ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ - ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Κατανεμημένα Συστήματα, Ασφάλεια και Αναδυόμενες Τεχνολογίες Πληροφορίας»

Μεταπτυχιακή Διατριβή

Τίτλος διατριβής	**Επιθέσεις και εργαλεία εντοπισμού ευπαθειών σε κατανεμημένα συστήματα** **Security attacks and vulnerability detection tools for distributed systems**
Όνοματεπώνυμο Φοιτητή	Ξενοφών Ζηνοβίου
Πατρώνυμο	Χαράλαμπος
Αριθμός Μητρώου	ΜΠΚΣΑ 19009
Επιβλέπων	**Παναγιώτης Κοτζανικολάου, Αναπληρωτής Καθηγητής**

Ημερομηνία Παράδοσης **Δεκέμβριος 2023**

Τριμελής Εξεταστική Επιτροπή

Παναγιώτης Κοτζανικολάου	Κωνσταντίνος Πατσάκης	Μιχαήλ Ψαράκης
Αναπληρωτής Καθηγητής	Αναπληρωτής Καθηγητής	Αναπληρωτής Καθηγητής

Περίληψη

Τα υπολογιστικά συστήματα με την πάροδο των ετών και της τεχνολογίας έχουν περάσει απο διάφορα στάδια αλλαγών. Απο απλά μονοδιάστατα συστήματα για εξυπηρέτηση ενός χρήστη με συγκεκριμένες απαιτήσεις και πεπερασμένους πόρους, είμαστε σήμερα σε θέση να έχουμε στη διάθεση μας κατανεμημένα υπολογιστικά συστήματα. Συστήματα τα οποία πια μπορεί να είναι μια συστοιχία είτε ανομοιογενών είτε ομοιογενών υπολογιστικών συστημάτων που αλληλεπιδρούν μεταξύ τους σαν μια οντότητα. Μια οντότητα με δυνατότητα διαμοιρασμού των πόρων της και σχεδόν ανεξάντλητες δυνατότητες αναβάθμισης και κλιμάκωσης, εξυπηρετώντας παράλληλα πολύ μεγάλο αριθμό χρηστών και ροές όγκων δεδομένων που παλαιότερα θα φάνταζαν ασύλληπτες. Η εξέλιξη αυτή οδήγησε παράλληλα σε πολύ μεγάλες αλλαγές στην αρχιτεκτονική, στο σχεδιασμό και την υλοποίηση των κατανεμένων συστημάτων με αποτέλεσμα να αυξηθεί σημαντικά η επιφάνεια επίθεσης και έκθεσης σε νέους και μεγαλύτερους κινδύνους. Η μελέτη αυτή αποτελεί μια παρουσίαση των κατανεμημένων συστημάτων και των προκλήσεων απο την πλευρά της ασφαλείας ενός τέτοιου συστήματος.

Abstract

Computing systems have evolved during the last decades. They are not simply just a single computer with limited resources destined to serve only one specific user. Nowadays instead, we have reached the point where complex clusters of multiple computers of the same characteristics, or even not, can be combined and form a distributed system that resembles a single entity but with endless capabilities. They have the power to share and distribute their resources to whichever of their machine is needed, they can split the workload and work faster and more effective and then retrieve all the resources back so they can be reallocated to other machines, serving this way thousands of users and managing millions of data. This development has led to system architectural and design changes and as a result to the implementation of such systems. As a result the attack surface area has been increased and their exposure to new and sophisticated attacks. This study attempts to present the distributed systems and their challenges from a security point of view.

Ευχαριστίες

Θέλω να ευχαριστήσω τον καθηγητή κο Παναγιώτη Κοτζανικολάου για την κατανόηση, το ενδιαφέρον και την πάντα άμεση του ανταπόκριση, καθώς και τον υποψήφιο διδάκτορα Βαγγέλη Μάλαμα για την πολύτιμη βοήθεια του.

Τέλος, θα ήθελα να ευχαριστήσω τη σύζυγο μου Βιβή για την αμμέριστη υπομονή και συμπαράσταση που έχει επιδείξει κατά τη διάρκεια των σπουδών μου.

Περιεχόμενα

Πίνακας εικόνων

Εισαγωγή

Τα κατανεμημένα συστήματα αποτελούν σήμερα ένα τύπο συστήματος-μονόδρομο για υποδομές όπως αυτές των παρόχων υπηρεσιών υπολογιστικού νέφους. Καθώς ο όγκος των δεδομένων και των χρηστών είναι πολύ μεγάλος η διαχείριση και εξυπηρέτηση τους απαιτεί συστήματα που να επιδεικνύουν σημαντικά χαρακτηριστικά προσαρμογής σε τέτοιες απαιτήσεις: Γρήγορη εξυπηρέτηση, μηδενική απώλεια παροχής υπηρεσίας, αύξηση πόρων όταν οι περιστάσεις το απαιτούν και αντίστοιχος διαμοιρασμός τους με βάση τη ζήτηση. Ο σχεδιασμός και η αρχιτεκτονική τέτοιων συστημάτων μαζί με την υλοποίηση τους αποτελεί μια πρόκληση για τους ανθρώπους πίσω από τέτοια έργα.

Μαζί όμως με τα χαρακτηριστικά αυτά δημιουργούνται και σοβαρές προκλήσεις σε επίπεδο ασφάλειας. Καθώς δεν υπάρχει πια μια μοναδική υπολογιστική οντότητα με πιο προβλέψιμα σημεία που χρήζουν θωράκισης αλλά μια συστοιχία υπολογιστικών συστημάτων , συχνά ανομοιογενής, με τέτοια αλληλεπίδραση μεταξύ τους ώστε ο εκάστοτε χρήστης να την αντιλαμβάνεται ως ένα υπολογιστικό σύστημα. Σε αυτή την περίπτωση η επιφάνεια επίθεσης αυξάνεται κατά πολύ καθώς έχουμε πολλά υπολογιστικά συστήματα που αλληλεπιδρούν μεταξύ τους άρα το καθένα έχει τα δικά του σημεία και ιδιαιτερότητες που πρέπει να θωρακιστούν και παράλληλα η επικοινωνία τους γίνεται μέσω δικτύου με εντελώς διαφορετικά σημεία κινδύνου. Οι προκλήσεις ασφαλείας ενός τέτοιου συστήματος, όπως γίνεται αντιληπτό, είναι πολλές, σημαντικές και πολύπλοκες και σε συνδυασμό με τις σύγχρονες και εξεζητημένες τεχνικές επιθέσεων που εφαρμόζονται δημιουργούν ένα πολύπλοκο γρίφο προς επίλυση.

Οι προκλήσεις αφορούν κατά βάση: την αυθεντικοποίηση και τη σωστή εξουσιοδότηση των χρηστών, την ασφαλή διαχείριση και αποθήκευση των δεδομένων, την ασφαλή επικοινωνία και μεταφορά των δεδομένων ανάμεσα στον χρήστη και το σύστημα και φυσικά την ίδια τη φυσική ασφάλεια της υποδομής του συστήματος.

Η μελέτη αυτή δεν θα ασχοληθεί με την ασφάλεια των φυσικών εγκαταστάσεων και της υποδομής ενός κατανεμημένου συστήματος, αλλά θα επιχειρήσει να παρουσιάσει τις προκλήσεις της ασφάλειας στο επίπεδο της πληροφορίας: πως γίνεται η διαχείριση η καταγραφή και η αποθηκευσή της και πως επικοινωνεί το σύστημα. Πως τα δίκτυα επικοινωνίας και οι χρήστες αλληλεπιδρούν με το σύστημα, ποιά η σημαντικότητα τους, ποιοί κίνδυνοι υπάρχουν, πως μπορούν να γίνουν αντικείμενο εκμετάλλευσης και ποιοί οι τρόποι αντιμετώπισης τους.

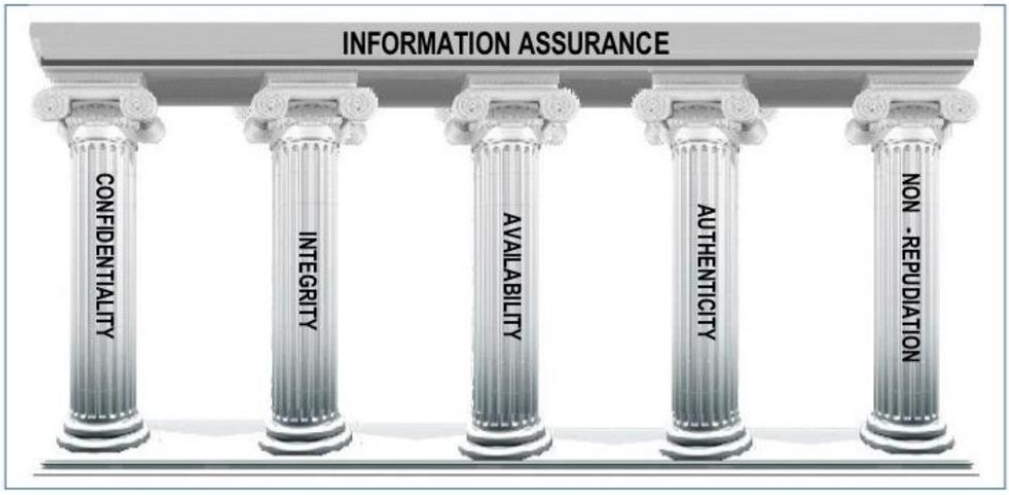
1. Βασικοί Όρισμοί

1.1 Κατανεμημένο Σύστημα

Με τον όρο κατανεμημένο σύστημα εννοούμε ένα σύνολο ανεξάρτητων υπολογιστικών μονάδων (υπολογιστών) για το οποίο οι χρήστες έχουν την εικόνα ότι είναι μια οντότητα, ένα υπολογιστικό σύστημα [1], [2].

1.2 Ασφάλεια

Αρχικά θα πρέπει να αποσαφηνιστεί τι εννοούμε με τον όρο ασφάλεια πληροφορίας σε ένα υπολογιστικό σύστημα. Είναι η δυνατότητα που έχει ένα υπολογιστικό σύστημα να προστατεύει και να κρατάει ασφαλή τα δεδομένα του , τις εφαρμογές που είναι εγκατεστημένες σε αυτό καθώς και να διατηρεί ασφαλές και προστατευμένο το ίδιο το σύστημα. Αποτελείται απο 5 βασικούς πυλώνες: **εμπιστευτικότητα** (Confidentiality), **ακεραιότητα** (Integrity), **διαθεσιμότητα** (Availability), **αυθεντικότητα** (Authenticity) και **μη-αποποίηση ευθύνης** (Non-Repudiation) [3], [4].



Εικόνα 1: Οι 5 πυλώνες της ασφάλειας της πληροφορίας [5]

Εμπιστευτικότητα (Confidentiality): η δυνατότητα που έχει ένα σύστημα να κρατάει τα δεδομένα των χρηστών του ασφαλή. Πρακτικά κανείς μη-εξουσιοδοτημένος χρήστης δεν μπορεί να έχει πρόσβαση σε συγκεκριμένα δεδομένα για τα οποία απαιτείται εξουσιοδότηση.

Ακεραιότητα (Integrity): η δυνατότητα ενός συστήματος να εγγυάται στο χρήστη ότι τα δεδομένα του, που βρίσκονται αποθηκευμένα σε αυτό, δεν θα μεταβληθούν-καταστραφούν-αλλοιωθούν από κάποιον άλλο μη-εξουσιοδοτημένο χρηστή χωρίς την έγκριση του.

Διαθεσιμότητα (Availability): η δυνατότητα ενός συστήματος να είναι πάντα σε θέση να έχει τα δεδομένα διαθέσιμα στον χρήστη, όποτε αυτός τα ζητήσει.

Αυθεντικότητα (Authenticity): η δυνατότητα που έχει το σύστημα να αυθεντικοποιεί ένα χρήστη βάσει των διαπιστευτηρίων που αυτός παρέχει και παρράλληλα να προστατεύει τα δεδομένα των χρηστών από κακόβουλα άτομα που θα προσπαθήσουν να υποκριθούν άλλους χρήστες.

Μη-αποποίηση ευθύνης (Non-Repudiation): η δυνατότητα που έχει το σύστημα να διασφαλίζει ότι κάθε συναλλαγή του χρήστη με αυτό σημαίνεται-μαρκάρεται με τέτοιο τρόπο που ο χρήστης δεν μπορεί να αρνηθεί την ενέργεια του αυτή.

Καθώς πολλές φορές υπάρχει σύγχυση όταν αναφερόμαστε σε αυθεντικοποιημένο χρήστη και εξουσιοδοτημένο χρήση να αναφέρουμε τη βασική διαφορά τους.

1.3 Αυθεντικοποίηση (Authentication)

Η αυθεντικοποίηση στα υπολογιστικά συστήματα αναφέρεται στη διαδικασία επαλήθευσης της ταυτότητας ενός χρήστη ή συσκευής πριν επιτραπεί η πρόσβαση σε πόρους ή δεδομένα, δηλαδή το «ποιός» είναι. Αυτό συνήθως περιλαμβάνει την παροχή αποδεικτικών στοιχείων ταυτότητας, όπως ένας κωδικός πρόσβασης, ένα ψηφιακό πιστοποιητικό ή βιομετρικά δεδομένα [6] [7]. Η διαδικασία αυτή διασφαλίζει ότι μόνο εξουσιοδοτημένα άτομα ή συσκευές έχουν πρόσβαση σε ευαίσθητους πόρους, παρέχοντας έτσι ένα επίπεδο ασφάλειας στα συστήματα και δίκτυα. Η αυθεντικοποίηση είναι ένα κρίσιμο στοιχείο στην προστασία των πληροφοριακών συστημάτων και πρέπει να διεξάγεται με ακρίβεια και αξιοπιστία.

1.4 Εξουσιοδότηση (Authorization)

Η εξουσιοδότηση στα υπολογιστικά συστήματα αφορά τη διαδικασία καθορισμού των δικαιωμάτων και προνομίων ενός χρήστη ή ενός συστήματος, δηλαδή το «τι» μπορεί να προσπελάσει ο χρήστης. Αφού πραγματοποιηθεί η αυθεντικοποίηση και επιβεβαιωθεί η ταυτότητα του χρήστη, η εξουσιοδότηση καθορίζει ποιες προσβάσεις, δεδομένα ή πόρους μπορεί να χρησιμοποιήσει ο εν λόγω χρήστης [8]. Αυτό συμπεριλαμβάνει την παροχή ή τον περιορισμό δικαιωμάτων για την ανάγνωση, εγγραφή, εκτέλεση ή τροποποίηση αρχείων και πόρων στο σύστημα. Η εξουσιοδότηση είναι ένας θεμελιώδης μηχανισμός για τη διασφάλιση της ασφάλειας και της σωστής διαχείρισης των πληροφοριακών

συστημάτων, καθώς εξασφαλίζει ότι κάθε χρήστης έχει μόνο την απαραίτητη πρόσβαση που απαιτείται για τις εργασίες του.

Ενώ η αυθεντικοποίηση ασχολείται με τον έλεγχο της ταυτότητας ενός χρήστη ή συσκευής, η εξουσιοδότηση ασχολείται με τον ορισμό των επιτρεπτών ενεργειών που μπορεί να εκτελέσει αυτός ο χρήστης ή συσκευή μετά την επιτυχή αυθεντικοποίηση. Αυτό σημαίνει ότι ένας χρήστης μπορεί να είναι αυθεντικοποιημένος, αλλά αυτό δεν του εξασφαλίζει αυτομάτως πρόσβαση σε όλους τους πόρους ή δεδομένα του συστήματος. Η εξουσιοδότηση επιτρέπει στους διαχειριστές των συστημάτων να ελέγχουν προσεκτικά την πρόσβαση σε ευαίσθητα δεδομένα και να παρέχουν διαφορετικά επίπεδα πρόσβασης ανάλογα με το ρόλο ή τη θέση του χρήστη. Για παράδειγμα, ένας διαχειριστής δικτύου μπορεί να έχει πλήρη πρόσβαση σε όλα τα συστήματα και δεδομένα, ενώ ένας τυπικός χρήστης μπορεί να έχει περιορισμένη πρόσβαση μόνο στα απαραίτητα για την εργασία του.

2. Θεωρητικό Πλαίσιο

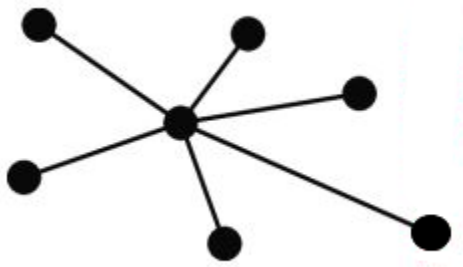
2.1 Ιστορική Εξέλιξη Υπολογιστικών Συστημάτων

Ένας απο τους πιο διαδεδομένους τύπους συστήματος είναι αυτός του κεντρικοποιημένου συστήματος. Αποτέλεσε βάση σχεδιασμού για τους περισσότερους και σημαντικότερους οργανισμούς σε πάρα πολλούς τομείς: τραπεζικός τομέας, τηλεπικοινωνίες. Με την πάροδο των ετών, την εξέλιξη της τεχνολογίας και παράλληλα την ολοένα και περισσότερο αυξανόμενη πολυπλοκότητα των αναγκών, οι περιορισμοί ενός τέτοιου συστήματος άρχισαν να βγαίνουν στην επιφάνεια.

Η ανάγκη για ένα πιο σύγχρονο σύστημα με καλύτερη ανταπόκριση και προσαρμογή στις σύγχρονες απαιτήσεις της εποχής συνετέλεσε στο σχεδιασμό πιο σύγχρονων τύπων συστημάτων. Ακολουθούν οι σημαντικότεροι τύποι συστημάτων.

2.2 Κεντρικοποιημένα Συστήματα

Είναι συστήματα που βασίζονται στην αρχιτεκτονική πελάτη-εξυπηρετητή (client-server). Χαρακτηρίζονται από την απλότητα σχεδιασμού όπου ένας κεντρικός κόμβος αποτελεί τον εξυπηρετητή (server) και όλοι οι υπόλοιποι κόμβοι (πελάτες) είναι απευθείας συνδεδεμένοι μαζί του. Αποτελούν τον πιο διαδεδομένο τύπο συστήματος. Το κεντρικοποιημένο σύστημα αποτελείται από ένα πλήρες υπολογιστικό σύστημα (πρακτικά ένας υπολογιστής) το οποίο δεν αλληλεπιδρά με κάποιο άλλο. Οι πόροι που διαθέτει (μνήμη, επεξεργαστές, αποθηκευτικά μέσα) είναι προς αποκλειστική του χρήση και χρησιμοποιούνται με βάση τη ζήτηση από τους χρήστες.



Εικόνα 2: Κεντρικοποιημένο Σύστημα [6]

2.2.1 Βασικά χαρακτηριστικά των κεντρικοποιημένων συστημάτων

Υπάρχει μια κεντρική μονάδα η οποία είναι υπεύθυνη για την εξυπηρέτηση και το συγχρονισμό όλων των υπολοίπων κόμβων. Υπάρχει ένα καθολικό ρολόι που βρίσκεται στον εξυπηρετητή και κάθε κόμβος συγχρονίζεται βάσει αυτού. Ο/Οι επεξεργαστές μπορεί να διαθέτουν κάποια cache μνήμη και να αποθηκεύουν τοπικά κάποιο μέρος της, ώστε να επιταχύνουν τη διαδικασία προσπέλασης στα δεδομένα. Πιθανό σφάλμα και μη διαθεσιμότητα του εξυπηρετητή θα έχει ως συνέπεια τη μη λειτουργία του πλήρους συστήματος για όλους τους χρήστες. Το scaling είναι δυνατό με την έννοια του κάθετου: Αύξηση των πόρων του συστήματος όπως η μνήμη και οι δίσκοι. Ωστόσο η αύξηση αυτή πέραν του μεγάλου κόστους που μπορεί να έχει (πχ ένας δίσκος μεγαλύτερης χωρητικότητας θα κοστίσει πολύ περισσότερο από 2 μικρότερης χωρητικότητας , αλλά σωρευτικά ίδιας χωρητικότητας) κάποιο φυσικό όριο ενώ αν οι απαιτήσεις προς τον εξυπηρετητή αυξηθούν πάνω από ένα βαθμό το πιθανότερο είναι ότι δεν θα μπορεί το σύστημα να ανταποκριθεί. Επίσης μπορούν να παρατηρηθούν συχνά φαινόμενα bottleneck, όταν τα αιτήματα ξεπεράσουν τον αριθμό που μπορεί ο εξυπηρετητής να εξυπηρετήσει.

Κλασικό παράδειγμα κεντρικοποιημένου συστήματος μπορεί να είναι ένας εξυπηρετητής σχεσιακής βάσης δεδομένων ή ένα σύστημα τραπεζικής διαχείρισης.

2.2.2 Πλεονεκτήματα και μειονεκτήματα των κεντρικοποιημένων συστημάτων

Στα πλεονεκτήματα των κεντρικοποιημένων συστημάτων μπορούμε να συμπεριλάβουμε ότι είναι:

- Απλούστερα στη χρήση και στην παραμετροποίηση ασφαλείας. Καθώς αποτελείται από ένα σύστημα, ουσιαστικά υπάρχει μόνο ένα κέντρο επιβολής πολιτικών ασφαλείας και όλοι οι χρήστες συγχρονίζονται και υπακούουν αυτό.
- Οι πόροι τους είναι προς αποκλειστική του χρήση.
- Το καθολικό ρολόι στον εξυπηρετητή σημαίνει ότι όλοι οι υπόλοιποι συγχρονίζονται με αυτό.
- Έχουν μικρότερο χρόνο αντίδρασης-εξυπηρέτησης , καθώς δεν μεσολαβούν άλλα συστήματα και δίκτυα επικοινωνίας.
- Έχουν λιγότερα πρωτόκολλα και πιο απλή αρχιτεκτονική που οδηγούν σε ταχύτερη και απλούστερη υλοποίηση και καλύτερη αποδοτικότητα υπό προϋποθέσεις.
- Πολύ πιο εύκολη και γρήγορη αναβάθμιση: ένας εξυπηρετητής σημαίνει ένα σχέδιο κεντρικής αναβάθμισης και όλοι συνδέονται με αυτόν και συγχρονίζουν αντίστοιχα.

Τα σημαντικότερα μειονεκτήματα είναι ότι:

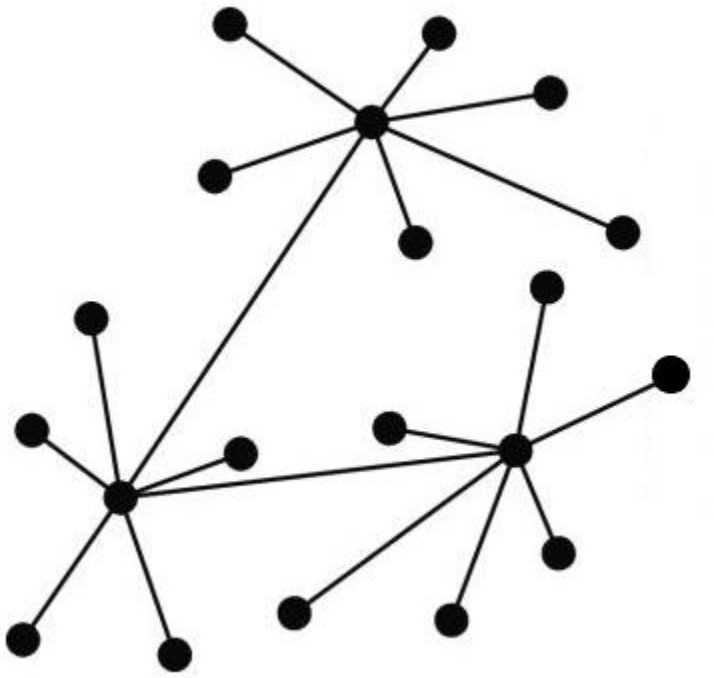
- Αν το φορτίο αυξηθεί σημαντικά οι περιορισμοί στο scaling είναι σημαντικοί. Δεν έχει τη δυνατότητα να αυξήσει τους πόρους πάνω από ένα επίπεδο είτε λόγω κόστους είτε λόγω αποδοτικότητας.
- Προβλήματα στη διαθεσιμότητα εξαιτίας ενός μοναδικού σημείου αποτυχίας (Single point of Failure). Αν για οποιονδήποτε λόγο σημαντικό ή ασήμαντο ο εξυπηρετητής είναι μη διαθέσιμος , τότε το σύστημα είναι μη διαθέσιμο για όλους τους χρήστες ταυτόχρονα.
- Επειδή είναι ένας και μοναδικός ο εξυπηρετητής, η συντήρηση του απαιτεί ιδιαίτερο χειρισμό: Ενημέρωση και προγραμματισμό σε ώρες εκτός λειτουργίας ώστε να μπορέσει να πραγματοποιηθεί χωρίς προβλήματα.
- Ρίσκα ασφαλείας: καθώς υπάρχει μόνο μια κεντρική αρχή επιβολής πολιτικών ασφαλείας, αν αυτή παραβιαστεί τότε υπάρχει πλήρης πρόσβαση σε όλα τα δεδομένα.
- Έλλειψη transparency: ως κεντρική αρχή έχει τον πλήρη έλεγχο του συστήματος και έτσι αυτό μπορεί να οδηγήσει σε φαινόμενα μεροληψίας υπέρ χρηστών.

2.3 Αποκεντρωμένα Συστήματα

Είναι συστήματα που βασίζονται σε 2 βασικούς αρχιτεκτονικούς κανόνες:

Peer-to-Peer αρχιτεκτονική: Κάθε κόμβος είναι ισότιμος του άλλου και κανείς δεν έχει κάποιο στοιχείο που να τον καθιστά ανώτερο από τους υπόλοιπους.

Master-Slave αρχιτεκτονική: Ένας κόμβος μπορεί να γίνει Master (κεντρικός κόμβος) βάσει εκλογής από τους υπόλοιπους ώστε να έχει την ευθύνη συντονισμού των υπολοίπων αλλά πάντα θα θεωρείται ισότιμος των υπολοίπων.



Εικόνα 3: Αποκεντρωμένο Σύστημα [6]

Βάσει των 2 αυτών σημαντικών αρχιτεκτονικών και ενώ με μια πρώτη ματιά φαίνονται να αποτελούνται από πιο μικρά κεντρικοποιημένα συστήματα, αντιλαμβανόμαστε οτι οι σημαντικές διαφορές εδώ είναι ότι οι κεντρικοί κόμβοι τους λειτουργούν ως αρχή του υποσυστήματος τους και μπορούν να επικοινωνήσουν μεταξύ τους. Έτσι όχι μόνο σχηματίζεται ένα δίκτυο κεντρικών και μη κόμβων αλλά το σημαντικότερο είναι οτι δεν υπάρχει μια αποκλειστική αρχή επιβολής αποφάσεων και πολιτικών ασφαλείας για όλο το σύστημα. Κάθε κεντρικός κόμβος λαμβάνει τις δικές του αποφάσεις και έπειτα επικοινωνούνται στους υπόλοιπους κεντρικούς κόμβους. Και από την άλλη κάθε κόμβος μπορεί να ανα πάσα στιγμή να εκλεγεί ως κεντρικός κόμβος ώστε να αναλάβει αυτός το συντονισμό των υπολοίπων.

Αυτό το χαρακτηριστικό συμβάλλει στο να μην έχει τη δυνατότητα ένας κεντρικός κόμβος να ανακτήσει τον πλήρη έλεγχο όλου του δικτύου και να επιβάλλει τις δικές του πολιτικές και αποφάσεις αλλά πάντα να είναι ελεγχόμενος από το υπόλοιπο δίκτυο. Χαρακτηριστικό παράδειγμα τέτοιου συστήματος είναι το Bitcoin blockchain.

2.3.1 Βασικά χαρακτηριστικά των αποκεντρωμένων συστημάτων

Δεν υπάρχει κάποιο καθολικό ρολόι: ο εκάστοτε κεντρικός κόμβος έχει το δικό του βάσει του οποίου συγχρονίζονται και όλοι οι υπόλοιποι κόμβοι που είναι συνδεδεμένοι μαζί του.

Υπάρχουν πολλές κεντρικές μονάδες επεξεργασίας: υπάρχουν περισσότεροι του ενός κεντρικοί κόμβοι για να εξυπηρετήσουν τους κόμβους.

Μια απώλεια κεντρικού κόμβου δεν σημαίνει απαραίτητα οτι ολο το σύστημα δεν είναι διαθέσιμο: Τα υπόλοιπα μέρη του συστήματος λειτουργούν χωρίς κανένα πρόβλημα.

Scaling: Και εδώ είναι δυνατή η κάθετη αναβάθμιση-αύξηση των πόρων του συστήματος ανα κόμβο.

Δεν προτείνεται για μικρά συστήματα καθώς διαθέτει σημαντική πολυπλοκότητα.

2.3.2 Πλεονεκτήματα και μειονεκτήματα των αποκεντρωμένων συστημάτων

Στα πλεονεκτήματα των αποκεντρωμένων συστημάτων μπορούμε να συμπεριλάβουμε ότι:

- Δεν παρουσιάζει συχνά bottlenecks: Η κατανομή του φορτίου επιμερίζεται με τέτοιο τρόπο ώστε πάντα να υπάρχει η μικρότερη δυνατή συμφόρηση.
- Παρουσιάζει υψηλή διαθεσιμότητα: Πάντοτε κάποιοι κόμβοι θα έιναι διαθέσιμοι.
- Μεγάλη αυτονομία και έλεγχος των πόρων: Κάθε κόμβος ελέγχει τη συμπεριφορά του και μαζί διαχειρίζεται και τους πόρους του.
- Αντίσταση στο λάθος (Fault-Tolerance) :Βάσει σχεδιασμού τα αποκεντρωμένα συστήματα δεν υπακούουν στην αρχή του Single Point of Failure , οπότε έχει προβλεφθεί να μπορεί το σύστημα να εξακολουθεί να βρίσκεται σε λειτουργία ακόμη και αν κάποιοι κόμβοι του δεν λειτουργούν.
- Αυξημένο Transparency: Κάθε κόμβος έχει την ίδια πρόσβαση στην πληροφορία, καθώς είναι όλοι ισότιμοι.
- Καλύτερη ασφάλεια: Τα δεδομένα είναι διασκορπισμένα στους κόμβους, οπότε αν υπάρξει επίθεση σε ένα κόμβο δεν είναι απαραίτητο ότι θα υπάρξει πρόβλημα στους υπόλοιπους.
- Καλύτερο Scaling: Στα αποκεντρωμένα συστήματα υπάρχει η δυνατότητα να προστεθούν νέοι κόμβοι, οπότε αυτομάτως ο επιμερισμός και η κατανομή του φορτίου βελτιώνονται άμεσα.

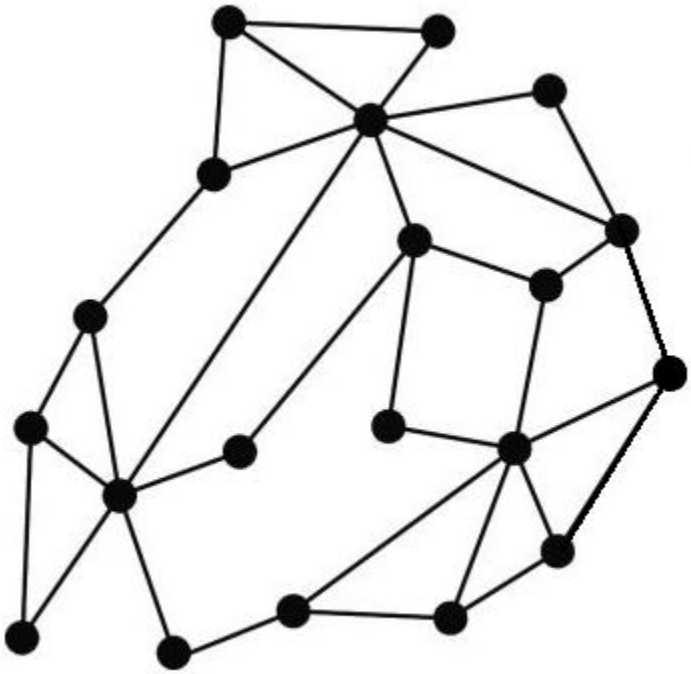
Τα σημαντικότερα μειονεκτήματα είναι ότι:

- Δεν είναι εύκολη η επίβλεψη ενός κανονιστικού πλαισίου
- Δυσκολία αναγνώρισης εσφαλμένου κόμβου: Δεν είναι εύκολο να προσδιοριστεί άμεσα ποιος κόμβος έχει πρόβλημα. Η λογική είναι αυτή της συνδεδεμένης λίστας όπου κάθε κόμβος πρέπει να ερωτηθεί ξεχωριστά.

- Δυσκολία αναγνώρισης του κόμβου εξυπηρέτησης: Δεν είναι εύκολο και εδώ να προσδιοριστεί ποιος είναι ο κόμβος που ανταποκρίθηκε σε ένα αίτημα μας.
- Μεγαλύτερες προκλήσεις ασφαλείας: τέτοια συστήματα είναι πιο πολύ ευάλωτα σε επιθέσεις όπως DDoS (Κατανεμημένη Άρνηση Υπηρεσίας - Distributed Denial of Service) και επιθέσεις του 51% (Απόκτηση ελέγχου ποσοστού κυριαρχίας των κεντρικών κόμβων)
- Προκλήσεις Scaling: από ένα μέγεθος και πάνω το scaling γίνεται πονοκέφαλος, καθώς κάθε κόμβος πχ μπορεί να κρατάει τοπικό αντίγραφο της βάσης. Η υποστήριξη και ο συντονισμός όλης αυτής της διαδικασίας μπορεί να αποβεί πολύ πολύπλοκος.
- Δεν υπάρχει διαμοιρασμός πόρων: Κάθε κόμβος έχει τους δικούς του πόρους αλλά όταν κάποιοι είναι χωρίς χρήση οι πόροι τους είναι πρακτικά ανενεργοί και δεν μπορούν να χρησιμοποιηθούν από άλλους.
- Κόστος συναλλαγών: Κάθε συναλλαγή κοστίζει σε χρόνο καθώς πρέπει να επιβεβαιωθεί και να επικυρωθεί και από τους υπόλοιπους κόμβους.
- Έλλειψη κανονιστικού πλαισίου (standardization): η ολοκλήρωση με άλλα συστήματα μπορεί να οδηγήσει σε δια-λειτουργικά προβλήματα.

2.4 Κατανεμημένα Συστήματα

Όπως αναφέρεται και στον ορισμό: Με τον όρο κατανεμημένο σύστημα εννοούμε ένα σύνολο ανεξαρτήτων υπολογιστικών μονάδων (υπολογιστών) για το οποίο οι χρήστες έχουν την εικόνα ότι είναι μια οντότητα , ένα υπολογιστικό σύστημα [1]. Αποτελούνται δηλαδή από ένα σύνολο υπολογιστών και λογισμικού που συνεργάζεται για την επίτευξη ενός στόχου. Όλοι οι κόμβοι είναι συνδεδεμένοι μεταξύ τους και βασικό τους χαρακτηριστικό είναι η αποφυγή της συμφόρησης ή η μη-διαθεσιμότητα.



Εικόνα 4: Κατανεμημένο Σύστημα [6]

Ένα κατανεμημένο σύστημα προσφέρει μεγάλο transparency, δίνει δηλαδή την εικόνα στον χρήστη οτι πρόκειται για ένα μοναδικό υπολογιστή πίσω από το σύστημα ενώ στην πραγματικότητα το σύστημα μπορεί να αποτελείται από πολλά μικρά υπολογιστικά υποσυστήματα με μεγάλη ομοιογένεια είτε ανομοιογένεια μεταξύ τους. Οι κόμβοι επικοινωνούν μεταξύ τους μέσω ιδιωτικών δικτύων είτε απευθείας μέσω του διαδικτύου.

Χαρακτηριστικό παράδειγμα κατανεμημένου συστήματος είναι οι μηχανές αναζήτησης.

2.4.1 Βασικά χαρακτηριστικά των κατανεμημένων συστημάτων

Διαμοιρασμός πόρων: εδώ πια ο διαμοιρασμός πόρων μπορεί να γίνει με μεγάλη ακρίβεια και με στεγανά. Κάποιοι κόμβοι μπορεί να μοιράζονται υλικό και λογισμικό αλλά όχι μνήμη ή χώρο στον ίδιο δίσκο.

Ταυτόχρονη Επεξεργασία: Δεν μιλάμε για ψευδοπαράλληλη επεξεργασία εδώ αλλά για πολλά μηχανήματα που επεξεργάζονται ταυτόχρονα μέρος της ίδιας υπολογιστικής εργασίας.

Scaling: Δεν υπάρχει όριο καθώς στην περίπτωση αυτή μιλάμε για οριζόντιο Scaling με αύξηση όσων μηχανημάτων απαιτείται.

Εντοπισμός Λάθους: Ο εντοπισμός λάθους γίνεται πολύ εύκολα.

Transparency: Κάθε κόμβος μπορεί να επικοινωνήσει με τους υπόλοιπους είτε μέσω εσωτερικών ιδιωτικών δικτύων είτε απευθείας μέσω του διαδικτύου.

Ομογενοποιημένα συστήματα: Έχουν εγκατεστημένο το ίδιο λογισμικό (λειτουργικό , εφαρμογές κλπ) και τα δεδομένα μπορεί να είναι διαμοιρασμένα ανάμεσα τους.

Ανομοιογενή συστήματα: Έχουν διαφορετικό λειτουργικό, λογισμικό εγκατεστημένα και συνήθως υπάρχουν ολοκλήρωση μεταξύ τους με διάφορες υλοποιήσεις, αποκρύπτοντας τις τεχνικές λεπτομέρειες του πως αυτό επιτυγχάνεται.

2.4.2 Πλεονεκτήματα και μειονεκτήματα των κατανεμημένων συστημάτων

Στα πλεονεκτήματα των κατανεμημένων συστημάτων μπορούμε να συμπεριλάβουμε:

- Μικρή Υστέρηση (Low Latency): Συνήθως χαμηλή υστέρηση στη απόκριση λόγω της μεγάλης γεωγραφικής διασποράς των συστημάτων.
- Κλιμάκωση (Scalability): Πολύ μεγάλες δυνατότητες λόγω της οριζόντιας κλιμάκωσης, μπορούμε πάντα να προσθέσουμε νέους κόμβους.
- Ανοχή στο λάθος (Fault Tolerance): Αν κάποιοι κόμβοι καταστούν μη διαθέσιμοι, η δουλεία εξακολουθεί να διαμοιράζεται σε διαφορετικούς κόμβους και το σύστημα εξακολουθεί να λειτουργεί με την ίδια αποδοτικότητα.
- Αυξημένη Αξιοπιστία: Πάντα υπάρχουν κόμβοι διαθέσιμοι.
- Αποδοτικότητα Κόστους: Για τη δημιουργία ενός κατανεμημένου συστήματος μπορούν να χρησιμοποιηθούν οτι μηχανήματα υπάρχουν διαθέσιμα άμεσα, χωρίς να χρειάζεται άμεσα η αγορά νέου υλισμικού.
- Αυτονομία: Κάθε εγκατάσταση που φιλοξενεί μέρος του συστήματος , μπορεί να έχει ξεχωριστούς κανόνες ελέγχου τοπικά.
- Κοινή Χρήση Δεδομένων: Οι τελικοί χρήστες που βρίσκονται σε μια γεωγραφική περιοχή μπορούν να προσπελάσουν δεδομένα που ανήκουν σε κόμβους εγκατάστασης σε άλλη περιοχή.

Τα σημαντικότερα μειονεκτήματα είναι:

- Μεγάλη πολυπλοκότητα: Σημαντικά ζητήματα όπως η καταγραφή και απεικόνιση των αρχείων αυτών σε πραγματικό χρόνο συχνά αποτελούν δύσκολο σημείο στις υλοποιήσεις και απαιτούν μεγάλους τεχνικούς συμβιβασμούς (trade-offs).
- Επικοινωνία Δικτύων: Απαιτείται πολύπλοκη υποδομή δικτύων με αποτέλεσμα να υπάρχουν πάντα οι πιθανότητες συμφόρησης η μη διαθεσιμότητας μέρους των δικτύων αυτών ή ακόμη και του συνόλου τους.

- Δυσκολία Ομοφωνίας (Consensus): Είναι πολύ δύσκολο να επιτευχθεί ομοφωνία ταυτόχρονα σε όλο το σύστημα.
- Προκλήσεις Ασφαλείας: Τα κατανεμημένα συστήματα μπορεί να αποτελούν στόχο κυβερνοεπιθέσεων, διαρροής-δεδομένων (data-breach) και μη-αυθεντικοποιημένης πρόσβασης. Η διασφάλιση της ασφάλειας και της εμπιστευτικότητας των δεδομένων αποτελεί δύσκολο έργο.
- Συγχρονισμός: Η συνοχή των δεδομένων (data consistency) ανάμεσα σε όλους τους κόμβους αποτελεί μεγάλη πρόκληση που επιτυγχάνεται με αλγόριθμους και πρωτόκολλα μεγάλης πολυπλοκότητας.
- Κόστος: Η ανάπτυξη, λειτουργία και συντήρηση τέτοιων συστημάτων έχουν πολύ υψηλά κόστη.
- Περιορισμοί Κλιμάκωσης (Scalability Limitations): Μπορεί να υπάρξουν περιορισμοί κλιμάκωσης βάσει των περιορισμών των δικτύων , του εξοπλισμού κλπ.

2.5 Βασικές αρχές ασφάλειας στα Κατανεμημένα Συστήματα

Βασικές αρχές ασφαλείας ενός κατανεμημένου συστήματος είναι οι ακόλουθες σύμφωνα με τον [7] :

- Η διασφάλιση της ασφαλούς επικοινωνίας: Η ασφαλής επικοινωνία μεταξύ των κόμβων (χρηστών και μη).
- Ο έλεγχος της πρόσβασης: ο έλεγχος και η διασφάλιση της πρόσβασης σε οποιοδήποτε δεδομένο από τον οποιονδήποτε κόμβο ή χρήστη.
- Η διαθεσιμότητα του συστήματος: το σύστημα πάντα θα πρέπει να έχει διαθέσιμους κόμβους και πόρους ώστε να ανταποκριθεί στα αιτήματα των χρηστών.
- Διαχείριση Απειλών Ασφαλείας: Οι πιθανοί κίνδυνοι και οι απειλές ασφαλείας θα πρέπει να αναγνωρίζονται έγκαιρα και να αντιμετωπίζονται προτού γίνουν αντικείμενο εκμετάλλευσης.
- Καταγραφή των γεγονότων: Πρέπει πάντα να υπάρχει καταγραφή σε κάθε γεγονός/συμβάν (event monitoring) ώστε να παρέχεται ιχνηλασιμότητα και επιβεβαίωση των κινήσεων και ενεργειών κάθε χρήστη

2.6 Κρίσιμες Προκλήσεις και Ευπάθειες

Με τον όρο ευπάθεια εννοούμε ότι υπάρχουν σημεία του συστήματος που είτε είναι σχεδιασμένα με κενά ασφαλείας είτε επιτρέπουν σε ένα κακόβουλο χρήστη να τα χρησιμοποιήσει με τέτοιο τρόπο ώστε να χρησιμοποιήσει το σύστημα με τρόπο που δεν σχεδιάστηκε και να παρακάμψει τις διάφορες πολιτικές ασφαλείας του, αποκτώντας πρόσβαση , σε δεδομένα που δεν δικαιούται [11].

Δίκτυα και Επικοινωνία: Η πιθανότητα του να παρεισφρήσει κάποιος στο δίκτυο και στην επικοινωνία ανάμεσα στους χρήστες και τους κόμβους είναι μεγαλύτερη από των απλών ολοκληρωμένων υπολογιστικών συστημάτων [10]. Υπάρχουν μεγάλοι κίνδυνοι από κακόβουλους χρήστες που μπορεί να δοκιμάσουν να παρακολουθούν τα μηνύματα που ανταλλάσσουν οι κόμβοι μεταξύ τους είτε ακόμη και να μπορέσουν να τα αλλάξουν (Main-in-the-Middle).

Άρνηση εξυπηρέτησης (Denial of Service): Η πιθανότητα του να μπορεί κάποιος να αποτρέψει έναν αυθεντικοποιημένο χρήστη από το να έχει πρόσβαση σε συγκεκριμένα δεδομένα είτε σε ολόκληρο το σύστημα.

Προσπάθειες εξαπάτησης του συστήματος (Faking - Impersonation): Η προσπάθεια που μπορεί να κάνει κάποιος να πείσει το σύστημα οτι είναι κάποιος άλλος ώστε να του επιτραπεί η πρόσβαση.

Απειλές Replay: Η δυνατότητα που μπορεί να έχει κάποιος ώστε να προσπαθήσει να επανα-χρησιμοποιήσει παλαιότερα δεδομένα και μηνύματα ώστε να εκμεταλλευθεί τυχόν κενά του συστήματος. Για παράδειγμα να δοκιμάσει κάποιος να επαναχρησιμοποιήσει πληροφορία για πληρωμή από τρίτο σε κάποιον λογαριασμό που του ανήκει.

Ανάλυση της ροής των πακέτων του δικτύου (Traffic Monitoring): Πολύ σημαντικό ώστε να αντιληφθούμε έγκαιρα αν κάποια συμπεριφορά είναι μη φυσιολογική ώστε να αξιολογηθεί σωστά.

Τυχαία πρόσβαση: Μπορεί να συμβεί να αποκτήσει κάποιος πρόσβαση σε δεδομένα που δεν έχει πρόσβαση λόγω κενών του συστήματος. Συμβαίνει αρκετές φορές πιθανότατα από λάθη του software (bugs), κενά πολιτικών ασφαλείας.

Να σημειωθεί ότι για την αυθεντικοποίηση είναι συνήθως υπεύθυνη η εφαρμογή και όχι το σύστημα οπότε ένα σύστημα όσο καλά και αν είναι θωρακισμένο δεν μπορεί να προβλέψει όλα τα κενά μιας εφαρμογής την οποία διαθέτει στον έξω κόσμο μέσω του διαδικτύου.

3. Ανάλυση Κατανεμημένων Συστημάτων

3.1 Σχεδιασμός και Προκλήσεις

Οι βασικές απαιτήσεις που θα πρέπει να ικανοποιεί ένα κατανεμημένο σύστημα και θα πρέπει να ληφθούν σοβαρά υπόψιν κατά το σχεδιασμό του είναι οι εξής [1], [9]:

Διαμοιρασμός Πόρων (Resource Sharing): Θα πρέπει να είναι εύκολο για τους χρήστες να έχουν πρόσβαση και χρήση σε διαμοιραζόμενους πόρους, όπως μνήμη, δεδομένα , αρχεία, αποθηκευτικός χώρος κλπ. Υπάρχει σημαντικό όφελος οικονομίας κλίμακας όταν μπορούμε να έχουμε έναν κοινόχρηστο χώρο για παράδειγμα αποθήκευσης και να τον χρησιμοποιούν όλοι οι χρήστες , ανάλογα με τις απαιτήσεις τους. Σε αντίθεση με το να πρέπει να έχουμε πολλούς μικρούς χώρους αποκλειστικά για κάθε χρήστη. Το ίδιο μπορεί να συμβεί με πόρους που μπορούμε να διαθέσουμε όταν κάποιοι κόμβοι το απαιτούν λόγω φόρτου υπολογιστικών εργασιών μέχρι το πέρας των εργασιών τους. Και αντίστοιχα να επιστραφούν οι πόροι πίσω και να δοθούν σε άλλους κόμβους ώστε να εκτελέσουν και αυτοί τις δικές τους υπολογιστικές εργασίες [13].

Διαφάνεια (Transparency): Η διαφάνεια έχει περισσότερο αφαιρετικό χαρακτήρα και υποδεικνύει την απόκρυψη των επιμέρους μερών του κατανεμημένου συστήματος από τον χρήστη. Δηλαδή ο χρήστης αγνοεί ότι η πληροφορία που του επιστρέφει ο υπολογιστής προέρχεται απο κάποιο κόμβο ενός διασκορπισμένου δικτύου, αλλά θεωρεί πως πίσω απο το σύστημα βρίσκεται απλώς ένα ολοκληρωμένο υπολογιστικό σύστημα που κάνει τη δουλειά αυτή (ένας υπολογιστής επι της ουσίας εξής) [14], [15].

Έτσι λοιπόν αγνοεί της διάφορες πτυχές της διαφάνειας του συστήματος που είναι οι ακόλουθες:

- Διαφάνεια Περιοχής (Location Transparency): Ο χρήστης δεν ενδιαφέρεται για το που βρίσκεται η φυσική εγκατάσταση των συστημάτων που χρησιμοποιεί.
- Διαφάνεια Μεταφοράς-Μετακίνησης (Migration Transparency): Οι πόροι μπορούν να μετακινηθούν κατ’ ανάγκη χωρίς ο χρήστης να το αντιληφθεί.
- Διαφάνεια Αντιγράφων (Replication Transparency): Ο χρήστης δεν αντιλαμβάνεται αν υπάρχουν ένα ή περισσότερα αντίγραφα της πληροφορίας που ζητάει.
- Διαφάνεια Συγχρονισμού (Concurrency Transparency): Οι χρήστες μοιράζονται δεδομένα συγχρονισμένα χωρίς καμία ειδοποίηση-παρεμβολή-ενημέρωση.
- Διαφάνεια Παραλληλίας (Parallelism Transparency): Οι λειτουργίες γίνονται παράλληλα

Ανοιχτότητα (Openness): Απο τα πολύ σημαντικά στοιχεία ενός κατανεμημένου συστήματος: Να είναι ένα σύστημα ανοιχτό. Δηλαδή να βασίζεται σε μέρη και υλοποιήσεις που μπορούν να χρησιμοποιηθούν να ενσωματωθούν σε άλλα συστήματα ενώ και τα ίδια μπορεί να προέρχονται απο κάποιο άλλο υποσύστημα. Θα μπορούσαμε να το αναφέρουμε και ως **Διαλειτουργικότητα (Interoperability)** [9] , την ικανότητα δηλαδή

των μερών που απαρτίζουν το σύστημα να μπορούν να λειτουργήσουν και να συνδυαστούν με άλλα μέρη άλλων συστημάτων χωρίς μεγάλα προβλήματα. Αντίστοιχα να μπορούν να μεταφερθούν **(Μεταφερσιμότητα, Portability)** και να χρησιμοποιηθούν από άλλα συστήματα χωρίς μεγάλο κόστος: Καθώς και να είναι και εύκολα στην επέκταση τους **(Επεκτασιμότητα, Extensibility).**

Προσαρμοστικότητα (Flexibility): Η ευκολία με την οποία πρέπει να μπορεί να αναπτυχθεί ένα κατανεμημένο σύστημα [1]. Πολύ δημοφιλής προσέγγιση είναι αυτή του μικρο-πυρήνα (microkernel) [15], όπου προσπαθεί να χειριστεί τις απολύτως βασικές λειτουργίες (ενα μέρος της μνήμης , λίγες βασικές διεργασίες) και όλα τα υπόλοιπα ανατίθενται στους εξυπηρετητές του επιπέδου των χρηστών.

Διαθεσιμότητα (Availability): Διαθεσιμότητα θεωρείται η μικρο-ποσότητα χρόνου κατα την οποία το σύστημα λειτουργεί [15]. Συνήθως επιτυγχάνεται μέσω της αφαίρεσης και όχι μέσω συνεχούς λειτουργίας πολλών και μεγάλων τμημάτων του συστήματος.

Αξιοπιστία (Reliability): Σε συνδυασμό με την ανοχή στα σφάλματα και τη διαθεσιμότητα, ίσως οι πιο σημαντικοί παράγοντες για ένα κατανεμημένο σύστημα [11]. Βάσει αυτής τα δεδομένα δεν πρέπει να χαθούν, να καταστραφούν ή να αλλοιωθούν. Πρέπει πάντα να είναι ασφαλή και το σύστημα πάντα πρέπει να είναι ανθεκτικό στα σφάλματα (Fault-Tolerant)

Ανοχή στα σφάλματα (Fault-Tolerance): Η ανοχή δηλαδή του συστήματος όταν θα έχει σφάλματα να γίνονται με τέτοιο τροπο και να υπάρχει τέτοια διαχείριση , που το σύστημα πάντα θα είναι διαθέσιμο. Θα υπάρχει δηλαδή πάντοτε κόμβος να εξυπηρετήσει τα αιτήματα του χρήστη.

Απόδοση (Performance): Απαιτείται πλήρης κατανόηση του περιβάλλοντος στο οποίο το σύστημα θα λειτουργήσει, ώστε να μπορέσουμε να ρυθμίσουμε όλες τις παραμέτρους για τη βελτιστοποίηση της απόδοσης του συστήματος [15]. Για παράδειγμα, αν έχουμε πολύ κακή απόδοση στο δίκτυο, μπορεί με την αύξηση της παραλληλίας σε κάποιες λειτουργίες να βελτιώσουμε την απόκριση του συστήματος στα αιτήματα του χρήστη.

Κλιμάκωση-Επεκτασιμότητα (Scalability): Ιδανικά ένα κατανεμημένο σύστημα θα μπορεί να επεκτείνεται χωρίς κανένα όριο [15]. Ωστόσο πάντα τέτοια βήματα γίνονται κατόπιν σκέψης και εφαρμογής πολύπλοκων αλγορίθμων (κατανεμημένων αλγορίθμων).

3.2 Αρχιτεκτονική

Διακρίνονται 3 βασικές αρχιτεκτονικές ενώ πρέπει να σημειωθεί πως στα υλοποιημένα συστήματα πολλές φορές οι αρχιτεκτονικές αυτές συνδυάζονται:

- **Ομότιμο Δίκτυο (Peer-To-Peer):** Όλοι οι κόμβοι είναι ισότιμοι και συνεργάζονται για ένα κοινό αποτέλεσμα.
- **Πελάτης-Εξυπηρετητής (Client-Server):** Κάποιοι κόμβοι αναλαμβάνουν το ρόλο του συντονιστή ώστε να συντονίζουν τη διαδικασία.

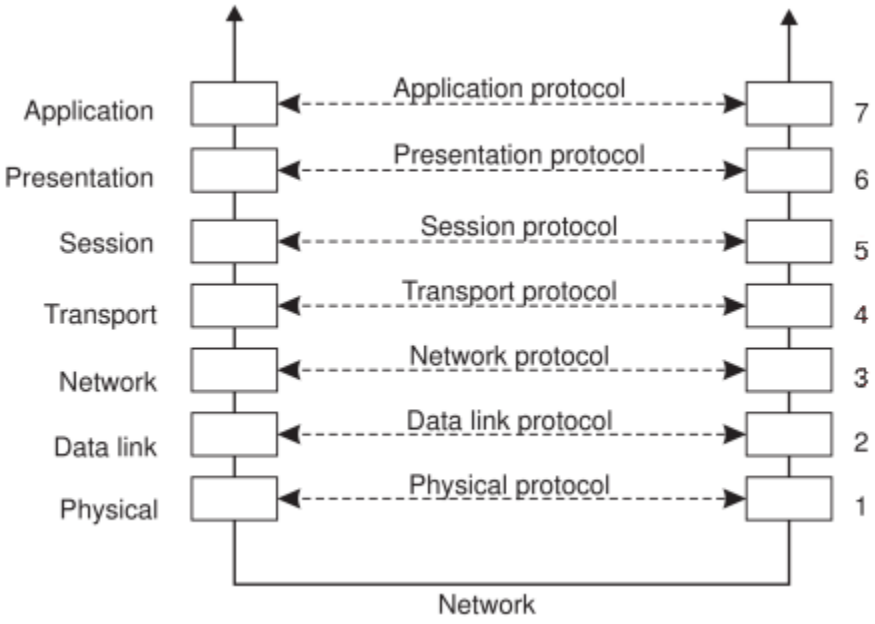
- **Αρχιτεκτονική Ν-Επιπέδων (n-tier Architecture):** Η εφαρμογή μπορεί να είναι διασκορπισμένη σε διάφορους κόμβους και όλοι μαζί συνεργάζονται ως μια.

3.3 Πρωτόκολλα Επικοινωνίας

Η επικοινωνία ανάμεσα στις διεργασίες των τμημάτων (συστατικών, components) ενός κατανεμημένου συστήματος , αποτελεί ένα απο τα βασικότερα σημεία ολόκληρου του συστήματος. Η ανταλλαγή δηλαδή μηνυμάτων ανάμεσα σε διαφορετικά μηχανήματα. Συχνά σε ένα κατανεμημένο σύστημα λειτουργούν παράλληλα χιλιάδες ίσως και εκατομμύρια διεργασίες οι οποίες επικοινωνούν μεταξύ τους. Μέχρι πρότινος η επικοινωνία αυτή βασιζόταν σε πολύ χαμηλού επιπέδου πρωτόκολλα, ωστόσο στις μέρες μας τα 2 σημαντικότερα πρωτόκολλα επικοινωνίας που συναντούμε ευρέως σε κατανεμημένα συστήματα είναι τα εξής [6]:

- Remote Procedure Calls (RPC)
- Message-Oriented- Middleware (MOM)

Υπενθυμίζουμε οτι τα βασικά πρωτόκολλα επικοινωνίας αποτελούν το θεμέλιο της εξέλιξης και χρήσης πιο σύγχρονων πρωτοκόλλων. Με σημαντικότερο το στρωματοποιημένο πρωτόκολλο OSI (Layered Protocol) όπου η πληροφορία μεταφέρεται από το ένα επίπεδο στο άλλο επίπεδο , αρχικά από το φυσικό μέχρι το επίπεδο της εφαρμογής , μέσα στο δίκτυο.



Εικόνα 5 - OSI Model Layers, Interfaces & Protocols [6]

3.3.1 Πρωτόκολλο OSI

Το πρωτόκολλο OSI ορίζει τη λειτουργία στο κάθε ένα απο τα 7 επίπεδα ώστε να μπορέσει να υπάρξει επικοινωνία ανάμεσα σε 2 συστήματα [16].

Ξεκινώντας απο κάτω προς τα πάνω τα επίπεδα έχουν ως εξής:

Φυσικό επίπεδο (Physical layer): Είναι υπεύθυνο για την κωδικοποίηση και μετάδοση των δεδομένων bit (1,0) μέσω του φυσικού μέσου είτε αυτό ειναι καλώδιο, οπτική ίνα ή ασύρματη ζεύξη. Τα δεδομένα κωδικοποιούνται ως ηλεκτρικά, οπτικά ή ηλεκτρομαγνητικά σήματα. Επίσης ασχολείται με τις διεπαφές (interfaces), τον συνδετήρα (connector) και με το συγχρονισμό των συσκευών.

Επίπεδο ζεύξης δεδομένων (Data Link layer): Είναι υπεύθυνο για την παροχή της φυσικής διευθυνσιοδότησης (MAC Address: Media Access Control addresses), τον έλεγχο του μέσου μετάδοσης και το πότε μπορούν να εκπεμφθούν δεδομένα απο αυτό καθώς και την εξααφάλιση της αξιόπιστης επικοινωνίας ανάμεσα σε 2 άμεσα συνδεδεμένους κόμβους του ιδίου τοπικού δικτύου. Είναι επίσης υπεύθυνο για την ανίχνευση και διόρθωση σφαλμάτων στα δεδομένα καθώς και για τη ροή των πληροφοριών και τον συγχρονισμό των πλαισίων δεδομένων (data frames).

Επίπεδο δικτύου (Network layer): Είναι υπεύθυνο για τη λογική διευθυνσιοδότηση και τη δρομολόγηση των πακέτων (routing). Παράλληλα επανασυνθέτει τα πακέτα απο τα μέρη που θα λάβει ανεξαρτήτως της σειράς λήψης τους. Μπορεί να παρέχει τις υπηρεσίες του με ή χωρίς σύνδεση.

Επίπεδο μεταφοράς (Transport layer): Είναι υπεύθυνο για την εξασφάλιση του επιπέδου ποιότητας επικοινωνίας (QoS, Quality of Service) και κάνει όλες τις απαραίτητες ενέργειες για να το επιτύχει. Ενέργειες που αφορούν ζητήματα όπως η καθυστέρηση της αποκατάστασης επικοινωνίας, πιθανή απώλεια σύνδεσης, ικανοποιητικό ρυθμό διακίνησης δεδομένων (throughput), βαθμό προτεραιότητας και ασφάλεια. Μπορεί να παρέχει τις υπηρεσίες είτε προσανατολισμένες στη σύνδεση (connection oriented) είτε χωρίς σύνδεση (connectionless).

Επίπεδο συνόδου (Session layer): Είναι υπεύθυνο για την οργάνωση και το συγχρονισμό της επικοινωνίας στα ανώτερα επίπεδα. Επιτρέπει ή απαγορεύει την παροχή υπηρεσίας, αποκαθιστά τη σύνδεση αν χρειαστεί, αυθεντικοποιεί και εξουσιοδοτεί το χρήστη.

Επίπεδο παρουσίασης (Presentation layer): Είναι υπεύθυνο για την αναπαράσταση της πληροφορίας που μεταφέρεται απο εφαρμογή σε εφαρμογή καθώς και για τη δομή των δεδομένων. Συμπιέζει, κρυπτογραφεί και αποκρυπτογραφεί τα δεδομένα ώστε να είναι κατανοητά απο την εφαρμογή.

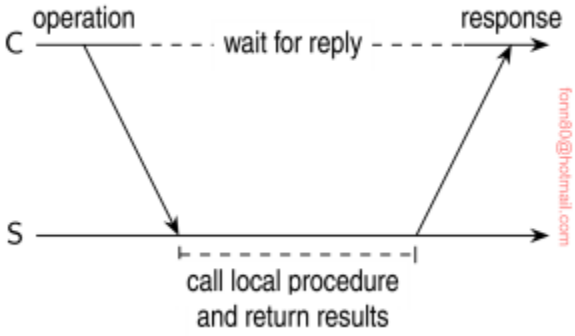
Επίπεδο εφαρμογής (Application layer): Είναι το ανώτερο επίπεδο και είναι υπεύθυνο για την παροχή τρόπου που μια εφαρμογή θα μπορέσει να επικοινωνήσει με μια άλλη. Παράλληλα παρέχει γκάμα πρωτοκόλλων (HTTP, SMTP,FTP)

3.3.2 Remote Procedure Calls (RPC)

Στοχεύει στην απόκρυψη (διαφάνεια) των τεχνικών λεπτομερειών της χαμηλού επιπέδου ανταλλαγής μηνυμάτων και αποτελεί ιδανικό υποψήφιο για τα συστήματα που βασίζονται στην αρχιτεκτονική πελάτη-εξυπηρετητή. Η συγκεκριμένη μορφή επικοινωνίας είναι συγχρονισμένη Στη συγχρονισμένη επικοινωνία μια διεργασία στέλνει ένα αίτημα επικοινωνίας και από την άλλη πλευρά μια άλλη διεργασία σε ένα άλλο κόμβο (εξυπηρετητής) αναλαμβάνει να εξυπηρετήσει το αίτημα. Το πρωτόκολλο της επικοινωνίας RPC απεικονίζεται στην εικόνα 6.

Μέχρι να ολοκληρωθεί η ανταλλαγή αυτή, δηλαδή να πάρει απάντηση ή σφάλμα, η διεργασία μένει σε κατάσταση αναμονής και περιμένει, χωρίς να μπορεί να τερματίσει ή να επιστρέψει τους πόρους που χρησιμοποιεί πίσω στο σύστημα.

Η διαφορά με τη χρήση του RPC είναι ότι μια απομακρυσμένη διεργασία μπορεί η ίδια να καλέσει μια τοπική διεργασία στον ενδιάμεσο κόμβο ώστε αυτή να τερματίσει και η τοπική να καλέσει τον επόμενο κόμβο στόχο. Όταν σε εκείνο τον στόχο μια άλλη διεργασία ολοκληρώσει την εξυπηρέτηση, τότε θα απαντήσει στην τοπική και η τοπική θα στείλει πίσω την απάντηση στην αρχική απομακρυσμένη διεργασία. Με τον τρόπο αυτό επιτυγχάνεται διαφάνεια (transparency) στην κλήση , καθώς αποκρύπτονται οι λεπτομέρειες της διαδικασίας της τοπικής κλήσης στο χρήστη.



Εικόνα 6 - Remote Call Procedure Communication [6]

Με την συγχρονισμένη επικοινωνία μια διεργασία πάντα θα πάρει απάντηση, ωστόσο πολλές φορές μπορεί να δημιουργηθεί συμφόρηση (bottleneck) εάν τα αιτήματα ξεπεράσουν έναν αριθμό ενώ δημιουργείται και πρόβλημα στους πόρους καθώς δεν επιστρέφονται μέχρι να τερματίσει η διεργασία.

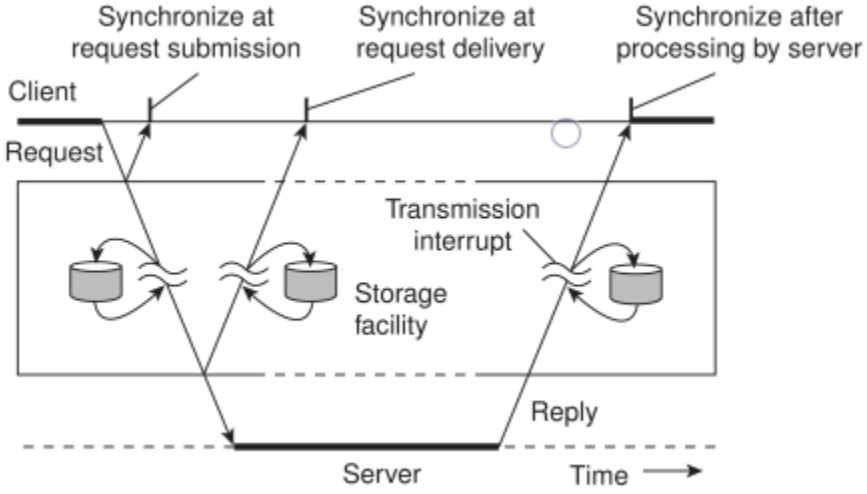
3.3.3 Message-Oriented- Middleware (MOM)

Υπάρχουν όμως και πολλές φορές που η χαμηλού επιπέδου επικοινωνία δεν εξυπηρετεί απόλυτα , συνήθως λόγω της έλλειψης διαφάνειας. Σε αυτές τις περιπτώσεις

χρησιμοποιούμε ένα υψηλού-επιπέδου ενδιάμεσο σύστημα (middleware), βασισμένο σε ένα μοντέλο ουράς μηνυμάτων (messaging queue). Βασικό χαρακτηριστικό της επικοινωνίας αυτής ότι η επικοινωνία είναι ασύγχρονη. Η αρχιτεκτονική της απεικονίζεται στην εικόνα 7.

Σε αντίθεση με την συγχρονισμένη επικοινωνία, εδώ η διεργασία στέλνει το μήνυμα και δεν περιμένει απάντηση, αλλά όταν κάποια στιγμή είναι δυνατό το σύστημα θα αποστείλει την απάντηση. Τα μηνύματα πηγαίνουν στο middleware σύστημα, ένα ενδιάμεσο σύστημα, και συνήθως αποθηκεύονται εκεί και απο εκεί προωθούνται πια νέα μηνύματα στον τελικό παραλήπτη κόμβο.

Μόλις το αρχικό μήνυμα φτάσει στο middleware , συνήθως μια ουρά προτεραιότητας FIFO (First-In-First-Out) , τότε σε 1ο χρόνο η αρχική διεργασία τερματίζει και σε 2ο το middleware δοκιμάζει να στείλει στον τελικό προορισμό το μήνυμα. Σε περίπτωση που δεν τα καταφέρει , μπορεί να ξαναδοκιμάσει βάσει της πολιτικής επαναπροσπαθειών που έχει οριστεί.



Εικόνα 7 - Asynchronous Communication [6]

3.4 Μοντέλα Ασφαλείας και Πολιτικές Ασφάλειας

Η επικοινωνία όπως είδαμε σε ένα κατανεμημένο σύστημα είναι πολύπλοκη, ωστόσο είναι κάτι που πρέπει να γίνεται με τέτοιο τρόπο που να διασφαλίζονται η **Ακεραιότητα**, η **Εμπιστευτικότητα** και η **Διαθεσιμότητά** της [9]. Αυτό είναι απαραίτητο ώστε να είμαστε σίγουροι ότι τα μηνύματα που ανταλλάσσουν οι κόμβοι ενός κατανεμημένου συστήματος γίνονται με απόλυτη ασφάλεια.

Διακρίνουμε 3 βασικές απειλές ασφαλείας [17]:

- Μη εξουσιοδοτημένη αποκάλυψη/κοινοποίηση πληροφορίας

- Μη εξουσιοδοτημένη τροποποίηση πληροφορίας
- Μη εξουσιοδοτημένη άρνηση της χρήσης της πληροφορίας.

Είναι φανερό ότι τον κοινό παρονομαστή στις απειλές αυτές αποτελεί η μη εξουσιοδότηση. Για να μπορέσει ένα σύστημα να ανταποκριθεί σε αυτές τις προκλήσεις και δυνητικές απειλές, πρέπει να υπάρχει μια πολιτική ασφαλείας. Η πολιτική ασφαλείας αποτελείται ουσιαστικά από τις προδιαγραφές βάσει των οποίων θα υλοποιηθούν οι μηχανισμοί ασφάλειας οι οποίοι θα έρθουν να εφαρμόσουν την πολιτική.

Η πολιτική ασφαλείας επικεντρώνεται στις 4 ακόλουθες κατηγορίες

- Κρυπτογράφηση (Encryption)
- Αυθεντικοποίηση (Authentication)
- Εξουσιοδότηση (Authorization)
- Παρακολούθηση και Καταγραφή (Monitoring and Auditing)

Κρυπτογράφηση: αποτελεί θεμελιώδες μέρος της ασφάλειας όλων των υπολογιστικών συστημάτων. Μέσω της κρυπτογράφησης μπορούμε να μετατρέψουμε δεδομένα σε κάτι που ένας επίδοξος επιτιθέμενος δεν μπορεί να αναγνωρίσει, άρα και να εκμεταλλευτεί. Παράλληλα μας εξασφαλίζει οτι τα δεδομένα μας δεν μπορούν να τροποποιηθούν. Αρα μας εξασφαλίζει ακεραιότητα και εμπιστευτικότητα. Με τον όρο κρυπτογραφία εννοούμε όλα εκείνα τα μέσα που θα μας επιτρέψουν να κάνουμε κρυπτογράφηση: Τις μορφές κρυπτογράφησης: Συμμετρική & Ασύμμετρη, τα ιδιωτικά και δημόσια κλειδιά και τους αλγόριθμους κρυπτογράφησης και αποκρυπτογράφησης.

Αυθεντικοποίηση: είναι χρήσιμη ώστε να μπορέσουμε να επιβεβαιώσουμε ότι ένας χρήστης που ισχυρίζεται ότι είναι αυτός που είναι βάσει των αναγνωριστικών που έχει δώσει, είναι όντως αυτός και όχι κάποιος που τον υποδύεται. Η αυθεντικοποίηση έχει να κάνει με το *«Ποιός»*.

Εξουσιοδότηση: έρχεται αμέσως μετά την αυθεντικοποίηση ως επόμενο βήμα. Το σύστημα μόλις αναγνωρίσει το χρήστη θα επιβεβαιώσει αν ο συγκεκριμένος χρήστης έχει δικαιοδοσία να έχει πρόσβαση στα δεδομένα που ζητά. Η εξουσιοδότηση έχει να κάνει με το *«Τι»* μπορεί να προσπελάσει ένας αυθεντικοποιημένος χρήστης.

Με τα **αρχεία παρακολούθησης και καταγραφής** το σύστημα καταγράφει όλη τη συμπεριφορά των αυθεντικοποιημένων χρηστών. Το σύστημα γνωρίζει την ακριβή ώρα, βάσει χρονοσφραγίδας, της εγγραφής του στο σύστημα, της εκάστοτε προσπάθειας επιτυχημένης και ανεπιτυχούς εισόδου του στο σύστημα. Το σύστημα κρατάει αρχεία καταγραφής για κάθε του ενέργεια: ποιά δεδομένα ζήτησε να δεί, ποιά ζήτησε να τροποποιήσει, όπως και το αν ζήτησε δεδομένα στα οποία δεν έχει δικαιοδοσία. Όλος αυτός ο πληροφοριακός πλούτος μας βοηθάει να καταλάβουμε τη συμπεριφορά των χρηστών μέσα στο σύστημα, ειδικά όταν έχουμε σφάλματα, και να κάνουμε ιχνηλάτηση (tracing) και να δούμε απο που προέκυψε κάποιο πρόβλημα. Μας βοηθάει επίσης να δούμε συμπεριφορές και να φτιάξουμε μοτίβα ώστε να ισχυροποιήσουμε τους κανόνες ασφαλείας.

3.5 Αρχές Ασφαλείας

Ένα κατανεμημένο σύστημα παρά την πολυπλοκότητα του δεν θα πρέπει να θυσιάζει κάποιες βασικές αρχές ασφαλείας. Οι αρχές αυτές θα πρέπει να αποτελούν μέρος του σχεδιασμού του συστήματος και όχι να ενσωματωθούν στο τέλος (on-top) [14], [15].

3.5.1 Fail-Safe Defaults

Κάθε σετ default στοιχείων σε περίπτωση που δεν τροποποιηθούν. Για παράδειγμα όταν ένας χρήστης εγγραφεί σε ένα σύστημα ίσως το σύστημα του δημιουργεί λογαριασμό με κάποια default στοιχεία που μπορεί να έχει ένας admin χρήστης username: admin, password: admin.

3.5.2 Open Design

Είναι σημαντικό κάθε στοιχείο ενός κατανεμημένου συστήματος από πλευράς ασφαλείας να είναι ανοιχτό προς αξιολόγηση. Υπάρχει μια πολύ μεγάλη κοινότητα και η έκθεση των πρακτικών που χρησιμοποιούνται για την ασφάλεια του συστήματος μπορεί να αξιολογηθεί από την κοινότητα [7]. Αν κάτι εμπεριέχει κάποια ευπάθεια ή κενό ασφαλείας μπορεί να έχει ήδη επισημανθεί από κάποια άλλα μέλη και χρήστες προηγουμένως, οπότε μας βοηθά να αναβαθμίσουμε το συγκεκριμένο σημείο με μια λύση-προσέγγιση που ήδη έχει δοκιμαστεί και κριθεί επιτυχώς.

3.5.3 Separation of privilege

Η εγγύηση που πρέπει να υπάρχει στο σύστημα οτι ποτέ μια και μοναδική οντότητα δεν θα έχει τον αποκλειστικό έλεγχο σε κρίσιμα στοιχεία του. Για παράδειγμα αν υπάρχει ένα πολύ σημαντικό αρχείο που θα πρέπει να κρυπτογραφηθεί, αυτό θα πρέπει να γίνει με τη συμβολή κλειδιών που ανήκουν σε τουλάχιστον 2 διαφορετικούς χρήστες. Αντίστοιχα το να πρέπει να απενεργοποιηθούν κρίσιμες υπηρεσίες του συστήματος θα πρέπει να δίνεται έγκριση ταυτόχρονα από 2 τουλάχιστον διαχειριστές με ξεχωριστούς διακόπτες αποδοχής.

3.5.4 Least privilege

Κάθε χρήστης θα πρέπει να μπορεί να χρησιμοποιεί το σύστημα με τις λιγότερες δυνατές εξουσιοδοτήσεις. Δηλαδή ποτέ να μη δίνεται μεγαλύτερη εξουσιοδότηση από αυτή που είναι απαραίτητη για το ρόλο και τις εργασίες του. Αν απαιτηθεί για κάποια εργασία μεγαλύτερη , θα πρέπει να μπορεί να την εκτελέσει με δικαιώματα διαχειριστή και με

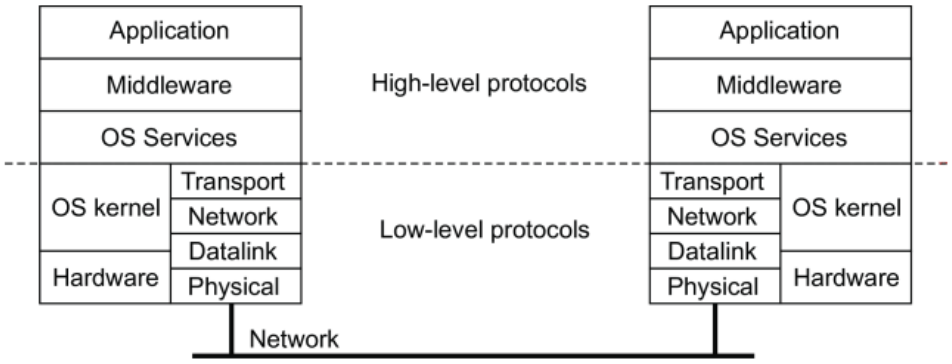
παροχή αναγνωριστικών και όχι να αναβαθμιστεί ο ρόλος του και η εξουσιοδότηση του μόνιμα.

3.5.5 Least common mechanism

Ο μηχανισμός αυτός του συστήματος θα πρέπει να σχεδιαστεί με τέτοιο τρόπο που αν χρειαστεί άλλα συστήματα να τον ενσωματώσουν τότε θα πρέπει να μπορεί να υλοποιηθεί με τον ίδιο τρόπο και στα υπόλοιπα. Αυτό συμβάλει σε μεγάλο βαθμό στην απλότητα της υλοποίησης και στην ευκολία συντήρησης των μηχανισμών.

3.6 Υλοποίηση Μηχανισμών Ασφαλείας

Μια πολύ σημαντική απόφαση είναι το που θα πρέπει να υλοποιηθούν οι μηχανισμοί ασφαλείας, σε ποιές ακριβώς περιοχές του συστήματος. Συνηθίζεται μια δόμηση – οργάνωση του κατανεμημένου συστήματος βάσει των πρωτοκόλλων χαμηλού και υψηλού επιπέδου, ώστε να καταστεί πιο εύκολη η απόφαση [6].



Εικόνα 8 - Λογική οργάνωση ενός κατανεμημένου συστήματος σε επίπεδα [6]

Στο δίκτυο που σηματοδοτείται ως πρωτόκολλο χαμηλού επιπέδου θα ακολουθηθεί λύση επιπέδου ασφαλείας δικτύων όπως για παράδειγμα εφαρμογή VPN (Virtual Private Network). Ένα κρυπτογραφημένο τούνελ δηλαδή μεταξύ 2 κόμβων στα 2 δίκτυα (host, remote) ώστε να ανταλλάσσουν μέσα από την προστασία του πληροφορίες οι οποίες στον έξω κόσμο εμφανίζονται κρυπτογραφημένες.

Πιο πάνω στο επίπεδο μεταφοράς (Transport) υλοποιείται το TLS (Transport Layer Security) και παρέχει κρυπτογράφηση στα website που χρησιμοποιούν το πρωτόκολλο HTTPS, ώστε η ανταλλαγή πληροφοριών μεταξύ πελάτη και εξυπηρετητή να είναι κρυπτογραφημένη και ασφαλής.

Επίσης μια προσέγγιση που αυξάνεται είναι πια το να προσφέρουν οι ίδιες οι εφαρμογές (Application level) end-to-end ασφάλεια, με το να κρυπτογραφούν τα μηνύματα πριν τα στείλουν στον έξω κόσμο και ξανά πάλι αποκρυπτογράφηση στον πελάτη ή στον εξυπηρετητή.

4. Επιθέσεις σε κατανεμημένα συστήματα

Οι επιθέσεις είναι το επόμενο βήμα των κακόβουλων χρηστών πάνω στις ευπάθειες και τα κενά ασφαλείας του συστήματος που περιγράφονται στις προηγούμενες ενότητες [16]. Ακολουθούν κάποιες από τις πιο σημαντικές μορφές επιθέσεων που μπορεί να δεχθεί ένα κατανεμημένο σύστημα (Peer-to-peer) [21].

Attack	Availability	Integrity	Confidentiality	Functionality
DoS/DDoS	✓	✗	✗	P-OP
Collusion	✓	✓	✓	P-OP
Pollution	✗	✓	✗	P-DS
White washing & censorship	✓	✓	✗	P-DS
Routing	✓	✓	✗	P-DS
Buffer map cheating	✓	✓	✗	P-OP
Sybil	✓	✗	✓	P-OP
Eclipse	✓	✓	✓	P-DS, P-OP

Εικόνα 9 - Distributed Systems Attacks & Security Goals (P2P) [17]

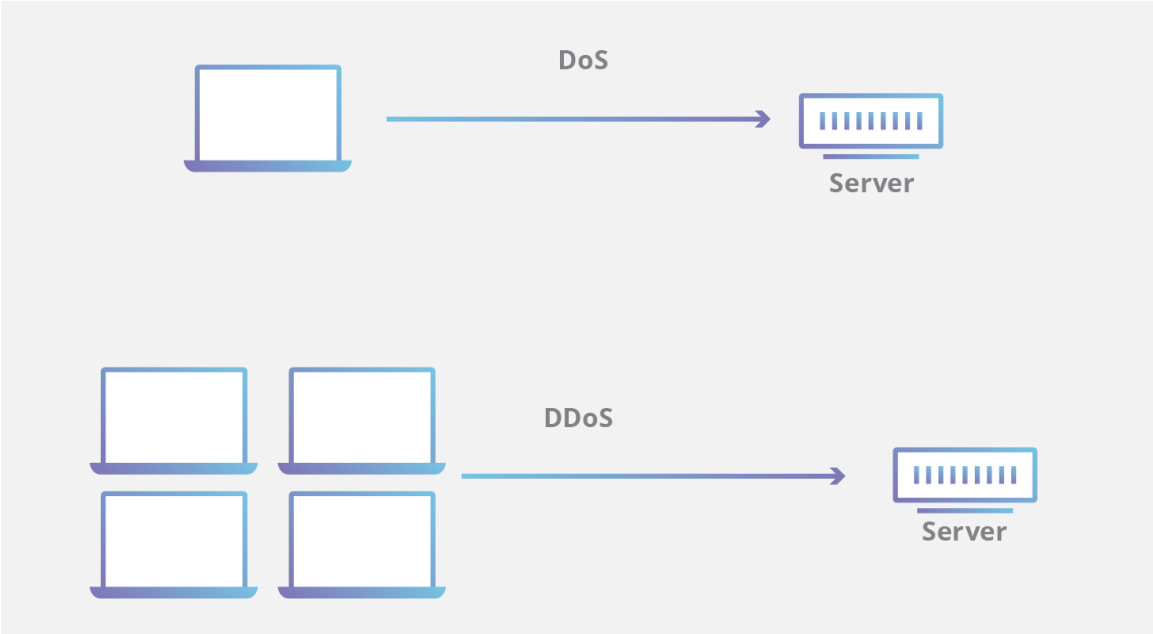
Βάσει της εικόνας 9 βλέπουμε ποιούς βασικούς πυλώνες του 3-πτύχου Διαθεσιμότητα-Ακεραιότητα-Εμπιστευτικότητα ακουμπάει κάθε επίθεση

- Denial of Service (DoS) - Distributed Denial of Service (DDoS)
- Collusion
- Pollution
- White washing (Censorship)
- Routing
- Buffer map cheating
- Sybil
- Eclipse

4.1 Denial of Service (DoS) - Distributed Denial of Service (DDoS)

Είναι η μορφή επίθεσης κατά την οποία το σύστημα βομβαρδίζεται με αιτήματα σε πολύ μικρό χρονικό διάστημα [18] ώστε να μη μπορέσει αυτό και οι πόροι του να ανταποκριθούν στον αριθμό των αιτημάτων και μοιραία να πέσει [23]. Στοχεύει αποκλειστικά στην διαθεσιμότητα της υπηρεσίας. Πολλές φορές δεν αποτελεί αποκλειστικά πρόβλημα του συστήματος αλλά κακής υλοποίησης κάποιας εφαρμογής [24]. Η διαφορά του DoS με το

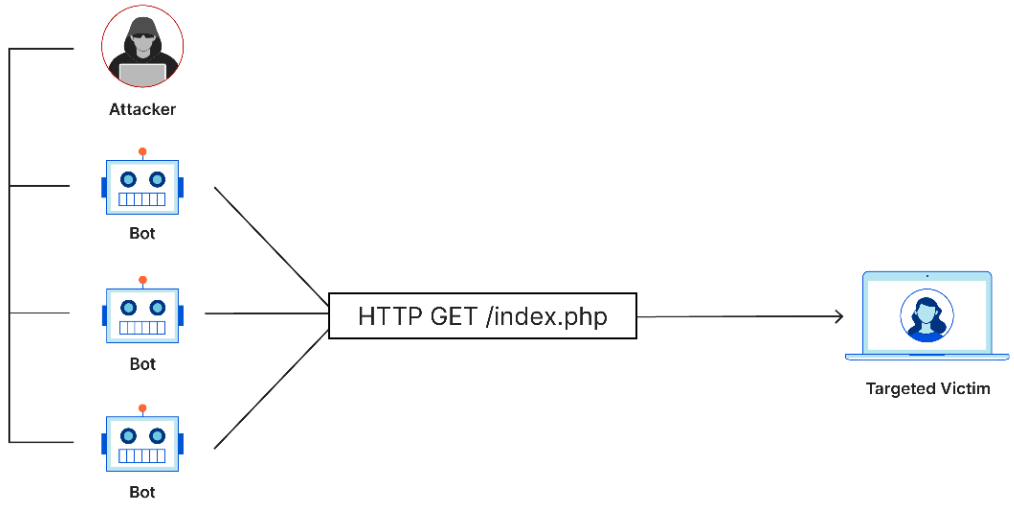
DDoS είναι οτι μια επίθεση DoS γίνεται απο ένα υπολογιστικό σύστημα ενώ οι επιθέσεις DDoS απο μια ομάδα ή συστάδα υπολογιστικών συστημάτων (συνήθως κάποιο δίκτυο botnet).



Εικόνα 10 - DoS vs DDoS επιθέσεις [21]

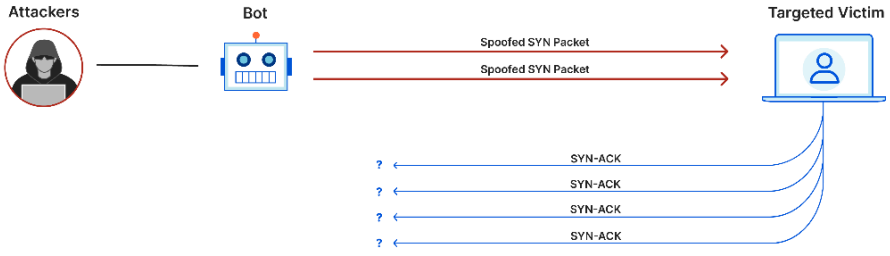
Οι επιθέσεις DDoS είναι συνήθως πολύ πιο εξεζητημένες τεχνικά και έχουν μεγαλύτερο και σοβαρότερο αντίκτυπο.

DDoS Application Layer Attack - HTTP Flood: Αφορά το ανώτερο επίπεδο 7, το επίπεδο εφαρμογής. Απλή σε λογική επίθεση αλλά πολύ αποτελεσματική: μια ιστοσελίδα ζητείται ταυτόχρονα απο πάρα πολλούς clients, κάτι που θα κάνει τον διεκομιστή της σελίδας να μην μπορεί να ανταποκριθεί στα αιτήματα και να πέσει.



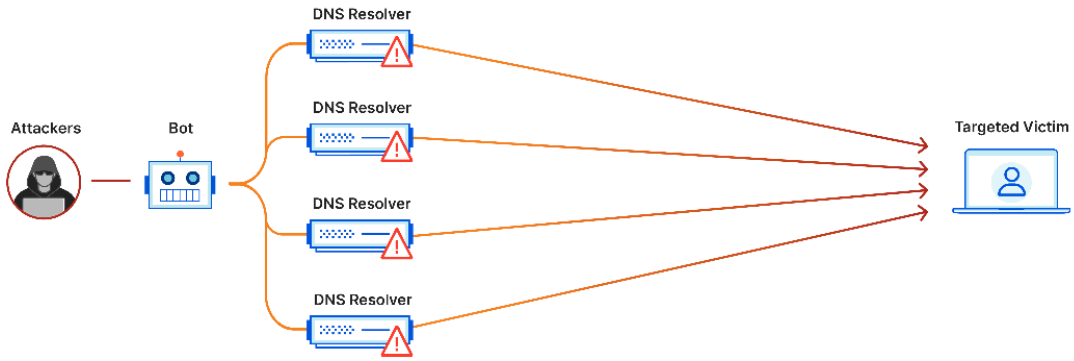
Εικόνα 11 - DDoS Application Layer Attack – HTTP flood [22]

DDoS Protocol Attack - SYN Flood: Αφορά τα επίπεδα 3 και 4, Επίπεδα δικτύου και μεταφοράς. Οι επιθέσεις αυτές είναι γνωστές και ως State-Exhaustion (εξάντληση κατάστασης πόρων). Χαρακτηριστική είναι η επίθεση SYN flood. Ο εξυπηρετητής βομβαρδίζεται απο αρχικά αιτήματα SYN με το πρόσχημα οτι θα γίνει χειραψία χωρίς ποτέ να ακολουθήσουν τα υπόλοιπα βήματα της χειραψίας του TCP (TCP handshake) οπότε περιμένοντας αυτά και ενώ κατακλύζεται παράλληλα απο ολοένα και περισσότερα νέα SYN αιτήματα, πέφτει.



Εικόνα 12 - DDoS Protocol Attack - SYN Flood [22]

DDoS Volumetric Attack (Ογκομετρική) – DNS Amplification: Η κατηγορία των επιθέσεων αυτών έχει σαν στόχο να δημιουργήσει συμφόρηση και να απορροφήσει όλο το bandwidth μεταξύ του δικτύου και του συστήματος στόχου. Ο σκοπός είναι να σταλούν αιτήματα που πριν τη διαδικασία επεξεργασίας τους θα αποκρύπτεται το πραγματικό τους μέγεθος, με συνέπεια να καταναλωθεί όλο το διαθέσιμο bandwidth. Η επίθεση DNS Amplification γίνεται με πολύ μικρό κόστος: Χρησιμοποιεί DNS resolvers και κάνει ερωτήματα για όλες τις διαθέσιμες IP αλλά η IP απάντησης ανήκει στο σύστημα στόχο. Έτσι οι DNS resolvers απαντούν στο σύστημα στόχο και αυτό δέχεται πολύ μεγάλο αριθμό απαντήσεων με πολύ μεγάλο μέγεθος με συνέπεια να πέσει.

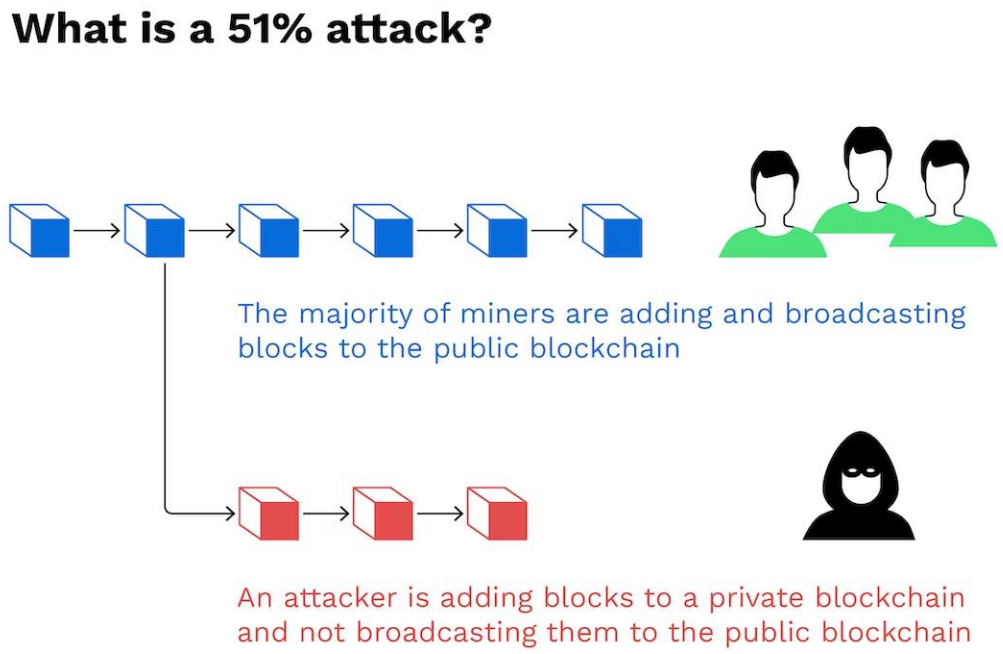


Εικόνα 13 - DDoS - Volumetric Attack - DNS Amplification [22]

4.2 Collusion Attack

Ο συντονισμός πολλών κόμβων ώστε να εφαρμόσουν μια στρατηγική που θα πλήξει κάποιες υπηρεσίες του συστήματος και κατ' επέκταση το ίδιο το σύστημα. Τυπικό παράδειγμα η προσπάθεια παράκαμψης των μηχανισμών ελέγχου. Οι επιθέσεις **Sybil** , **Eclipse** είναι βασισμένες πάνω σε αυτή τη λογική. Σε συστήματα blockchain η επίθεση του **51%** αποτελεί ίσως την πιο επίφοβη μέθοδο επίθεσης με το σκεπτικό οτι θα είναι και η πιο επιζήμια. Ωστόσο μέχρι σήμερα δεν υπάρχουν ακόμη καταγεγραμμένα περιστατικά επιτυχούς εκτέλεσης αυτής της επίθεσης.

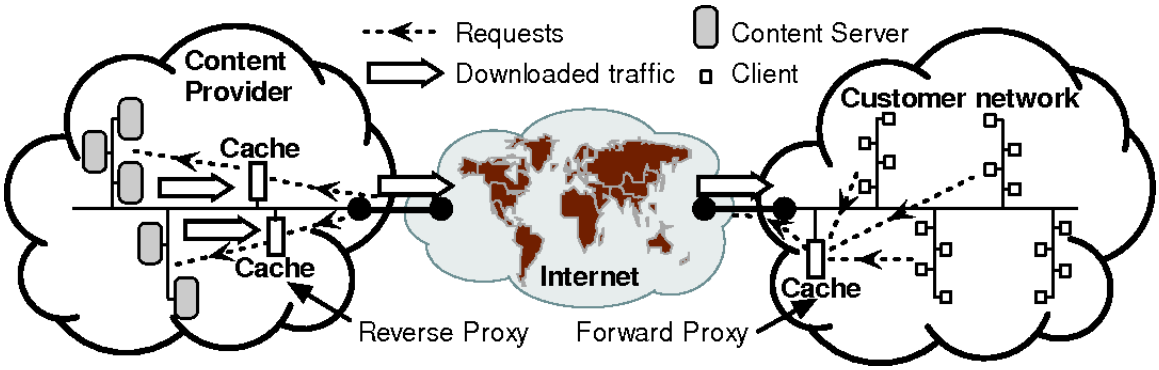
51% Attack: Ο επιτιθέμενος χρησιμοποιώντας αρκετούς πόρους καταφέρνει να αποκτήσει τον έλεγχο του 51% του δικτύου blockchain και να προβεί σε πολύ επικίνδυνες ενέργειες όπως να ξαναγράψει το ιστορικό των συναλλαγών [27].



Εικόνα 14 - 51% Attack [24]

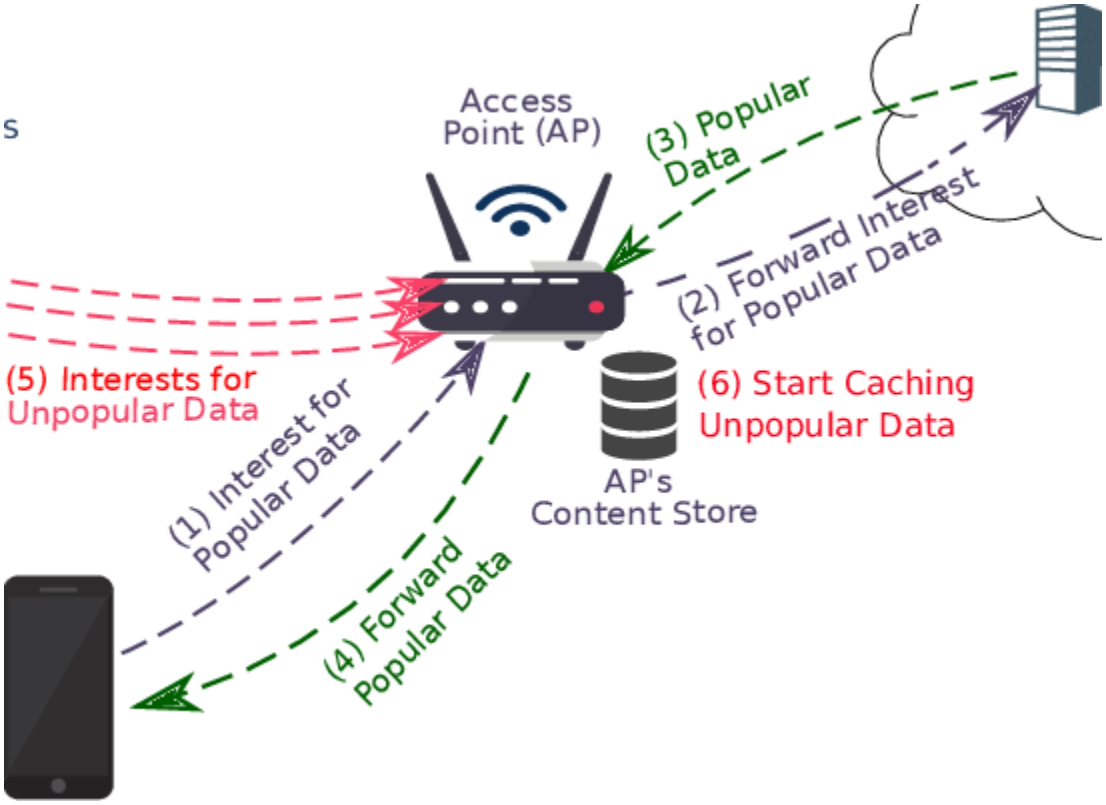
4.3 Pollution Attack

Έχουν ως στόχο να προσβάλλουν την ακεραιότητα και να τροποποιήσουν δεδομένα, προσθέτοντας πληροφορία η οποία δεν είναι πραγματική. Τυπικό παράδειγμα οι ad-ware επιθέσεις στους λογαριασμούς των emails, όπου ένας χρήστης που έχει μολυνθεί εξαπλώνει την επίθεση σε όλο τον κατάλογο των peers του. Παραδείγματα επιθέσεων pollution είναι επίσης η επίθεση **cache pollution** όπως και η **content poisoning** επίθεση.



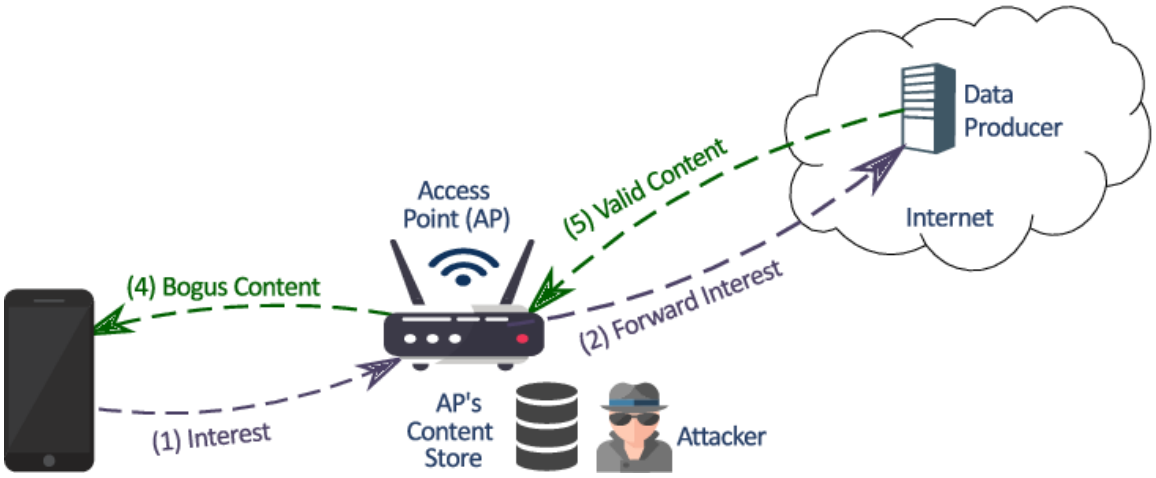
Εικόνα 15 - Pollution Attack [29]

Cache Pollution: Οι πληροφορίες που υπάρχουν στην cache μνήμη της εφαρμογής μολύνονται (αλλοιώνονται) ώστε ο χρήστης που θα ζητήσει πληροφορίες και θα ανταποκριθεί η cache μνήμη θα πάρει αλλοιωμένη πληροφορία.



Εικόνα 16 - Cache Pollution [26]

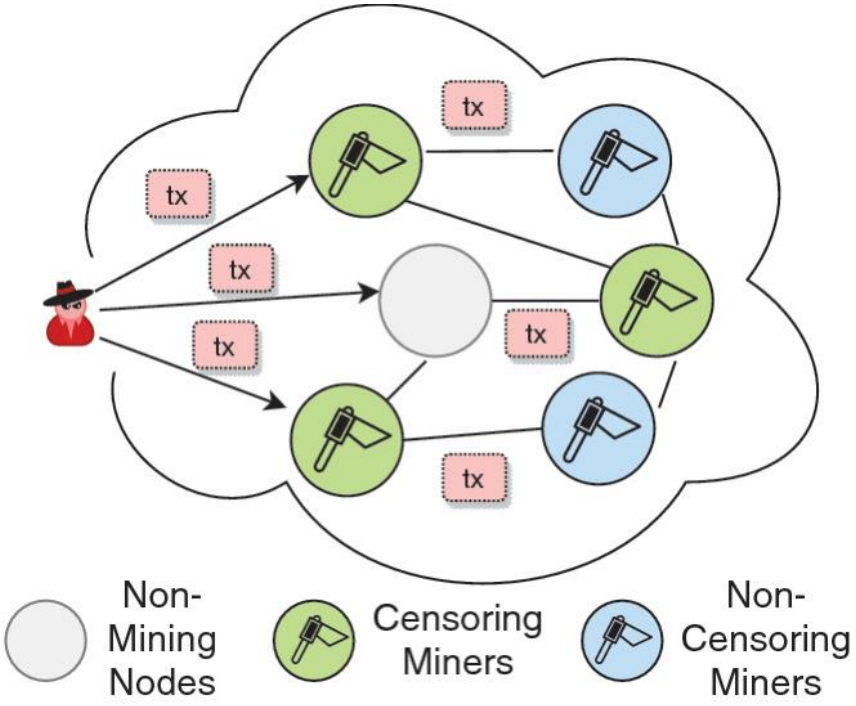
Content Poisoning: Αντίστοιχα με την αλλοίωση της πληροφορίας που βρίσκεται στην cache μνήμη έτσι αλλοιώνεται και το περιεχόμενο.



Εικόνα 17 - Content poisoning [26]

4.4 White washing (Censorship)

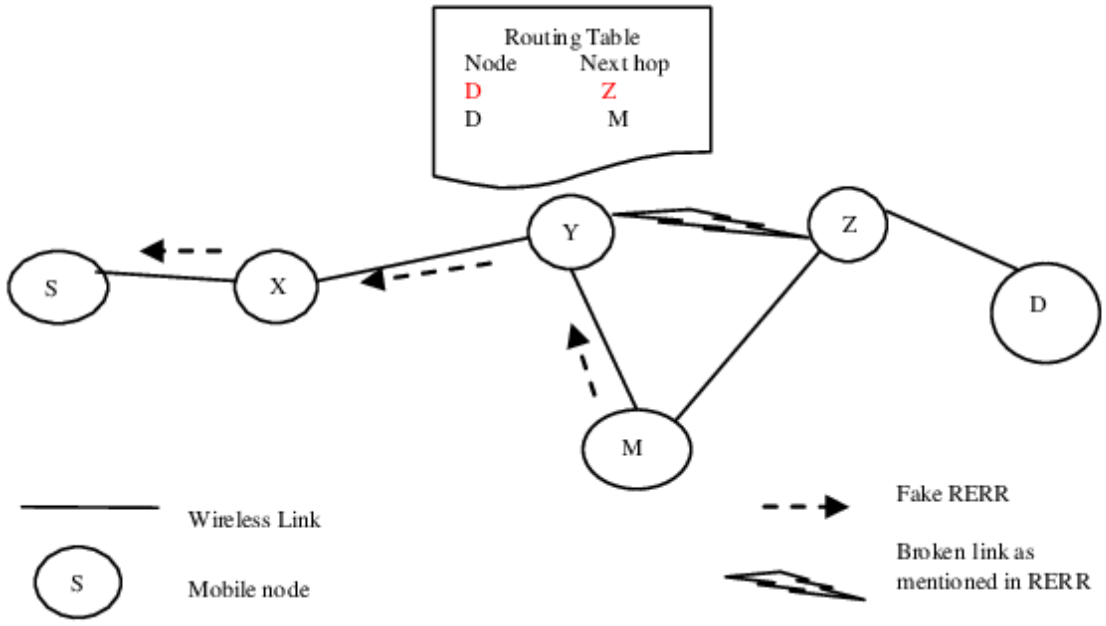
Έχουν ως στόχο να δημιουργήσουν προβλημα στην ακεραιότητα και τη διαθεσιμότητα των συστημάτων τροποποιώντας και διαγράφοντας αρχεία ή αρνούνται εξουσιοδότηση και προσπέλαση του χρήστη στα δεδομένα.



Εικόνα 18 - Blockchain Censorship [27]

4.5 Routing Attack

Έχουν ως στόχο να προσβάλλουν την διαθεσιμότητα του δικτύου, επηρεάζοντας τη δυνατότητα ενός κόμβου να στέλνει και να λαμβάνει μηνύματα , είτε καθυστερώντας τον είτε καταστρέφοντας πλήρως τα μηνύματα. Διάσημες παραλλαγές της επίθεσης είναι το **Routing Table Poisoning** [32] κατά την οποία ο επιτιθέμενος τροποποιεί τους πίνακες δρομολόγησης των υπολοίπων κόμβων (**Routing Tables**) καθώς και η επίθεση **Attraction and Repulsion** κατά την οποία αυξάνουν (attraction) ή μειώνουν (repulsion) την πιθανότητα επιλογής του κόμβου στον πίνακα δρομολόγησης.



Εικόνα 19 - Routing Table Poisoning [28]

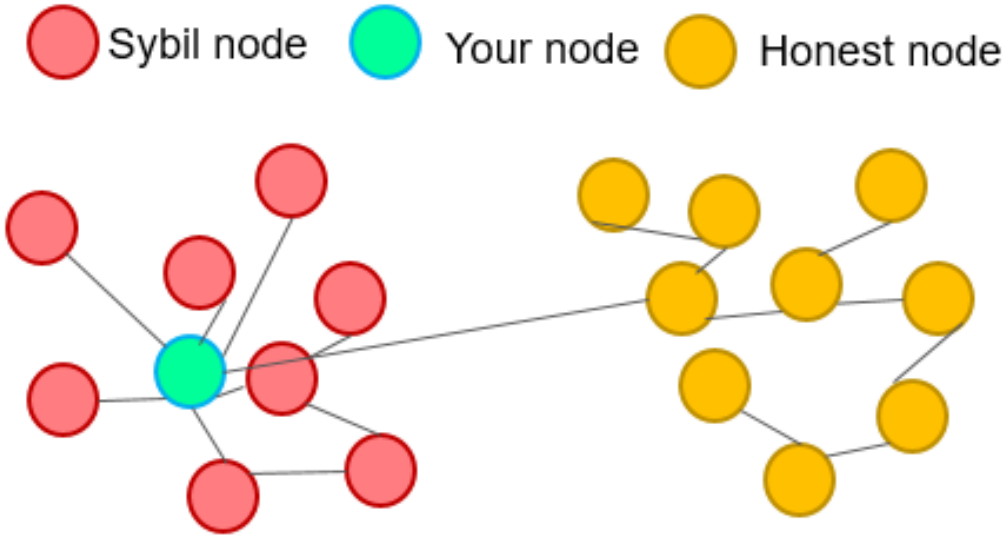
4.6 Buffer map cheating

Έχει ως στόχο να μειώσει τη διαθεσιμότητα των δικτύων, ειδικά αυτών που προορίζονται για media streaming. Επιτίθεται στη δυνατότητα του outgoing φορτίου της κίνησης του κόμβου.

4.7 Sybil

Οι επιθέσεις Sybil έχουν ως στόχο να προσβάλλουν τη διαθεσιμότητα και την εμπιστευτικότητα του δικτύου. Προσπαθούν να εισάγουν μια σειρά από κόμβους οι οποίοι

είναι στον έλεγχο των επιτιθέμενων, επηρεάζοντας με αυτό τον τρόπο την εκλογή τους και επηρεάζοντας στη συνέχεια το σύστημα εφόσον έχουν εκλεγεί [33].

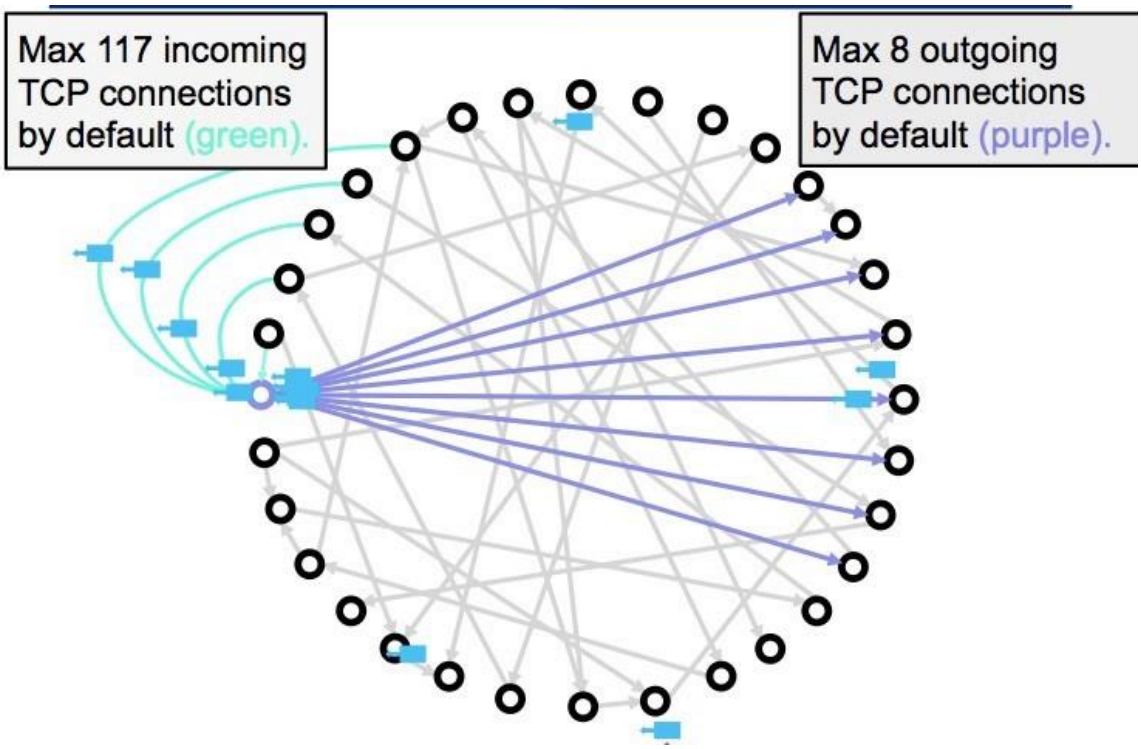


Εικόνα 20 - Sybil Attack [30]

4.8 Eclipse

Έχουν ως στόχο να μειώσουν τη διαθεσιμότητα, την ακεραιότητα και την εμπιστευτικότητα του δικτύου. Ιδανικά ένας μη-κακόβουλος κόμβος περικυκλώνεται από κακόβουλους οι οποίοι του αποκρύπτουν μερικώς ή πλήρως την εικόνα του υπόλοιπου συστήματος. Έτσι κάνουν spoof τις εξωτερικές αλληλεπιδράσεις αυτού του κόμβου. Αυτή αποτελεί μια ιδιαίτερα εξεζητημένη μορφή επίθεσης καθώς συνδυάζει πολλές μορφές από τις προηγούμενες.

Όπως φαίνεται στην ακόλουθη εικόνα, σε ένα δίκτυο blockchain ένας κόμβος μπορεί να έχει μέχρι 8 εξερχόμενες και μέχρι 117 εισερχόμενες συνδέσεις TCP απο άλλους κόμβους. Ο στόχος του επιτιθέμενου είναι να πιέσει έναν κόμβο έτσι ώστε να δέχεται εισερχόμενες συνδέσεις απο δικούς του κατασκευασμένους ή ελεγχόμενους κόμβους ώστε να μπορεί να τον επηρεάσει.



Εικόνα 21 - Eclipse Attack [31]

5. Τεχνικές & Εργαλεία εντοπισμού ευπαθειών σε κατανεμημένα συστήματα

Υπάρχουν διάφορες τεχνικές και εργαλεία ώστε να μπορέσουμε να εντοπίσουμε ευπάθειες σε ένα κατανεμημένο σύστημα [36].

5.1 Τεχνικές

Υπάρχει ένα σύνολο τεχνικών το οποίο μας βοηθάει να εντοπίσουμε ευπάθειες σε ένα σύστημα. Κάποιες έχουν πιο θεωρητική προσέγγιση και κάποιες απαιτούν πιο εξειδικευμένες τεχνικές δυνατότητες. Ωστόσο είναι πολύ σημαντικό να γίνεται συνδυαστική χρήση όλων των ακολούθων τεχνικών ώστε να επιτύχουμε ποιοτικότερο αποτέλεσμα.

Threat Modelling: Είναι η συστηματική διαδικασία αναγνώρισης και ανάλυσης όλων των δυνητικών απειλών και ευπαθειών του συστήματος. Μας βοηθάει να προτεραιοποιήσουμε τα ρίσκα και να σχεδιάσουμε μέτρα αντιμετώπισης.

Penetration Testing: Είναι η προσομοίωση επίθεσης στο σύστημα ώστε να ανιχνεύσουμε και να εκμεταλλευτούμε τις αδυναμίες του. Μας βοηθάει να αξιολογήσουμε την αποτελεσματικότητα και το βαθμό ασφαλείας του συστήματος καθώς και να βρούμε τα κενά ή προβλήματα στην υλοποίηση που επιτρέπουν σε έναν επιτιθέμενο να εισχωρήσει στο σύστημα μας. Μπορεί να γίνει με διάφορες παραλλαγές (black-box, white-box, gray-box)

Fuzz Testing: Βασίζεται στην τεχνική του να στέλνουμε τυχαία ή κακοσχηματισμένα input στο σύστημα ώστε να δούμε πως θα συμπεριφερθεί και αν κάτι μπορεί να ενεργοποιήσει μια συμπεριφορά που δεν έχουμε προβλέψει. Μας βοηθά να ανιχνεύσουμε bugs, σφάλματα , memory leaks και ευπάθειες που μπορούν να γίνουν αντικείμενο εκμετάλλευσης από κακόβουλους χρήστες. Μπορεί να πραγματοποιηθεί σε πολλά επίπεδα: εφαρμογής , δικτύου και πρωτοκόλλου και με πολλές στρατηγικές (mutation, generation, feedback-driven fuzzing)

Compliance Testing: Είναι μια τεχνική testing που επιβεβαιώνει οτι το σύστημα πληροί τις συνθήκες ασφαλείας και του κανονιστικού πλαισίου που έχουμε θέσει. Μας βοηθά να διασφαλίσουμε οτι το σύστημα εναρμονίζεται με τις σωστές πρακτικές και τις οδηγίες του κλάδου και οτι συμμορφώνεται με το θεσμικό πλαίσιο. Συνήθως διεξάγεται απο εξωτερικούς συνεργάτες.

Monitoring – Logging: Είναι η συλλογή και ανάλυση όλων των αρχείων καταγραφής για τα δεδομένα και τα γεγονότα που συμβαίνουν μέσα στο σύστημα κάθε στιγμή και έχουν να κάνουν με τη συμπεριφορά και τις επιδόσεις του συστήματος. Μας βοηθά να έχουμε μια εικόνα για τα επίπεδα ασφάλειας , αξιοπιστίας και αποδοτικότητας και να αναγνωρίσουμε άμεσα τυχόν ανωμαλίες ή συμβάντα. Γίνεται με διάφορα εργαλεία και

χρησιμοποιώντας μετρικές, δείκτες, πίνακες ελέγχου, γραφήματα, dashboards, και αναφορές ώστε να μπορούμε πάντα να έχουμε στοιχεία συγκρίσιμα. Γενικά μπορεί να είναι proactive, reactive και predictive.

5.2 Εργαλεία

Με τον όρο Vulnerability Management Tools (VM) εννοούμε όλα εκείνα τα εργαλεία που μας βοηθούν να ανιχνεύσουμε αδυναμίες και ευπάθειες στο σύστημα μας, τις οποίες κάποιος κακόβουλος μπορεί να εκμεταλλευθεί. Αξιολογούν το δίκτυο και μας προτείνουν μέτρα εξομάλυνσης (mitigation).

Dynamic Discovery and Inventory: Παρέχουν ευρετήριο για όλες τις εφαρμογές, υπηρεσίες, χρήστες και οτιδήποτε έχει γίνει deploy ή είναι συνδεδεμένο με το σύστημα.

Asset Visibility: Σκανάρει ότι asset υπάρχει (hardware, iot) συνδεδεμένο στο σύστημα και τρέχει κάποιους ελέγχους επικινδυνότητας.

Host Assets Organization: Σκανάρουν το δίκτυο του συστήματος και βλέπουν ανοιχτές πόρτες και υπηρεσίες. Βάσει κανόνων υποδεικνύει ποια σημεία είναι λειτουργικά και αποκαλύπτει σημεία που μπορεί να αφήσουν το δίκτυο ανοιχτό και ακάλυπτο (ένας web server πχ που δεν χρησιμοποιείται πλέον αλλά εξακολουθεί να λειτουργεί και να ακούει σε μια πόρτα).

Attack Vector Coverage: Σκανάρουν και ψάχνουν για πιθανά διανύσματα και επιφάνεια επίθεσης. Ψάχνουν για κακή παραμετροποίηση και λάθη σε εφαρμογές, δίκτυο, weak passwords κλπ.

Real-Time Monitoring and Analysis: Προσφέρουν μια πλατφόρμα (ένα dashboard συνήθως, ταμπλό) για να γίνεται εύκολα και γρήγορα το σκανάρισμα και η απεικόνιση της πληροφορίας.

Advanced AI and Machine Learning: Μπορούν να εκπαιδεύονται και να μαθαίνουν από τα διάφορα γεγονότα και συμβάντα και να αναγνωρίζουν μοτίβα και συμπεριφορές που είναι πιθανώς κακόβουλα.

Remediate Vulnerabilities: Παρέχουν μια λίστα προτεραιοτήτων των ευπαθειών βάσει της σημαντικότητας τους.

Custom Reports Anytime, Anywhere: Παρέχουν αναφορές προγραμματισμένες και κατ' επιθυμία, οι οποίες είναι πάντα αναφορές που μπορούν να διατεθούν online και μπορούν να παραχθούν είτε σε PDF είτε σε CSV φορμά.

6. Περιπτώσεις Μελέτης

6.1 Μελέτη Περίπτωσης Χρήσης - Google Search Cluster Architecture

Αφορά την αρχιτεκτονική συστάδας της μηχανή αναζήτησης της Google στα μέσα της δεκαετίας του 2000. Πολλά έχουν αλλάξει απο τότε, ωστόσο αποτελεί σημαντική περίπτωση μελέτης ειδικά στο κομμάτι της κλιμάκωσης και πως αυτή επετεύχθη με τη χρήση απλών εμπορικών υπολογιστών με τη χρήση παραλληλίας [15].

Η μηχανή αναζήτησης της Google είναι η πλέον γνωστή: Δέχεται εκατοντάδες megabytes δεδομένων και χρησιμοποιεί δισεκατομμύρια κύκλων CPU. Το περιβάλλον χρειάζεται να μπορεί να διαχειριστεί χιλιάδες ερωτήματα (queries) απο τους χρήστες ανα δευτερόλεπτο. Θα πρέπει να είναι ανθεκτικό στα σφάλματα (fault-tolerant). Επίσης το κόστος πάντοτε θα πρέπει να λαμβάνεται υπόψιν, έχει σημασία η σχέση τιμής-απόδοσης. Εξ ού και θα πρέπει να είναι και αποδοτικό στην κατανάλωση ρεύματος. Κάτι επίσης σημαντικό είναι οτι το φορτίο εργασίας που θα δίνεται θα πρέπει να μπορεί να διαμεριστεί σε μικρά μέρη και να γίνεται παράλληλη επεξεργασία τους. Ωστόσο η απόδοση των CPU πάντα θα έχει μικρότερη προτεραιότητα απο αυτής του κόστους της σχέσης τιμής/απόδοσης.

Ας δούμε τις βασικές αρχές σχεδίασης της μηχανής.

Αξιοπιστία: Η αξιοπιστία βασίστηκε στο software και όχι στο hardware. Χρησιμοποιήθηκαν οικονομικά εμπορικά μοντέλα υπολογιστών για τη δημιουργία της συστάδας, ενώ έγινε μεγάλο replication των υπηρεσιών ανάμεσα στα μηχανήματα αυτά καθώς και αναγνώριση των σφαλμάτων τους.

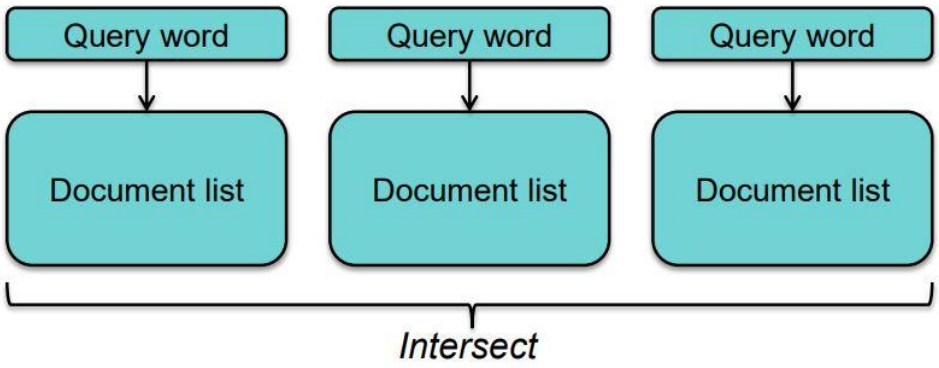
Σχεδίαση για μέγιστη ρυθμαπόδοση ανα πάσα στιγμή και όχι για απόδοση όταν οι απαιτήσεις είναι οι μέγιστες (peak server response time): Ο χρόνος ανταπόκρισης θα πρέπει να μπορεί να είναι διαχειρίσιμος μέσα από παραλληλισμό των αιτημάτων.

Η σχέση τιμής/απόδοσης: Πιο σημαντική απο την απόδοση του συστήματος στο μέγιστο των απαιτήσεων σε συγκεκριμένα χρονικά διαστήματα (peak performance)

Ας δούμε τα βήματα που ακολουθεί ένα αίτημα απο την αρχή μέχρι να εξυπηρετηθεί.

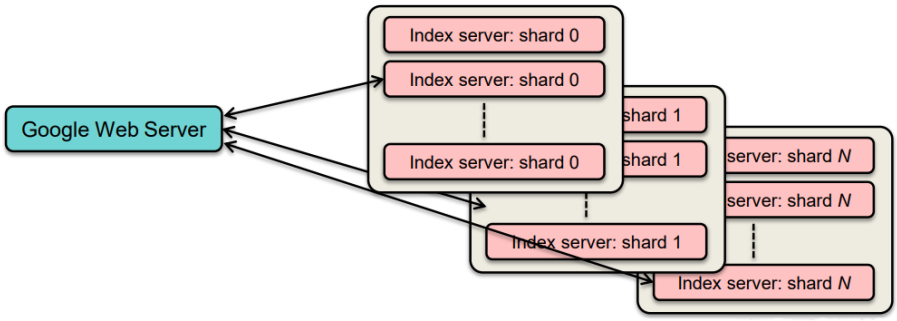
1. Ο χρήστης κάνει ένα αίτημα και ο browser του κάνει την αντιστοίχιση του domain με την IP της Google.
2. Στο domain google.com απαντούν πάρα πολλά clusters σε όλο τον κόσμο με χιλιάδες μηχανήματα το καθένα.
3. DNS Load Balancing - Η επιλογή του cluster γίνεται βάσει γεωγραφικής και δικτυακής εγγύτητας του χρήστη.
4. Το Load Balancing γίνεται σε επίπεδο cluster.
5. Η IP πηγαίνει σε έναν Load Balancer κάποιου cluster.

6. Ο Load Balancer παρακολουθεί τους servers (Google Web Servers) και μοιράζει (load balancing) τα αιτήματα στους servers βάσει του φορτίου και διαθεσιμότητας που έχει ο καθένας.
7. Ένας υπολογιστής του Google Web Servers τελικά θα πάρει το ερώτημα του χρήστη και θα συντονίσει την εκτέλεση του.
8. Κατά την εκτέλεση του θα πάει σε index servers, θα δημιουργήσει μια λίστα απο πιθανά έγγραφα απάντησης για κάθε keyword του ερωτήματος και απο τα σύνολα τους θα κρατήσει την τομή τους. Όλη αυτή η διεργασία γίνεται με μεγάλο παραλληλισμό καθώς τα ευρετήρια (indexes) είναι διασκορπισμένα και κάθε μέρος (shard) έχει ένα σετ απο έγγραφα.



Εικόνα 22 – Η τομή των λίστών με τα έγγραφα [11]

9. Το τελικό αποτέλεσμα θα είναι μια λίστα απο identifiers εγγράφων σε σειρά (docids, document ids).
10. Εκεί πάλι θα γίνει παράλληλη επεξεργασία και για κάθε docid το GWS θα φέρει τον τίτλο της σελίδας , το URL και τη σχετική μικρή περιγραφή που βλέπουμε να εμφανίζεται στα αποτελέσματα.



Εικόνα 23 - Index Sharding [11]

Όπως είναι φανερό πολύ μεγάλο μέρος της διαδικασίας είναι βασισμένο στην παράλληλη επεξεργασία. Δεν γίνεται, για παράδειγμα, αναζήτηση σε ένα πολύ μεγάλο ευρετήριο αλλά τα ευρετήρια χωρίζονται σε πολύ μικρότερης χωρητικότητας χώρους και γίνεται παράλληλη αναζήτηση. Στο τέλος γίνεται ένωση των αποτελεσμάτων η οποία έχει πολύ μικρότερο κόστος και είναι απλή σε διαδικασία.

6.2 Μελέτη Περίπτωσης – Επιθέσεις (Mirai botnet)

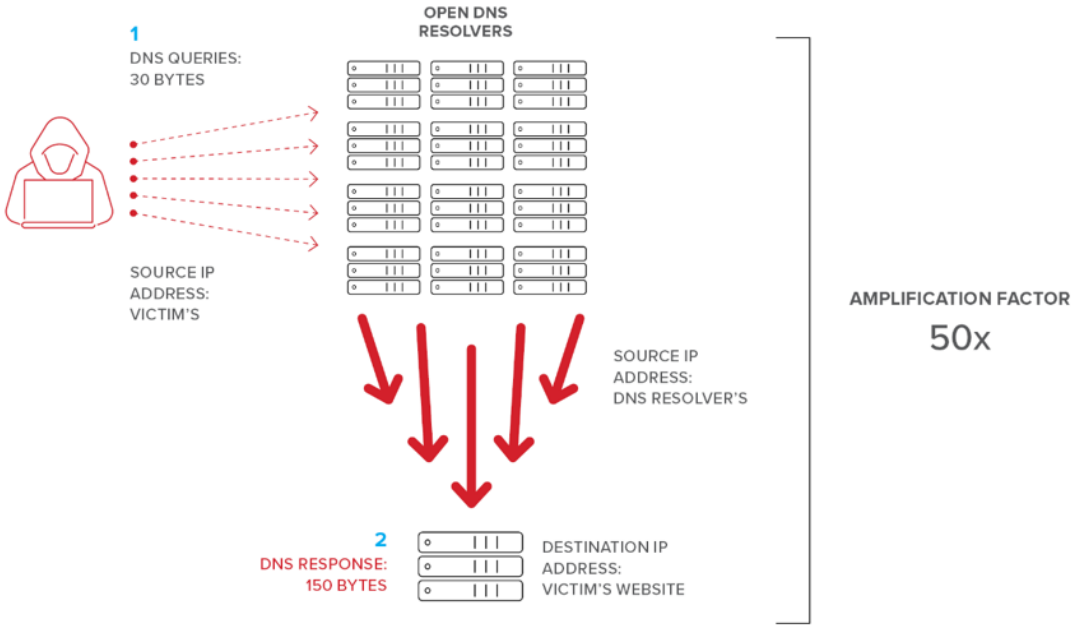
Στις 19/09/2016 ξεκίνησε η 1η επίθεση του Mirai botnet στο OVH, έναν απο τους μεγαλύτερους hosting providers στην Ευρώπη, η οποία διήρκεσε 7 ημέρες και ολοκληρώθηκε μετά την επίθεση στον Krebs [37]. Ο OVH παρέχει υπηρεσίες hosting σε περίπου 18 εκατομύρια εφαρμογές για πάνω απο 1 εκατομμύριο πελάτες. Η επίθεση έγινε σε έναν πελάτη, του οποίου το όνομα δεν έγινε ποτέ γνωστό απο περίπου 145.000 bot τα οποία δημιούργησαν φορτίο κίνησης της τάξεως των 1.1 Terabit ανα δευτερόλεπτο. Ωστόσο η επίθεση αυτή δεν ήταν η μοναδική.

Στις 20/09/2016 το διάσημο blog του Brian Krebs “KrebsOnSecurity.com”, ειδικού στην κυβερνοασφάλεια, έγινε στόχος μιας πολύ μεγάλης DDoS επίθεσης με σκοπό να ρίξει το website [38]. Η επίθεση δεν τα κατάφερε καθώς η Akamai , η εταιρεία που προστατεύει το συγκεκριμένο ιστότοπο, κατάφερε να αμυνθεί αποτελεσματικά. Ωστόσο το μέγεθος της επίθεσης ήταν σχεδόν το διπλάσιο απο την, έως τότε, μεγαλύτερη αντίστοιχη επίθεση που είχε ξανασυμβεί στα χρονικά του διαδικτύου [37].

Κατά τη διάρκεια της επίθεσης διοχετεύθηκε φορτίο κίνησης περίπου στα 665 Gigabits κίνησης ανα δευτερόλεπτο! Αυτό είναι πολλές τάξεις μεγέθους πάνω απο τα όρια των περισσοτέρων ιστότοπων.

Βάσει στοιχείων που παρέθεσε η Akamai, η προηγούμενη μεγαλύτερη επίθεση κατα τη διάρκεια της χρονιάς ήταν στα 363 Gbps. Με την εξής σημαντική διαφορά: Η επίθεση των 363 Gbps θεωρήθηκε οτι προήλθε απο ένα δίκτυο bots (botnet) απο συστήματα που ήταν ήδη εκτεθειμένα και με τη χρήση γνωστών τεχνικών μπόρεσαν να κατευθύνουν μια μεγαλύτερου μεγέθους επίθεση σε ένα μεγαλύτερο σύστημα.

Αντίθετα η επίθεση των 665 Gbps φαίνεται να έχει γίνει απο ένα πολύ μεγάλο δίκτυο συσκευών που έχουν χακαριστεί. Η επίθεση αυτή είναι αποτέλεσμα μιας tried-and-true τεχνικής, γνωστή και ως DNS Reflection επίθεση. Κατά τη διάρκεια μιας τέτοιας επίθεσης οι επιτιθέμενοι καταφέρνουν να χρησιμοποιοήσουν DNS εξυπηρετητές που υπάρχουν ελεύθεροι χωρίς καμία διαχείριση και να δημιουργήσουν έτσι μεγάλα traffic floods.



Εικόνα 24 - DNS Reflection Attack [35]

Ιδανικά οι DNS εξυπηρετητές παρέχουν μόνο υπηρεσίες σε trusted domain. Αλλά αυτό το είδος επίθεσης βασίζεται στα modem, router των καταναλωτών και εταιριών τα οποία έχουν ενσωματωμένους DNS servers και δεν έχουν κάποια σοβαρή παραμετροποίηση, οπότε δέχονται DNS ερωτήματα απο παντού. Έτσι οι επιτιθέμενοι στέλνουν spoof DNS ερωτήματα σε αυτούς τους DNS εξυπηρετητές και τροποποιώντας το ερώτημα ώστε να φαίνεται οτι προέρχεται απο το δίκτυο του στόχου, οι εξυπηρετητές στελνουν τις απαντήσεις στη διεύθυνση του στόχου.

Παράλληλα οι επιτιθέμενοι μπορούν να μεγεθύνουν με τέτοιο τρόπο την ανάκλαση και επίδραση της επίθεσης ώστε οι απαντήσεις που θα στέλνουν οι DNS εξυπηρετητές να είναι ακόμα μεγαλύτερες σε μέγεθος απο οτι συνήθως, εκμεταλλευόμενοι μια κατάληξη (extension) στο πρωτόκολλο του DNS που επιτρέπει μεγάλα μηνύματα. Με αυτό τον τρόπο ένα αίτημα μικρότερο των 100 byte μπορεί να φτάσει να έχει μια απάντηση 60-70 φορές μεγαλύτερη.

Όμως σύμφωνα πάντα με την Akamai καμία από αυτές τις τεχνικές επίθεσης που χρησιμοποιήθηκαν δεν βασίστηκε στη μεγέθυνση (amplification) ή στην ανάκλαση (reflection). Αντίθετα, κατά βάση προέρχονταν απο απόλυτα νόμιμες συνδέσεις με SYN, GET, POST flood αιτημάτων. Η ανάλυση της κίνησης όμως που ακολούθησε έδειξε και κάτι άλλο: Οτι το μεγαλύτερο τμήμα της επίθεσης προήλθε ως κίνηση σχεδιασμένη να δείχνει ως πακέτα GRE (Generic Routing Encapsulation), ένα πρωτόκολλο το οποίο χρησιμοποιείται για να φτιάξει σύνδεση ανάμεσα σε δύο κόμβους δικτύου. Το GRE

ωστόσο αφήνει 2 κόμβους να συνδεθούν αλλά μόνο αυτοί μεταξύ τους επιτρέπεται να ανταλλάσσουν δεδομένα και δεν επιτρέπεται να τα κάνουν κοινά στο διαδίκτυο.

Αυτό ήταν κάτι πρωτόγνωρο για την Akamai , και μπορούσε να σημαίνει μόνο ένα πράγμα: η επίθεση ενορχηστρώθηκε πίσω απο ένα ολόκληρο δίκτυο συσκευών που χακαρίστηκαν, πιθανότατα χιλιάδες συσκευές οι οποίες προέρχονταν από πολλά υποδίκτυα , απο πολλά Regions. Υπήρχαν σοβαρές ενδείξεις οτι το δίκτυο πιθανότατα αποτελείτο απο συσκευές IOT (internet of things): δρομολογητές, IP κάμερες, ψηφιακές κάμερες. Οτιδήποτε ήταν «έξυπνη συσκευή» και εκτεθειμένη στο διαδίκτυο, δυνητικά θα μπορούσε να ανήκει στο botnet.

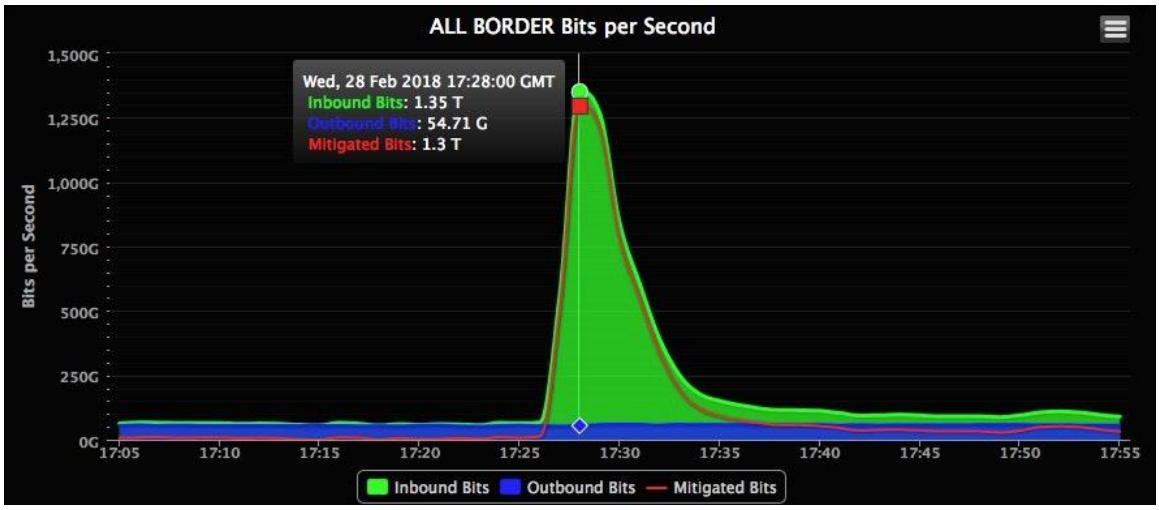
Σύμφωνα με μια αναφορά της περιόδου [36], η απειλή απο τέτοια IOT botnets μπορούσε να δημιουργηθεί από ένα κακόβουλο λογισμικό (malware) γνωστό ως: Lizkebab ή BASHLITE ή Tolus ή gafgyt. Σύμφωνα με την έρευνα ο πηγαίος κώδικας για το malware είχε διαρρεύσει στις αρχές του 2015 και είχαν δημιουργηθεί περισσότερες απο 12 παραλλαγές του. Κάθε bot διέσπειρε σε νέους κόμβους σκανάροντας για ευπαθείς συσκευές ώστε να εγκαταστήσει το malware. Το σκανάρισμα γινόταν με 2 τρόπους: Είτε σκάναρε τις πόρτες της υπηρεσίας του telnet και έκανε brute force για να μάθει τα credentials (username, password). Ή χρησιμοποιούσε εξωτερικά σκάνερ για να βρεί νέα bot.

Γιατί όμως ήταν τόσο σημαντικές αυτές οι επιθέσεις? Το Mirai botnet αποτέλεσε ένα σημαντικό βήμα εξέλιξης στο πόσο ισχυρές μπορούν να είναι πλέον οι επιθέσεις DDoS. Το μέγεθος και οι εξεζητημένες τεχνικές πίσω απο τις επιθέσεις αυτές ήταν πρωτόγνωρες για τα δεδομένα της εποχής, όπως χωρίς προηγούμενο ήταν και η κλίμακα της επίθεσης.

6.3 Μελέτη Περίπτωσης – Επιθέσεις (DDoS στο Github)

Στις 28/02/2018 το Github, δημοφιλής πλατφόρμα για προγραμματιστές, έγινε στόχος DDoS επίθεσης, μεγέθους 1.35 Terabit ανα δευτερόλεπτο και διάρκειας περίπου 20 λεπτών [33]. Σύμφωνα με το Github η κίνηση ιχνηλατήθηκε και οδήγησε σε ένα δίκτυο απο περισσότερα των 1000 αυτόνομων συστημάτων (ASN) σε πάνω απο 10.000 σημεία.

Στην εικόνα 25 είναι φανερή η διαφορά μεταξύ των επιπέδων της κίνησης υπο φυσιολογικές συνθήκες και της διαφοράς του επιπέδου κίνησης την ώρα της επίθεσης.



Εικόνα 25 – Γράφημα της DDoS επίθεσης στο Github to 2018 [37]

Παρόλο που το Github ήταν καλά προετοιμασμένο για DDoS επιθέσεις, κανείς δεν περίμενε τέτοιο μέγεθος. Κανείς δεν μπορούσε πραγματικά να προβλέψει μια επίθεση τέτοιου βεληνεκούς ώστε να είναι κατάλληλα προετοιμασμένος. Σύμφωνα με την αναφορά του Github, είχαν ήδη αυξήσει τα μέτρα ασφαλείας και είχαν διπλασιάσει τη δυνατότητα διαχείρισης του φορτίου κίνησης ώστε να προβλέψουν τέτοιες καταστάσεις. Ωστόσο μια τέτοια επίθεση δεν ήταν δυνατό να είναι διαχειρίσιμη χωρίς υποστήριξη.

Η επίθεση στο Github ήταν σημαντική λόγω της κλίμακας και της δυνατότητας εκμετάλλευσης μιας εντολής της Memcached (Βάση αποκλειστικά για Caching). Η τεχνική αυτή, που πλέον καλείται τεχνική επίθεσης Memcached DDoS, αποδείχθηκε πολύ αποτελεσματική καθώς έχει πολύ μεγάλο παράγοντα μεγέθυνσης επιτυγχάνοντας έως και 51.200 φορές παραπάνω αύξηση του μεγέθους του αρχικού αιτήματος.

6.4 Ανάλυση προκλήσεων ασφάλειας και λύσεων

Γιατί όμως οι επιθέσεις DDoS είναι οι σημαντικότερες σε ένα κατανεμημένο σύστημα? Ο λόγος είναι οτι υπάρχει πολύ μεγάλο αποτύπωμα (impact) όχι μόνο σε τεχνικό επίπεδο αλλά και ψυχολογικό.

Απο τεχνικής πλευράς είναι σύνηθες οι επιτιθέμενοι να είναι σχεδόν πάντα ένα βήμα μπροστά και δεν είναι πάντα δεδομένο οτι ο οργανισμός που δέχεται την επίθεση διαθέτει το προσωπικό με τις τεχνικές ικανότητες ή τους πόρους να αντιμετωπίσει μια τέτοια επίθεση , ειδικά απο ένα μέγεθος και πάνω. Επίσης δύσκολο είναι οτι θα πρέπει να αντιμετωπιστεί η επίθεση και παράλληλα να γίνεται η απαραίτητη προσαρμογή στους μηχανισμούς ασφαλείας του συστήματος ώστε να μειωθεί η πιθανότητα να υπάρχουν τέτοια φαινόμενα ξανά στο μέλλον.

Σημαντικά βήματα που πρέπει να υλοποιηθούν ώστε να αναβαθμιστεί η ασφάλεια του συστήματος είναι κάποια απο τα ακόλουθα:

Αλλαγή των σταθμών εργασίας: πιο σύγχρονοι με πιο αυστηρές πολιτικές ασφαλείας.

Επιλογή έμπιστου και ασφαλούς παρόχου για web hosting υπηρεσίες: Αποτελεί έναν απο τους ακρογωνιαίους λίθους στην κατανόηση και εξομάλυνση των επιθέσεων.

Ειδικοί Σύμβουλοι Ασφαλείας: συζητήσεις και αναφορές απο ειδικούς συμβούλους σε θέματα ασφαλείας για το πως μπορεί να μειωθεί ο κίνδυνος. Είτε ανήκουν στον πάροχο hosting είτε είναι εξωτερικοί συνεργάτες, είναι μείζονος σημασίας να υπάρχει τμήμα forensics το οποίο μπορεί μέσω ιχνηλάτησης και εξειδικευμένων τεχνικών να μας δώσει λεπτομερή αναφορά του τι συνέβη.

Προετοιμασία συστήματος: κάθε σύστημα πρέπει να είναι προετοιμασμένο , όσο το δυνατόν καλύτερα, για τέτοιου είδους επιθέσεις. Άρα η επιλογή παρόχου θα πρέπει να γίνεται και με βάση τα εργαλεία που διαθέτει για την πρόληψη και αντιμετώπιση τέτοιων επιθέσεων.

Αλλαγές σε συμπεριφορά και στον οργανισμό: πρέπει όλοι οι εργαζόμενοι-μέλη σε έναν οργανισμό να συνειδητοποιήσουν την σημαντικότητα κάποιων ενεργειών και να λάβουν εκπαίδευση σε βασικά θέματα συμπεριφοράς γύρω από την ασφάλεια.

Εσωτερικές διαδικασίες: Να υπάρχει ομάδα αντίδρασης (Incident Response Team) η οποία να έχει λάβει εκπαίδευση για τέτοιου είδους περιστατικά και το τι πρέπει να κάνει άμεσα ώστε να μη χαθεί χρόνος. Καταγραφή συλλογή και αξιολόγηση όλων των αρχείων καταγραφής ώστε να αξιολογηθεί κάθε σημείο της επίθεσης.

7. Ασφάλεια σε Ειδικές Εφαρμογές

7.1 Κατανεμημένα Συστήματα στο Cloud

Οι απαιτήσεις των οργανισμών απο τα cloud περιβάλλοντα είναι αυξημένες, ειδικά μετά την εποχή της πανδημίας του covid-19. Είναι πολύ σημαντικό το cloud περιβάλλον που θα επιλέξει κάποιος οργανισμός να είναι απόλυτα ασφαλές [38]. Η σημαντικότερη αρχιτεκτονική αρχή πάνω στην οποία είναι σχεδιασμένο ένα τέτοιο περιβάλλον είναι η **Zero Trust Architecture** [43]**.**

7.1.1 Προκλήσεις στο Cloud

Increased Attack Surface: Λόγω και της αυξημένης επιφάνειας επίθεσης, η κακή παραμετροποίηση και ασφάλεια του δικτύου προσφέρει πεδίο δράσης για τους επιτιθέμενους [40].

Lack of Visibility and Tracking: Οι χρήστες του cloud πολλές φορές δεν έχουν την επιθυμητή ορατότητα και έλεγχο σε ένα cloud περιβάλλον.

Ever-Changing Workloads: Καθώς το περιβάλλον και τα τμήματα (συστατικά, components) του cloud είναι διαρκώς μεταβαλλόμενα υπάρχει πολύ μεγάλη δυσκολία σε ένα τέτοιο περιβάλλον να εφαρμοστούν κανόνες ασφαλείας με τόσο μεγάλη προσαρμοστικότητα στις συχνές αλλαγές.

Devops, DevSecOps and Automation: Η κουλτούρα CI/CD (Continuous Integration – Continuous Delivery) πρέπει να είναι σχεδιασμένη πάνω στις αρχές ασφαλείας. Η ασφάλεια θα πρέπει να αποτελεί μέρος της διαδικασίας και όχι να εφαρμόζεται on-top εκ των υστέρων.

Granular Privilege and Key Management: Η παραμετροποίηση των ρόλων πρέπει να γίνεται με πολύ μεγάλη σοβαρότητα. Συχνά η κακή παραμετροποίηση ή το γεγονός οτι κάποιος χρήστης έχει ξεχαστεί με κάποιον ρόλο μεγαλύτερης εξουσιοδότησης απ’ οτι χρειάζεται οδηγεί σε κενό του συστήματος που μπορεί να καταλήξει σε μια επίθεση.

Complex Environments: Όλοι οι μέθοδοι που χρησιμοποιούνται και τα εργαλεία πρέπει να συνεργάζονται άψογα μεταξύ τους χωρίς να αφήνουν κενά που μπορεί κάποιος να εκμεταλλευθεί.

Cloud Compliance and Governance: Εναρμόνιση και συμμόρφωση με βασικές αρχές, κανονιστικά πλαίσια και directives όπως τα GDPR, PCI 3.2 κλπ.

7.1.2 Zero Trust Architecture (ZTA)

Το σημείο της εμπιστοσύνης (**Trust**) αποτέλεσε απο πολύ νωρίς σημαντικό ζήτημα στα κατανεμημένα συστήματα [45]. Με την εξέλιξη της τεχνολογίας τη συνεχή παρακολούθηση των προσβάσεων , τις αυστηρές πολιτικές, τους μηχανισμούς αυθεντικοποιήσης τελευταίας τεχνολογίας (MFA, BIOMETRICS, Passwordless Solutions) καθώς και με τις καταγραφές σε όλα τα events και τις ενέργειες των χρηστών και του συστήματος, εξασφαλίζεται πολύ υψηλό επίπεδο παροχής υπηρεσιών. Η τεχνολογία που εφαρμόζει την αρχή ZTA εξασφαλίζει προδιαγραφές εισόδου στο σύστημα και προαπαιτούμενα κοινά για όλους, θεωρώντας όλους τους χρήστες ισότιμα μέλη. Ελαχιστοποιεί την πιθανότητα εκμετάλευσης των δεδομένων και προσφέρει ένα μοντέλο ιδωτικότητας ανεξαρτήτως προσβάσεων. Σε κάθε πιθανότητα ή συσχέτιση με κάποιο μοτίβο επικίνδυνο, ανακαλούνται αμέσως όλα τα προνόμια του χρήστη καθώς και η δυνατότητα εισόδου του.

Η αρχή ZTA δεν μειώνει απλώς την επιφάνεια επιθέσεως αλλά αυξάνει παράλληλα τον έλεγχο και τις δυνατότητες της αυθεντικοποίησης ενώ παράλληλα βελτιστοποιεί την ορατότητα του χρήστη στο σύστημα και τις δυνατότητες περιήγησής του.

Αφορά 3 σημαντικά τμήματα:

- Χρήστες: Αυστηρή αυθεντικοποίηση και πολιτικές Least Access.
- Εφαρμογές: Αφαίρεση της εμπιστοσύνης ανάμεσα σε συστατικά (components) της εφαρμογής που επικοινωνούν μεταξύ τους χωρίς να είναι απαραίτητο και πλήρης και λεπτομερής καταγραφή κάθε ενέργειας και δυνατοτήτων.
- Υποδομή: Όλη η υποδομή θα πρέπει να διέπεται απο την αρχή ZTA, δίκτυα, δρομολογητές, μόντεμ, switches.

7.1.3 Προστασία στο Cloud

Granular, policy-based IAM and authentication controls across complex infrastructures: Πάντα είναι προτιμότερο να υπάρχουν γκρούπ και ρόλοι αντί να μπαίνει κάθε χρήστης σε ξεχωριστή πολιτική εξουσιοδότησης. Μπορούμε έτσι να διαχειριστούμε καλύτερα τις αλλαγές και τις προσβάσεις.

Zero-trust cloud network security controls across logically isolated networks and micro-segments: διαχωρισμός στα δίκτυα είτε αφορά τις εφαρμογές, είτε υπηρεσίες και πολιτικές ασφαλείας ώστε όταν χρειάζεται να επικοινωνούν μεταξύ τους.

Enforcement of virtual server protection policies and processes such as change management and software updates: Συνεχής επικαιροποίηση των πολιτικών ασφαλείας και επιβολή τους σε κάθε deployment.

Safeguarding all applications (and especially cloud-native distributed apps) with a next-generation web application firewall: Προστασία των εφαρμογών με τείχος προστασίας (WAF, Web Application Firewall) και συνεχής αναβάθμιση του.

Enhanced data protection: Κρυπτογράφηση σε όλα τα επίπεδα μεταφοράς, κανόνες ασφαλείας στην ανταλλαγή αρχείων και πάντα έλεγχος και σωστή παραμετροποίηση σε αποθηκευτικούς χώρους και buckets (AWS S3 Buckets) καθώς και συνεχής έλεγχος και διαχείριση για πόρους οι οποίοι δεν χρησιμοποιούνται ώστε να επιστρέφονται στο σύστημα.

7.2 Κατανεμημένα Συστήματα σε IoT (Internet of Things)

Όσο περισσότεροι τρόποι υπάρχουν για να συνδεθεί μια συσκευή με μια άλλη, τόσο περισσότερο μεγαλώνει η επιφάνεια επίθεσης και τόσο περισσότεροι κίνδυνοι υπάρχουν ώστε κάποιος να το εκμεταλλευτεί αυτό [42]. Να σημειωθεί οτι με τον όρο IoT δεν είναι απαραίτητα μόνο συσκευές που συνδέονται στο διαδίκτυο αλλά και συσκευές που απλώς συνδέονται με άλλες μέσω πρωτοκόλλων επικοινωνίας όπως το Bluetooth για παράδειγμα.

7.2.1 Προκλήσεις IoT

Κάποιες απο τις πιο σημαντικές προκλήσεις των συσκευών-συστημάτων IoT είναι οι ακόλουθες [43]:

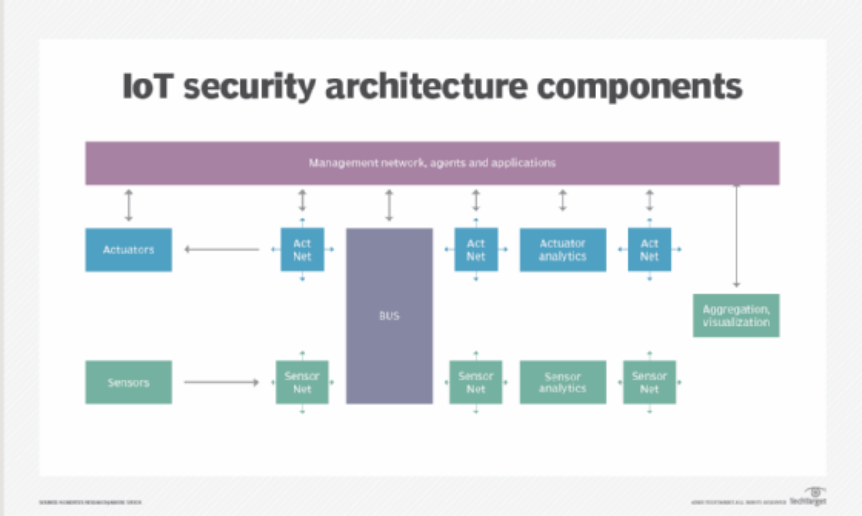
- Remote Exposure: Λόγω της μεγάλης έκθεσης (συνδεδεμένες μόνιμα στο δίκτυο χωρίς ιδιαίτερες πολιτικές ασφαλείας) που έχουν, διαθέτουν και αντίστοιχα μεγάλη επιφάνειας επίθεσης, οπότε καθίστανται ιδιαίτερα ευάλωτες. Είναι ο λόγος που οι επιθέσεις σε συσκεύες σαν και αυτές είναι συνήθως πολύ επιτυχημένες.
- Lack of Industry foresight: Δεν υπάρχει ένα συγκεκριμένο κανονιστικό πλαίσιο για αυτές τις συσκευές. Όλες έχουν πολλά κενά ασφαλείας και διαφορετικούς τρόπους κατασκευής η μια απο την άλλη με την βιομηχανία να εξαρτάται τελικά απο συσκευές που δεν είχε προβλέψει πόσο αδύναμες μπορεί να είναι σε θέματα ασφάλειας.
- Resource constraints: Οι συγκεκριμένοι πόροι ανα συσκευή μπορεί να δημιουργούν μεγάλα προβλήματα (πχ ελάχιστη μνήμη).
- Weak default passwords: Συχνά έρχονται με εργοστασιακά πολύ αδύναμα passwords.
- Multiple connected devices: Αν μια συσκευή έχει πολύ χαμηλό βαθμό ασφαλείας και παρακαμφθεί τότε είναι εύκολο και για τις υπόλοιπες που είναι συνδεδεμένες ακόμη και αν έχουν καλύτερα χαρακτηριστικά ασφαλείας.
- Lack of encryption: Συνήθως οι περισσότερες συσκευές στέλνουν δεδομένα χωρίς κρυπτογράφηση.

7.2.2 Προστασία IoT

Ασφάλεια απο το σχεδιασμό: Ο σχεδιασμός να εμπεριέχει την ασφάλεια και όχι η ασφάλεια να μπαίνει on-top στο τελικό προιόν [43], [44].

PKI και Ψηφιακά πιστοποιητικά: (PKI: Public Key Infrastructure) Χρήση ασύμμετρης κρυπτογράφησης με ιδωτικά και δημόσια κλειδιά ώστε να είναι κρυπτογραφημένη η ανταλλαγή μηνυμάτων μεταξύ 2 συσκευών.

Ασφάλεια Δικτύων: Προστασία στις θύρες , χρήση τειχών προστασίας (firewalls) , απενεργοποίση port-forwarding , IDS (Intrusion Detection Systems) και IPS (Intrusion Prevention Systems) συστήματα και γενικά προστασία και ισχυροποίηση των μηχανισμών ασφαλείας του δικτύου [49].



Εικόνα 26 - Ασφάλεια IoT [43]

Ασφάλεια API: Σχεδιασμός των API με βάση την ασφάλεια και αυστηρές υλοποιήσεις που δεν αφήνουν κενά ασφαλείας.

Security Gateways: Οι ενδιάμεσες πύλες (Gateways) μπορούν να διαθέτουν περισσότερους πόρους και ισχύ και να προστατεύσουν σε μια πιθανή επίθεση την απευθείας πρόσβαση στις συσκευές.

Security Updates: Αναβαθμίσεις συνεχώς στις συσκευές στο λογισμικό και σε οτι μπορεί να αναβαθμιστεί ώστε να μην χρησιμοποιεί λογισμικό που έχει κενά ασφαλείας γνωστά και εκμεταλλεύσιμα.

8. Συμπεράσματα και Προτάσεις

8.1 Συνοπτική Αξιολόγηση Ευρημάτων

Μέσα απο αυτή τη μελέτη είναι φανερό οτι τα κατανεμημένα συστήματα είναι πολύπλοκα συστήματα που όμως προσφέρουν μεγάλες δυνατότητες λύσεων με πολύ αποδοτικότερο τρόπο. Οι δυνατότητες που ένα κλασικό υπολογιστικό σύστημα δεν μπορεί να εγγυηθεί (Αποδοτικότητα, Διαθεσιμότητα, Κλιμάκωση, Fail-Safe Μηχανισμοί) μας δείχνουν ξεκάθαρα την τάση που θα ακολουθήσει ο χώρος των υποδομών και των συστημάτων της πληροφορικής τα επόμενα χρόνια [50].

Παρότι όμως η τάση για χρήση και εξέλιξη αυτών των συστημάτων αυξάνεται με μεγάλο ρυθμό, μαζί αυξάνονται και οι προκλήσεις. Σε ένα τεχνολογικό περιβάλλον που μεταλάσσεται καθημερινά δεν εξελίσσονται μόνο οι τεχνολογίες δημιουργίας, σχεδιασμού και υλοποίησης αλλά καθημερινά επίδοξοι κακόβουλοι χρήστες βλέπουν ευκαιρίες προς εκμετάλλευση σε διάφορα κενά του σχεδιασμού ή της υλοποίησης των συστημάτων αυτών. Βλέπουμε οτι στις επιθέσεις χρησμοποιούνται ολοένα και πιο εξεζητημένες τεχνικές, πρωτόγνωρες σε μέγεθος και έκταση. Γι’ αυτό και τα νέα συστήματα σχεδιάζονται με γνώμονα το οτι θα δεχθούν επιθέσεις μεγάλου βαθμού ώστε να μπορούν να ανταποκριθούν με τον καλύτερο τρόπο. Εκτός όμως απο τη δυνατότητα να αμυνθούν πρέπει να είναι και εξοπλισμένα με όλα εκείνα τα εφόδια που θα μπορέσουν να περιορίσουν την έκταση τη ζημιάς και παράλληλα να έχουν τη δυνατότητα να αναλύσουν όλη την πληροφορία ώστε να διδαχθούν απο τις επίθεσεις και να προετοιμαστούν καλύτερα για μελλοντικές παραλλαγές.

8.2 Μελλοντικές Προκλήσεις και Έρευνα

Η εξέλιξη της τεχνολογίας δεν είναι σύμμαχος μόνο των δημιουργών συστημάτων και υποδομών αλλά και των κακόβουλων χρηστών. Είδαμε οτι μεμονωμένα συστήματα μπορούν να συγκροτήσουν ένα μέρος κατανεμημένων συστημάτων το οποίο θα λειτουργήσει ως όπλο στα χέρια ομάδων ώστε να εξαπολύσουν πολύ ισχυρές επιθέσεις σε συστήματα και οργανισμούς. Υπάρχει πολύ μεγάλο πεδίο έρευνας στα επόμενα χρόνια με την εξέλιξη της τεχνητής νοημοσύνης και στο πως αυτή θα χρησιμοποιηθεί και απο τις 2 πλευρές. Και απο την πλευρά των συστημάτων ώστε να βελτιστοποιήσουν την παροχή υπηρεσιών τους και να τη χρησιμοποιήσουν ως αμυντικό μηχανισμό απο τον οποίο θα διδαχθούν και θα εξελιχθούν σαν συστήματα. Αλλά και απο την άλλη οι επιτιθέμενοι έχουν πολύ μεγάλο πεδίο έρευνας και δοκιμής μπροστά τους. Μην ξεχνάμε οτι σχεδόν πάντα οι επιτιθέμενοι βρίσκονται ένα βήμα μπροστά. Εκεί θα έχει πολύ μεγάλο ενδιαφέρον να εστιάσει κάποιος , στο πως θα μπορέσει κάποιος να χρησιμοποιήσει αυτό τον πληροφοριακό και τεχνογνωσιακό πλούτο ώστε να σχεδιάσει και να αναπτύξει συστήματα

με λύσεις που θα μπορέσουν να αμυνθούν αποτελεσματικά σε επιθέσεις πρωτόγνωρες για το κάθε σύστημα.

9. Βιβλιογραφία

[1] A. S. Tanenbaum και M. Van Steen, Distributed Systems - Principles and Paradigms, Amsterdam: Pearson, 2007.

[2] G. Coulouris, J. Dollimore και T. Kindberg, Distributed Systems - Concepts and Design, 4th ed., London, England: Addison - Wesley, 2005.

[3] M. Firdhous, «Implementation of Security in Distributed Systems - A Comparative Study,» *International Journal of Computer Information Systems,* τόμ. 2, αρ. 2, p. 6, 2011.

[4] R. Hasan, S. Myagmar, A. J. Lee και W. Yurcik, «Toward a threat model for storage systems,» σε *Proceedings of the 2005 ACM Workshop on Storage Security and Survivability (StorageSS '05)*, Fairfax, VA, 2005.

[5] S. T. Haji-Tajuddin, The role of 'perceptions of information value' in information security compliance behaviour: a study in Brunei Darussalam's public organisations, Loughborough: Loughborough University, 2016.

[6] V. Malamas, G. Palaiologos, P. Kotzanikolaou, M. Burmester και D. Glynos, «Janus: Hierarchical multi-blockchain-based access control (hmbac) for multi-authority and multi-domain environments,» *Applied Sciences,* τόμ. 13, αρ. 1, p. 566, 2023.

[7] D. Koutras, V. Malamas, P. Kotzanikolaou και T. Dasaklis, «A Risk Assessment Methodology for Supply Chain Tracking Services,» σε *International Conference On Cyber Management And Engineering (CyMaEn)*, Bangkok, Thailand, 2023.

[8] T. Dasaklis και Vangelis Malamas, «A review of the lightning network’s evolution: Unraveling its present state and the emergence of disruptive digital business models,» *ournal of Theoretical and Applied Electronic Commerce Research,* τόμ. 18, αρ. 3, pp. 1338--1364, 2023.

[9] M. Van Steen and A. S. Tanenbaum, Distributed Systems, 4th ed., Amsterdam: Maarten van Steen, 2023.

[10] S. Wilbur, Distributed Systems Security, London: Universisty College London, 2000.

[11] K. Nadiminti, M. Dias de Assunção και R. Buyya, «Distributed Systems and Recent Innovations: Challenges and Benefits,» *InfoNet Magazine,* τόμ. 16, αρ. 3, September 2006.

[12] M. A. Bauer, N. Coburn, D. L. Erickson, P. J. Finnigan, J. W. Hong, P. A. Larson, J. Pachl, J. Slonim, D. J. Taylor και T. J. Teorey, «A Distributed System Architecture for a Distributed Application Environment,» *IBM System Journals,* τόμ. 33, αρ. 3, 1994.

[13] O. Chukwuemeka, A. B. Ikharo και V. Oisamoje, «Security in Distributed System: A Review Perspective,» *International Journal of Latest Technology in Engineering, Management & Applied Science (IJLTEMAS),* τόμ. XI, αρ. X, p. 8, October 2022.

[14] V. Malamas, T. Dasaklis, V. Arakelian και G. Chondrokoukis, «A blockchain framework for digitizing securities issuance: the case of green bonds,» *Journal of Sustainable Finance & Investment,* pp. 1--27, 2023.

[15] P. Krzyzanowski, Distributed Systems, New Jersey: Rutgers University, 2023.

[16] A. S. Tanenbaum και D. J. Wetherall, Computer Networks (Firth Edition), Boston: Prentice Hall, 2011.

[17] J. Saltzer και M. Kaashoek, Principles of Computer System Design, An Introduction, San Mateo, CA: Morgan Kaufman, 2009.

[18] Infosec Institute, «Secure system design principles and the CISSP,» 13 May 2017. [Ηλεκτρονικό]. Available: https://resources.infosecinstitute.com/certifications/cissp/secure-system-design-principles/.

[19] R. J. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, Second Edition, Crosspoint Boulevard Indianapolis: Wiley Publishing, 2008.

[20] D. Harinath, P. Satyanarayana και M. V. R. Murthy, «A Review on Security Issues and Attacks in Distributed Systems,» *Journal of Advances in Information Technology Vol. 8, No. 1,* 2017.

[21] S. Neeraj, Distributed Systems Security Knowledge Area Issue 1.0, Lancaster: The Cyber Security Body Of Knowledge, 2019.

[22] J. Mirkovic και P. Reiher, «A Taxonomy of DDoS Attack,» *ACM SIGCOMM Computer Communications Review,* τόμ. 34, αρ. 2, p. 16, April 2004.

[23] G. Kumar, «Denial of service attacks – an updated perspective,» *Systems Science & Control Engineering - An Open Access Journal,* p. 11, 2016.

[24] C. Douligeris και A. Mitrokotsa, «DDoS attacks and defense mechanisms: Classification and state-of-the-art,» *Computer Networks,* 2004.

[25] cloudflare.com, «What is a denial-of-service (DoS) attack?,» [Ηλεκτρονικό]. Available: https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/.

[26] cloudflare.com, «What is a DDoS attack?,» [Ηλεκτρονικό]. Available: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/.

[27] hacken.io, «51% Attack: The Concept, Risks & Prevention,» [Ηλεκτρονικό]. Available: https://hacken.io/discover/51-percent-attack/.

[28] bitpanda.com, «What is a 51% attack and how is it prevented?,» [Ηλεκτρονικό]. Available: https://www.bitpanda.com/academy/en/lessons/what-is-a-51-attack-and-how-is-it-prevented/.

[29] L. Deng, Y. Gao, Y. Chen και A. Kuzmanovic, «Pollution attacks and defenses for Internet caching systems,» *Computer Networks,* 2008.

[30] B. Nour, S. Mastorakis, R. Ullah και N. Stergiou, «Information-Centric Networking in Wireless Environments: Security Risks and Challenges,» IEEE Wireless Communications, 2021.

[31] Z. Wang, X. Xiong και W. J. Knottenbelt, «Blockchain Transaction Censorship: (In)secure and (In)efficient?,» σε *4th International Conference MARBLE 2023*, London, United Kingdom, 2023.

[32] T. Bhatia και A. Verma, «Security Issues in Manet: A Survey on Attacks and Defense,» *International Journal of Advanced Research in Computer Science and Software Engineering,* τόμ. 3, αρ. 6, p. 14, June 2013.

[33] E. Heilman, A. Kendler και A. Zohar, «Eclipse Attacks on Bitcoin’s Peer-to-Peer Network,» σε *24th USENIX Security Symposium*, Washington, D.C., 2015.

[34] coincarp.com, «What Is A Sybil Attack in Blockchain?,» 20 July 2022. [Ηλεκτρονικό]. Available: https://www.coincarp.com/de/learn/what-is-a-sybil-attack-in-blockchain/.

[35] K. Raj, «What can Blockchain developers learn from Eclipse Attacks in a Bitcoin network,» 11 April 2019. [Ηλεκτρονικό]. Available: https://hub.packtpub.com/what-can-blockchain-developers-learn-from-eclipse-attacks-in-a-bitcoin-network-koshik-raj/.

[36] M. Benattou, L. Cacciari, R. Pasini και O. Rafiq, «Principles and tools for testing open Distributed Systems,» σε *Testing of Communicating Systems: Methods and Applications*, Budapest, Hungary, September 1999.

[37] P. Nicholson, «Five Most Famous DDoS Attacks and Then Some,» 21 January 2022. [Ηλεκτρονικό]. Available: https://www.a10networks.com/blog/5-most-famous-ddos-attacks/.

[38] B. Krebs, «KrebsOnSecurity Hit With Record DDoS,» 21 September 2016. [Ηλεκτρονικό]. Available: https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/.

[39] cyberhoot.com, «DNS Reflection and Amplification Attacks,» [Ηλεκτρονικό]. Available: https://cyberhoot.com/cybrary/dns-reflection-attack/.

[40] Flashpoint Intel Team και Level 3 Threat Research Labs , «Attack of Things,» 17 September 2016. [Ηλεκτρονικό]. Available: https://flashpoint.io/blog/attack-of-things/.

[41] wired.com, «GitHub Survived the Biggest DDoS Attack Ever Recorded,» 2018. [Ηλεκτρονικό]. Available: https://www.wired.com/story/github-ddos-memcached/.

[42] A. Singh και K. Chatterjee, «Cloud security issues and challenges: A survey,» *Journal of Network and Computer Applications,* τόμ. 79, αρ. 1, 2017.

[43] Y. He, D. Huang και L. Chen, «A Survey on Zero Trust Architecture: Challenges and Future Trends,» *Wireless Communications and Mobile Computing,* p. 13, 2022.

[44] M. A. Himmel και F. Grossman, «Security on distributed systems: Cloud security versus traditional IT,» *IBM Journal of Research and Development ,* τόμ. 58, αρ. 1, 2014.

[45] M. Blaze, J. Feigenbaum, J. loannidis και A. D. Keromytis, «The Role of Trust Management in Distributed Systems Security - Lecture Notes in Computer Science,» 1999.

[46] fortinet.com, «What Is IoT Security? Challenges and Requirements,» [Ηλεκτρονικό]. Available: https://www.fortinet.com/resources/cyberglossary/iot-security.

[47] K. Yasar, S. Shea και I. Wigmore, «IoT security (internet of things security),» 2023. [Ηλεκτρονικό]. Available: https://www.techtarget.com/iotagenda/definition/IoT-security-Internet-of-Things-security.

[48] kaspersky.com, «Internet of Things security challenges and best practices,» [Ηλεκτρονικό]. Available: https://www.kaspersky.com/resource-center/preemptive-safety/best-practices-for-iot-security.

[49] checkpoint.com, «Biggest IoT Security Challenges,» [Ηλεκτρονικό]. Available: https://www.checkpoint.com/es/cyber-hub/network-security/what-is-iot-security/biggest-iot-security-challenges/.

[50] T. Salah, M. Zemerly, c. Yeun, M. Al-Qutayri και Y. Al-Hammadi, «The evolution of distributed systems towards microservices architecture,» *International Conference for Internet Technology and Secured Transactions,* December 2016.

[51] confluent.io, «What is a Distributed System?,» [Ηλεκτρονικό]. Available: https://www.confluent.io/learn/distributed-systems/.

[52] P. Krzyzanowski, A taxonomy of distributed systems, New Jersey: Rutgers University, 2000-2003.

[53] V. Malamas, T. Dasaklis, T. Voutsinas και P. Kotzanikolaou, «Blockchain Service Layer for ERP data interoperability among multiple supply chain stakeholders,» Rome, Italy, 2023.