ταυτότητας. Μία σύνοδος SSL ξεκινά πάντα με ανταλλαγή μηνυμάτων που ονομάζεται *χειραψία SSL*. Η χειραψία επιτρέπει στον εξυπηρετούμενο να πιστοποιήσει την ταυτότητα του εξυπηρέτη και στη συνέχεια υποστηρίζει τη συνεργασία μεταξύ εξυπηρέτη και εξυπηρετούμενου για δημιουργία συμμετρικών κλειδιών που θα χρησιμοποιηθούν για την κρυπτογράφηση, αποκρυπτογράφηση και ανίχνευση παρεμβάσεων στην κρυπτογραφημένη επικοινωνία. Η χειραψία περιλαμβάνει επίσης προαιρετικά την πιστοποίηση της ταυτότητας του εξυπηρετούμενου από τον εξυπηρέτη.

Τα βήματα που περιλαμβάνονται κατά τη διάρκεια της χειραψίας είναι γενικώς τα ακόλουθα (είναι δυνατόν να διαφοροποιούνται λίγο ανάλογα με τις τεχνικές ανταλλαγής κλειδιών):

1. *Client Hello*. ο εξυπηρετούμενος αποστέλλει στον εξυπηρέτη τον αριθμό έκδοσης του SSL του εξυπηρετούμενου, μία λίστα υποστηριζόμενων αλγόριθμων κρυπτογράφησης και αντιστοίχων μεγεθών κλειδιών, την ταυτότητα της συνόδου κ.τ.λ.
2. *Server Hello*. Ο εξυπηρέτης επιλέγει τον πιο κατάλληλο αλγόριθμο κρυπτογράφησης που υποστηρίζει τόσο αυτός όσο και ο εξυπηρετούμενος και αποστέλλει την ταυτότητα του στον εξυπηρετούμενο. Η αποστελλόμενη ταυτότητα εμπεριέχει και το μήκος των σχετικών κλειδιών.
3. *Certificate*. (προαιρετικό) Αν ο αλγόριθμος κρυπτογράφησης που θα επιλεχθεί απαιτεί πιστοποίηση του εξυπηρέτη, ο εξυπηρέτης αποστέλλει το πιστοποιητικό του στον εξυπηρετούμενο. Το πιστοποιητικό περιέχει το δημόσιο κλειδί του εξυπηρέτη. Βάσει του πιστοποιητικού ο εξυπηρετούμενος μπορεί να διακριβώσει την ταυτότητα του εξυπηρέτη. Αν η διακρίβωση αποτύχει (το πιστοποιητικό είναι άκυρο, έχει λήξει, δεν αντιστοιχεί στον εξυπηρέτη που το έστειλε ή δεν μπορεί να επαληθευθεί από έμπιστη αρχή πιστοποίησης), το σφάλμα σημειώνεται και ο εξυπηρετούμενος πρέπει να αποφασίσει αν θα συνεχίσει ή όχι σε μία κρυπτογραφημένη μεν, μη πιστοποιημένη δε επικοινωνία. Η απόφαση συνήθως λαμβάνεται με προτροπή του χρήστη.
4. *Certificate request.* (προαιρετικό) Αν ο εξυπηρετούμενος ζητά έναν πόρο στον εξυπηρέτη που απαιτεί πιστοποίηση εξυπηρετούμενου, ο εξυπηρέτης αποστέλλει ένα μήνυμα με το οποίο ζητά το πιστοποιητικό του εξυπηρετούμενου.
5. *Server key exchange.* (προαιρετικό)Το μήνυμα αυτό αποστέλλεται μόνο αν το πιστοποιητικό του εξυπηρέτη (που περιέχει το δημόσιο κλειδί του εξυπηρέτη) δεν είναι επαρκές για την ανταλλαγή κλειδιών που θα ακολουθήσει.
6. *Server Hello Done.* Με το μήνυμα αυτό ο εξυπηρέτης υποδεικνύει ότι έχει τελειώσει την προκαταρκτική φάση εγκαθίδρυσης της συνόδου.
7. *Certificate.* (προαιρετικό) Αν ο εξυπηρετούμενος έχει λάβει μήνυμα *Certificate request*, με το μήνυμα αυτό αποστέλλεται το πιστοποιητικό του εξυπηρετούμενου προς τον εξυπηρέτη. Ο εξυπηρέτης θα προβεί στη διακρίβωση της ταυτότητας του εξυπηρετούμενου κατά τρόπο αντίστοιχο με αυτόν που χρησιμοποιεί ο εξυπηρετούμενος για διακρίβωση της ταυτότητας του εξυπηρέτη.
8. *Client key exchange*. Βάσει των μέχρι τώρα ανταλλαχθέντων δεδομένων, ο εξυπηρετούμενος δημιουργεί το *προκαταρκτικό μυστικό* (premaster secret) για τη συγκεκριμένη σύνοδο, το κρυπτογραφεί με το δημόσιο κλειδί του εξυπηρέτη και το αποστέλλει σ' αυτόν.
9. *Certificate verify*. (προαιρετικό) Αν ο εξυπηρέτης έχει ζητήσει το πιστοποιητικό του εξυπηρετούμενου, τι μήνυμα αυτό του επιτρέπει να ολοκληρώσει τη διαδικασία επαλήθευσης του πιστοποιητικού.
10. *Change cipher spec*. Ο εξυπηρετούμενος πληροφορεί τον εξυπηρέτη ότι είναι έτοιμος να μεταβεί σε ασφαλή επικοινωνία.

11. *Finished*. Ο εξυπηρετούμενος πληροφορεί τον εξυπηρέτη ότι έχει τελειώσει το δικό του τμήμα της χειραψίας.
12. *Change cipher spec*. Ο εξυπηρέτης πληροφορεί τον εξυπηρετούμενο ότι είναι έτοιμος να μεταβεί σε ασφαλή επικοινωνία.
13. *Finished*. Ο εξυπηρέτης πληροφορεί τον εξυπηρετούμενο ότι έχει τελειώσει το δικό του τμήμα της χειραψίας.

Η όλη διαδικασία της χειραψίας απεικονίζεται στο σχήμα που ακολουθεί.



6.2.9.2 Η ανταλλαγή δεδομένων στο πρωτόκολλο SSL

Μετά το πέρας της χειραψίας ο εξυπηρέτης και ο εξυπηρετούμενος επικοινωνούν μόνο με κρυπτογραφημένα δεδομένα. Η κρυπτογράφηση γίνεται με *συμμετρικό αλγόριθμο*, προκειμένου να επιτυγχάνεται μεγαλύτερη ταχύτητα στην επικοινωνία. Ο συγκεκριμένος συμμετρικός αλγόριθμος που θα χρησιμοποιηθεί έχει αποφασισθεί στο βήμα 2 της χειραψίας (Server hello), όταν ο εξυπηρέτης έχει επιλέξει τον πιο κατάλληλο αλγόριθμο και έχει ενημερώσει τον εξυπηρετούμενο σχετικά.

Για τη δημιουργία των κατάλληλων κλειδιών για τον συμμετρικό αλγόριθμο, αξιοποιείται το *προκαταρκτικό μυστικό* που αναφέρθηκε στο βήμα 8 (client key exchange) της χειραψίας. Ανακαλέστε ότι ο εξυπηρετούμενος στο βήμα αυτό έχει δημιουργήσει κάποια δεδομένα, τα έχει κρυπτογραφήσει με το δημόσιο κλειδί του εξυπηρέτη (το οποίο έχει εξάγει από το επαληθευμένο πιστοποιητικό του εξυπηρέτη) και τα έχει αποστείλει στον εξυπηρέτη. Ο εξυπηρέτης, ως μόνος κάτοχος του αντίστοιχου ιδιωτικού κλειδιού, είναι ο μόνος που μπορεί να αποκρυπτογραφήσει το μήνυμα αυτό και να εξάγει το *προκαταρκτικό μυστικό* που απέστειλε ο

εξυπηρετούμενος. Έτσι, στο σημείο αυτό εξυπηρέτης και εξυπηρετούμενος κατέχουν από κοινού ένα *διαμοιραζόμενο μυστικό*, το οποίο δεν είναι γνωστό σε κανέναν άλλο. Στη συνέχεια, εξυπηρέτης και εξυπηρετούμενος προβαίνουν παράλληλα σε μετασχηματισμούς πάνω στο διαμοιραζόμενο μυστικό, έτσι ώστε να καταλήξουν στο τελικό *κλειδί συνόδου*, το κλειδί δηλαδή για τον συμμετρικό αλγόριθμο κρυπτογράφησης που θα χρησιμοποιηθεί για τη συγκεκριμένη σύνοδο. Το κλειδί συνόδου είναι το αποτέλεσμα μιας συνάρτησης

SessionKey = genKey(premasterSecret, cipher, keyLen)

όπου *premasterSecret* είναι το προκαταρκτικό μυστικό, *cipher* είναι ο αλγόριθμος κρυπτογράφησης και *keyLen* το μήκος των κλειδιών που θα χρησιμοποιηθούν στον αλγόριθμο. Με δεδομένο ότι τόσο ο εξυπηρέτης όσο και ο εξυπηρετούμενος χρησιμοποιούν τις ίδιες παραμέτρους στη συνάρτηση αυτή (το προκαταρκτικό μυστικό έχει ανταλλαχθεί στο βήμα 8 της χειραψίας ενώ ο αλγόριθμος κρυπτογράφησης και το μήκος του κλειδιού έχουν ανταλλαχθεί στο βήμα 2 της χειραψίας), το αποτέλεσμα που λαμβάνουν είναι το ίδιο, συνεπώς μπορούν να το χρησιμοποιήσουν ως κλειδί σε συμμετρικό αλγόριθμο κρυπτογραφίας.

6.2.10 Πού γίνεται η κρυπτογράφηση

Με δεδομένο ότι οι δύο υπολογιστές που επικοινωνούν μεταξύ τους μπορούν να βρίσκονται σε διαφορετικά υποδίκτυα, τίθεται ένα ζήτημα επιλογής του αν η κρυπτογράφηση θα γίνεται απ' άκρου εις άκρον ή σε κάθε επικοινωνιακή γραμμή ξεχωριστά. Στην πρώτη περίπτωση ο υπολογιστής που αποστέλλει την πληροφορία την κρυπτογραφεί και η πληροφορία αποκρυπτογραφείται από τον τελικό της παραλήπτη. Στη δεύτερη προσέγγιση η πληροφορία κρυπτογραφείται κατά την είσοδό της σε κάθε επικοινωνιακή γραμμή (συνήθως από τον σχετικό δρομολογητή) και αποκρυπτογραφείται κατά την έξοδό της από αυτήν (από τον συζυγή δρομολογητή).

Η πρώτη προσέγγιση απαιτεί ένα κλειδί για κάθε επικοινωνιακό εταίρο και είναι ευάλωτη σε επιθέσεις βασισμένες σε ανάλυση κυκλοφορίας (δηλ. παρακολούθηση του ποιος επικοινωνεί με ποιον). Στη δεύτερη προσέγγιση κάθε επικοινωνιακός κόμβος πρέπει να γνωρίζει μόνο τους άμεσους γείτονές του, αλλά η πληροφορία είναι εκτεθειμένη σε κάθε ενδιάμεσο επικοινωνιακό κόμβο. Τέλος απαιτείται πρόσβαση σε όλα τα ενδιάμεσα κανάλια, κάτι που δεν είναι πάντα εφικτό, π.χ. μία εταιρία δεν θα έχει πρόσβαση στα δημόσια κανάλια επικοινωνίας του ΟΤΕ.

6.2.11 Τύποι «επιθέσεων» σε κρυπτογραφικά συστήματα

Οι παραδοσιακές μέθοδοι κρυπτανάλυσης περιλαμβάνουν μερικούς ενδιαφέροντες συνδυασμούς αναλυτικής σκέψης, εφαρμογής μαθηματικών εργαλείων, αναγνώρισης προτύπων, υπομονής, αποφασιστικότητας και τύχης. Οι πιο σύγχρονες μέθοδοι προσανατολίζονται περισσότερο σε μαθηματικές έννοιες, όπως η παραγοντοποίηση ακεραίων και η εύρεση διακριτών λογαρίθμων. Ειδικότερα για συστήματα δημόσιου κλειδιού, οι θεωρητικοί της συνδυαστικής είναι οι καλύτερα εφοδιασμένοι για να επιτεθούν σ' αυτά.

Μία τυπική επίθεση κρυπτανάλυσης είναι να ξέρει κανείς κάποιο μη κρυπτογραφημένο κείμενο που ταιριάζει με ένα δοσμένο τμήμα κρυπτογραφημένου κειμένου και να προσπαθήσει να βρει το κλειδί που πρέπει να χρησιμοποιηθεί για να απεικονισθεί το ένα στο άλλο. Οι επιθέσεις αυτές ονομάζονται *επιθέσεις γνωστού*

κειμένου. Το μη κρυπτογραφημένο κείμενο μπορεί να είναι γνωστό είτε γιατί είναι τυποποιημένο (π.χ. συνήθης χαιρετισμός, ή γνωστή επικεφαλίδα) ή γιατί ήταν δυνατόν να μαντευθεί. Αν το μη κρυπτογραφημένο κείμενο έχει μαντευθεί, τότε και η ακριβής του θέση πιθανότατα δεν είναι γνωστή, αλλά το συνολικό μήνυμα είναι συνήθως αρκετά μικρό ώστε να επιχειρηθούν παράλληλες «υποεπιθέσεις» σε κάθε μία από τις οποίες το κείμενο τοποθετείται και σε διαφορετική θέση, καλύπτοντας έτσι όλες τις δυνατές περιπτώσεις. Είναι μάλιστα δυνατόν να επιλεχθεί ως μη κρυπτογραφημένο κείμενο κάτι τόσο κοινό που να είναι σχεδόν βέβαιο ότι θα βρεθεί μέσα στο μήνυμα (π.χ. το άρθρο *the* σε ένα Αγγλικό κείμενο).

Ένας ισχυρός αλγόριθμος κρυπτογράφησης θα είναι αδύνατον να «σπάσει» όχι μόνο από *επιθέσεις γνωστού κειμένου* (υποθέτοντας ότι ο εχθρός γνωρίζει όλο το μη κρυπτογραφημένο κείμενο για ένα δοθέν κρυπτογραφημένο κείμενο), αλλά και σε επιθέσεις *προσαρμοζόμενου επιλεγμένου μη κρυπτογραφημένου κειμένου*. Στον τύπο αυτό της επίθεσης ο επιτιθέμενος επιλέγει κάποιο συγκεκριμένο κείμενο και το αναλύει τόσο στη μη κρυπτογραφημένη αλλά και στην κρυπτογραφημένη μορφή. Τα ευρήματα κάθε φάσης αξιοποιούνται για την επιλογή επόμενου κειμένου για ανάλυση, με τελικό στόχο την αποκρυπτογράφηση του επιθυμητού κειμένου (που πιθανότατα έχει υποκλαπεί).

Συνοπτικά, οι τέσσερις βασικοί τύποι επιθέσεων κρυπτανάλυσης, κατά φθίνουσα σειρά δυσκολίας για τον επιτιθέμενο είναι οι εξής:

- *Μόνο βάσει κρυπτογραφημένου κειμένου.* Στην περίπτωση αυτή ο επιτιθέμενος γνωρίζει μόνο το κρυπτογραφημένο κείμενο και προσπαθεί μα βρει το μη κρυπτογραφημένο, χωρίς να έχει καμία πρότερη γνώση του τελευταίου. Η αντοχή ενός κρυπτογραφικού κώδικα σε αυτόν τον τύπο επίθεσης είναι το βασικότερο χαρακτηριστικό της ποιότητάς του.
- *Βάσει γνωστού μη κρυπτογραφημένου κειμένου.* Στην περίπτωση αυτή ο επιτιθέμενος γνωρίζει το μη κρυπτογραφημένο κείμενο καθώς και το αντίστοιχο κρυπτογραφημένο. Το συγκεκριμένο μήνυμα θεωρείται ότι έχει «σπάσει». Σε μερικά συστήματα, η γνώση ενός και μόνου ζεύγους *(μη κρυπτογραφημένο κείμενο, κρυπτογραφημένο κείμενο)* επαρκεί για να απωλεσθεί όλη η ασφάλεια του συστήματος, τόσο των προγενεστέρων όσο και των μελλοντικών κρυπτογραφήσεων. Φυσικά είναι ιδιαίτερα επιθυμητό για ένα κρυπτογραφικό σύστημα να μην μηδενίζεται η προσφερόμενη ασφάλεια από μία τέτοια διαρροή.

Βάσει των τύπων επιθέσεων που ακολουθούν, ο επιτιθέμενος έχει τη δυνατότητα να «εξαναγκάσει» τον αποστολέα στην κρυπτογράφηση ή αποκρυπτογράφηση επιλεγμένων κειμένων (μη κρυπτογραφημένων ή κρυπτογραφημένων, αντίστοιχα). Αν το κρυπτογραφικό σύστημα μπορεί να αντιμετωπίσει και τέτοιες επιθέσεις θεωρείται ότι είναι ιδιαίτερα ασφαλές.

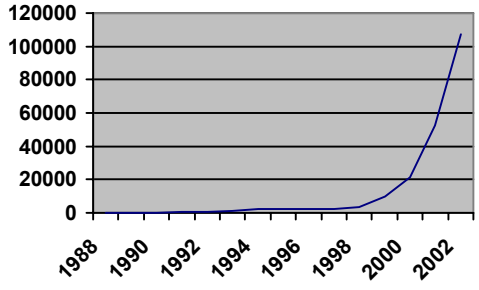
- *Επιλεγμένο μη κρυπτογραφημένο κείμενο.* Στην επίθεση αυτή ο επιτιθέμενος μπορεί να βρει το κρυπτογραφημένο κείμενο που αντιστοιχεί σε ένα μη κρυπτογραφημένο κείμενο που αυτός επιλέγει.
- *Επιλεγμένο κρυπτογραφημένο κείμενο.* Ο επιτιθέμενος μπορεί να επιλέξει όποιο κρυπτογραφημένο κείμενο επιθυμεί και να υπολογίσει το αντίστοιχο μη κρυπτογραφημένο κείμενο. Ο τύπος αυτός επίθεσης μπορεί να απαντηθεί σε συστήματα δημόσιου κλειδιού όπου μπορεί να αποκαλυφθεί το ιδιωτικό κλειδί.

- *Προσαρμοζόμενο επιλεγμένο μη κρυπτογραφημένο κείμενο.* Ο επιτιθέμενος μπορεί να προσδιορίσει το κρυπτογραφημένο κείμενο για επιλεγμένα μη κρυπτογραφημένα κείμενα σε μία επαναληπτική διαδικασία, βάσει των προηγούμενων αποτελεσμάτων.

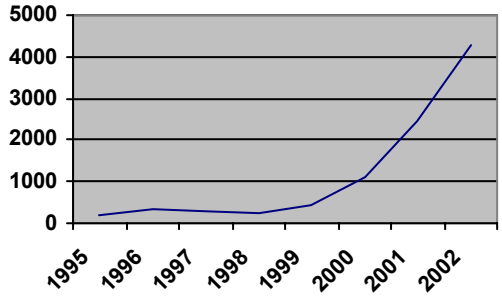
7 Ασφάλεια στο διαδίκτυο

Η εξάπλωση του διαδικτύου κατά τα τελευταία χρόνια έχει αναδείξει και πολλά προβλήματα ασφάλειας που υπάρχουν σ' αυτό για τους υπολογιστές και κατ' επέκταση για τους χρήστες τους. Το πρώτο μείζον περιστατικό που εμφανίστηκε το 1988 και έχει μείνει γνωστό με το όνομα «το σκουλήκι του Internet» είχε ως αποτέλεσμα να τεθούν εκτός λειτουργίας το 10% των υπολογιστών του τότε διαδικτύου (6.000 από τους 60.000 συνδεδεμένους τότε υπολογιστές). Το «σκουλήκι» αξιοποιούσε τρεις γνωστές αδυναμίες σε προγράμματα ηλεκτρονικού ταχυδρομείου ή παροχής πληροφοριών για να «μολύνει» έναν υπολογιστή και, αφού τον μόλυνε, τον χρησιμοποιούσε ως ορμητήριο για επιθέσεις σε άλλους υπολογιστές. Οι μολυσμένοι υπολογιστές κατέρρεαν διότι τελικά εκτελούσαν τόσο πολλά αντίγραφα του «σκουληκιού» ώστε ήταν αδύνατον να κάνουν οποιαδήποτε χρήσιμη εργασία.

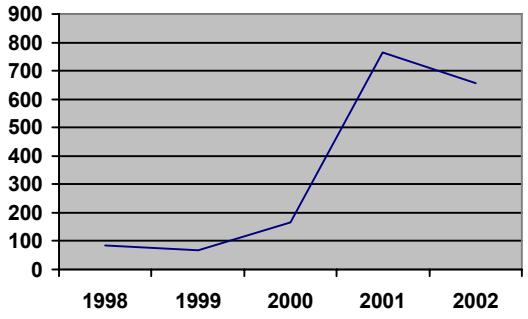
Ως αντίδραση στο συμβάν αυτό δημιουργήθηκε τότε η ομάδα CERT (Computer Emergency Response Team), με στόχο να εντοπίζει περιστατικά ασφάλειας στο διαδίκτυο και να συμβουλεύει διαχειριστές και χρήστες για το τι πρέπει για να διατηρήσουν υψηλά επίπεδα ασφάλειας. Η ομάδα αυτή κατέγραψε έξι περιστατικά το 1988, ενώ το 1995 σημειώθηκαν 2412 περιστατικά το 1995 με επιπτώσεις σε 12.000 δικτυακές περιοχές. Η αλματώδης αύξηση έχει συνεχισθεί και τα επόμενα έτη, όπως φαίνεται και στα διαγράμματα που ακολουθούν. Τα στοιχεία των διαγραμμάτων προέρχονται από την ομάδα CERT.



Περιστατικά ασφάλειας στο Internet



Ευπάθειες που αναφέρθηκαν



Επισημάνσεις ασφάλειας

Η ιλιγγιώδης αύξηση στα μεγέθη των προβλημάτων ασφάλειας οφείλεται στην αντίστοιχη αύξηση των υπολογιστών που είναι συνδεδεμένοι στο Internet, των υπηρεσιών που προσφέρονται σ' αυτό και των χρηστών που τις προσπελαύνουν. Κάθε υπολογιστής, χρήστης και υπηρεσία παρέχει θαυμάσιες ευκαιρίες στους επίδοξους εισβολείς, οι οποίοι βέβαια δεν τις αφήνουν ανεκμετάλλευτες. Οι δύο πιο συνηθισμένες δίοδοι επίθεσης σε συστήματα σήμερα είναι το ηλεκτρονικό ταχυδρομείο και οι υπηρεσίες WWW, τις οποίες θα αναλύσουμε στις επόμενες παραγράφους.

7.1 Ζητήματα ασφάλειας ηλεκτρονικού ταχυδρομείου

Το ηλεκτρονικό ταχυδρομείο ήταν αρχικά ένα ιδιαίτερα ασφαλές μέσο επικοινωνίας μεταξύ χρηστών, καθώς η λειτουργία του περιοριζόταν απλά στη διακίνηση κειμένου. Καθώς ωστόσο το απλό κείμενο κρίθηκε ιδιαίτερα περιοριστικό για τους χρήστες, σύντομα το ηλεκτρονικό ταχυδρομείο επεκτάθηκε μέσω του πρωτοκόλλου MIME, προκειμένου να καταστεί δυνατή η *επισύναψη εγγράφων* στα μηνύματα του ηλεκτρονικού ταχυδρομείου. Οι πρώτες εφαρμογές ανάγνωσης εμπλουτισμένων μηνυμάτων επέτρεπαν στους χρήστες να αποθηκεύσουν τα συνημμένα αρχεία στον δίσκο, απ' όπου και στη συνέχεια τα «άνοιγαν» για επεξεργασία μέσα από τη σχετική εφαρμογή. Για λόγους μείωσης των απαιτούμενων από πλευράς χρηστών λειτουργιών όμως, οι εφαρμογές διαχείρισης ηλεκτρονικού ταχυδρομείου ενσωμάτωσαν τη δυνατότητα άμεσης εκτέλεσης της κατάλληλης εφαρμογής για τη διαχείριση ενός συνημμένου εγγράφου, ενώ παράλληλα τα λειτουργικά συστήματα ενοποίησαν τη διαχείριση των εκτελέσιμων προγραμμάτων και των αρχείων που παραδοσιακά θεωρούνταν «αρχεία δεδομένων». Έτσι ένας χρήστης μπορεί να ζητήσει την «εκτέλεση» ενός αρχείου και, ανάλογα με το αν πρόκειται για πρόγραμμα ή «αρχείο δεδομένων» το λειτουργικό σύστημα είτε θα εκτελέσει τον κώδικα που εμπεριέχεται στο αρχείο είτε θα καλέσει την κατάλληλη εφαρμογή, αντιστοίχως. Επίσης, για λόγους «φιλικότητας» προς τον χρήστη, ορισμένα λειτουργικά συστήματα (βλέπε Windows) «κρύβουν» από τον χρήστη την κατάληξη του ονόματος του αρχείου, δηλαδή το επίθεμα exe, doc κ.λπ.

Τα τρία αυτά χαρακτηριστικά έχουν αξιοποιηθεί σε πολλές επιθέσεις μέσω ηλεκτρονικού ταχυδρομείου βάσει του ακόλουθου σχήματος:

1. ο επιτιθέμενος επισυνάπτει σε ένα μήνυμα ηλεκτρονικού ταχυδρομείου ένα πρόγραμμα και το μεταμφιέζει σε άλλο τύπο εγγράφου. Αυτό επιτυγχάνεται αποδίδοντας στο αρχείο ένα όνομα με δύο διαδοχικές καταλήξεις π.χ. *message.doc.exe*
2. Δεδομένου ότι το λειτουργικό σύστημα αποκρύπτει την τελική κατάληξη του αρχείου, ο παραλήπτης βλέπει ότι το συνημμένο ονομάζεται *message.doc*, εκλαμβάνοντάς το εσφαλμένα ως έγγραφο του Word.
3. Ο χρήστης ζητά την «εκτέλεση» του αρχείου που –αν επρόκειτο για κανονικό έγγραφο του Word– θα προκαλούσε την εκτέλεση του Word το οποίο θα παρουσίαζε τα περιεχόμενα του εγγράφου. Αντ' αυτού όμως, η «εκτέλεση» του αρχείου συνεπάγεται την εκτέλεση του προγράμματος που έχει επισυνάψει ο επιτιθέμενος. Το πρόγραμμα αυτό, τυπικά, μπορεί να εκτελέσει μία ή περισσότερες από τις κάτωθι ενέργειες:
    a. Αποστολή μηνυμάτων σε άλλους, τα οποία συνήθως περιέχουν ένα αντίγραφο του ίδιου συνημμένου

b. Αποστολή πληροφοριών στον επιτιθέμενο, όπως λίστες διευθύνσεων ηλεκτρονικού ταχυδρομείου, λίστες συνθηματικών, πληροφορίες για εγκατεστημένα προγράμματα κ.λπ.

c. Σβήσιμο αρχείων ή μόλυνσή τους.

d. Μετάδοση μόλυνσης σε άλλους υπολογιστές.

Προκειμένου να αντιμετωπισθούν τα προβλήματα που προξενεί η διακίνηση εμπλουτισμένων μηνυμάτων ηλεκτρονικού ταχυδρομείου, μπορεί να υιοθετηθεί ο έλεγχος των συνημμένων εγγράφων. Ο έλεγχος μπορεί να περιλαμβάνει την αναζήτηση προγραμμάτων που είναι γνωστό ότι χρησιμοποιούνται σε επιθέσεις τέτοιου τύπου ή/και απόπειρες απόκρυψης του πραγματικού ονόματος των συνημμένων εγγράφων. Μία πιο αυστηρή προσέγγιση συνίσταται στη συνολική απαγόρευση της επισύναψης συνημμένων εγγράφων. Οι ως άνω έλεγχοι μπορεί να διενεργούνται στους υπολογιστές των παραληπτών ή, προτιμότερα, στους εξυπηρέτες διακίνησης ηλεκτρονικού ταχυδρομείου, προκειμένου (α) να εξασφαλίζεται ότι ελέγχεται όλη η διακινούμενη ηλεκτρονική αλληλογραφία και (β) να υπάρχει ένα μόνο σημείο που θα πρέπει να ενημερώνεται όταν είναι απαραίτητο να αντιμετωπισθούν νέοι τύποι επιθέσεων. Η σημαντικότερη γραμμή άμυνας ωστόσο απέναντι στην απειλή αυτή είναι η ευαισθητοποίηση των χρηστών, έτσι ώστε να χειρίζονται προσεκτικά τα μηνύματα που παραλαμβάνουν. Τρία βασικά στοιχεία που πρέπει να προσέχουν οι χρήστες είναι τα εξής:

1. Να ρυθμίζουν τα προγράμματα διαχείρισης ηλεκτρονικής αλληλογραφίας ώστε να δείχνουν τα πλήρη ονόματα των συνημμένων αρχείων, χωρίς να αποκρύπτουν τις τυχόν καταλήξεις.
2. Αποφυγή της άμεσης «εκτέλεσης» ενός αρχείου μέσα από το πρόγραμμα διαχείρισης ηλεκτρονικής αλληλογραφίας. Η συνιστώμενη διαδικασία είναι να αποθηκεύεται το έγγραφο στο δίσκο και κατόπιν να ανοίγεται μέσα από τη σχετική εφαρμογή, έστω και αν αυτό είναι κατά τι πιο χρονοβόρο.
3. Αποφυγή του συνολικού ανοίγματος συνημμένων με άγνωστη προέλευση ή αμφιβόλου περιεχομένου.

7.2 Ζητήματα ασφάλειας κατά την πλοήγηση στο διαδίκτυο

Στα πρώτα χρόνια της εξέλιξης του διαδικτύου η πλοήγηση σ' αυτό ήταν μία εξαιρετικά ασφαλής διαδικασία, δεδομένου ότι τα προγράμματα πλοήγησης απλώς ανακτούσαν μορφοποιημένο κείμενο και εικόνες από τους κατάλληλους εξυπηρέτες και τα παρουσίαζαν στην οθόνη του υπολογιστή, παρέχοντας παράλληλα στους χρήστες τη δυνατότητα να πλοηγηθούν μεταξύ των σελίδων διατρέχοντας τους συνδέσμους. Τα σημερινά προγράμματα πλοήγησης όμως κάνουν πολύ περισσότερα από να παρουσιάζουν απλά κείμενα και εικόνες: οι τεχνολογίες της Javascript, της Java, των ActiveX, των plug-ins κ.λπ. επιτρέπουν μία πολύ πιο δυναμική διαμόρφωση των περιεχομένων που παρουσιάζονται στον χρήστη. Η διαμόρφωση αυτή όμως γίνεται κατ' ουσίαν μέσω *εκτέλεσης κώδικα*, για τον οποίο μάλιστα πολλές φορές δεν υπάρχει καμία απολύτως εγγύηση για την προέλευσή του ή για τους στόχους του. Ορισμένες από αυτές τις τεχνολογίες, όπως τα ActiveX και τα plug-ins ακολουθούν μία λογική «άσπρου-μαύρου» στο ζήτημα της ασφάλειας, υπό την έννοια ότι αν επιλέξουμε να εκτελέσουμε (ή να εγκαταστήσουμε) ένα τέτοιο κομμάτι κώδικα, ουσιαστικά επιτρέπουμε σ' αυτό να ενεργήσει κατά βούληση στον υπολογιστή μας. Στις περιπτώσεις αυτές απλώς εμπιστευόμαστε τον κατασκευαστή

του κώδικα, υποθέτοντας ότι δεν έχει κακό σκοπό. Σε άλλες τεχνολογίες, όπως η Javascript και η Java επιχειρείται μία πιο λεπτομερής διάκριση δικαιωμάτων. Στις παραγράφους που ακολουθούν θα αναλυθούν περαιτέρω τα περιβάλλοντα ασφάλειας για τις τεχνολογίες της Javascript και της Java.

7.2.1 Η ασφάλεια στη γλώσσα Javascript

Η Javascript είναι μία απλή σχετικά γλώσσα που έχει ως στόχο να προσδώσει ενεργό συμπεριφορά στα προγράμματα πλοήγησης. Προκειμένου να επιτευχθεί ο στόχος αυτός απαιτείται να δοθεί στη γλώσσα η δυνατότητα πρόσβασης σε πληροφορίες που φυλάσσονται στο πρόγραμμα πλοήγησης ή στο σύστημα αρχείων του υπολογιστή, οι οποίες δυνατότητες όμως είναι πιθανόν να χρησιμοποιηθούν για την εκτέλεση ανεπιθύμητων πράξεων όπως:

1. *Ανάγνωση ή τροποποίηση στοιχείων του προγράμματος πλοήγησης*. Το πρόγραμμα πλοήγησης είναι δυνατόν να φυλάσσει ορισμένες πληροφορίες, όπως η διεύθυνση ηλεκτρονικού ταχυδρομείου του χρήστη, *cookies* που χρησιμοποιούνται για την αναγνώριση του χρήστη από δικτυακούς τόπους, το ιστορικό των δικτυακών τόπων που έχει επισκεφθεί ο χρήστης ή ακόμη και συνθηματικά. Ένα πρόγραμμα Javascript που δρα ανεξέλεγκτα, είναι δυνατόν να υποκλέψει τα στοιχεία αυτά και να τα αποστείλει στον συγγραφέα του, ή και να τα τροποποιήσει.

2. *Αποστολή μηνυμάτων*. Σε πολλά προγράμματα πλοήγησης η διαδικασία της αποστολής ηλεκτρονικών μηνυμάτων είναι ολοκληρωμένη με τη διαδικασία πλοήγησης, για ευκολία των χρηστών. Είναι έτσι δυνατόν σε ένα πρόγραμμα Javascript να επιχειρήσει να αποστείλει μηνύματα ηλεκτρονικού ταχυδρομείου σε κάποια λίστα αποδεκτών.

3. *Ανάγνωση των στοιχείων του συστήματος*. Τα προγράμματα γλώσσας Javascript χρειάζονται πρόσβαση στο σύστημα αρχείων προκειμένου να μπορούν να χειριστούν τις φόρμες μεταφόρτωσης αρχείων (file upload). Χωρίς κατάλληλη προστασία, ένα πρόγραμμα Javascript θα μπορούσε να διαβάσει οποιοδήποτε αρχείο είναι εξουσιοδοτημένος να διαβάσει ο χρήστης που εκτελεί το πρόγραμμα πλοήγησης και να το στείλει οπουδήποτε στο διαδίκτυο μέσω μιας σχετικής φόρμας.

4. *Ανάγνωση ή τροποποίηση στοιχείων άλλων προγραμμάτων Javascript που εκτελούνται σε άλλα παράθυρα του προγράμματος πλοήγησης*. Τα σύγχρονα προγράμματα πλοήγησης επιτρέπουν στους χρήστες να ανοίγουν πολλαπλά παράθυρα και σε κάθε ένα από αυτά να προσπελαύνουν διαφορετικές σελίδες. Χωρίς επαρκείς μηχανισμούς προστασίας, είναι δυνατόν κάποιο πρόγραμμα που εκτελείται σε ένα παράθυρο να διαβάζει ή να τροποποιεί στοιχεία κάποιου άλλου προγράμματος που εκτελείται σε άλλο παράθυρο, με αποτέλεσμα τη διαρροή πληροφοριών ή/και τη δυσλειτουργία του τελευταίου.

Το μείζον ζήτημα με την ασφάλεια της γλώσσας Javascript είναι ότι δεν υπάρχει τυπικό μοντέλο βάσει του οποίου να εφαρμόζονται κάποιοι κανόνες ασφάλειας. Η παροχή ασφάλειας προς τους χρήστες βασίζεται αποκλειστικά στην έμπνευση που μπορεί να έχει ο κατασκευαστής του προγράμματος πλοήγησης και στο πόσο καλά αντιμετωπίζει όλες τις πιθανές περιπτώσεις προβλημάτων ασφάλειας που μπορεί να εμφανισθούν. Οι βασικότεροι μηχανισμοί που έχουν ενσωματωθεί στη γλώσσα Javascript για επαύξηση της ασφάλειάς της παρατίθενται στη συνέχεια.

7.2.1.1 Επίπεδα προστασίας

Στη γλώσσα Javascript ορισμένα αντικείμενα έχουν συσχετισθεί με *επίπεδα προστασίας* και αντιστοίχως κάθε εκτελούμενο πρόγραμμα έχει ένα σύνολο δικαιωμάτων. Ένα αντικείμενο μπορεί να προσπελασθεί με κάποιο συγκεκριμένο τρόπο από ένα πρόγραμμα μόνο αν το σύνολο δικαιωμάτων του προγράμματος του παρέχει τη σχετική εξουσιοδότηση. Τα δικαιώματα που παρέχονται στα προγράμματα Javascript είναι τα ακόλουθα:

- *UniversalBrowserRead, UniversalBrowserWrite.* Το πρώτο από αυτά τα δικαιώματα είναι απαραίτητο σε ένα πρόγραμμα Javascript για να διαβάσει κάποια στοιχεία που τηρεί το πρόγραμμα πλοήγησης, π.χ. την ηλεκτρονική τη διεύθυνση της σελίδας που είναι φορτωμένη σε ένα άλλο παράθυρο ή κάποιες μεταβλητές της. Το δεύτερο προνόμιο δίνει τη δυνατότητα τροποποίησης των αντίστοιχων στοιχείων. Τέλος, το προνόμιο *UniversalBrowserAccess* συνδυάζει τα δύο ανωτέρω προνόμια δίνοντας ταυτόχρονα δικαιώματα ανάγνωσης και εγγραφής.
- *UniversalFileRead.* Το προνόμιο αυτό εκχωρεί σε ένα πρόγραμμα Javascript τη δυνατότητα να χρησιμοποιήσει ένα αρχείο ως όρισμα σε μία φόρμα μεταφόρτωσης αρχείων.
- *UniversalPreferencesRead, UniversalPreferencesWrite*. Όμοια με τα προνόμια *UniversalBrowserRead, UniversalBrowserWrite*, με τη διαφορά ότι αφορούν στοιχεία ρυθμίσεων π.χ. αρχική σελίδα, ηλεκτρονική διεύθυνση κ.ά.
- *UniversalSendMail,* προνόμιο το οποίο δίνει στη δυνατότητα σε ένα πρόγραμμα να αποστέλλει μηνύματα ηλεκτρονικού ταχυδρομείου.

Σε μερικές περιπτώσεις τα προνόμια μπορούν να ανατίθενται αυτόματα, π.χ. βάσει του εξυπηρέτη από τον οποίο προέρχεται το πρόγραμμα ή από την ψηφιακή υπογραφή που τυχόν αυτό φέρει. Στις περισσότερες όμως περιπτώσεις τα προγράμματα ξεκινούν χωρίς κανένα προνόμιο και όταν χρειαστούν κάποιο τότε ο χρήστης ενημερώνεται για τη σχετική απαίτηση με κάποιο πλαίσιο διαλόγου και αποφασίζει αν επιθυμεί να επιτρέψει στο πρόγραμμα να προχωρήσει ή όχι.

Ένα πρόβλημα με τον μηχανισμό των επιπέδων προστασίας είναι ότι οι κατασκευαστές δεν προστατεύουν πάντα κατάλληλα όλα τα αντικείμενα έτσι ώστε να υπάρχει ο σχετικός έλεγχος. Για παράδειγμα, όταν στο Netscape ένα πρόγραμμα Javascript άνοιγε ένα παράθυρο μέσω της μεθόδου *window.open* το παράθυρο αυτό εθεωρείτο «ιδιοκτησία» του προγράμματος και το πρόγραμμα μπορούσε να διαβάσει όλα τα στοιχεία από το παράθυρο αυτό. Αν ωστόσο το URL του παραθύρου ετίθετο σε *about:javascript* το πρόγραμμα που άνοιξε το παράθυρο ουσιαστικά αποκτούσε πρόσβαση σε όλα τα μηνύματα από τη γλώσσα Javascript που αφορούσαν οποιοδήποτε παράθυρο του προγράμματος πλοήγησης.

7.2.1.2 Πολιτική κοινής προέλευσης

Ένα ζήτημα που πρέπει να αντιμετωπισθεί στα πλαίσια της Javascript είναι η δυνατότητα των διαφόρων προγραμμάτων που εκτελούνται σε διαφορετικά παράθυρα ή διαφορετικά πλαίσια του ίδιου παραθύρου να προσπελαύνουν το ένα τις μεταβλητές του άλλου. Προκειμένου η δυνατότητα αυτή να παρέχεται σε συνεργαζόμενα προγράμματα, αλλά να αποκλείεται για οποιοδήποτε άλλο τυχαίο ζεύγος προγραμμάτων, έχει υιοθετηθεί η *πολιτική κοινής προέλευσης*, βάσει της οποίας ένα πρόγραμμα Javascript δεν μπορεί να διαβάσει ή να γράψει μεταβλητές άλλου

προγράμματος, εκτός αν προέρχονται από τον ίδιο εξυπηρέτη, μέσω του ίδιου πρωτοκόλλου και μέσω της ίδιας θύρας. Ως παράδειγμα, αν ένα πρόγραμμα Javascript προέρχεται από το URL *http://company.com/dir/page.html*, ο κάτωθι πίνακας δείχνει τα αποτελέσματα του ελέγχου κοινής προέλευσης με άλλα προγράμματα Javascript που έχουν ληφθεί από το πρόγραμμα πλοήγησης:

URL	**Αποτέλεσμα**	**Λόγος**
`http://company.com/dir2/this.html`	✓	
`http://company.com/dir3/dir4/that.html`	✓	
`http://www.company.com/dir/pg.html`	✘	Διαφορετικοί εξυπηρέτες
`file://D|/myPage.htm`	✘	Διαφορετικό πρωτόκολλο
`http://company.com:8080/dir/etc.html`	✘	Διαφορετική θύρα

Η πολιτική ίδιας προέλευσης είναι ωστόσο υπερβολικά περιοριστική για τις ακόλουθες περιπτώσεις:

1. μία εταιρία μπορεί να έχει θυγατρικές ή συνεργαζόμενες εταιρίες με διαφορετικούς ιστοχώρους, και τα προγράμματα Javascript των ιστοχώρων αυτών θα πρέπει να μπορούν να συνεργάζονται. Για παράδειγμα, προγράμματα από τον ιστοχώρο `www.symantec.com` θα πρέπει να συνεργάζονται με προγράμματα από τον ιστοχώρο `www.sarc.com` (Symantec Antivirus Research Center).
2. Μία εταιρία μπορεί να έχει διαφορετικούς εξυπηρέτες ιστοσελίδων για λόγους εξισορρόπησης φόρτου, π.χ. `www1.ibm.com`, `www2.ibm.com`, και προγράμματα από τους εξυπηρέτες αυτούς να θα πρέπει να έχουν τη δυνατότητα να συνεργάζονται.

Προκειμένου να δοθεί η δυνατότητα συνεργασίας σε προγράμματα που πρέπει να συνεργασθούν αλλά δεν έχουν κοινή προέλευση βάσει των ανωτέρω κανόνων, έχουν υιοθετηθεί οι εξής λύσεις:

1. κάθε πρόγραμμα Javascript μπορεί να θέσει την «πηγή προέλευσής του» σε ένα URL που αποτελεί γενίκευση του URL από το οποίο έχει ανακτηθεί. Για παράδειγμα, ένα πρόγραμμα Javascript που προέρχεται από τον εξυπηρέτη `www1.ibm.com` μπορεί να θέσει την προέλευσή του σε *ibm.com*. Η ανάθεση αυτή πραγματοποιείται με την εντολή

   ```
   document.domain = "ibm.com";
   ```

   και έχει ως αποτέλεσμα να παρέχεται σε οποιοδήποτε πρόγραμμα Javascript που προέρχεται από οποιονδήποτε εξυπηρέτη του οποίου το όνομα τελειώνει σε *ibm.com* να προσπελαύνει στοιχεία του προγράμματος που εκτέλεσε την εντολή αυτή. Σημειώνεται ότι δεν ισχύει το αντίστροφο, δηλαδή το πρόγραμμα που εκτελεί αυτή την εντολή δεν αποκτά πρόσβαση σε στοιχεία άλλων προγραμμάτων που προέρχονται από εξυπηρέτες των οποίων το όνομα τελειώνει σε *ibm.com*, καθώς αυτό θα επέτρεπε σε κακόβουλα προγράμματα να ταιριάζουν την πηγή προέλευσής τους με αυτή άλλων προγραμμάτων που έχουν φορτωθεί, αποκτώντας έτσι πρόσβαση στα στοιχεία τους.
2. Τα προγράμματα Javascript μπορούν να υπογραφούν ψηφιακά, στην οποία περίπτωση προγράμματα που φέρουν την ίδια ψηφιακή υπογραφή μπορούν να

συνεργαστούν μεταξύ τους. Μια πρόσθετη δυνατότητα για τα ψηφιακά υπογεγραμμένα προγράμματα είναι να τους ανατίθενται αυτόματα επίπεδα προστασίας (βλ. εδάφιο 7.2.1.1), βάσει προεπιλογών του χρήστη.

3. Προγράμματα Javascript μπορούν να *εξάγουν* συναρτήσεις που θέλουν να καταστήσουν διαθέσιμες σε άλλα προγράμματα ανεξαρτήτως προέλευσης. Τα υπόλοιπα προγράμματα μπορούν να *εισάγουν* τις συναρτήσεις και να τις καλέσουν, αποκτώντας έτσι ελεγχόμενη πρόσβαση στα στοιχεία του προγράμματος που τις εξήγαγε. Η πρόσβαση είναι ελεγχόμενη διότι αφ' ενός προσπελαύνονται έμμεσα μόνο τα στοιχεία που αναφέρονται στη συνάρτηση, αφ' ετέρου δε διότι η συνάρτηση έχει τη δυνατότητα να πραγματοποιήσει ελέγχους σε σχέση με την προέλευση του καλούντος, τις ψηφιακές υπογραφές του κ.τ.λ.

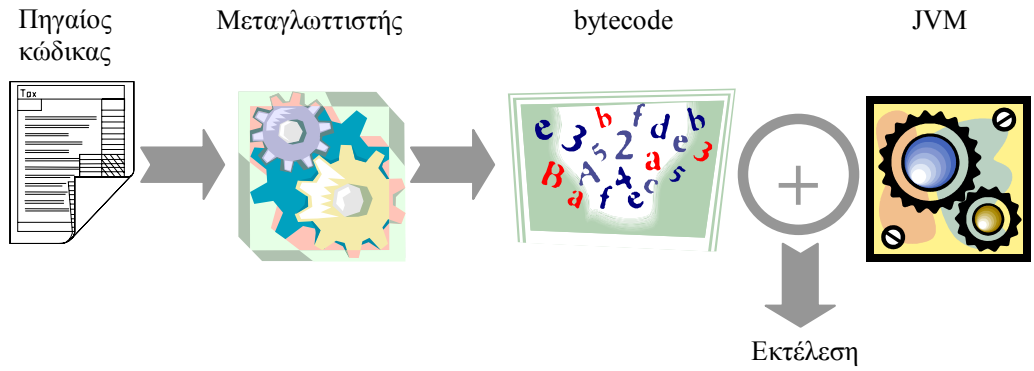
7.2.1.3 Πρόληψη διαρροής ευαίσθητων πληροφοριών

Τα προγράμματα Javascript προκειμένου να λειτουργήσουν σωστά και πλήρως πρέπει να έχουν τη δυνατότητα να προσπελάσουν στοιχεία του περιβάλλοντος (όπως π.χ. τη λίστα των εγκατεστημένων plug-ins, τον πίνακα των cookies), παρ' όλα αυτά οι πληροφορίες αυτές δεν πρέπει να αποστέλλονται στο δίκτυο με οποιονδήποτε τρόπο, για παράδειγμα αναθέτοντας τις τιμές αυτές σε ένα πεδίο φόρμας που θα υποβληθεί στη συνέχεια. Προκειμένου να παρασχεθεί προστασία από τέτοιου είδους διαρροές έχει εισαχθεί η έννοια των *σεσημασμένων πληροφοριών* (tainted information).

Ως *σεσημασμένες* χαρακτηρίζονται οι πληροφορίες των οποίων η διαρροή πρέπει να αποφευχθεί, καθώς και οποιαδήποτε πληροφορία παράγεται ως αποτέλεσμα υπολογισμού πάνω σε σεσημασμένες πληροφορίες. Για παράδειγμα, η πληροφορία `navigator.plugins` είναι εξ αρχής σεσημασμένη, και το ίδιο ισχύει και για τις πληροφορίες `navigator.plugins[0]` είναι `navigator.plugins.length + 2` διότι ο υπολογισμός τους περιλαμβάνει σεσημασμένες πληροφορίες. Όταν το πρόγραμμα πλοήγησης ανιχνεύσει προσπάθεια διαρροής σεσημασμένων πληροφοριών ειδοποιεί σχετικά τον χρήστη ο οποίος και τελικά θα αποφασίσει αν θα επιτρέψει την αποστολή τους ή όχι στο διαδίκτυο.

7.2.2 Η ασφάλεια στη γλώσσα Java

Η γλώσσα Java, σε αντίθεση με τη Javascript, είναι μία πλήρης γλώσσα προγραμματισμού, η οποία μπορεί να χρησιμοποιηθεί τόσο για συγγραφή εφαρμογών που θα εκτελούνται αυτόνομα (applications) αλλά και για τη συγγραφή εφαρμογών που θα εκτελούνται στο περιβάλλον προγραμμάτων πλοήγησης (applets). Τα προγράμματα Java συγγράφονται ως πηγαίος κώδικας και κατόπιν μεταγλωττίζονται για να παραχθεί μία ενδιάμεση αναπαράστασή τους που ονομάζεται *bytecode*. Για την εκτέλεση του bytecode απαιτείται ένα περιβάλλον εκτέλεσης Java, το οποίο περιλαμβάνει τουλάχιστον την *εικονική μηχανή Java* (Java Virtual Machine – JVM) και κάποιες βασικές κλάσεις. Η όλη διαδικασία ανάπτυξης-εκτέλεσης εφαρμογών Java αναπαρίσταται στο σχήμα που ακολουθεί.



Σε περιβάλλον διαδικτύου, η Java μπορεί να χρησιμοποιηθεί και για συγγραφή προγραμμάτων που εκτελούνται από τους εξυπηρέτες WWW για τη διαμόρφωση των σελίδων που θα αποσταλούν στους εξυπηρετούμενους. Στη συνέχεια του παρόντος εδαφίου η περίπτωση αυτή δεν θα μας απασχολήσει, καθώς η εγκατάσταση κώδικα σε κάποιον εταιρικό εξυπηρέτη υποδηλώνει και εμπιστοσύνη στον συγγραφέα του κώδικα. Η μελέτη της ασφάλειας για τη γλώσσα Java θα επικεντρωθεί κυρίως στις εφαρμογές που εκτελούνται σε περιβάλλον προγραμμάτων πλοήγησης (applets) ενώ αρκετά από τα στοιχεία που θα παρουσιασθούν αφορούν και τα προγράμματα που εκτελούνται αυτόνομα.

7.2.2.1 Τα πιθανά προβλήματα από τη χρήση της Java

Χρησιμοποιώντας τη Java σε ένα πληροφοριακό σύστημα υπάρχει μία σειρά από αρνητικά ενδεχόμενα, σε σχέση με την ασφάλεια, που είναι δυνατόν να εμφανισθούν. Κάθε τέτοιο ενδεχόμενο μπορεί να έχει ως συνέπεια τη διαρροή πληροφοριών, την ελάττωση της διαθεσιμότητας των πόρων του συστήματος (επιθέσεις τύπου άρνησης παροχής υπηρεσιών – denial of service), την απώλεια της ακεραιότητας πληροφοριών με καταστροφή ή παραφθορά τους, ή απλά την ενόχληση του χρήστη, χρησιμοποιώντας κατά μη πρέποντα τρόπο έναν ή περισσότερους από τους πόρους του συστήματος. Η αντιστοιχία μεταξύ προβλημάτων ασφάλειας και κατηγοριών πόρων συστήματος φαίνεται στον πίνακα που ακολουθεί.

Πόρος	**Διαρροή**	**Διαθεσιμότητα**	**Ακεραιότητα**	**Ενόχληση**
Σύστημα αρχείων	✓	✓	✓	✓
Δίκτυο	✓			✓
Μνήμη	✓	✓	✓	✓
Συσκευές εξόδου				✓
Συσκευές εισόδου	✓	✓		✓
Διαχείριση διεργασιών		✓		✓
Περιβάλλον χρήστη	✓		✓	✓
Κλήσεις συστήματος	✓	✓	✓	✓

Για να αντιμετωπισθούν τα αρνητικά αυτά ενδεχόμενα η Java ενσωματώνει διάφορους μηχανισμούς που ελαττώνουν τις πιθανότητες να χρησιμοποιηθεί κατά μη πρέποντα τρόπο κάποιος πόρος του συστήματος. Οι μηχανισμοί αυτοί παρέχουν υψηλά επίπεδα προστασίας απέναντι στη διαρροή πληροφοριών και την απώλεια της ακεραιότητάς τους, ενώ οι άμυνες απέναντι στην απώλεια της διαθεσιμότητας των

πόρων και στην ενόχληση του χρήστη είναι ασθενέστερες. Η ανισοκατανομή των μηχανισμών άμυνας είναι σχεδιαστική επιλογή, καθώς τα ενδεχόμενα της διαρροής και της παραφθοράς ή καταστροφής των δεδομένων είναι σαφώς δυσμενέστερα σενάρια από την απώλεια διαθεσιμότητας των πόρων ή την ενόχληση του χρήστη.

7.2.2.2 Μοντέλο ασφάλειας στη Java

Δεδομένου ότι η Java είχε σχεδιαστεί εξ αρχής για διαδικτυακή χρήση, ενσωματώνει ένα τυπικό μοντέλο ασφάλειας, το οποίο ξεκίνησε με μία αρχική μορφή, η οποία εμπλουτίστηκε στη συνέχεια με πρόσθετα στοιχεία. Το πρωταρχικό ζήτημα σε κάθε περίπτωση είναι το *αν εμπιστευόμαστε ή όχι τον κώδικα*, που ουσιαστικά ανάγεται στο αν (α) εμπιστευόμαστε τον κατασκευαστή του κώδικα και (β) αν εμπιστευόμαστε τη διαδρομή διαμέσου της οποίας ο κώδικας έφθασε στον υπολογιστή μας. Ο κώδικας που εμπιστευόμαστε εκτελείται έχοντας *πλήρη πρόσβαση στο σύστημα* (μέσω της εικονικής μηχανής Java), ενώ ο κώδικας που δεν εμπιστευόμαστε εκτελείται εντός ενός χώρου αυξημένης ασφάλειας που ονομάζεται *sandbox*. Το sandbox στη γενική περίπτωση απαγορεύει τις κάτωθι λειτουργίες:

- Ανάγνωση, διαγραφή, μετονομασία, έλεγχος ύπαρξης, αναφορά ιδιοτήτων αρχείων
- Δημιουργία ή αναφορά περιεχομένων για καταλόγους
- Σύνδεση προς διαφορετικό υπολογιστή από τον εξυπηρέτη προέλευσής του και δημιουργία θυρών προς υποδοχή συνδέσεων
- Δημιουργία παραθύρου πρώτου επιπέδου χωρίς προειδοποίηση ότι πρόκειται για ανασφαλή εφαρμογή
- Συλλογή πληροφοριών για τον χρήστη (όνομα, προσωπικός κατάλογος)
- Ορισμός ιδιοτήτων του συστήματος
- Εκτέλεση προγραμμάτων
- Τερματισμός της εκτέλεσης της εικονικής μηχανής
- Φόρτωση δυναμικών βιβλιοθηκών
- Δημιουργία και πρόσβαση νημάτων ελέγχου εκτός των δικών της
- Δημιουργία περιβάλλοντος φόρτωσης κλάσεων ή διαχείρισης ασφάλειας
- Δημιουργία διαδικασιών ελέγχου δικτύου π.χ. URLStreamHandlerFactory
- Ορισμός κλάσεων που ενσωματώνονται στις κλάσεις του υπολογιστή

Η αρκετά μακροσκελής αυτή λίστα απαγορεύσεων αφήνει ουσιαστικά τα προγράμματα που δεν εμπιστευόμαστε με τη δυνατότητα να χρησιμοποιούν μόνο την κεντρική μονάδα επεξεργασίας και τη μνήμη. Το μοντέλο αυτό είναι σαφώς ιδιαίτερα περιοριστικό, καθώς δεν επιτρέπει τη χρήση πολλών διαδεδομένων προγραμματιστικών πρακτικών, π.χ. τη δημιουργία προσωρινών αρχείων. Μια πιο λεπτομερής απόδοση αρμοδιοτήτων είναι εφικτή στη δεύτερη έκδοση της Java, όπου ψηφιακά υπογεγραμμένες εφαρμογές μπορούν να ζητήσουν από τον χρήστη να τους εκχωρήσει προνόμια που αρχικά δεν τους είναι διαθέσιμα. Σε κάθε περίπτωση ωστόσο είναι εμφανές ότι όσο αυξάνεται το σύνολο προνομίων που έχει στη διάθεσή του ένα πρόγραμμα, τόσο πιο εύκολο του είναι να προβεί σε κάποια ενέργεια που αντίκειται στους κανόνες ασφάλειας.

7.2.2.3 Δομικά στοιχεία της γλώσσας για ασφάλεια

Πριν περιγράψουμε τους μηχανισμούς διαμέσου των οποίων το Sandbox υλοποιεί την ασφάλεια, θα αναφερθούμε σε δομικά στοιχεία της γλώσσας Java τα οποία έχουν συμπεριληφθεί στον σχεδιασμό της προκειμένου να αυξάνεται η παρεχόμενη ασφάλεια. Τα δομικά αυτά στοιχεία κυρίως σχετίζονται με τη φύση της γλώσσας Java, η οποία είναι μία αντικειμενοστρεφής γλώσσα που εμπεριέχει τις έννοιες των *κλάσεων, στιγμιοτύπων, μεταβλητών και πακέτων*. Οι τρεις πρώτες έννοιες είναι οι συνήθεις που απαντώνται στις αντικειμενοστρεφείς γλώσσες προγραμματισμού, ενώ η έννοια του *πακέτου* αντιστοιχεί σε ένα σύνολο κλάσεων οι οποίες είναι τοποθετημένες εντός του ίδιου αρχείου.

Περιορισμοί στην πρόσβαση δεδομένων

Όταν ορίζεται μία κλάση είναι δυνατόν για τον προγραμματιστή να χαρακτηρίσει τις μεταβλητές ή μεθόδους της κλάσης ως *ιδιωτικές*, *προστατευμένες* ή *δημόσιες*. Ανάλογα με τον χαρακτηρισμό τους τα στοιχεία αυτά μπορούν να προσπελασθούν όπως φαίνεται στον πίνακα που ακολουθεί:

Χαρακτηρισμός μεταβλητής-μεθόδου	**Προσπελαύνεται από**
Ιδιωτική	Την ίδια την κλάση
Προστατευμένη	Την ίδια την κλάση, τις υποκλάσεις και κλάσεις στο ίδιο πακέτο
Δημόσια	Όλες τις κλάσεις
Εξ ορισμού συμπεριφορά (απουσία χαρακτηρισμού)	Την ίδια την κλάση και κλάσεις στο ίδιο πακέτο

Μέσω της χρήσης χαρακτηρισμών είναι δυνατόν στον προγραμματιστή να περιορίσει την πρόσβαση σε μεταβλητές και συναρτήσεις έτσι όπως αυτός επιθυμεί.

Χαρακτηρισμοί κλάσεων και μεθόδων ως «τελικές»

Ο αντικειμενοστρεφής προγραμματισμός δίνει στη γενική περίπτωση τη δυνατότητα δημιουργίας υποκλάσεων και επανορισμού μεθόδων. Μολονότι το χαρακτηριστικό αυτό προσθέτει ευελιξία, μπορεί να χρησιμοποιηθεί σε μία σειρά από επιθέσεις. Η Java, για να περιορίσει τους κινδύνους που προκύπτουν από τα χαρακτηριστικά αυτά δίνει τη δυνατότητα στους προγραμματιστές να χαρακτηρίσουν κλάσεις ή μεθόδους ως *τελικές*, αφαιρώντας έτσι τη δυνατότητα ορισμού υποκλάσεων ή επανορισμού, αντιστοίχως.

Άλλοι μηχανισμοί

Πέρα από τους δύο ανωτέρω μηχανισμούς για παροχή ασφάλειας, στη Java έχουν ενσωματωθεί και οι κάτωθι ασφαλιστικές δικλείδες:

- *Τα όρια των πινάκων ελέγχονται σε κάθε πρόσβαση,* εξασφαλίζοντας ότι μέσω ενός πίνακα προσπελαύνονται μόνο τα στοιχεία που ανήκουν σ' αυτόν και όχι αυθαίρετα δεδομένα.
- *Η μετατροπή τύπων είναι ιδιαίτερα περιορισμένη*. Η ορθή αντίληψη του περιβάλλοντος για τους τύπους δεδομένων είναι ιδιαίτερα σημαντική, και εξηγείται πιο αναλυτικά στη συνέχεια.

- *Οι μεταβλητές δεν μπορούν να χρησιμοποιηθούν πριν αρχικοποιηθούν*, προκειμένου να μην είναι δυνατή η επόπτευση των δεδομένων που έχουν μείνει στη στοίβα από διαδικασίες που κλήθηκαν σε προγενέστερα χρονικά σημεία.
- *Η αυτόματη συλλογή απορριμμάτων ελευθερώνει τη μνήμη που δεν χρειάζεται*. Η ύπαρξη μηχανισμού για αυτόματη συλλογή απορριμμάτων αφ' ενός προστατεύει από μία σειρά συνηθισμένων προγραμματιστικών λαθών που σχετίζονται με τη δέσμευση και απελευθέρωση μνήμης, αφ' ετέρου δε καταργεί συνολικά τους δείκτες που είναι ορατοί στον προγραμματιστή, η πρόσβαση διαμέσου των οποίων δεν είναι δυνατόν να ελεγχθεί.

7.2.2.4 Το sandbox

Όπως έχει ήδη αναφερθεί το Sandbox είναι ένας χώρος αυξημένης ασφάλειας, ο οποίος χρησιμοποιείται για να αποτρέψει τον κώδικα που δεν απολαμβάνει της εμπιστοσύνης του χρήστη να προσπελαύνει κατά αυθαίρετο τρόπο τους πόρους του συστήματος. Το Sandbox αποτελείται από τα τρία συστατικά στοιχεία:

- *τον επαληθευτή*, ο οποίος εξασφαλίζει την ορθότητα της μορφής των προγραμμάτων Java και την ορθότητα των πράξεων αναφορικά με τους τύπους δεδομένων.
- *τον φορτωτή κλάσεων*, ο οποίος φορτώνει δυναμικά κλάσεις από το περιβάλλον εκτέλεσης
- *τον διαχειριστής ασφάλειας*, ο οποίος εποπτεύει την εκτέλεση του προγράμματος και αποτρέπει ενδεχομένως επισφαλή λειτουργικότητα.

Τα συστατικά στοιχεία του sandbox παρουσιάζονται στις ακόλουθες παραγράφους.

Ο επαληθευτής

Ο επαληθευτής είναι τμήμα του περιβάλλοντος εκτέλεσης, στο οποίο προωθείται οποιοδήποτε τμήμα κώδικα Java πριν προωθηθεί για περαιτέρω επεξεργασία. Ο επαληθευτής είναι απροσπέλαστος για τα προγράμματα που έχει ως αρμοδιότητα να ελέγχει:

- *Αν η μορφή του κώδικα είναι σωστή*. Η Java έχει ένα σύνολο προδιαγραφών για τη μορφή των αρχείων, το οποίο πρέπει να τηρείται. Για παράδειγμα ένα αρχείο που περιέχει μία μεταγλωττισμένη κλάση Java πάντα ξεκινά με τα bytes CA FE BA BE (σε δεκαεξαδικό), τα ονόματα των κλάσεων που ορίζονται πρέπει να έχουν συγκεκριμένο αλφάβητο, τα πακέτα της Java κ.λπ. Οι προδιαγραφές αυτές τηρούνται *πάντα* αν το αρχείο έχει παραχθεί με έναν μεταγλωττιστή της Java, κανείς όμως δεν μπορεί να αποκλείσει το ενδεχόμενο ένα αρχείο να μην έχει μεταγλωττισθεί νομότυπα, αλλά να έχει κατασκευαστεί με κάποιο πρόγραμμα όπου γράφει κανείς απ' ευθείας σε δεκαεξαδικό ή να έχει τροποποιηθεί από τη στιγμή που παράχθηκε από τον μεταγλωττιστή της Java.
- *Αν ο κώδικας παραποιεί δείκτες, παραβιάζει περιορισμούς πρόσβασης ή χρησιμοποιεί λάθος πληροφορίες τύπων*. Μολονότι η Java δεν έχει ορατούς προς τον προγραμματιστή δείκτες, κάποια προγράμματα μπορεί να προσπαθήσουν να κατασκευάσουν δείκτες και να τους παρουσιάσουν στο σύστημα προς χρήση. Επίσης, κάποια προγράμματα μπορεί να προσπαθήσουν

να προσπελάσουν μεταβλητές άλλων κλάσεων που έχουν χαρακτηρισθεί ως *ιδιωτικές* ή να προσπαθήσουν να εκτελέσουν μη παραδεκτές μετατροπές τύπων. Όλες αυτές οι εκδοχές ελέγχονται από τον επαληθευτή.

- *Έλεγχος συνέπειας εκδόσεων κλάσεων*. Κατά την εξέλιξή της μία κλάση μπορεί να αποκτήσει νέες μεθόδους, ή να αλλάξει τύπο ή αριθμό παραμέτρων σε κάποιες υπάρχουσες ή να διαγράψει συνολικά μερικές από τις μεθόδους της. Ο επαληθευτής ελέγχει ότι οι μέθοδοι που καλούνται εξακολουθούν να υπάρχουν και ότι οι παράμετροι της κλήσης είναι συμβατά με τις παραμέτρους που καθορίζονται στον ορισμό.
- *Έλεγχος τελεστών του bytecode*. Ο επαληθευτής ελέγχει τέλος αν όλοι οι τελεστές που αναφέρονται στο bytecode είναι έγκυροι, ενώ ελέγχεται επίσης και η πιθανότητα υπερχείλισης της στοίβας ή προσπάθειας άντλησης στοιχείων από κενή στοίβα, στο βαθμό που είναι δυνατόν να ελεγχθεί. Εξακριβώνεται, τέλος, ότι οι καταχωρητές της εικονικής μηχανής Java που μνημονεύονται στις εντολές ανάγνωσης και εγγραφής του bytecode είναι υπαρκτοί.

Η διαδικασία της επαλήθευσης είναι σχετικά χρονοβόρα, και πολλές φορές διαρκεί περισσότερο από τον χρόνο που χρειάζεται για να ολοκληρωθεί η λήψη του αρχείου, αλλά το χρονικό τίμημα αιτιολογείται από την αύξηση στην παρεχόμενη ασφάλεια.

Ο φορτωτής κλάσεων

Όταν το περιβάλλον εκτέλεσης της γλώσσας Java διαπιστώσει ότι πρέπει να φορτωθεί μία κλάση (δεδομένα και μέθοδοι) από ένα αρχείο περιγραφής της, πρέπει να ελέγξει τις επιπτώσεις που η ενέργεια αυτή μπορεί να έχει στην ασφάλεια. Για παράδειγμα, αν επιχειρείται να φορτωθεί μία κλάση με το όνομα *Διαχειριστής ασφάλειας*, φόρτωση η οποία θα προκαλέσει την επικάλυψη της προκαθορισμένης κλάσης *Διαχειριστής ασφάλειας*, είναι *πιθανόν* (αλλά όχι βέβαιο) ότι έχουμε μία προσπάθεια παραβίασης της ασφάλειας. Θα έχουμε *πραγματικά* μία προσπάθεια παραβίασης της ασφάλειας αν ο κώδικας προέρχεται από ένα εντελώς άγνωστο εξυπηρέτη στο Internet· η φόρτωση αυτή πρέπει να αποτραπεί. Αντιθέτως, αν ο κώδικας προέρχεται από τον κεντρικό εξυπηρέτη της εταιρίας μας, τότε απλώς επιχειρείται η φόρτωση του εταιρικού μοντέλου ασφάλειας και πρέπει έτσι η φόρτωση να επιτραπεί. Ο φορτωτής κλάσεων είναι υπεύθυνος για την αναγνώριση και «αναχαίτιση» των πραγματικά μη επιτρεπτών ενεργειών.

Στο περιβάλλον εκτέλεσης της Java μπορούμε να αναγνωρίσουμε πέντε κατηγορίες φορτωτών κλάσεων:

- *Πρωταρχικός φορτωτής κλάσεων*. Ο πρωταρχικός φορτωτής φορτώνει τις βασικές κλάσεις του περιβάλλοντος εκτέλεσης από τον δίσκο, χωρίς μάλιστα να τις εξετάζει μέσω του επαληθευτή. Η ύπαρξη του πρωταρχικού φορτωτή είναι απολύτως απαραίτητη για την αρχικοποίηση του περιβάλλοντος της Java, καθώς η λειτουργία του επαληθευτή προϋποθέτει κάποιες κλάσεις, οι οποίες δεν θα μπορούσαν να ελεγχθούν από τον ίδιο τον επαληθευτή!
- *Γενικός φορτωτής κλάσεων*, ο οποίος υλοποιεί τη συνήθη λειτουργικότητα φόρτωσης κλάσεων με χρήση του επαληθευτή.
- *Φορτωτής κλάσεων applet,* ο οποίος έχει το χαρακτηριστικό ότι πριν φορτώσει μία κλάση από ένα applet ελέγχει αν μπορεί να τη φορτώσει ή την έχει ήδη φορτώσει ο πρωταρχικός φορτωτής. Με τον τρόπο αυτό διασφαλίζεται ότι

κανένα applet δεν μπορεί να επανορίσει κλάσεις του βασικού περιβάλλοντος εκτέλεσης.

- *Java.Security.ClassLoader,* που είναι παρόμοιος με τον φορτωτή κλάσεων applet με τη διαφορά ότι απαιτεί τα αρχεία να προέρχονται από τον εξυπηρέτη που ορίζεται στην ιδιότητα *java.app.class.path*.
- *Java.Net.URLClassLoader,* που είναι παρόμοιος με τον φορτωτή κλάσεων applet με τη διαφορά ότι απαιτεί τα αρχεία να προέρχονται από τον εξυπηρέτη που ορίζεται στην ιδιότητα *rmi.server.codebase*.

Ο φορτωτής κλάσεων υποστηρίζει επίσης την έννοια των *περιοχών ονοματολογίας*, αντιμετωπίζοντας έτσι προβλήματα που θα μπορούσαν να προκύψουν αν πάνω από μία εφαρμογές ή μία εφαρμογή και το περιβάλλον εκτέλεσης της Java χρησιμοποιούσαν το ίδιο όνομα κλάσης/μεθόδου. Για παράδειγμα, μία εφαρμογή λεξικού μπορεί να χρησιμοποιεί το όνομα μεθόδου *ΕύρεσηΜετοχής*, η οποία θα επιστρέφει τη μετοχή ενός ρήματος, αλλά το ίδιο όνομα μπορεί να χρησιμοποιεί και μία εφαρμογή χρηματιστηρίου που θα δίνει το κλείσιμο ή το ιστορικό διακύμανσης μιας μετοχής. Επίσης μία εφαρμογή που αναπτύχθηκε στην έκδοση 1.1 της Java μπορεί να χρησιμοποιεί ένα όνομα κλάσης που ήταν τότε διαθέσιμο αλλά δεσμεύτηκε από την έκδοση 1.2 της Java και μετά. Για να αντιμετωπισθούν τα προβλήματα αυτά, η κάθε εφαρμογή της Java μπορεί να ορίζει τον δικό της χώρο ονοματολογίας, ο οποίος είναι απολύτως ανεξάρτητος από των άλλων εφαρμογών, ενώ όλοι αυτοί οι χώροι είναι με τη σειρά τους ανεξάρτητοι από τον χώρο ονοματολογίας του πυρήνα της Java. Φυσικά, συνεργαζόμενες εφαρμογές μπορούν να χρησιμοποιήσουν τον ίδιο χώρο ονοματολογίας για να έχουν μία κοινή δεξαμενή ονομάτων. Όταν μία αφαρμογή αναφέρεται σε μία κλάση, ελέγχεται πρώτα αν αυτή εντάσσεται στις κλάσεις του περιβάλλοντος εκτέλεσης της Java και αν ναι, τότε χρησιμοποιείται αυτή. Διαφορετικά, αντλείται η κλάση από τον χώρο ονοματολογίας της συγκεκριμένης εφαρμογής.

Συνοψίζοντας, ο φορτωτής κλάσεων ενεργεί ως ακολούθως:

- Ελέγχει αν η κλάση έχει φορτωθεί – αν ναι, επιστρέφει τη φορτωμένη.
- Ζητά από τον πρωταρχικό φορτωτή να φορτώσει την κλάση. Αν ναι, τότε επιστρέφεται η κλάση αυτή.
- Ελέγχει αν ο φορτωτής που χρησιμοποιείται έχει το δικαίωμα δημιουργίας της κλάσης. Αν όχι, η διαδικασία τερματίζεται.
- Διαβάζεται η κλάση σε έναν πίνακα από bytes, είτε από το δίκτυο είτε από αρχεία.
- Δημιουργεί το αντικείμενο και τις μεθόδους
- Προσδιορίζονται οι κλάσεις που απαιτούνται άμεσα από την μόλις φορτωθείσα (π.χ. πρόγονοι, εκφράσεις αρχικοποίησης) και διενεργούνται αντίστοιχοι έλεγχοι γι' αυτές.
- Ο πίνακας από bytes με την αναπαράσταση της κλάσης προωθείται στον επαληθευτή για έλεγχο.

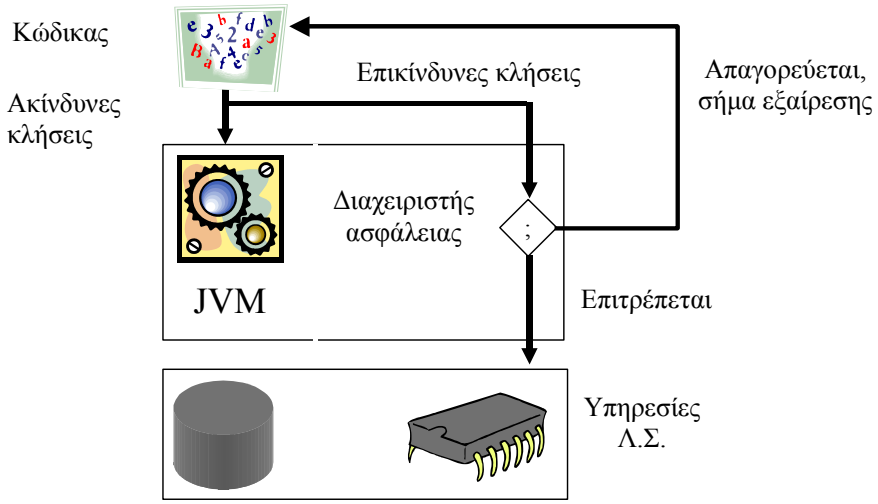
Ο διαχειριστής ασφάλειας

Όπως έχει αναφερθεί, η Java είναι μία γλώσσα που παρέχει πλήρη λειτουργικότητα και πρόσβαση στους πόρους του συστήματος, μια και έχει ως στόχο την ανάπτυξη

εφαρμογών γενικού σκοπού. Όταν όμως εφαρμογές γραμμένες σε Java εκτελούνται μέσα από το περιβάλλον προγραμμάτων πλοήγησης, δεν θα πρέπει η λειτουργικότητα αυτή (η οποία παρέχεται μέσω κλήσεων σε διαδικασίες βιβλιοθήκης) να είναι πλήρως προσβάσιμη, καθώς αυτό θα αποτελούσε σημαντικό ρήγμα στην ασφάλεια. Ο διαχειριστής ασφάλειας έχει ως στόχο να περιορίζει την πρόσβαση αυτή, ελέγχοντας ποια προγράμματα καλούν ποιες διαδικασίες βιβλιοθήκης και επιτρέποντας ή απαγορεύοντας την πραγματοποίηση των κλήσεων αυτών, ανάλογα με τις ρυθμίσεις ασφάλειας και τις επιλογές του χρήστη. Πιο αναλυτικά, η λειτουργία του διαχειριστή ασφάλειας έχει ως ακολούθως:

1. Ο διαχειριστής ασφάλειας παγιδεύει όλες τις κλήσεις προς διαδικασίες που μπορεί να είναι εν δυνάμει επικίνδυνες για την ασφάλεια του συστήματος.
2. Όταν ένα πρόγραμμα Java καλεί μία εν δυνάμει επικίνδυνη διαδικασία βιβλιοθήκης, ο διαχειριστής ασφάλειας παρεμβαίνει και ελέγχει αν η κλήση είναι επιτρεπτή ή όχι. Για τη λήψη της απόφασης εξετάζεται η πολιτική ασφάλειας του συστήματος ενώ είναι δυνατόν να ερωτηθεί και ο χρήστης.
3. Αν η κλήση δεν επιτρέπεται, ο διαχειριστής ασφάλειας παρεμποδίζει την πραγματοποίησή της και δημιουργεί ένα *σήμα εξαίρεσης*, το οποίο αποστέλλεται στην εφαρμογή. Αν η εφαρμογή είναι προετοιμασμένη για τέτοιου είδους σήματα, χειρίζεται την εμφάνισή τους ανάλογα, ειδάλλως τερματίζεται.
4. Αν η κλήση επιτρέπεται, τότε ο διαχειριστής ασφάλειας δεν παρεμποδίζει την πραγματοποίησή της.

Η όλη λειτουργία του διαχειριστή ασφάλειας απεικονίζεται στο σχήμα που ακολουθεί.



7.2.2.5 Ασφάλεια τύπων δεδομένων στη Java

Μία ιδιαίτερα σημαντική παράμετρος για την ασφάλεια του όλου περιβάλλοντος της γλώσσας Java είναι η εξασφάλιση ότι όλες οι πράξεις γίνονται σε ορθούς τύπους δεδομένων. Οι αυθαίρετες μετατροπές τύπων δεδομένων, τόσο στα πλαίσια πρόσβασης μεταβλητών, όσο και στα πλαίσια κλήσεων μεθόδων θα μπορούσαν να οδηγήσουν σε πλήρη παραβίαση των κανόνων ασφάλειας, όπως φαίνεται στα παραδείγματα που ακολουθούν.

Παράδειγμα 1

Έστω μία υποθετική κλάση ελέγχου προνομίων που έχει δηλωθεί ως ακολούθως:

```
class permissionsBox {
      private bool doFileIO;
      private bool doNetConnections;
      private bool openSafeWindows;
      bool checkPermission(what);
      bool setPerm(what, who);
      bool clearPerm(what, who);
}
```

Ας υποθέσουμε ότι το στιγμιότυπο *programPerms* αυτής της κλάσης χρησιμοποιείται για έλεγχο του αν ένα πρόγραμμα έχει δικαίωμα να προβεί σε συγκεκριμένες πράξεις. Ένα πρόγραμμα που θέλει να παρακάμψει την ασφάλεια θα μπορούσε να ενεργήσει ως εξής:

1. Ορίζει την κλάση *hackYou* με την κάτωθι δήλωση:

   ```
   class hackYou {
         public bool p1;
         public bool p2;
         public bool p3;
   }
   ```

   Η κλάση αυτή έχει το χαρακτηριστικό ότι έχει το ίδιο πλήθος και τύπο μεταβλητών στιγμιοτύπου με την κλάση *permissionsBox*, οι οποίες όμως είναι χαρακτηρισμένες ως *public*, επιτρέποντας έτσι την πρόσβασή τους από οποιοδήποτε πρόγραμμα.

2. Το πρόγραμμα πραγματοποιεί μετατροπή τύπου στο στιγμιότυπο *programPerms*, μετατρέποντάς το σε τύπο *hackYou*.

3. Το πρόγραμμα τροποποιεί τις μεταβλητές στιγμιοτύπου της μεταβλητής *programPerms* μέσα από το συνώνυμό του που είναι τύπου *hackYou*. Αυτό είναι εφικτό διότι οι μεταβλητές στιγμιοτύπου της κλάσης *hackYou* είναι χαρακτηρισμένες ως *public*.

Ο κώδικας που υλοποιεί τα βήματα 2 και 3 είναι ο εξής:

```
((hackYou)programPerms).p1 = TRUE;
```

Με τον τρόπο αυτό το πρόγραμμα «εξασφαλίζει για τον εαυτό του» τα δικαιώματα που θα του παρείχε η μεταβλητή `doFileIO` αν είχε την τιμή TRUE, ενώ το ίδιο θα μπορούσε να κάνει και με τις λοιπές δύο μεταβλητές στιγμιοτύπου, τις `doNetConnections` και `openSafeWindows`.

Παράδειγμα 2

Έστω ότι η κλάση *permissionsBox* έχει όπως στο παράδειγμα 1. Το «πονηρό» πρόγραμμα μπορεί να ορίσει την κλάση *hackYou* ως ακολούθως:

```
class hackYou {
      private bool p1;
      private bool p2;
      private bool p3;
      bool lordOfPermissions {
            p1 = p2 = p3 = TRUE;
      }
}
```

και ο κώδικας «επίθεσης» λαμβάνει τη μορφή:

```
((hackYou)programPerms).lordOfPermissions();
```

Στο παράδειγμα αυτό βλέπουμε ότι και πάλι ορίζεται μία κλάση που έχει το ίδιο πλήθος και τύπο μεταβλητών στιγμιοτύπου με την «υπό επίθεση» κλάση, αλλά ο χαρακτηρισμός τους δεν αλλάζει. Αντ' αυτού, ορίζεται μία μέθοδος η οποία θέτει όλες τις μεταβλητές στιγμιοτύπου στην τιμή TRUE, κάτι που έχει το δικαίωμα να κάνει μια και οι μεταβλητές αυτές ανήκουν στην ίδια κλάση με τη μέθοδο.

Προκειμένου να αποφευχθούν τέτοιου είδους προβλήματα, η Java ενσωματώνει εκτενείς ελέγχους που επιτρέπουν μόνο τις μετατροπές τύπων που δεν ενέχουν τέτοιους κινδύνους. Η λεπτομερής ανάλυση των επιτρεπτών μετατροπών τύπων είναι πέρα από τους σκοπούς του μαθήματος και έτσι δεν αναλύονται περαιτέρω.

7.2.2.6 Κλάσεις ασφάλειας

Τα δικαιώματα της κάθε εφαρμογής στη Java προσδιορίζονται από την *κλάση ασφάλειας* στην οποία εντάσσεται η κάθε εφαρμογή. Το μοντέλο των κλάσεων ασφάλειας, το οποίο διαφέρει ριζικά μεταξύ της πρώτης και της δεύτερης έκδοσης της γλώσσας, περιγράφεται στις επόμενες παραγράφους.

Οι κλάσεις ασφάλειας στην πρώτη έκδοση της Java

Στην πρώτη έκδοση της Java ο κώδικας διαχωριζόταν σε τρεις μεγάλες κλάσεις ασφάλειας:

1. *τοπικός κώδικας που δεν εμπιστευόμαστε.* Ο κώδικας αυτός έχει συγγραφεί τοπικά, από προγραμματιστές που εργάζονται στο εταιρικό περιβάλλον, έχει μεταγλωττισθεί και έχει έτσι παραχθεί το αντίστοιχο bytecode. Το bytecode αυτό διέρχεται από τον επαληθευτή, κατόπιν τα πακέτα και οι κλάσεις φορτώνονται από τον φορτωτή applets, και παραδίδονται προς εκτέλεση στο σχετικό τμήμα της εικονικής μηχανής της Java. Κατά τη διάρκεια της εκτέλεσης του ο κώδικας αυτός παρακολουθείται από τον διαχειριστή ασφάλειας.
2. *Εξωτερικός κώδικας που δεν εμπιστευόμαστε.* Ο κώδικας αυτός, ο οποίος προσκομίζεται μέσα από το διαδίκτυο και μπορεί να έχει παραχθεί νομότυπα από κάποιον μεταγλωττιστή της Java μπορεί όμως να μην ακολουθεί τα πρότυπα της γλώσσας. Σε κάθε περίπτωση, αρχικά αναλύεται από τον επαληθευτή, κατόπιν τα πακέτα και οι κλάσεις φορτώνονται από τον φορτωτή applets, και παραδίδονται προς εκτέλεση στο σχετικό τμήμα της εικονικής μηχανής της Java. Κατά τη διάρκεια της εκτέλεσης του ο κώδικας αυτός παρακολουθείται από τον διαχειριστή ασφάλειας.
3. *Ενσωματωμένος κώδικας και ψηφιακά υπογεγραμμένος κώδικας που εμπιστευόμαστε.* Ο κώδικας αυτός φορτώνεται από τον φορτωτή συστήματος αρχείων άμεσα, χωρίς να αναλυθεί από τον επαληθευτή και παραδίδεται προς

εκτέλεση στο σχετικό τμήμα της εικονικής μηχανής της Java. Η εκτέλεση του κώδικα δεν υπόκειται σε ελέγχους από τον διαχειριστή ασφάλειας.

Οι κλάσεις ασφάλειας στη δεύτερη έκδοση της Java

Στη δεύτερη έκδοση της γλώσσας Java το μοντέλο των κλάσεων ασφαλείας έχει εκλεπτυνθεί προκειμένου να υπάρχει καλύτερος διαχωρισμός μεταξύ των διαφορετικών προγραμμάτων και να εκχωρούνται δικαιώματα με πιο λεπτομερή ανάλυση. Κατ' αρχήν, σε κάθε bytecode αντιστοιχίζεται μία *ταυτότητα*, η οποία συντίθεται από *την προέλευση*, και τις *υπογραφές*. Η προέλευση αντιστοιχεί στον δικτυακό τόπο ή τον κατάλογο του δίσκου από τον οποίο ανακτήθηκε το bytecode, ενώ οι υπογραφές είναι ένα σύνολο ψηφιακών υπογραφών που φέρει το bytecode και καταδεικνύουν ποιος εγγυάται για την αξιοπιστία του. Όταν εκτελείται ο κώδικας Java, η ταυτότητα του κώδικα αντιπαραβάλλεται με την *πολιτική ασφάλειας* του συστήματος και, σε συνδυασμό με τους μηχανισμούς της *επόπτευσης στοίβας* και του *ελέγχου πρόσβασης* φιλτράρονται οι εν δυνάμει επικίνδυνες κλήσεις. Στις παραγράφους που ακολουθούν περιγράφονται πιο αναλυτικά οι έννοιες των προνομίων, της ταυτότητας, της επόπτευσης στοίβας και της πολιτικής ασφάλειας.

Τα προνόμια

Στη δεύτερη έκδοση της γλώσσας Java έχει εισαχθεί η έννοια των *προνομίων* τα οποία καθορίζουν τα δικαιώματα χρήσης πόρων που έχει μία εφαρμογή. Υπάρχουν έξι κατηγορίες προνομίων που είναι:

- java.io.FilePermission, που δίνει δικαιώματα πρόσβασης σε αρχεία
- java.net.SocketPermission που παρέχει πρόσβαση στο δίκτυο
- java.lang.PropertyPermission που επιτρέπει την πρόσβαση στις ιδιότητες και ρυθμίσεις του περιβάλλοντος Java
- java.lang.RuntimePermission που παρέχει πρόσβαση στους πόρους του περιβάλλοντος εκτέλεσης
- java.security.NetPermission που επιτρέπει τη χρήση διαδικασιών πιστοποίησης
- java.awt.AWTPermission, που είναι απαραίτητη για τη χρήση πόρων γραφικών, π.χ. παράθυρα

Τα προνόμια δημιουργούνται μέσω κλήσεων συγκεκριμένων *συναρτήσεων κατασκευής*, π.χ.:

- p = new FilePermission("/applets/tmp/scratch", "read");

  Το προνόμιο *p* αντιστοιχεί στο δικαίωμα ανάγνωσης στο αρχείο /applets/tmp/scratch

- p = new SocketPermission("www.di.uoa.gr:0-1023", "connect")

  Το προνόμιο *p* αντιστοιχεί στο δικαίωμα σύνδεσης στις θύρες 0-1023 του υπολογιστή www.di.uoa.gr

Για τα προνόμια έχουν ορισθεί σχέσεις *συνεπαγωγής* όπου ένα πιο γενικό προνόμιο συνεπάγεται ειδικότερα προνόμια. Για παράδειγμα, το προνόμιο `FilePermission("/tmp/*", "read")` που δίνει δικαίωμα ανάγνωσης σε όλα τα αρχεία του καταλόγου `/tmp`, συνεπάγεται το προνόμιο

`FilePermission("/tmp/scratch", "read")` που δίνει δικαίωμα ανάγνωσης στο συγκεκριμένο αρχείο `/tmp/scratch`.

Επόπτευση στοίβας

Στην πρώτη έκδοση της γλώσσας Java το μοντέλο του SandBox εκτελεί τον πολύ γενικό διαχωρισμό ανάμεσα σε κώδικα που εμπιστευόμαστε και κώδικα που δεν εμπιστευόμαστε. Η διαδρομή εκτέλεσης όμως ενός προγράμματος μπορεί να αναμειγνύει κώδικα των δύο αυτών κατηγοριών κατά αυθαίρετο τρόπο, και στην περίπτωση αυτή δεν υπάρχει μονοσήμαντη αντιστοίχηση στο αν ο «μη έμπιστος» κώδικας που καλείται από «έμπιστο» κώδικα πρέπει να έχει αυξημένα προνόμια ή όχι. Για παράδειγμα, μία μη έμπιστη εφαρμογή κανονικά δεν μπορεί να καλέσει τη διαδικασία βιβλιοθήκης `file.open` για να ανοίξει ένα τυχόν αρχείο του συστήματος. Μία μη έμπιστη εφαρμογή μπορεί ωστόσο να χρησιμοποιήσει την κλήση συστήματος `url.open` για να διαβάσει ένα URL στο Internet, και η ανάγνωση αυτή μπορεί να πραγματοποιείται προσκομίζοντας τα δεδομένα για το συγκεκριμένο URL στην κρυφή μνήμη και κατόπιν χρησιμοποιώντας τη διαδικασία `file.open` για να ανοιχθεί το συγκεκριμένο αρχείο της κρυφής μνήμης και να διαβασθούν τα περιεχόμενά του. Παρατηρούμε εδώ ότι ενώ η απ' ευθείας κλήση της `file.open` απαγορεύεται, η κλήση της μέσω της `url.open` επιτρέπεται να εκτελεσθεί.

Για να είναι εφικτός ο διαχωρισμός μεταξύ των περιπτώσεων όπου διαφορετικές διαδρομές εκτέλεσης καταλήγουν σε διαφορετικά δικαιώματα, χρησιμοποιείται ο μηχανισμός της *επόπτευσης στοίβας*. Σύμφωνα με τον μηχανισμό αυτό, οι προνομιούχες διαδικασίες σε μία διαδρομή εκτέλεσης μπορούν να παραχωρήσουν σε διαδικασίες που θα κληθούν στη συνέχεια συγκεκριμένα προνόμια, έτσι ώστε αυτές να επιτελέσουν έναν συγκεκριμένο σκοπό. Η διαχείριση των προνομίων γίνεται μέσω των ακόλουθων τεσσάρων διαδικασιών βιβλιοθήκης:

- *checkPrivilege()*, η οποία ελέγχει αν υπάρχουν τα προσδιορισμένα δικαιώματα
- *enablePrivilege()*, η οποία παραχωρεί συγκεκριμένα δικαιώματα στις διαδικασίες που θα κληθούν μετέπειτα
- *disablePrivilege()*, η οποία αφαιρεί από τους επόμενους τα καθοριζόμενα δικαιώματα
- *revertPrivilege()*, η οποία αντιστρέφει τα προσδιοριζόμενα δικαιώματα, δηλαδή τα παραχωρεί αν δεν υπάρχουν ή τα αφαιρεί, αν υπάρχουν.

Σημειώνεται εδώ ότι αν μία διαδικασία που παραχώρησε προνόμια τερματίσει (εκτελέσει μια εντολή *return* ή φτάσει στο τέλος του κώδικά της), αυτόματα αναιρούνται όλα τα προνόμια που αυτή έχει παραχωρήσει. Με τον τρόπο αυτό προφυλάσσονται οι προγραμματιστές από τυχόν αβλεψίες σε σχέση με την ανάκληση των δικαιωμάτων που θα μπορούσαν να αφήσουν ενεργό ένα προνόμιο για το υπόλοιπο της εκτέλεσης του προγράμματος.

Με χρήση των διαδικασιών διαχείρισης προνομίων, η διαδικασία βιβλιοθήκης `url.open` θα μπορούσε να κωδικοποιηθεί ως εξής:

```
url.open {
      //η url.open είναι προνομιακή διαδικασία βιβλιοθήκης
      file.open(cache_file, write);
      get_url_contents_and_store_into_file();
      // η επόμενη εντολή προετοιμάζει την κλήση της file.open
      enable_privilege(FilePermission(cache_file, "read"));
      file.open(cache_file, "read"); // μπορεί πια να κληθεί
      // ανάκληση του προνομίου
      disable_privilege(FilePermission(cache_file, "read"));
} // τυχόν άλλα προνόμια θα ανακληθούν αυτόματα
```

Ταυτότητα κώδικα και πολιτική

Όπως αναφέρθηκε προηγουμένως, στη δεύτερη έκδοση της γλώσσας Java κάθε κομμάτι κώδικα έχει μία ταυτότητα που συντίθεται από την προέλευση του κώδικα και τις ψηφιακές υπογραφές που αυτός φέρει. Η *πολιτική* από την άλλη πλευρά συνίσταται στην εκχώρηση συγκεκριμένων προνομίων σε συγκεκριμένες ταυτότητες, ορίζοντας έτσι ότι εμπιστευόμαστε κώδικες που προέρχονται από καθορισμένους εξυπηρέτες και για τα οποία υπάρχουν οι ζητούμενες έξωθεν πιστοποιήσεις να προσπελαύνουν πόρους του συστήματός μας με δεδομένο τρόπο. Η πολιτική ενός συστήματος είναι ουσιαστικά ένα σύνολο από δηλώσεις της μορφής

```
grant CodeBase "http://www.di.uoa.gr/java", SignedBy "Sun.com" {
      permission java.io.FilePermission "read,write", "/tmp/java/*";
      permission java.net.SocketPermission "connect", "*.uoa.gr";
}
```

Η πρώτη γραμμή ορίζει την προέλευση του κώδικα (`http://www.di.uoa.gr/java`) και την υπογραφή που ζητάμε (Sun.com), ενώ στην περιοχή που περικλείεται από τα άγκιστρα ορίζονται τα προνόμια που εκχωρούνται στον κώδικα με την ταυτότητα αυτή.

Στον ορισμό της πολιτικής χρειάζεται προσοχή στο ότι αν ένα κομμάτι κώδικα φέρει πάνω από μία υπογραφές τότε θα έχει *το άθροισμα των προνομίων* που αντιστοιχούν σ' αυτές, σε συνδυασμό φυσικά με τη μοναδική του προέλευση. Έτσι, αν ένας κανόνας της πολιτικής εκχωρεί σε ένα πρόγραμμα το δικαίωμα να διαβάζει αρχεία του δίσκου και ένας άλλος κανόνας της πολιτικής εκχωρεί το δικαίωμα σύνδεσης στο δίκτυο, προγράμματα για τα οποία ισχύουν και οι δύο κανόνες μπορούν να στείλουν οποιοδήποτε αρχείο μας στο δίκτυο!

7.2.2.7 Συνολική αποτίμηση της ασφάλειας στη γλώσσα Java

Η Java είχε σχεδιαστεί εξ αρχής ως γλώσσα που θα μπορούσε να χρησιμοποιηθεί στο διαδίκτυο, συνεπώς οι σχεδιαστές της είχαν κατά νου ζητήματα ασφάλειας. Οι προδιαγραφές ασφάλειας στην Java δεν είναι τέλειες, κάτι που αποδεικνύεται και από τις τροποποιήσεις που έγιναν από την πρώτη στη δεύτερη έκδοση της γλώσσας, αλλά είναι μία πολύ καλή αφετηρία. Προβλήματα επίσης ανακύπτουν στην υλοποίηση των μηχανισμών ασφάλειας, παρά τα βήματα προόδου που έχουν γίνει. Η ίδια η ομάδα της Java μιλά πλέον για «διαχείριση κινδύνων» παρά για «ασφάλεια», κάτι που θέτει το πρόβλημα στη σωστή του βάση.

Οι κατευθύνσεις που εξετάζονται για την περαιτέρω θωράκιση της γλώσσας Java από πλευράς ασφάλειας περιλαμβάνουν τα ακόλουθα:

1. *Δυνατότητα για τήρηση ημερολογίων.* Τα ημερολόγια φυσικά δεν αποτρέπουν την οποιαδήποτε παραβίαση της ασφάλειας, επιτρέπουν όμως την αποτίμηση

του μεγέθους της ζημιάς και παρέχουν ενδείξεις για το τι πρέπει να προσεχθεί στο μέλλον. Επίσης, μπορούν να αποτελέσουν είσοδο για συστήματα αυτοματοποιημένης ανίχνευσης ζητημάτων ασφάλειας ή αποδεικτικό στοιχείο στα δικαστήρια.

2. *Χρήση αφηρημένων συντακτικών δένδρων αντί του bytecode*. Ένα σύνολο ζητημάτων ανακύπτει διότι δεν είναι εφικτό να αναλυθεί στον βαθμό που θα έπρεπε ο κώδικας Java πριν την εκτέλεσή του. Η δυσχέρεια αυτή οφείλεται στο ότι η μορφή bytecode δεν είναι η πλέον κατάλληλη για τέτοιου είδους αναλύσεις. Για την αντιμετώπιση του ζητήματος αυτού έχει προταθεί η αντικατάσταση του bytecode από μία γενική μορφή που ονομάζεται *αφηρημένο συντακτικό δένδρο* που συνδυάζει την απαιτούμενη ταχύτητα εκτέλεσης με τη δυνατότητα διεξοδικής ανάλυσης.

3. *Αντίστροφη μεταγλώττιση και επαναμεταγλώττιση*. Για προστασία από τεχνητά κατασκευασμένα κακόβουλα προγράμματα, τα οποία δεν έχουν δηλαδή παραχθεί από κανονικό μεταγλωττιστή της γλώσσας Java και παραβιάζουν τους κανόνες της γλώσσας, έχει προταθεί να διενεργείται *αντίστροφη μεταγλώττιση*, δηλαδή επαναφορά του bytecode σε κάποια μορφή πηγαίου κώδικα, και κατόπιν να μεταγλωττίζεται εκ νέου ο κώδικας αυτός και να εκτελείται το αποτέλεσμα της μεταγλώττισης αυτής, εφ' όσον βέβαια είναι επιτυχής. Πέρα από το χρονικό κόστος, η προσέγγιση αυτή απαιτεί ωστόσο την ύπαρξη ενός μεταγλωττιστή Java σε κάθε υπολογιστή, κάτι που δεν είναι ιδιαίτερα ευμενώς δεκτό από τους χρήστες.

8 Ασφάλεια συστημάτων βάσεων δεδομένων

Με δεδομένο ότι πάνω από το 90% των σύγχρονων συστημάτων χρησιμοποιεί κάποιο είδος βάσης δεδομένων, η ασφάλεια των βάσεων δεδομένων αποκτά ιδιαίτερη σημασία, αν μάλιστα λάβουμε υπ' όψιν ότι η αξία της πληροφορίας είναι το κύριο «περιουσιακό στοιχείο» των πληροφοριακών συστημάτων.

Με τον όρο «συστήματα βάσεων δεδομένων» στο παρόν κεφάλαιο θα αναφερόμαστε τόσο στις ίδιες τις συλλογές δεδομένων όσο και στα συστήματα που τις διαχειρίζονται. Τα συστήματα διαχείρισης βάσεων δεδομένων είναι συνήθως λογισμικό που εκτελείται πάνω από το λειτουργικό σύστημα και υλοποιεί τις λειτουργίες που είναι απαραίτητες για τη δημιουργία, χρήση και συντήρηση της βάσης δεδομένων. Τα δεδομένα εντός μιας βάσης είναι συνήθως καλά δομημένα βάσει ενός από τα ακόλουθα μοντέλα δεδομένων:

- σχεσιακό
- αντικειμενοστραφές
- ιεραρχικό
- δικτυακό

Όταν σε ένα πληροφοριακό σύστημα εισάγεται κάποια βάση δεδομένων, οι συνήθεις διαστάσεις της ασφάλειας (ακεραιότητα, έλεγχος προσπέλασης, εμπιστευτικότητα, διαθεσιμότητα, έλεγχος -audit) επαυξάνονται με μερικές ακόμη και συγκεκριμένα:

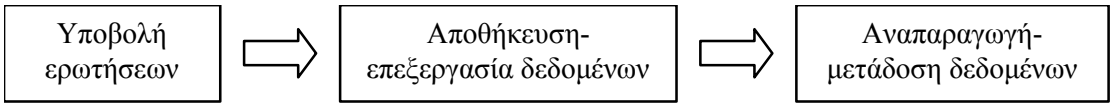
1. *διακριτότητα (granularity)*. Στα συνήθη συστήματα ένα υποκείμενο είτε έχει ένα δικαίωμα πάνω σε ένα αντικείμενο είτε δεν το έχει, ενώ το αντικείμενο καθορίζεται από τη φυσική του οντότητα. Για παράδειγμα, ένας εκτυπωτής

είναι ένα διακριτό αντικείμενο και ένας χρήστης μπορεί να έχει δικαίωμα να τυπώσει ή όχι· ένα τερματικό είναι επίσης ένα διακριτό αντικείμενο και μπορεί να παρέχει το δικαίωμα σύνδεσης ή να μην το παρέχει. Σε μία βάση δεδομένων ωστόσο, το *αντικείμενο* που αφορούν οι εξουσιοδοτήσεις μπορεί να είναι μία ολόκληρη βάση δεδομένων, μία σχέση, μία γραμμή ή στήλη σχέσης ή ακόμη και μία μεμονωμένη τιμή[3]. Η έννοια της διακριτότητας σχετίζεται με το πόσο λεπτομερής είναι η διάκριση των αντικειμένων πάνω στα οποία εφαρμόζονται οι εξουσιοδοτήσεις.

2. *Συμπερασμός ή έμμεση προσπέλαση* (inference). Υπάρχουν περιπτώσεις κατά τις οποίες κάποιος χρήστης δεν έχει δικαίωμα άμεσης προσπέλασης σε κάποια δεδομένα, μπορεί όμως να τα συνάγει με κατάλληλες εντολές προς τη βάση δεδομένων. Για παράδειγμα, ένα σύστημα βάσεων δεδομένων πιθανόν να μη μας επιτρέπει να ζητήσουμε τον μισθό ενός συγκεκριμένου εργαζόμενου βάσει αριθμού ταυτότητας, αλλά μόνο μέσω φύλου, ηλικίας και ετών προϋπηρεσίας (προκειμένου για εξαγωγή στατιστικών στοιχείων). Αν εμείς γνωρίζουμε ότι υπάρχει ένας μόνο άνδρας σε ηλικία 45 ετών με 16 χρόνια προϋπηρεσίας, τότε μπορούμε χρησιμοποιώντας τα νομότυπα κριτήρια να εξάγουμε πληροφορία που δεν θα έπρεπε.
3. *Συνάθροιση (aggregation)*. Με τον όρο συνάθροιση αναφερόμαστε στη συλλογή δεδομένων από διαφορετικές πηγές και τον συνδυασμό τους για την εξαγωγή πρόσθετων πληροφοριών. Το πρόβλημα αυτό έχει ενταθεί με την πρόοδο των τεχνικών εξόρυξης δεδομένων.
4. *Φιλτράρισμα (filtering)*, η απόκρυψη δηλαδή από τον χρήστη δεδομένων που δεν πρέπει να έχει τη δυνατότητα να δει.
5. *καταγραφή (journaling)*, η τήρηση δηλαδή πλήρους ημερολογίου σχετικά με τις ενέργειες που έχουν γίνει επί των δεδομένων και τους συσχετισμούς τους με τους χρήστες.

8.1 Γενικές αρχές ασφάλειας βάσεων δεδομένων

Η βάση δεδομένων είναι για τους χρήστες της ένα εργαλείο για να αποθηκεύουν, επεξεργάζονται και να μεταδίδουν δεδομένα, σύμφωνα με το μοντέλο που παρουσιάζεται στο ακόλουθο σχήμα:



Τόσο κατά τη φάση της επεξεργασίας όσο και κατά τη φάση της μετάδοσης χρησιμοποιούνται ειδικοί μηχανισμοί για τη διασφάλιση ότι:

1. οι *δοσοληψίες* θα εκτελεστούν στο σύνολό τους ή καθόλου
2. θα εφαρμόζονται όλοι οι *κανόνες ακεραιότητας* που έχουν ορισθεί για τη βάση δεδομένων.

Σε σχέση με την ασφάλεια βάσεων δεδομένων, θα πρέπει να λαμβάνουμε υπ' όψιν ότι η βάση δεδομένων είναι ένα σύστημα που εκτελείται σε έναν υπολογιστή, πάνω από ένα λειτουργικό σύστημα, και έτσι επηρεάζεται άμεσα από τους μηχανισμούς ασφάλειας που παρέχει ο συνδυασμός αυτός υλικού/λογισμικού. Αν για παράδειγμα

[3] Οι έννοιες αναφέρονται στο σχεσιακό μοντέλο.

το λειτουργικό σύστημα δεν παρέχει επαρκείς μηχανισμούς διακρίβωσης ταυτότητας, η βάση δεδομένων θα πρέπει να υλοποιήσει δικούς της. Επίσης, αν η βάση δεδομένων αποθηκεύεται σε αρχεία που δεν προστατεύονται επαρκώς από το λειτουργικό σύστημα, οι μηχανισμοί ελέγχου πρόσβασης που υλοποιούνται από τη βάση δεδομένων μπορούν να παρακαμφθούν, απλά διαβάζοντας ή τροποποιώντας τα αρχεία σε επίπεδο λειτουργικού συστήματος.

Θεμελιώδης απαίτηση από τα συστήματα βάσεων δεδομένων είναι η *ακεραιότητα* των δεδομένων. Τα δεδομένα πρέπει να διασώζονται σε περιπτώσεις βλαβών υλικού και δυσλειτουργιών του λογισμικού (στο μέτρο του δυνατού βεβαίως), οι τροποποιήσεις πρέπει να γίνονται μόνο από εξουσιοδοτημένους χρήστες και κάθε φορά να επιστρέφονται τα δεδομένα που έχουν αποθηκευτεί. Αν υπάρχει οποιαδήποτε παραβίαση της ακεραιότητας, οι ενδιαφερόμενοι χρήστες πρέπει τουλάχιστον να ειδοποιούνται.

Η διαθεσιμότητα είναι επίσης μία σημαντική διάσταση που ορίζει ότι τα δεδομένα πρέπει να είναι πάντα διαθέσιμα στους εξουσιοδοτημένους χρήστες.

Περιοδικά, ή σε επιλεγμένα χρονικά σημεία, πρέπει να διενεργούνται στη βάση δεδομένων *έλεγχοι ορθότητας* (audits) για εντοπισμό πιθανών προβλημάτων. Οι έλεγχοι αυτοί πρέπει να είναι κατά το δυνατόν λεπτομερείς και διεξοδικοί, ωστόσο δεν πρέπει να επηρεάζουν δυσανάλογα την απόδοση του συστήματος.

Συνοψίζοντας, θα πρέπει μία βάση δεδομένων να διαφυλάσσει την εμπιστευτικότητα των πληροφοριών επιτρέποντας την προσπέλασή τους μόνο από εξουσιοδοτημένους χρήστες, να προστατεύει την ακεραιότητα των δεδομένων, ενώ παράλληλα να μεγιστοποιεί την απόδοση του συστήματος και τη διαθεσιμότητα των δεδομένων. Οι στόχοι αυτοί είναι αλληλοσυγκρουόμενοι, έτσι συνήθως βρίσκεται κάποια χρυσή τομή, ανάλογα με τις προτεραιότητες και τις ανάγκες του οργανισμού.

Στις επόμενες παραγράφους θα αναλύσουμε εκτενέστερα τις απαιτήσεις για ασφάλεια που σχετίζονται με τις βάσεις δεδομένων.

8.2 Φυσική ακεραιότητα της βάσης δεδομένων

Η φυσική ακεραιότητα της βάσης δεδομένων συσχετίζεται με τη φθορά που μπορούν να υποστούν τα μαγνητικά μέσα αποθήκευσης από διακοπές ρεύματος, βλάβες κυκλωμάτων ή φυσιολογική φθορά. Το σύστημα θα πρέπει να παρέχει μηχανισμούς ώστε κατόπιν εμφανίσεως τέτοιων περιστατικών να είναι δυνατή η ανάκαμψη από το σφάλμα και η ανάκτηση των δεδομένων.

Ένας σημαντικός μηχανισμός υποστήριξης της φυσικής ακεραιότητας είναι η *τήρηση εφεδρικών αντιγράφων*, η αποτύπωση δηλαδή των περιεχομένων των δίσκων όπου φυλάσσεται η βάση δεδομένων σε άλλα μέσα μακροπρόθεσμης αποθήκευσης (π.χ. μαγνητικές ταινίες, DVD κ.λπ.). Για τα εφεδρικά αντίγραφα είναι σημαντικό να μπορούν να λαμβάνονται ενόσω η βάση δεδομένων βρίσκεται *εν λειτουργία*, με άλλα λόγια να είναι δυνατόν να διενεργούνται δοσοληψίες στη βάση δεδομένων ενόσω διαρκεί η λήψη του εφεδρικού αντιγράφου. Παράλληλα είναι επιθυμητό να υπάρχουν *ταχείες διαδικασίες ανάκαμψης*, δηλαδή ο χρόνος που μεσολαβεί από την έναρξη της διαδικασίας ανάκαμψης μέχρι να τεθεί το σύστημα σε πλήρη διαθεσιμότητα να είναι κατά το δυνατόν μικρότερος.

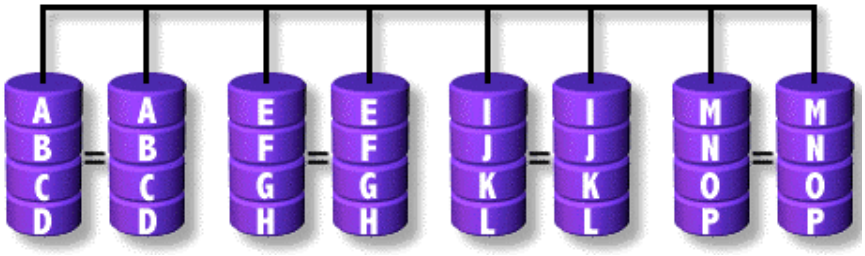
Τα εφεδρικά αντίγραφα βάσεων δεδομένων διακρίνονται –εν γένει- σε δύο κατηγορίες, τα *φυσικά αντίγραφα* και τα *λογικά αντίγραφα*. Τα *φυσικά αντίγραφα* αποτυπώνουν τα περιεχόμενα των δίσκων, όπως ακριβώς τα αποθηκεύει η βάση

δεδομένων, χωρίς να ενδιαφέρονται για τη λογική τους δομή. Τα φυσικά αντίγραφα λαμβάνονται σε μικρότερο χρόνο και αποκαθίστανται ταχύτερα, συνήθως όμως απαιτούν να διακόπτεται η λειτουργία της βάσης δεδομένων κατά τη λήψη τους και – ανάλογα με το σύστημα βάσης δεδομένων– είναι πιθανόν να μπορούν να λειτουργήσουν μόνο σε σύστημα «όμοιο» με αυτό από το οποίο ελήφθησαν. Τα *λογικά αντίγραφα* αποτυπώνουν τα δεδομένα της βάσης σε μορφή που αντικατοπτρίζει τη λογική τους δομή, χωρίς να αποτυπώνουν τον επακριβή τρόπο αποθήκευσης των δεδομένων στους δίσκους. Απαιτούν περισσότερο χρόνο για να ληφθούν και η αποκατάστασή τους διαρκεί περισσότερο, ωστόσο μπορούν εν γένει να λαμβάνονται ενόσω η βάση δεδομένων λειτουργεί και μπορούν να λειτουργήσουν και σε συστήματα «ανόμοια» προς αυτό από το οποίο ελήφθησαν.

Μία δεύτερη τεχνική για την υποστήριξη της φυσικής ακεραιότητας είναι η χρήση *τεχνολογίας RAID*. Η τεχνολογία RAID χρησιμοποιεί πλεονάζοντες δίσκους για την αποθήκευση των δεδομένων, εγγράφοντας στους πλεονάζοντες δίσκους αθροίσματα ελέγχου και διόρθωσης, κατά τρόπο ώστε βλάβες σε έναν από τους δίσκους να επιτρέπουν στο σύστημα να συνεχίσει τη λειτουργία του. Ανάλογα με τη χρήση των πλεοναζόντων δίσκων υπάρχουν οι εξής κύριες κατηγορίες RAID (περιλαμβάνονται μόνο αυτές που προσφέρουν κάποια πρόσθετη ασφάλεια, καθώς κάποιες τεχνικές RAID εστιάζονται στην απόδοση):

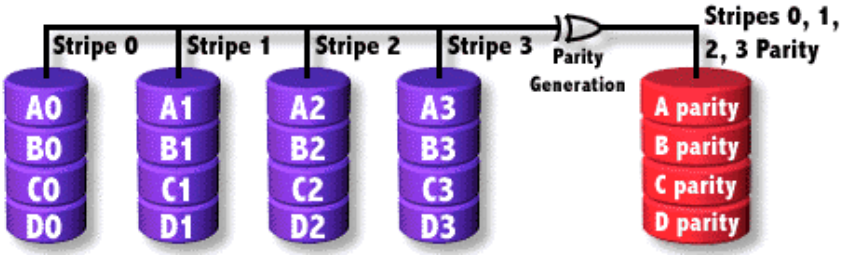
- RAID 1 – χρήση κατοπτρικών δίσκων

  Για κάθε δίσκο χρησιμοποιείται ένας ίσης χωρητικότητας. Το σύστημα εγγράφει τα δεδομένα και στους δύο δίσκους, όπως φαίνεται στο σχήμα που ακολουθεί:



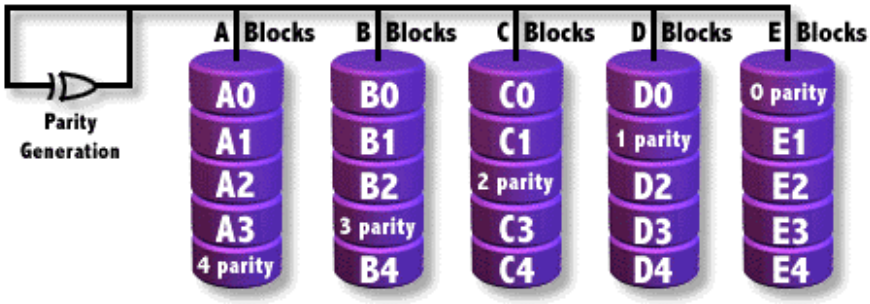
- RAID 3 – παράλληλη μεταφορά και ψηφία ισοτιμίας

  Τα δεδομένα διαμοιράζονται σε πολλαπλούς δίσκους (stripe 0-3 στην εικόνα που ακολουθεί) και ψηφία ισοτιμίας δημιουργούνται και εγγράφονται στον πλεονάζοντα δίσκο:



- RAID 5 – ανεξάρτητοι δίσκοι δεδομένων με κατανεμημένα ψηφία ισοτιμίας

  Κάθε φυσικός δίσκος περιέχει τόσο διαμερίσεις δεδομένων όσο και διαμερίσεις ψηφίων ισοτιμίας. Η ανεξαρτησία των δίσκων επιτρέπει τη μεγιστοποίηση της ταχύτητας ανάγνωσης και την αύξηση της ευελιξίας.



Τέλος, για τη διατήρηση της φυσικής ακεραιότητας των βάσεων δεδομένων συνήθως παρέχονται ασφαλείς διαδικασίες για τον τερματισμό της λειτουργίας τους, τόσο σε επίπεδο λογισμικού όσο και σε επίπεδο υλικού.

8.3 Λογική ακεραιότητα της βάσης δεδομένων

Η λογική ακεραιότητα των μιας βάσης δεδομένων αναφέρεται στην εξασφάλιση της λογικής συνοχής και συνέπειας της βάσης δεδομένων. Στη διάσταση αυτή περιλαμβάνονται οι ακόλουθες ενότητες:

1. *Πεδία ορισμού.* Πρέπει οι τιμές που αποθηκεύονται σε συγκεκριμένα πεδία να είναι σύμφωνες με το αντίστοιχο πεδίο ορισμού – π.χ. σε αριθμητικά πεδία να αποθηκεύονται μόνο αριθμητικές τιμές, σε πεδία τύπου ακεραίου να μην αποθηκεύονται δεκαδικά κ.ο.κ.

2. *Αποδοχή ή όχι τιμών null.* Αν πεδία έχουν ορισθεί ως μη επιδεχόμενα τιμές *null* (π.χ. `customerId varchar(20) not null`) πρέπει η βάση δεδομένων να εξασφαλίζει ότι για τα πεδία αυτά θα παρασχεθούν τιμές κατά την εισαγωγή των δεδομένων.

3. *Εύρος τιμών.* Ορισμένα δεδομένα πρέπει να δέχονται τιμές μόνο από ένα υποσύνολο του πεδίου ορισμού, όπως ορίζεται από εντολές ελέγχου π.χ.

   ```
   grade number(2) check(grade >= 0 and grade <= 10)
   ```

   ```
   isSecure char(1) check(isSecure = 'y' or isSecure = 'n')
   ```

4. *Εύρος τιμών.* Ορισμένα δεδομένα πρέπει να δέχονται τιμές μόνο από ένα υποσύνολο του πεδίου ορισμού, όπως ορίζεται από εντολές ελέγχου π.χ.

5. *Σχέση μεταξύ τιμών διαφορετικών πεδίων στην ίδια εγγραφή.* Είναι δυνατόν να έχουν ορισθεί περιορισμοί που πρέπει να ισχύουν μεταξύ πεδίων π.χ.

   ```
   check (endDate is NULL or endDate > startDate)
   ```

6. *Έλεγχος μοναδικότητας τιμών.* Πεδία ή συνδυασμοί πεδίων θα πρέπει να είναι μοναδικά μεταξύ οντοτήτων που αποθηκεύονται στον ίδιο πίνακα, π.χ.

   ```
   customerId number(10, 0) unique
   ```

   ```
   countryId varchar(3), passportNo varchar(10)
   primary key(countryId, passportNo)
   ```

7. *Έλεγχος αναφοράς τιμών.* Οι τιμές που αποθηκεύονται σε έναν πίνακα πρέπει να υπάρχουν σε στήλη/στήλες του άλλου πίνακα:

   ```
   countryId varchar(3) references coyntry(countryId)
   ```

   ```
   foreign key(countryId, passportNo)
           references immigrant(countryId, passportNo)
           on update cascade
   ```

8. *Έλεγχος μεταβάσεων κατάστασης*. Ελέγχονται τόσο η κατάσταση της βάσης δεδομένων *πριν* την τροποποίηση και *μετά* την τροποποίηση. Αν για παράδειγμα ένας εταιρικός κανόνας απαγορεύει την αύξηση μισθών άνω του 20%, ο μόνος τρόπος να διασφαλίσουμε την εφαρμογή του κανόνα είναι να ελέγχουμε την κατάσταση της βάσης πριν και μετά την ενημέρωση:

```
if (new.salary / old.salary > 1.2) then
       raise_application_error(100,
              'only raises up to 20% are allowed')
```

Δύο απαραίτητοι μηχανισμοί για την εξασφάλιση της ακεραιότητας των δεδομένων της βάσης είναι οι *δοσοληψίες* και ο *έλεγχος ταυτοχρονισμού*. Οι δοσοληψίες είναι ενότητες εντολών που θεωρούνται *αδιαίρετες* υπό την έννοια ότι θα εκτελεστούν είτε στο σύνολό τους (όλες οι εντολές που περιλαμβάνουν) είτε καθόλου (καμία από τις εντολές). Ως παράδειγμα, ας θεωρήσουμε τη μεταφορά ενός ποσού μεταξύ δύο τραπεζικών λογαριασμών, που υλοποιείται χρεώνοντας τον ένα λογαριασμό με το προς μεταφορά ποσό, το οποίο πιστώνεται στον άλλο. Οι πράξεις αυτές πρέπει να εκτελεστούν είτε *και οι δύο* είτε *καμία από τις δύο*, καθώς αν εκτελεστεί η μία μόνο θα έχουμε ασυνεπή δεδομένα (έλλειμμα ή περίσευμα). Για την υλοποίηση των δοσοληψιών χρησιμοποιούνται κυρίως τεχνικές τήρησης ημερολογίων, οι οποίες διακρίνονται σε δύο βασικές κατηγορίες:

- *Ημερολόγια αναίρεσης με πρότερη εγγραφή των δεδομένων*. Τα δεδομένα γράφονται άμεσα ενώ στο ημερολόγιο φυλάσσονται πληροφορίες σχετικά με το ποιες πράξεις εγγραφής πρέπει αναιρεθούν (και πως) για να καταργηθούν συνολικά ημιτελείς δοσοληψίες
- *Ημερολόγια επανάληψης με ύστερη εγγραφή των δεδομένων*. Τα γράφονται μόνο μετά το πέρας της δοσοληψίας, ενώ στο ημερολόγιο τηρούνται πληροφορίες για τις εγγραφές που πρέπει να γίνουν για να ολοκληρωθούν ημιτελείς δοσοληψίες.

Ο έλεγχος ταυτοχρονισμού φροντίζει ώστε να μην υπάρχουν αλληλοπαρεμβολές ανάμεσα σε δοσοληψίες που προσπελαύνουν ταυτόχρονα τα ίδια δεδομένα. Ως παράδειγμα θεωρήστε τις δύο ακόλουθες δοσοληψίες:

- `select sum(balance) from account;`
- `update account set balance = balance + interest, interest = 0`

η πρώτη από τις οποίες υπολογίζει το συνολικό ταμειακό υπόλοιπο όλων των λογαριασμών ενώ η δεύτερη κεφαλαιοποιεί τους τόκους. Για τις δοσοληψίες αυτές είναι σημαντικό η δοσοληψία ανάγνωσης (υπολογισμού συνολικού ταμειακού υπόλοιπου) να «διαβάσει» *όλα* τα ταμειακά υπόλοιπα είτε *πριν* την κεφαλαιοποίηση είτε *μετά* την κεφαλαιοποίηση, και σε καμία περίπτωση *μερικά πριν* και *μερικά μετά*, καθώς αυτό θα οδηγήσει σε εννοιολογικά εσφαλμένο αποτέλεσμα. Για τον έλεγχο ταυτοχρονισμού χρησιμοποιούνται τεχνικές κλειδώματος δεδομένων (όπου η κάθε δοσοληψία αποκλείει τις άλλες από συγκρουόμενες προσβάσεις στα δεδομένα που αυτή χρησιμοποίησε, μέχρι την περάτωσή της) ή τεχνικές *χρονοσήμων* (όπου η κάθε δοσοληψία *σημαδεύει χρονικά* τα δεδομένα που προσπελαύνει, ελέγχοντας παράλληλα και τα χρονόσημα που έθεσαν άλλες δοσοληψίες).

Ένα τελευταίο σημείο που σχετίζεται με τη λογική ακεραιότητα της βάσης δεδομένων είναι ο χρόνος διενέργειας του ελέγχου για το αν τηρούνται οι περιορισμοί ακεραιότητας. Οι περισσότεροι έλεγχοι μπορούν να διενεργηθούν κατά τη στιγμή της ενημέρωσης της βάσης δεδομένων – π.χ. η τιμή ενός πεδίου είτε θα είναι είτε δεν θα

είναι σε μία ορισμένη περιοχή. Μία συγκεκριμένη κατηγορία ελέγχων ωστόσο, μπορεί να διενεργηθεί *μόνον στο πέρας της δοσοληψίας*. Ας θεωρήσουμε ως παράδειγμα τον περιορισμό «μία εταιρεία πρέπει να έχει *ακριβώς έναν* πρόεδρο», που υλοποιείται με τις κάτωθι εντολές:

```
select count(*) into :numPresidents
    from employee where position = 'President';
if (:numPresidents <> 1) then /* error */ ...
```

Η αλλαγή προέδρου μπορεί να γίνει με τις εντολές:

```
1. update employee set position = 'Retired'
    where position = 'President';
2. update employee set position = 'President'
    where empId = :newPresident;
```

Παρατηρήστε ότι μετά την πρώτη εντολή ο περιορισμός δεν ισχύει (δεν υπάρχει κανείς εργαζόμενος με θέση προέδρου), οπότε αν το σύστημα διενεργήσει άμεσα τον έλεγχο θα απορρίψει την ενημέρωση. Η αντιστροφή της σειράς των εντολών δεν είναι λύση, καθώς μετά την εκτέλεση της πρώτης εντολής και πάλι ο περιορισμός δεν θα ισχύει (θα υπάρχουν δύο πρόεδροι). Η μόνη λύση είναι να διενεργείται ο έλεγχος *μόνο μετά το πέρας και των δύο εντολών* οι οποίες θα πρέπει να περιέχονται σε μία δοσοληψία.

8.4 Διακρίβωση ταυτότητας χρηστών σε συστήματα βάσεων δεδομένων

Τα συστήματα βάσεων δεδομένων πρέπει να έχουν ένα πρώτο επίπεδο ελέγχου πρόσβασης, όπου διαπιστώνεται αν ένας χρήστης έχει *συνολικά* το δικαίωμα να χρησιμοποιήσει το σύστημα βάσεων δεδομένων ή όχι. Συνήθως παρέχονται οι εξής δυνατότητες για διακρίβωση της ταυτότητας των χρηστών:

8.4.1 Διακρίβωση ταυτότητας με όνομα χρήστη-συνθηματικό

Κατά τρόπο πλήρως αντίστοιχο με τα λειτουργικά συστήματα, μία βάση δεδομένων μπορεί να ζητά από τους χρήστες της ένα όνομα χρήστη και ένα συνθηματικό ως διαπιστευτήρια της σύνδεσης. Το ΣΔΒΔ οφείλει να διατηρεί έναν κατάλογο με τις έγκυρες αντιστοιχίες ονομάτων χρηστών και συνθηματικών ώστε να αποφασίζει για το αν τα παρουσιασθέντα διαπιστευτήρια είναι έγκυρα. Δεν είναι απαραίτητο να υπάρχει οποιαδήποτε συσχέτιση ανάμεσα στα διαπιστευτήρια της βάσης δεδομένων και του λειτουργικού συστήματος.

Η τεχνική αυτή είναι χρήσιμη όταν το λειτουργικό σύστημα δεν παρέχει αξιόπιστους μηχανισμούς διακρίβωσης ταυτότητας των χρηστών ή όταν πραγματοποιούνται συνδέσεις μέσω δικτύου στη βάση δεδομένων, οπότε η ταυτότητα του χρήστη στο λειτουργικό σύστημα δεν είναι διαθέσιμη (ή αξιόπιστη).

Στο ΣΔΒΔ Oracle μπορούμε να δημιουργήσουμε έναν χρήστη του οποίου η ταυτότητα διακριβώνεται βάσει ονόματος χρήστη και συνθηματικού με την κάτωθι εντολή:

```
create user auser identified by apassword;
```

8.4.2 Διακρίβωση ταυτότητας από το λειτουργικό σύστημα

Σ' αυτή την περίπτωση το ΣΔΒΔ επαφίεται στους μηχανισμούς του λειτουργικού συστήματος να εκτελέσουν ορθή διακρίβωση ταυτότητας. Από τη στιγμή που ένας χρήστης έχει αναγνωριστεί από το λειτουργικό σύστημα *και* ο χρήστης λειτουργικού

συστήματος είναι εξουσιοδοτημένος να χρησιμοποιεί τη βάση δεδομένων, δεν ζητάται κανένα πρόσθετο στοιχείο για την προσπέλαση του χρήστη στη βάση δεδομένων. Αυτό είναι βολικό για τους χρήστες καθώς δεν είναι απαραίτητο να γνωρίζουν οποιαδήποτε άλλα συνθηματικά, πέρα από αυτά που χρησιμοποιούν για τη σύνδεσή τους στο σύστημα.

Ο μηχανισμός αυτός δεν μπορεί (ή δεν είναι σκόπιμο) να χρησιμοποιείται ως *αποκλειστικός* μηχανισμός διακρίβωσης ταυτότητας σε συστήματα όπου επιτρέπεται δικτυακή πρόσβαση στη βάση δεδομένων, καθώς επιτάσσει κάθε χρήστης να έχει λογαριασμό στο λειτουργικό σύστημα (κάτι που μπορεί να μην είναι επιθυμητό). Επίσης, πρέπει να χρησιμοποιείται *μόνον* όταν το λειτουργικό σύστημα έχει επαρκώς αξιόπιστους μηχανισμούς διακρίβωσης ταυτότητας.

Στο ΣΔΒΔ Oracle μπορούμε να δημιουργήσουμε έναν χρήστη του οποίου η ταυτότητα διακριβώνεται από το λειτουργικό σύστημα μέσω της εντολής:

```
create user auser identified externally;
```

8.4.3 Διακρίβωση ταυτότητας μέσω καθολικών υπηρεσιών καταλόγου

Ο χρήστης εισάγει στο ΣΔΒΔ ένα όνομα και ένα συνθηματικό και το ΣΔΒΔ διασυνδέεται με καθολικές υπηρεσίες καταλόγου (π.χ. X509, DCE, Kerberos, LDAP) για τη διακρίβωση της ορθότητας των διαπιστευτηρίων του χρήστη. Η προσέγγιση αυτή έχει το πλεονέκτημα ότι προωθεί τη χρήση *κεντρικού σημείου φύλαξης* των διαπιστευτηρίων σύνδεσης. Έχοντας ένα κεντρικό σημείο φύλαξης, είναι δυνατόν όλες οι ενότητες λογισμικού που απαιτούν πιστοποίηση (λειτουργικό σύστημα, βάση δεδομένων κ.λπ.) να συνδιαλέγονται με το σημείο αυτό, ούτως ώστε κάθε χρησιμοποιεί ένα μόνο ζεύγος διαπιστευτηρίων για προσπέλαση σε όλους τους πόρους.

8.5 Έλεγχος προσπέλασης

Από τη στιγμή που το σύστημα βάσης δεδομένων έχει διακριβώσει την ταυτότητα του χρήστη, αποδίδει σ' αυτόν συγκεκριμένα προνόμια για την εργασία του μέσα στη βάση δεδομένων, όπως καθορίζονται από την πολιτική ασφάλειας. Τα προνόμια χωρίζονται σε δύο κύριες κατηγορίες:

1. *προνόμια επί του συστήματος*. Τα προνόμια αυτά καθορίζουν τις γενικές δυνατότητες που έχει ο χρήστης σε σχέση με το σύστημα βάσεων δεδομένων. Τέτοια προνόμια μπορεί να καθορίζουν τη δυνατότητα δημιουργίας συνόδου (create session), τη δυνατότητα χρήσης πόρων (resource), τη δυνατότητα δημιουργίας πινάκων (create table), τη δυνατότητα δημιουργίας δεικτών (create index), τη δυνατότητα δημιουργίας διαδικασιών (create procedure) κ.λπ. Τα προνόμια επί του συστήματος μπορούν επίσης να αφορούν όρια χρήσης αποθηκευτικού χώρου όρια χρόνου εκτέλεσης, όρια εισόδου-εξόδου κ.λπ. Τα προνόμια παραχωρούνται με την εντολή *grant* αφαιρούνται δε με την εντολή *revoke*. Τα όρια χρήσης καθορίζονται με την παράμετρο quota της εντολής alter.

   Παραδείγματα:

   - `grant create session to user1;`
   - `grant create table to user1 with admin option;`
   - `revoke create table from user1;`
   - `alter user user1 quota 20M on users;`

Προσέξτε τη χρήση της πρότασης `with admin option` στη δεύτερη εντολή. Η πρόταση αυτή, όταν είναι παρούσα, καθορίζει ότι ο χρήστης που λαμβάνει το προνόμιο μπορεί πλέον *να το παραχωρεί και σε άλλους χρήστες* – καθίσταται έτσι ένας «διαχειριστής περιορισμένης εμβέλειας».

2. *προνόμια επί συγκεκριμένων αντικειμένων*. Τα προνόμια αυτά καθορίζουν τι δικαιώματα έχει ο χρήστης πάνω σε συγκεκριμένα αντικείμενα της βάσεις – πίνακες, γραμμές ή στήλες πινάκων ή συγκεκριμένες τιμές. Για τα προνόμια αυτά υπάρχουν δύο βασικές στρατηγικές ορισμού των, ο *κατ' επιλογήν έλεγχος προσπέλασης* και ο *υποχρεωτικός έλεγχος προσπέλασης*. Οι δύο αυτές κατευθύνσεις αναλύονται σε επόμενα εδάφια.

8.5.1 Κατ' επιλογήν έλεγχος προσπέλασης

Ο κατ' επιλογήν έλεγχος προσπέλασης είναι το σχήμα δικαιοδοσίας σύμφωνα με το οποίο:

1. ο δημιουργός ενός αντικειμένου είναι και ο *ιδιοκτήτης του αντικειμένου*.
2. ο ιδιοκτήτης ενός αντικειμένου έχει όλα τα δικαιώματα επί του αντικειμένου. Μεταξύ των δικαιωμάτων που έχει ο ιδιοκτήτης του αντικειμένου είναι η *παραχώρηση προνομίων* σε άλλους χρήστες και η *ανάκληση των παραχωρηθέντων δικαιωμάτων*.

Τα συγκεκριμένα δικαιώματα που εφαρμόζονται σε ένα αντικείμενο εξαρτώνται από τη *φύση του αντικειμένου*. Έτσι:

- για τους πίνακες, τα σχετικά προνόμια είναι η επιλογή, εισαγωγή, διαγραφή, ενημέρωση, τροποποίηση, δημιουργία δεικτών, δημιουργία αναφορών προς τον πίνακα
- για τις όψεις, τα σχετικά προνόμια είναι η επιλογή, εισαγωγή, διαγραφή, ενημέρωση, τροποποίηση
- για τις αποθηκευμένες διαδικασίες, το σχετικό προνόμιο είναι η εκτέλεσή τους
- για τους δείκτες, το σχετικό προνόμιο είναι η αλλαγή δομής αποθήκευσης.

8.5.1.1 Βασικός χειρισμός προνομίων με τη γλώσσα SQL

Η γλώσσα SQL υποστηρίζει άμεσα τον κατ' επιλογή έλεγχο προσπέλασης, μέσω των εντολών `grant` και `revoke`. Η εντολή grant χρησιμοποιείται για την παραχώρηση προνομίων επί συγκεκριμένων αντικειμένων σε κάποιον χρήστη, σύμφωνα με την ακόλουθη σύνταξη:

```
grant προνόμιο [, προνόμιο …]
on αντικείμενο [, αντικείμενο …]
to χρήστης [, χρήστης …]
[with grant option]
```

Κατά την παραχώρηση προνομίων εισαγωγής και ενημέρωσης επί πινάκων ή όψεων, καθώς και κατά την παραχώρηση προνομίων δημιουργίας αναφορών επί πινάκων, είναι δυνατόν να ορίζουμε *συγκεκριμένες στήλες*, για τις οποίες παραχωρείται το δικαίωμα. Στο δικαίωμα διαγραφής δεν υποστηρίζεται τέτοιος προσδιορισμός καθώς δεν έχει νόημα (η πλειάδα πάντα διαγράφεται εξ ολοκλήρου), ενώ για τον περιορισμό του δικαιώματος επιλογής θα πρέπει να χρησιμοποιηθεί ο μηχανισμός των *όψεων* που αναλύεται στη συνέχεια. Σημειώνεται ότι η παροχή προνομίου εισαγωγής που δεν περιλαμβάνει όλες τις υποχρεωτικές στήλες είναι άσκοπη.

Ως προσδιορισμός προνομίου μπορεί να χρησιμοποιηθεί η λέξη `all` που περιλαμβάνει όλα τα προνόμια. Ως προσδιορισμός χρήστη μπορεί να χρησιμοποιηθεί η λέξη `public`, η οποία υπονοεί *όλους τους χρήστες* που έχουν δικαίωμα χρήσης της βάσης δεδομένων. Τέλος, αν παρατεθεί η πρόταση `with grant option`, ο χρήστης στον οποίο παραχωρείται το προνόμιο έχει τη δυνατότητα να το παραχωρήσει με τη σειρά του σε άλλους χρήστες.

Παραδείγματα:

```
grant select, delete on table1 to user1;
grant insert on table1(col1, col2, col3) to user1;
grant update on table1(col3) to public;
grant select on table1 to user2 with grant option;
revoke select on table1 from user2;
grant references on table1(col1) to user1, user2;
```

8.5.1.2 Χρήση όψεων για παραχώρηση προνομίων

Όπως αναφέρθηκε προηγουμένως, για την παραχώρηση προνομίου επιλογής σε *συγκεκριμένες στήλες* πρέπει να δημιουργηθεί μία *όψη* (view). Η όψη αντιστοιχεί στην έννοια του *εξωτερικού σχήματος* του σχεσιακού μοντέλου και μπορεί να προσδιορίζει τόσο επιλογή στηλών (πράξη *προβολής* της σχεσιακής άλγεβρας) όσο και επιλογή γραμμών (πράξη *επιλογής* της σχεσιακής άλγεβρας) από συγκεκριμένο πίνακα[4]. Βάσει του στηριζόμενου σε όψεις σχήματος δικαιοδοσίας, για την παραχώρηση προνομίων επί συγκεκριμένων στηλών ή γραμμών ενός πίνακα, προβαίνουμε στις εξής ενέργειες:

1. δημιουργούμε την όψη που περιέχει ακριβώς τις στήλες και τις γραμμές που θέλουμε
2. παραχωρούμε τα κατάλληλα δικαιώματα επί της όψης *και όχι* επί του πίνακα βάσει του οποίου ορίζεται η όψη.

Παραδείγματα:

- ```
  create view view1 as select col1, col2, col3 from table1;
  grant select on view1 to user1;
  ```

  Με την πρώτη εντολή δημιουργείται η όψη `view1` που αποτελείται από τις στήλες `col1, col2, col3` του πίνακα `table1`. Με τη δεύτερη εντολή παραχωρείται προνόμιο επιλογής επί της όψης στον χρήστη `user1`, το οποίο ουσιαστικά τον εξουσιοδοτεί να επιλέξει *μόνο* τις στήλες `col1, col2, col3` του πίνακα `table1`.

- ```
  create view technicians as
  select empId, empName, empPhone from employee
  where empType = 'Technician';
  grant select, delete, update(empPhone) on technicians
  to techmanager;
  ```

  Η πρώτη εντολή δημιουργεί την όψη `technicians` που αποτελείται από τις στήλες `empid, empname, empphone` του πίνακα `employee`, αλλά ταυτόχρονα περιορίζει *και τις γραμμές* σ' αυτές όπου η στήλη `empType` έχει

[4] Ο ορισμός όψεων μπορεί να υποστηρίζει και πιο πολύπλοκες πράξεις, όπως συνένωση πινάκων, υπολογισμό τιμών κ.λπ. Οι δυνατότητες αυτές δεν αναλύονται, καθώς είναι πέρα από τους σκοπούς του μαθήματος.

την τιμή 'Technician'. Με τη δεύτερη εντολή παραχωρείται προνόμιο επιλογής και διαγραφής επί της όψης αυτής στον χρήστη techmanager. Έτσι ο χρήστης αυτός μπορεί να προσπελαύνει *μόνο* πλειάδες που αφορούν τεχνικούς. Για τις πλειάδες αυτές μπορεί να βλέπει την ταυτότητα, το όνομα και το τηλέφωνο του εργαζομένου, ενώ δεν έχει κανένα δικαίωμα προσπέλασης σε άλλες στήλες που τυχόν περιέχει ο βασικός πίνακας (employee – π.χ. μισθός). Ο χρήστης techmanager έχει επίσης προνόμιο ενημέρωσης του τηλεφώνου, καθώς και να διαγράψει πλειάδες από την όψη. Αυτό θα έχει ως αποτέλεσμα να διαγραφεί *ολόκληρη η πλειάδα* από τον βασικό πίνακα.

- ```
  create view mysalary as
  select empId, salary from employee
  where empid = userid();
  grant select on mysalary to public;
  ```

  Με την πρώτη εντολή δημιουργείται η όψη mysalary που αποτελείται από τις στήλες empId, salary του πίνακα employee αλλά ταυτόχρονα περιορίζει *και τις γραμμές* σ' αυτές όπου η στήλη empid είναι ίση με την ταυτότητα του τρέχοντος χρήστη (υποθέτουμε ότι το ΣΔΒΔ διαθέτει τη συνάρτηση userid που επιστρέφει ακριβώς αυτή την πληροφορία). Η δεύτερη εντολή παρέχει προνόμια ανάκτησης σε όλους τους χρήστες επί της συγκεκριμένης όψης. Το αποτέλεσμα των εντολών είναι να μπορεί κάθε χρήστης μέσω της όψης mysalary να προσπελάσει τη δική του ταυτότητα και τον δικό του μισθό μόνο.

8.5.1.3 Χρήση αποθηκευμένων διαδικασιών για παραχώρηση προνομίων

Πιο σύνθετες περιπτώσεις παραχώρησης δικαιωμάτων μπορούν να μοντελοποιηθούν με χρήση *αποθηκευμένων διαδικασιών*. Βάσει του σχήματος αυτού προκειμένου να παραχωρηθούν προνόμια για συγκεκριμένες λειτουργίες πάνω σε έναν ή περισσότερους βασικούς πίνακες ή όψεις, οι λειτουργίες αυτές κωδικοποιούνται σε μία διαδικαστική γλώσσα που υποστηρίζει το ΣΔΒΔ και φυλάσσονται εντός του ΣΔΒΔ με τη μορφή μιας *αποθηκευμένης διαδικασία.* Οι αποθηκευμένες διαδικασίες μπορούν γενικώς να δέχονται παραμέτρους και να επιστρέφουν αποτελέσματα, ενώ μπορούν να ενσωματώνουν οποιουσδήποτε ελέγχους έχει την ικανότητα να εκφράσει η σχετική γλώσσα. Κατόπιν παραχωρείται στους επιθυμητούς χρήστες *δικαίωμα εκτέλεσης* επί της συγκεκριμένης διαδικασίας, *χωρίς* να παραχωρείται σ' αυτούς οποιοδήποτε προνόμιο στα αντικείμενα που χρησιμοποιεί η διαδικασία. Η εντολή για την παραχώρηση προνομίων εκτέλεσης σε μια διαδικασία είναι

```
grant execute on procedure1 to user1, user2;
```

Ο ιδιοκτήτης (δημιουργός) της αποθηκευμένης διαδικασίας θα πρέπει να έχει τα σχετικά προνόμια επί όλων των αντικειμένων που αυτή χρησιμοποιεί.

Είναι συνήθης πρακτική να μην παρέχονται προνόμια απ' ευθείας σε πίνακες αλλά να δημιουργούνται οι σχετικές διαδικασίες και να παραχωρούνται προνόμια εκτέλεσης σ' αυτές. Οι διαδικασίες έχουν επίσης την ικανότητα να δημιουργούν *αρχεία καταγραφής ενεργειών*, εισάγοντας σε έναν πίνακα σχετικές πληροφορίες, π.χ.
```

```
create procedure makeDeposit(accountNo char(15) not null,
      amount float not null) as
begin
    update account
        set balance = balance + amount
        where accountId = accountNo;
    insert into accountLog(who, when, what, amount, info)
        values (userid(), sysdate(), 'deposit', amount, '-');
end
```

8.5.1.4 Ανάκληση των προνομίων

Η ανάκληση των προνομίων γίνεται μέσω της εντολής revoke, η σύνταξη της οποίας είναι:

```
revoke προνόμιο [, προνόμιο …]
on αντικείμενο [, αντικείμενο …]
from χρήστης [, χρήστης …]
```

Από έναν χρήστη μπορεί να αφαιρεθεί ένα *υποσύνολο των προνομίων* που του έχουν παραχωρηθεί.

Ιδιαίτερης προσοχής χρήζει η περίπτωση κατά την οποία ανακαλείται κάποιο προνόμιο έναν χρήστη X, το οποίο του είχε παραχωρηθεί με παράθεση της πρότασης `with admin option`, και στο μεσοδιάστημα ο χρήστης X είχε παραχωρήσει το προνόμιο σε άλλους χρήστες, π.χ. X1, X2, κ.λπ. Στην περίπτωση αυτή το προνόμιο *ανακαλείται αυτόματα* από όλους τους χρήστες στους οποίους έχει παραχωρηθεί από τον χρήστη X. Φυσικά, αν σε κάποιο χρήστη (π.χ. X2) το προνόμιο έχει παραχωρηθεί και απ' ευθείας από τον ιδιοκτήτη του αντικειμένου, ο χρήστης αυτός εξακολουθεί να απολαμβάνει του προνομίου. Γενικεύοντας, ένας χρήστης έχει κάποιο προνόμιο επί ενός αντικειμένου αν υπάρχει ένα *μονοπάτι εμπιστοσύνης* που να ξεκινά από τον ιδιοκτήτη του αντικειμένου και να καταλήγει σ' αυτόν. Το *μονοπάτι εμπιστοσύνης* είναι μία ακολουθία από *ακμές εμπιστοσύνης*, όπου κάθε ακμή συνδέει τον χρήστη που παραχωρεί ένα προνόμιο με τον χρήστη στον οποίο παραχωρείται το προνόμιο.

8.5.1.5 Διαχείριση προνομίων με ρόλους

Η διαχείριση των προνομίων με τους τρόπους που προαναφέρθηκαν μπορεί να είναι δυσχερής όταν το πλήθος των χρηστών είναι ιδιαίτερα μεγάλο. Επιπρόσθετα, όταν ομάδες χρηστών πρέπει να έχουν κοινά προνόμια (π.χ. οι ταμίες σε μία τράπεζα, οι υπάλληλοι του λογιστηρίου, οι υπάλληλοι του τμήματος παραγγελιών κ.λπ.) είναι πιθανόν κατά την ενημέρωση των προνομίων (π.χ. η τράπεζα παρέχει μία νέα υπηρεσία και πρέπει οι ταμίες να αποκτήσουν τα σχετικά δικαιώματα) να παραλειφθούν κάποιες ενημερώσεις ή να πραγματοποιηθούν κατά μη συνεπή τρόπο, δημιουργώντας προβλήματα στη λειτουργικότητα (ανεπαρκή προνόμια) ή την ασφάλεια (υπερβολικά προνόμια). Έτσι απαιτείται πρόσθετος κόπος προκειμένου να διατηρηθούν συγχρονισμένα τα προνόμια των χρηστών που εντάσσονται σε μία ομάδα και ελλοχεύει και κίνδυνος σφαλμάτων.

Για την απλοποίηση της διαχείρισης των προνομίων σε τέτοιες περιπτώσεις είναι δυνατόν να χρησιμοποιήσουμε την οντότητα των *ρόλων*. Οι ρόλοι είναι διαχειριστικές οντότητες στις οποίες μπορούμε να απονέμουμε και να αφαιρούμε προνόμια. Οι ρόλοι συσχετίζονται επίσης με χρήστες της βάσης δεδομένων και κάθε χρήστης αυτόματα αποκτά όλα τα προνόμια των ρόλων με τους οποίους είναι συσχετισμένος (ένας χρήστης μπορεί να είναι συσχετισμένος με πάνω από έναν ρόλους). Αν τροποποιηθούν τα δικαιώματα που έχουν απονεμηθεί σε κάποιο ρόλο, η

τροποποίηση αυτή εφαρμόζεται αυτόματα σε όλους τους χρήστες που είναι συσχετισμένοι με τον συγκεκριμένο ρόλο.

Τέλος, ένας ρόλος μπορεί να αποδίδεται σε χρήστες άνευ συνθήκης, ή να απαιτείται από τον χρήστη να γνωρίζει κάποιο συνθηματικό, *επιπροσθέτως προς τη συσχέτιση* – με άλλα λόγια πρέπει *και* να έχει συσχετισθεί ο χρήστης με τον ρόλο *και* να γνωρίζει το συνθηματικό για να απολάβει τα προνόμια του ρόλου. Ρόλοι που προστατεύονται με συνθηματικά πρέπει να ενεργοποιούνται ρητώς από τον χρήστη, έτσι ώστε να υπάρχει η δυνατότητα εισαγωγής του συνθηματικού.

Παραδείγματα:

```
1. create role personnel not identified;
2. create role accountant identified by secret;
3. grant all on tbl1, tbl2, tbl3 to personnel;
4. grant all on tbl1, tbl5, tbl6 to accountant;
5. grant personnel to user1, user2;
6. grant accountant to user1, user3, user5;
7. revoke update, delete on tbl1 from accountant;
8. set role accountant identified by secret;
9. revoke accountant from user3;
```

Η πρώτη εντολή δημιουργεί έναν ρόλο με το όνομα `personnel` ο οποίος δεν προστατεύεται από συνθηματικό, ενώ η δεύτερη εντολή δημιουργεί τον ρόλο `accountant` ο οποίος προστατεύεται από το συνθηματικό `secret`. Η τρίτη εντολή αποδίδει στον ρόλο `personnel` όλα τα προνόμια πάνω στους πίνακες `tbl1, tbl2, tbl3` και η τέταρτη εντολή αποδίδει στον ρόλο `accountant` όλα τα προνόμια πάνω στους πίνακες `tbl1, tbl5, tbl6`. Οι εντολές 5 και 6 συσχετίζουν τους ρόλους με συγκεκριμένους χρήστες, αποδίδοντάς τους όλα τα προνόμια που έχουν απονεμηθεί στους ρόλους. Η εντολή 7 αφαιρεί κάποια δικαιώματα από τον ρόλο `accountant`, επηρεάζοντας έτσι τα δικαιώματα των χρηστών `user1, user3, user5` οι οποίοι έχουν συσχετισθεί με τον ρόλο. Σημειώστε ότι ο χρήστης `user1` εξακολουθεί να έχει όλα τα προνόμια επί του πίνακα `tbl1`, δυνάμει της συσχέτισής του με τον ρόλο `personnel`, στον οποίο έχουν απονεμηθεί τα προνόμια αυτά. Η εντολή 8 μπορεί να χρησιμοποιηθεί από τους χρήστες `user1, user3, user5` για να ενεργοποιηθεί ο ρόλος accountant με τον οποίο έχουν συσχετισθεί – ουσιαστικά προκειμένου να ισχύσουν γι' αυτούς τα προνόμια που έχουν αποδοθεί στον ρόλο. Για τον ρόλο `personnel` δεν απαιτείται αντίστοιχη εντολή, καθώς ο ρόλος αυτός δεν προστατεύεται από συνθηματικό. Τέλος, η εντολή 9 καταργεί τη συσχέτιση του χρήστη `user3` με τον ρόλο `accountant`, συνεπώς ο χρήστης παύει να απολαμβάνει των προνομίων του ρόλου.

8.5.2 Ευαίσθητα δεδομένα

Πριν προχωρήσουμε στην παρουσίαση των τεχνικών υποχρεωτικού ελέγχου προσπέλασης για βάσεις δεδομένων, θα αναπτύξουμε έννοιες σχετικές με την *ευαισθησία των δεδομένων*, οι οποίες χρησιμοποιούνται στον υποχρεωτικό έλεγχο πρόσβασης. Ως *ευαίσθητα* χαρακτηρίζονται τα δεδομένα που δεν πρέπει να αποκαλυφθούν. Η ύπαρξη τέτοιων δεδομένων και ο βαθμός ευαισθησίας του καθενός εξαρτάται από τη βάση δεδομένων και τη σημασιολογία των δεδομένων. Για παράδειγμα, π.χ. ο Χρυσός οδηγός δεν περιέχει ευαίσθητες πληροφορίες, ενώ το αρχείο καινοτόμων προϊόντων μιας εταιρίας είναι εξ ολοκλήρου απόρρητο. Οι ακραίες αυτές καταστάσεις είναι οι πιο εύκολες στον χειρισμό τους, καθώς μπορούν να αντιμετωπισθούν με συνολική παροχή ή άρνηση δικαιώματος πρόσβασης στη

βάση δεδομένων. Οι ενδιάμεσες καταστάσεις ωστόσο είναι πιο περίπλοκες, καθώς θα πρέπει κάποια δεδομένα να είναι προσβάσιμα και κάποια όχι. Για παράδειγμα, ας θεωρήσουμε μία βάση δεδομένων φοιτητών:

- το όνομα και η διεύθυνση δεν είναι ευαίσθητα
- η οικονομική βοήθεια και το ιατρικό ιστορικό είναι απόρρητα
- το φύλλο, η ηλικία και η φυλή είναι στο ενδιάμεσο
- όλοι θα έχουν πρόσβαση σε όνομα και διεύθυνση, κάποιοι στο φύλλο, ηλικία και φυλή, ελάχιστοι στην οικονομική βοήθεια & το ιατρικό ιστορικό
- η πολιτική ασφάλειας μπορεί να ορίζουν ότι κανείς δεν πρέπει να έχει πρόσβαση σε όλα τα στοιχεία

Για τον χαρακτηρισμό ενός δεδομένου ως ευαίσθητο ή όχι λαμβάνονται υπόψη πολλοί παράγοντες. Τα δεδομένα μπορεί να είναι ευαίσθητα για έναν από τους κάτωθι λόγους:

- *εκ της φύσεως τους*. Η τοποθεσία των οπλικών συστημάτων δεν πρέπει να αποκαλυφθεί λόγω στρατιωτικού απορρήτου. Το οικονομικό απόρρητο επίσης επιτάσσει να μην αποκαλυφθεί το μέσο εισόδημα των πεταλωτών αλόγων σε μία πόλη με μόνο έναν πεταλωτή, καθώς αποκαλύπτει το εισόδημα ενός συγκεκριμένου ατόμου.
- *λόγω της πηγής τους*. Η αποκάλυψη συγκεκριμένων δεδομένων μπορεί να καταδείξει την πηγή της πληροφορίας, η οποία πρέπει να παραμείνει μυστική. Ίγια παράδειγμα, αν μία πληροφορία είναι γνωστή σε δύο μόνο άτομα και αποκαλυφθεί ότι τη γνωρίζει και κάποιος τρίτος, γίνεται άμεσα αντιληπτό ποιος διέρρευσε την πληροφορία.
- *λόγω ρητού χαρακτηρισμού* π.χ. διαβαθμισμένα στρατιωτικά μυστικά
- *ως τμήματα ευαίσθητων δεδομένων ή ευαίσθητων εγγραφών*. Για παράδειγμα τα επιδόματα ενός εργαζόμενου είναι ευαίσθητα (ως τμήμα της μισθοδοσίας που είναι ευαίσθητο δεδομένο). Μια εγγραφή που υποδεικνύει ότι συγκεκριμένο άτομο συμμετέχει σε μία απόρρητη αποστολή είναι ευαίσθητη ως τμήμα της περιγραφής της απόρρητης αποστολής.
- *λόγω πληροφοριών που αποκαλύφθηκαν προηγουμένως*. Αν αποκαλυφθεί το γεωγραφικό πλάτος ενός ορυχείου χρυσού δεν πρέπει να αποκαλυφθεί παράλληλα και το γεωγραφικό μήκος.

8.5.2.1 Είδη αποκαλύψεων ευαίσθητων δεδομένων

Για τα ευαίσθητα δεδομένα πρέπει να λαμβάνεται ειδική μέριμνα καθώς υπάρχει ένα σύνολο αποκαλύψεων σχετικά με αυτά που πρέπει να αποφευχθεί:

- *αποκάλυψη επακριβών τιμών*: η πιο σοβαρή περίπτωση όπου τα ευαίσθητα δεδομένα αποκαλύπτονται. Σημειώνεται ότι είναι δυνατόν οι χρήστες που ζητούν τα δεδομένα να μην γνωρίζουν ότι πρόκειται για ευαίσθητα δεδομένα και η αποκάλυψη να οφείλεται σε ανεπαρκή σχεδιασμό της πολιτικής ασφάλειας ή κακή υλοποίησή της.
- *Αποκάλυψη ορίων*. Η επακριβής τιμή ενός ευαίσθητου δεδομένου `x` δεν γίνεται γνωστή, αποκαλύπτεται όμως ότι για την τιμή του `x` ισχύει

`min ≤ x ≤ max`. Για παράδειγμα μπορεί ένας χρήστης να μάθει ότι ο μισθός κάποιου υπαλλήλου βρίσκεται μεταξύ 2000 και 3000 ευρώ, ενώ ο ίδιος έχει μικρότερες μηνιαίες απολαβές. Η ευπάθεια στην αποκάλυψη ορίων είναι ιδιαίτερα επικίνδυνη αν μπορεί να αξιοποιηθεί αναδρομικά αποκαλύπτοντας ότι `min ≤ x ≤ (min + max) / 2` κ.ο.κ., μια και αυτό θα οδηγήσει σε αποκάλυψη επακριβών τιμών. Επίσης και η αποκάλυψη μεμονωμένων άνω ή κάτω ορίων είναι εν δυνάμει επικίνδυνη, π.χ. αν αποκαλυφθεί ότι ότι το πλήθος γιατρών με ειδίκευση στον βιολογικό πόλεμο είναι πάνω από ένα όριο.

- *Αρνητικός συμπερασμός*, η άντληση δηλαδή της πληροφορίας ότι ένα ευαίσθητο δεδομένο *δεν έχει* κάποια συγκεκριμένη τιμή. Για παράδειγμα, το να αποκαλυφθεί ότι κάποιο πρόσωπο *δεν έχει μηδέν καταδίκες για κακουργήματα* είναι σημαντικό (παρατηρήστε ότι η διαφορά μεταξύ 99 και 100 καταδικών είναι αδιάφορη, ενώ η διαφορά από 0 σε 1 είναι σημαντική). Οι συνέπειες της αποκάλυψης μπορεί να κυμαίνονται ανάλογα με τη φύση των δεδομένων και το πλήθος των εναλλακτικών τιμών – π.χ. η αποκάλυψη ότι κάποιος φοιτητής δεν αποφοίτησε με άριστα είναι λιγότερο σημαντική καθώς το δυνατό εύρος τιμών 5.00 – 8.49 είναι ιδιαίτερα μεγάλο

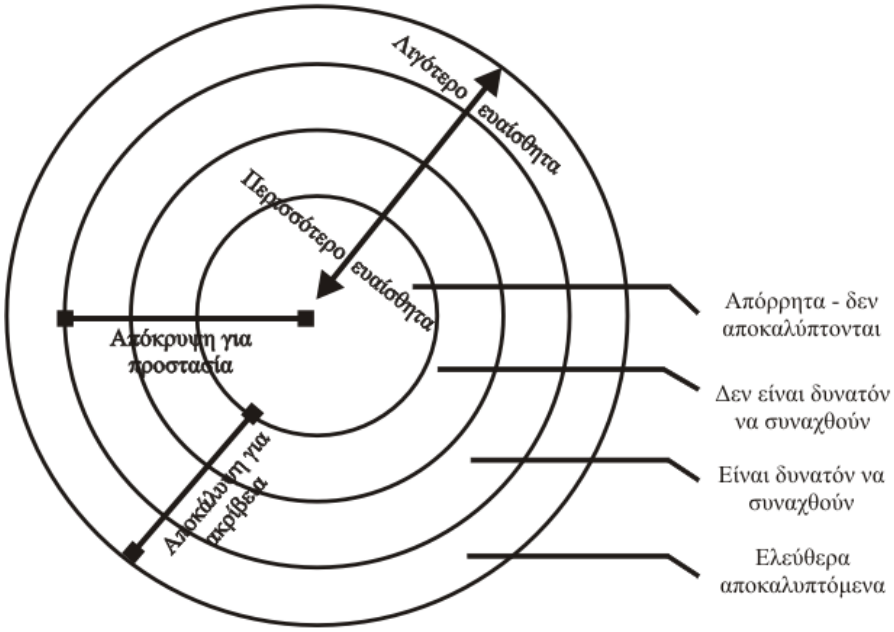
- *Ύπαρξη.* Μερικές φορές είναι σημαντικό να μην αποκαλυφθεί καν ότι κάποια δεδομένα *υπάρχουν*, ανεξάρτητα από το αν θα γίνει γνωστή η τιμή τους. Για παράδειγμα, αν αποκαλυφθεί ότι σε ένα σύστημα διαχείρισης προσωπικού υπάρχει πεδίο *πολιτικά φρονήματα* οι συνέπειες για τον οργανισμό θα είναι ιδιαίτερα δυσάρεστες, ανεξάρτητα από το τι καταχωρείται στα πεδία αυτά.

- *Πιθανή τιμή.* Κατά περίπτωση, είναι δυνατό να συναχθεί ότι ένα δεδομένο έχει κάποια *πιθανή τιμή*. Για παράδειγμα, αν υποβάλλουμε την ερώτηση «πόσοι κατοικούν στη διεύθυνση *Αρίων 03*» και λάβουμε απάντηση *3* και κατόπιν την ερώτηση «πόσοι κατοικούν στη διεύθυνση Αρίων 03 και έχουν καταδικασθεί για φοροδιαφυγή» και λάβουμε την απάντηση 1, συνάγουμε ότι υπάρχει πιθανότητα 33% για κάθε έναν από τους κατοίκους της συγκεκριμένης διεύθυνσης να έχει καταδικασθεί για φοροδιαφυγή. Αν παράλληλα γνωρίζουμε –από άλλες πηγές- ότι μόνον ένας είναι ελεύθερος επαγγελματίας και οι άλλοι δύο μισθωτοί, η κατανομή των πιθανοτήτων αλλάζει σημαντικά.

8.5.2.2 Ακρίβεια έναντι ασφάλειας

Είναι δυνατόν να επιτύχουμε απόλυτη ασφάλεια στα ευαίσθητα δεδομένα φροντίζοντας να μην αποκαλύπτονται καθόλου, παρά μόνον στους χρήστες που αντλούν το σχετικό δικαίωμα από την πολιτική ασφάλειας. Από την άλλη πλευρά, είναι πιθανόν κάποια ευαίσθητα δεδομένα να είναι απαραίτητα για καθ' όλα σύννομους σκοπούς που δεν διακυβεύουν την ασφάλεια. Για παράδειγμα θα μπορούσε το ιατρικό ιστορικό των φοιτητών να μελετηθεί μαζί με τους βαθμούς τους για εξαγωγή ιατρικών συμπερασμάτων σχετικά με το πόσο επηρεάζεται η απόδοση των φοιτητών από συγκεκριμένες παθήσεις. Επίσης, ο μισθός και το φύλο των εργαζομένων θα μπορούσαν να χρησιμοποιηθούν για να εξαχθούν στατιστικά σχετικά με το αν υπάρχει ανισοκατανομή αμοιβών μεταξύ των δύο φύλλων. Οι δύο τύποι αποκαλύψεων ευαίσθητων δεδομένων που αναφέρθηκαν στα παραδείγματα δεν αποκαλύπτουν σε καμία περίπτωση την ταυτότητα των υποκειμένων, συνεπώς δεν θεωρούνται επικίνδυνοι.

Συνολικά σε σχέση με τα ευαίσθητα δεδομένα έχουμε το ζήτημα της *ακρίβειας*, να προστατευθούν όσο το δυνατόν καλύτερα τα δεδομένα, επιτρέποντας ταυτόχρονα την πρόσβαση σε όσο το δυνατόν περισσότερα μη ευαίσθητα δεδομένα. Η συσχέτιση μεταξύ ακρίβειας και ασφάλειας φαίνεται στο σχήμα που ακολουθεί, όπου οι κατηγορίες δεδομένων αναπαρίστανται με ομόκεντρους κύκλους. Στο κέντρο υπάρχουν τα πλέον ευαίσθητα δεδομένα και ο βαθμός ευαισθησίας μειώνεται όσο απομακρυνόμαστε προς την περιφέρεια. Στη χάραξη της πολιτικής ασφάλειας θα πρέπει να καταταχθούν τα δεδομένα σε περιοχές ενώ στην υλοποίηση της πολιτικής θα πρέπει να ληφθούν τα κατάλληλα μέτρα προκειμένου να εξασφαλισθεί η τήρηση των αποφάσεων. Θα πρέπει να σημειωθεί ότι, στη γενική περίπτωση, δεν είναι εύκολο να εξαλειφθεί πλήρως η αποκάλυψη ευαίσθητων δεδομένων, όπως θα φανεί στο επόμενο εδάφιο.



8.5.3 Συμπερασμός

Το πρόβλημα του συμπερασμού σχετίζεται με τη συναγωγή ευαίσθητων δεδομένων από δεδομένα που δεν είναι ευαίσθητα και συνεπώς δεν προστατεύονται με την ίδια αυστηρότητα. Θα καταδείξουμε διάφορους τρόπους συμπερασμού, χρησιμοποιώντας παραδείγματα μία βάση δεδομένων φοιτητών που περιέχει το όνομα, την πτέρυγα κοιτώνα, το φύλο, τη φυλή, την ηλικία, την λαμβανόμενη οικονομική βοήθεια και τη χρήση φαρμακευτικών ουσιών από φοιτητές. Στη βάση δεδομένων αυτή το όνομα και η πτέρυγα κοιτώνα είναι αδιαβάθμητα, το φύλο η φυλή και η ηλικία διαβαθμισμένα και η οικονομική βοήθεια καθώς και η χρήση φαρμακευτικών ουσιών είναι απόρρητα. Στα παραδείγματα θα χρησιμοποιηθεί το κάτωθι στιγμιότυπο του πίνακα:

Όνομα	ΠτέρΚοιτ	Φύλο	Φυλή	Ηλικία	ΟικΒοηθ	Φάρμακα
Adams	Holmes	M	C	32	5000	1
Bailey	Grey	M	B	28	0	0
Chin	West	F	A	27	3000	0
Dewitt	Gray	M	B	28	1000	3
Earhart	Holmes	F	C	31	2000	1
Fein	West	F	C	26	1000	0
Groff	West	M	C	34	4000	3
Hill	Holmes	F	B	23	5000	2
Koch	West	F	C	21	0	1
Liu	Grey	F	A	28	0	2
Majors	Grey	M	C	22	2000	2

(η μονή υπογράμμιση υποδηλώνει διαβαθμισμένα γνωρίσματα, η διπλή υπογράμμιση υποδηλώνει απόρρητα).

8.5.3.1 Ευθεία επίθεση για συμπερασμό

Σε μία ευθεία επίθεση ο χρήστης προσπαθεί να συμπεράνει τιμές ευαίσθητων δεδομένων μέσω ερωτήσεων που επιστρέφουν λίγες εγγραφές. Η επίθεση επιτυγχάνει πλήρως όταν μπορεί να σχηματισθεί ερώτηση που να επιστρέφει *ακριβώς* μία εγγραφή. Για παράδειγμα η ερώτηση

```
select όνομα from φοιτητές where φύλο = 'M' and φάρμακα = 1
```

επιστρέφει μία μόνο εγγραφή, αυτή του Adams, αποκαλύπτοντας έτσι ότι ο χρήστης αυτός έχει τιμή 1 στο πεδίο *φάρμακα*. Η περίπτωση αυτή είναι εξαιρετικά απλή ωστόσο, διότι το ΣΔΒΔ μπορεί εύκολα να δει ότι η συνθήκη επιλογής είναι μία σύζευξη που περιέχει ένα απόρρητο δεδομένο και να αρνηθεί να την απαντήσει. Μία πιο πολύπλοκη περίπτωση είναι η ακόλουθη:

```
select όνομα from φοιτητές
where (φύλο = 'M' and φάρμακα = 1) OR
           (φύλο <> 'M' and φύλο <> 'F') OR
           (πτερΚοιτ = 'ATX0NN@45Q!46')
```

Εδώ υπάρχει η ίδια συνθήκη με προηγουμένως, αλλά επαυξημένη με δύο διαζεύξεις στις οποίες μετέχουν γνωρίσματα χαρακτηρισμένα ως αδιαβάθμητα (πτέρυγα κοιτώνα) ή διαβαθμισμένα (φύλο). Το ΣΔΒΔ βλέποντας τις διαζεύξεις θα μπορούσε να επιτρέψει την εκτέλεση της ερώτησης, η οποία όμως είναι ταυτόσημη με την προηγούμενη, καθώς οι δύο συνθήκες που προστέθηκαν με τις διαζεύξεις δεν αληθεύουν για καμία εγγραφή (η πρώτη είναι αδύνατη καθώς το φύλο πρέπει να είναι M ή F, ενώ κανείς φοιτητές δεν μένει στην πτέρυγα `ATX0NN@45Q!46`).

Ένα μέτρο ενάντια σε τέτοιου είδους επιθέσεις είναι η εφαρμογή του κανόνα *περισσότερες από ν γραμμές απάντησης*, βάσει του οποίου η απάντηση *δεν δίνεται στον χρήστη* αν περιέχει λιγότερες από *ν* εγγραφές. Ο κανόνας αυτός όμως εύκολα παρακάμπτεται, καθώς είναι π.χ. δυνατόν να σχηματισθεί η ερώτηση

```
select φοιτητές.όνομα from φοιτητές, ν_συν_ένα_εγγραφές
where (φοιτητές.φύλο = 'M' φοιτητές.and φάρμακα = 1) OR
          (φοιτητές.φύλο <> 'M' and φοιτητές.φύλο <> 'F') OR
          (φοιτητές. πτερΚοιτ = 'ATX0NN@45Q!46')
```

Ο πίνακας ν_συν_ένα_εγγραφές στην πιο πάνω ερώτηση περιέχει *ν+1* οποιεσδήποτε εγγραφές και δεν παίζει κανέναν απολύτως ρόλο στην ερώτηση πέραν του να προκαλεί την εμφάνιση κάθε εγγραφής του αποτελέσματος *ν+1 φορές*, καλύπτοντας έτσι τις προϋποθέσεις του κανόνα *περισσότερες από ν γραμμές απάντησης*. Και οι *ν+1* γραμμές ωστόσο αναφέρονται στον Adams.

Ένας πιο αποτελεσματικός κανόνας είναι ο *ν αντικείμενα άνω του κ%.* Ο κανόνας αυτός ορίζει ότι το αποτέλεσμα δεν θα εμφανιστεί αν *ν* (ή λιγότερα) αντικείμενα αντιπροσωπεύουν το *κ%* (ή περισσότερο) της απάντησης. Στο προηγούμενο παράδειγμα, *ένα μόνο αντικείμενο* (ο Adams) αντιπροσωπεύει το 100% της απάντησης, συνεπώς το αποτέλεσμα δεν εμφανίζεται.

8.5.3.2 Έμμεση επίθεση για συμπερασμό

Ένας κανόνας που χρησιμοποιείται από πολλούς οργανισμούς, συμπεριλαμβανομένων των στατιστικών υπηρεσιών, είναι να επιτρέπουν την εξαγωγή μόνον στατιστικών (αθροίσματα, μέσοι όροι, πληθάριθμοι κ.λπ.) και όχι μεμονωμένων τιμών οι οποίες θα μπορούσαν να αποκαλύψουν την ταυτότητα του υποκειμένου (π.χ. διεύθυνση, όνομα κ.ά.). Η έμμεση επίθεση προσπαθεί να αποκαλύψει εξατομικευμένες πληροφορίες με βάση ανώνυμα στατιστικά μεγέθη. Πολλές φορές η έμμεση επίθεση απαιτεί εργασία και εκτός της βάσης δεδομένων. Στις επόμενες παραγράφους θα παραθέσουμε παραδείγματα εμμέσων επιθέσεων για συμπερασμό.

Άθροισμα

Μία μορφή επίθεσης είναι προσπάθεια συναγωγής μιας τιμής μέσω αθροίσματος. Για παράδειγμα, στον πίνακα που απεικονίζεται στο εδάφιο 8.5.3 θα μπορούσε να θεωρηθεί ασφαλές να αναφερθεί η συνολική οικονομική βοήθεια κατά φύλο και πτέρυγα κοιτώνα. Το αποτέλεσμα είναι ο πίνακας

	Holmes	Grey	West	Σύνολο
M	5000	3000	4000	12000
F	7000	0	4000	11000
Σύνολο	12000	3000	8000	23000

Παρατηρήστε τη μηδενική τιμή στο κελί (F, Grey) που υποδεικνύει ότι καμία από τις γυναίκες που διαμένει στην πτέρυγα Grey δεν λαμβάνει οικονομική βοήθεια. Ο τύπος αυτός επίθεσης χρησιμοποιείται ΄κυρίως για *αρνητικό συμπερασμό*.

Πληθάριθμος

Οι πληθάριθμοι μπορούν να χρησιμοποιηθούν σε συνδυασμό με τα αθροίσματα για να αποκαλύψουν επακριβείς τιμές. Συνήθως, στατιστικές υπηρεσίες ανακοινώνουν αθροίσματα και πληθάριθμους για να υπολογίζονται μέσοι όροι (ή μέσους όρους και πληθάριθμους, που οδηγούν σε υπολογισμό των αθροισμάτων).

Θεωρήστε τον ακόλουθο πίνακα, που περιέχει τους πληθάριθμους φοιτητών ανά φύλο και πτέρυγα κοιτώνα. Παρατηρήστε τα κελιά (M, Holmes) και (M, West) που έχουν τιμή 1. Σε συνδυασμό με τον προηγούμενο πίνακα, αποκαλύπτεται ότι ο

μοναδικός άρρεν φοιτητής που διαμένει στην πτέρυγα Holmes λαμβάνει οικονομική βοήθεια ίση με 5000 και ο μοναδικός άρρεν φοιτητής που διαμένει στην πτέρυγα West λαμβάνει οικονομική βοήθεια ίση με 4000. Δεδομένου μάλιστα ότι η πτέρυγα κοιτώνα και τα ονόματα είναι αδιαβάθμητα, είναι δυνατόν να εξαχθούν και τα ονόματα των υποκειμένων.

	Holmes	Grey	West	Σύνολο
M	1	3	1	5
F	2	1	3	6
Σύνολο	3	4	4	11

Διάμεσοι

Η επίθεση για αποκάλυψη με διαμέσους είναι λίγο πιο περίπλοκη. Για να αποκαλύψουμε την τιμή ενός ευαίσθητου δεδομένου A της εγγραφής r, χρειαζόμαστε δύο σύνολα X και Y τέτοια ώστε:

1. $X \cap Y = \{r\}$
2. ελάχιστος(X, A) = μέγιστος(Y, A)
3. $t[A] \neq t'[A]\ \forall t, t' \in X \cup Y, t \neq t'$

τότε r[A] = ελάχιστος(X, A). Η επίθεση καλείται *επίθεση με διαμέσους* διότι η εγγραφή βρίσκεται στο «μέσον» των δύο συνόλων.

Ερωτήσεις εντοπισμού

Όπως αναφέρθηκε στο εδάφιο 8.5.3.1, τα ΣΔΒΔ είναι δυνατόν να αρνηθούν να δώσουν απάντηση στον χρήστη αν παραβιάζεται ο κανόνας του *ν αντικείμενα άνω του κ%.* Ο επιτιθέμενος εδώ αντί να δώσει μία ερώτηση που επιστρέφει ένα αντικείμενο, δίνει μία ερώτηση που επιστρέφει *ν+1* αντικείμενα και άλλη μία ερώτηση που επιστρέφει *ν* αντικείμενα, τηρώντας έτσι τον κανόνα. Η διαφορά μεταξύ των δύο απαντήσεων μας αποκαλύπτει τιμές για το αντικείμενο που διαφοροποιεί τα δύο σύνολα.

Ας θεωρήσουμε τη συνθήκη

```
where (sex = F) and (race = C) and (address = Holmes)
```

η οποία προσδιορίζει *ακριβώς μία πλειάδα* στη βάση δεδομένων, συνεπώς το σύστημα θα μπορούσε να αρνηθεί να παρουσιάσει ΄την απάντηση σε οποιαδήποτε ερώτηση τη χρησιμοποιεί. Η συνθήκη μπορεί να γραφεί ισοδύναμα

```
where (φύλο = F) and NOT ((φυλή <> C) OR (πτερΚοιτ <> Holmes))
```

Διασπούμε τώρα τη συνθήκη σε δύο τμήματα

```
1. where (φύλο = F)
2. where (φύλο = F) and ((φυλή <> C) OR (πτερΚοιτ <> Holmes))
```

Παρατηρήστε ότι το σύνολο που προσδιορίζεται από τη συνθήκη (2) περιλαμβάνει 5 εγγραφές, δηλαδή *ακριβώς μία εγγραφή λιγότερη* από το σύνολο που προσδιορίζεται από τη συνθήκη (1). Έτσι, από τα αποτελέσματα των δύο ερωτήσεων είναι δυνατόν να συναχθούν πληροφορίες για την εγγραφή που διαφοροποιεί τα δύο σύνολα απαντήσεων.

Γραμμική ευπάθεια συστημάτων

Η γραμμική ευπάθεια συστημάτων είναι μία γενίκευση των ερωτήσεων εντοπισμού. Χρησιμοποιώντας αλγεβρικά συστήματα (και εφ' όσον τα περιεχόμενα της βάσης δεδομένων συμβεί να είναι «βολικά») είναι δυνατόν να αποκαλυφθούν ευαίσθητα δεδομένα βάσει μιας συλλογής απαντήσεων. Πιο συγκεκριμένα, ας θεωρήσουμε της ερωτήσεις ε_1 έως ε_5 που δίνουν τα ακόλουθα αθροίσματα:

$$\varepsilon_1 = \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4 + \sigma_5$$
$$\varepsilon_2 = \sigma_1 + \sigma_2 + \sigma_4$$
$$\varepsilon_3 = \sigma_3 + \sigma_4$$
$$\varepsilon_4 = \sigma_4 + \sigma_5$$
$$\varepsilon_5 = \sigma_2 + \sigma_5$$

Καμία από τις ερωτήσεις δεν αποκαλύπτει κάποιο συγκεκριμένο δεδομένο, καθώς αποκαλύπτονται μόνον αθροίσματα. Ουσιαστικά όμως, το σύνολο ερωτήσεων αυτό είναι ένα γραμμικό σύστημα πέντε εξισώσεων με πέντε αγνώστους, το οποίο επιλυόμενο θα μας δώσει τις επακριβείς τιμές για όλα τα στοιχεία δεδομένων σ_i. Με κατάλληλη δόμηση, το σχήμα αυτό μπορεί να χρησιμοποιηθεί και για αποκάλυψη μη αριθμητικών τιμών.

Συνάθροιση

Η συνάθροιση είναι μία τεχνική συμπερασμού που προσπαθεί να συνάγει ευαίσθητα δεδομένα από λιγότερο ευαίσθητα. Η βασική εργασία με τη συνάθροιση είναι να ξεκινά ο επιτιθέμενος με ένα μεγάλο σύνολο, το οποίο να περιορίζει διαδοχικά μέχρις ότου να φτάσει σε μία και μόνη εγγραφή. Εναλλακτικά, μπορεί να δημιουργεί ανεξάρτητα σύνολα που να περιέχουν όλα την επιθυμητή εγγραφή, και που παράλληλα η τομή τους είναι ακριβώς η επιθυμητή εγγραφή, οπότε και πάλι είναι δυνατή η αποκάλυψη της τιμής του ευαίσθητου δεδομένου.

Η αντιμετώπιση της συνάθροισης είναι δύσκολη, μια και πρέπει το ΣΔΒΔ να απομνημονεύει τις απαντήσεις που έχει δώσει έτσι ώστε να μπορεί να αποφασίσει αν πρέπει να αποκαλύψει μία απάντηση στον χρήστη ή όχι. Η συνάθροιση είναι

Αντιμετώπιση προσπαθειών συμπερασμού

Για την αντιμετώπιση των προσπαθειών συμπερασμού μπορούν να χρησιμοποιηθούν δύο κύριες τεχνικές:

1. να ελέγχονται οι ερωτήσεις που υποβάλλονται στη βάση δεδομένων
2. να ελέγχονται τα μεμονωμένα στοιχεία δεδομένων που χρησιμοποιούνται για τη διαμόρφωση των απαντήσεων

Βάσει της τεχνικής ελέγχου των ερωτήσεων, κάθε ερώτηση που υποβάλλεται στη βάση δεδομένων αναλύεται πριν την εκτέλεσή της, προκειμένου να διαπιστωθεί αν αποκαλύπτει ευαίσθητα δεδομένα. Από τα προηγούμενα παραδείγματα ωστόσο έχει καταστεί σαφές ότι δεν είναι πάντα εύκολο να καταλάβουμε αν μία ερώτηση αποκαλύπτει ή όχι ευαίσθητα δεδομένα, εκτός ίσως από την περίπτωση της ευθείας επίθεσης. Συνεπώς στα πραγματικά συστήματα, ο έλεγχος ερωτήσεων χρησιμοποιείται μόνον για άμυνα έναντι της συγκεκριμένης μορφής επίθεσης.

Σε σχέση με τον έλεγχο των μεμονωμένων στοιχείων δεδομένων, είναι δυνατόν να εφαρμοσθούν δύο κύριες προσεγγίσεις, η *απόκρυψη* (suppression) και η *παραλλαγή*

(concealing). Όταν χρησιμοποιείται απόκρυψη, ερωτήσεις που αναφέρονται σε ευαίσθητα δεδομένα απορρίπτονται συνολικά και δεν παρέχεται καμία απάντηση. Όταν χρησιμοποιείται παραλλαγή, δίνεται απάντηση που είναι *προσεγγιστική* της πραγματικής τιμής, αλλά *όχι* η ίδια η πραγματική τιμή.

Οι δύο αυτές προσεγγίσεις αντικατοπτρίζουν την αντίθεση μεταξύ ακρίβειας και ασφάλειας. Η απόκρυψη δίνει πάντα ακριβείς απαντήσεις, αλλά *δεν απαντά καθόλου* σε ένα σύνολο ερωτημάτων. Η παραλλαγή παρέχει απαντήσεις σε περισσότερα ερωτήματα, αλλά δίνει λιγότερο ακριβή αποτελέσματα. Η επιλογή εξαρτάται από τη σημασιολογία της βάσης δεδομένων και –κατ' επέκταση– με τη σημασία της ακρίβειας των αποτελεσμάτων και της ασφάλειας των δεδομένων.

Τεχνική 1: απόκρυψη αποκαλυπτικών απαντήσεων

Ο κανόνας *ν αντικείμενα άνω του κ%* μπορεί να χρησιμοποιηθεί για την αξιολόγηση του αν συγκεκριμένα στατιστικά μεγέθη πρέπει να εμφανιστούν. Ας θεωρήσουμε τον ακόλουθο πίνακα που δίνει το πλήθος των φοιτητών ανά φύλο και πτέρυγα κοιτώνα. Όπως είδαμε στο εδάφιο περί επιθέσεων με πληθαρίθμους, τα κελιά με τιμή «1» μπορούν να χρησιμοποιηθούν για συμπερασμό δεδομένων (σε συνδυασμό με αθροίσματα και μέσους όρους), οπότε πρέπει να αποκρυφθούν. (Τα κελιά αυτά εντοπίζονται με τον κανόνα *ν αντικείμενα άνω του κ%* διότι στη διαμόρφωση της τιμής τους συμμετέχει κατά 100% ένα μόνο αντικείμενο).

	Holmes	Grey	West	Σύνολο
M	1	3	1	5
F	2	1	3	6
Σύνολο	3	4	4	11

Σημειώστε ότι, δεδομένης της εμφάνισης συνόλων, η απόκρυψη των συγκεκριμένων κελιών μόνο δεν αρκεί, καθώς η τιμή τους μπορεί να συναχθεί από τα άλλα κελιά της γραμμής/στήλης. Είναι έτσι απαραίτητο να αποκρύπτονται τουλάχιστον δύο κελιά ανά γραμμή/στήλη. Αν ωστόσο δεν εμφανίζονται σύνολα, η απόκρυψη ενός μόνο κελιού ανά γραμμή/στήλη είναι επαρκής.

Τεχνική 2: Συνδυασμός αποτελεσμάτων

Μια εναλλακτική προσέγγιση προς την απόκρυψη αποκαλυπτικών απαντήσεων είναι ο συνδυασμός γραμμών ή στηλών της απάντησης προκειμένου να προστατευθούν ευαίσθητα δεδομένα. Για παράδειγμα, ο ακόλουθος πίνακας που δίνει πληθάριθμους φοιτητών ανά φύλο και χρήση φαρμάκων περιέχει αρκετά κελιά με την τιμή «1» που δεν πρέπει να εμφανισθούν, καθώς μπορούν να χρησιμοποιηθούν για συμπερασμό δεδομένων σε συνδυασμό με αθροίσματα και μέσους όρους).

	0	1	2	3
M	1	1	1	2
F	2	2	2	0

Βάσει της τεχνικής του συνδυασμού αποτελεσμάτων είναι δυνατόν να παρουσιάσουμε συγκεντρωτικά στοιχεία για τα ζεύγη στηλών (0, 1) και (2, 3), έτσι ώστε το τελικώς παρουσιαζόμενο αποτέλεσμα είναι το ακόλουθο:

	0 ή 1	2 ή 3

M	2	3
F	4	2

Μία ακόμη εκδοχή του συνδυασμού αποτελεσμάτων είναι να παρουσιάζονται όχι επακριβείς τιμές αλλά διαστήματα τιμών. Για παράδειγμα, αν μία ερώτηση ζητά τον μέσο όρο της οικονομικής βοήθειας θα μπορούσε να δίνεται ως απάντηση ένα από τα *0 έως 1999 ευρώ*, *2000 έως 3999 ευρώ*, *πάνω από 4000 ευρώ*. Με τον τρόπο αυτό, ακόμη και αν υπάρχει *ακριβώς μία* εγγραφή στο σύνολο πάνω στο οποίο υπολογίζεται ο μέσος όρος, η ακριβής τιμή δεν γίνεται γνωστή.

Τέλος, μία παραλλαγή της ανωτέρω δυνατότητας είναι η στρογγύλευση τιμών. Οι τιμές μπορούν να στρυγγυλεύονται σε κάποιο πολλαπλάσιο, π.χ. του 10, δίνοντας περιοχές τιμών 0-5, 6-15, 16-25 κ.ο.κ.

Τεχνική 3: Τυχαίο δείγμα

Βάσει της τεχνικής του τυχαίου δείγματος, τα στατιστικά μεγέθη στις απαντήσεις δεν υπολογίζονται κατόπιν επεξεργασίας *όλων* των δεδομένων αλλά ενός τυχαίου δείγματος. Το δείγμα πρέπει να είναι αρκετά μεγάλο και στατιστικά αντιπροσωπευτικό του πληθυσμού των δεδομένων ώστε να δίνονται έγκυρα αποτελέσματα. Σημειώστε ότι αν η *ίδια* ερώτηση υποβληθεί δύο φορές *χωρίς να έχει μεσολαβήσει αλλαγή στη βάση δεδομένων* πιθανότατα θα δώσει διαφορετικές απαντήσεις, καθ' ότι θα υπολογιστεί χρησιμοποιώντας διαφορετικά υποσύνολα δεδομένων. Η τεχνική αυτή αχρηστεύει όλους τους συμπερασμούς που βασίζονται σε στατιστικά μεγέθη, καθώς δεν υπάρχει καμία εγγύηση ότι τα αποτελέσματα βασίζονται στα ίδια δεδομένα.

Τεχνική 4: Εισαγωγή τυχαίου θορύβου

Βάσει της τεχνικής της εισαγωγής τυχαίου θορύβου, σε κάθε τιμή ευαίσθητου δεδομένου *ΤΕΔ* που διαβάζεται από τη βάση δεδομένων προκειμένου για υπολογισμού στατιστικού μεγέθους προστίθεται μία τυχαία τιμή X με $-\varepsilon \leq X \leq \varepsilon$ που ακολουθεί (συνήθως) την ομοιόμορφη κατανομή. Ως συνέπεια, κάποιες τιμές θα εμφανίζονται μεγαλύτερες από τις πραγματικές και κάποιες άλλες μικρότερες, στον υπολογισμό ωστόσο αθροισμάτων και μέσων όρων τα αποτελέσματα θα είναι εν γένει ακριβή, ειδικότερα αν αυτά βασίζονται σε μεγάλο πλήθος εγγραφών. Μερικές φορές ο *θόρυβος* που συσχετίζεται με μία τιμή δεν υπολογίζεται τυχαία κατά την επεξεργασία αλλά κατά την *αποθήκευση της τιμής* και αποθηκεύεται μαζί μ' αυτή, προκειμένου να δίνονται ταυτόσημες απαντήσεις σε διαδοχικές εκτελέσεις της ίδιας ερώτησης.

Τεχνική 5: Απομνημόνευση αποτελεσμάτων ερωτήσεων

Βάσει της τεχνικής της αυτής, το ΣΔΒΔ απομνημονεύει τα σύνολα που χρησιμοποιεί για να υπολογίσει τις απαντήσεις που έχει δώσει στους χρήστες και αρνείται να αποκαλύψει απαντήσεις που βασίζονται σε *παραπλήσια σύνολα*. Με τον τρόπο αυτό αντιμετωπίζονται αρκετές προσπάθειες συμπερασμού που προσπαθούν να εντοπίσουν μία εγγραφή r με ερωτήσεις πάνω στα σύνολα X και $X \cup \{r\}$.

8.5.4 Υποχρεωτικός έλεγχος προσπέλασης

Σε αντίθεση με τον *κατ' επιλογήν έλεγχο προσπέλασης* όπου κάθε χρήστης-ιδιοκτήτης πινάκων έχει το δικαίωμα να ορίσει *ποιος χρήστης έχει ποιο δικαίωμα* στα δεδομένα των πινάκων του, στον υποχρεωτικό έλεγχο προσπέλασης το σύστημα *επιβάλλει* να διενεργούνται έλεγχοι κατά την προσπέλαση των δεδομένων *ανεξαρτήτως*

ιδιοκτησίας των δεδομένων. Στο σχήμα αυτό, κάθε δεδομένο έχει *μία διαβάθμιση* και κάθε χρήστης ένα *επίπεδο εξουσιοδότησης*. Οι κανόνες που ακολουθούνται στις προσπελάσεις δεδομένων είναι:

- Η ανάγνωση επιτρέπεται μόνο αν η εξουσιοδότηση του χρήστη είναι μεγαλύτερη ή ίση από τη διαβάθμιση του δεδομένου (no read-up)
- Η εγγραφή επιτρέπεται μόνο αν η εξουσιοδότηση του χρήστη είναι μικρότερη ή ίση από τη διαβάθμιση του δεδομένου (no write-down)

Ως παραδείγματα διαβάθμισης–εξουσιοδότησης μπορούμε να θεωρήσουμε τα (άκρως απόρρητο, απόρρητο, εμπιστευτικό, αδιαβάθμητο). Σε μερικά συστήματα ενδέχεται οι χαρακτηρισμοί διαβάθμισης και εξουσιοδότησης να συμπληρώνονται και με *κατηγορίες ασφάλειας*, όπως περιγράφεται στο εδάφιο 4.1.2. Τα σχήματα υποχρεωτικού ελέγχου προσπέλασης

8.5.4.1 Διακριτότητα χαρακτηρισμών ασφάλειας

Οι χαρακτηρισμοί ασφάλειας μπορούν να αποδίδονται σε μικρές ή μεγάλες ενότητες δεδομένων, ανάλογα με τις ανάγκες ασφάλειας:

- *Σε επίπεδο πλειάδας:* η πλειάδα που περιγράφει την *Επιχείρηση Moonraker* είναι άκρως απόρρητη, ενώ αυτή που περιγράφει την *Καθαροί δρόμοι 2003* είναι αδιαβάθμητη
- *Σε επίπεδο γνωρίσματος (attribute).* Το γνώρισμα *προϋπολογισμός* για την πλειάδα που αφορά τα ειδικά κονδύλια του υπουργείου Άμυνας είναι άκρως απόρρητο – το ίδιο γνώρισμα για την πλειάδα που αφορά συνδετήρες και συρραπτικά του υπουργείου Άμυνας είναι αδιαβάθμητο.
- *Σε επίπεδο συνδυασμού γνωρισμάτων.* Έστω γνωρίσματα εργοδότης και θέση με δυνατές τιμές (Lottery Inc, Ocean Travel, CIA) και (υπάλληλος καθαριότητας, γραμματέας, πράκτορας, διευθυντής). Το κάθε γνώρισμα είναι εμπιστευτικό, ο συνδυασμός τους είναι άκρως απόρρητος καθώς δεν βλάπτει να γνωρίζει κανείς ότι ένας εργαζόμενος είναι *πράκτορας* (μπορεί να είναι πράκτορας λαχείων ή ταξιδίων) , ούτε να γνωρίζει ότι δουλεύει στη CIA (μπορεί να είναι υπάλληλος καθαριότητας ή γραμματέας), αλλά ο *συνδυασμός* είναι εν δυνάμει επικίνδυνο να αποκαλυφθεί (πράκτορας CIA, διευθυντής CIA).
- *Σε επίπεδο στατιστικών μεγεθών.* Μέσοι όροι, μέγιστα, ελάχιστα, αθροίσματα και άλλα στατιστικά μεγέθη μπορεί να έχουν διαφορετικό επίπεδο διαβάθμισης από τα επί μέρους στοιχεία βάσει των οποίων υπολογίζονται. Για παράδειγμα ο συνολικός προϋπολογισμός του υπουργείου Εθνικής Άμυνας είναι αδιαβάθμητος (ανακοινώνεται στον προϋπολογισμό), το κονδύλι που κατανέμεται στα πυρηνικά όπλα είναι άκρως απόρρητο. Οι επί μέρους προμήθειες καυσίμων για τους πυραύλους είναι εμπιστευτικές

8.5.4.2 Γενικό σχήμα για πολυεπίπεδη ασφάλεια

Προκειμένου να είναι δυνατόν να υποστηριχθούν οι απαιτήσεις για τη διακριτότητα των χαρακτηρισμών ασφάλειας, είναι απαραίτητο να επεκταθεί το σχήμα των σχέσεων που αποθηκεύονται στο ΣΔΒΔ. Πιο συγκεκριμένα κάθε σχήμα σχέσης $R(A_1, A_2, \ldots, A_n)$ επεκτείνεται σε $R(A_1, C_1, A_2, C_2, \ldots, A_n, C_n, C_T)$ όπου C_i είναι η διαβάθμιση του στοιχείου δεδομένων A_i και C_T είναι ο χαρακτηρισμός ασφάλειας της

πλειάδας. Για τα C_i, C_T θα πρέπει $C_i \leq C_T$, $\forall 1 \leq i \leq n$ και το σύστημα εξετάζει τις τιμές τους σε σχέση με το επίπεδο εξουσιοδότησης του χρήστη, προκειμένου να αποφασίσει αν θα παρουσιάσει ή όχι κάποια στοιχεία στην απάντηση. Οι κανόνες που ισχύουν για την ανάκτηση δεδομένων είναι:

1. για να εμφανιστεί *συνολικά* μία εγγραφή στον χρήστη πρέπει η εξουσιοδότησή του να είναι ίση ή μεγαλύτερη από *μία τουλάχιστον* διαβάθμιση στοιχείου δεδομένων της εγγραφής. Αν η εξουσιοδότηση του χρήστη είναι μικρότερη *από όλες* τις διαβαθμίσεις στοιχείων της εγγραφής, η εγγραφή δεν εμφανίζεται καθόλου.

2. Για τις εγγραφές που θα εμφανιστούν, παρουσιάζονται οι αποθηκευμένες τιμές μόνο για τα στοιχεία δεδομένων εκείνα που η εξουσιοδότηση του χρήστη είναι μεγαλύτερη ή ίση από το διαβάθμιση ασφάλειας του δεδομένου. Για τα υπόλοιπα παρουσιάζεται η τιμή NULL, χαρακτηρισμένες με επίπεδο ασφαλείας ίσο με το επίπεδο εξουσιοδότησης του χρήστη. Επίσης, το επίπεδο ασφαλείας της πλειάδας παρουσιάζεται επίσης ίσο με το επίπεδο εξουσιοδότησης του χρήστη.

Ως παράδειγμα, ας θεωρήσουμε τον πίνακα *Εργαζόμενος* με τα ακόλουθα στοιχεία:

Όνομα		**Μισθός**		**Θέση**		**C_T**
Money Penny	U	5000	C	Secretary	U	C
Bond James	C	7000	S	Secret Agent	TS	TS

Αν κάποιος χρήστης με επίπεδο εξουσιοδότησης *C* επιχειρήσει να ανακτήσει όλες τις εγγραφές του πίνακα θα λάβει το ακόλουθο αποτέλεσμα:

Όνομα		**Μισθός**		**Θέση**		**C_T**
Money Penny	U	5000	C	Secretary	U	C
Bond James	C	null	C	Null	C	C

ενώ αν κάποιος χρήστης με επίπεδο εξουσιοδότησης *U* επιχειρήσει να ανακτήσει όλες τις εγγραφές του πίνακα, το αποτέλεσμα θα διαμορφωθεί ως εξής:

Όνομα		**Μισθός**		**Θέση**		**C_T**
Money Penny	U	null	U	Secretary	U	U

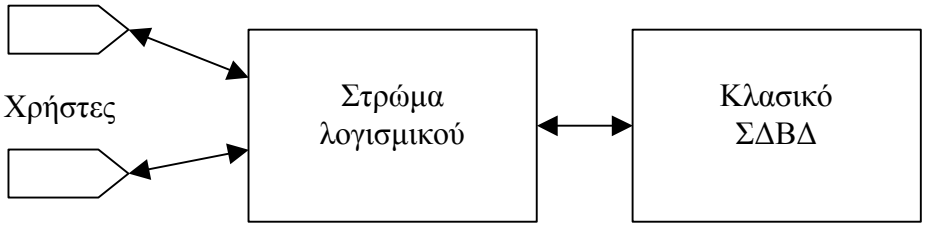
Ας υποθέσουμε τώρα ότι ένας χρήστης με επίπεδο εξουσιοδότησης *U* επιχειρεί να εισάγει την εγγραφή *('Bond James', U, 6000, U, 'Commander', U, U)*. Αν το σύστημα απορρίψει την εισαγωγή λόγω της ύπαρξης της $2^{ης}$ πλειάδας ουσιαστικά αποκαλύπτει την ύπαρξη μιας πλειάδας με υψηλότερο επίπεδο ασφάλειας. Η «σιωπηλή» απόρριψη κατά την οποία το σύστημα δεν εμφανίζει κάποιο μήνυμα λάθους αλλά δεν καταχωρεί την εγγραφή δεν αυξάνει την ασφάλεια, διότι αν ο χρήστης προσπαθήσει να ανακτήσει την εγγραφή που μόλις εισήγαγε θα διαπιστώσει την έλλειψή της και έτσι θα οδηγηθεί στο ίδιο συμπέρασμα (ύπαρξης εγγραφής με μεγαλύτερο επίπεδο ασφάλειας). Αντίστοιχες παρατηρήσεις ισχύουν για την περίπτωση που ο ίδιος χρήστης προσπαθήσει να ενημερώσει τον μισθό της Money Penny σε 4000.

Προκειμένου το σύστημα να μην αποκαλύπτει συνολικά την ύπαρξη δεδομένων με υψηλότερο επίπεδο ασφάλειας από την εξουσιοδότηση του χρήστη, εισαγωγές και ενημερώσεις όπως οι ανωτέρω τελικά *επιτρέπονται*, δημιουργώντας όμως *νέα στιγμιότυπα των εγγραφών*, όπως φαίνεται στο σχήμα που ακολουθεί:

Όνομα		**Μισθός**		**Θέση**		$\mathbf{C_T}$
Money Penny	U	5000	C	Secretary	U	C
Bond James	C	7000	S	Secret Agent	TS	TS
Bond James	U	6000	U	Commander	U	U
Money Penny	U	4000	C	Secretary	U	C

Τελικά καταλήγουμε σε ύπαρξη περισσοτέρων πλειάδων με *διαφορετικά επίπεδα ασφάλειας*, φαινόμενο που ονομάζεται *πολυστιγμιοτυπία* (multi-instantiation). Στους πίνακες που εφαρμόζεται η πολυστιγμιοτυπία ορίζεται η έννοια του *φαινόμενου κλειδιού*, το οποίο ορίζεται ως τα γνωρίσματα που θα αποτελούσαν το *πρωτεύον κλειδί* του πίνακα αν δεν εφαρμοζόταν η πολυεπίπεδη ασφάλεια. Στον πίνακα του παραδείγματος το φαινόμενο κλειδί είναι το γνώρισμα *Όνομα*. Όπως φαίνεται από τα παραδείγματα, σε ένα σύστημα με πολυεπίπεδη ασφάλεια είναι δυνατόν να υπάρξουν πολλές εγγραφές με το ίδιο φαινόμενο κλειδί, κάτι που δεν θα ήταν δυνατό αν δεν εφαρμοζόταν η πολυεπίπεδη ασφάλεια.

Προκειμένου να υποστηριχθούν από ένα ΣΔΒΔ χαρακτηριστικά πολυεπίπεδης ασφάλειας όπως αυτά που περιγράφηκαν ανωτέρω, η βέλτιστη λύση είναι να χρησιμοποιηθεί ένα ΣΔΒΔ που υποστηρίζει εγγενώς τις έννοιες αυτές. Ωστόσο, με δεδομένο ότι δεν υπάρχει μεγάλο πλήθος ΣΔΒΔ που να ενσωματώνει πλήρως όλα τα απαραίτητα χαρακτηριστικά, συχνά οδηγούμαστε σε υιοθέτηση *ημιβέλτιστων λύσεων* που περιλαμβάνουν τη χρήση ενός «συμβατικού» ΣΔΒΔ στο οποίο προστίθενται με διάφορες προσεγγίσεις τα επιθυμητά χαρακτηριστικά ασφάλειας ως ένα επιπλέον *στρώμα λογισμικού* όπως απεικονίζεται στο ακόλουθο σχήμα:



Οι πιο διαδεδομένες προσεγγίσεις είναι:

- Κλείδωμα ακεραιότητας με έμπιστη διαδικασία ελέγχου πρόσβασης (trusted access controller)
- Έμπιστο μετωπικό επίπεδο πρόσβασης (trusted front end)
- Φίλτρα περιορισμού (commutative filters)
- Κατανεμημένες-ομόσπονδες βάσεις δεδομένων
- Παράθυρα-όψεις

Σε αρκετές από τις προσεγγίσεις η μυστικότητα και η ακεραιότητα των δεδομένων προστατεύονται με τεχνικές όπως η *κρυπτογράφηση*, το *κλείδωμα ακεραιότητας* και το *κλείδωμα ευαισθησίας*. Αυτό είναι απαραίτητο διότι τα δεδομένα αποθηκεύονται

τελικά σε ένα κλασικό ΣΔΒΔ, συνεπώς θα μπορούσε κανείς να μην τα προσπελάσει μέσα από το πρόσθετο στρώμα λογισμικού αλλά να προσπαθήσει να τα προσπελάσει απ' ευθείας μέσα από το ΣΔΒΔ που τα αποθηκεύει. Στα επόμενα εδάφια περιγράφονται οι βασικές αυτές τεχνικές και εν συνεχεία αναλύονται οι προσεγγίσεις στην ενσωμάτωση πολυεπίπεδης ασφάλειας σε «συμβατικά» ΣΔΒΔ.

Τεχνική 1: Κρυπτογράφηση

Τα δεδομένα κρυπτογραφούνται και φυλάσσονται στη βάση δεδομένων σε κρυπτογραφημένη μορφή, έτσι ώστε ακόμη και αν ένας μη εξουσιοδοτημένος χρήστης μπορέσει να τα προσπελάσει να μην είναι καταληπτά (και άρα χρήσιμα) σ' αυτόν.

Η πρώτη προσέγγιση στην κρυπτογράφηση είναι να χρησιμοποιείται το ίδιο κλειδί για να κρυπτογραφούνται όλες οι τιμές. Η τεχνική αυτή ωστόσο έχει το μειονέκτημα ότι για την ίδια πραγματική τιμή πεδίου παράγεται *πάντα* η ίδια κρυπτογραφημένη τιμή και έτσι σε πεδία με μικρό διακριτό πλήθος τιμών (π.χ. φύλο (άνδρας/γυναίκα), φυλή (λευκός, μαύρος, ασιάτης)) είναι εύκολο να βρεθεί η αντιστοιχία με έναν από τους ακόλουθους δύο τρόπους:

1. επιλέγουμε εγγραφές των οποίων οι τιμές για το συγκεκριμένο γνώρισμα είναι γνωστές και εξάγουμε το κρυπτογραφημένο κείμενο
2. εισάγουμε εγγραφές με τις επιθυμητές τιμές και καταγράφουμε το κρυπτογραφημένο κείμενο που παράγεται.

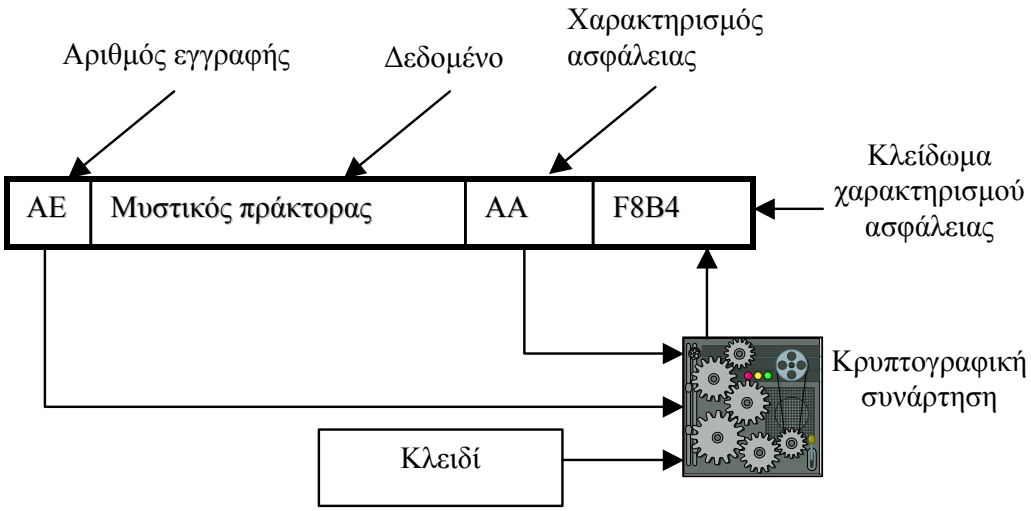
Για να αντιμετωπίσουμε το ζήτημα αυτό πρέπει να χρησιμοποιούμε διαφορετικό κλειδί ανά εγγραφή. Αυτό βέβαια καθιστά αναγκαία την εισαγωγή ενός σχήματος δημιουργίας, αποθήκευσης και ανάκτησης κλειδιών, ενώ επιπλέον απαιτεί πρόσθετο αποθηκευτικό χώρο για την αποθήκευση των κλειδιών και επιβαρύνει τις πράξεις εισόδου-εξόδου, καθώς για την ανάγνωση ενός δεδομένου πρέπει να ανακτάται τόσο η κρυπτογραφημένη τιμή όσο και το κλειδί.

Μία εναλλακτική προσέγγιση είναι αυτή της αλυσιδωτής κρυπτογράφησης. Βάσει της τεχνικής αυτής πέραν της συνάρτησης κρυπτογράφησης *cipher(data, key)* χρησιμοποιείται και μία συνάρτηση κερματισμού *hash(data)*, ενώ επίσης οι εγγραφές μίας σχέσης διατάσσονται σύμφωνα με την αποθήκευσή τους στον δίσκο. Στη θέση του κρυπτογραφούμενου γνωρίσματος A_c της ν-οστής εγγραφής γράφεται η τιμή $cipher(r_v[A_c], hash(r_{v-1}[A_c]))$ δηλαδή κρυπτογραφημένη τιμή του ίδιου γνωρίσματος της *προηγούμενης* πλειάδας χρησιμοποιείται ως κλειδί για την κρυπτογράφηση της τιμής του γνωρίσματος της τρέχουσας πλειάδας. Η τεχνική αυτή πλεονεκτεί στο ότι δεν σπαταλά χώρο για την αποθήκευση κλειδιών (απαιτείται να αποθηκευθεί μόνον το κλειδί της πρώτης εγγραφής), απαιτεί όμως να διαβάζονται οι πλειάδες *αυστηρά σειριακά* (δεν μπορούμε να διαβάσουμε απ' ευθείας την εγγραφή *ν* διότι δεν θα ξέρουμε το κλειδί αποκρυπτογράφησης παρά μόνον αν διαβάσουμε όλες τις προηγούμενες), ενώ και σε περιπτώσεις διαγραφής ή ενημέρωσης μιας ενδιάμεσης εγγραφής πρέπει να επανακωδικοποιήσουμε τις τιμές σε όλες τις επόμενές της.

Τεχνική 2: Κλείδωμα χαρακτηρισμού ασφάλειας

Το κλείδωμα χαρακτηρισμού ασφάλειας είναι συνδυασμός ενός μοναδικού προσδιοριστή (π.χ. της ταυτότητα εγγραφής) και του χαρακτηρισμού ασφάλειας. Ο στόχος της είναι να προστατεύσει τον χαρακτηρισμό ασφάλειας από (α) ανάγνωση από μη εξουσιοδοτημένους χρήστες (β) τροποποίηση. Ο υπολογισμός του

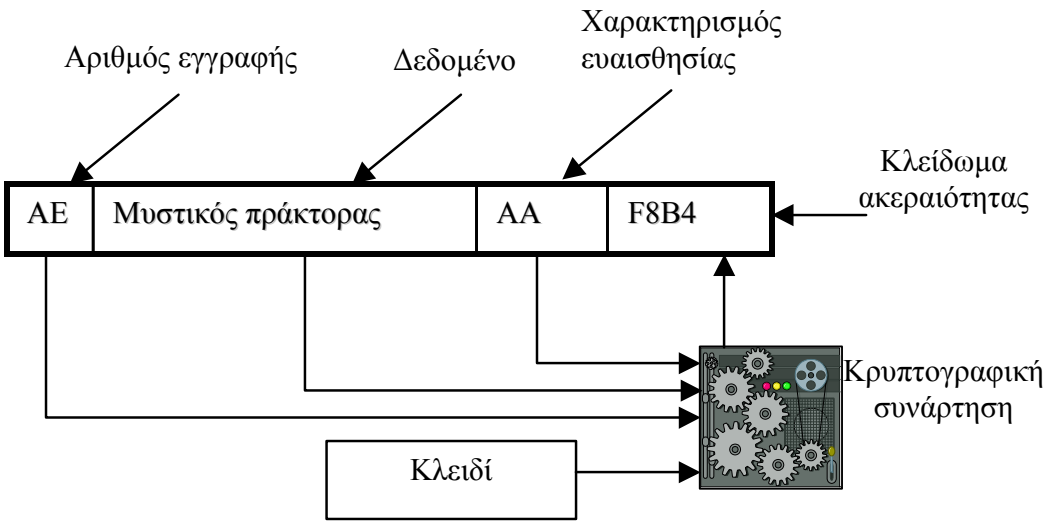
κλειδώματος γίνεται από το ενδιάμεσο στρώμα λογισμικού, ενώ σε κάθε προσπέλαση μέσω του στρώματος ελέγχεται αν η τιμή του κλειδώματος είναι έγκυρη σε σχέση με τον χαρακτηρισμό ασφάλειας και την ταυτότητα εγγραφής. Η παραγωγή του κλειδώματος χαρακτηρισμού ασφάλειας γίνεται όπως φαίνεται στο ακόλουθο σχήμα:



Το κλείδωμα χαρακτηρισμού ασφάλειας που παράγεται από την κρυπτογραφική συνάρτηση αποθηκεύεται μαζί με τον χαρακτηρισμό ασφάλειας. Αν η συνάρτηση είναι αμφίδρομη, είναι δυνατόν να αποθηκευθεί *μόνο* το κλείδωμα, με συνέπεια ο χαρακτηρισμός ασφάλειας να μην είναι αναγνώσιμος. Επιπρόσθετα, μη εξουσιοδοτημένες τροποποιήσεις στον χαρακτηρισμό ασφάλειας θα γίνουν αντιληπτές, καθώς το κλείδωμα ελέγχεται σε κάθε πρόσβαση για το αν είναι έγκυρο, σε σχέση με τον αριθμό εγγραφής και τον χαρακτηρισμό ασφάλειας. Σημειώνεται ότι μια και ο υπολογισμός του κλειδώματος χαρακτηρισμού ασφάλειας περιλαμβάνει και τον αριθμό εγγραφής, η τιμή του είναι διαφορετική για γραμμές που περιέχουν τον ίδιο χαρακτηρισμό ασφάλειας.

Τεχνική 3: Κλείδωμα ακεραιότητας

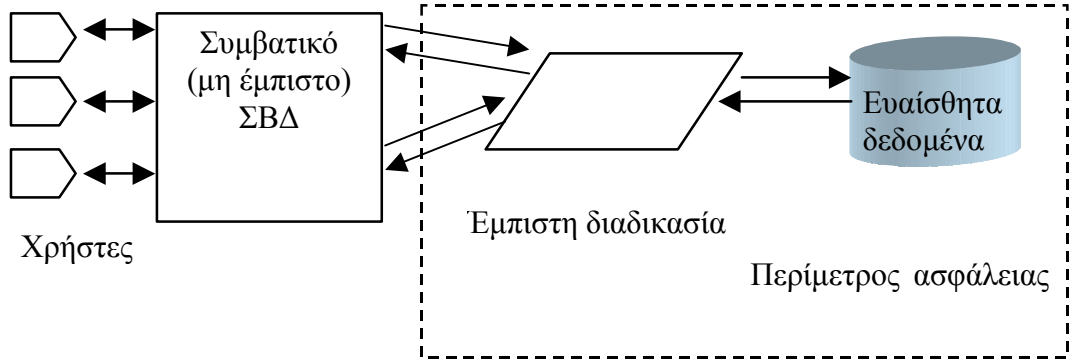
Το κλείδωμα ακεραιότητας είναι ένας συνδυασμός ενός μοναδικού προσδιοριστή, της τιμής του δεδομένου και του χαρακτηρισμού ευαισθησίας. Ο στόχος του είναι να προστατεύσει τα δεδομένα και τον χαρακτηρισμό ευαισθησίας από παράτυπες τροποποιήσεις. Ο υπολογισμός του κλειδώματος γίνεται από το ενδιάμεσο στρώμα λογισμικού κατά την αποθήκευση της εγγραφής, ενώ σε κάθε προσπέλαση μέσω του στρώματος ελέγχεται αν η τιμή του κλειδώματος είναι έγκυρη σε σχέση με τα δεδομένα, τον χαρακτηρισμό ευαισθησίας και την ταυτότητα εγγραφής. Η παραγωγή του κλειδώματος ακεραιότητας γίνεται όπως φαίνεται στο ακόλουθο σχήμα:



Το κλείδωμα ακεραιότητας που παράγεται από την κρυπτογραφική συνάρτηση αποθηκεύεται μαζί με την εγγραφή. Μη εξουσιοδοτημένες τροποποιήσεις στον χαρακτηρισμό ευαισθησίας ή στα δεδομένα θα γίνουν αντιληπτές, καθώς το κλείδωμα ελέγχεται σε κάθε πρόσβαση για το αν είναι έγκυρο, σε σχέση με τον αριθμό εγγραφής, τα δεδομένα και τον χαρακτηρισμό ευαισθησίας. Σημειώνεται ότι μια και ο υπολογισμός του κλειδώματος χαρακτηρισμού ευαισθησίας περιλαμβάνει και τον αριθμό εγγραφής, η τιμή του είναι διαφορετική για γραμμές που περιέχουν τα ίδια δεδομένα και τον ίδιο χαρακτηρισμό ευαισθησίας.

Προσέγγιση 1: Έμπιστη διαδικασία ελέγχου πρόσβασης

Η πρώτη προσέγγιση στην ενσωμάτωση πολυεπίπεδης ασφάλειας σε ένα συμβατικό ΣΔΒΔ είναι η προσθήκη μιας *έμπιστης διαδικασίας ελέγχου πρόσβασης* στο εσωτερικό του ΣΔΒΔ, πιθανότατα με τη μορφή αποθηκευμένων διαδικασιών, triggers και τύπων δεδομένων, όπως φαίνεται στο ακόλουθο σχήμα:



Η έμπιστη διαδικασία μεριμνά για την εφαρμογή των τεχνικών 1-3 που έχουν επιλεχθεί για τη συγκεκριμένη βάση (κρυπτογράφηση των δεδομένων, κρυπτογράφηση του χαρακτηρισμού ασφάλειας και τον υπολογισμό-αποθήκευση του κλειδώματος χαρακτηρισμού ασφάλειας, υπολογισμό-αποθήκευση του κλειδώματος ακεραιότητας, καθώς και για τους ελέγχους των κλειδωμάτων κατά τις προσπελάσεις).

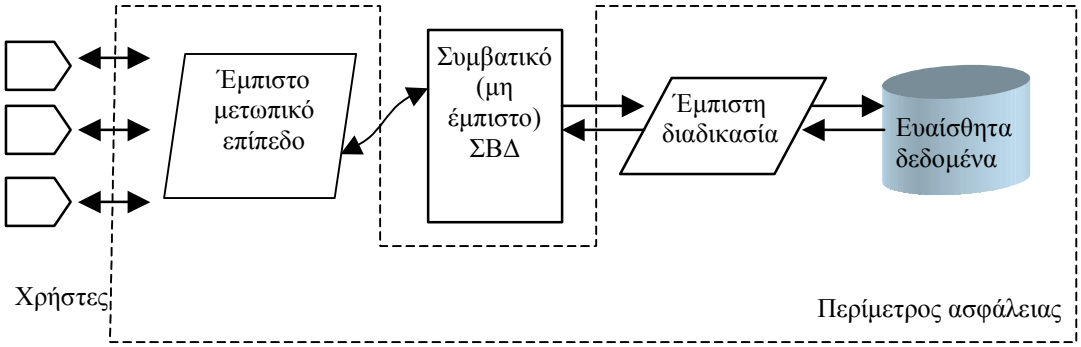
Η έμπιστη διαδικασία, λόγω του τρόπου υλοποίησής της, δεν μπορεί να εφαρμόσει καμία βελτιστοποίηση στην αποθήκευση των χαρακτηρισμών ασφάλειας, με αποτέλεσμα να απαιτείται αποθήκευση χαρακτηρισμού ασφάλειας ξεχωριστά για

κάθε δεδομένο, με συνέπεια να σπαταλάται πολύς χώρος. Επίσης, η διαδικασία κρυπτογράφησης-αποκρυπτογράφησης όπου απαιτείται (συμπεριλαμβανομένου του υπολογισμού και της εξέτασης των κλειδωμάτων) είναι ιδιαίτερα χρονοβόρα, καθώς πρέπει να εκτελείται σε κάθε πρόσβαση, είναι ωστόσο δυνατόν να την παραλείψουμε, αν κρίνουμε ότι το αρχείο όπου φυλάσσονται τα ευαίσθητα δεδομένα είναι επαρκώς προστατευμένο.

Το πιο σημαντικό μειονέκτημα της προσέγγισης αυτής ωστόσο είναι το γεγονός ότι όλοι οι έλεγχοι γίνονται με διαδικασίες μέσα στο συμβατικό ΣΔΒΔ. Αυτό έχει ως συνέπεια κάποιος χρήστης με επαρκή εξουσιοδότηση στο επίπεδο του ΣΔΒΔ (π.χ. ο διαχειριστής του) να μπορεί να εξάγει ή να παραποιήσει ευαίσθητα δεδομένα αναπαράγοντας τον τρόπο λειτουργίας της έμπιστης διαδικασίας.

Προσέγγιση 2: Έμπιστο μετωπικό επίπεδο πρόσβασης

Για να αντιμετωπισθούν τα μειονεκτήματα της προσέγγισης της έμπιστης διαδικασίας, είναι δυνατόν να επαυξήσουμε το προηγούμενο σχήμα με ένα ακόμη στρώμα λογισμικού που καλείται *έμπιστο μετωπικό επίπεδο* (trusted front-end), το οποίο τοποθετείται μεταξύ των χρηστών και του εμπορικού-μη έμπιστου ΣΔΒΔ όπως φαίνεται στο σχήμα που ακολουθεί:



Το έμπιστο μετωπικό επίπεδο παραλαμβάνει τα δεδομένα προς αποθήκευση *πριν* αυτά προωθηθούν στο συμβατικό ΣΔΒΔ, συνεπώς είναι κατά περίπτωση δυνατόν να εφαρμόσει βελτιστοποιήσεις σχετικά με την αποθήκευση του χαρακτηρισμού ασφάλειας ή να εκτελέσει κρυπτογραφήσεις και να διαχειριστεί κλειδώματα χαρακτηρισμού ασφάλειας και ακεραιότητας πριν τα δεδομένα φτάσουν στο ΣΔΒΔ, καθιστώντας έτσι ανέφικτη την ανάγνωση ή τροποποίηση δεδομένων στο επίπεδο του ΣΔΒΔ. Αν βέβαια εφαρμόζεται κρυπτογραφία *εκτός* του ΣΔΒΔ, τα κρυπτογραφημένα δεδομένα δεν μπορούν να τύχουν επεξεργασίας εντός του ΣΔΒΔ, π.χ. να χρησιμοποιηθούν σε μία συνθήκη ερώτησης.

Κατά την ανάκτηση των δεδομένων επίσης, το έμπιστο μετωπικό επίπεδο μπορεί να φιλτράρει τα δεδομένα που επιστρέφει το ΣΔΒΔ, αποκρύπτοντας αυτά που ο χρήστης δεν έχει το δικαίωμα να δει. Το μειονέκτημα ωστόσο αυτής της τεχνικής φιλτραρίσματος είναι ότι πολλά δεδομένα ανακτώνται από το ΣΔΒΔ για να απορριφθούν τελικά σε επόμενο στάδιο επεξεργασίας, κάτι που προφανώς συνιστά σπατάλη πόρων.

Πιο αναλυτικά, τα βήματα για την ανάκτηση δεδομένων με χρήση έμπιστου μετωπικού επιπέδου πρόσβασης έχουν ως ακολούθως:

1. η ταυτότητα του χρήστη διακριβώνεται από το έμπιστο μετωπικό επίπεδο πρόσβασης

2. ο χρήστης εισάγει μία ερώτηση στο μετωπικό επίπεδο
3. το μετωπικό επίπεδο επαληθεύει ότι ο χρήστης έχει δικαίωμα να προσπελάσει τα δεδομένα
4. το μετωπικό επίπεδο ερωτά το συμβατικό ΣΒΔ
5. το συμβατικό ΣΒΔ εκτελεί την είσοδο-έξοδο σε φυσικό επίπεδο και τις πράξεις της σχεσιακής άλγεβρας
6. το ΣΒΔ επιστρέφει τα αποτελέσματα στο μετωπικό επίπεδο
7. το μετωπικό επίπεδο αναλύει τα επίπεδα ασφάλειας των δεδομένων και επιλέγει εκείνα που αντιστοιχούν στο επίπεδο ασφάλειας του χρήστη
8. τα επιλεχθέντα δεδομένα προωθούνται στο συμβατικό ΣΒΔ για μορφοποίηση
9. τα μορφοποιημένα αποτελέσματα επιστρέφουν στον χρήστη

Προσέγγιση 3: Φίλτρα περιορισμού

Τα φίλτρα περιορισμού χρησιμοποιούνται σε συνδυασμό με την προσέγγιση του έμπιστου μετωπικού επιπέδου πρόσβασης. Στην προσέγγιση αυτή το μετωπικό επίπεδο, αφού παραλάβει την αίτηση του χρήστη, την τροποποιεί κατάλληλα ώστε να ανακτώνται *μόνον* τα δεδομένα τα οποία έχει δικαίωμα να προσπελάσει ο χρήστης. Η προσέγγιση αυτή επιτρέπει να απαντώνται πιο αποτελεσματικά οι ερωτήσεις, καθώς ανακτώνται λιγότερα δεδομένα (τα δεδομένα που δεν μπορεί να προσπελάσει ο χρήστες δεν ανακτώνται καθόλου), ενώ επίσης μεταφέρει και τον κύριο όγκο δουλειάς στο ΣΔΒΔ, απλοποιώντας έτσι την κατασκευή του έμπιστου μετωπικού επιπέδου. Σημειώνεται ωστόσο ότι είναι δυνατόν κάποιοι έλεγχοι να μην είναι δυνατόν να εκτελεστούν από το ΣΔΒΔ (π.χ. συνδυασμός στηλών για την εφαρμογή του κανόνα *ν αντικείμενα άνω του κ%*, στην οποία περίπτωση θα πρέπει οι κανόνες αυτοί να εφαρμοστούν από το έμπιστο μετωπικό επίπεδο. Οι διαδικασίες για την εφαρμογή των πρόσθετων κανόνων γίνονται σε δεύτερη φάση, αφού τα αποτελέσματα έχουν ανακτηθεί από το ΣΔΒΔ.

Τα φίλτρα περιορισμού μπορούν να εφαρμόζονται σε επίπεδο γραμμής, γνωρίσματος ή μεμονωμένου στοιχείου δεδομένων ως ακολούθως:

- σε επίπεδο γραμμής το φίλτρο ανακτά τα επιθυμητά δεδομένα συν τα κρυπτογραφικά αθροίσματα ελέγχου και επαληθεύει την ακρίβεια και την προσβασιμότητα από τον χρήστη των δεδομένων
- σε επίπεδο γνωρίσματος το φίλτρο
    - ελέγχει αν τα γνωρίσματα είναι προσβάσιμα στον χρήστη. Αν ναι, η ερώτηση προωθείται στο συμβατικό ΣΒΔ για επεξεργασία
    - στην επιστροφή το φίλτρο διαγράφει τα επί μέρους δεδομένα στα οποία δεν έχει πρόσβαση ο χρήστης
- Σε επίπεδο στοιχείου το σύστημα ανακτά τα δεδομένα συν τα κρυπτογραφικά αθροίσματα και συγκρίνει τον χαρακτηρισμό ασφάλειας κάθε στοιχείου με αυτόν του χρήστη, απαλείφοντας τα στοιχεία στα οποία ο χρήστης δεν έχει πρόσβαση.

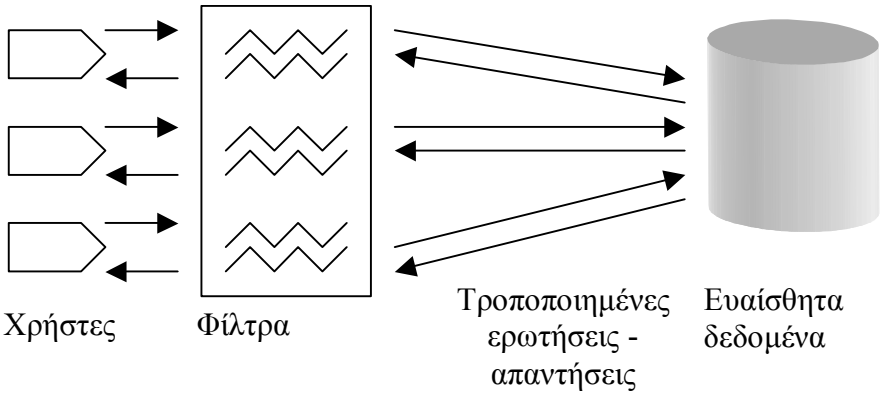
Ως παράδειγμα θεωρήστε την ερώτηση

```
select name
from doctors
where ((OCCUP = 'PHYSICIST') AND (CITY = 'LONDON'))
```

Ένα φίλτρο περιορισμού θα μπορούσε να μεταγράψει την ερώτηση αυτή ως ακολούθως:

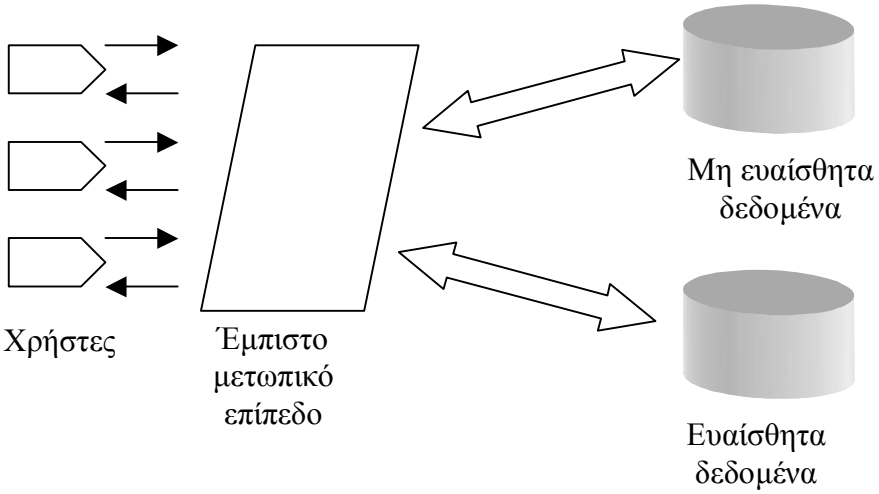
```
select name
from doctors
where ((occup = 'PHYSICIST') AND (city = 'LONDON')) AND
      (name_secrecy_level <= USER_SECRECY_LEVEL) AND
      (occup_secrecy_level <= USER_SECRECY_LEVEL) AND
      (city_secrecy_level <= USER_SECRECY_LEVEL)
```

με αποτέλεσμα το ΣΔΒΔ να περιορίζει την επεξεργασία μόνο στις γραμμές εκείνες για τις οποίες ο χρήστης έχει δικαίωμα πρόσβασης σε *όλα* τα γνωρίσματα που χρησιμοποιούνται στην ερώτηση. Η χρήση των φίλτρων περιορισμού απεικονίζεται στο ακόλουθο σχήμα.



Προσέγγιση 4: Ομόσπονδες βάσεις δεδομένων

Η προσέγγιση των ομόσπονδων βάσεων δεδομένων μοιάζει με αυτή του έμπιστου μετωπικού επιπέδου πρόσβασης, με τη διαφορά ότι δεν χρησιμοποιείται ένα ΣΔΒΔ αλλά δύο, ένα για την αποθήκευση των μη ευαίσθητων δεδομένων και ένα για την αποθήκευση των ευαίσθητων δεδομένων. Γενικεύοντας, μπορούν να χρησιμοποιηθούν *ν* ΣΔΒΔ, ένα για την αποθήκευση δεδομένων με συγκεκριμένο βαθμό ευαισθησίας. Το αρχιτεκτονικό σχήμα της προσέγγισης αυτής φαίνεται στην εικόνα που ακολουθεί.



Για να εξασφαλισθεί η ασφάλεια των ευαίσθητων δεδομένων, η προσέγγιση των ομόσπονδων βάσεων δεδομένων εφαρμόζει τους κάτωθι κανόνες λειτουργίας:

- Το μετωπικό επίπεδο παραλαμβάνει τις ερωτήσεις του χρήστη και ζητά δεδομένα από τα επί μέρους ΣΒΔ. Για χρήστες χωρίς επαρκή εξουσιοδότηση, τα δεδομένα ζητούνται μόνο από αυτό που περιέχει τα μη ευαίσθητα δεδομένα, ενώ για χρήστες με επαρκή εξουσιοδότηση και από τα δύο.
- Τα αποτελέσματα παραλαμβάνονται από το έμπιστο μετωπικό επίπεδο. Αν μόνο το ένα ΣΔΒΔ δώσει απάντηση, τότε τα δεδομένα προωθούνται άμεσα στον χρήστη. Αν ωστόσο παραληφθούν αποτελέσματα *και από τα δύο* ΣΔΒΔ, είναι πιθανόν να απαιτείται πρόσθετη επεξεργασία (π.χ. αν ο χρήστης θέλει να πραγματοποιήσει τη συσχέτιση *εργαζομένων-έργων* και τα στοιχεία κάποιου εργαζόμενου βρίσκονται στο ένα ΣΔΒΔ ενώ τα στοιχεία του έργου στο άλλο, ο συνδυασμός πρέπει να γίνει από το έμπιστο μετωπικό επίπεδο).
- Τα τελικά αποτελέσματα προωθούνται στον χρήστη.

Το βασικό μειονέκτημα της προσέγγισης αυτής είναι ότι το έμπιστο μετωπικό επίπεδο πρέπει να ενσωματώνει μεγάλο μέρος της λειτουργικότητας ενός πλήρους ΣΔΒΔ, καθιστώντας έτσι δυσχερή την υλοποίησή του. Επιπρόσθετα, για κάθε επίπεδο ασφάλειας απαιτείται και ξεχωριστό ΣΔΒΔ, κάτι που συνήθως συνεπάγεται υψηλό κόστος, τόσο σε άδειες λογισμικού όσο και σε μηχανήματα για την εκτέλεση των ξεχωριστών ΣΔΒΔ.

Προσέγγιση 5: Παράθυρα-όψεις

Τα ΣΔΒΔ, εκ κατασκευής τους, υποστηρίζουν την έννοια του *εξωτερικού σχήματος*, το οποίο είναι η εξειδικευμένη παρουσίαση των περιεχομένων της βάσης δεδομένων κατά τρόπο ώστε να ταιριάζει στις απαιτήσεις και περιορισμούς μιας συγκεκριμένης ομάδας χρηστών. Το χαρακτηριστικό αυτό αυτή μπορεί να αξιοποιηθεί στα πλαίσια της πολυεπίπεδης ασφάλειας δίνοντας σε κάθε χρήστη ακριβώς τα δεδομένα που έχει το δικαίωμα να προσπελαύνει. Βάσει του σχήματος αυτού:

- Στήλες αποκρύπτονται συνολικά, εκτός αν ο χρήστης έχει το δικαίωμα να προσπελάσει τουλάχιστον ένα στοιχείο του αποτελέσματος στη σχετική στήλη
- Γραμμές αποκρύπτονται συνολικά, εκτός αν ο χρήστης έχει το δικαίωμα να προσπελάσει τουλάχιστον ένα στοιχείο του αποτελέσματος στη σχετική γραμμή
- Για τα εναπομένοντα στοιχεία, αν ο χρήστης δεν έχει το δικαίωμα να προσπελάσει το συγκεκριμένο στοιχείο η τιμή του στοιχείου αποκρύπτεται αντικαθιστώμενη από την τιμή UNDEFINED.

Για παράδειγμα, για να εμφανίσουμε μόνο της γραμμές ενός πίνακα *Εργαζόμενος* με γνωρίσματα (Όνομα, ΑΣΦ_ΟΝΟΜΑΤΟΣ, Μισθός, ΑΣΦ_ΜΙΣΘΟΥ, Θέση, ΑΣΦ_ΘΕΣΗΣ), θα μπορούσαμε να ορίσουμε μια όψη ως ακολούθως:

```
CREATE VIEW Εργαζόμενος1 AS
SELECT Όνομα, Μισθός, Θέση
FROM Εργαζόμενος
WHERE ΑΣΦ_ΟΝΟΜΑΤΟΣ <= USER_SECRECY_LEVEL AND
      ΑΣΦ_ΜΙΣΘΟΥ <= USER_SECRECY_LEVEL AND
      ΑΣΦ_ΘΕΣΗΣ <= USER_SECRECY_LEVEL
```

Όταν οι χρήστες ανακτούν δεδομένα από την όψη *Εργαζόμενος1* αυτόματα εφαρμόζονται οι περιορισμοί που παρατίθενται στον ορισμό της όψης.

Προκειμένου να αποκρύψουμε και μεμονωμένα στοιχεία από τις γραμμές που επιλέγονται, θα χρειαστούμε μία συνάρτηση IF/THEN/ELSE που θα μας επιστρέφει την τιμή της $2^{ης}$ ή της $3^{ης}$ παραμέτρου της, ανάλογα με την τιμή αληθείας της $1^{ης}$. Με μία τέτοια συνάρτηση θα μπορούσαμε να ορίσουμε μία όψη πάνω στον πίνακα *Εργαζόμενος* ως εξής:

```
CREATE VIEW Εργαζόμενος2 AS
SELECT
      IFTHENELSE(ΑΣΦ_ΟΝΟΜΑΤΟΣ <= USL, Όνομα, NULL) AS Όνομα,
      IFTHENELSE(ΑΣΦ_ΜΙΣΘΟΥ <= USL, Μισθός, NULL) AS Μισθός,
      IFTHENELSE(ΑΣΦ_ΘΕΣΗΣ <= USL, Θέση, NULL) AS Θέση
FROM Εργαζόμενος
WHERE ΑΣΦ_ΟΝΟΜΑΤΟΣ <= USER_SECRECY_LEVEL AND
      ΑΣΦ_ΜΙΣΘΟΥ <= USER_SECRECY_LEVEL AND
      ΑΣΦ_ΘΕΣΗΣ <= USER_SECRECY_LEVEL
```

(η συντομογραφία `USL` αντικαθιστά το `USER_SECRECY_LEVEL` για λόγους οικονομίας χώρου). Με τον τρόπο αυτό, το ΣΔΒΔ θα αντικαταστήσει με τιμές NULL όλα τα μεμονωμένα στοιχεία για τα οποία ο χρήστης δεν έχει τη δυνατότητα πρόσβασης.

9 Ιοί

Οι ιοί αποτελούν πλέον μία από τις πιο σημαντικές απειλές για ασφάλεια των πληροφοριακών συστημάτων. Ως *ιός* γενικά χαρακτηρίζεται ένα πρόγραμμα το οποίο γράφτηκε ειδικά για να εισέρχεται σε συστήματα χωρίς να το ξέρει ή να το επιτρέψει ο ιδιοκτήτης-χρήστης. Από τη στιγμή που ο ιός καταφέρει να εισέλθει σε κάποιο σύστημα συνήθως προβαίνει σε ενέργειες που είναι ανεπιθύμητες για τον κάτοχο-χρήστη του συστήματος, όπως:

- *Αναπαραγωγή*. Ο ιός δημιουργεί αντίγραφα του εαυτού του, τα οποία ενδεχομένως θα μεταδοθούν και σε άλλα συστήματα.
- *Υποβάθμιση της ασφάλειας του συστήματος*. Ο ιός μπορεί να δίνει σε άλλους, μη εξουσιοδοτημένους χρήστες του διαδικτύου, τη δυνατότητα να συνδέονται στον υπολογιστή μας, να αντλούν δεδομένα κ.λπ.
- *Καταστροφή/φθορά δεδομένων*. Ο ιός μπορεί να διαγράφει αρχεία, να αλλοιώνει το περιεχόμενό τους ή ακόμη και να σβήνει ολόκληρους σκληρούς δίσκους.

Το μοντέλο αυτό συμπεριφοράς των ιών έχει περιγραφεί από τον Fred Cohen στην κλασική εργασία του "Computer Viruses - Theory and Experiments" που δημοσιεύθηκε το 1984. Στην εργασία αυτή το μοντέλο των ιών περιγράφεται με το τμήμα ψευδοκώδικα που ακολουθεί:

```
program virus:= {
1234567;
subroutine infect-executable := {
      loop: file = get-random-executable-file;
         if first-line-of-file = 1234567 then goto loop;
      prepend virus to file;
}
subroutine do-damage := {
   whatever damage is to be done;
}
subroutine trigger-pulled := {
   return true if some condition holds
}
main-program :=  {
   infect-executable;
   if trigger-pulled then do-damage;
   goto next;
}
next:
}
```

Στο τμήμα αυτό διακρίνουμε αρχικά ότι ο ιός περιέχει έναν χαρακτηριστικό αριθμό στην αρχή του. Ο αριθμός αυτός είναι χρήσιμος προκειμένου ο ιός να αναγνωρίζει την παρουσία του σε κάποιο εκτελέσιμο και να μην το μολύνει επισυνάπτοντας σ' αυτό τον εαυτό του πάνω από μία φορές. Τον έλεγχο αυτό μπορούμε να τον δούμε στη διαδικασία *μόλυνση εκτελέσιμου* (infect-executable) όπου πράγματι ελέγχεται αν το υποψήφιο προς μόλυνση πρόγραμμα ξεκινά με την «υπογραφή» του ιού, και μολύνεται τότε και μόνον όταν βρεθεί «καθαρό», ειδάλλως αναζητάται κάποιο άλλο «θύμα».

9.1 Οι φάσεις ενός ιού

Οι υπόλοιπες γραμμές του μοντέλου υλοποιούν τις δύο κύριες φάσεις ενός ιού, τη *φάση μόλυνσης* και τη *φάση επίθεσης*. Στη φάση μόλυνσης, ο ιός αναζητά νέα αντικείμενα υποψήφια προς μόλυνση, και επισυνάπτει τον εαυτό του σ' αυτά. Οι συγγραφείς ιών πολλές φορές αντισταθμίζουν την αμεσότητα και αποτελεσματικότητα της μόλυνσης με την ευκολία αποκάλυψης του ιού: ένας ιός που μολύνει αμέσως όλους τους πιθανούς «στόχους» του είναι εύκολο να αποκαλυφθεί διότι η εκτέλεση του ιού συνοδεύεται πάντα με κάποια ασυνήθιστη συμπεριφορά του συστήματος. Αντιθέτως, ένας ιός που αποφασίζει αν θα μολύνει ή όχι κάποιον στόχο εξετάζοντας κάποια συνθήκη (όπως την ημερομηνία, το πλήθος εκτελέσεών του, κάποια εξωτερικά συμβάντα κ.τ.λ.) είναι πιο δύσκολο να εντοπισθεί διότι η ασυνήθιστη συμπεριφορά εμφανίζεται πιο σπάνια και μπορεί να περάσει απαρατήρητη από τους χρήστες.

Πολλοί από τους ιούς χρησιμοποιούν για τη φάση μόλυνσης την τεχνική της δημιουργίας *παραμενόντων προγραμμάτων*: ο ιός φορτώνεται στη μνήμη με την εκτέλεση ενός μολυσμένου αντικειμένου (του *ξενιστή*) αλλά από το σημείο αυτό και μετά φροντίζει να απεξαρτηθεί από τον φορέα του και να παραμείνει ως αυτόνομη οντότητα στη μνήμη, ακόμη και μετά τον τερματισμό της εκτέλεσης του προγράμματος που προκάλεσε τη φόρτωσή του. Έχοντας εγκατασταθεί στη μνήμη ουσιαστικά περιμένει για τη στιγμή που θα θεωρηθεί κατάλληλη για να επιχειρήσει τη μόλυνση ενός νέου θύματος. Ιοί που χρησιμοποιούν την τεχνολογία των παραμενόντων προγραμμάτων φροντίζουν πολλές φορές να κρατούν την παρουσία τους κρυφή από τους χρήστες ή από ειδικά προγράμματα που έχουν ως στόχο να τους

ανιχνεύσουν. Αυτό επιτυγχάνεται είτε με *τεχνικές απόκρυψης* (stealth techniques) ή με *τεχνικές πολυμορφισμού*. Αντίθετα προς τις ανωτέρω προσεγγίσεις που ακολουθούν προσεκτικά βήματα κατά την εξάπλωσή των ιών, η κατηγορία των ιών που είναι γνωστή με το όνομα *«σκουλήκια»* προσπαθεί να διαδοθεί άμεσα και σε όσο το δυνατόν μεγαλύτερη έκταση.

Η φάση της επίθεσης, η οποία ακολουθεί τη φάση μόλυνσης, είναι προαιρετική για έναν ιό, μπορεί δηλαδή κάποιος ιός να μην περιέχει καθόλου τη φάση επίθεσης. Σε κάθε περίπτωση βέβαια, ένας ιός που έχει εισέλθει στο σύστημα σίγουρα έχει αρνητικά αποτελέσματα από την άποψη ότι καταναλώνει πόρους του συστήματος, κατ' ελάχιστον τον επιπλέον χώρο στα αρχεία που έχει μολύνει και τη μνήμη που δεσμεύει προκειμένου να εκτελέσει τον κώδικά του. Για τους ιούς που όντως περιλαμβάνουν τη φάση της επίθεσης, οι πιο συχνές ενέργειες που απαντώνται στη φάση αυτή είναι οι εξής:

- Διαγραφή ή παραφθορά αρχείων.
- Αναπαραγωγή μουσικής ή μηνύματα στην οθόνη.
- Αποστολή μηνυμάτων ηλεκτρονικού ταχυδρομείου.

Όπως αντιλαμβανόμαστε, η έκταση της ζημιάς που θα προκληθεί μπορεί να ποικίλει από απλή ενόχληση του χρήστη έως την απώλεια πολύτιμων ή/και αναντικατάστατων δεδομένων. Σημειώνεται δε ότι οι καταστροφές που τυχόν θα προκληθούν μπορεί να είναι προτιθέμενες από τον συγγραφέα του ιού, αλλά μπορεί να οφείλονται και σε προγραμματιστική αβλεψία: για παράδειγμα, ο ιός *stoned* στην προσπάθειά του να αποκρύψει την παρουσία του χρησιμοποιούσε μία τεχνική που λειτουργούσε θαυμάσια σε δισκέτες χωρητικότητας 360K, κατέστρεφε όμως τις δισκέτες χωρητικότητας 1.2M. Ευτυχώς, το πρόβλημα αντιμετωπίσθηκε σε επόμενη έκδοση του ιού.

9.2 Υπάρχουν καλοί ιοί;

Ένα φιλοσοφικό θέμα που έχει τεθεί σχετικά με τους ιούς εισάγει τον προβληματισμό του αν κάποιος πρόγραμμα που έχει όλα τα χαρακτηριστικά του ιού μπορεί να χρησιμοποιηθεί για καλό σκοπό, περιλαμβάνοντας στο κομμάτι που ονομάζεται *πρόκληση ζημιάς* κώδικα ο οποίος θα απέβαινε έπ' ωφελεία του χρήστη. Ως συγκεκριμένα παραδείγματα «καλών ιών» έχουν προβληθεί τα κάτωθι:

- Ο «αντιβιοτικός» ιός, ο οποίος εντοπίζει και «σκοτώνει» τους κακούς ιούς.
- Ο ιός «συμπιεστής», που εντοπίζει τα αρχεία με σπάνια χρήση και τα συμπιέζει, επισυνάπτοντας στο συμπιεσμένο αρχείο και μία παραλλαγή του εαυτού του που στοχεύει στην αποσυμπίεση του αρχείου.
- Ο ιός «κρυπτογράφος», ο οποίος εγκαθίσταται στο σκληρό δίσκο και τον κρυπτογραφεί βάσει ενός συνθηματικού που δίνει ο χρήστης, προκειμένου να κρατήσει τα περιεχόμενά του αθέατα για τα αδιάκριτα βλέμματα.
- Ο ιός «συντηρητής» που πραγματοποιεί κάποιες λειτουργίες συντήρησης π.χ. διαγραφή προσωρινών αρχείων.

Σε κάθε περίπτωση, η τοποθέτησή μας πάνω στο ζήτημα ύπαρξης «καλών ιών» πρέπει να είναι κατηγορηματικά αρνητική. Σ' αυτή την κατεύθυνση συντρέχουν τόσο τεχνικοί λόγοι, όσο και ηθικά, νομικά και ψυχολογικά ζητήματα.

1) Τεχνικοί λόγοι

i) *Αδυναμία ελέγχου*. Οι ιοί εκ κατασκευής δεν έχουν δικλείδες ελέγχου: η δράση τους υπαγορεύεται από το σχήμα «μόλυνση-επίθεση» (εδώ η επίθεση θα πρέπει να είναι μια δέσμη θετικών ενεργειών) και δεν υπάρχει άμεσος τρόπος να υπαγορεύσουμε σε κάποιον ιό να κάνει ή να μην κάνει κάποια πράγματα, μέσω π.χ. ενός συνόλου ρυθμίσεων, όπως θα συνέβαινε με ένα «κανονικό» πρόγραμμα.

ii) *Δυσκολία διάκρισης*. Προκειμένου να αντιμετωπισθεί ένας «κακός» ιός που έχει εισέλθει σε ένα σύστημα πρέπει πρώτα να εντοπισθεί η ύπαρξή του, είτε από τον χρήστη είτε από κάποιο αυτοματοποιημένο σύστημα. Ο εντοπισμός του ιού, η διάκρισή του δηλαδή από τα «κανονικά» προγράμματα που υπάρχουν στον υπολογιστή είναι μία ήδη πολύ δύσκολη υπόθεση, η οποία θα δυσχερανθεί σε πολύ μεγάλο βαθμό αν εισάγουμε και μία διάκριση μεταξύ «καλών» και «κακών» ιών.

iii) *Σπατάλη πόρων*. Αν χρησιμοποιούμε κάποιο «κανονικό» πρόγραμμα για ένα συγκεκριμένο σκοπό, προφανώς θα έχουμε μόνο ένα αντίγραφό του. Αντίθετα αν σκοποί του συστήματός μας εξυπηρετούνται από ιούς, το πλήθος των αντιγράφων θα ισούται με το πλήθος των αρχείων που έχει μολύνει ο ιός. Επίσης, τίποτε δεν αποκλείει την πιθανότητα να εισέλθουν στο σύστημα πάνω από ένα είδη ιών που ασχολούνται με το ίδιο ζήτημα (π.χ. διαγραφή προσωρινών αρχείων), μεγαλώνοντας έτσι ακόμη περισσότερο τη σπατάλη πόρων.

iv) *Πιθανότητα ύπαρξης σφαλμάτων*. Κανείς δεν μας εγγυάται την ορθότητα των αλγορίθμων ή της υλοποίησης των ιών, ενώ δεν μπορεί να αναμένει κανείς ότι όλοι οι συγγραφείς τους θα επιδείξουν την ίδια υπευθυνότητα που επιδεικνύουν οι οίκοι λογισμικού.

v) *Ζητήματα συμβατότητας*. Υπάρχει μία κατηγορία προγραμμάτων που περιέχουν κώδικα αυτοεπαλήθευσης, είτε για λόγους προστασίας από αντιγραφή είτε για λόγους εγγυημένης καλής λειτουργίας. Αν ένας «καλός ιός» μολύνει ένα τέτοιο αρχείο θα προκαλέσει μοιραία και την διακοπή της ομαλής λειτουργίας του, καθιστώντας το άχρηστο.

2) Ηθικά, νομικά και ψυχολογικά ζητήματα

i) *Μη εξουσιοδοτημένη τροποποίηση δεδομένων*. Ο «καλός ιός» εισέρχεται στο σύστημα χωρίς τη γνώση και τη ρητή συγκατάθεση του χρήστη και ξεκινά να τροποποιεί προγράμματα και δεδομένα χωρίς και πάλι ο χρήστης (ή ο ιδιοκτήτης) να γνωρίζει κάτι σχετικά. Αυτό αντιβαίνει, εκτός των άλλων, σε έναν από τους βασικούς κανόνες της ασφάλειας όπου επιθυμούμε όλες οι ενέργειες να γίνονται κατόπιν κατάλληλης εξουσιοδότησης.

ii) *Ζητήματα ιδιοκτησίας και πνευματικής ιδιοκτησίας*. Οι κατασκευαστές των «κανονικών προγραμμάτων» μπορεί να εγείρουν τέτοιου είδους ζητήματα, καθώς η άδεια χρήσης ενός προγράμματος συνήθως δεν περιλαμβάνει την παροχή δυνατότητας στον χρήστη να τροποποιήσει το πρόγραμμα καθ' οιονδήποτε τρόπο.

iii) *Πιθανή κακή χρήση*. Καμία εγγύηση δεν παρέχεται από τους –πιθανότατα άγνωστους συνολικά– κατασκευαστές «καλών ιών» ότι θα έχουν μόνο θετικές επιπτώσεις για το σύστημα και δεν θα προβαίνουν και σε ύποπτες ή επιζήμιες ενέργειες.

iv) *Υπευθυνότητα και καταλογισμός ευθυνών*. Δεν υπάρχει η δυνατότητα καταλογισμού ευθυνών για αρνητικές συνέπειες που θα έχει ένας «καλός ιός», καθώς ο κατασκευαστής του είναι συνήθως άγνωστος. Ακόμη και αν τυχόν προβλήματα προέρχονται από προγραμματιστικά σφάλματα παρά από κακή πρόθεση, δεν είναι βέβαιο ότι οι κατασκευαστές των ιών θα φροντίσουν να τα διορθώσουν και αν ακόμη το κάνουν, δεν είναι σαφές πώς οι επιδιορθωμένες εκδόσεις θα φτάσουν στους υπολογιστές των τελικών χρηστών, μια και οι ίδιοι οι χρήστες δεν γνωρίζουν ότι ο ιός είναι εγκατεστημένος στο σύστημά τους, συνεπώς δεν θα σπεύσουν να προμηθευτούν την τελευταία έκδοση.

v) Αντίληψη του όρου «ιός». Τόσο από τον χώρο της βιολογίας και της ιατρικής, όσο και από τον χώρο της πληροφορικής, ο όρος «ιός» είναι αρνητικά φορτισμένος και έτσι ελάχιστοι χρήστες θα δεχόταν θετικά την ύπαρξη ενός τέτοιου λογισμικού στον υπολογιστή τους.

vi) Ζητήματα εμπιστοσύνης και αισθήματος ασφάλειας. Οι χρήστες αισθάνονται καλύτερα έχοντας την ψευδαίσθηση ότι γνωρίζουν τι συμβαίνει στον υπολογιστή τους, ή τουλάχιστον γνωρίζοντας ότι έχουν επιλέξει το λογισμικό που χρησιμοποιούν. Η πιθανότητα για ανεξέλεγκτη είσοδο και λειτουργία λογισμικού στον υπολογιστή ελαττώνει την ασφάλεια που νοιώθει ο μέσος χρήστης και τον κάνει να μην εμπιστεύεται ιδιαίτερα τη χρήση του υπολογιστή.

9.3 Ταξινόμηση των ιών

Οι ιοί μπορούν να διαχωριστούν σε κατηγορίες βάσει δύο χαρακτηριστικών τους, του *τι μολύνουν*, δηλαδή ποια είναι τα αντικείμενα του συστήματος τα οποία αλλοιώνουν και του *πώς μολύνουν*, δηλαδή τον τρόπο που χρησιμοποιούν για να επιτύχουν τη μόλυνση. Στις επόμενες παραγράφους αναλύονται οι διάφορες κατηγορίες ιών, ανάλογα με τα χαρακτηριστικά αυτά.

9.3.1 Ιοί που μολύνουν τους τομείς εκκίνησης

Σε κάθε σκληρό δίσκο ή δισκέτα, ακόμη και αν δεν περιέχει λειτουργικό σύστημα, υπάρχει ένας τομέας που ονομάζεται *τομέας εκκίνησης* (boot sector). Ο τομέας αυτός ονομάζεται έτσι διότι το βασικό σύστημα εισόδου-εξόδου του υπολογιστή (BIOS) τον διαβάζει κατά την εκκίνηση του υπολογιστή και εκτελεί τον κώδικα που περιέχεται σ' αυτόν, προκειμένου να φορτωθεί το λειτουργικό σύστημα (αν είναι εγκατεστημένο) ή να εκτυπωθεί κάποιο διαγνωστικό μήνυμα λάθους, αν δεν έχει εγκατασταθεί το λειτουργικό σύστημα. Στους σκληρούς δίσκους υπάρχει ένας ακόμη τομέας συστήματος, ο *κύριος τομέας εκκίνησης*, ο οποίος μεταξύ άλλων περιέχει τον πίνακα διαμερίσεων (partition table) και κώδικα που φροντίζει να διαβάσει και να εκτελέσει τον τομέα εκκίνησης κάποιας συγκεκριμένης διαμέρισης.

Οι ιοί που επιτίθενται στους τομείς εκκίνησης φροντίζουν να εγκατασταθούν σε έναν από αυτούς, μετατοπίζοντας τον κανονικό κώδικα σε κάποιο άλλο σημείο του δίσκου. Για αποφυγή επικάλυψης της περιοχής που περιέχει τα κανονικά δεδομένα εκκίνησης με δεδομένα (κατά την εκχώρηση της περιοχής σε ένα αρχείο) οι ιοί συνήθως σημαδεύουν τη συγκεκριμένη περιοχή ως «κατεστραμμένη», με αποτέλεσμα το λειτουργικό σύστημα να μην τη χρησιμοποιεί συνολικά. Όταν τώρα ο υπολογιστής εκκινηθεί θα διαβάσει τον μολυσμένο τομέα εκκίνησης και θα ξεκινήσει την εκτέλεση του ιού. Ο ιός θα εγκατασταθεί στη μνήμη, και κατόπιν θα συνεχίσει με το φόρτωμα του λειτουργικού συστήματος, χρησιμοποιώντας τον κώδικα που έχει

αποθηκεύσει. Όντας στη μνήμη ο ιός θα φροντίζει να μολύνει όλους τους διαθέσιμους δίσκους ή τις δισκέτες που τυχόν θα προσπελασθούν.

9.3.2 Ιοί αρχείων

Οι ιοί αρχείων αποτελούν την πολυπληθέστερη αριθμητικά ποικιλία, αν και η δυσκολία στη συγγραφή τους έχει αποθαρρύνει πλέον τους συγγραφείς ιών από το να αυξάνουν τον πληθάριθμο αυτής της κατηγορίας. Οι ιοί αυτοί φροντίζουν να τροποποιούν τα εκτελέσιμα προγράμματα, όπως π.χ. .COM και .EXE σε περιβάλλον προσωπικών υπολογιστών. Στην απλούστερη μορφή τους απλά επικαλύπτουν το αρχικό τμήμα του προγράμματος με τον δικό τους κώδικα, με επακόλουθο η εκτέλεση του προγράμματος να μην έχει τη συνήθη για τους χρήστες λειτουργικότητα, αφού απλώς προκαλεί την εκτέλεση του ιού. Αυτό γίνεται άμεσα αντιληπτό από τους χρήστες με συνέπεια ο εντοπισμός του ιού να είναι τάχιστος, μειώνοντας τις δυνατότητές του για εξάπλωση.

Στην πιο εξελιγμένη μορφή τους, οι ιοί αρχείων μετατοπίζουν τον αρχικό κώδικα του αρχείου ή επισυνάπτουν τον δικό τους κώδικα στο τέλος, με τις κατάλληλες εντολές σύνδεσης. Με τον τρόπο αυτό αφ' ενός εκτελείται ο κώδικας του ιού, αφ΄ ετέρου εκτελείται και ο κώδικας του προγράμματος, με αποτέλεσμα οι χρήστες να μην αντιλαμβάνονται εύκολα την ύπαρξη του ιού, και έτσι αυτός να έχει μεγαλύτερο χρόνο δράσης. Ένα παράδειγμα ιού αρχείων που μολύνει το COMMAND.COM φαίνεται στο σχήμα που ακολουθεί. Στο αριστερό τμήμα του σχήματος διακρίνεται το αρχείο COMMAND.COM προ της μολύνσεώς του ενώ στα δεξιά διακρίνεται το ίδιο αρχείο μετά τη μόλυνσή του.

```
0100 06       PUSH    ES
0101 17       POP     SS
0102 BE1B02   MOV     SI,021B
```

```
0100 E9C1CB  JMP     52420
0103 DW      021Bh
...
CCC4 ; Κώδικας ιού
...
CDE2 B80617  MOV   AX, 01706h
CDE5 A30001  MOV   [0100h], AX
CDE8 B8BE1B  MOV   AX, 1BBEh
CDEB A30201  MOV   [0102h], AX
CDEE E90F33  JMP   100
```

Παρατηρούμε ότι κατά τη μόλυνση αντικαθίστανται τα τρία πρώτα bytes του αρχείου με μία εντολή άλματος, η οποία μεταβιβάζει τον έλεγχο στον κώδικα του ιού, ο οποίος έχει προσκολληθεί στο τέλος του αρχείου. Όταν τελειώσει ο καθ' εαυτός κώδικας του ιού, ακολουθεί ένα σύνολο εντολών που έχει ως στόχο να αποκαταστήσει τα πρώτα bytes του αρχείου στις αρχικές τους τιμές και να φροντίσει για την κανονική πλέον εκτέλεση του προγράμματος.

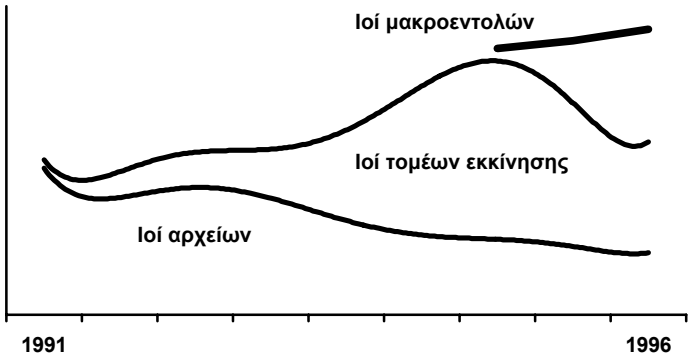
Σημειώνεται ότι οι ιοί αρχείων δεν είναι απαραίτητο να μολύνουν μόνο τα «άμεσα» εκτελέσιμα αρχεία, όπως τα .COM και .EXE: αρχεία επικάλυψης (OVL), οδηγοί συσκευών (DRV, SYS), δυαδικά εκτελέσιμα (BIN) ή δυναμικά συνδεόμενες βιβλιοθήκες (DLL) έχουν κατά καιρούς αποτελέσει στόχους ιών. Οι ιοί αρχείων μπορεί να τερματίζουν την εκτέλεσή τους μαζί με το πέρας εκτέλεσης του αρχείου στο οποίο έχουν προσκολληθεί, μπορεί όμως να παραμένουν στη μνήμη και μετά το πέρας της εκτέλεσης του ξενιστή τους.

9.3.3 Ιοί μακροεντολών

Οι ιοί μακροεντολών είναι μία ειδική κατηγορία ιών που μολύνει αρχεία δεδομένων. Τα αρχεία δεδομένων κανονικά δεν είναι ενεργές οντότητες, κατά συνέπεια δεν είναι άμεσα υποψήφια για το ρόλο του ξενιστή ιών. Πολλές εφαρμογές ωστόσο, προκειμένου να δώσουν στα αρχεία τους δυνατότητες αυτοματισμού ή εν γένει ενεργά χαρακτηριστικά ενσωματώνουν διερμηνευτές γλωσσών μακροεντολών, οι οποίοι εκτελούν ειδικές κατηγορίες προγραμμάτων που είναι ενσωματωμένες στα αρχεία. Επί παραδείγματι, η οικογένεια εφαρμογών Microsoft Office ενσωματώνει ως γλώσσα μακροεντολών τη Visual Basic for Applications, η οποία μπορεί να προσδώσει πολλά ενεργά χαρακτηριστικά στα έγγραφα. Κατ΄ αυτή την έννοια τα αρχεία δεδομένων είναι πλέον άριστοι υποψήφιοι για την μόλυνση από ιούς. Επίσης, τα συνημμένα έγγραφα σε μηνύματα ηλεκτρονικού ταχυδρομείου μπορεί να περιέχουν και αυτά ιούς, είτε με στόχο να εκτελεστούν από την εφαρμογή που θα ανοίξει το συνημμένο έγγραφο είτε και από την ίδια την εφαρμογή διαχείρισης ηλεκτρονικής αλληλογραφίας.

Σημειώνουμε εδώ ότι όσο πιο ισχυρή είναι η γλώσσα μακροεντολών, τόσο πιο καταστροφικός μπορεί να αποβεί ένας ιός που την αξιοποιεί.

Στο διάγραμμα που ακολουθεί παρουσιάζεται η χρονική εξέλιξη των τριών κατηγοριών ιών που αναφέρθηκαν πιο πάνω και φαίνεται η άμεση αποδοχή που γνώρισε η τεχνολογία των μακροεντολών από τους συγγραφείς ιών.



9.3.4 Ιοί συστοιχίας

Οι ιοί συστοιχίας έχουν έναν ιδιόμορφο τρόπο προσκόλλησης στα αρχεία, καθώς δεν τροποποιούν καθ' οιονδήποτε τρόπο το ίδιο το αρχείο *αλλά τις πληροφορίες καταλόγου*. Ο ιός εγκαθιστά ένα αντίγραφο του εαυτού του σε κάποιο μπλοκ του δίσκου και, για κάθε πρόγραμμα που μολύνει, τροποποιεί τις πληροφορίες καταλόγου ώστε να δείχνουν πως το αρχείο ξεκινά από το μπλοκ που περιέχει τον ιό. Ο ιός φροντίζει να διατηρεί βέβαια και τις σωστές πληροφορίες για την κατανομή των αρχείων στο δίσκο και ένα κομμάτι του που παραμένει στη μνήμη φροντίζει έτσι ώστε τα μολυσμένα αρχεία να διαβάζονται «σωστά», δηλαδή πρώτα να διαβάζεται ο κώδικας του ιού και κατόπιν τα κανονικά δεδομένα. Το ίδιο κομμάτι του ιού φροντίζει ώστε οι τυχόν εγγραφές στο αρχείο να μην επικαλύπτουν το τμήμα του δίσκου που περιέχει τον ιό αλλά να κατευθύνονται στον πραγματικό χώρο δεδομένων του αρχείου.

Μία «ευχάριστη» παρενέργεια της τεχνικής αυτής μόλυνσης είναι ότι δημιουργείται ένα μόνο αντίγραφο του ιού στον δίσκο, μειώνοντας έτσι δραστικά τη σπατάλη χώρου από τα πολλαπλά αντίγραφα του ιού. Από την άλλη πλευρά όμως, αν συμβεί

να ξεκινήσει ο υπολογιστής με «καθαρό» σύστημα –με αποτέλεσμα να μην βρίσκεται στη μνήμη το κομμάτι του ιού που διευθετεί τις εγγραφές και τις αναγνώσεις– και εκτελεσθεί ένα πρόγραμμα ελέγχου της ακεραιότητας του δίσκου, θα αναφέρει ότι τα μολυσμένα αρχεία είναι προβληματικά, καθώς θα δείχνουν να ξεκινούν όλα από τον ίδιο τομέα του δίσκου, και κάτι τέτοιο είναι κανονικά αδύνατον. Αν μάλιστα επιλέξουμε να επιτρέψουμε στο πρόγραμμα ελέγχου να επιχειρήσει να διορθώσει τη δομή του συστήματος αρχείων, η ζημιά που θα προξενηθεί θα είναι πιθανότατα πέραν οποιασδήποτε δυνατότητας επανόρθωσης.

9.3.5 Ιοί συνοδείας

Οι ιοί συνοδείας ήταν ιδιαίτερα δημοφιλής στο περιβάλλον DOS καθώς δεν τροποποιούσαν τα αρχεία που «μόλυναν» αλλά φρόντιζαν να εκτελεσθούν πριν από αυτά. Η απουσία της αναγκαιότητας για τροποποίηση των αρχείων απλοποιούσε ιδιαίτερα τη συγγραφή τους και έτσι ήταν πολύ αγαπητοί στην κοινότητα των συγγραφέων ιών. Για να επιτύχουν την εκτέλεση πριν το «κανονικό» πρόγραμμα αξιοποιούσαν το χαρακτηριστικό του DOS σύμφωνα με το οποίο αν ο χρήστης έδινε την εντολή `X`, το DOS έψαχνε πρώτα για το αρχείο `X.COM` και κατόπιν για το `X.EXE`. Έτσι, προκειμένου να «μολυνθεί» το αρχείο `X.EXE` αρκούσε να εγκατασταθεί στον ίδιο κατάλογο το αρχείο `X.COM`. Όταν ο χρήστης δώσει την εντολή «`X`» εκτελείται το `X.COM`, δηλαδή ο ιός, ο οποίος στη συνέχεια φροντίζει για την εκτέλεση του `X.EXE`. Οι ιοί συνοδείας είναι εύκολο να εντοπισθούν και να καθαρισθούν, καθώς ελάχιστα προγράμματα έχουν «νομότυπα» στον ίδιο κατάλογο και τους δύο τύπους εκτελέσιμων αρχείων (το μοναδικό ίσως είναι το DOSSHELL).

9.3.6 Ιοί ειδικά για Windows

Χωρίς να προξενείται ιδιαίτερη έκπληξη σε κανέναν, το λειτουργικό σύστημα Windows έδωσε νέες θαυμάσιες ευκαιρίες στους συγγραφείς ιών για να αναπτύξουν το ταλέντο τους. Οι ιοί μπορούν πλέον να εκτελούνται ως διεργασίες εξυπηρέτησης (services) ή προγράμματα οδήγησης συσκευών, ή ακόμη και ως «κρυφές» εφαρμογές, εξασφαλίζοντας έτσι συνεχή παρουσία στη μνήμη και μικρές πιθανότητες ανίχνευσης από τους χρήστες. Οι ιοί έχουν επίσης αξιοποιήσει το χαρακτηριστικό του μητρώου (registry) των Windows που συσχετίζει τύπους αρχείων με τρόπους εκτέλεσης. Έτσι, για τον τύπο αρχείων EXEFILE (που αντιστοιχεί σε όλα τα εκτελέσιμα αρχεία με επίθεμα .EXE) μπορεί να ορισθεί ένας νέος τρόπος εκτέλεσης μέσω του κλειδιού του μητρώου

```
HKEY_CLASSES_ROOT\exefile\shell\open\command=virus.exe "%1" "%*"
```

το οποίο ορίζει ότι «για να εκτελεσθεί *οποιοδήποτε* πρόγραμμα τύπου EXE θα κληθεί το πρόγραμμα `virus.exe` με παραμέτρους το όνομα του καλούμενου προγράμματος και τα τυχόν ορίσματα που έχουν αρχικά χρησιμοποιηθεί. Ο ιός (`virus.exe`) αφού εκτελέσει τις ενέργειες που επιθυμεί μπορεί να προβεί στην εκτέλεση του προγράμματος που αρχικά κλήθηκε, καθ' όσον έχει στη διάθεσή του όλα τα στοιχεία για την εκτέλεσή του.

Ένας ακόμη τρόπος για διάδοση ιών συνίσταται στην αξιοποίηση της *εναλλακτικής ροής δεδομένων αρχείου* που υποστηρίζεται από το σύστημα αρχείων NTFS. Η εναλλακτική ροή δεδομένων επιτρέπει για κάθε αρχείο Α να ορίζεται απεριόριστο πλήθος εναλλακτικών ροών που σχετίζονται μ' αυτό και σημειώνονται ως A:ext1, A:ext2, …, A:extn, όπου ext1, ext2 κ.λπ. είναι αυθαίρετες συμβολοσειρές. Τα δεδομένα που περιέχει κάθε εναλλακτική ροή είναι απολύτως ανεξάρτητα από αυτά

των άλλων ροών, με μοναδικό συνδετικό κρίκο ότι αν σβησθεί το αρχείο Α αυτόματα σβήνονται όλες οι συνδεδεμένες μ' αυτό ροές.

Ο τρόπος που μπορούν να αξιοποιηθούν οι εναλλακτικές ροές για τη διάδοση των ιών είναι ο εξής:

1. αρχικά το εκτελέσιμο αρχείο A.EXE αντιγράφεται στο A.EXE:ORIG, δηλαδή σε μία εναλλακτική ροή δεδομένων.
2. Τα περιεχόμενα του A.EXE τροποποιούνται έτσι ώστε το αρχείο A.EXE να περιέχει πλέον τον ιό, καθώς και ένα τμήμα κώδικα που θα φροντίζει για την εκτέλεση του A.EXE:ORIG.

Η αποκάλυψη των ιών αυτών είναι σχετικά δύσκολη καθώς οι εναλλακτικές ροές δεδομένων δεν εμφανίζονται στους καταλόγους, ενώ ακόμη και μερικά αυτοματοποιημένα προγράμματα δεν ενσωματώνουν κώδικα που να μπορεί να εντοπίσει τις εναλλακτικές ροές δεδομένων.

9.3.7 Προγράμματα εναπόθεσης ιών

Τα προγράμματα εναπόθεσης ιών δεν είναι ιοί από μόνα τους, έχουν όμως το χαρακτηριστικό ότι εκτελούμενα σε κάποιο σύστημα μολύνουν αντικείμενα του συστήματος με κάποιον ιό. Στην κατηγορία αυτή συγκαταλέγονται οι *δούρειοι ίπποι* που μεταφέρουν ιούς.

9.3.8 Πολυμορφικοί ιοί

Οι πολυμορφικοί ιοί είναι μία ειδική κατηγορία ιών η οποία μπορεί να μολύνει με οποιονδήποτε από τους τρόπους που περιγράφηκαν ανωτέρω, έχει όμως το ιδιαίτερο χαρακτηριστικό η μόλυνση να λαμβάνει κάθε φορά διαφορετική μορφή, δυσχεραίνοντας έτσι το έργο της ανίχνευσης. Οι πολυμορφικοί ιοί έχουν αυξηθεί κατά τα τελευταία έτη, καθώς υπάρχουν ακόμη και *εργαλειοθήκες* τις οποίες μπορεί να χρησιμοποιήσει ο κάθε συγγραφέας ιών για να προσδώσει πολυμορφικά χαρακτηριστικά στο δημιούργημά του.

Στην πιο απλή του μορφή, ένας πολυμορφικός ιός έχει ως συστατικά έναν απλό ιό, έναν αλγόριθμο κρυπτογράφησης/αποκρυπτογράφησης και τα σχετικά κλειδιά. Ένα μολυσμένο αντικείμενο έχει προσκολλημένα πάνω του τον αλγόριθμο αποκρυπτογράφησης, τον ιό σε κρυπτογραφημένη μορφή και το κλειδί αποκρυπτογράφησης. Κατά την ενεργοποίηση του ιού, η διαδικασία αποκρυπτογράφησης εκτελείται με είσοδο τον κρυπτογραφημένο ιό και το κλειδί αποκρυπτογράφησης, παράγοντας το «κανονικό» σώμα του ιού, το οποίο και ακολούθως εκτελείται. Κατά τη φάση της μόλυνσης, δημιουργείται ένα νέο ζεύγος κλειδιών κρυπτογράφησης-αποκρυπτογράφησης, ο ιός κρυπτογραφείται βάσει του κλειδιού κρυπτογράφησης και το αποτέλεσμα, μαζί με τον αλγόριθμο αποκρυπτογράφησης και το νέο κλειδί αποκρυπτογράφησης προσκολλώνται στο αντικείμενο-θύμα.

Η πιο εξελιγμένη μορφή των πολυμορφικών ιών φροντίζει να μεταλλάσσει *και τη διαδικασία αποκρυπτογράφησης*, έτσι ώστε δύο αντικείμενα που έχουν μολυνθεί με τον ίδιο ιό να μην έχουν αναγνωρίσιμα κοινά τμήματα, ώστε να κινήσουν υποψίες για δράση ιού.

9.3.9 Τεχνικές απόκρυψης

Οι τεχνικές απόκρυψης (stealth techniques) αποσκοπούν στο να αποτρέψουν την ανίχνευση των ιών, συγκαλύπτοντας τις τροποποιήσεις που αυτοί έχουν επιφέρει είτε στη μνήμη (όπου εγκαθίστανται περιμένοντας να μολύνουν και άλλα αντικείμενα) είτε σε τμήματα του δίσκου, δηλαδή αρχεία και τομείς εκκίνησης. Η γενική φιλοσοφία των τεχνικών απόκρυψης συνίσταται στην παγίδευση των κλήσεων συστήματος που θα μπορούσαν να αποκαλύψουν την ύπαρξη του ιού και την τροποποίηση των αποτελεσμάτων τους κατά τρόπο ώστε η εξέταση του συστήματος διαμέσου των κλήσεων αυτών να δίνει την εικόνα ενός «καθαρού» συστήματος.

Για παράδειγμα, ένας τρόπος ανίχνευσης ιών θα ήταν να ζητηθεί από το λειτουργικό σύστημα η λίστα των δεσμευμένων μπλοκ μνήμης και να συγκριθεί με τη λίστα των μπλοκ που αναφέρουν ότι έχουν δεσμεύσει οι «νόμιμες» εφαρμογές. Τυχόν ύπαρξη επιπλέον δεσμευμένων μπλοκ θα αποκάλυπτε την ύπαρξη ιών. Η τεχνική απόκρυψης εδώ συνίσταται στο «φιλτράρισμα» του αποτελέσματος που επιστρέφει η σχετική κλήση συστήματος του λειτουργικού, έτσι ώστε να απομακρύνεται από αυτή το μπλοκ μνήμης που έχει δεσμεύσει ο ιός, με συνέπεια η τελική αναφορά όντως να δίνει την εικόνα ενός «καθαρού» συστήματος, μολονότι το σύστημα είναι μολυσμένο. Αντιστοίχως θα μπορούσαν να τροποποιηθούν οι διαδικασίες ανάγνωσης αρχείων, έτσι ώστε αν κανείς «διαβάζει» ένα μολυσμένο εκτελέσιμο πρόγραμμα (πιθανώς για να το αναλύσει για ύπαρξη ιών) να λαμβάνει μία «καθαρή» εικόνα του, ενώ αν *εκτελεί* το ίδιο πρόγραμμα (διαφορετική κλήση συστήματος) το αρχείο να χρησιμοποιείται στη μολυσμένη του μορφή.

Οι τεχνικές απόκρυψης είναι ο κύριος λόγος για τον οποίο η χρήση εργαλείων για εντοπισμό ιών πρέπει να γίνεται εκτελώντας λειτουργικό σύστημα που είναι 100% απαλλαγμένο από ιούς. Η απομάκρυνση των ιών αυτών πρέπει να γίνεται πάντα με ειδικά προγράμματα, καθώς υπάρχει κίνδυνος να επηρεασθεί η προσβασιμότητα στα δεδομένα.

9.4 Αντιμετώπιση των ιών

Με δεδομένο ότι οι ιοί είναι ένας εν δυνάμει ιδιαίτερα καταστροφικός εχθρός της ασφάλειας οποιουδήποτε πληροφοριακού συστήματος, είναι απαραίτητο να υπάρχει μία στρατηγική πρόληψης και αντιμετώπισής τους. Η στρατηγική αυτή έχει δύο σκέλη, το *διαδικαστικό σκέλος* που συνίσταται σε ενέργειες που πρέπει ή δεν πρέπει να γίνονται από τους χρήστες και τους διαχειριστές, ενώ το *τεχνικό σκέλος* περιλαμβάνει κυρίως λογισμικό και ρυθμίσεις. Το διαδικαστικό σκέλος της αντιμετώπισης ιών περιέχει τα ακόλουθα βήματα:

1. *Η επεξεργασία των δεδομένων γίνεται μόνο με συγκεκριμένα και ελεγμένα προγράμματα.* Από τη στιγμή που σε έναν υπολογιστή εκτελούνται μόνο προγράμματα που δεν περιέχουν ιούς, ο υπολογιστής είναι βέβαιο ότι δεν θα μολυνθεί. Νέα προγράμματα που υπόσχονται περισσότερη λειτουργικότητα ή ευχρηστία μπορούν κάλλιστα να είναι δούρειοι ίπποι που θα εναποθέσουν ιούς στο σύστημά μας. Ως ακραίο μέτρο για την αποφυγή εισόδου νέων ανέλεγκτων προγραμμάτων στα συστήματά τους, πολλές εταιρίες έχουν αφαιρέσει τις μονάδες δισκέτας από τους υπολογιστές τους.
2. *Αποφυγή λήψης και εκτέλεσης αρχείων που έχουν επισυναφθεί σε ύποπτα μηνύματα ή από αμφιβόλου αξιοπιστίας ιστοχώρους ή από newsgroups*. Τα προγράμματα αυτά είναι το κυριότερο μέσο διάδοσης ιών και οι χρήστες πρέπει συνειδητά να αποφεύγουν την εκτέλεσή τους.

3. *Συχνή λήψη εφεδρικών αντιγράφων και τήρησή τους επί μακρόν.* Ανεξαρτήτως των μέτρων που θα ληφθούν για πρόληψη, είναι τελικά πιθανό να μολυνθεί το σύστημα από ιό. Στην περίπτωση αυτή θα πρέπει να είμαστε σε θέση να αποκαταστήσουμε την προ της μολύνσεως εικόνα του συστήματος. Καθώς η αποκάλυψη του ιού μπορεί να μην είναι άμεση, θα πρέπει να τηρούμε τα εφεδρικά αντίγραφα για πολύ καιρό ώστε η αντικατάσταση των μολυσμένων αρχείων με τα αντίστοιχα «καθαρά» να είναι εφικτή.

Στο τεχνικό σκέλος μπορούμε να διακρίνουμε τις κάτωθι συνιστώσες:

1. *Απαγόρευση της εκκίνησης από μονάδες δισκέτας.* Οι περισσότεροι ιοί τομέων εκκίνησης μολύνουν τους υπολογιστές κατά την εκκίνηση από μολυσμένη δισκέτα. Οι χρήστες δεν πρέπει να ξεκινούν τους υπολογιστές με δισκέτες μέσα στη μονάδα, ενώ το BIOS των υπολογιστών πρέπει να ρυθμίζεται κατάλληλα ώστε να μην προσπαθεί να εκκινήσει τον υπολογιστή από τη δισκέτα.
2. *Ρύθμιση του υπολογιστή στο μέγιστο επίπεδο ασφάλειας.* Η οδηγία αυτή ισχύει κατά μείζονα λόγο για προγράμματα πλοήγησης και ανάγνωσης ηλεκτρονικού ταχυδρομείου, καθώς αυτά είναι εν δυνάμει πύλες εισόδου ιομορφικού λογισμικού στο σύστημα.
3. *Χρήση ειδικού λογισμικού για αντιμετώπιση ιών.* Το λογισμικό αντιμετώπισης ιών είναι ένα ζήτημα που αναλύεται εκτενώς στις παραγράφους που ακολουθούν.

9.5 Λογισμικό αντιμετώπισης των ιών

Το λογισμικό αντιμετώπισης ιών είναι μία πολυσύνθετη κατηγορία λογισμικού που περιλαμβάνει εργαλεία με στόχο:

1. *την ανίχνευση των ιών.* Τα εργαλεία ανίχνευσης προσδιορίζουν και αναφέρουν αν το σύστημά μας έχει μολυνθεί από ιό. Η ανίχνευση μπορεί να γίνεται είτε με *ανάλυση των αντικειμένων του συστήματος* σε περιοδική βάση ή κατά τη χρησιμοποίησή τους, με *παρεμπόδιση των παράνομων ενεργειών* ή *με ανίχνευση των αναιτιολόγητων αλλαγών*.
2. *τον προσδιορισμό της ταυτότητας των ιών.* Αν το σύστημά μας έχει μολυνθεί από ιό, ένα εργαλείο προσδιορισμού ταυτότητας θα μας πληροφορήσει για το ποιος συγκεκριμένος ιός έχει προκαλέσει τη μόλυνση. Ο προσδιορισμός ταυτότητας είναι χρήσιμος αφ' ενός για να μπορέσουμε να αποτιμήσουμε το μέγεθος της ζημιάς και τον πιθανό κίνδυνο που διατρέχουμε, αφ' ετέρου δε για να χαράξουμε τη βέλτιστη στρατηγική επανόρθωσης.
3. *τον καθαρισμό των ιών.* Σε πολλές περιπτώσεις οι αλλαγές που έχουν επιφέρει οι ιοί στο σύστημα είναι αντιστρέψιμες με αυτοματοποιημένο τρόπο. Τα εργαλεία καθαρισμού φροντίζουν για την αναίρεση των αλλαγών που έχουν προκαλέσει οι ιοί.

9.6 Κριτήρια επιλογής εργαλείων

Για να επιλέξουμε ποια κατηγορία εργαλείων αντιμετώπισης ιών είναι πιο κατάλληλη για το σύστημά μας, ή ακόμη και για επιλογή μεταξύ εργαλείων της ίδιας κατηγορίας θα χρησιμοποιήσουμε τέσσερις άξονες κριτηρίων: την *ακρίβεια*, την *ευχρηστία*, τη *διαχειριστική επιβάρυνση* και την *επιβάρυνση του συστήματος*. Στις παραγράφους που ακολουθούν αναλύονται οι τέσσερις αυτοί άξονες.

9.6.1 Ακρίβεια

Ένα ακριβές εργαλείο πρέπει να επιτελεί τον σκοπό του σωστά σε όλες τις περιπτώσεις. Η ακρίβεια λαμβάνει διαφορετική σημασία για κάθε μία από τις κατηγορίες εργαλείων ως εξής:

1. *για τα εργαλεία ανίχνευσης ιών*, ως ακριβή ορίζονται τα εργαλεία που ανιχνεύουν τους ιούς, όλους τους ιούς και μόνον τους ιούς. Η ακρίβεια χάνεται όταν έχουμε αναφορά ανύπαρκτων ιών (*false positives*) ή αδυναμία εντοπισμού υπαρκτών ιών (false negatives). Η αδυναμία εντοπισμού υπαρκτών ιών είναι σαφώς πιο επικίνδυνη καθώς δεν λαμβάνονται μέτρα για τον ιό που έχει ήδη μολύνει το σύστημα, αλλά από την άλλη πλευρά η συχνή αναφορά ανύπαρκτων ιών μειώνει την αξιοπιστία του συστήματος και ενδεχομένως η χρήστες να μην λάβουν υπόψη τους μία ορθή αναφορά ύπαρξης ιού.
2. *για τα εργαλεία προσδιορισμού ταυτότητας ιών*, ως ακριβή ορίζονται τα εργαλεία που προσδιορίζουν επακριβώς τον ιό που έχει μολύνει το σύστημα. Η ακρίβεια εδώ χάνεται όταν δεν είναι δυνατόν να προσδιοριστεί συγκεκριμένος ιός ή προσδιορίζεται ιός διαφορετικός από αυτόν που πραγματικά έχει μολύνει το σύστημα. Το θέμα «ποιοι ιοί είναι διαφορετική μεταξύ τους» έχει εκλάβει φιλοσοφικές διαστάσεις στην κοινότητα που ασχολείται με την αντιμετώπιση των ιών. Στα πλαίσια του μαθήματος, ως κριτήρια διαφορετικότητας των ιών θα χρησιμοποιείται (α) η διαφορά στη ζημιά που μπορούν να προκαλέσουν και (β) η διαφορά στη στρατηγική ή τα εργαλεία αντιμετώπισής τους.
3. *για τα εργαλεία καθαρισμού ιών*, ως ακριβή ορίζονται τα εργαλεία εκείνα που καταφέρνουν να φέρουν το κάθε μολυσμένο αντικείμενο στην κατάσταση που βρισκόταν πριν μολυνθεί από τον ιό. Η ακρίβεια εδώ μειώνεται όταν η επάνοδος αυτή δεν επιτυγχάνεται πλήρως, αλλά στην απώλεια ακρίβειας μπορούμε να έχουμε δύο υποπεριπτώσεις. Η πρώτη είναι η *συνολική αποτυχία*, όπου παράγεται αντικείμενο που δεν λειτουργεί (εκτελέσιμο πρόγραμμα που δεν «τρέχει» ή τομέας εκκίνησης που δεν εκκινεί το σύστημα) ή όταν η απομάκρυνση του ιού είναι συνολικά αδύνατη. Η δεύτερη είναι η *μερική αποτυχία*, όπου παράγεται μεν ένα αντικείμενο που λειτουργεί, αλλά είναι διαφορετικό από το αρχικό.

9.6.2 Ευχρηστία

Τα ζητήματα ευχρηστίας αξιολογούν το κατά πόσο είναι εύκολο για τον χρήστη να χρησιμοποιήσει το λογισμικό, δηλαδή να το ενεργοποιήσει και να ανταποκριθεί σωστά στα μηνύματα που τυχόν του εμφανίζεται. Επιπρόσθετα, αξιολογείται το κατά πόσο τροποποιείται η συνήθης λειτουργία του συστήματος αναφορικά με τους χρήστες, δηλαδή αν αυτοί αναγκάζονται να κάνουν πρόσθετες ενέργειες ή να εκτελούν κάποιες διαδικασίες με πιο δύσκολο τρόπο.

9.6.3 Διαχειριστική επιβάρυνση

Κάθε λογισμικό που σε ένα υπολογιστικό σύστημα έχει ένα *διαχειριστικό κόστος* που αντανακλά το πόσο πρέπει να απασχοληθεί η ομάδα διαχείρισης του συστήματος με το συγκεκριμένο λογισμικό. Το διαχειριστικό κόστος είναι το άθροισμα του κόστους εγκατάστασης, του κόστους ρύθμισης, του κόστους συντήρησης και του κόστους υποστήριξης των τελικών χρηστών.

9.6.4 Επιβάρυνση συστήματος

Τα εργαλεία αντιμετώπισης ιών, όταν εγκατασταθούν σε κάποιο σύστημα καταναλώνουν πόρους του συστήματος, όπως χώρος στον δίσκο, μνήμη και κύκλοι της κεντρικής μονάδας επεξεργασίας. Όλα αυτά θα προκαλέσουν μία επιβράδυνση της λειτουργίας του συστήματος, η οποία οφείλει να είναι όσο το δυνατόν μικρότερη. Στην ιδεώδη περίπτωση, οι πόροι (κυρίως η μνήμη και η ΚΜΕ) θα καταναλώνονται σε χρόνο που δεν θα επικαλύπτεται με τις ώρες «κανονικής» χρήσης του μηχανήματος από τους χρήστες, αν και αυτό δεν είναι πάντα εφικτό.

9.7 Εργαλεία και τεχνικές

Στις παραγράφους που ακολουθούν θα παρουσιασθούν οι κυριότερες κατηγορίες εργαλείων λογισμικού και τεχνικών για την αντιμετώπιση των ιών.

9.7.1 Εντοπισμός υπογραφών

Η μέθοδος αυτή χρησιμοποιείται κυρίως από εργαλεία εντοπισμού ιών, τα οποία δρουν παράλληλα και ως εργαλεία προσδιορισμού ταυτότητας των ιών, και τα οποία επιχειρούν να ανιχνεύσουν αν υπάρχει ιός προσκολλημένος σε αντικείμενα του δίσκου ή εγκατεστημένος στη μνήμη. Τα εργαλεία αυτά είναι δυνατόν να ελέγχουν τα αντικείμενα σε κάθε πρόσβαση που γίνεται σε αυτά, ή σε περιοδική βάση, π.χ. μία φορά κάθε εβδομάδα.

Με τον όρο «υπογραφή ιού» περιγράφεται μία ακολουθία από bytes τα οποία είναι γνωστό ότι ανήκουν σε ιούς ή οικογένειες ιών. Για παράδειγμα, αν ο κώδικας ενός συγκεκριμένου ιού για προσωπικούς υπολογιστές περιλαμβάνει τις εντολές συμβολικής γλώσσας:

```
push ax
mov ax, 2032
push bx
mov bx, 6782
call 8776
pop bx
pop ax
```

αυτό σημαίνει ότι το αντικείμενο στο οποίο ο ιός έχει προσκολληθεί θα περιλαμβάνει την ακολουθία bytes

```
50 B8 32 20 53 BB 82 67 E8 6B 86 5B 58
```

Μία ακολουθία από bytes που θα χρησιμοποιηθεί ως υπογραφή ενός ιού πρέπει να έχει τα εξής δύο χαρακτηριστικά:

1. να υπάρχει σε όλα τα αρχεία που έχουν μολυνθεί από το ιό.
2. να είναι αδύνατον (ή τουλάχιστον σχετικά απίθανο) να εμφανιστεί η συγκεκριμένη ακολουθία σε αρχεία που δεν έχουν μολυνθεί από τον ιό.

Οι υπογραφές συλλέγονται από μολυσμένα αντικείμενα, μετά από ανάλυσή τους.

Μία υπογραφή μπορεί να περιέχει μεταχαρακτήρες, προκειμένου να αντιμετωπίζονται περιπτώσεις όπου συγκεκριμένα bytes μπορούν να αλλάζουν ανάμεσα σε διαφορετικά αντικείμενα που έχουν μολυνθεί από τον ίδιο ιό. Για παράδειγμα, στον κώδικα που παρατίθεται ανωτέρω, η διεύθυνση που καλείται μέσω εντολή `call` της πέμπτης γραμμής θα μπορούσε να είναι διαφορετική για δύο

μολυσμένα αρχεία, ανάλογα με το σημείο του αρχείου όπου έχει προσκολληθεί ο ιός. Στην περίπτωση αυτή η υπογραφή του ιού θα μπορούσε να διαμορφωθεί σε

```
50 B8 32 20 53 BB ? ? E8 6B 86 5B 58
```

όπου κάθε χαρακτήρας `?` έχει την έννοια «ένα, οποιοδήποτε byte». Παρομοίως θα μπορούσαν να χρησιμοποιηθούν και άλλοι μεταχαρακτήρες, όπως το `*` (οσαδήποτε, οποιαδήποτε bytes – αν και η εμφάνισή του είναι σχετικά ασυνήθιστη) ή ειδικές γραφές για την αναπαράσταση ενός byte που μπορεί να λάβει μία τιμή από ένα καθορισμένο σύνολο (π.χ. `{02, 2A, C0, DF}`) κ.λπ.

Η υπογραφή μπορεί επίσης να συμπληρώνεται από κάποια *ένδειξη θέσης εντός του αντικειμένου*, π.χ. «τα πρώτα bytes του αντικειμένου» (περίπτωση που θα κάλυπτε τους ιούς που επικαλύπτουν τον εκτελέσιμο κώδικα), «τα τελευταία bytes του αντικειμένου» (π.χ. ιοί που προσκολλώνται στο τέλος και τοποθετούν εντολές άλματος στην αρχή των αντικειμένων) ή «μέσα στα 300 πρώτα bytes του αντικειμένου». Η παράθεση της ένδειξης θέσης είναι χρήσιμη αφ' ενός διότι μειώνει το πλήθος των bytes που πρέπει να ελεγχθούν για ύπαρξη της υπογραφής, αφ' ετέρου δε διότι μειώνει την πιθανότητα να ανιχνευθεί η υπογραφή σε κάποιο σημείο όπου η παρουσία της δεν υποδηλώνει ύπαρξη ιού.

Οι πολυμορφικοί ιοί είναι δύσκολο να ανιχνευθούν μέσω υπογραφών καθώς μεταλλάσσονται τόσο πολύ μεταξύ δύο μολύνσεων που δεν έχουν κάποια σταθερή «υπογραφή». Για τους ιούς αυτούς απαιτείται αλγοριθμική ανίχνευση.

9.7.1.1 Ακρίβεια

Η μέθοδος του εντοπισμού υπογραφών είναι απόλυτα ακριβής αν έχουν ελεγχθεί από τον κατασκευαστή του σχετικού λογισμικού όλοι οι ιοί και όλα τα «κανονικά» εκτελέσιμα, κάτι που προφανώς είναι αδύνατον. Έτσι είναι πιθανόν να υπάρξουν ψευδείς αναφορές για ύπαρξη ιών, όταν κάποια «υπογραφή» συμβεί να υπάρχει σε κάποιο «κανονικό» πρόγραμμα, ή να μην αναφερθεί η ύπαρξη κάποιου υπαρκτού ιού, είτε διότι αυτός χρησιμοποιεί τεχνικές απόκρυψης (stealth techniques) είτε διότι το «πακέτο υπογραφών» που χρησιμοποιεί το λογισμικό είναι ελλιπές ή παρωχημένο και δεν περιλαμβάνει την «υπογραφή» του ιού. Με δεδομένο ότι τα εργαλεία που βασίζονται στον εντοπισμό υπογραφών λειτουργούν και ως εργαλεία προσδιορισμού ταυτότητας, η διάσταση της ακρίβειας περιλαμβάνει και τον ορθό προσδιορισμό της ταυτότητας του ιού, και πιθανά προβλήματα σ' αυτή τη διάσταση περιλαμβάνουν την αναφορά λανθασμένης ταυτότητας ιού ή εσφαλμένης παραλλαγής του ίδιου ιού, κάτι που μπορεί να συμβεί αν οι ιοί «μοιάζουν» πολύ και αν δεν έχουν διαμορφωθεί με αρκετή ακρίβεια τα «πακέτα υπογραφών».

9.7.1.2 Ευχρηστία

Τα εργαλεία που βασίζονται στον εντοπισμό υπογραφών είναι ιδιαίτερα εύχρηστα για τους τελικούς χρήστες, καθώς απαιτούν πολύ λίγες γνώσεις. Ουσιαστικά οι χρήστες πρέπει μόνο να ζητούν την εκτέλεσή τους και να ανταποκρίνονται σε μηνύματα του τύπου «το τάδε αρχείο είναι μολυσμένο από ιό», μήνυμα που είναι καταληπτό ακόμη και από μη ειδικούς στην πληροφορική.

9.7.1.3 Διαχειριστική επιβάρυνση

Δεδομένου ότι κάθε μέρα εμφανίζονται νέοι ιοί, ένα «πακέτο υπογραφών» που φτιάχνεται σήμερα σε μία εβδομάδα θα υπολείπεται σημαντικά σε γνώση για νέους

ιούς. Η διαρκής ενημέρωση που απαιτείται μπορεί να αποτελέσει πρόβλημα σε μεγάλους οργανισμούς, ειδικά αν δεν υπάρχει η δυνατότητα αυτοματοποίησής της.

Η εγκατάσταση των σχετικών εργαλείων είναι εύκολη, ακόμη και για απλούς χρήστες και, στον βαθμό που οι διαγνώσεις είναι σωστές, δεν απαιτείται υποστήριξη των χρηστών. Αν, ωστόσο, το πρόγραμμα αναφέρει κάποιον ιό που δεν υπάρχει στην πραγματικότητα (false positive), η συνδρομή του διαχειριστή είναι απαραίτητη.

9.7.1.4 Επιβάρυνση συστήματος

Τα εργαλεία που βασίζονται σε εντοπισμό υπογραφών είναι αρκετά αποδοτικά, καθώς χρησιμοποιούνται για πολύ καιρό και οι σχετικοί αλγόριθμοι έχουν βελτιστοποιηθεί σε μεγάλο βαθμό. Ειδικότερα, αν λειτουργούν σε περιοδική βάση (σε αντιδιαστολή με τη διαρκή παρακολούθηση), η επιβάρυνση του συστήματος είναι αμελητέα, καθώς μάλιστα μπορούν να διενεργούν τους ελέγχους σε περιόδους που το σύστημα δεν χρησιμοποιείται παραγωγικά (π.χ. βραδινές ώρες, Σαββατοκύριακα).

9.7.1.5 Εντοπισμός υπογραφών – Σύνοψη

Ο πίνακας που ακολουθεί συνοψίζει τα θετικά και αρνητικά σημεία της τεχνικής εντοπισμού υπογραφών και των εργαλείων που βασίζονται σ' αυτή.

Υπέρ	***Κατά***
Τα καλά συντηρούμενα συστήματα εντοπίζουν άνω του 95% των ιών	Βρίσκουν μόνο τους ιούς που ήταν γνωστοί κατά την ανάπτυξη του «πακέτου υπογραφών»
Δρουν και ως εργαλεία προσδιορισμού ταυτότητας, μειώνοντας τον χρόνο ανάκαμψης	Πρέπει να συντηρούνται διαρκώς
Δοκιμασμένη τεχνολογία με βελτιστοποιημένους αλγόριθμους	Είναι επιρρεπή σε εσφαλμένους προσδιορισμούς ταυτότητας
Απαιτείται ελάχιστη γνώση	Οι χρήστες παρανοούν το «δεν ανιχνεύθηκε ιός» πιστεύοντας ότι σημαίνει «δεν υπάρχει ιός»

9.7.2 Έλεγχος ακεραιότητας

Ο έλεγχος ακεραιότητας είναι μία τεχνική που χρησιμοποιείται για την ανίχνευση μόνο των ιών, χωρίς να δίνει τη δυνατότητα προσδιορισμού της ταυτότητάς τους. Η τεχνική αυτή έχει δύο στάδια:

1. Στο πρώτο στάδιο δημιουργείται μία βάση δεδομένων με αθροίσματα ελέγχου, για κάθε αντικείμενο του συστήματος που είναι πιθανό θύμα επίθεσης ιού. Τα αθροίσματα ελέγχου μπορούν να δημιουργούνται με χρήση κυκλικών πλεοναστικών κωδίκων (CRC) ή με κρυπτογραφικές μεθόδους. Καθώς η χρήση κρυπτογραφικών μεθόδων είναι ιδιαίτερα δαπανηρή υπολογιστικά, μία τρίτη επιλογή είναι να δημιουργείται πρώτα μία συνάρτηση κερματισμού (hash function) στο αντικείμενο και στη συνέχεια να εφαρμόζεται στο αποτέλεσμά της κάποια κρυπτογραφική μέθοδος. Η δημιουργία της βάσης δεδομένων πρέπει να γίνει σε «καθαρό» σύστημα, δηλαδή σε σύστημα που είναι βέβαιο πως δεν έχει μολυνθεί από ιό.

2. Στο δεύτερο στάδιο για κάθε αντικείμενο του συστήματος επανυπολογίζεται το άθροισμα ελέγχου και συγκρίνεται με το αντίστοιχο άθροισμα ελέγχου που έχει αποθηκευθεί στη βάση δεδομένων. Αν τα αθροίσματα ελέγχου είναι διαφορετικά, τότε το αντικείμενο έχει τροποποιηθεί, πιθανότατα λόγω δράσης κάποιου ιού.

9.7.2.1 Ακρίβεια

Η τεχνική του ελέγχου ακεραιότητας εντοπίζει όλους τους ιούς, αρκεί ο αρχικός υπολογισμός να έχει γίνει σε καθαρό σύστημα, συνεπώς δεν υπάρχει πιθανότητα να μην αναφερθεί κάποιος υπάρχων ιός (false negative). Υπάρχει όμως σημαντική πιθανότητα για ψευδείς αναφορές ύπαρξης ιών, καθώς δεν είναι σπάνιες οι περιπτώσεις προγραμμάτων που *τροποποιούν τον εαυτό τους*, π.χ. για να αποθηκεύσουν ρυθμίσεις, στατιστικά στοιχεία ή οποιαδήποτε άλλη πληροφορία. Επίσης, η τεχνική αυτή δεν μπορεί να αντιμετωπίσει καθόλου τους ιούς μακροεντολών, καθώς αυτοί μολύνουν τα αρχεία δεδομένων, τα οποία είναι φυσιολογικό να αλλάζουν ως αποτέλεσμα της επεξεργασίας τους.

9.7.2.2 Ευχρηστία

Για να είναι αποτελεσματική η τεχνική αυτή, θα πρέπει η βάση δεδομένων των αθροισμάτων ελέγχου πρέπει να αποθηκεύεται σε μη προσβάσιμη περιοχή, καθώς αν η περιοχή είναι προσβάσιμη ένας πιο «έξυπνος» ιός θα μπορούσε εκτός από το αντικείμενο να τροποποιήσει και το σχετικό άθροισμα ελέγχου στη βάση δεδομένων. Η απαίτηση αυτή εισάγει διαδικασίες που δεν είναι αρεστές στους μέσους χρήστες.

Πέραν αυτού, ο μέσος χρήστης δεν γνωρίζει αν ένα πρόγραμμα αυτοτροποποιείται ή όχι, και τέτοιου είδους ψευδείς αναφορές δημιουργούν μεγάλα προβλήματα στους χρήστες. Αν δε, το πλήθος των ψευδών αναφορών γίνει μεγάλο, οι χρήστες θα γίνουν ιδιαίτερα επιφυλακτικοί στις προειδοποιήσεις.

9.7.2.3 Διαχειριστική επιβάρυνση

Η εγκατάσταση των εργαλείων ελέγχου ακεραιότητας είναι ιδιαίτερα εύκολη, και στη φάση της εγκατάστασης υπάρχει η δυνατότητα να ολοκληρωθεί και ο αρχικός υπολογισμός των αθροισμάτων ελέγχου. Είναι ωστόσο απαραίτητο η διαδικασία του υπολογισμού αθροισμάτων να επαναλαμβάνεται σε *κάθε* εγκατάσταση ή αναβάθμιση λογισμικού ή εγκατάσταση επιδιορθωτικών προγραμμάτων. Υπάρχει τέλος αναγκαιότητα για την διαρκή υποστήριξη των χρηστών, καθώς αυτοί δεν έχουν συνήθως τις ικανότητες να αντιμετωπίσουν μηνύματα του τύπου «το τάδε αρχείο έχει αλλάξει».

9.7.2.4 Επιβάρυνση συστήματος

Η τεχνική του ελέγχου ακεραιότητας δεν επηρεάζει τη συνήθη λειτουργία του συστήματος, καθώς η όλη διαδικασία μπορεί να λαμβάνει χώρα σε περιόδους όπου οι υπολογιστές δεν χρησιμοποιούνται παραγωγικά. Ο χρόνος υπολογισμού των αθροισμάτων ελέγχου μπορεί ωστόσο να είναι σημαντικός, ιδιαίτερα αν χρησιμοποιούνται αμιγώς κρυπτογραφικές μέθοδοι για την παραγωγή τους.

9.7.2.5 Έλεγχος ακεραιότητας – Σύνοψη

Ο πίνακας που ακολουθεί συνοψίζει τα θετικά και αρνητικά σημεία της τεχνικής ελέγχου ακεραιότητας και των εργαλείων που βασίζονται σ' αυτή.

Υπέρ	*Κατά*
Δεν χρειάζονται ενημέρωση	Πρέπει να υπολογίζονται αθροίσματα ελέγχου σε κάθε εγκατάσταση-αναβάθμιση προγράμματος
	Δεν βρίσκουν τους ιούς – μόνο τις αλλαγές, συνεπώς απαιτούνται συμπληρωματικά εργαλεία προσδιορισμού ταυτότητας
	Εσφαλμένες αναφορές, κυρίως θετικές
	Δεν είναι καθόλου αποτελεσματικοί για ιούς μακροεντολών

9.7.3 Επόπτες γενικού σκοπού

Οι επόπτες γενικού σκοπού προστατεύουν το σύστημα από τη διάδοση ιών ή τη δράση των Δούρειων Ίππων αναχαιτίζοντας κακόβουλες ενέργειες. Προκειμένου να αναχαιτισθεί μία κακόβουλη ενέργεια θα πρέπει πρώτα να είναι δυνατόν να διαχωρισθούν οι ενέργειες που λαμβάνουν χώρα σε ένα σύστημα σε «φυσιολογικές» και «κακόβουλες». Οι κατασκευαστές των σχετικών εργαλείων μοντελοποιούν τη συμπεριφορά των ιών και των δούρειων ίππων και δημιουργούν κώδικα που προσπαθεί να ανιχνεύσει και να παρεμποδίσει τις ενέργειες αυτές. Ως παραδείγματα μοντέλων κακόβουλων συμπεριφορών μπορούμε να παραθέσουμε τα κάτωθι:

- ένα πρόγραμμα ζητά μνήμη που «αυτονομείται»
- ένα πρόγραμμα ανοίγει αρχεία συστήματος (π.χ. command.com)
- ένα πρόγραμμα ανοίγει εκτελέσιμα σε άλλους καταλόγους
- μακροεντολή σε ένα έγγραφο Word διαγράφει αρχεία MP3

Οι ενέργειες αυτές δεν είναι «φυσιολογικές» και δεν είναι πολύ πιθανό να γίνονται από «κανονικά» προγράμματα. Κατά συνέπεια, η εμφάνιση μιας τέτοιας ενέργειας είναι πολύ πιθανό να σηματοδοτεί τη δράση ενός ιού.

9.7.3.1 Ακρίβεια

Για να είναι δυνατή η ανίχνευση των κακόβουλων ενεργειών πρέπει οι ιοί να συμπεριφερθούν βάσει των μοντέλων που έχουν καθορισθεί από τους κατασκευαστές των εργαλείων. Κάτι τέτοιο δεν συμβαίνει πάντα, καθώς νέες τεχνικές ιών μπορεί να ενεργούν με εντελώς διαφορετικό ή απρόσμενο τρόπο, με αποτέλεσμα να έχουμε ιούς που δεν αναφέρονται (false negatives). Οι ιοί επίσης μπορεί να προσπαθήσουν να απενεργοποιήσουν τον επόπτη, ακυρώνοντας τη δράση του. Αν τα καταφέρουν, το σύστημα θα είναι ανυπεράσπιστο, και έτσι θα πρέπει ο επόπτης να είναι προετοιμασμένος να «αμυνθεί» σε τέτοια ενδεχόμενα. Υπάρχει επίσης η περίπτωση μερικά «κανονικά προγράμματα» να προβαίνουν σε ενέργειες που εντάσσονται στα μοντέλα ιών, προκαλώντας έτσι ψευδείς αναφορές ύπαρξης ιών (false positives).

9.7.3.2 Ευχρηστία

Ο μέσος χρήστης δεν έχει τις γνώσεις να χειρισθεί εργαλεία που βασίζονται στην τεχνική των εποπτών γενικού σκοπού, καθώς η λειτουργία τους απαιτεί λεπτομερειακή διαμόρφωση και ρύθμιση. Επίσης, τα μηνύματα με τα οποία θα έλθουν αντιμέτωποι οι χρήστες δεν είναι πάντα κατανοητά (*το πρόγραμμα ζητά την*

αυτονόμηση ενός μπλοκ μνήμης μεγέθους 32K στη διεύθυνση 8023:AB56 και ο κώδικας σ' αυτό έχει παγιδεύσει το διάνυσμα διακοπής 19).

9.7.3.3 Διαχειριστική επιβάρυνση

Δεδομένου ότι οι επόπτες γενικού σκοπού απαιτούν ρύθμιση, η διαχειριστική επιβάρυνση κατά την εγκατάσταση είναι σημαντική, ειδικά αν υπάρχουν ετερογενή συστήματα ή συστήματα με διαφορετικές απαιτήσεις. Οι απαιτήσεις για υποστήριξη των χρηστών είναι επίσης αυξημένες, εξαρτωμένου βέβαια και από το προφίλ των χρηστών (τεχνικές γνώσεις, πλήθος προγραμμάτων που χρησιμοποιούν, ειδικά χαρακτηριστικά των ενεργειών τους κ.τ.λ.). Από την άλλη πλευρά, ένα καλά διαμορφωμένο σύστημα επόπτη γενικού σκοπού απαιτείται να ενημερωθεί μόνο όταν εμφανίζονται νέοι τρόποι δράσης ιών, ενώ η εμφάνιση απλά νέων ιών που χρησιμοποιούν «γνωστές» στο σύστημα πρακτικές δεν εγείρει ζητήματα ενημέρωσης.

9.7.3.4 Επιβάρυνση συστήματος

Η χρήση εργαλείων που βασίζονται στην τεχνική των εποπτών γενικού σκοπού εισάγει κάποια επιβάρυνση στο σύστημα, η οποία οφείλεται στην παρακολούθηση και ανάλυση των ενεργειών που λαμβάνουν χώρα στο σύστημα, προκειμένου να εντοπισθούν και να αναχαιτισθούν οι κακόβουλες πράξεις. Συνολικά, η επιβάρυνση δεν είναι ιδιαίτερα σημαντική.

9.7.3.5 Επόπτες γενικού σκοπού – Σύνοψη

Ο πίνακας που ακολουθεί συνοψίζει τα θετικά και αρνητικά σημεία της τεχνικής των εποπτών γενικού σκοπού και των εργαλείων που βασίζονται σ' αυτή.

Υπέρ	***Κατά***
Αρκετά γενική τεχνική	Δύσχρηστο για τον μέσο χρήστη
Κανονικά λειτουργεί και για άγνωστους ιούς	Αρκετές ψευδείς αναφορές ύπαρξης ιών
Μικρή συχνότητα ενημερώσεων	Μεγάλο διαχειριστικό κόστος
	Ευάλωτο σε νέες τεχνικές ιών
	Μπορεί να απενεργοποιηθεί από τους ιούς

9.7.4 Κελύφη ελέγχου πρόσβασης

Τα εργαλεία που βασίζονται στην τεχνική των κελυφών ελέγχου πρόσβασης ενσωματώνονται στο λειτουργικό σύστημα και έχουν ως στόχο να επιβάλλουν πολιτικές που ορίζουν *ποιος χρήστης* μπορεί να χρησιμοποιήσει *ποιο πρόγραμμα* για να πραγματοποιήσει *συγκεκριμένους τύπους ενεργειών* σε *ορισμένους τύπους αρχείων*. Με δεδομένο ότι μερικά λειτουργικά συστήματα δεν περιλαμβάνουν ισχυρούς μηχανισμούς διακρίβωσης της ταυτότητας των χρηστών ή περιορισμού της πρόσβασης σε αρχεία (π.χ. Windows 95, 98), μερικά κελύφη ελέγχου πρόσβασης μπορούν να εισάγουν τους δικούς τους μηχανισμούς διακρίβωσης ταυτότητας ή εργαλεία κρυπτογράφησης. Τα εργαλεία κρυπτογράφησης δεν αποτρέπουν την πρόσβαση στα αρχεία, αλλά καθιστούν την πρόσβαση από μη εξουσιοδοτημένους χρήστες ουσιαστικά άχρηστη, καθώς το αρχείο θα είναι σε ακατάληπτη μορφή. Ως παραδείγματα κανόνων πρόσβασης που μπορεί να χρησιμοποιεί ένα κέλυφος ελέγχου πρόσβασης μπορούμε να παραθέσουμε τα εξής:

```
+ (*, winword, "*.doc", rw)
+ (admin, winword, "*.dot", rw)
+ (*, winword, "*.dot", r)
+ (admin, windowsUpdate, "c:\windows\*", "rw")
- (*, *, "c:\windows\system\*", "w")
```

Η πρώτη γραμμή ορίζει ότι όλοι οι χρήστες μπορούν να χρησιμοποιήσουν την εφαρμογή Winword (Word για Windows) για να διαβάσουν ή να τροποποιήσουν αρχεία με επέκταση «.doc». Οι δύο επόμενες γραμμές δίνουν τη δυνατότητα στον διαχειριστή να διαβάζει και να τροποποιεί πρότυπα εγγράφων και στους υπόλοιπους χρήστες να τα διαβάζουν, πάντα μέσω της εφαρμογής Winword. Η τέταρτη γραμμή επιτρέπει στον διαχειριστή να ενημερώνει αρχεία στον κατάλογο `c:\windows\` μέσω της εφαρμογής WindowsUpdate, ενώ η τελευταία γραμμή απαγορεύει όλες τις υπόλοιπες ενημερώσεις στον κατάλογο `c:\windows\system` από οποιονδήποτε χρήστη και με οποιαδήποτε εφαρμογή.

9.7.4.1 Ακρίβεια

Τα κελύφη ελέγχου πρόσβασης ανιχνεύουν όλους τους ιούς που συμπεριφέρονται βάσει των κωδικοποιημένων προτύπων. Ιοί που δεν συμπεριφέρονται σύμφωνα με τα κωδικοποιημένα πρότυπα δεν είναι δυνατόν να ανιχνευθούν. Αν ένας ιός εισέλθει στο σύστημα και μολύνει κάποιο εκτελέσιμο, τότε μπορεί να μολύνει κατ' επέκταση όλα τα αρχεία που επιτρέπεται στον ξενιστή του να τροποποιήσει. Αυτό σημαίνει ότι οι ιοί μακροεντολών διαδίδονται ελεύθερα, καθώς ο ιός εκτελείται πάντα μέσα από το πρόγραμμα που έχει το δικαίωμα να τροποποιήσει τα σχετικά αρχεία δεδομένων. Επίσης, απαιτείται καλή ρύθμιση για να αποφευχθούν εσφαλμένες αναφορές ύπαρξης ιών: για παράδειγμα, αν για το πρόγραμμα Winword δεν περιληφθεί η ρύθμιση για τα αρχεία τύπου ".dot", ένα κέλυφος ελέγχου πρόσβασης θα ανέφερε (εσφαλμένα) την ύπαρξη ιών.

9.7.4.2 Ευχρηστία

Σε ένα περιβάλλον όπου οι χρήστες υποστηρίζονται από διαχειριστές, οι οποίοι αναλαμβάνουν την εγκατάσταση και τη ρύθμιση, οι χρήστες απλά δουλεύουν όπως πριν, ζητώντας επιπλέον προνόμια όταν το σύστημα τους αρνείται λειτουργίες. Για περιβάλλον «οικιακής χρήσης», η τεχνική αυτή είναι ακατάλληλη διότι οι τεχνικές γνώσεις που απαιτούνται υπερβαίνουν κατά πολύ αυτές του μέσου χρήστη.

9.7.4.3 Διαχειριστική επιβάρυνση

Οι τεχνικές που βασίζονται στα κελύφη ελέγχου πρόσβασης απαιτούν σημαντική προσπάθεια από πλευράς διαχειριστών κατά την αρχική εγκατάσταση, καθώς απαιτείται να τεθούν όλοι οι κανόνες των επιτρεπτών ενεργειών. Υπολογίσιμο είναι επίσης το διαχειριστικό κόστος όταν αλλάζει το λογισμικό (εγκαθίσταται νέο ή τροποποιείται το υπάρχον) ή ακόμη και όταν αλλάζουν οι ρόλοι στο εταιρικό περιβάλλον (π.χ. αν ένας εργαζόμενος μεταπηδήσει από το λογιστήριο στη διοίκηση, οι επιτρεπτές γι' αυτόν ενέργειες θα είναι διαφορετικές, συνεπώς οι κανόνες πρέπει να αλλάξουν.

Από την άλλη πλευρά, η ενημέρωση του ίδιου του λογισμικού είναι σπάνια απαραίτητη.

9.7.4.4 Επιβάρυνση συστήματος

Η επιβάρυνση που εισάγεται από τα κελύφη ελέγχου πρόσβασης είναι μικρή, προκειμένου για την επιβολή των πολιτικών. Αν, ωστόσο, είναι απαραίτητη η χρήση μηχανισμών κρυπτογράφησης, η επιβάρυνση είναι αρκετά μεγαλύτερη.

9.7.5 Ευρεστική ανάλυση κώδικα

Η ευρεστική ανάλυση κώδικα έχει ως στόχο να εντοπίζει την ύπαρξη ιών σε αντικείμενα του συστήματος μέσω στατικής ανάλυσης του περιεχομένου τους. Σε αντίθεση με την τεχνική εντοπισμού υπογραφών, η ευρεστική ανάλυση κώδικα δεν προσπαθεί να εντοπίσει συγκεκριμένες ακολουθίες από bytes, αλλά κώδικα που *μοιάζει* με ιό. Για παράδειγμα, ένα πρόγραμμα που έχει ως πρώτη εντολή του μία εντολή άλματος στο τέλος του αρχείου, όπου υπάρχει αυτοτροποποιούμενος κώδικας που καταλήγει με μία εντολή άλματος στην αρχή, είναι πιθανότατα μολυσμένο από ιό. Η τεχνικές αυτές καταλήγουν σε εντοπισμό *πιθανώς μολυσμένων αρχείων*, και για περαιτέρω ενδυνάμωση των συμπερασμάτων πολλές φορές συνδυάζονται με τεχνικές εντοπισμού υπογραφών.

Τα εργαλεία που βασίζονται στην ευρεστική ανάλυση κώδικα είναι συνήθως εύκολα στη χρήση, καθώς δεν απαιτούν ιδιαίτερες ρυθμίσεις, πλην της αρχικής εγκατάστασής τους. Έχουν τη δυνατότητα να ανιχνεύουν άγνωστους ή πολυμορφικούς ιούς, οι οποίοι εμπίπτουν στους ευρεστικούς κανόνες. Από την άλλη πλευρά μπορεί να μην εντοπίσουν όλους τους ιούς, αν δεν περιγράφονται από τον κατάλληλο ευρεστικό κανόνα, ενδέχεται να αναφέρουν ανύπαρκτους ιούς (false positives) αν κάποιο «νομότυπο» πρόγραμμα ταιριάζει με κάποιον από τους ευρεστικούς κανόνες του εργαλείου. Τέλος, η λειτουργία τους απαιτεί πολύ επεξεργαστική ισχύ, καθώς η ανάλυση του κώδικα είναι πιο επαχθής υπολογιστικά από την αναζήτηση ακολουθιών bytes.

9.7.6 Εργαλεία καθαρισμού ιών

Τα εργαλεία καθαρισμού ιών έχουν ως στόχο να απομακρύνουν τους ιούς από το σύστημα, *πραγματοποιώντας τις αντίστροφες αλλαγές από αυτές που επέφερε ο ιός*. Για τη δημιουργία των εργαλείων καθαρισμού, οι οίκοι λογισμικού αναλύουν τη δράση του ιού και αναπτύσσουν κατάλληλους αλγόριθμους αναίρεσης. Οι αλγόριθμοι αναίρεσης ενσωματώνονται ακολούθως στα εργαλεία, είτε μεμονωμένα, καταλήγοντας σε εργαλεία που καθαρίζουν έναν μόνο συγκεκριμένο ιό (κάτι που συνήθως συμβαίνει μετά από «επιδημίες» συγκεκριμένου ιού), είτε κατά ομάδες, οδηγώντας σε εργαλεία καθαρισμού πλειάδων ιών.

Βασικές προϋποθέσεις για να λειτουργήσει σωστά ένα εργαλείο καθαρισμού ιών είναι οι ακόλουθες:

1. *οι αλλαγές πρέπει να είναι αντιστρέψιμες*. Αν ένας ιός καταστρέφει αντικείμενα, επικαλύπτοντας το περιεχόμενό τους με άλλες ακολουθίες bytes χωρίς να αποθηκεύει κάπου το αρχικό περιεχόμενο, η αποκατάσταση της αρχικής μορφής του αντικειμένου δεν είναι εφικτή.
2. *να προσδιορισθεί σωστά ο ιός που έχει μολύνει το αντικείμενο, έτσι ώστε να εφαρμοσθεί ο σωστός αλγόριθμος αναίρεσης*. Προβλήματα εδώ δημιουργούνται πολλές φορές όταν *παραλλαγές* του ίδιου ιού μοιάζουν μεν πολύ, αναφορικά με την αναγνώρισή τους, αλλά ο τρόπος που μολύνουν είναι αρκετά διαφορετικός ώστε να απαιτεί διαφορετικό πρόγραμμα αναίρεσης. Για παράδειγμα, οι παραλλαγές ιών Jerusalem-DC και Jerusalem-E2 απαιτούσαν

διαφορετικό πρόγραμμα καθαρισμού, αν και οι *υπογραφές* τους έμοιαζαν αρκετά για να «μπερδέψουν» τα περισσότερα προγράμματα προσδιορισμού ταυτότητας.

Ο καθαρισμός των ιών είναι δύσκολος ή αδύνατος για πολλαπλές μολύνσεις, δηλαδή για περιπτώσεις όπου το ίδιο αντικείμενο έχει μολυνθεί διαδοχικά από περισσότερους του ενός ιούς. Επίσης, σε κάθε περίπτωση είναι προτιμότερη η αντικατάσταση του μολυσμένου αντικειμένου με ένα καθαρό, π.χ. από το CD διανομής του σχετικού λογισμικού.

10 Συστήματα ανίχνευσης εισβολών

Με τον όρο *ανίχνευση εισβολών* αναφερόμαστε στην παρακολούθηση και ανάλυση των συμβάντων που λαμβάνουν χώρα σε υπολογιστές ή δίκτυα, με σκοπό να εντοπισθούν ενδείξεις προσπαθειών εισβολής. Οι «προσπάθειες εισβολής» περιλαμβάνουν ίχνη από απόπειρες για παραβίαση της ακεραιότητας, εμπιστευτικότητας ή διαθεσιμότητας των πληροφοριακών πόρων, καθώς επίσης και προσπάθειες για παράκαμψη των μηχανισμών ασφάλειας. Μία τέτοια εισβολή μπορεί να προέρχεται:

1. από «εξωτερικούς» προς το εταιρικό δίκτυο χρήστες, οι οποίοι κανονικά δεν έχουν δικαίωμα πρόσβασης στο πληροφοριακό σύστημα, αλλά προσπαθούν να το προσπελάσουν.
2. από «εσωτερικούς» χρήστες που έχουν περιορισμένα δικαιώματα πρόσβασης αλλά επιχειρούν ενέργειες που η πολιτική ασφάλειας τους απαγορεύει.
3. από «εσωτερικούς» χρήστες, οι οποίοι έχουν κατάλληλα δικαιώματα πρόσβασης για τις πράξεις στις οποίες προβαίνουν, αλλά ασκούν τα δικαιώματα αυτά με καταχρηστικό τρόπο. Για παράδειγμα, ένας υπάλληλος της μισθοδοσίας έχει δικαίωμα να τροποποιεί τους μισθούς των υπαλλήλων, αλλά η παροχή στον εαυτό του αύξησης 80% χωρίς τη σχετική εντολή από τη διοίκηση είναι μία περίπτωση καταχρηστικής άσκησης του δικαιώματός του.

Τα *συστήματα ανίχνευσης εισβολών* (ΣΑΕ) είναι συστήματα που συντίθενται από υλικό και λογισμικό και έχουν ως στόχο την αυτοματοποίηση της ανίχνευσης εισβολών.

10.1 Λόγοι εισαγωγής Συστημάτων Ανίχνευσης Εισβολών

Υπάρχουν πολυάριθμοι λόγοι για τους οποίους ένας οργανισμός θα επιθυμούσε να εγκαταστήσει και να θέσει σε λειτουργία ένα σύστημα ανίχνευσης εισβολών. Οι πιο σημαντικοί παρατίθενται στη συνέχεια.

1. *πρόληψη προβλημάτων*. Τα συστήματα ανίχνευσης εισβολών συνεισφέρουν στην πρόληψη προβλημάτων κατά δύο τρόπους: αφ' ενός είναι πιθανόν να επισημάνουν τις προσπάθειες εισβολής σε ένα πρώιμο στάδιο, οπότε και θα ληφθούν τα κατάλληλα μέτρα για την αντιμετώπισή τους πριν γίνει κάποια σημαντική ζημιά. Αφ' ετέρου, γνωρίζοντας οι επίδοξοι εισβολείς ότι υφίσταται κάποιο τέτοιο σύστημα, ξέρουν ότι η πιθανότητα αποκάλυψης και τιμωρίας τους είναι σαφώς μεγαλύτερη, και κατά συνέπεια ενδέχεται να μην εκδηλώσουν συνολικά την επίθεσή τους.
2. *ανίχνευση επιθέσεων και παραβιάσεων που δεν ανιχνεύονται με άλλα μέσα*. Επί παραδείγματι, οι καταχρήσεις δικαιωμάτων από εσωτερικούς χρήστες δεν είναι δυνατόν να αντιμετωπισθούν με σχήματα διακρίβωσης ταυτότητας και

ελέγχου πρόσβασης, διότι τα σχήματα αυτά δεν είναι σχεδιασμένα για να αντιμετωπίζουν τέτοιου είδους ζητήματα.

3. *εντοπισμός και αντιμετώπιση προσπαθειών ανίχνευσης.* Ένα τυπικό σχήμα επίθεσης σε πληροφοριακά συστήματα χωρίζεται σε τρεις φάσεις: αρχικά ανιχνεύεται το πληροφοριακό σύστημα για να διαπιστωθεί η διαμόρφωσή του και οι προσφερόμενες από αυτό υπηρεσίες. Στη συνέχεια, ανασύρονται από «βιβλιοθήκες» οι τεχνικές που είναι δυνατόν να χρησιμοποιηθούν για να παραβιαστεί η ασφάλεια του συστήματος και, τέλος, οι τεχνικές αυτές χρησιμοποιούνται. Ενώ τα υπόλοιπα μέτρα ασφάλειας (firewalls, προγράμματα επιδιόρθωσης, έλεγχος πρόσβασης κ.ά.) εστιάζονται στην αντιμετώπιση της τελευταίας φάσης, τα συστήματα ανίχνευσης εισβολών μπορούν να ανιχνεύσουν τις προσπάθειες ανίχνευσης και να τις αναχαιτίσουν ή να ενημερώσουν σχετικά τους διαχειριστές για λήψη μέτρων. Η άμεση αντίδραση σε τέτοια ενδεχόμενα θωρακίζει το σύστημα και αποθαρρύνει τους επίδοξους εισβολείς.

4. *Τεκμηρίωση υπαρκτών απειλών.* Τα συστήματα ανίχνευσης εισβολών μπορούν να αποδείξουν το γεγονός ότι ένα πληροφοριακό σύστημα αντιμετωπίζει απειλές, *πριν* κάποια από αυτές δημιουργήσει σημαντικές ζημιές. Μία τέτοια τεκμηρίωση είναι πολλαπλώς χρήσιμη, καθώς (α) πείθει τη διοίκηση του οργανισμού-εταιρίας για κατανομή πόρων στα συστήματα ασφάλειας (β) βοηθά στον προσδιορισμό των μέτρων ασφάλειας που είναι πιο κατάλληλα για το σύστημα, καθώς η φύση των απειλών προσδιορίζει σε μεγάλο βαθμό και τα αντίμετρα που πρέπει να εφαρμοσθούν (γ) βοηθά στην πιο αποτελεσματική κατανομή των πόρων ασφάλειας στα διάφορα τμήματα του πληροφοριακού συστήματος, ανάλογα με τις απειλές που το καθένα αντιμετωπίζει και την αξία του για τον οργανισμό.

5. *Έλεγχος ποιότητας για το σχεδιασμό ασφάλειας και τη διαχείριση*. Τόσο το σχέδιο ασφάλειας του οργανισμού όσο και η υλοποίησή του από τους διαχειριστές ασφάλειας και συστημάτων είναι πιθανόν να παρουσιάζουν ατέλειες. Τα συστήματα ανίχνευσης εισβολών μπορούν να καταδείξουν τις ατέλειες, βοηθώντας έτσι στη διόρθωσή τους, πριν αυτές γίνουν αντικείμενο εκμετάλλευσης.

6. Τα συστήματα ανίχνευσης εισβολών μπορούν να παράσχουν πληροφορίες για *επιτυχείς επιθέσεις,* συνεισφέροντας στην αποτίμηση του μεγέθους της ζημιάς, στη διαμόρφωση της λίστας ενεργειών για την ανάκαμψη και στον σχεδιασμό και εφαρμογή προληπτικών μέτρων για μελλοντική αποφυγή αντίστοιχων περιστατικών.

7. *Θωράκιση παλαιών συστημάτων*. Σε αρκετές περιπτώσεις είναι απαραίτητη η διατήρηση σε λειτουργία παλαιών συστημάτων τα οποία δεν υποστηρίζονται πια από τους κατασκευαστές τους και που, ως εκ τούτου, είναι πιο ευάλωτα σε επιθέσεις. Τα πεπαλαιωμένα συστήματα μπορούν να προστατευθούν με τη χρήση συστημάτων ανίχνευσης εισβολών.

8. *Συμπλήρωση των διαδικασιών εγκατάστασης επιδιορθωτικών προγραμμάτων*. Ακόμη και στην περίπτωση που τα συστήματα του οργανισμού υποστηρίζονται από τους κατασκευαστές και έτσι υπάρχουν τα σχετικά επιδιορθωτικά προγράμματα, η διαθεσιμότητα των προγραμμάτων αυτών δεν

είναι πάντα άμεση, ενώ για πολύπλοκα περιβάλλοντα η εγκατάστασή τους μπορεί να καθυστερεί για διάφορους λόγους.

9. *Αναγκαιότητα ύπαρξης ευπαθών υπηρεσιών*. Μολονότι για μερικές υπηρεσίες είναι γνωστό ότι είναι επισφαλείς από την πλευρά της ασφάλειας, οι χρήστες ή η διοίκηση οργανισμών απαιτούν μερικές φορές τη διατήρησή τους διότι θεωρούνται πιο εύχρηστες και άρα πιο παραγωγικές. Τυπικό παράδειγμα είναι η υπηρεσία FTP που σαφώς είναι προβληματική καθώς διακινεί μη κρυπτογραφημένα συνθηματικά, ωστόσο το ασφαλέστερο αντίστοιχο, το ασφαλές πρωτόκολλο μεταφοράς αρχείων, είναι σημαντικά πιο δύσχρηστο. Τα συστήματα ανίχνευσης εισβολών μπορούν να ελέγχουν τις επισφαλείς υπηρεσίες, εντοπίζοντας περιστατικά όπου αυτές προξενούν αυξημένους κινδύνους.

10. *Αξιολόγηση των ενεργειών των χρηστών ή των διαχειριστών*. Οι μηχανισμοί ασφάλειας που παρέχονται από το σύστημα είναι δυνατόν να μην χρησιμοποιούνται σωστά ή αποτελεσματικά από τους χρήστες και τους διαχειριστές. Το σύστημα ανίχνευσης εισβολών μπορεί να επισημαίνει τις σχετικές δυνατότητες βελτίωσης.

11. *Έλεγχος συνέπειας μεταξύ πολιτικής ασφάλειας και κανόνων πρόσβασης*. Η πολιτική ασφάλειας που ισχύει στα πλαίσια του οργανισμού είναι δυνατόν να μην απεικονίζεται πιστά στους κανόνες πρόσβασης που έχουν θεσπίσει οι διαχειριστές . Μέσω αρχείων καταγραφών που τηρούνται από τα συστήματα ανίχνευσης εισβολών είναι δυνατόν να εντοπισθούν οι ασυνέπειες και να διορθωθούν.

10.2 Γενικό μοντέλο για ανίχνευση εισβολών

Προκειμένου να δράσει ένα σχήμα ανίχνευσης εισβολών σε οποιοδήποτε σύστημα, απαιτείται να καθορισθούν οι εξής γενικές παράμετροι:

1. *Πηγές πληροφοριών*. Όπως αναφέρθηκε, τα συστήματα ανίχνευσης εισβολών παρακολουθούν και αναλύουν συμβάντα που λαμβάνουν χώρα στο πληροφοριακό σύστημα, προκειμένου να εντοπίσουν τις προσπάθειες εισβολής. Θα πρέπει έτσι να ορισθούν τα συμβάντα που θα παρακολουθούνται και τα αντίστοιχα συστήματα από τα οποία θα αντλούνται οι πληροφορίες αυτές. Οι πηγές πληροφοριών συνολικά κατατάσσονται σε τρεις κατηγορίες (α) τα δίκτυα και δικτυακά στοιχεία (β) τους υπολογιστές και (γ) τις εφαρμογές.

2. *Τρόποι ανάλυσης πληροφοριών*. Έχοντας συλλέξει τις σχετικές πληροφορίες από τις καθορισθείσες πηγές, το σύστημα ανίχνευσης εισβολών θα πρέπει να τις αξιολογήσει για να συμπεράνει αν τα καταγραφέντα συμβάντα συνιστούν επίθεση. Υπάρχουν δύο τρόποι ανάλυσης των πληροφοριών, η *ανίχνευση καταχρήσεων*, που προσπαθεί να εντοπίσει συμβάντα τα οποία είναι γνωστό ότι εντάσσονται σε διαδικασίες επίθεσης, και η *ανίχνευση ανωμαλιών*, που επιχειρεί να εντοπίσει συμπεριφορές συστημάτων που αποκλίνουν από το «φυσιολογικό».

3. *Αντίδραση*. Η παράμετρος αυτή καθορίζει το πώς θα αντιδράσει το σύστημα όταν διαπιστώσει ότι κάποια προσπάθεια εισβολής είναι εν εξελίξει ή ότι έχει ήδη πραγματοποιηθεί. Δύο διακριτές κατευθύνσεις είναι η *παθητική αντίδραση*, που κυρίως συνίσταται στην ενημέρωση των αρμοδίων και η

ενεργός αντίδραση, η οποία ορίζει ότι το ίδιο το σύστημα ανίχνευσης εισβολών θα προσπαθήσει να αναχαιτίσει την επίθεση.

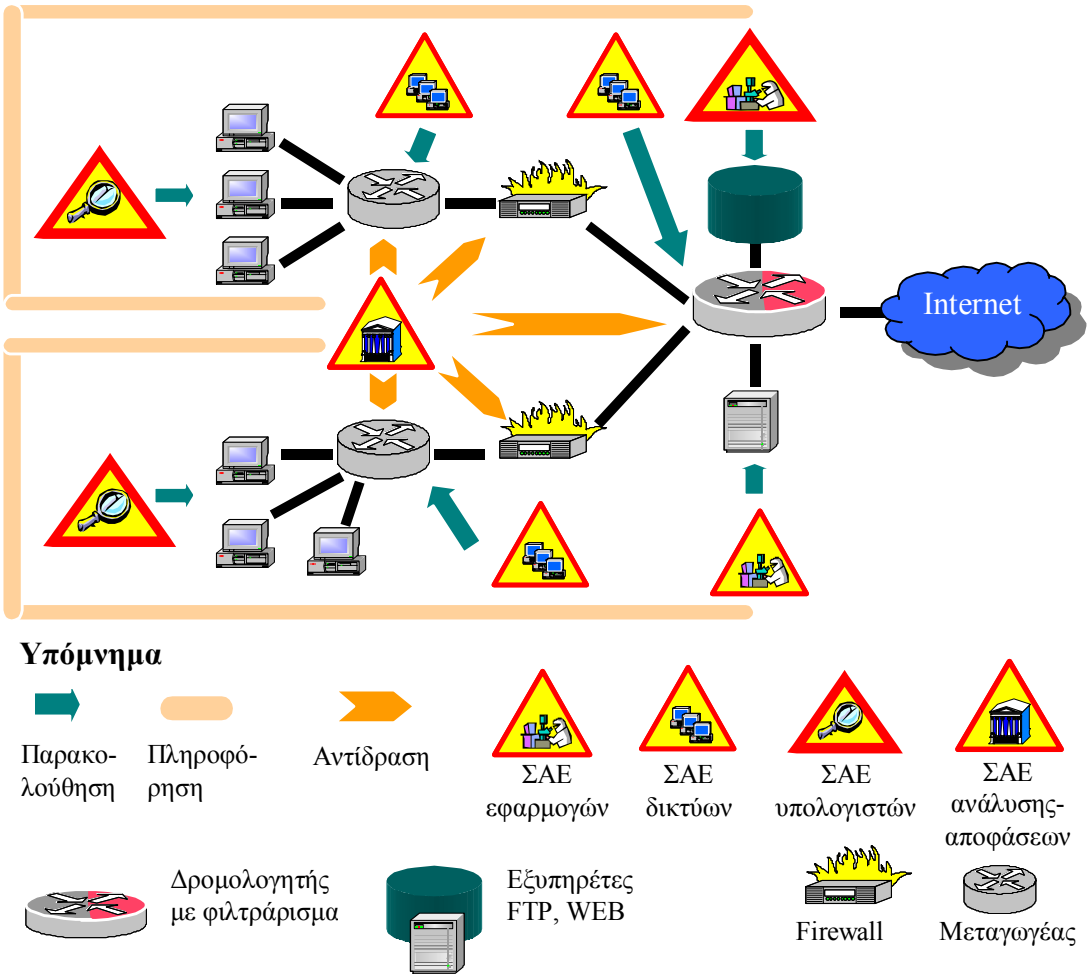
Οι τρεις αυτές γενικές παράμετροι αναλύονται σε επόμενα εδάφια.

10.3 Αρχιτεκτονική συστημάτων ανίχνευσης εισβολών

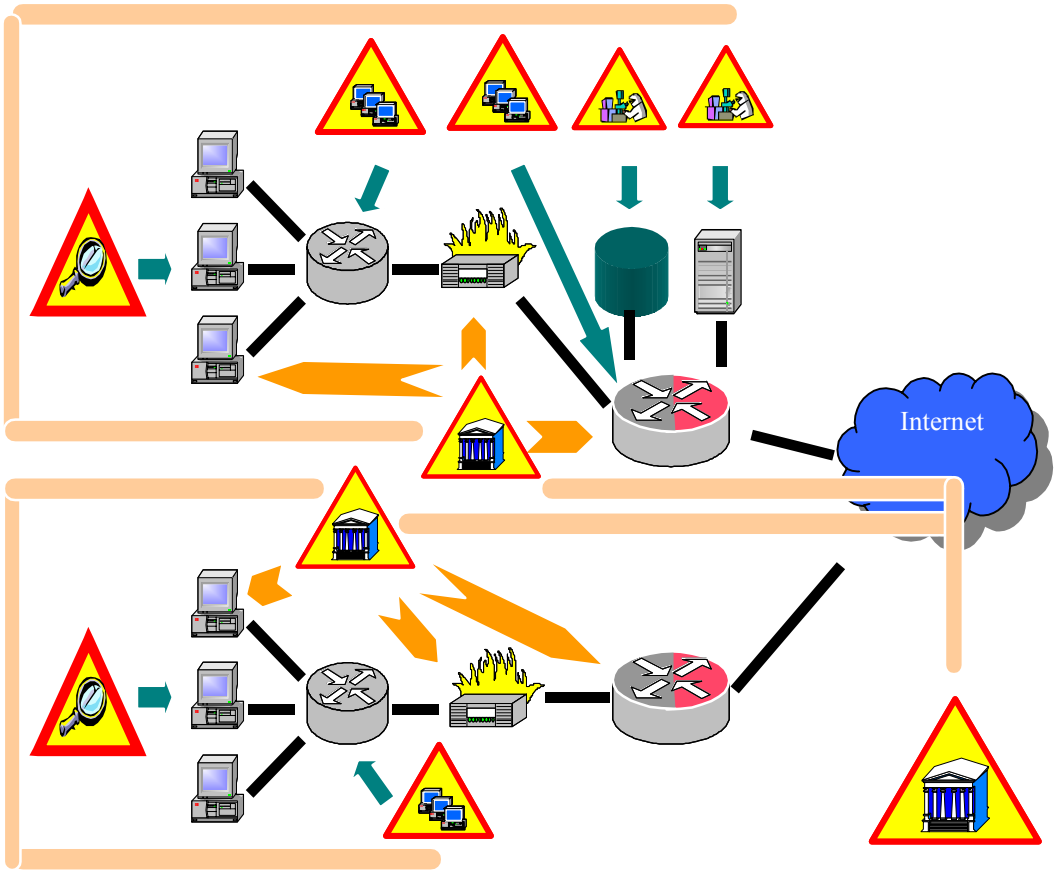
Ένα ιδιαίτερα σημαντικό ζήτημα σε σχέση με την αρχιτεκτονική είναι το αν θα «συστεγάζεται» η παρακολουθούμενη οντότητα και το σύστημα ανίχνευσης εισβολών στην ίδια υπολογιστική πλατφόρμα ή θα χρησιμοποιούνται διακριτά συστήματα. Για παράδειγμα, προκειμένου να παρακολουθούμε ένα σύστημα βάσης δεδομένων θα μπορούσαμε να εγκαταστήσουμε το σύστημα ανίχνευσης εισβολών στον ίδιο τον εξυπηρέτη βάσεων δεδομένων ή σε ένα ξεχωριστό σύστημα. Η συστέγαση παρουσιάζει το ιδιαίτερα σημαντικό πλεονέκτημα ότι έχει σαφώς μικρότερο κόστος, καθώς δεν απαιτεί την αγορά πρόσθετου εξοπλισμού. Αυτό είναι ιδιαίτερα σημαντικό στις περιπτώσεις όπου έχουμε εγκαταστάσεις με μεγάλους υπολογιστές, οι οποίοι είναι εξαιρετικά δαπανηροί. Από την άλλη πλευρά, εγκαθιστώντας το σύστημα ανίχνευσης εισβολών στην ίδια πλατφόρμα με το υπό παρακολούθηση σύστημα μειώνεται η παρεχόμενη ασφάλεια, καθώς αν ο εισβολέας κατορθώσει να «σπάσει» το σύστημα έχει τη δυνατότητα να απενεργοποιήσει συνολικά το σύστημα ανίχνευσης εισβολών. Αντίθετα, αν το σύστημα ανίχνευσης εισβολών είναι εγκατεστημένο σε διακριτό υπολογιστικό σύστημα, ο εισβολέας μπορεί να μην γνωρίζει καν την ύπαρξή του

Πέραν του ζητήματος της συστέγασης ή όχι παρακολουθούμενου και συστήματος ανίχνευσης εισβολών, μία ακόμη βασική παράμετρος της αρχιτεκτονικής είναι η *στρατηγική ελέγχου*, δηλαδή η τοποθέτηση του σημείου όπου αναλύονται τα συμβάντα και λαμβάνονται οι αποφάσεις για τις πιθανές αντιδράσεις.

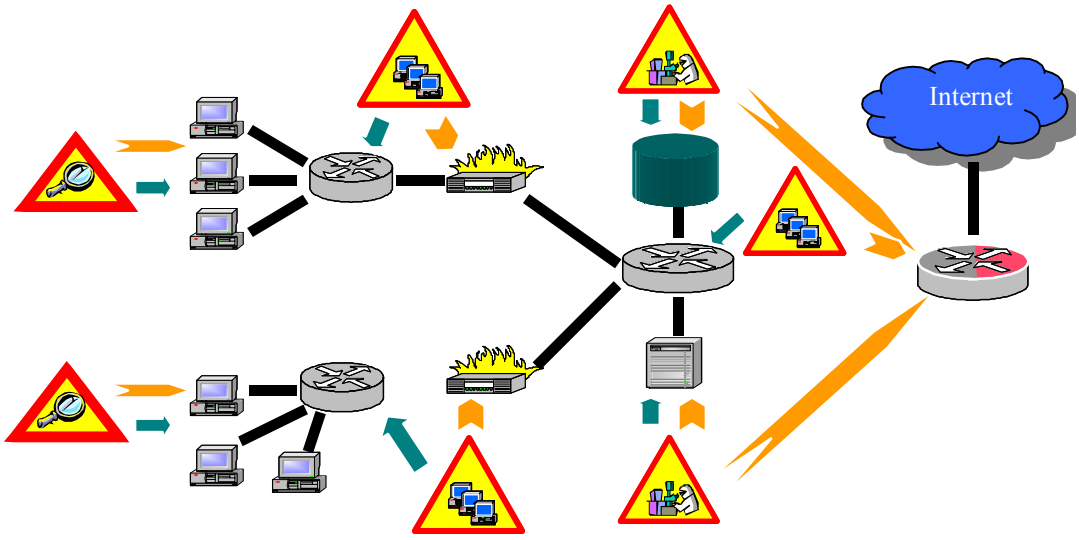
Η πρώτη προσέγγιση σχετικά με τη στρατηγική ελέγχου είναι η *συγκεντρωτική στρατηγική ελέγχου.* Σύμφωνα με τη στρατηγική αυτή, τα συμβάντα που συλλέγονται από την παρακολούθηση προωθούνται σε έναν κεντρικό κόμβο του συστήματος ανίχνευσης εισβολών, ο οποίος μεριμνά για την ανάλυσή τους και την τυχόν λήψη μέτρων. Κατά προτίμηση, η διακίνηση των στοιχείων για τα συμβάντα πρέπει να γίνεται από ξεχωριστό επικοινωνιακό κανάλι από αυτό που διακινούνται τα λειτουργικά δεδομένα του πληροφοριακού συστήματος, ή, αν αυτό κριθεί ιδιαίτερα δαπανηρό, η διακίνηση των στοιχείων για τα συμβάντα πρέπει να είναι κρυπτογραφημένη. Η συγκεντρωτική στρατηγική ελέγχου παρουσιάζεται στο σχήμα που ακολουθεί.



Η δεύτερη προσέγγιση στη στρατηγική ελέγχου είναι ο *ημιαποκεντρωμένος έλεγχος*. Σύμφωνα με την προσέγγιση αυτή, τα παρακολουθούμενα συστήματα κατατμώνται σε διάφορες ζώνες, και σε κάθε ζώνη τοποθετείται ένα ΣΑΕ ανάλυσης-αποφάσεων, το οποίο πληροφορείται σχετικά με τα συμβάντα της ζώνης αυτής, αναλύει τις πληροφορίες και λαμβάνει τις σχετικές αποφάσεις. Είναι επίσης δυνατόν να υπάρχει και ένα «κεντρικό» ΣΑΕ, το οποίο πληροφορείται από τα ΣΑΕ ανάλυσης-αποφάσεων των διαφόρων ζωνών σχετικά με τα πιο αξιοσημείωτα συμβάντα ή για συμβάντα τα οποία πρέπει να αναλυθούν συνδυαστικά για όλες τις ζώνες. Η ημιαποκεντρωμένη στρατηγική ελέγχου παρουσιάζεται στο σχήμα που ακολουθεί.



Η τρίτη προσέγγιση σε σχέση με τη στρατηγική ελέγχου είναι ο πλήρως αποκεντρωμένος έλεγχος. Στην προσέγγιση αυτή δεν υπάρχει ΣΑΕ ανάλυσης-αποφάσεων, αλλά το κάθε ΣΑΕ που συλλέγει τις πληροφορίες είναι επίσης υπεύθυνο για την ανάλυσή τους και τη διαμόρφωση των αντιδράσεων. Η προσέγγιση αυτή απεικονίζεται στο ακόλουθο σχήμα.



10.4 Χρονισμός της ανάλυσης

Σε σχέση με το *πότε* αναλύονται οι πληροφορίες που συλλέγονται από ένα σύστημα ανίχνευσης εισβολών ακολουθούνται γενικά οι εξής δύο πρακτικές:

1. *Ανάλυση σε πραγματικό χρόνο*. Οι συλλεγόμενες πληροφορίες αναλύονται αμέσως μόλις παραληφθούν. Προϋπόθεση για μία τέτοια προσέγγιση είναι δίκτυα με μεγάλες ταχύτητες και ισχυροί υπολογιστές, καθώς η ταχύτητα διακίνησης και ανάλυσης των σχετικών πληροφοριών πρέπει να είναι μεγαλύτερη από την ταχύτητα γέννησής τους. Η προσέγγιση αυτή παρέχει δυνατότητα για άμεση αντίδραση.
2. *Περιοδική ή μαζική ανάλυση*. Οι συλλεγόμενες πληροφορίες αποθηκεύονται σε αρχεία καταγραφής, τα οποία αναλύονται μαζικά σε περιόδους που κρίνονται πιο κατάλληλες, π.χ. περιόδους ελαττωμένου υπολογιστικού ή δικτυακού φορτίου. Η προσέγγιση αυτή είναι ιδιαίτερα βολική όταν η παρακολουθούμενη οντότητα και το σύστημα ανίχνευσης εισβολών φιλοξενούνται στην ίδια υπολογιστική πλατφόρμα, καθώς η ανάλυση σε πραγματικό χρόνο θα είχε επιπτώσεις στις επιδόσεις του συστήματος, από την άλλη πλευρά όμως η προσέγγιση αυτή στερεί τη δυνατότητα άμεσης αντίδρασης.

10.5 Κατάταξη των ΣΑΕ σε σχέση με την πηγή πληροφοριών

Το πιο συνηθισμένο κριτήριο κατάταξης των συστημάτων ανίχνευσης εισβολών είναι η πηγή συλλογής των πληροφοριών. Οι πληροφορίες μπορεί να συλλέγονται από τη δικτυακή κυκλοφορία, από τα συμβάντα ενός υπολογιστή ή τις δραστηριότητες συγκεκριμένων εφαρμογών. Στις παραγράφους που ακολουθούν αναλύεται η κάθε προσέγγιση.

10.5.1 ΣΑΕ με συλλογή πληροφοριών από το δίκτυο

Τα συστήματα ανίχνευσης εισβολών που συλλέγουν την πληροφορία από το δίκτυο προσκολλώνται σε ενεργά (δρομολογητές, μεταγωγείς, επαναλήπτες) ή παθητικά (καλώδια) στοιχεία του δικτύου και συλλέγουν από εκεί τη δικτυακή κυκλοφορία. Τα πακέτα δικτύου αναλύονται βάσει διαφόρων κριτηρίων επικεφαλίδας ή/και περιεχομένου, και από αυτά συνάγονται τα όποια συμπεράσματα σχετικά με πραγματοποίηση εισβολών. Συνολικά, τα ΣΑΕ που συλλέγουν την πληροφορία από το δίκτυο είναι η πολυπληθέστερη κατηγορία.

Στα θετικά στοιχεία της προσέγγισης αυτής συγκαταλέγεται το ότι με κατάλληλη τοποθέτηση έχουν τη δυνατότητα να προστατεύουν πολλούς υπολογιστές. Έτσι, ένα ΣΑΕ της κατηγορίας αυτής που τοποθετείται στον δρομολογητή που συνδέει το δίκτυο της εταιρίας με το διαδίκτυο μπορεί να ανιχνεύσει (θεωρητικά) όλες τις επιθέσεις από το διαδίκτυο προς υπολογιστές του εταιρικού δικτύου. Επίσης, καθώς στην κανονική τους λειτουργία απλώς συλλέγουν πληροφορία, είναι δυνατόν να είναι εντελώς αόρατα για τους εισβολείς, ενώ και για όταν χρειαστεί να παρέμβουν στη δικτυακή κυκλοφορία, μπορούν να χρησιμοποιήσουν τεχνικές απόκρυψης για να μην αποκαλυφθούν.

Η χρήση ΣΑΕ που συλλέγουν την πληροφορία από το δίκτυο εγείρει μερικά ζητήματα:

1. Αν το δίκτυο είναι υπερφορτωμένο, τότε δεν είναι εύκολο ή συνολικά δυνατόν να αναλυθούν όλα τα πακέτα δικτύου. Η υλοποίηση του ΣΑΕ σε υλικό είναι μία λύση, ωστόσο οι υλοποιήσεις ΣΑΕ σε υλικό είναι περιορισμένες και λιγότερο ευέλικτες, σε σχέση με αυτές που υλοποιούνται σε λογισμικό. Η εναλλακτική προσέγγιση είναι να περιοριστεί το πλήθος των ελέγχων που γίνεται από το ΣΑΕ, είτε στοχεύοντας στις *πιο πιθανές* επιθέσεις, είτε στις

επιθέσεις με τις *σημαντικότερες επιπτώσεις*, είτε εξαιρώντας τους ελέγχους για επιθέσεις των οποίων η ανίχνευση είναι *περισσότερο χρονοβόρα*.

2. Αν η δικτυακή επικοινωνία βασίζεται στη *μεταγωγή*, τότε η συλλογή του συνόλου της δικτυακής κυκλοφορίας είναι δύσκολη. Υπάρχουν ενεργά στοιχεία που παρέχουν μία *θύρα παρακολούθησης* (monitoring port), η οποία αναμεταδίδει *όλα τα πακέτα που διέρχονται μέσα από το ενεργό στοιχείο*, αλλά τα συγκεκριμένα μοντέλα είναι σαφώς πιο ακριβά από τα αντίστοιχα που δεν διαθέτουν τέτοια θύρα.
3. Η παρακολούθηση των πακέτων δικτύου δεν μπορεί να εντοπίσει επιθέσεις που γίνονται μέσω κρυπτογραφημένης επικοινωνίας, π.χ. σε πρωτόκολλα ασφαλούς HTTP.
4. Με την παρακολούθηση μόνο των δικτυακών πακέτων είναι εξαιρετικά δύσκολο να υπάρξει συμπέρασμα για το αν η επίθεση πέτυχε το στόχο της.
5. Αρκετές υλοποιήσεις συστημάτων ανίχνευσης εισβολών που συλλέγουν την πληροφορία από το δίκτυο έχουν προβλήματα ευστάθειας με συγκεκριμένες τεχνικές επίθεσης που βασίζονται στη χρήση τμημάτων πακέτων ή σε επιθέσεις που δημιουργούν πολλές συνδέσεις με στόχο την εξάντληση των πόρων.

Ένα πρόσθετο θέμα αρχιτεκτονικού σχεδιασμού για τα συστήματα ανίχνευσης εισβολών που συλλέγουν την πληροφορία από το δίκτυο είναι το *που θα τοποθετηθούν*, εντός της δικτυακής αρχιτεκτονικής ενός εταιρικού δικτύου. Οι πιο διαδεδομένες πρακτικές είναι οι εξής:

1. *Πίσω από το εξωτερικό firewall*. Θεωρώντας δεδομένη την ύπαρξη ενός firewall που παρεμβαίνει μεταξύ του εταιρικού δικτύου και του διαδικτύου, η πρώτη επιλογή είναι να τοποθετείται το σύστημα ανίχνευσης εισβολών μεταξύ του firewall και του εταιρικού δικτύου. Η τοποθέτησή του στη θέση αυτή του δίνει τη δυνατότητα να ανιχνεύει επιθέσεις που ξεπερνούν την περιμετρική άμυνα (δηλαδή το firewall) και να αποκαλύπτει προβλήματα στη διαμόρφωση του firewall (π.χ. πακέτα που τελικά διέρχονται από το firewall ενώ δεν θα έπρεπε). Η συγκεκριμένη τοποθέτηση επιτρέπει επίσης την ανίχνευση επιθέσεων που στοχεύουν στους εξυπηρέτες δημόσιας πρόσβασης του εταιρικού δικτύου, όπως εξυπηρέτες Web και FTP, ενώ μπορεί επίσης να αποκαλύψει και την ύπαρξη επιθέσεων που επέτυχαν, καθώς πιθανότατα θα υπάρξει «εξερχόμενη» κυκλοφορία από το «θύμα» της επίθεσης, η οποία θα ανιχνευθεί από το σύστημα ανίχνευσης εισβολών.
2. *Μπροστά από το εξωτερικό firewall*. Βάσει της προσέγγισης αυτής το σύστημα ανίχνευσης εισβολών τοποθετείται *πριν* το εξωτερικό firewall με κύριο στόχο να ανιχνεύσει και να τεκμηριώσει το σύνολο των επιθέσεων που θα δεχθεί το εταιρικό δίκτυο από το διαδίκτυο. Η δυνατότητα αυτή δεν υπάρχει στην περίπτωση τοποθέτησης πίσω από το firewall, καθώς το firewall θα έχει ήδη «φιλτράρει» αρκετούς τύπους επιθέσεων.
3. *Σε μεγάλους δικτυακούς κόμβους στο εσωτερικό του εταιρικού δικτύου*. Η τοποθέτηση αυτή υιοθετείται όταν κύριος στόχος είναι να ελεγχθούν όσο το δυνατόν περισσότεροι υπολογιστές του εσωτερικού δικτύου. Εποπτεύοντας μεγάλο μέρος της δικτυακής κυκλοφορίας είναι δυνατόν να ανιχνευθεί μεγάλο ποσοστό των πραγματοποιούμενων επιθέσεων, ενώ επίσης υπάρχει και η

δυνατότητα ανίχνευσης επιθέσεων που πραγματοποιούνται από *εσωτερικούς χρήστες*.

4. *Σε υποδίκτυα μεγάλης σημασίας*. Η επιλογή αυτή υιοθετείται συνήθως όταν υπάρχουν υποδίκτυα με μεγάλο βαθμό κρισιμότητας, π.χ. ένα υποδίκτυο που συγκεντρώνει τους εξυπηρέτες εφαρμογών, βάσεων δεδομένων και αρχείων μιας εταιρίας. Με τον τρόπο αυτό προστατεύονται οι πιο πολύτιμοι πόροι, και η προσέγγιση αυτή αποτελεί την πρώτη επιλογή όταν οι οικονομικοί πόροι για αγορά και χρήση ΣΑΕ είναι περιορισμένοι.

10.5.2 ΣΑΕ με συλλογή πληροφοριών από υπολογιστές

Σε αντίθεση με τα ΣΑΕ συλλογής πληροφοριών δικτύου, τα ΣΑΕ συλλογής πληροφοριών από υπολογιστές παρακολουθούν τις δραστηριότητες που λαμβάνουν χώρα σε ένα υπολογιστικό σύστημα προκειμένου να διαπιστώσουν αν υπάρχει προσπάθεια εισβολής. Οι πληροφορίες τους προέρχονται από αρχεία ημερολογίου, διεργασίες, αρχεία συστήματος, μνήμης, στοιχεία κατάστασης, ταυτότητες χρηστών, ή οποιαδήποτε άλλη δυνατότητα αναφοράς παρέχει ο παρακολουθούμενος υπολογιστής.

Τα ΣΑΕ συλλογής πληροφοριών από υπολογιστές έχουν τη δυνατότητα να ανιχνεύουν εισβολές που είτε χρησιμοποιούν «νομότυπα» πακέτα δικτύου, είτε λαμβάνουν χώρα «τοπικά» σε έναν υπολογιστή (δεν ξεκινούν δηλαδή από άλλον υπολογιστή του δικτύου). Μπορούν να χειρισθούν περιπτώσεις κρυπτογραφημένης επικοινωνίας, η ανάλυση ωστόσο πρέπει να γίνει στα μη κρυπτογραφημένα δεδομένα, δηλαδή πριν την κρυπτογράφηση των προς αποστολή δεδομένων και μετά την αποκρυπτογράφηση των λαμβανόμενων δεδομένων. Η χρήση τεχνικών μεταγωγής σε δικτυακό επίπεδο δεν επηρεάζει την κατηγορία αυτή των ΣΑΕ, ενώ με ανάλυση των αρχείων καταγραφής ενεργειών μπορούν να αποκαλύψουν δούρειους ίππους ή άλλες προσπάθειες παραβίασης της ασφάλειας.

Ζητήματα που τίθενται όταν χρησιμοποιούντα ΣΑΕ συλλογής πληροφοριών από υπολογιστές περιλαμβάνουν τα ακόλουθα:

1. η εγκατάσταση και ρύθμιση είναι δυσκολότερη από αυτή των δικτυακών ΣΑΕ, καθώς απαιτείται ξεχωριστή εγκατάσταση για κάθε υπολογιστή και, ενδεχομένως, ξεχωριστές ρυθμίσεις.

2. αν παραβιασθεί η ασφάλεια του παρακολουθούμενου υπολογιστή, ως αποτέλεσμα μιας επιτυχούς επιθέσεως, το ΣΑΕ μπορεί να απενεργοποιηθεί, καθώς ο εισβολέας μπορεί να σταματήσει τη διαδικασία αποστολής στοιχείων στο ΣΑΕ.

3. έχοντας μεμονωμένη εικόνα του κάθε υπολογιστή, είναι δύσκολο να εντοπισθούν επιθέσεις ή προσπάθειες ανίχνευσης σε ένα εταιρικό δίκτυο. Για παράδειγμα, η πραγματοποίηση αιτήσεων σύνδεσης στη θύρα 80 κάθε υπολογιστή ενός εταιρικού δικτύου είναι μία προφανής απόπειρα να εντοπισθούν οι υπολογιστές του εταιρικού δικτύου που προσφέρουν υπηρεσίες WEB, παρ' όλα αυτά ένα ΣΑΕ συλλογής πληροφοριών από υπολογιστές θα έχει εικόνα για *έναν μόνο υπολογιστή* και έτσι δεν θα μπορέσει να εντοπίσει την προσπάθεια ανίχνευσης.

4. Τα λειτουργικά συστήματα δίνουν συνήθως χαμηλούς βαθμούς προτεραιότητας στους μηχανισμούς καταγραφής και αναφοράς (στους οποίους βασίζονται τα ΣΑΕ), και πολλές φορές τους απενεργοποιούν

συνολικά σε συνθήκες υψηλού φόρτου. Είναι έτσι δυνατόν ένας επίδοξος εισβολέας να δημιουργήσει τεχνητά συνθήκες υψηλού φόρτου, με αποτέλεσμα να απενεργοποιηθεί ουσιαστικά το ΣΑΕ, και κατόπιν να εξαπολύσει την επίθεσή του.

5. αν ένα ΣΑΕ συλλογής πληροφοριών από υπολογιστές βασίζεται κυρίως σε αρχεία καταγραφής, μπορεί να υπάρξει πρόβλημα χώρου αποθήκευσης, καθώς ο όγκος των απαιτούμενων στοιχείων μπορεί να είναι ιδιαίτερα σημαντικός.
6. η λειτουργία τους απαιτεί χρήση πόρων του συστήματος, ακόμη και αν το ΣΑΕ δεν συστεγάζεται με το υπολογιστικό σύστημα. Οι ελάχιστοι απαιτούμενοι πόροι είναι αυτοί που θα χρησιμοποιηθούν για την καταγραφή και την αναφορά των συμβάντων που πρέπει να αναλυθούν από το ΣΑΕ.

Σημειώνουμε εδώ ότι ένα ΣΑΕ που παρακολουθεί ένα υπολογιστικό σύστημα δεν είναι απαραίτητο να εκτελείται στο παρακολουθούμενο υπολογιστικό σύστημα: το ΣΑΕ μπορεί να είναι εγκατεστημένο σε άλλο σύστημα και να προσπελαύνει τις πληροφορίες π.χ. μέσω αρχείων που προσαρτά με το πρωτόκολλο NFS, ή να τις λαμβάνει από το δίκτυο μέσω του πρωτοκόλλου SNMP.

Αν μία εταιρία επιλέξει την τεχνική των ΣΑΕ συλλογής πληροφοριών από υπολογιστές για την προστασία του εταιρικού της υπολογιστικού περιβάλλοντος είναι σκόπιμο να ξεκινήσει την εγκατάσταση πρώτα από τους κρίσιμους εξυπηρέτες της και κατόπιν να προχωρήσει στους υπόλοιπους υπολογιστές. Αυτό θα δώσει και στο προσωπικό ασφάλειας τη δυνατότητα να εξοικειωθεί με το σύστημα όταν θα έχει εγκατασταθεί σε λίγους υπολογιστές, πριν κληθεί να διαχειρισθεί το σύστημα σε μεγάλη κλίμακα. Όταν ΣΑΕ συλλογής πληροφοριών από υπολογιστές πρόκειται να εγκατασταθούν σε μεγάλο αριθμό υπολογιστών, αφ' ενός θα πρέπει να είναι της ίδιας τεχνολογίας, αφ΄ ετέρου δε θα πρέπει να χρησιμοποιηθεί ΣΑΕ με αναφορά σε ένα κεντρικό σύστημα ανάλυσης-αποφάσεων, καθώς είναι πρακτικώς αδύνατο οι διαχειριστές να επισκέπτονται ξεχωριστά τα μηχανήματα και να ελέγχουν αν υπάρχει κάποιο πρόβλημα.

10.5.3 ΣΑΕ με συλλογή πληροφοριών από εφαρμογές

Τα ΣΑΕ συλλογής πληροφοριών από εφαρμογές αποτελούν μία ειδική περίπτωση των ΣΑΕ συλλογής πληροφοριών από υπολογιστές. Συνήθως παρακολουθούν τα αρχεία καταγραφής, τα ημερολόγια δοσοληψιών ή την επικοινωνία της εφαρμογής και έχοντας αυξημένη γνώση για τη συγκεκριμένη εφαρμογή, μπορούν να διαγνώσουν μεγάλο πλήθος παραβιάσεων που θα ήταν αδύνατο σε λιγότερο «εξειδικευμένα» ΣΑΕ να διαγνώσουν. Τέτοιου είδους ΣΑΕ μπορούν να διαγνώσουν αρκετά αποτελεσματικά περιπτώσεις κατάχρησης δικαιωμάτων. Η λειτουργία τους δεν επηρεάζεται από τη χρήση κρυπτογράφησης στην επικοινωνία, καθώς είναι τοποθετημένα σε σημείο όπου και η μη κρυπτογραφημένη μορφή των πληροφοριών είναι διαθέσιμη.

Από την άλλη πλευρά, η χρήση των ΣΑΕ συλλογής πληροφοριών από εφαρμογές πρέπει οπωσδήποτε να συμπληρώνεται από πιο γενικά ΣΑΕ συλλογής πληροφοριών από υπολογιστές ή από το δίκτυο, προκειμένου να προφυλάσσεται και το υπολογιστικό σύστημα που φιλοξενεί την εφαρμογή, πέρα από την ίδια την εφαρμογή. Επίσης οι διαχειριστές πρέπει να ρυθμίσουν σωστά την ασφάλεια του περιβάλλοντος, καθώς τα αρχεία καταγραφής των εφαρμογών είναι συνήθως πλημμελέστερα προστατευμένα, σε σχέση με αυτά του Λ.Σ. και έτσι οι επιτιθέμενοι

μπορεί να τα αλλοιώσουν πριν το ΣΑΕ αξιοποιήσει τις πληροφορίες που κατεγράφησαν σε αυτά.

10.6 Τεχνικές ανάλυσης συμβάντων

Από τη στιγμή που το σύστημα ανίχνευσης εισβολών έχει στη διάθεσή του τα απαραίτητα στοιχεία, τα οποία μπορούν να έχουν συλλεχθεί από το δίκτυο, από υπολογιστές ή από συγκεκριμένες εφαρμογές, θα προχωρήσεις στην ανάλυση και αξιολόγησή τους, προκειμένου να συνάγει αν τα ευρήματα στοιχειοθετούν προσπάθειες εισβολής. Για τη διαδικασία αυτή υπάρχουν δύο γενικές κατευθύνσεις, η *ανίχνευση καταχρήσεων* και η *ανίχνευση ανωμαλιών*. Οι δύο προσεγγίσεις αυτές παρουσιάζονται στις ακόλουθες παραγράφους.

10.6.1 Ανίχνευση καταχρήσεων

Η ανίχνευση καταχρήσεων είναι η τεχνική που χρησιμοποιείται από τα περισσότερα συστήματα ανίχνευσης εισβολών. Βάσει της προσέγγισης αυτής, οι συλλεχθείσες πληροφορίες εξετάζονται για τον εντοπισμό συμβάντων ή συνόλων συμβάντων που αντιστοιχούν σε γνωστές επιθέσεις. Ως παράδειγμα μπορούμε να αναφέρουμε την ύπαρξη καταχωρήσεων της μορφής

```
GET ../..
GET http://www.domain.com/scripts\..\..\scriptname
```

σε συστήματα που προσφέρουν υπηρεσίες Web. Οι καταχωρήσεις αυτές καταδεικνύουν ότι έχουν υποβληθεί στον εξυπηρέτη Web αιτήσεις που είναι γνωστό ότι προσπαθούν να αξιοποιήσουν κενά ασφαλείας του εξυπηρέτη IIS. Επίσης η ακολουθία συμβάντων που περιλαμβάνει τα κάτωθι

```
CONNECT (FROM_IP = a.b.c.d, FROM_PORT = 2020,
         DEST_IP = w.x.y.z, DEST_PORT = 0)
CONNECT (FROM_IP = a.b.c.d, FROM_PORT = 2020,
         DEST_IP = w.x.y.z, DEST_PORT = 1)
CONNECT (FROM_IP = a.b.c.d, FROM_PORT = 2020,
         DEST_IP = w.x.y.z, DEST_PORT = 2)
```

αναδεικνύει μία προσπάθεια να σαρωθούν σειριακά οι θύρες εξυπηρέτησης δικτύου του μηχανήματος με διεύθυνση IP `w.x.y.z`, πιθανώς στα πλαίσια μιας προετοιμασίας για επίθεση.

Κατ' αναλογία με τις τεχνικές ανίχνευσης ιών, και η τεχνική ανίχνευσης καταχρήσεων στοχεύει στον εντοπισμό γνωστών *υπογραφών επιθέσεων*. Οι απλούστερες τεχνικές περιλαμβάνουν μόνο ένα συμβάν σε αυτές τις υπογραφές, ενώ οι πιο εξελιγμένες εξετάζουν και στοιχεία *κατάστασης*, δηλαδή τα συμβάντα που έχουν λάβει χώρα προηγουμένως και την κατάσταση που αυτά έχουν φέρει το σύστημα.

Η τεχνική της ανίχνευσης καταχρήσεων είναι πολύ αποτελεσματική τεχνική για ανίχνευση επιθέσεων, η οποία μάλιστα δεν παράγει πολλές ψευδείς αναφορές επιθέσεων. Έχει τη δυνατότητα να ανιχνεύσει έγκαιρα συγκεκριμένες επιθέσεις, ενδεχομένως και τα εργαλεία που χρησιμοποιούνται σ' αυτές, ώστε να θωρακιστεί το σύστημα, ενώ είναι κατάλληλη και για διαχειριστές χωρίς ιδιαίτερες τεχνικές γνώσεις. Από την άλλη πλευρά, τα εργαλεία που βασίζονται στην τεχνική της ανίχνευσης καταχρήσεων ανιχνεύουν μόνο τις επιθέσεις για τις οποίες γνωρίζουν (δηλαδή υπάρχουν στοιχεία στη βάση δεδομένων τους που αφορά τις γνωστές επιθέσεις), καθιστώντας έτσι απαραίτητη την τακτική ενημέρωση της βάσης

δεδομένων αυτής με στοιχεία για νέες επιθέσεις. Επίσης, τα περισσότερα εργαλεία δεν ανιχνεύουν παραλλαγές γνωστών επιθέσεων, κάτι που έχει σαφείς επιπτώσεις στην αποτελεσματικότητά τους.

10.6.2 Ανίχνευση ανωμαλιών

Η τεχνική της ανίχνευσης ανωμαλιών βασίζεται στη γενική υπόθεση ότι το σύστημα έχει κάποια *συγκεκριμένη συμπεριφορά* όταν βρίσκεται υπό κανονική χρήση, η οποία *διαφοροποιείται* όταν το σύστημα βρίσκεται υπό επίθεση. Για παράδειγμα, υπό «κανονικές» συνθήκες ο εξυπηρέτης ηλεκτρονικού ταχυδρομείου μπορεί να στέλνει από 0 έως 10 μηνύματα το λεπτό· αν όμως χρησιμοποιηθεί για διακίνηση διαφημιστικής ηλεκτρονικής αλληλογραφίας (spam mail), τότε η κυκλοφορία αυτή θα αυξηθεί σημαντικά. Η αύξηση της κυκλοφορίας υποδηλώνει την εισβολή που δέχεται το σύστημα.

Προκειμένου να είναι δυνατή η ανίχνευση των ανωμαλιών, είναι απαραίτητο να δημιουργηθούν *κωδικοποιήσεις της κανονικής συμπεριφοράς*. Οι κωδικοποιήσεις αυτές παράγονται από ιστορικά στοιχεία που συλλέγονται σε κάποια χρονική περίοδο και που μπορεί να περιλαμβάνουν δεδομένα για χρήστες, υπολογιστές ή δικτυακές συνδέσεις. Έχοντας κατόπιν τις κωδικοποιήσεις κανονικής συμπεριφοράς στη διάθεσή του, το σύστημα ανίχνευσης εισβολών παρατηρεί τη συμπεριφορά των αντίστοιχων οντοτήτων και συγκρίνει την παρατηρούμενη συμπεριφορά με αυτή που περιλαμβάνεται στις κωδικοποιήσεις. Οι τυχόν αποκλίσεις που θα εντοπισθούν, εκλαμβάνονται ως ενδείξεις εισβολής.

Οι πιο συχνά χρησιμοποιούμενες κωδικοποιήσεις κανονικής συμπεριφοράς περιλαμβάνουν τα εξής:

1. συγκεκριμένα χαρακτηριστικά της συμπεριφοράς χρηστών και του συστήματος, τα οποία εκφράζονται ως πληθάριθμοι. Τα χαρακτηριστικά περιλαμβάνουν π.χ. το πλήθος αρχείων που προσπελαύνονται σε χρονικό διάστημα, το πλήθος σφαλμάτων σύνδεσης (login) για κάθε χρήστη, τον φόρτο της ΚΜΕ για συγκεκριμένη διεργασία κ.λπ. Στα χαρακτηριστικά αυτά αντιστοιχίζονται και *περιθώρια σφάλματος*, καθώς η ιδιαιτερότητα κάποιων εργασιών μπορεί να οδηγήσει τους χρήστες ή το σύστημα σε –περιορισμένες– αποκλίσεις από την «κανονική» τους συμπεριφορά. Τέλος, τα όρια που τίθενται στους πληθάριθμους μπορεί να είναι *στατικά ή δυναμικά*. Τα στατικά όρια προσδιορίζονται μία φορά και δεν αλλάζουν, παρά μόνο με εκκίνηση συγκεκριμένης διαδικασίας επανυπολογισμού τους, ενώ στην περίπτωση των δυναμικών ορίων το σύστημα προσαρμόζει μόνο του τα όρια, όταν παρατηρεί μία συγκεκριμένη πορεία στα παρατηρούμενα μεγέθη. Για παράδειγμα, αν το άνω όριο για τον πληθάριθμο μηνυμάτων ηλεκτρονικού ταχυδρομείου που αποστέλλει ένας χρήστης 120 και στη διάρκεια τριών συνεχών εβδομάδων ο χρήστης αποστέλλει 112, 116 και 119 μηνύματα, αντίστοιχα, το σύστημα θα μπορούσε να προσαρμόσει το άνω αυτό όριο σε 140, ούτως ώστε να αποφευχθεί η αναφορά προβλήματος για μία κατάσταση που είναι «προβλέψιμα φυσιολογική».
2. στατιστικά μεγέθη (μέσοι όροι, τυπικές αποκλίσεις κ.λπ.) που μπορεί να εκφράζονται παραμετρικά –να υποτίθεται δηλαδή ότι το μέγεθος ακολουθεί μία συγκεκριμένη κατανομή– ή μη παραμετρικά, στην οποία περίπτωση το μέγεθος συνάγεται από ιστορικά στοιχεία.

3. Στοιχεία βασιζόμενα σε κανόνες που ορίζουν αποδεκτές συμπεριφορές αλλά με *ποιοτικό και όχι ποσοτικό* τρόπο. Για παράδειγμα, το σύστημα θα μπορούσε να γνωρίζει τους καταλόγους εγγράφων που χρησιμοποιεί ένας χρήστης και αν διαπιστώσει ότι σε κάποια χρονική στιγμή ο χρήστης προσπελαύνει άλλους καταλόγους, αυτό μπορεί να θεωρηθεί ως ένδειξη εισβολής.
4. Νέες τάσεις για την ανάλυση των δεδομένων στα πλαίσια ανίχνευσης ανωμαλιών περιλαμβάνουν γενετικούς αλγόριθμους, νευρωνικά δίκτυα ή άλλες προηγμένες τεχνικές. Οι μέθοδοι αυτοί ωστόσο είναι ακόμη σε πειραματικό στάδιο και δεν χρησιμοποιούνται από εμπορικά διαθέσιμα συστήματα.

Οι πιο διαδεδομένες μέθοδοι ανάλυσης δεδομένων είναι οι (1) και (2). Αποτιμώντας την τεχνική της ανίχνευσης ανωμαλιών, μπορούμε να σημειώσουμε ότι έχει τη δυνατότητα να ανιχνεύσει νέους τύπους επιθέσεων, στο βαθμό που αυτές θα προκαλέσουν αποκλίσεις τις κωδικοποιήσεις «κανονικής» συμπεριφοράς, ενώ έχουν επίσης τη δυνατότητα να παράγουν κανόνες που θα τροφοδοτούν τα συστήματα ανίχνευσης καταχρήσεων. Από την άλλη πλευρά όμως τείνουν να δημιουργούν ψευδείς αναφορές εισβολής, ενώ για τη δημιουργία των κωδικοποιήσεων «κανονικής συμπεριφοράς» απαιτούνται εκτεταμένα στοιχεία, συλλεγμένα από μακρά περίοδο λειτουργίας του συστήματος.

10.7 Αντιδράσεις των συστημάτων ανίχνευσης εισβολών

Οι αντιδράσεις των συστημάτων ανίχνευσης εισβολών διακρίνονται γενικά σε δύο κατηγορίες, τις *ενεργές αντιδράσεις* και τις *παθητικές αντιδράσεις*.

10.7.1 Ενεργές αντιδράσεις

Στις ενεργές αντιδράσεις το σύστημα προσπαθεί να λάβει κάποια μέτρα για να τεκμηριώσει καλύτερα ή να αναχαιτίσει την επίθεση. Προς την κατεύθυνση αυτή το σύστημα ανίχνευσης εισβολών μπορεί να προβεί σε μία ή περισσότερες από τις ακόλουθες ενέργειες:

1. *Συλλογή περισσοτέρων πληροφοριών,* με κύριο στόχο την καλύτερη αξιολόγηση της επίθεσης ή/και τη συλλογή στοιχείων για νομικές ενέργειες. Προς την κατεύθυνση αυτή μπορεί να αυξηθεί η ευαισθησία των «αισθητήρων» π.χ. αρχείων καταγραφής, πακέτων δικτύου που αναλύονται κ.λπ. ή να υπάρξουν «ερωτήσεις» προς το σύστημα από το οποίο εκπορεύεται η επίθεση για να διαπιστωθεί ποιοι χρήστες είναι συνδεδεμένοι κ.ά
2. *Τροποποίηση περιβάλλοντος*. Η κατεύθυνση αυτή αποσκοπεί στο να οδηγήσει την επίθεση σε αποτυχία. Αυτό μπορεί να επιτευχθεί με αποστολή προς τον επιτιθέμενο πακέτων τερματισμού σύνδεσης που να φαίνεται ότι προέρχονται από το υπό επίθεση σύστημα, με επαναρύθμιση firewalls και δρομολογητών και υπηρεσιών εισάγοντας απαγορεύσεις για διευθύνσεις IP, θυρών, δικτυακών πρωτοκόλλων, υπηρεσιών ή φυσικών συνδέσεων.
3. *Αντεπίθεση*, η οποία συνίσταται σε χρήση τεχνικών για αδρανοποίηση του επιτιθέμενου ή συλλογή πληροφοριών για αυτόν. Θα μπορούσε έτσι να υπάρξει καταιγισμός δικτυακών πακέτων προς το σύστημα απ' όπου φαίνεται να ξεκινά η επίθεση, ή εξαπόλυση επιθέσεων προς υπηρεσίες που αυτός προσφέρει. Η αντεπίθεση δεν είναι πρακτική που πρέπει να εφαρμόζεται στη γενική περίπτωση, καθώς μπορεί να έχει νομικές επιπτώσεις (η αυτοδικία δεν

θεωρείται νόμιμη ενέργεια) και μπορεί επίσης να «θυμώσει» τους εισβολείς, με συνέπεια να εξαπολύσουν πιο «σκληρές» επιθέσεις. Είναι τέλος πιθανόν μία αντεπίθεση να έχει ως αποτέλεσμα να «χτυπηθούν» αθώοι, καθώς σε δημόσια δίκτυα (π.χ. δίκτυα IP) δεν υπάρχει ισχυρή διακρίβωση της ταυτότητας προέλευσης των δικτυακών πακέτων, και έτσι αυτή μπορεί να έχει πλαστογραφηθεί. Σε μία περίπτωση πλαστογραφίας της ταυτότητας προέλευσης, στη διάρκεια της αντεπίθεσης θα «χτυπηθεί» το σύστημα που φαίνεται στην ταυτότητα προέλευσης των δικτυακών πακέτων, το οποίο όμως δεν θα είναι το σύστημα από το οποίο προέρχεται η επίθεση. Αν πρόκειται σε οποιαδήποτε περίπτωση να χρησιμοποιηθεί τεχνική αντεπίθεσης, αυτή πρέπει να γίνει υπό την εποπτεία ειδικών.

10.7.2 Παθητικές αντιδράσεις

Οι παθητικές αντιδράσεις συνίστανται κυρίως σε ειδοποιήσεις και συναγερμούς για το προσωπικό ασφάλειας. Οι ειδοποιήσεις αυτές μπορούν να έχουν κυμαινόμενο βαθμό λεπτομέρειας και μπορούν να εμφανίζονται σε ειδικό χώρο του συστήματος ανίχνευσης εισβολών, σε παράθυρο μηνύματος, σε συσκευές τηλεειδοποίησης, με μηνύματα σε κινητά κ.ά. Μολονότι και το ηλεκτρονικό ταχυδρομείο θα μπορούσε να χρησιμοποιηθεί για τέτοιου ειδοποιήσεις αυτής της μορφής, είναι επισφαλές να βασισθεί κανείς σ' αυτό καθώς ο εισβολέας ενδέχεται να «μπλοκάρει» την αποστολή μηνυμάτων.

Για την αναφορά των προβλημάτων μπορεί να χρησιμοποιηθεί και το πρωτόκολλο SNMP, το οποίο είναι ένα ευρέως διαδεδομένο στάνταρ. Η προσέγγιση αυτή έχει το πλεονέκτημα ότι παρέχει τη δυνατότητα ολοκλήρωσης με συστήματα διαχείρισης δικτύου, αξιοποιώντας περαιτέρω μια δαπανηρή υποδομή (λογισμικού και επικοινωνίας), και ολοκληρώνοντας τις λειτουργίες διαχείρισης.

Συνολικά, η παθητικές αντιδράσεις είναι λιγότερο απαιτητικές σε πόρους από τις ενεργές αντιδράσεις.

10.8 Η «αυτοάμυνα» των συστημάτων ανίχνευσης εισβολών

Σε πολλές περιστάσεις το ίδιο το σύστημα ανίχνευσης εισβολών μπορεί να αποτελέσει στόχο επιθέσεων με στόχο την ανίχνευση, την παράκαμψη ή την αχρήστευσή του. Θα πρέπει έτσι να λαμβάνονται μέτρα ώστε το ίδιο το Σ.Α.Ε. να μην γίνεται στόχος ή/και να μπορεί να αποκρούει τέτοιου είδους επιθέσεις. Στα μέτρα αυτά μπορούν να εντάσσονται:

1. η αποφυγή της κοινοποίησης της παρουσίας του Σ.Α.Ε. με δικτυακά μηνύματα, ακόμη και σε περιπτώσεις συναγερμού. Αν είναι απολύτως απαραίτητο να εκπεμφθεί μήνυμα, είναι σκόπιμο να εκπέμπεται με πλαστή δικτυακή διεύθυνση.
2. Συνολικά είναι καλό τα στοιχεία που συλλέγονται από το σύστημα ανίχνευσης εισβολών να διακινούνται από ξεχωριστά κανάλια επικοινωνίας, προκειμένου να μην εντοπίζεται η κυκλοφορία αυτή από τους πιθανούς εισβολής. Αν αυτό είναι αδύνατον, επιβάλλεται η χρήση κρυπτογράφησης και ισχυρών μηχανισμών διακρίβωσης ταυτότητας. Η κρυπτογράφηση προστατεύει τα διακινούμενα στοιχεία από το να αποκαλυφθούν στους εισβολείς, ενώ η ισχυρή διακρίβωση ταυτότητας αποτρέπει τους εισβολείς από το να αποστείλουν πλαστογραφημένα στοιχεία στο σύστημα ανίχνευσης εισβολών με στόχο την παραπλάνησή του.

3. Το ίδιο το σύστημα ανίχνευσης εισβολών δεν πρέπει να παρέχει δικτυακώς προσπελάσιμες υπηρεσίες, όπως απομακρυσμένης σύνδεσης, ηλεκτρονικού ταχυδρομείου κ.λπ., καθώς αυτές αφ' ενός θα αποκαλύψουν την ύπαρξή του, αφ' ετέρου μπορούν να αξιοποιηθούν από τους εισβολείς σε επιθέσεις εναντίον του συστήματος ανίχνευσης εισβολών.