ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Παρακολούθηση διαθεσιμότητας και επιδόσεων υποδομής κέντρου δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Αλέξανδρου Γεωργόπουλου

Επιβλέπων: Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π

Αθήνα, Οκτώβριος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Παρακολούθηση διαθεσιμότητας και επιδόσεων υποδομής κέντρου δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Αλέξανδρου Γεωργόπουλου

Επιβλέπων: Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 8η Νοεμβρίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	….....
Ευστάθιος Συκάς Καθηγητής Ε.Μ.Π	Νικόλαος Μήτρου Καθηγητής Ε.Μ.Π	Ιωάννα Ρουσσάκη Αναπλ. Καθηγήτρια

Αθήνα, Οκτώβριος 2022

(Υπογραφή)

.....
Αλέξανδρος Γεωργόπουλος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

Περίληψη

Αδιαμφισβήτητα, αποτελεί ισχυρό εργαλείο για την βέλτιστη αξιοποίηση των δυνατοτήτων ενός συστήματος, η επίγνωση των αδυναμιών του, καθώς και των πιθανών δυσλειτουργιών που δύναται να λάβουν χώρα. Ωστόσο, η επίγνωση αυτή είναι άρρηκτα συνδεδεμένη με την διαδικασία της παρακολούθησης (monitoring) της ίδιας της υποδομής.
Με τον όρο παρακολούθηση, υποδηλώνεται η διαδικασία επίβλεψης στοιχείων και μονάδων ενός συστήματος, δια μέσου της συλλογής δεδομένων και ύστερα της ανάλυσης αυτών με απεικονιστικές μεθόδους, με απώτερο σκοπό την βέλτιστη εκμετάλλευση των πόρων της υποδομής.

Στην παραπάνω προσέγγιση και συγκεκριμένα στην επίβλεψη και πρόληψη πιθανών κινδύνων, συνεισφέρει ο μηχανισμός της αποστολής ειδοποιητικών συναγερμών (Alerting), κατά τον οποίο αποστέλλονται ειδοποιήσεις, για την ταχεία ενημέρωση των διαχειριστών του συστήματος σε περίπτωση πιθανής διακοπής της λειτουργίας των υπηρεσιών.

Στο πρώτο μέρος της παρούσας διπλωματικής εργασίας, πραγματοποιείται μια παρουσίαση του σχεδιασμού του συστήματος παρακολούθησης, καθώς και των στοιχείων και λογισμικών που αυτό περιλαμβάνει.

Στο δεύτερο μέρος της εργασίας, περιγράφεται ο αναλυτικός τρόπος εγκατάστασης και παραμετροποίησης των λογισμικών και υπηρεσιών που χρησιμοποιούνται, ώστε να λειτουργούν ομαδικά με παραγωγικό και αξιόπιστο τρόπο.

Λέξεις κλειδιά

Monitoring, metrics, Prometheus, Prometheus exporters, Node exporter, Blackbox exporter, VMware exporter, BigBlueButton exporter, Gitlab exporter, SNMP exporter, Alerting, Prometheus Alertmanager, Alerta, log files, Grafana, Elasticsearch, Kibana, Logstash, Filebeat.

Abstract

Undoubtedly, being aware of the weaknesses of a system, as well as the possible malfunctions that can occur, is a powerful tool for making the most of a system's potential. However, this awareness is linked to the process of monitoring an infrastructure.
The term monitoring refers to the process of supervising elements and units of a system, through the collection of data. Thus, the next step to that methodology lays emphasis on the analysis of these visualization methods, with the ambition being an optimal exploitation of infrastructure resources.

The "Alerting" mechanism contributes to the above approach, and specifically to the supervision and prevention of potential risks, during which notifications are sent to quickly inform the system administrators in the event of a possible service interruption.

The first part of the thesis outlines the design of the monitoring system, as well as the elements and software it includes.

The second part of the thesis describes the detailed way of installing and configuring the software and services that are used, in order to operate efficiently and reliably with each other.

Keywords

Monitoring, metrics, Prometheus, Prometheus exporters, Node exporter, Blackbox exporter, VMware exporter, BigBlueButton exporter, Gitlab exporter, SNMP exporter, Alerting, Prometheus Alertmanager, Alerta, log files, Grafana, Elasticsearch, Kibana, Logstash, Filebeat.

Ευχαριστίες

Με την περάτωση της διπλωματικής μου εργασίας, θα ήθελα να ευχαριστήσω ιδιαιτέρως τον καθηγητή και επιβλέποντα μου κύριο Ευστάθιο Συκά, για την ευκαιρία που μου έδωσε να λάβω την εκπόνηση και μελέτη ενός τόσο ενδιαφέροντος θέματος.

Ένα μεγάλο ευχαριστώ οφείλω επίσης στον κύριο Πάρι Χαραλάμπου, για την άριστη συνεργασία που είχαμε κατά τη διάρκεια της διπλωματικής εργασίας, καθώς και τις πολύτιμες καθοδηγήσεις που μου παρείχε.

Δε θα μπορούσα να λησμονήσω τους καθηγητές που είχα κατά την διάρκεια των σπουδών μου, οι οποίοι μου μετέδωσαν πληθώρα πολύτιμων γνώσεων.

Επιπλέον, θα ήθελα να ευχαριστήσω την οικογένειά μου και όλους τους φίλους και φίλες μου για την υποστήριξη και το ενδιαφέρον τους καθόλη την διάρκεια των σπουδών μου.

Ιδιαιτέρως, θα ήθελα να ευχαριστήσω τον αδελφό μου Θεόδωρο που στάθηκε δίπλα μου σαν φίλος και αδερφός σε όλες τις δυσκολίες των τελευταίων χρόνων.

Τέλος, ένα μεγάλο ευχαριστώ ανήκει και στον καλό μου φίλο και συνάδελφο Μάνο Ρωμανό, με τον οποίο συνεργάστηκα άψογα τα τελευταία χρόνια και ο τρόπος σκέψης του οποίου με κινητροδότησε στο να πιστεύω στις δυνατότητες και στα όνειρά μου.

Περιεχόμενα

Κατάλογος Στιγμιοτύπων

Μέρος 1ο

Κεφάλαιο 1: Εισαγωγή

Αντικείμενο διπλωματικής εργασίας

Η παρούσα διπλωματική εργασία έχει ως αντικείμενο τον σχεδιασμό και την υλοποίηση μιας αρχιτεκτονικής για την παρακολούθηση της διαθεσιμότητας και του ποσοστού χρησιμοποίησης των υπηρεσιών του εργαστηρίου Δικτύων Υπολογιστών. Η υποδομή αυτή, περιλαμβάνει μια Time Series βάση δεδομένων, όπως είναι το Prometheus και ένα λογισμικό για την οπτικοποίηση (visualization), όπως είναι το Grafana. Επιπρόσθετα, γίνεται χρήση του λογισμικού συλλογής, επεξεργασίας ανάλυσης και οπτικοποίησης αρχείων καταγραφής (log files) της σουίτας του Elasticsearch (Elastic, Kibana, Logstash, Filebeat). Στην υλοποίηση αυτή, συνεισφέρουν δικτυακές συσκευές όπως εξυπηρετητές, δρομολογητές και εικονικά μηχανήματα (VMs).

Σκοπός διπλωματικής εργασίας

Κύριος στόχος της διπλωματικής εργασίας είναι η βέλτιστη αξιοποίηση των φυσικών πόρων του εργαστηρίου Δικτύων Υπολογιστών για την επίτευξη των στόχων (α) υψηλής διαθεσιμότητας των υπηρεσιών/υποδομών, αλλά και (β) την ελαχιστοποίηση της καταναλισκόμενης ηλεκτρικής ενέργειας. Για να επιτευχθεί αυτό, θα πρέπει να υλοποιηθεί ένα σύστημα το οποίο θα παρακολουθεί αδιάκοπα τις προς επίβλεψη δικτυακές συσκευές. Με αυτό το τρόπο, είναι δυνατόν να αποφευχθεί οποιαδήποτε δυσλειτουργία που μπορεί να οφείλεται σε αδυναμία ή ανωμαλία του συστήματος, την κατάχρηση πόρων, την υπερβολική δικτυακή κίνηση, καθώς και σε πολλούς άλλους πιθανούς παράγοντες. Γι’ αυτό το σκοπό, η προτεινόμενη λύση λαμβάνει μετρήσεις από το σύνολο των υποδομών του εργαστηρίου που αποτελείται από δικτυακές συσκευές, φυσικούς εξυπηρετητές (Dell), αλλά και υπηρεσίες που εκτελούνται σε εικονικές μηχανές (VM).
Ύστερα, οι μετρήσεις αυτές αποθηκεύονται σε μία Times Series βάση δεδομένων και στην συνέχεια παρουσιάζονται γραφικά με τη χρήση του λογισμικού Grafana. Για την καλύτερη παρακολούθηση και ταχύτερη ενημέρωση των διαχειριστών για πιθανές αστοχίες, αποστέλλονται μέσω ειδοποιητικών συναγερμών, συναγερμοί για πιθανές διακοπές στη λειτουργία των υπηρεσιών.
Τέλος, αναπτύσσονται μηχανισμοί συλλογής, αξιοποίησης και ανάλυσης αρχείων καταγραφής (log αρχεία) των εξυπηρετητών του εργαστηρίου, με χρήση της σουίτας του λογισμικού Elasticsearch (Kibana, Logstash, Filebeat). Η εγκατάσταση και η ανάπτυξη της εργασίας πραγματοποιείται σε υποδομές του εργαστηρίου Δικτύων Υπολογιστών.

Παρακολούθηση (Monitoring)

Είναι γεγονός, πως στις μέρες μας ένα σύνθετο δίκτυο υπολογιστών, όπως είναι εκείνο του εργαστηρίου Δικτύων Υπολογιστών του Ε.Μ.Π, περιλαμβάνει πληθώρα δικτυακών συσκευών και εξυπηρετητών. Πρόκειται για μια διασύνδεση τόσο φυσικών εξοπλισμών (hardware), όσο και λογισμικού (software). Η διασύνδεση αυτή, συνοδεύεται από τη μία πλευρά, από ένα μεγάλο πλήθος διεργασιών που τελούνται καθ' όλη τη διάρκεια της ημέρας σε εξυπηρετητές, αλλά και από την άλλη πλευρά από ανταλλαγή πολλών δεδομένων. Στα παραπάνω στοιχεία είναι δυνατόν σε κάποια φάση της λειτουργίας τους να εμφανιστούν εμπόδια στην ομαλή λειτουργία του συστήματος, όπως (α) η υπερφόρτωση (overloading), (β) η μη ανταπόκριση συστήματος ή ενός εξυπηρετητή, (γ) η υπερθέρμανση κάποιας συσκευής, είτε (δ) γενικότερα κάποια δυσλειτουργία. Γι' αυτό το λόγο, καθίσταται πολλές φορές απαραίτητη η σχεδίαση μιας αρχιτεκτονικής, η οποία θα βελτιστοποιεί την αξιοποίηση των πόρων του δικτύου, θα συγκεντρώνει κεντρικά την πληροφορία που ο διαχειριστής δικτύου επιθυμεί να διαχειριστεί, αλλά και θα επιβλέπει την απόδοσή του.

Καθίσταται συνεπώς πολύτιμη η χρήση μιας αρχιτεκτονικής, η οποία όχι μόνο θα προλαμβάνει παρόμοιες καταστάσεις, αλλά θα προειδοποιεί και για τυχόν επιπλοκές που μπορεί να προκληθούν. Για την καλύτερη προσέγγιση του προβλήματος αυτού θα πρέπει, να αναλυθεί η αρχιτεκτονική σε μικρότερα τμήματα και υποσυστήματα, δίνοντας έτσι την επιλογή της καλύτερης τεχνικής λύσης για κάθε υπο-πρόβλημα (best of breeds). Πιο συγκεκριμένα, θα πρέπει το πρώτο βήμα για την δημιουργία της αρχιτεκτονικής, να περιλαμβάνει μια μέθοδο συλλογής δεδομένων. Στην παρούσα διπλωματική εργασία, τα δεδομένα που είναι προς μελέτη είναι κυρίως μετρικές (metrics) και δεδομένα διαχείρισης, όπως είναι τα αρχεία καταγραφής (log files). Ο λόγος για τη μελέτη αυτή, θα αποσαφηνιστεί παρακάτω.

Οι μετρικές αποτελούν αριθμητικές μετρήσεις, οι οποίες λαμβάνονται δειγματοληπτικά ανά τακτά χρονικά διαστήματα [1]. Το χρονικό διάστημα που μεσολαβεί μεταξύ των μετρήσεων εξαρτάται από το είδος του εξοπλισμού και στα σύγχρονα συστήματα είναι μερικά δευτερόλεπτα. Οι μετρήσεις αυτές, είναι δυνατό να αφορούν σε διάφορα είδη πληροφορίας, αναλόγως την εφαρμογή και την πληροφορία που αυτή φέρει [2], [3], [4]. Στα πλαίσια τηλεκπαίδευσης ακαδημαϊκών ιδρυμάτων, συχνή μελέτη αποτελούν εξυπηρετητές ιστού (web servers), οι οποίοι φιλοξενούν πλατφόρμες τηλεκπαίδευσης, όπως είναι το Moodle ή το BigBlueButton. Σε αυτή τη περίπτωση, είναι πιθανό μια μετρική να είναι το πλήθος των ενεργών συνδέσεων σε μια τηλεδιάσκεψη, το μέγιστο πλήθος συμμετεχόντων που επιτεύχθηκε σε ένα συγκεκριμένο χρονικό διάστημα κ.ο.κ.

Πέραν όμως τον εφαρμογών, οι μετρικές μπορεί να είναι πολύ χρήσιμες και για την παρακολούθηση δικτυακών συσκευών, όπως είναι μεταγωγείς ή και φυσικών ή εικονικών εξυπηρετητών. Ειδικότερα, σε έναν εξυπηρετητή, οι μετρήσεις που ενδιαφέρουν, μπορεί να είναι η χρησιμοποίηση των επεξεργαστών (CPU Utilization), η χρησιμοποίηση του αποθηκευτικού χώρου (Disk Space Usage) ή της μνήμης RAM (RAM Usage), η δικτυακή

κίνηση (Network Traffic) [5], το φορτίο του συστήματος (System Load), η χρησιμοποίηση εισόδου/εξόδου (I/O Utilization), αλλά και πολλά ακόμη που θα παρουσιαστούν αργότερα αναλυτικότερα. Γι' αυτό το λόγο, για να αξιοποιηθούν τέτοιου είδους πληροφορίες, είναι απαραίτητες οι μετρικές ως είδος δεδομένου.

Στη παρούσα διπλωματική εργασία παρακολουθούνται τα παρακάτω είδη συστημάτων:

- Φυσικοί εξυπηρετητές και λογισμικά συστήματα
- Εξυπηρετητές BigBlueButton
- Εξυπηρετητές ιστού
- SNMP συσκευές και μεταγωγείς
- Εξυπηρετητές τύπου ESXi της VMware
- Εξυπηρετητές Gitlab

Για την παρακολούθηση των παραπάνω στοιχείων, απαιτούνται οι κατάλληλοι Prometheus exporters, οι οποίοι εξάγουν τις απαραίτητες μετρικές και είναι αντίστοιχα οι:

- Node exporter
- BigBlueButton exporter
- Blackbox exporter
- SNMP exporter
- VMware exporter
- Gitlab exporter

Περισσότερες πληροφορίες για τον κάθε exporter παρουσιάζονται αναλυτικά σε επόμενα κεφάλαια.
Όσον αφορά τα αρχεία καταγραφής, πρόκειται για αρχεία δεδομένων κειμένου, τα οποία αποθηκεύουν συμβάντα (events), διεργασίες (processes), μηνύματα (messages) και άλλα είδη δεδομένων από εφαρμογές (applications), λειτουργικά συστήματα (Operational Systems) ή συσκευές.

Για την συλλογή και ανάκτηση των μετρικών στην υλοποίηση της παρούσας διπλωματικής εργασίας, χρησιμοποιείται το λογισμικό Prometheus, ως βάση δεδομένων [6], ενώ για τα αρχεία καταγραφής το λογισμικό Elasticsearch [7].

Κεφάλαιο 2: Prometheus

Για τη αποθήκευση και την προσπέλαση των συλλεγμένων μετρικών, καθώς και την παρακολούθηση και διαχείριση ειδοποιητικών συναγερμών (Alerts), γίνεται χρήση του λογισμικού ανοιχτού κώδικα (open-source) Prometheus.

Στιγμιότυπο 2: Λογότυπο Prometheus

Είναι γεγονός, ότι έχει ενεργή κοινότητα, χαρακτηριστικό που συμβάλλει στη χρήση του από πολλούς οργανισμούς και μεγάλες εταιρείες. Πιο συγκεκριμένα, πρόκειται για μία Time Series βάση δεδομένων, η οποία αποθηκεύει δεδομένα δειγματοληπτικά με τη χρήση δηλαδή χρονοσφραγίδων (timestamps) [8].
Όπως είναι φυσικό, υπάρχουν και άλλες εναλλακτικές επιλογές για τη χρήση Time Series Database, όπως είναι η Graphite, η InfluxDB ή η OpenTSDB [9], [10].

Σε επόμενο βήμα, για την καλύτερη κατανόηση της αρχιτεκτονικής που χρησιμοποιείται για την διαχείριση των μετρικών, θα πρέπει να παρουσιαστεί ο τρόπος με τον οποίο το Prometheus χρησιμοποιεί και επεξεργάζεται τις μετρικές.

Το Prometheus, ως μια Time Series βάση δεδομένων, αποθηκεύει και σημειώνει χρονικά (timestamp) τα δεδομένα και τις ιδιότητές τους, με βάση το χρόνο. Πιο συγκεκριμένα, οι πληροφορίες οργανώνονται και αποθηκεύονται στην βάση με κριτήριο την χρονική τους εμφάνιση. Ένας από τους σημαντικότερους λόγους για τους οποίους χρησιμοποιείται μια Time Series βάση δεδομένων στα πλαίσια της συγκεκριμένης διπλωματικής, αποτελεί το χαρακτηριστικο αυτό.
Ειδικότερα, εφόσον επιθυμείται η παρακολούθηση της δραστηριότητας διαφόρων εφαρμογών, κρίνεται απαραίτητη η γνώση της μεταβολής των δεδομένων με βάση την χρονική εξέλιξη.
Ένα παράδειγμα, αποτελούν τα αιτήματα επισκέψεων μιας ιστοσελίδας ανα λεπτό. Σε περίπτωση που παρατηρηθεί κάποια ακραία σε σχέση με τον μέσο όρο, μεταβολή, τότε δίνεται η ευκαιρία στους διαχειριστές να αναζητήσουν την αιτία πίσω από το φαινόμενο αυτό.

2.1: Μετρικές στο Prometheus

Ο τρόπος με τον οποίο τα δεδομένα ανακτώνται από το Prometheus είναι η μέθοδος Pull over HTTP [6]. Ωστόσο, για να πραγματοποιηθεί το "pulling" των μετρικών, θα πρέπει αυτά να εκτεθούν (expose) σε συγκεκριμένη μορφή συμβατή με την βάση δεδομένων. Αναλυτικότερα, θα πρέπει να τηρηθεί το "Text-based format" του Prometheus [11].

Σε αυτό το σημείο, θα πρέπει να τονιστεί το γεγονός, ότι για να δημιουργηθεί η συμβατή αυτή μορφή θα πρέπει να δημιουργηθεί από τα client libraries, ώστε να γίνουν διαθέσιμες οι κατάλληλες μετρικές που είναι συμβατές με το Prometheus. Ορισμένα client libraries που υποστηρίζουν αυτή τη δυνατότητα παρουσιάζονται στην ιστοσελίδα του Prometheus [12]. Σε περίπτωση που κανείς χρησιμοποιεί κάποια γλώσσα προγραμματισμού που δεν έχει διαθέσιμα ένα από αυτά τα client libraries, θα πρέπει αυτό να υλοποιηθεί ξεχωριστά τηρώντας τους κανόνες που αναγράφονται στην ιστοσελίδα του Prometheus [11].

Θα πρέπει να σημειωθεί ότι υπάρχουν τέσσερις κύριοι **τύποι** των **μετρικών** που υποστηρίζουν τα client libraries του Prometheus [13].

1) "**counter**", ο οποίος αποτελεί έναν μονοτονικά αυξανόμενο μετρητή, ο οποίος είτε αυξάνεται, είτε μπορεί να τεθεί ίσος με τη τιμή 0 κατά την επανεκκίνηση.

2) "**gauge**", ο οποίος αναπαριστά μια αριθμητική τιμή, η οποία έχει τη δυνατότητα να αυξομειώνεται ανά πάσα στιγμή, σε αντίθεση με τον counter. Μία συνήθης χρήση αυτού είναι η μέτρηση κάποιας θερμοκρασίας ή της χρήσης μνήμης την τρέχουσα χρονική στιγμή.

3) "**histogram**", που συγκεντρώνει παρατηρήσεις όπως για παράδειγμα τη χρονική διάρκεια κάποιων αιτημάτων και τις καταμετρά σε διαχειρίσιμα buckets.
Στον τύπο αυτόν δίνεται επιπλέον δυνατότητα παραμετροποίησης. Μπορεί να χρησιμοποιηθούν αθροιστικοί μετρητές για την παρατήρηση των buckets ως εξής:

- <basename>**_bucket**{le="<upper inclusive bound>"}, όπου basename κάποιο metric.
- "**total sum**", συγκεντρώνει το συνολικό πλήθος των παρατηρήσεων και μπορεί να χρησιμοποιηθεί ως: <basename>**_sum**.
- Η επιλογή "**count**" καταμετρά το πλήθος των events που έχουν παρατηρηθεί μέσω της σύνταξης: <basename>**_count**.

4) "**summary**" όμοια με τον "**histogram**", συγκεντρώνει παρατηρήσεις. Ωστόσο, υπολογίζει διαμορφώσιμα ποσοτικά μεγέθη σε ένα κυλιόμενο χρονικό παράθυρο.

- Η επιλογή "**φ-quantiles**", εκθέτει διάφορες παρατηρήσεις σε ένα Scrape μέσω της σύνταξης: <basename>{**quantile**="<φ>"}.
- Οι επιλογές "**total sum**" και "**count**", έχουν την ίδια χρήση με εκείνη των αντίστοιχων επιλογών του τύπου histogram.

Για να αποσαφηνιστούν κάποιες από τις παραπάνω έννοιες, αποτυπώνονται στο επόμενο στιγμιότυπο τα αποτελέσματα ενός Query που εκτελέστηκε από τη γραφική διεπαφή του Prometheus αναφορικά με μια μετρική.
Η συγκεκριμένη μετρική είναι το "alertmanager_alerts", είναι τύπου "gauge" και αναπαριστά τους τρέχοντες ειδοποιητικούς συναγερμούς που υπάρχουν ανά κατάσταση (state).

Στιγμιότυπο 2.1.1: Querying μέσω της γραφικής διεπαφής του Prometheus

Όπως γίνεται αντιληπτό, υπάρχουν διάφοροι τρόποι για να αποτυπώσει κανείς τις μετρικές. Ο ένας είναι εκείνος που μόλις παρουσιάστηκε και συγκεκριμένα μέσω της γραφικής διεπαφής (Web UI) του Prometheus στο πεδίο /graphs.
Ο δεύτερος τρόπος, αποτελεί η χρήση του εργαλείου "curl" μέσω της γραμμής εντολών (Command Line Interface). Για παράδειγμα, η παραπάνω μετρική που αναζητήσαμε, μπορεί να αποτυπωθεί με "curl" με την εντολή:

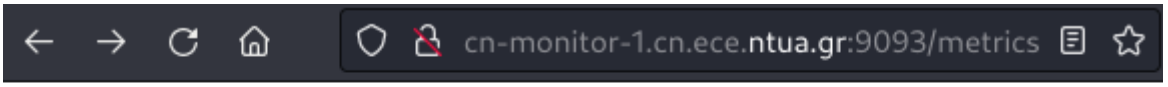
```
curl http://cn-monitor-1.cn.ece.ntua.gr:9093/metrics | grep alertmanager_alerts
```

Στο παρακάτω στιγμιότυπο διακρίνονται τα αποτελέσματα του αντίστοιχου "curl".

```
# HELP alertmanager_alerts How many alerts by state.
# TYPE alertmanager_alerts gauge
alertmanager_alerts{state="active"} 9
alertmanager_alerts{state="suppressed"} 0
```

Στιγμιότυπο 2.1.2: Αποτελέσματα εντολής "curl" με φίλτρο τη μετρική "alertmanager_alerts"

Ένας ακόμη τρόπος για να δούμε σε πραγματικό χρόνο τις μετρικές που είναι διαθέσιμες, είναι η χρήση ενός web browser [14]. Για τη συγκεκριμένη μετρική, αρκεί να πληκτρολογήσουμε το URL "http://cn-monitor-1.cn.ece.ntua.gr:9093/metrics" σε κάποιον web browser και αμέσως θα παρουσιαστούν όλα τα διαθέσιμα αποτελέσματα, όπως διακρίνεται στο επόμενο στιγμιότυπο.




```
# HELP alertmanager_alerts How many alerts by state.
# TYPE alertmanager_alerts gauge
alertmanager_alerts{state="active"} 10
alertmanager_alerts{state="suppressed"} 0
```

Στιγμιότυπο 2.1.3: Αποτελέσματα των μετρικών μέσω ενός web browser

Παρατηρεί κανείς, ότι οι τιμές ανάμεσα στα στιγμιότυπα διαφέρουν μεταξύ τους και αυτό, διότι, αφενός τα αποτελέσματα ελήφθησαν σε διαφορετικούς χρόνους και αφετέρου η μετρική πρόκειται για τύπου "**gauge**", η οποία αποθηκεύει την **τρέχουσα** τιμή της μέτρησης.

Για την προστασία των εξυπηρετητών, αν και τα δεδομένα που είναι διαθέσιμα μέσω του Prometheus exporter δεν χαρακτηρίζονται ως ευαίσθητα, υλοποιήθηκαν κανόνες ελέγχου πρόσβασης μέσω του λογισμικού UFW, όπως επισημαίνεται στο κεφάλαιο 6.9. Τα δεδομένα είναι διαθέσιμα μόνο σε μηχανήματα του εργαστηρίου Δικτύων Υπολογιστών.
Επομένως, συνίσταται η χρήση της πλατφόρμας του Prometheus, μέσα από την οποία υπάρχει άμεση πρόσβαση σε όλες τις μετρικές.

2.2: Prometheus Query Language (PromQL)

Το Prometheus, χρησιμοποιεί την δική του Query Language, η οποία ονομάζεται “PromQL” (Prometheus Query Language). Μέσω αυτής, δίνεται η δυνατότητα στον χρήστη να αποκτήσει και να επεξεργαστεί τα χρονικά δεδομένα, ώστε στη συνέχεια να εξαγάγει την πληροφορία που αναζητεί [15]. Στο κεφάλαιο 4.2 αναφορικά με την χρήση του Grafana παρουσιάζεται αναλυτικότερα ο τρόπος με τον οποίο αυτά χρησιμοποιούνται για την γραφική αναπαράσταση των δεδομένων.

Υπάρχουν τέσσερις τύποι εκφράσεων στη Query Language του Prometheus:

- instant vector
- range vector
- scalar
- string

Αναφορικά με τον “**Instant vector**”, πρόκειται για ένα σύνολο από Time Series δεδομένα, τα οποία περιέχουν ένα μόνο δείγμα για κάθε Time Series, αλλά μοιράζονται την ίδια χρονική σήμανση (timestamp). Πιο συγκεκριμένα, προσδίδουν την δυνατότητα στον χρήστη να επιλέξει όχι μόνο ένα σύνολο από τα Time Series δεδομένα, αλλά ταυτόχρονα και μια δεδομένη χρονική σφραγίδα (timestamp). Στην απλούστερη μορφή που αυτό μπορεί να χρησιμοποιηθεί είναι η χρήση μόνο του ονόματος μιας μετρικής, το οποίο θα έχει ως αποτέλεσμα να δημιουργήσει ένα διάνυσμα (vector) με στοιχεία για όλα τα Time Series δεδομένα που έχουν αυτό το όνομα.
Ένα τέτοιο παράδειγμα αποτελεί η μετρική “http_requests_total”, η οποία θα επιστρέψει στον χρήστη τα δεδομένα τα οποία αφορούν στα συνολικά HTTP αιτήματα που πραγματοποιήθηκαν.

Εάν κανείς επιθυμεί να φιλτράρει περαιτέρω τα δεδομένα, υπάρχει η επιλογή του άγκιστρου “**{}**”, μέσα στο οποίο μπορεί να προσδιοριστεί πιο συγκεκριμένα κάποιο χαρακτηριστικό για το Query. Ένα τέτοιο παράδειγμα αποτελεί το “http_requests_total{job="prometheus",group="node_exporter"}”, το οποίο Query προσδιορίζει ποιο “job” αλλά και ποιο “group” του Prometheus είναι προς ενδιαφέρον. Το συγκεκριμένο “job” αφορά στο service του Prometheus, ενώ το “group” στον Node exporter.

Προσδίδεται και επιπλέον ευελιξία σε αυτό το σκοπό μέσω κάποιον επιλογέων, όπως είναι το “**=**”, το οποίο επιλέγει τις ετικέτες, των οποίων το string αποδίδεται ύστερα από τον τελεστή αυτόν, αλλά και το “**!=**” το οποίο επιλέγει τις ετικέτες, των οποίων το string δεν είναι ίσο με εκείνο ύστερα από τον τελεστή αυτό.
Ο τελεστής “**=~**” επιλέγει τις ετικέτες, των οποίων η έκφραση (regex) ταιριάζει με το αποδιδόμενο string, ενώ ο “**!~**” επιλέγει τις ετικέτες, των οποίων η έκφραση δεν ταιριάζει με το αποδιδόμενο string.

Ο τύπος “**Range vector**”, χρησιμοποιώντας τον τελεστή “**[]**”, επιτρέπει την επιλογή όλων των τιμών που έχουν καταγραφεί το τελευταίο χρονικό διάστημα που ορίζεται εντός του τελεστή αυτού. Για παράδειγμα, το Query “http_requests_total{job="prometheus"}[5m]”, επιστρέφει στον χρήστη τα δεδομένα τα οποία αφορούν στα συνολικά HTTP αιτήματα που πραγματοποιήθηκαν τα τελευταία 5 λεπτά και που αφορούν στο “job” “prometheus”.

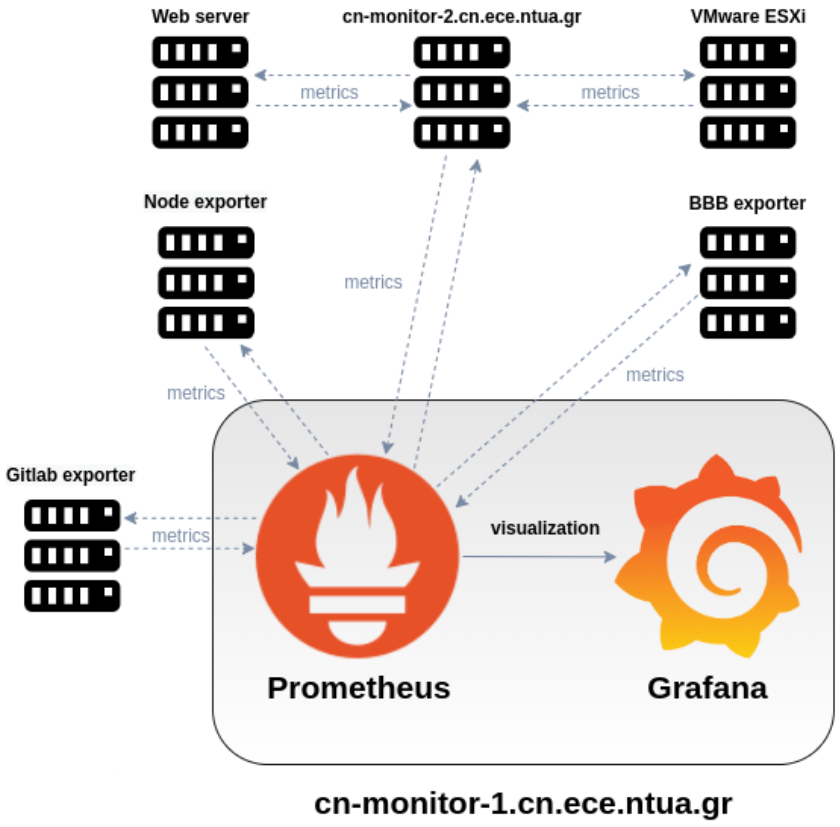
Τέλος, οι τύποι “**Scalar**” και “**String**”, αφορούν σε τιμή κινητής υποδιαστολής (floating point) και σε String αντίστοιχα. Περισσότερες πληροφορίες μπορεί να βρει κανείς στην ιστοσελίδα του Prometheus [15].

Στη συνέχεια του κεφαλαίου θα παρουσιαστούν οι διαφορετικοί τρόποι που χρησιμοποιήθηκαν για την παρακολούθηση συγκεκριμένων υπηρεσιών του εργαστηρίου.

2.3: Prometheus exporters

Όπως γίνεται αντιληπτό, η ιδιαιτερότητα, αλλά και η ποικιλία των συσκευών ή εφαρμογών, απαιτούν η καθεμία ξεχωριστά τη δική τους προσέγγιση, όσον αφορά την εξαγωγή και τη συγκέντρωση της ενδιαφέρουσας πληροφορίας. Εκτός τούτων, πολλές φορές σε κάποιες συσκευές, όπως για παράδειγμα κεντρικοί υπολογιστές τύπου Linux, δεν διατίθενται εκ των προτέρων η συμβατή μορφή μετρικών για το Prometheus. Για τον λόγο αυτό, απαιτείται ένας ενδιάμεσος exporter, ο οποίος θα κάνει την "μετάφραση", αν θέλετε, των μετρικών αυτών σε κατάλληλη μορφή για την αποθήκευση τους στο Prometheus.
Στο συγκεκριμένο παράδειγμα, αυτό επιτυγχάνεται μέσω του Node exporter [16]. Γενικότερα ωστόσο, αναλόγως την εφαρμογή, το έργο αυτό αναλαμβάνουν να υλοποιήσουν οι κατάλληλοι Prometheus exporters [17], [18].

Ονομάζονται έτσι ή πολλές φορές και ως "Prometheus Clients", διότι οι μετρικές τις οποίες συγκεντρώνουν και στη συνέχεια θέτουν ως διαθέσιμες, καθίστανται πλέον συμβατές με την Time Series βάση δεδομένων του Prometheus. Αυτό ορίζεται ως "Scrape".
Σε επόμενο κεφάλαιο, θα παρουσιαστούν διάφοροι exporters που τίθενται σε λειτουργία στα πλαίσια της διπλωματικής εργασίας. Στο επόμενο στιγμιότυπο, διακρίνεται η συνολική υποδομή που αναπτύχθηκε κατά τη διάρκεια της διπλωματικής εργασίας και που αφορά στο Prometheus και στην συλλογή των μετρικών από τις κατάλληλες εφαρμογές και τους αντίστοιχους exporters.



Στιγμιότυπο 2.3: Αναπαράσταση επικοινωνίας Prometheus με τους exporters

2.4: Node exporter

Ο στόχος του **Node exporter**, είναι να εκθέτει μετρικές από φυσικούς εξυπηρετητές και λογισμικά συστήματα [16].

Οι μετρικές αυτές φέρουν πληροφορίες, όπως είναι: το ποσοστό της χρησιμοποιούμενης μνήμης, το ποσοστό του ελεύθερου αποθηκευτικού χώρου, τον αριθμό των συνολικών δικτυακών bytes (network bytes) που λαμβάνονται ανά λεπτό κ.α.

Για την αποσαφήνιση των μετρικών που συλλέγονται από τον Node exporter, παρουσιάζονται στο κεφάλαιο 4.3 αναφορικά με το Grafana και τον Node exporter, ορισμένα παραδείγματα αυτών, μέσω γραφημάτων που δημιουργούνται με χρήση του Grafana.

Έχοντας πλέον ρυθμίσει τους προς επίβλεψη εξυπηρετητές, στην γραφική διεπαφή του Prometheus στο πεδίο "Status"→"Targets", διακρίνονται τα χαρακτηριστικά και η κατάσταση των hosts αυτών.
Στο παρακάτω στιγμιότυπο, η κατάσταση όλων τον Node exporters είναι λειτουργική (up).

node-exporter (10/10 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://bbb.cn.ntua.gr:9100/metrics	UP	instance="bbb.cn.ntua.gr:9100" job="node-exporter"	1.019s ago	24.95ms	
http://bbb2.cn.ntua.gr:9100/metrics	UP	instance="bbb2.cn.ntua.gr:9100" job="node-exporter"	2.567s ago	29.84ms	
http://bench-moodle-1:9100/metrics	UP	instance="bench-moodle-1:9100" job="node-exporter"	6.71s ago	63.43ms	
http://cn-monitor-1.cn.ece.ntua.gr:9100/metrics	UP	instance="cn-monitor-1.cn.ece.ntua.gr:9100" job="node-exporter"	9.202s ago	153.2ms	
http://cn-monitor-2.cn.ece.ntua.gr:9100/metrics	UP	instance="cn-monitor-2.cn.ece.ntua.gr:9100" job="node-exporter"	8.941s ago	185.4ms	
http://courses.cn.ece.ntua.gr:9100/metrics	UP	instance="courses.cn.ece.ntua.gr:9100" job="node-exporter"	11.839s ago	88.54ms	
http://gitlab.telecom.ece.ntua.gr:9100/metrics	UP	instance="gitlab.telecom.ece.ntua.gr:9100" job="node-exporter"	4.148s ago	27.71ms	
http://ladon.telecom.ece.ntua.gr:9100/metrics	UP	instance="ladon.telecom.ece.ntua.gr:9100" job="node-exporter"	8.292s ago	94.56ms	
http://onos.telecom.ece.ntua.gr:9100/metrics	UP	instance="onos.telecom.ece.ntua.gr:9100" job="node-exporter"	15.018s ago	139.1ms	
http://ulairi.telecom.ece.ntua.gr:9100/metrics	UP	instance="ulairi.telecom.ece.ntua.gr:9100" job="node-exporter"	7.337s ago	201ms	

Στιγμιότυπο 2.4.1: Κατάσταση του Node exporter μέσω της γραφικής διεπαφής του Prometheus

Σε περίπτωση που κάποιος δεν λειτουργεί σωστά, στη στήλη "Error", εμφανίζονται πληροφορίες σχετικά με το πρόβλημα που μπορεί να υπάρχει.

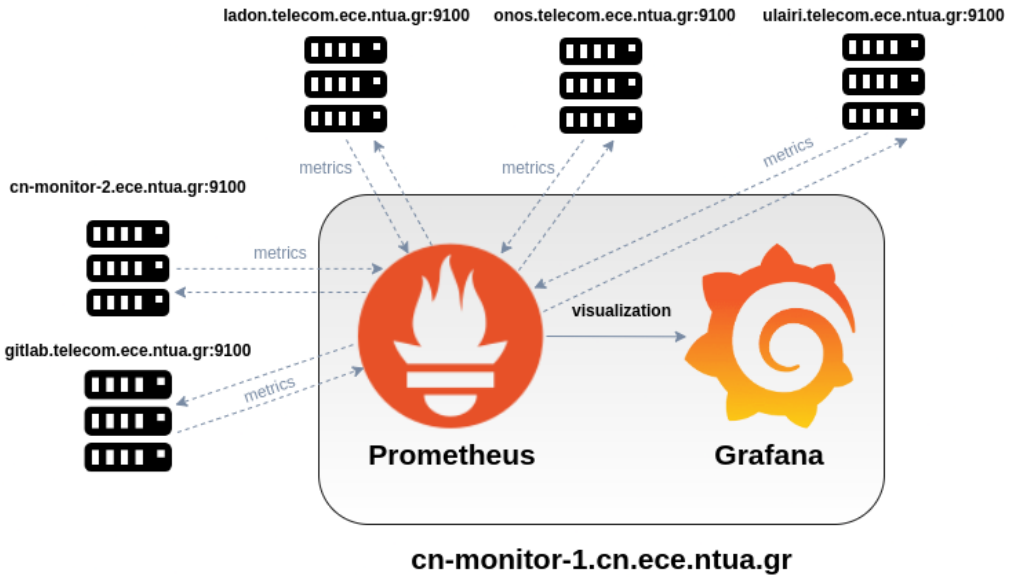
node-exporter (9/10 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://bbb.cn.ntua.gr:9100/metrics	UP	instance="bbb.cn.ntua.gr:9100" job="node-exporter"	8.323s ago	29.32ms	
http://bbb2.cn.ntua.gr:9100/metrics	UP	instance="bbb2.cn.ntua.gr:9100" job="node-exporter"	9.871s ago	27.9ms	
http://bench-moodle-1:9100/metrics	UP	instance="bench-moodle-1:9100" job="node-exporter"	14.013s ago	67.97ms	
http://cn-monitor-1.cn.ece.ntua.gr:9100/metrics	UP	instance="cn-monitor-1.cn.ece.ntua.gr:9100" job="node-exporter"	1.506s ago	149.9ms	
http://cn-monitor-2.cn.ece.ntua.gr:9100/metrics	DOWN	instance="cn-monitor-2.cn.ece.ntua.gr:9100" job="node-exporter"	1.244s ago	7.201ms	Get http://cn-monitor-2.cn.ece.ntua.gr:9100/metrics: dial tcp 147.102.40.79:9100: connect: connection refused
http://courses.cn.ece.ntua.gr:9100/metrics	UP	instance="courses.cn.ece.ntua.gr:9100" job="node-exporter"	4.143s ago	69.88ms	
http://gitlab.telecom.ece.ntua.gr:9100/metrics	UP	instance="gitlab.telecom.ece.ntua.gr:9100" job="node-exporter"	11.452s ago	28.12ms	
http://ladon.telecom.ece.ntua.gr:9100/metrics	UP	instance="ladon.telecom.ece.ntua.gr:9100" job="node-exporter"	596ms ago	91.37ms	
http://onos.telecom.ece.ntua.gr:9100/metrics	UP	instance="onos.telecom.ece.ntua.gr:9100" job="node-exporter"	7.322s ago	146.8ms	
http://ulairi.telecom.ece.ntua.gr:9100/metrics	UP	instance="ulairi.telecom.ece.ntua.gr:9100" job="node-exporter"	14.641s ago	198.6ms	

Στιγμιότυπο 2.4.2: Κατάσταση του Node exporter όταν κάποιος host δεν λειτουργεί

Εάν κανείς επιθυμεί να παρακολουθήσει συσκευές τύπου Windows, διατίθεται στη κοινότητα του διαδικτύου και αντίστοιχος **Windows exporter** [19].

Στο παρακάτω στιγμιότυπο, παρουσιάζεται μια υποδομή που περιλαμβάνει ένα Prometheus και ένα Grafana, τα οποία φιλοξενούνται στον εξυπηρετητή "cn-monitor-1.cn.ece.ntua.gr". Το Prometheus λαμβάνει μέσω "HTTP pulling" τις μετρικές που θέτουν ως διαθέσιμες οι απεικονιζόμενοι hosts στην θύρα (port) "9100". Γίνεται εμφανές, πως δεν απαιτείται κάποιος ενδιάμεσος εξυπηρετητής, ώστε να είναι ληφθούν οι μετρικές. Αυτό, διότι λαμβάνονται από τον κάθε εξυπηρετητή ξεχωριστά, εφόσον έχει εγκατασταθεί ο Node exporter σε καθέναν.



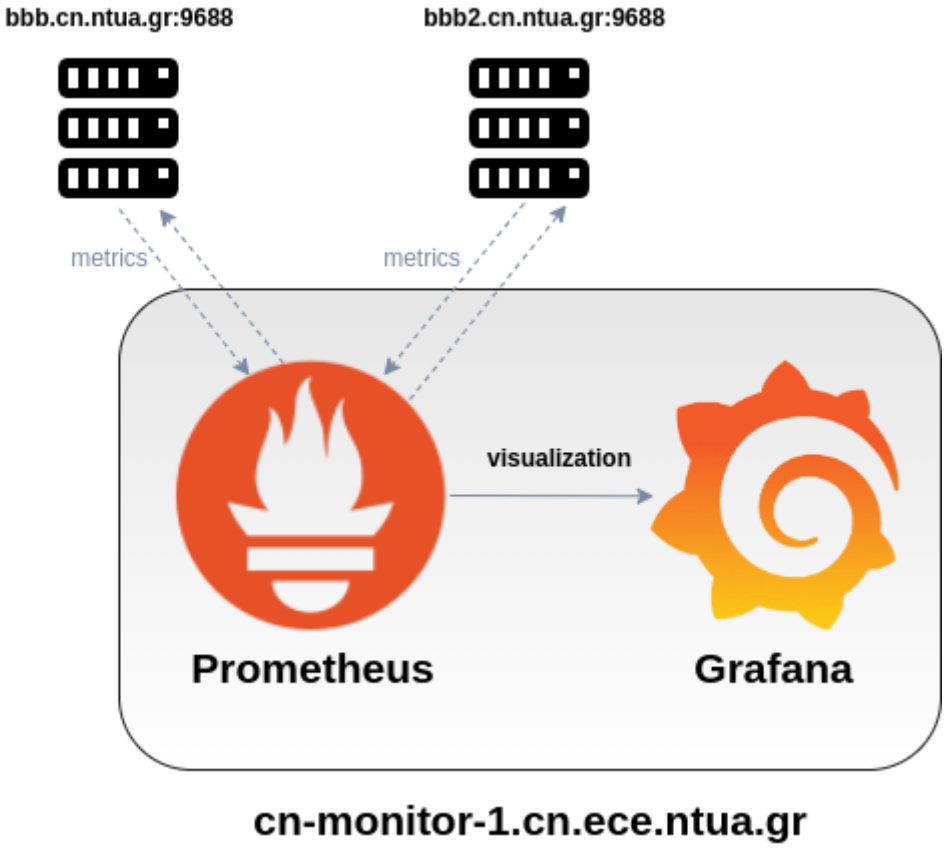
Στιγμιότυπο 2.4.3: Αναπαράσταση επικοινωνίας Prometheus με Node exporters

2.5: BigBlueButton exporter

Σε περίπτωση που κάποιος διαχειριστής επιθυμεί να παρακολουθήσει έναν εξυπηρετητή που φιλοξενεί την πλατφόρμα τηλεκπαίδευσης BigBlueButton, τότε για το σκοπό αυτό διατίθεται ο BigBlueButton exporter [20].

Στιγμιότυπο 2.5.1: Λογότυπο BigBlueButton

Στα πλαίσια της παρούσης διπλωματικής εργασίας, οι εξυπηρετητές αυτοί είναι οι “bbb.cn.ntua.gr” και “bbb2.cn.ntua.gr”. Επομένως, το “pulling” των μετρικών από τον Prometheus πραγματοποιείται με τις αντίστοιχες IP και την θύρα “9688” που “ακούει” ο exporter, όπως διακρίνεται στο επόμενο στιγμιότυπο.



Στιγμιότυπο 2.5.2: Αναπαράσταση επικοινωνίας Prometheus με BBB exporters

Οι μετρικές του BigBlueButton exporter φέρουν πληροφορίες σχετικά με τα στοιχεία που αφορούν σε έναν εξυπηρετητή που φιλοξενεί το λογισμικό τηλεκπαίδευσης BigBlueButton.

Ορισμένα από αυτά είναι: το πλήθος των συμμετεχόντων σε βιντεοκλήσεις, το πλήθος των recordings, το πλήθος των εικονικών δωματίων τηλεδιάσκεψης κ.α.

Για την αποσαφήνιση των μετρικών που συλλέγονται από τον BigBlueButton exporter, παρουσιάζονται στο κεφάλαιο 4.4 αναφορικά με το Grafana και τον BigBlueButton exporter, ορισμένα παραδείγματα αυτών, μέσω γραφημάτων που δημιουργούνται με χρήση του Grafana.

Ύστερα από την εγκατάσταση, μέσα από την γραφική διεπαφή του Prometheus, παρατηρούμε την κατάσταση και τα χαρακτηριστικά του BigBlueButton exporter.

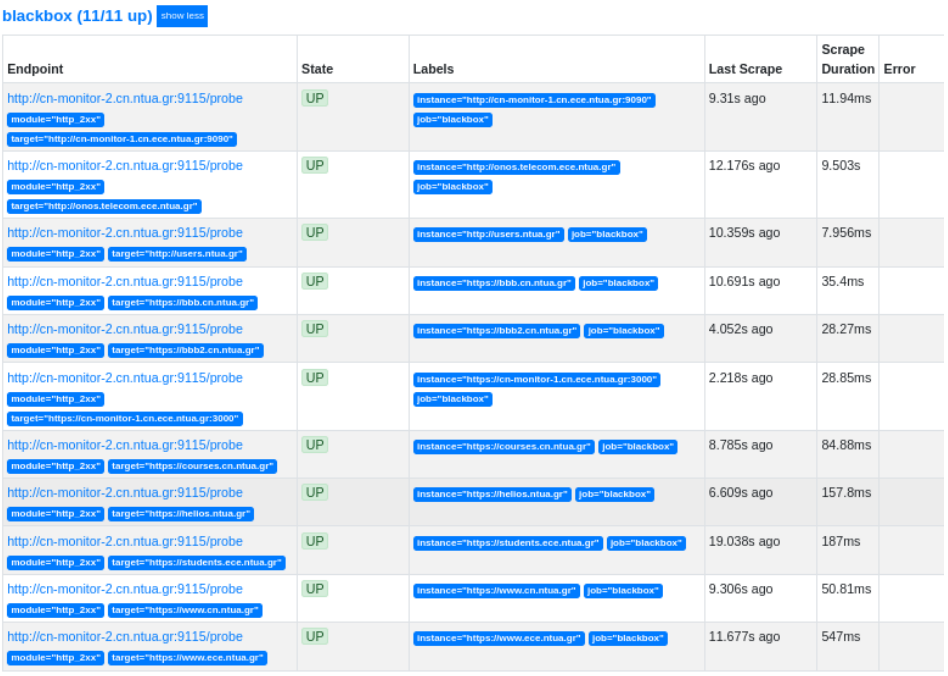
Στιγμιότυπο 2.5.3: Κατάσταση του BBB exporter μέσω της γραφικής διεπαφής του Prometheus

2.6: Blackbox exporter

Στα πλαίσια της παρούσης διπλωματικής εργασίας, ένας ακόμη στόχος αποτελεί η παρακολούθηση διαφόρων εξυπηρετητών ιστού. Για το λόγο αυτό, θα χρησιμοποιηθεί ένας HTTP(S)-exporter. Υπάρχουν διάφοροι τέτοιοι, όπως είναι ο Prometheus Blackbox exporter, ο Apache exporter ή ο Nginx exporter [21] , [22], [23].

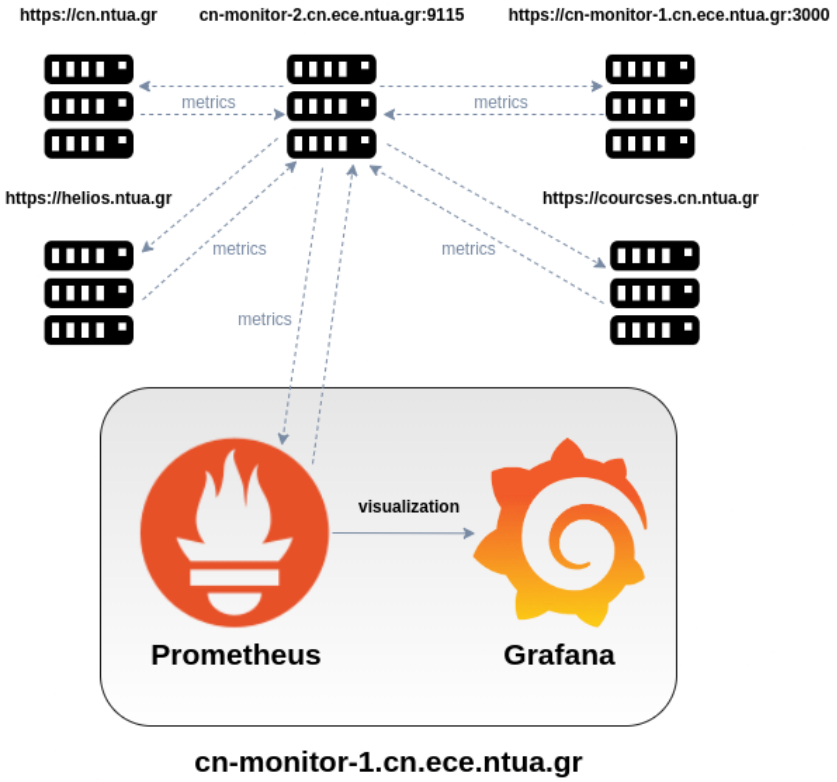
Από τους παραπάνω, επιλέχθηκε ο Prometheus Blackbox exporter. Ο exporter αυτός, επιτρέπει το "probing" over HTTP, HTTPS, DNS, TCP, ICMP και gRPC. Το χαρακτηριστικό του "probing" προσφέρει την ευελιξία, να χρησιμοποιείται ο exporter χωρίς να απαιτείται η εγκατάστασή του σε κάθε host που είναι προς μελέτη. Επιπλέον, ώντας εγκατεστημένος κεντρικά σε κάποιον εξυπηρετητή, επιτελείται μονάχα η διαδικασία του "probing" και τα δεδομένα λαμβάνονται απομακρυσμένα.

Οι μετρικές που συλλέγει ο Blackbox exporter φέρουν πληροφορίες σχετικά με τα στοιχεία που αφορούν σε έναν εξυπηρετητή ιστού. Ορισμένα από αυτά είναι: η ημερομηνία λήξης του SSL πιστοποιητικού, το HTTP response status, το πλήθος των HTTP ανακατευθύνσεων (HTTP redirects), την έκδοση (version) του IP πρωτοκόλλου κ.α.
Για την αποσαφήνιση των μετρικών που συλλέγονται από τον Node exporter, παρουσιάζονται στο κεφάλαιο 4.5 αναφορικά με το Grafana και τον Blackbox exporter, ορισμένα παραδείγματα αυτών, μέσω γραφημάτων που δημιουργούνται με χρήση του Grafana.
Ύστερα από την εγκατάσταση, μέσα από τη γραφική διεπαφή του Prometheus, παρατηρούμε την κατάσταση και τα χαρακτηριστικά του Blackbox exporter.



Στιγμιότυπο 2.6.1: Κατάσταση του Blackbox exporter μέσω της γραφικής διεπαφής του Prometheus

Σε αυτό το σημείο θα πρέπει να σημειωθεί, ότι ο Blackbox exporter εγκαθίσταται σε έναν ενδιάμεσο εξυπηρετητή, τον “cn-monitor-2” και ύστερα ο Prometheus λαμβάνει από αυτόν μέσω “HTTP pulling” τις μετρικές που θέτει ως διαθέσιμες στην θύρα “9115”. Στο επόμενο στιγμιότυπο παρουσιάζεται η διαδικασία αυτή για ορισμένους hosts.



Στιγμιότυπο 2.6.2: Αναπαράσταση επικοινωνίας Prometheus με Blackbox exporter

Σε αυτό το σημείο θα πραγματοποιηθεί μια αναφορά στο “**multi-target exporter pattern**” που υποστηρίζει το Prometheus [24].
Σε αντίθεση με τους υπόλοιπους exporters, στον SNMP exporter αλλά και στον Blackbox exporter, δύναται να χρησιμοποιηθεί το pattern αυτό. Πιο συγκεκριμένα, για να λάβουμε τις μετρικές μέσω “curl”, στους άλλους exporters, αρκούσε να εκτελέσουμε:

```
curl http://<desired_IP>:<desired_port>/metrics
```

Στον Blackbox exporter, θα πρέπει να δοθούν ως παράμετροι ένα “**target**” αλλά και ένα “**module**” για το GET αίτημα. Το “target” αποτελείται από ένα URI (Uniform Resource Identifier), είτε μια διεύθυνση IP, ενώ το “module” ορίζεται στο αρχείο παραμετροποίησης του exporter. Στην περίπτωσή μας, το module ισούται με “‘http_2xx”, το οποίο ορίζει στον exporter να περιμένει το “200 OK” ως response στο αίτημά του.

Με βάση τα παραπάνω, μέσω “curl” μπορούμε για παράδειγμα να λάβουμε τις μετρικές της ιστοσελίδας “courses.cn.ntua.gr” με την εντολή:

```
curl -vvv
'http://cn-monitor-2.cn.ntua.gr:9115/probe?target=https://courses.cn.ntua.gr&module=http_2xx'
```

Περισσότερες πληροφορίες σχετικά με τις επιλογές του “probing”, παρέχονται στην ιστοσελίδα του Blackbox exporter [25].

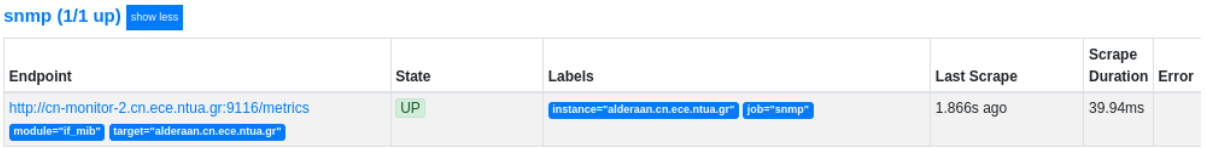
2.7: SNMP exporter

Για την παρακολούθηση δικτυακών συσκευών, όπως είναι ένας μεταγωγέας, απαιτείται ένας κατάλληλος exporter, ο οποίος θα μετατρέπει τις ενδιαφέρουσες μετρικές σε κατάλληλη διαχειρίσιμη μορφή για το Prometheus.

Για παράδειγμα, μεταγωγείς τύπου Cisco, δεν υποστηρίζουν τον Node exporter με αποτέλεσμα να πρέπει να ληφθούν οι μετρικές με άλλο τρόπο.
Αυτό επιτυγχάνεται με τη χρήση ενός ενδιάμεσου εξυπηρετητή, ο οποίος θα συγκεντρώνει τις ενδιαφέρουσες μετρικές.
Τον ρόλο αυτό, αναλαμβάνει ο "cn-monitor-2", στον οποίο είναι εγκατεστημένη η υλοποίηση ενός SNMP exporter από το Github [26].

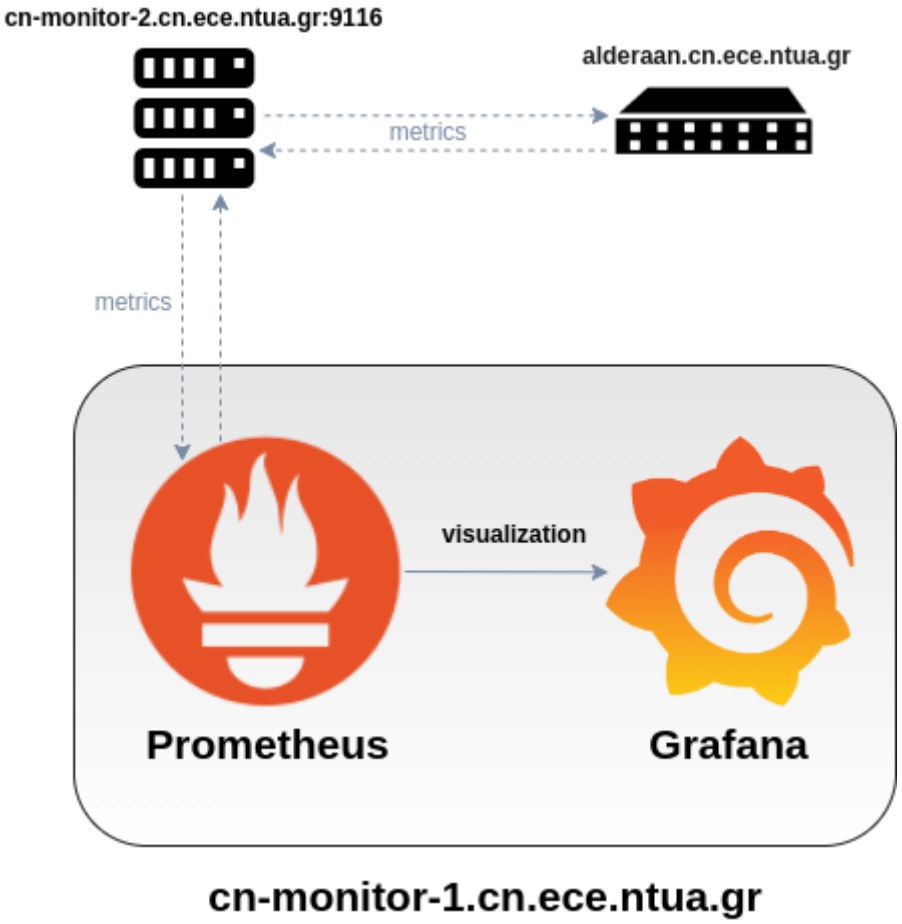
Θα πρέπει να σημειωθεί, ότι ο SNMP exporter συγκεντρώνει πληροφορίες που σχετίζονται με μια SNMP συσκευή, όπως είναι οι μεταγωγείς. Τα δεδομένα αυτά είναι δομημένα με βάση τα "OID trees" και περιγράφονται από τις MIBs [27].

Ύστερα από την εγκατάσταση του exporter στον εξυπηρετητή "cn-monitor-2", μέσα από τη γραφική διεπαφή του Prometheus, παρατηρούμε την κατάσταση και τα χαρακτηριστικά του SNMP exporter.



Στιγμιότυπο 2.7.1: Κατάσταση του SNMP exporter μέσω της γραφικής διεπαφής του Prometheus

Παρατηρεί κανείς, ότι και σε αυτήν τη περίπτωση ότι ο SNMP exporter εγκαθίσταται σε έναν ενδιάμεσο εξυπηρετητή, τον "cn-monitor-2" και ύστερα το Prometheus λαμβάνει από αυτόν μέσω "HTTP pulling" τις μετρικές που θέτει ως διαθέσιμες στην θύρα "9116", όπως διακρίνεται στο επόμενο στιγμιότυπο.



Στιγμιότυπο 2.7.2: Αναπαράσταση επικοινωνίας Prometheus με SNMP exporter

Όπως προαναφέρθηκε, στο κεφάλαιο περί Blackbox exporter, ο SNMP exporter υποστηρίζει το **multi-target exporter pattern** που υποστηρίζει το Prometheus. Γι' αυτό το σκοπό, παρομοίως μέσω "curl" μπορούμε να λάβουμε τις μετρικές ως εξής:

```
curl -vvv
http://cn-monitor-2.cn.ece.ntua.gr:9116/metrics?target=alderaan.cn.ece.ntua.gr&module=if_mib'
```

Σε αυτό το σημείο να σημειωθεί, ότι για την επίβλεψη SNMP συσκευών είναι προτιμότερη η υλοποίηση μέσω InfluxDB και Telegraf [28]. Γι 'αυτό το λόγο, η υλοποίηση μέσω Prometheus δεν πραγματοποιείται στα πλαίσια της διπλωματικής εργασίας.

2.8: VMware exporter

Αναφορικά με τη διαδικασία της παρακολούθησης σε εξυπηρετητές τύπου VMware ESXi, απαιτείται και σε αυτή τη περίπτωση ένας κατάλληλος exporter, ο οποίος θα μετατρέπει τις ενδιαφέρουσες μετρικές σε κατάλληλη διαχειρίσιμη μορφή για το Prometheus.

Στιγμιότυπο 2.8.1: Λογότυπο VMware

Αυτό, επιτυγχάνεται με τη χρήση ενός ενδιάμεσου εξυπηρετητή, ο οποίος θα συγκεντρώνει τις μετρικές μιας και δεν περιλαμβάνεται κάποιος Prometheus exporter από τη VMware. Τον ρόλο αυτό, αναλαμβάνει ο εξυπηρετητής "cn-monitor-2", στον οποίο είναι εγκατεστημένη η υλοποίηση ενός VMware exporter από το Github [29]. Ο ίδιος εξυπηρετητής χρησιμοποιείται για το σύνολο των ESXi του εργαστηρίου.
Ορισμένες από τις μετρήσεις αφορούν σε πληροφορίες που σχετίζονται με τον φυσικό ESXi host όπως: το ποσοστό χρήσης της τρέχουσας μνήμης, το ποσοστό χρήσης του δίσκου, το πλήθος των συνολικών εικονικών μηχανημάτων (VMs) κ.α.
Ωστόσο, πέραν αυτών των πληροφοριών, διατίθενται και μετρικές που σχετίζονται με τα εικονικά μηχανήματα, πληροφορίες των οποίων μπορεί να είναι: ο αριθμός των εικονικών μηχανημάτων που βρίσκονται εν λειτουργία, ο αριθμός των εικονικών CPUs (vCPUs) ανά VM, το ποσοστό χρήσης της τρέχουσας μνήμης κάθε VM κ.α.

Για την αποσαφήνιση των μετρικών που συλλέγονται από τον VMware exporter, παρουσιάζονται στο κεφάλαιο 4.6 αναφορικά με το Grafana και τον VMware exporter, ορισμένα παραδείγματα αυτών μέσω γραφημάτων που δημιουργούνται με χρήση του Grafana.

Ύστερα από την εγκατάσταση του VMware exporter στον εξυπηρετητή "cn-monitor-2", μέσα από τη γραφική διεπαφή του Prometheus, παρατηρούμε την κατάσταση και τα χαρακτηριστικά του VMware exporter.

vmware_vcenter (1/1 up) show less

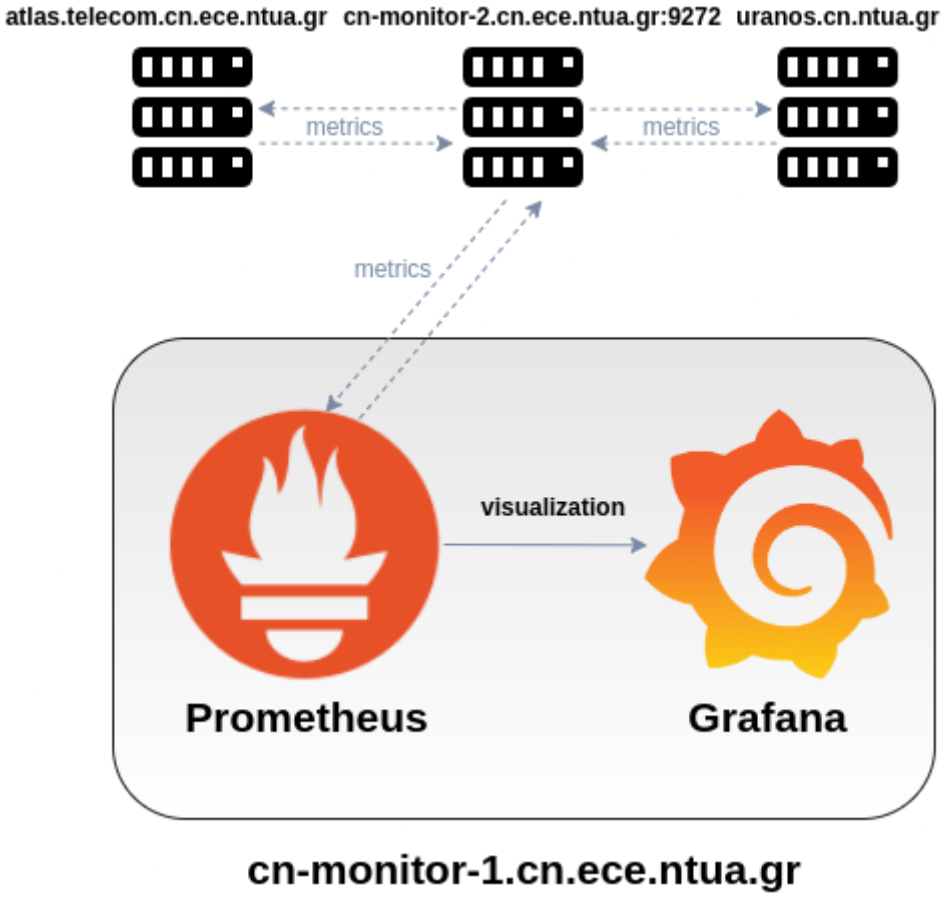
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://147.102.40.79:9272/metrics section="atlas" target="atlas.telecom.ece.ntua.gr"	UP	instance="atlas.telecom.ece.ntua.gr" job="vmware_vcenter"	11.935s ago	617.7ms	

vmware_vcenter2 (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://147.102.40.79:9272/metrics section="uranos" target="uranos.cn.ntua.gr"	UP	instance="uranos.cn.ntua.gr" job="vmware_vcenter2"	15.367s ago	614.1ms	

Στιγμιότυπο 2.8.2: Κατάσταση του VMware exporter μέσω της γραφικής διεπαφής του Prometheus

Ομοίως και σε αυτή τη περίπτωση, ο VMware exporter εγκαθίσταται σε έναν ενδιάμεσο εξυπηρετητή, τον “cn-monitor-2” και ύστερα το Prometheus λαμβάνει από αυτόν μέσω “HTTP pulling” τις μετρικές που θέτει ως διαθέσιμες στην θύρα “9272”, όπως διακρίνεται στο επόμενο στιγμιότυπο.



Στιγμιότυπο 2.8.3: Αναπαράσταση επικοινωνίας Prometheus με VMware exporter

Άλλες επιλογές για τη χρήση VMware exporter πέραν του VMware exporter του Daniel Pryor [29], είναι εκείνοι του Kugathasan Janarthanan και του Cloudchef [30], [31].

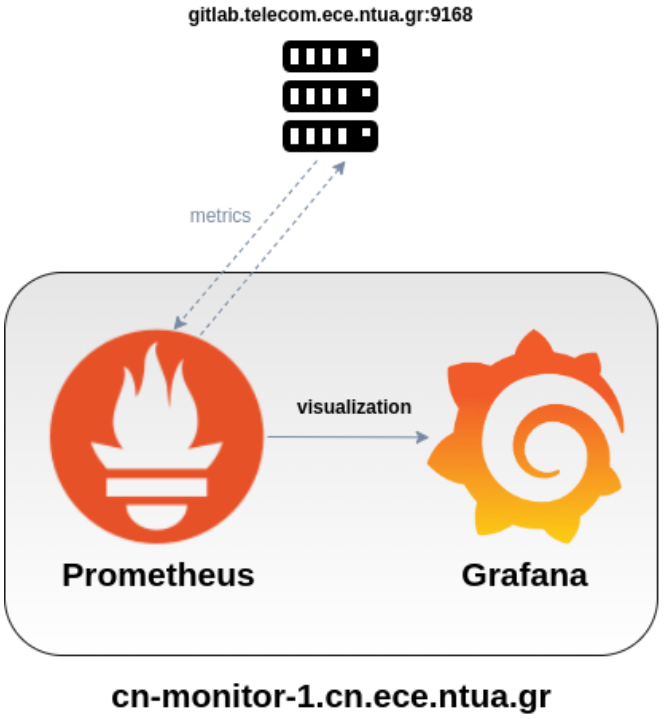
2.9: Gitlab exporter

Για την παρακολούθηση εξυπηρετητών που φιλοξενούν την υπηρεσία “Gitlab”, χρησιμοποιείται ο “Gitlab exporter” [32].

Για την χρήση του “Gitlab exporter” είναι απαραίτητη η προεγκατάσταση του “Gitlab” και στην συνέχεια η ρύθμιση του αρχείου παραμετροποίησής του, για την ενεργοποίηση του Gitlab exporter.

Στιγμιότυπο 2.9.1: Λογότυπο Gitlab

Αναφορικά με τον Gitlab exporter, εγκαθίσταται στον εξυπηρετητή που φιλοξενεί την υπηρεσία του Gitlab, εν προκειμένω, στον “gitlab.telecom.ece.ntua.gr”.
Επομένως και σε αυτή τη περίπτωση το “pulling” των μετρικών από το Prometheus επιτυγχάνεται απευθείας από την διεύθυνση του “gitlab.telecom.ece.ntua.gr” και συγκεκριμένα από την θύρα “9168”, όπως διακρίνεται από το παρακάτω στιγμιότυπο.



Στιγμιότυπο 2.9.2: Αναπαράσταση επικοινωνίας Prometheus με Gitlab exporter

Ο Gitlab exporter αποτελεί έναν “Prometheus Web exporter”, ο οποίος έχει τα παρακάτω χαρακτηριστικά:

- Συλλέγει τις μετρικές τύπου “Gitlab production”, μέσω probes.
- Τα “custom probes” έχουν τη δυνατότητα να συλλέγουν μετρήσεις με τη μορφή ζεύγους “key/value”.
- Για κάθε probe, δημιουργείται ένα “HTTP endpoint” “/<probe_name>” από προεπιλογή στην θύρα “9168”, όπου μεταφέρονται οι μετρικές σε έναν “Prometheus scraper”.

Οι μετρικές του Gitlab exporter φέρουν πληροφορίες σχετικά με την “Sidekiq queue”, καθώς και τα “Gitlab database rows”. Στο κεφάλαιο 4.7 αναφορικά με το Grafana και τον Gitlab exporter, δίδονται ορισμένα παραδείγματα αυτών μέσω γραφημάτων που δημιουργούνται με χρήση του Grafana.

Κεφάλαιο 3: Μηχανισμός ειδοποιητικών συναγερμών (Alerting)

Έχοντας αναπτύξει το σύστημα παρακολούθησης των μετρικών, επόμενο βήμα αποτελεί η ταχεία ενημέρωση των διαχειριστών για πιθανές αστοχίες ή διακοπές στη λειτουργία των υπηρεσιών, γεγονός που επιτυγχάνεται μέσω της αποστολής ειδοποιητικών συναγερμών με τη βοήθεια διαφόρων chat πλατφορμών, όπως είναι το Slack ή το PagerDuty. Στην υλοποίηση της παρούσης διπλωματικής χρησιμοποιείται η πλατφόρμα Slack.

3.1: Prometheus Alertmanager

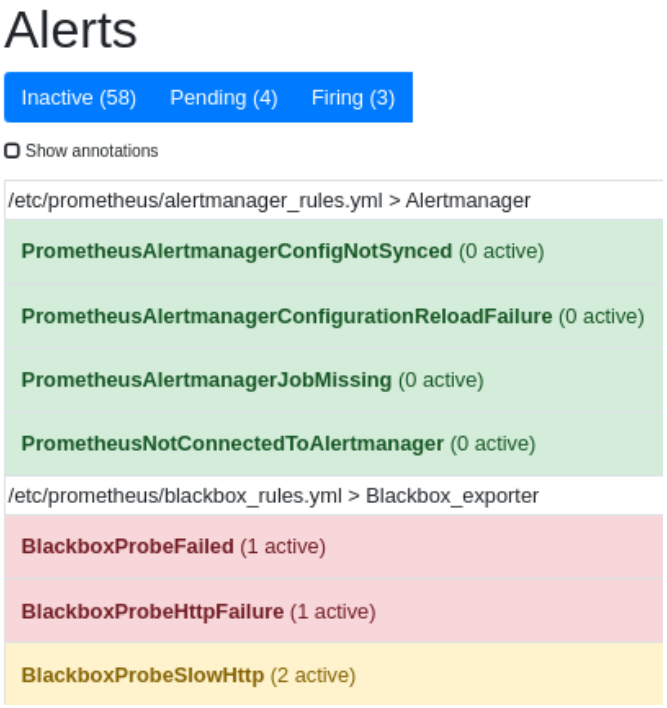
Για να επιτευχθεί αυτός ο σκοπός, είναι απαραίτητη η εγκατάσταση του **Prometheus Alertmanager** και η σύνδεσή του με το Prometheus [33].

Ο ρόλος του Prometheus Alertmanager, είναι να διαχειρίζεται διάφορους ειδοποιητικούς συναγερμούς (Alerts), που λαμβάνει από client εφαρμογές, όπως εν προκειμένω το Prometheus.

Κάποια σημαντικά χαρακτηριστικά του, περί της επεξεργασίας των ειδοποιητικών συναγερμών είναι τα εξής:

- Grouping: η δυνατότητα ομαδοποίησης σε ένα, εδιοποιητικών συναγερμών με όμοιο σκοπό, ώστε να υπάρχει καλύτερη ανάλυση του προβλήματος.
- Inhibition: η δυνατότητα καταστολής συγκεκριμένων ειδοποιητικών συναγερμών την ώρα που αποστέλλονται σημαντικότερες ειδοποιήσεις που αφορούν στο ίδιο πρόβλημα. (Για παράδειγμα, σε περίπτωση που μια βάση δεδομένων δεν είναι διαθέσιμη, θα ενεργοποιηθούν πολλοί συναγερμοί που αφορούν στο πρόβλημα αυτό, αποκρύπτοντας ωστόσο την πραγματική πηγή της δυσλειτουργίας.)
- Silence: η δυνατότητα σίγασης συγκεκριμένων ειδοποιητικών συναγερμών για ένα χρονικό διάστημα.

Η κατάσταση των συναγερμών αυτών, όσον αφορά εάν είναι ανενεργοί (inactive), σε εκκρεμότητα (pending), ή ενεργοί (firing), σημειώνεται με πράσινο, πορτοκαλί ή κόκκινο, αντίστοιχα χρώμα και μπορεί να ελεγχθεί από τη γραφική διεπαφή του Prometheus στην καρτέλα “/alerts”. Παρακάτω, παρουσιάζεται ένα στιγμιότυπο από την κατάσταση ορισμένων ειδοποιητικών συναγερμών.



Στιγμιότυπο 3.1.1: Αποτελέσματα ειδοποιητικών συναγερμών μέσω της γραφικής διεπαφής του Prometheus

Από τη γραφική διεπαφή του Prometheus, μπορεί κανείς στην καρτέλα "/rules", να παρατηρήσει τους κανόνες του Prometheus Alertmanager που έχουν ενσωματωθεί στο Prometheus, καθώς και τη σύνταξή τους, όπως φαίνεται από το παρακάτω στιγμιότυπο.

Στιγμιότυπο 3.1.2: Κανόνας του Alertmanager όπως διακρίνεται μέσω της διεπαφής του Prometheus

3.2: Ειδοποιητικοί συναγερμοί μέσω του Prometheus Alertmanager

Για λόγους πληρότητας, στο παρόν κεφάλαιο παρουσιάζεται ένας εναλλακτικός τρόπος αποστολής των ειδοποιητικών συναγερμών πέραν της παρακολούθησης και ανάλυσης αυτών μέσω του λογισμικού Alerta και συγκεκριμένα αφορά στην χρήση της πλατφόρμας **Slack**.

Στιγμιότυπο 3.2.1: Λογότυπο του Slack

Παρακάτω, απεικονίζεται ένα στιγμιότυπο με το αποτέλεσμα των ειδοποιήσεων που λαμβάνονται από τον Prometheus Alertmanager και αποτυπώνονται στο κανάλι του Slack, χρησιμοποιώντας μια προσαρμοσμένη συμβατή μορφή κειμένου (Text format).

Πρόκειται για δύο κρίσιμους συναγερμούς (critical Alerts), οι οποίοι αφορούν στο job "blackbox" του Prometheus. Επιπλέον, πληροφορούν πως το "probing" του Blackbox exporter απέτυχε, καθώς και ότι ο "HTTP status" κωδικός της ιστοσελίδας "http://onos.telecom.ece.ntua.gr" δεν είναι ανάμεσα στις τιμές 200-399.

Στιγμιότυπο 3.2.2: Ειδοποιητικοί συναγερμοί αποτυπωμένοι στο Slack

3.3: Alerta

Για την καλύτερη δυνατή αξιοποίηση των ειδοποιητικών συναγερμών που δημιουργούνται μέσω του Prometheus Alertmanager, χρησιμοποιείται στα πλαίσια της παρούσης εργασίας το λογισμικό **Alerta** [34]. Επιλέχθηκε το λογισμικό Alerta για τη συλλογή συναγερμών από διάφορες πηγές εκτός του Prometheus Alertmanager (όπως πχ. το Nagios) και τη παρουσίαση τους σε μία οθόνη στους διαχειριστές του εργαστηρίου.

Στιγμιότυπο 3.3.1: Λογότυπο Alerta

Το λογισμικό αυτό, είναι μια επεκτάσιμη λύση αναφορικά με την παρακολούθηση και οπτικοποίηση ειδοποιήσεων. Η γραφική διεπαφή του Alerta παρέχει μια ευδιάκριτη αποτύπωση των ειδοποιητικών συναγερμών, το οποίο είναι ιδιαίτερα χρήσιμο σε περίπτωση που απαιτείται η παρακολούθηση διαφόρων συστημάτων παρακολούθησης (monitoring systems) συγχρόνως. Συγκεκριμένα το Alerta υποστηρίζει ενσωματώσεις με Prometheus, Riemann, Nagios, Zabbix και άλλα συστήματα.
Επιπλέον, παρέχει τη δυνατότητα να δεχθεί οποιουδήποτε τύπου ειδοποιητικό συναγερμό, ο οποίος αποστέλλεται μέσω URL αιτήματος από οποιοδήποτε σύστημα. Στη συνέχεια, οι συναγερμοί αυτοί αποθηκεύονται τοπικά χρησιμοποιώντας ως βάση δεδομένων MongoDB και υποβάλλονται σε μορφή “JSON” σε ένα HTTP API.

Στα πλαίσια της διπλωματικής εργασίας, αυτό επιτυγχάνεται μέσω “Webhook”. Πρόκειται για ένα HTTP αίτημα τύπου POST, κατά το οποίο ο Prometheus Alertmanager αποστέλλει τους συγκεντρωμένους ειδοποιητικούς συναγερμούς στο API του Alerta. Το Webhook ορίζεται κατάλληλα στο αρχείο παραμετροποίησης “alertmanager.yml” του Prometheus Alertmanager.

Ένα ακόμη σημαντικό χαρακτηριστικό είναι η δυνατότητα “συσχέτισης” που υποστηρίζει, κατά την οποία ειδοποιήσεις με το ίδιο “environment” και “resource” θεωρούνται ως διπλότυπες, εάν διαθέτουν την ίδια κρισιμότητα (severity).

Σε περίπτωση που το παραπάνω αληθεύει, οι συναγερμοί συσχετίζονται ως ένας, έτσι ώστε να αποτυπώνονται μόνο οι πιο πρόσφατες ειδοποιήσεις.

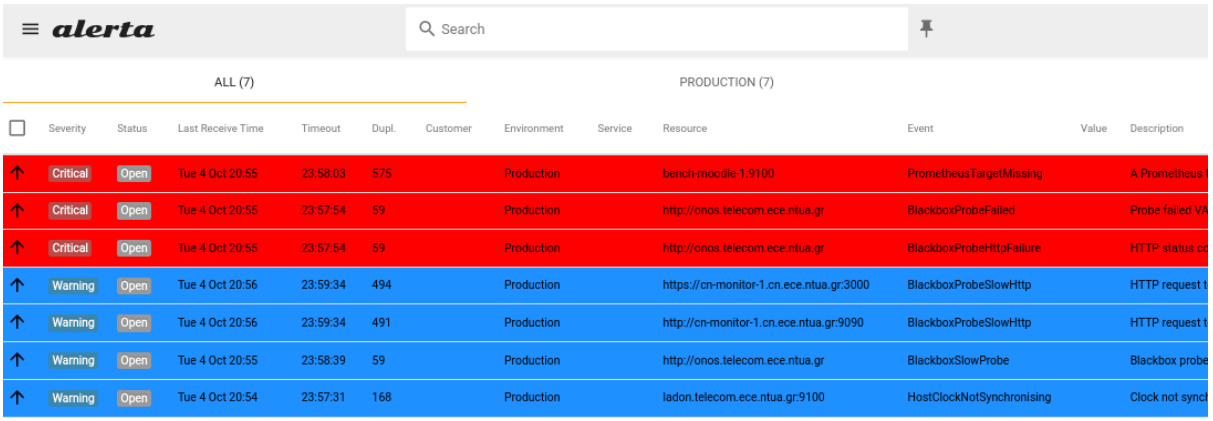
Ένα ακόμη από τα σημαντικά πλεονεκτήματα του Alerta, έγκειται στο γεγονός ότι επιτρέπει σε έναν ειδοποιητικό συναγερμό να συσχετιστεί με πολλές υπηρεσίες, να έχει οποιονδήποτε

αριθμό από “tags” σε οποιαδήποτε μορφή, καθώς και να προσδιοριστούν οσαδήποτε “attributes” επιθυμεί κανείς.

Κάποια από τα σημαντικότερα attributes των ειδοποιητικών συναγερμών που υποστηρίζει το Alerta είναι τα: resource, event, severity και status.

- ❖ resource: υποδηλώνει την εφαρμογή ή τον host.
- ❖ event: τίτλος περιγραφής του ειδοποιητικού συναγερμού.
- ❖ severity: κρισιμότητα του ειδοποιητικού συναγερμού. Η προκαθορισμένη τιμή είναι το “normal” με “severity code” ίσο με 9, ενώ οι υπόλοιπες μπορεί να είναι οι: security, critical, major, minor, warning, informational, debug, trace, indeterminate, cleared, ok και unknown, οι οποίες φέρουν “severity code” με τιμές στο εύρος 0 έως 10.
- ❖ status: υποδηλώνει την κατάσταση του ειδοποιητικού συναγερμού. Μπορεί να είναι open, assign, ack, closed, expired, blackout, shelved, unknown με τιμές “status code” 1 έως 9 αντίστοιχα.

Εν αντιθέσει με τη γραφική διεπαφή του Prometheus, το Alerta προσφέρει μια όσο το δυνατόν πιο οργανωμένη απεικόνιση για την καλύτερη οπτικοποίηση των ειδοποιητικών συναγερμών. Παρακάτω, διακρίνεται ένα στιγμιότυπο από την παρουσίαση των ειδοποιητικών συναγερμών μέσω του Alerta.



Στιγμιότυπο 3.3.2: Ειδοποιητικοί συναγερμοί αποτυπωμένοι στη γραφική διεπαφή του Alerta

Παρατηρεί κανείς, ότι η πλατφόρμα του Alerta προσδίδει ιδιαίτερη, οργανωμένη και ευδιάκριτη απεικόνιση των ειδοποιητικών συναγερμών, σε σχέση με την διεπαφή του Prometheus.

Κεφάλαιο 4: Οπτικοποίηση μετρικών

4.1: Grafana

Έχοντας συλλέξει **Time Series** δεδομένα και συγκεκριμένα **μετρικές** που αφορούν σε διάφορες εφαρμογές, κρίνεται συνετή, η ανάλυση αυτών με εύχρηστο και ποιοτικό τρόπο. Για τον λόγο αυτό, χρησιμοποιείται το λογισμικό Grafana [35].

Στιγμιότυπο 4.1: Λογότυπο Grafana

Το Grafana ξεκίνησε ως “fork project” του Kibana, για την καλύτερη ανάλυση time series δεδομένων, όπως είναι οι μετρικές, κάτι το οποίο το Kibana δεν παρείχε πριν την δημιουργία του Grafana.

Το Grafana πρόκειται για ένα εργαλείο, το οποίο προσφέρει τη δυνατότητα οπτικοποίησης (visualization) των μετρικών, μέσω γραφικών παραστάσεων, γραφημάτων, ιστογραμμάτων, αλλά και άλλων μορφών. Η γραφική διεπαφή του, παρέχει ευελιξία, μέσω της ελευθερίας αξιοποίησης και οργάνωσης της διαθέσιμης πληροφορίας. Πιο συγκεκριμένα, διατίθεται η δυνατότητα δημιουργίας προσαρμοσμένων panel για κάθε χρήση, αλλά και η οργάνωση αυτών σε dashboards, ανάλογα με την εργασία που αυτά αντιπροσωπεύουν. Σημειώνεται ότι, ένα panel αποτελεί μια γραφική αναπαράσταση της πληροφορίας, ενώ ένα dashboard περιλαμβάνει διάφορα panel.
Ένα πλεονέκτημα του Grafana, έγκειται στο γεγονός, ότι δεν απαιτείται επιπλέον αποθηκευτικός χώρος για την αποθήκευση των δεδομένων πέραν της βάσης δεδομένων που συγκεντρώνει τις μετρικές. Αυτό, διότι έχοντας εξασφαλίσει την σύνδεση μεταξύ Grafana και Prometheus μέσω της ρύθμισης του URL από το πεδίο των ρυθμίσεων, τα δεδομένα που έχουν συλλεχθεί στον Prometheus είναι διαθέσιμα μέσω “**Query pulling**”, από την διεύθυνση “http://cn-monitor-1.cn.ece.ntua.gr:9090”.

Επιπλέον, αναλόγως με την βάση δεδομένων (data source) που χρησιμοποιείται για την συλλογή των μετρικών, χρησιμοποιείται και η αντίστοιχη Query Language για την προσπέλαση αυτών. Στα πλαίσια της παρούσης εργασίας, γίνεται χρήση του Prometheus ως Time Series βάση δεδομένων, με αποτέλεσμα το Grafana να χρησιμοποιεί την PromQL για να λάβει τις μετρικές και στη συνέχεια να τις επεξεργαστεί.

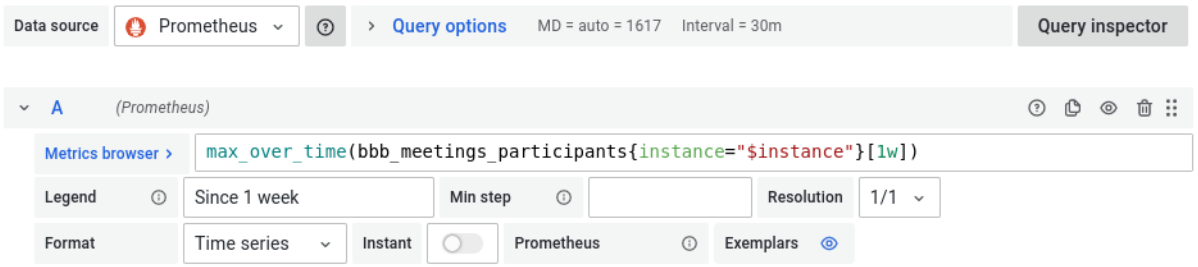
4.2: Χρήση του Grafana

Για την αναπαράσταση των δεδομένων, απαιτείται η δημιουργία και χρήση κατάλληλων dashboards. Στην ιστοσελίδα του Grafana, διατίθενται διάφορα dashboards έτοιμα για χρήση αναλόγως με το έργο που υποστηρίζουν. Για την παρούσα εργασία, χρησιμοποιήθηκαν διαθέσιμα dashboards για την ανάλυση δεδομένων του Node exporter, του BigBlueButton exporter, αλλά και για τον Prometheus Alertmanager [36], [37], [38]. Ωστόσο, παρότι διαθέτουν έτοιμα panel για ορισμένες μετρικές, δεν περιλαμβάνουν όλες τις μετρικές που συλλέγονται.

Για τον λόγο αυτό, τροποποιήθηκαν, ώστε να συμπεριληφθούν και να αναλυθούν όσο περισσότερες μετρικές σε κάθε dashboard είναι δυνατό.
Για να αναζητηθούν και να ληφθούν υπόψη όλες οι μετρικές, χρησιμοποιήθηκε το εργαλείο "curl". Ύστερα, όσες είναι ενδιαφέρουσες και ουσιαστικές για την διαδικασία της παρακολούθησης, χρησιμοποιούνται σε ξεχωριστά panel για να αναπαρασταθούν.

Η υποβολή ενός διαθέσιμου dashboard στο Grafana, επιτυγχάνεται από το κεντρικό μενού στην επιλογή "Create"⟶"import"⟶"Upload JSON file". Στη συνέχεια, το dashboard που δημιουργήθηκε είναι διαθέσιμο από το κεντρικό μενού στην επιλογή "Dashboards"⟶"Browse".
Έπειτα, τροποποιούνται κατάλληλα τα panel με βάση το επιθυμητό αποτέλεσμα.

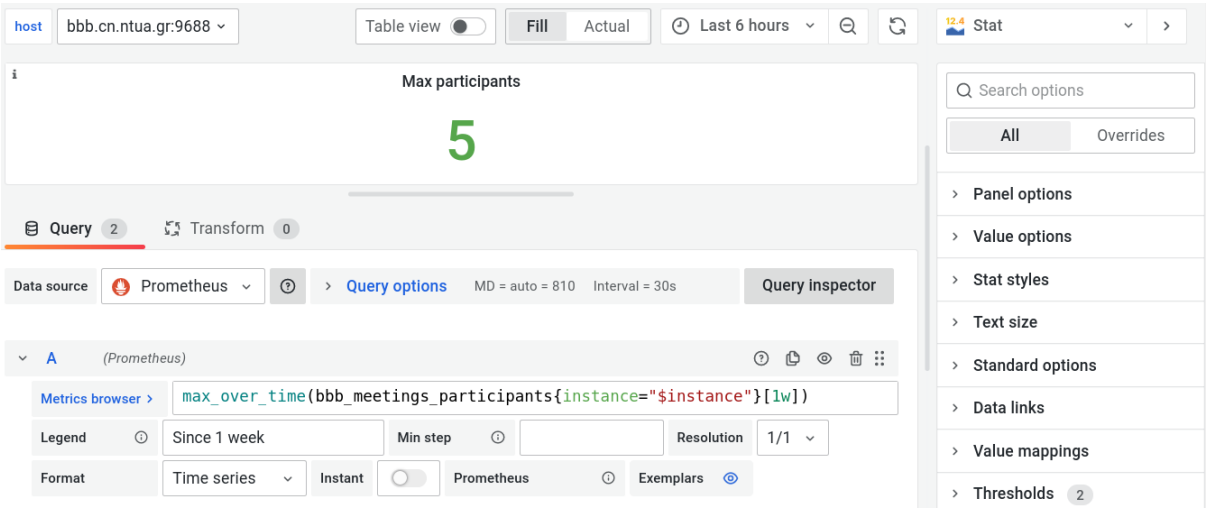
Όπως προαναφέρθηκε, για την ανάκτηση των μετρικών χρησιμοποιείται η Query Language PromQL. Δύναται να παρατηρήσει κανείς, ότι το Query επιτελείται σε ένα panel, μέσω του πεδίου "Metrics browser" όπως διακρίνεται στο παρακάτω στιγμιότυπο.



Στιγμιότυπο 4.2.1: Query μετρικής μέσω "panel" στο Grafana

Στη συγκεκριμένη περίπτωση, παρατηρεί κανείς ότι το "Data source" που έχει επιλεχθεί είναι το Prometheus. Επιπλέον, στο πεδίο "Metrics browser", επιτελείται το Query: "**max_over_time(bbb_meetings_participants{instance="$instance"}[1w])**".
Η μετρική, αποτελεί η παράμετρος **"bbb_meetings_participants"**, που δηλώνει τον αριθμό των συμμετεχόντων σε συνεδρίες του BigBlueButton, ενώ το "max_over_time" είναι μια συνάρτηση του Prometheus, η οποία επιστρέφει την μέγιστη τιμή των δειγμάτων που έχουν ληφθεί σε κάποιο χρονικό διάστημα που ορίζεται σε αγκύλες, εν προκειμένω στο διάστημα μιας εβδομάδος ([1w]).

Λαμβάνοντας υπόψη, ότι η μετρική είναι τύπου “gauge”, δηλαδή μια αριθμητική τιμή που μπορεί να αυξομειώνεται στο πέρας του χρόνου, επιλέγεται για τον τρόπο οπτικοποίησης στο Grafana, ένα panel τύπου “Stat”.
Το αποτέλεσμα αυτού του panel, παρουσιάζεται στην παρακάτω εικόνα.



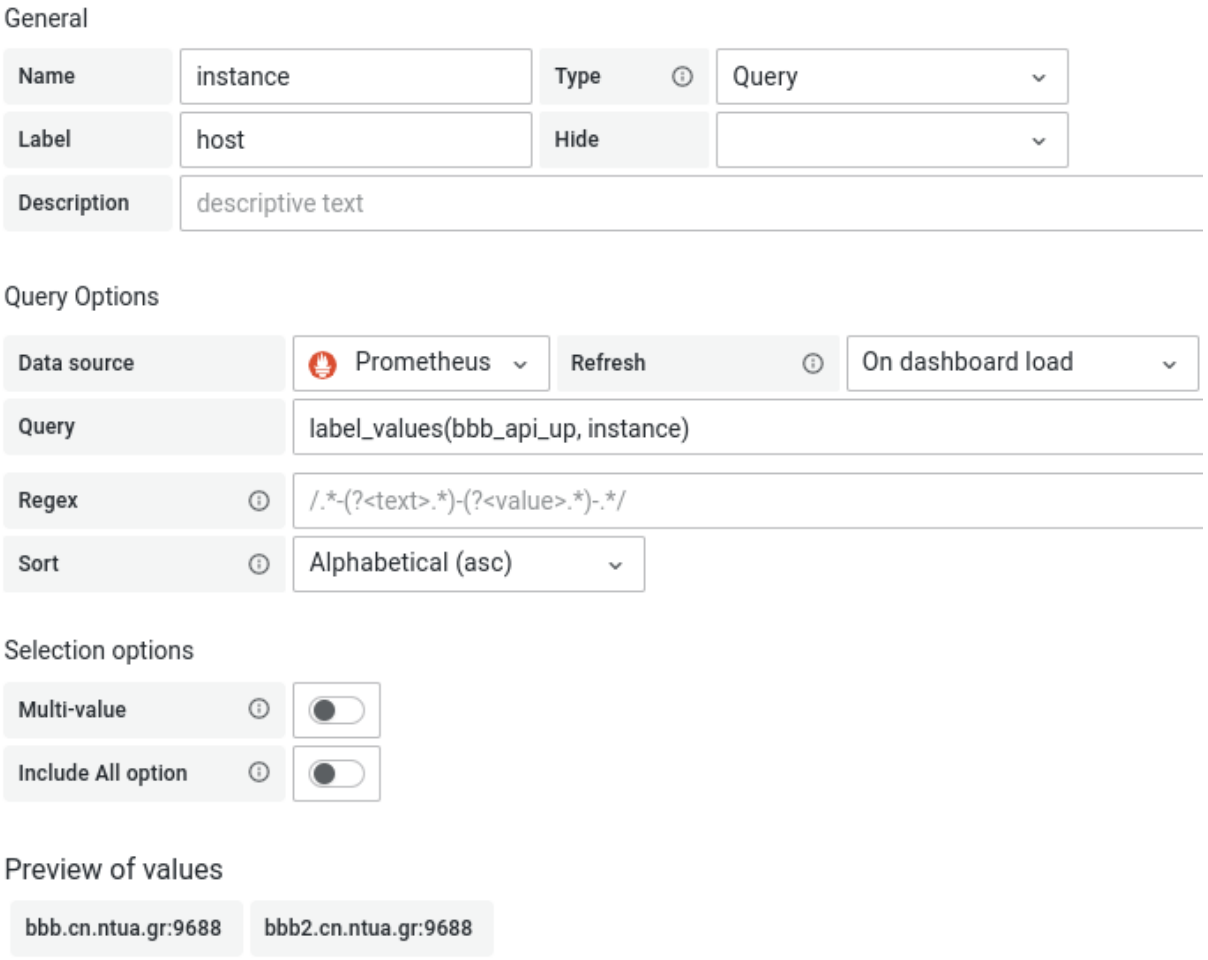
Στιγμιότυπο 4.2.2: Ένα “panel” και οι επιλογές του

Όπως διακρίνεται παραπάνω, η επιλογή του τύπου του panel γίνεται στην πάνω και δεξιά πλευρά του panel, όπου έχει επιλεχθεί το “Stat”, ενώ περαιτέρω επιλογές για το panel που αφορούν στην παρουσίασή του, προσδίδονται από το δεξί μέρος του panel.

Ακόμη, θα πρέπει να τονιστεί, ότι στην πάνω και αριστερή πλευρά της εικόνας, αναγράφεται η επιλογή “host” και έχει επιλεχθεί ο host “bbb.cn.ntua.gr:9688”. Το στοιχείο αυτό, αφορά στην επιλογή συγκεκριμένου εξυπηρετητή που φιλοξενεί τον BigBlueButton exporter του οποίου λαμβάνονται οι μετρικές.

Το πεδίο “host”, θα μπορούσε να έχει οποιαδήποτε ονομασία, αυτό που έχει όμως αυστηρή σημασία είναι πως αφορά ένα “**instance**” και αυτό είναι που διακριτοποιεί ένα Query μεταξύ διαφορετικών εξυπηρετητών.

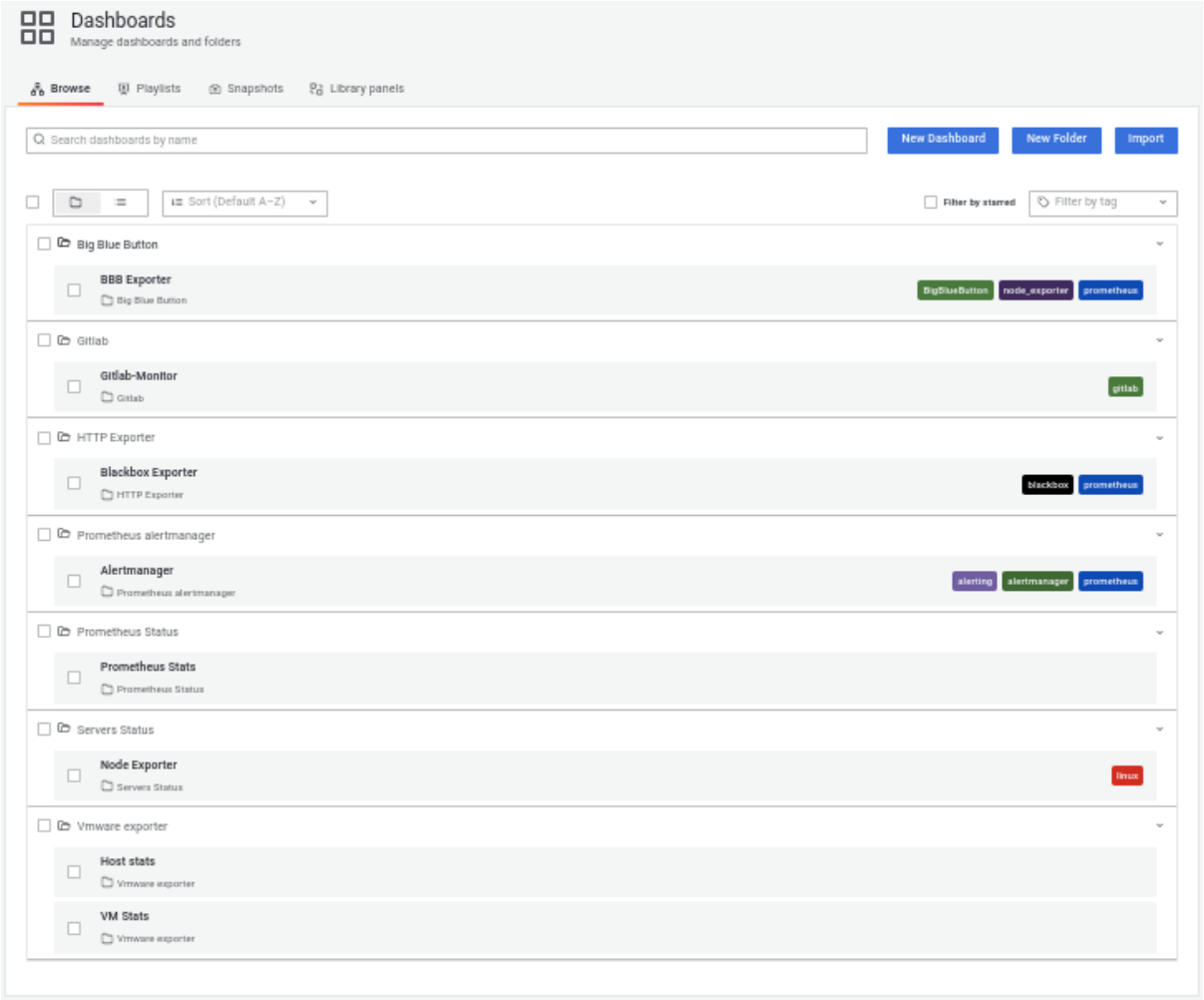
Ειδικότερα, στο πεδίο που αναγράφεται το Query, χρησιμοποιείται η σύνταξη “**instance=$instance**”, η οποία επιστρέφει τη μετρική που ανήκει στον εξυπηρετητή που έχει επιλεχθεί στο πεδίο “host”.
Η ρύθμιση των κατάλληλων επιλογών που αφορούν στο instance του συγκεκριμένου dashboard, πραγματοποιείται από το πεδίο “Dashboard settings”⟶“Variables”⟶“new”, όπως διακρίνεται από το επόμενο στιγμιότυπο.



Στιγμιότυπο 4.2.3: Ρύθμιση των "variables" ενός dashboard

Παρατηρούμε, ότι το πεδίο "Label" αφορά στο όνομα που εμφανίζεται στο "drop down" μενού για την επιλογή του host, όπως φαίνεται στο στιγμιότυπο 4.2.3. Το πεδίο "Name" περιλαμβάνει την μεταβλητή instance, η οποία χρησιμοποιείται στο Query για την επιλογή του κατάλληλου BBB εξυπηρετητή.
Η παραπάνω ρύθμιση πραγματοποιείται αντίστοιχα σε κάθε dashboard που χρησιμοποιείται για να καλύψει την εκάστοτε ανάγκη.

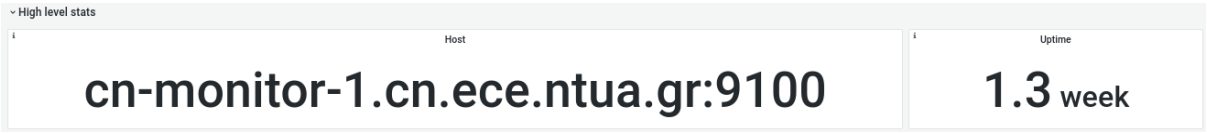
Για την καλύτερη δυνατή οργάνωση όλων των dashboard, δημιουργήθηκαν διάφοροι φάκελοι με βάση την εφαρμογή που επιθυμείται να εποπτευθεί.
Για το σκοπό αυτό, οι παρακάτω φάκελοι, όπως φαίνονται από το επόμενο στιγμιότυπο, αφορούν στον BigBlueButton exporter, στον Gitlab exporter, στον Blackbox exporter, στον Prometheus Alertmanager, στον Prometheus, στον Node exporter, καθώς και στον VMware exporter.



Στιγμιότυπο 4.2.4: Οργάνωση των dashboards του Grafana σε φακέλους

4.3: Grafana και Node exporter

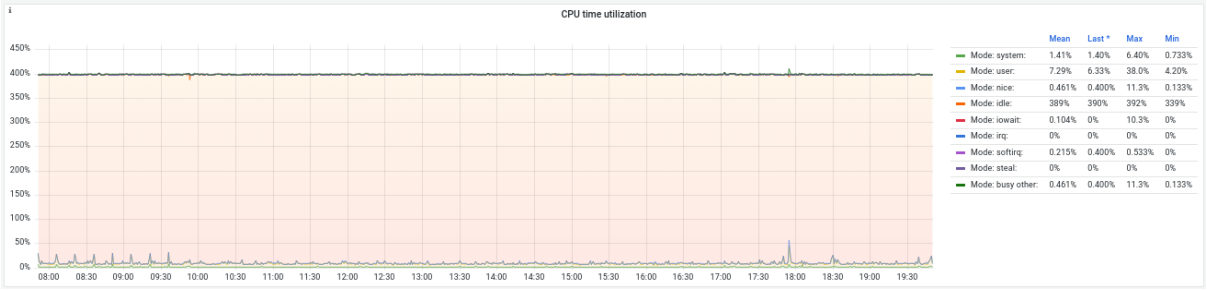
Έχοντας συγκεντρώσει τις απαραίτητες μετρικές που αφορούν στον Node exporter, επόμενο βήμα αποτελεί η οπτικοποίηση αυτών μέσω του Grafana. Αρχικά, στο dashboard του Node exporter, τα panel είναι ομαδοποιημένα με βάση το περιεχόμενο που αυτά αντιπροσωπεύουν. Παρακάτω, παρουσιάζονται ορισμένα στιγμιότυπα από τη γραφική διεπαφή του Grafana που αφορούν στις σημαντικότερες πληροφορίες που εξάγει ο Node exporter. Τα παρακάτω panel, ανήκουν στην ομάδα "High level stats", αναγράφουν τον (Linux) host για τον οποίο εξάγεται η πληροφορία, αλλά και τον χρόνο που εκείνος βρίσκεται σε λειτουργία.



Στιγμιότυπο 4.3.1: Όνομα host και χρόνος λειτουργίας του

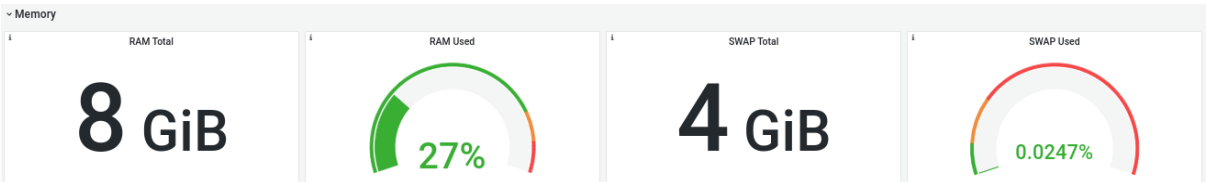
Μια ομάδα panel αποτελεί η "CPU", η οποία μεταφέρει πληροφορίες σχετικά με την CPU του host, όπως είναι η χρονική της χρήση ή το πλήθος των CPUs.

Στιγμιότυπο 4.3.2: Στατιστικά της CPU

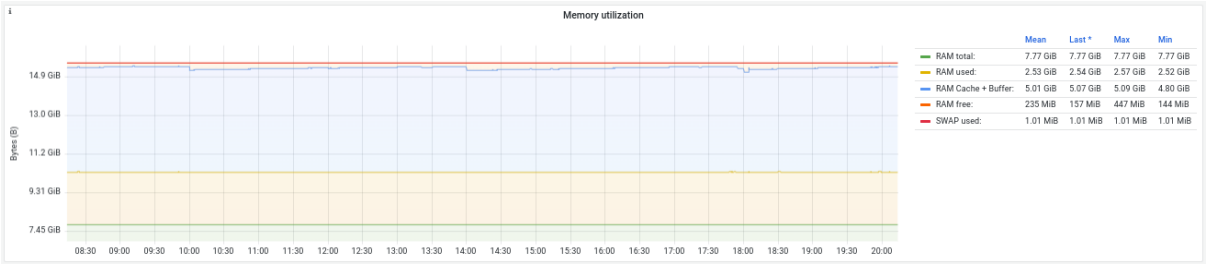


Στιγμιότυπο 4.3.3: Χρονική χρήση της CPU ανά τύπο (mode)

Η κατηγορία "Memory" φέρει πληροφορίες για την μνήμη του συστήματος, όπως είναι ενδεικτικά, το μέγεθος της διαθέσιμης μνήμης σε GB ή η ποσοστιαία χρήσης της.

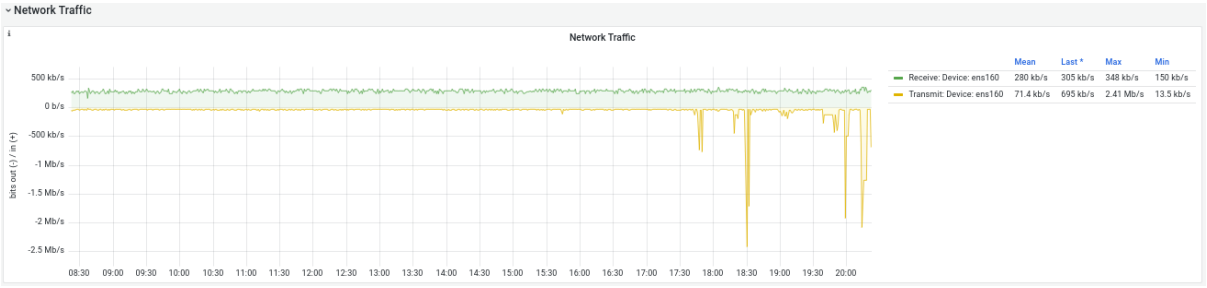


Στιγμιότυπο 4.3.4: Στατιστικά της μνήμης

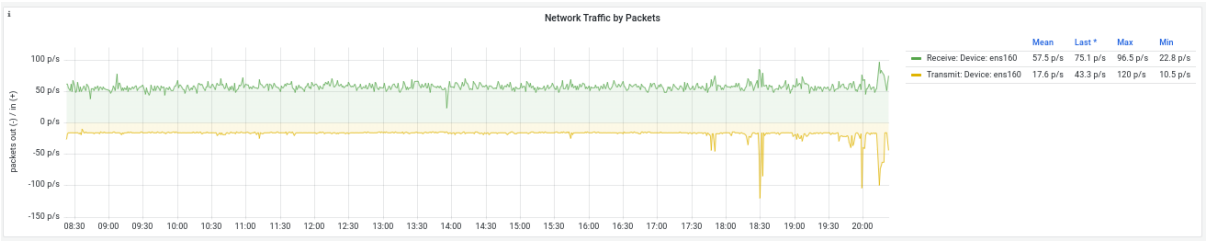


Στιγμιότυπο 4.3.5: Χρήση της μνήμης ανά κατηγορία, σε GB

Η κατηγορία “Network Traffic” φέρει πληροφορίες για την δικτυακή κίνηση του συστήματος, όπως είναι ενδεικτικά, η κίνηση σε Bytes/sec ή σε packets/sec.

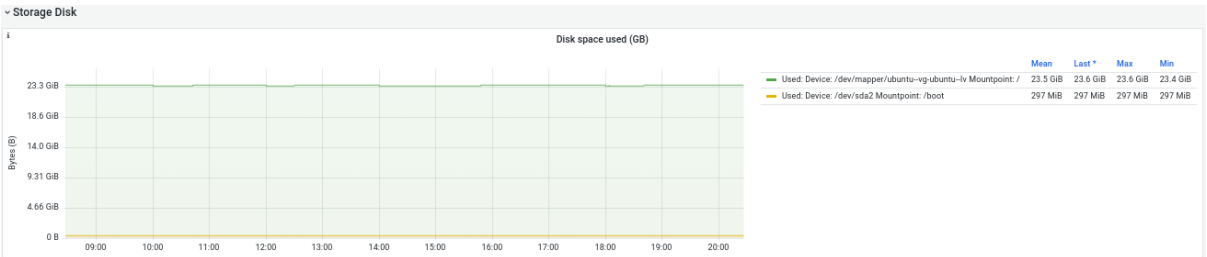


Στιγμιότυπο 4.3.6: Δικτυακή κίνηση σε Bytes/sec

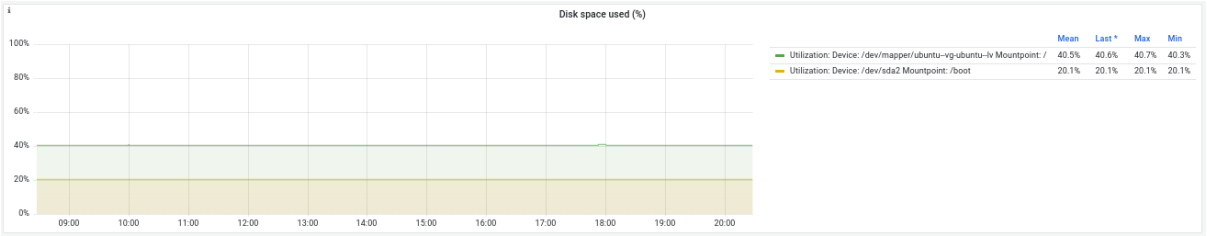


Στιγμιότυπο 4.3.7: Δικτυακή κίνηση σε Packets/sec

Τέλος, μια ακόμη κατηγορία αποτελεί η “Storage Disk”, panel της οποίας αναπαριστούν στοιχεία, όπως είναι ο χρησιμοποιούμενος αποθηκευτικός χώρος σε GB είτε μετρούμενος ποσοστιαία.



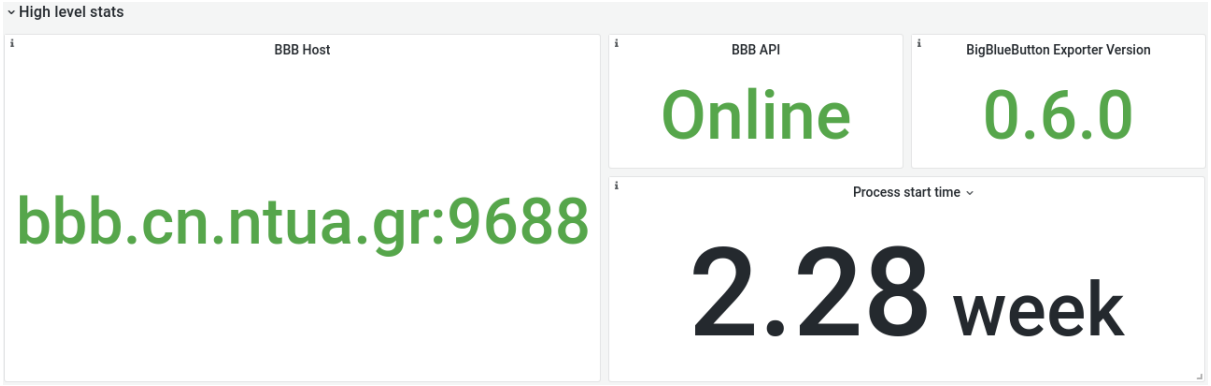
Στιγμιότυπο 4.3.8: Χρησιμοποιούμενος αποθηκευτικός χώρος σε GB



Στιγμιότυπο 4.3.9: Χρησιμοποιούμενος αποθηκευτικός χώρος σε ποσοστιαία κλίμακα

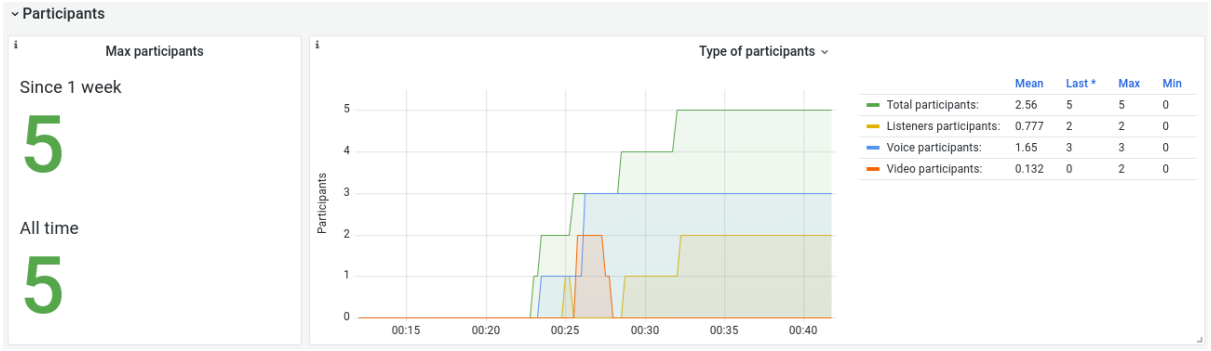
4.4: Grafana και BigBlueButton exporter

Τα παρακάτω panel, ανήκουν στην ομάδα “High level stats”, αναγράφουν τον εξυπηρετητή που φιλοξενεί το BigBlueButton, ενδεικνύουν εάν το API του είναι ενεργό, την έκδοση του BigBlueButton exporter, αλλά και τον συνολικό χρόνο που αυτός βρίσκεται σε λειτουργία.



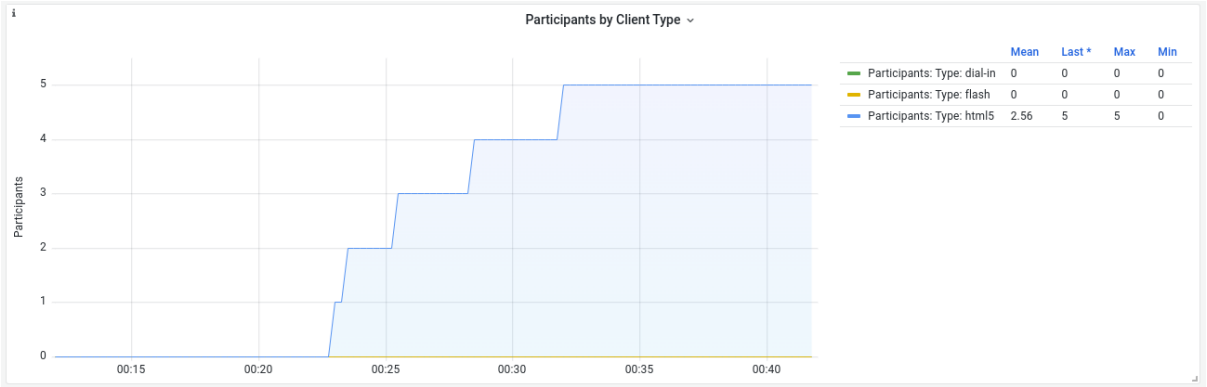
Στιγμιότυπο 4.4.1: Host και πληροφορίες του BBB exporter

Η κατηγορία “Participants” φέρει πληροφορίες για τους συμμετέχοντες σε συνεδρίες του BigBlueButton με βάση το είδος τους που μπορεί να είναι ακροατές, συμμετέχοντες σε βιντεοκλήσεις ή φωνητικοί συμμετέχοντες. Στο αριστερό panel του επόμενου στιγμιοτύπου διακρίνεται το μέγιστο πλήθος των συμμετεχόντων σε BBB κλήσεις, την τελευταία εβδομάδα αλλά και γενικότερα.



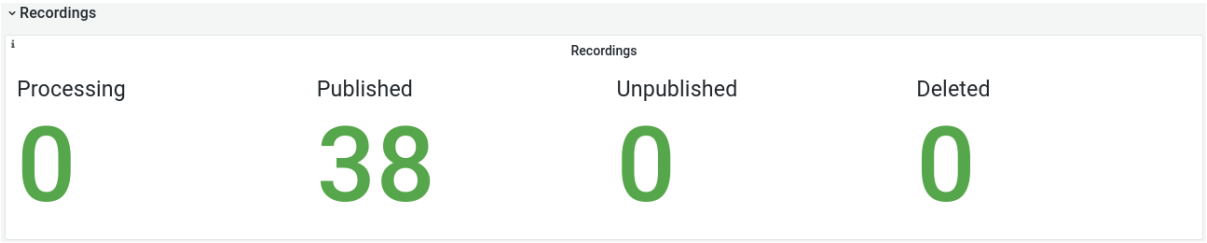
Στιγμιότυπο 4.4.2: Πλήθος και τύπος συμμετεχόντων σε τηλεδιασκέψεις

Ακόμη, το panel “Participants by client type” παρουσιάζει τον τύπο του client του web browser των συμμετεχόντων (dial-in, flash, html5).



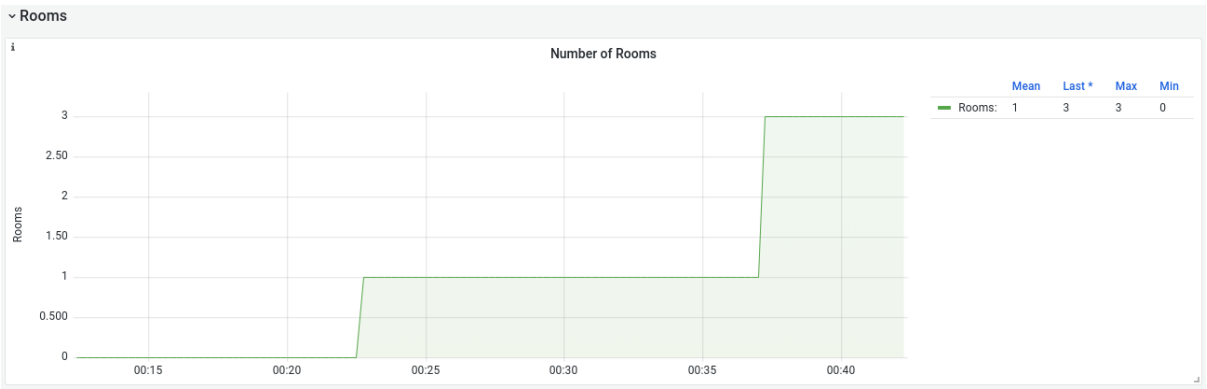
Στιγμιότυπο 4.4.3: Συμμετέχοντες σε τηλεδιασκέψεις ανά client type

Μια ακόμη κατηγορία αποτελεί η “Recordings”, panel της οποίας αναγράφουν τον αριθμό των καταγραφών (recordings) των συνεδριών που βρίσκονται σε τρέχουσα επεξεργασία, που έχουν δημοσιευθεί ή όχι, αλλά και εκείνων που έχουν διαγραφεί.
Μια καταγραφή, αποθηκεύει όλα τα συμβάντα (events) και τα δεδομένα κατά τη διάρκεια μιας συνεδρίας και αυτά τίθενται διαθέσιμα για εκ των υστέρων αναπαραγωγή.



Στιγμιότυπο 4.4.4: Τύποι καταγραφών (recordings)

Τέλος, η κατηγορία “Rooms”, περιλαμβάνει ένα panel, το οποίο απεικονίζει τον αριθμό των διαδικτυακών δωματίων που χρησιμοποιούνται. Στο πλαίσιο των δοκιμασιών, δημιουργήθηκε ένα δωμάτιο τηλεδιάσκεψης και άλλα δύο “Breakout rooms” και γι’ αυτό το λόγο το συνολικό πλήθος των δωματίων που ενδεικνύεται ισούται με 3.



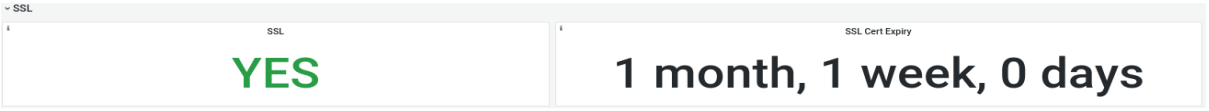
Στιγμιότυπο 4.4.5: Πλήθος δωματίων (Rooms)

4.5: Grafana και Blackbox exporter

Ομοίως και με τα προηγούμενα dashboards, τα παρακάτω panel, ανήκουν στην ομάδα “High level stats”, αναγράφουν τον HTTP(S) εξυπηρετητή για τον οποίο εξάγεται η πληροφορία, την έκδοση του IP πρωτοκόλλου και του HTTP, καθώς και το HTTP response status code.

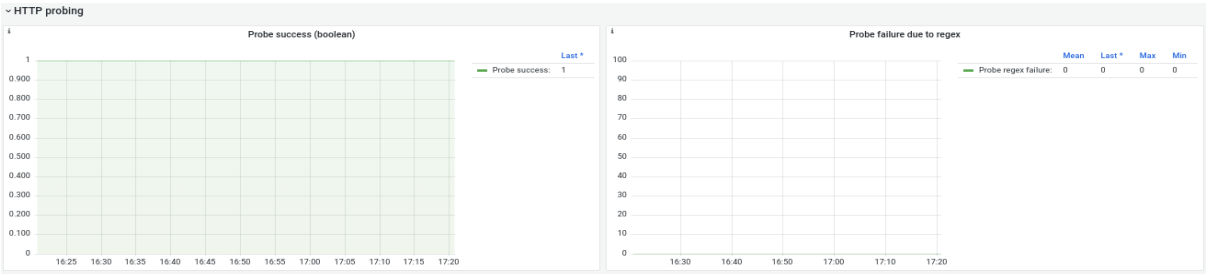
Στιγμιότυπο 4.5.1: Γενικά στοιχεία του HTTP(S) εξυπηρετητή

Η κατηγορία “SSL” ενδεικνύει εάν χρησιμοποιείται το SSL πρωτόκολλο και εάν αυτό αληθεύει, την ημερομηνία λήξης του πιστοποιητικού του.

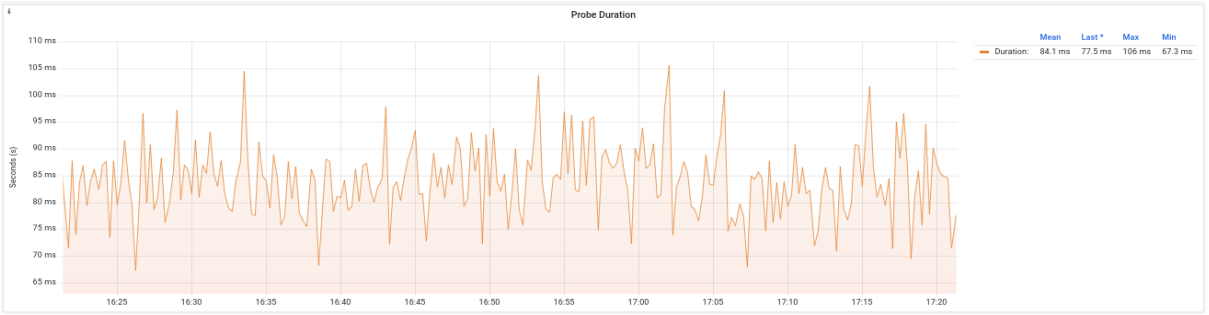


Στιγμιότυπο 4.5.2: Κατάσταση του SSL και ημερομηνία λήξης του πιστοποιητικού του

Αναφορικά με την κατηγορία “HTTP probing”, παρουσιάζονται πληροφορίες σχετικά με την επιτυχία (ή αποτυχία λόγω regex failure) του HTTP probe, καθώς και την χρονική διάρκεια που απαιτήθηκε για το “probing”.

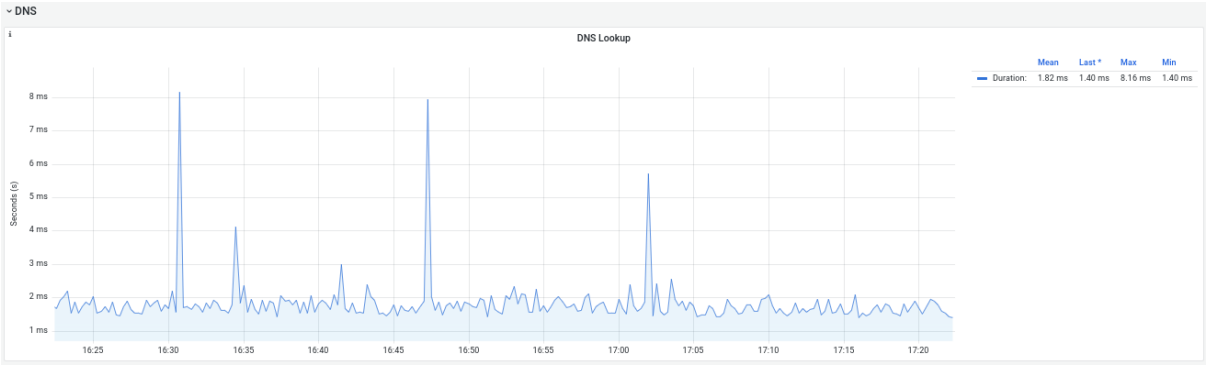


Στιγμιότυπο 4.5.3: Επιτυχία και αποτυχία (λόγω regex failure) του HTTP probe



Στιγμιότυπο 4.5.4: Χρονική διάρκεια του HTTP probe

Τέλος, στην κατηγορία “DNS” αποτυπώνονται μέσω γραφικών παραστάσεων πληροφορίες σχετικά με την υπηρεσία του DNS, όπως είναι η χρονική διάρκεια του “DNS Lookup”.

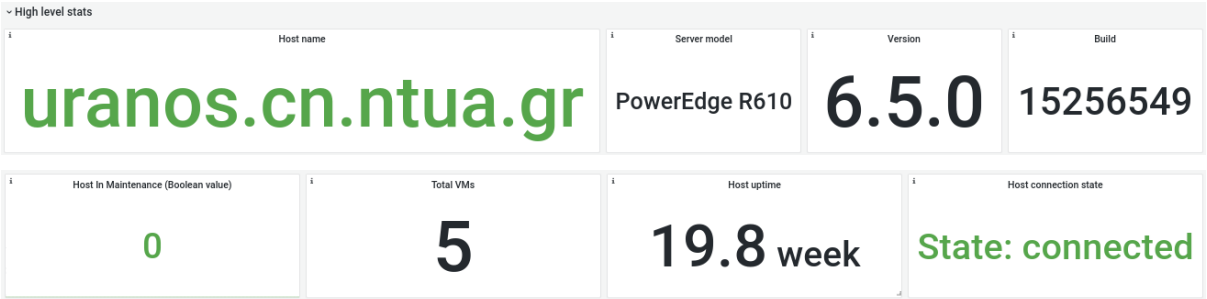


Στιγμιότυπο 4.5.5: Χρονική διάρκεια του DNS Lookup

4.6: Grafana και VMware exporter

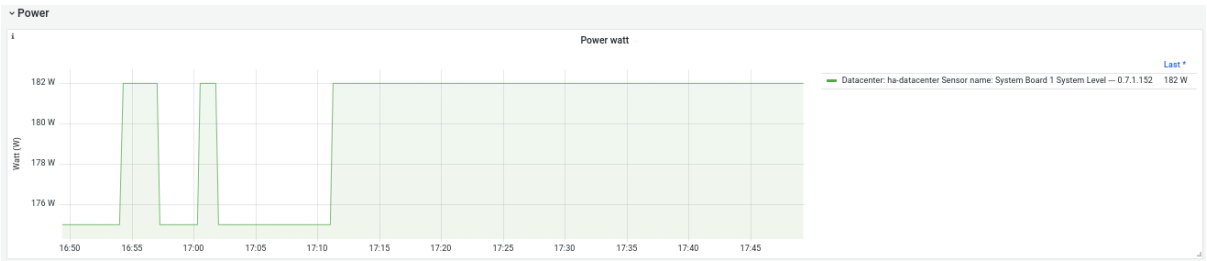
Οι μετρικές αναφορικά με τον VMware exporter χωρίζονται σε δύο κατηγορίες: εκείνες περί του host (ESXi) και σε αυτές που αφορούν στα εικονικά μηχανήματα (VMs).

Για τον λόγο αυτό, δημιουργήθηκαν δύο dashboards: το “Host” και το “VM stats”, ώστε να διακριθεί η πληροφορία και να είναι ευκολότερη η ανάλυση των μετρικών του VMware exporter.
Αρχικά, στο dashboard “Host”, στην κατηγορία “High level stats”, αποτυπώνονται στοιχεία που αφορούν στον ESXi host, όπως είναι το μοντέλο του εξυπηρετητή, η έκδοσή του και ο αριθμός build.
Ακόμη, διατίθενται και γενικές πληροφορίες, όπως είναι το συνολικό πλήθος των εικονικών μηχανημάτων που περιλαμβάνει ή η χρονική διάρκεια που βρίσκεται σε λειτουργία.

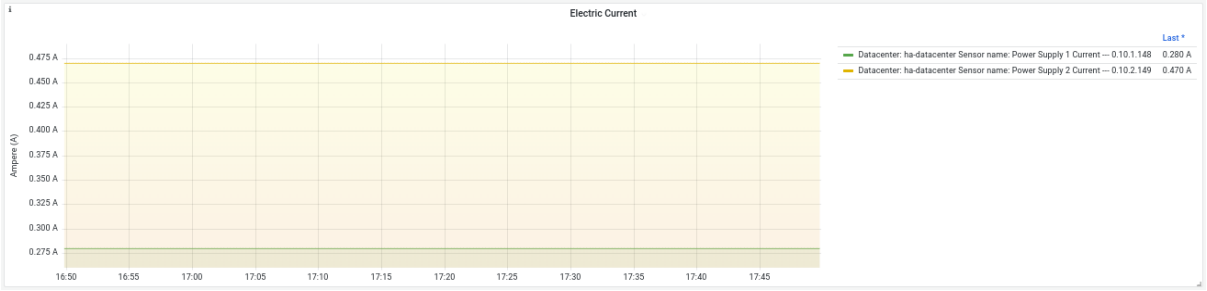


Στιγμιότυπο 4.6.1: ESXi host και γενικές πληροφορίες

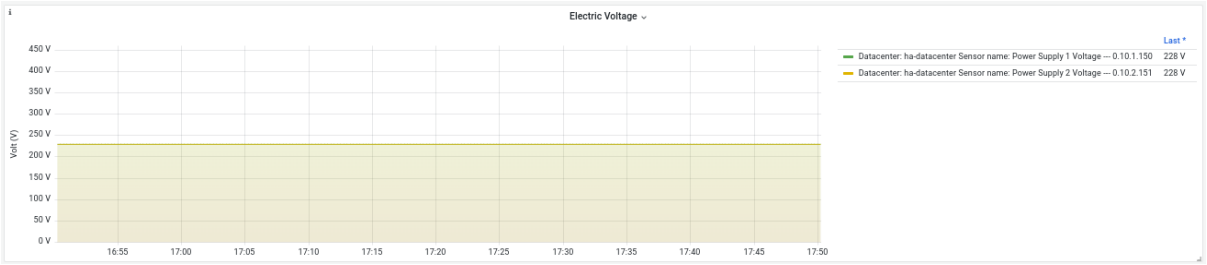
Στην κατηγορία “Power” διατίθενται δεδομένα σχετικά με την κατανάλωση του συστήματος όσον αφορά την ηλεκτρική ισχύ, την τιμή του ρεύματος σε Ampere ή της τάσης σε Volt.



Στιγμιότυπο 4.6.2: Καταναλισκόμενη ηλεκτρική ισχύς



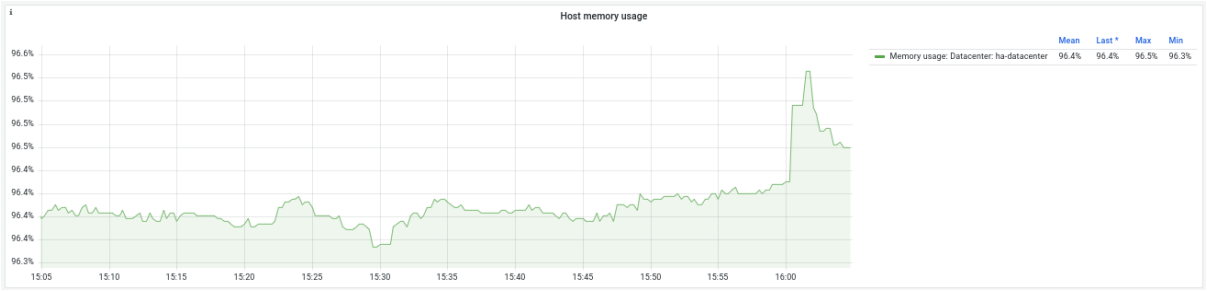
Στιγμιότυπο 4.6.3: Τιμή ηλεκτρικού ρεύματος σε A



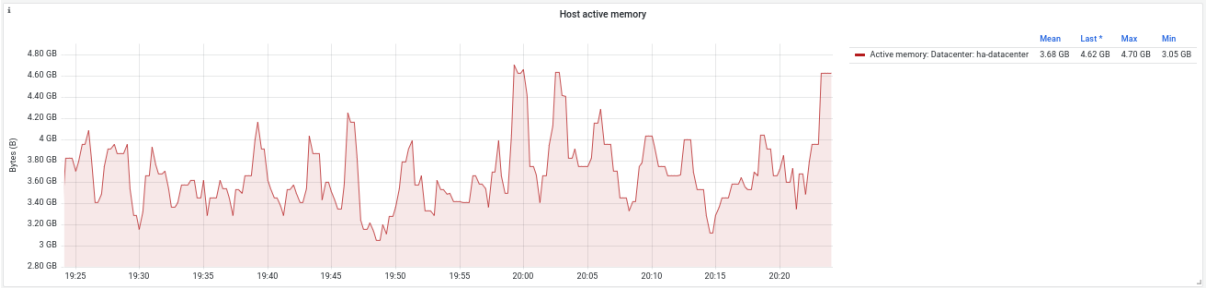
Στιγμιότυπο 4.6.4: Τιμή ηλεκτρικής τάσης σε V

Αναφορικά με την κατηγορία “Memory” παρουσιάζονται πληροφορίες σχετικά με την μνήμη του ESXi host, όπως είναι η συνολική χωρητικότητα μνήμης του συστήματος, η συνολική χρήση της μνήμης, καθώς και η ενεργή τρέχουσα μνήμη.

Στιγμιότυπο 4.6.5: Συνολική χωρητικότητα μνήμης

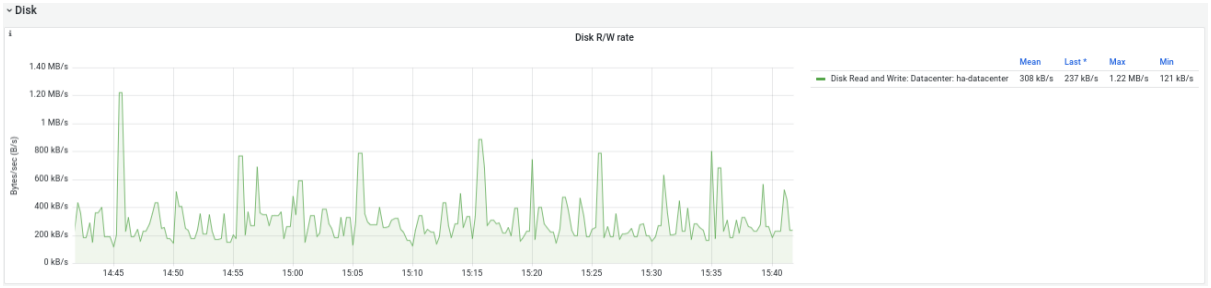


Στιγμιότυπο 4.6.6: Συνολική χρήση μνήμης σε ποσοστιαία κλίμακα



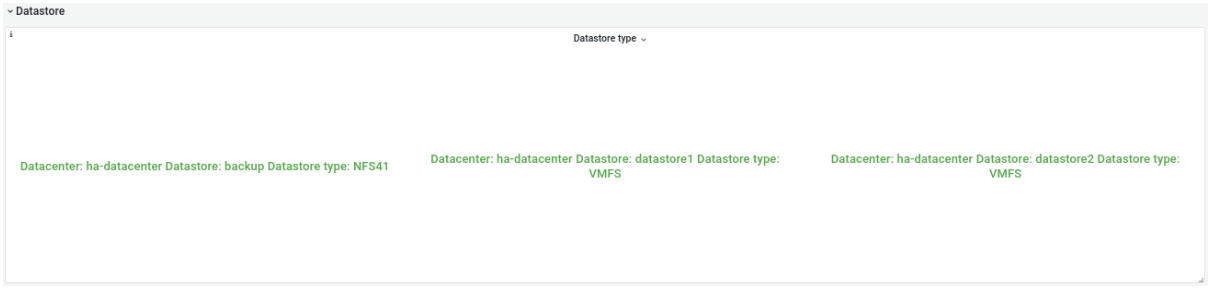
Στιγμιότυπο 4.6.7: Ενεργή μνήμη σε GB

Στην ομάδα “Disk”, αποτυπώνονται μέσω γραφικών παραστάσεων πληροφορίες σχετικά με την χρήση του δίσκου του συστήματος, όπως είναι ο ρυθμός ανάγνωσης ή εγγραφής στον δίσκο σε Bytes/sec.



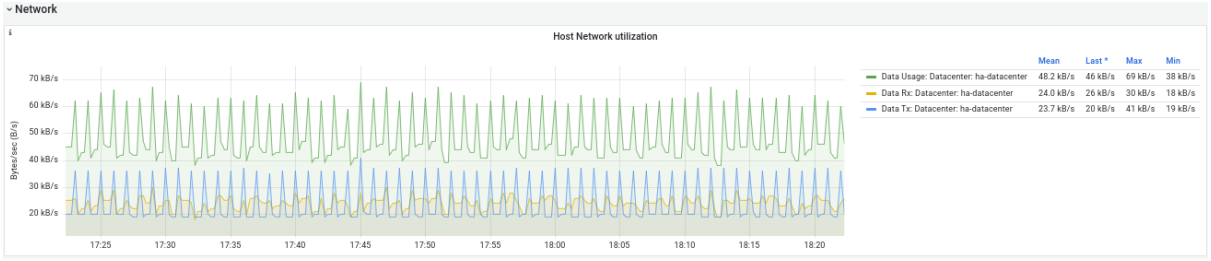
Στιγμιότυπο 4.6.8: Ρυθμός ανάγνωσης σε Bytes/sec

Επιπλέον, η ομάδα “Datastore” απεικονίζει στατιστικά περί του Datastore του host, όπως για παράδειγμα ο τύπος και το όνομα του Datastore που χρησιμοποιείται.



Στιγμιότυπο 4.6.9: Τύπος και όνομα ανά Datastore

Τέλος, η ομάδα “Network” περιλαμβάνει panel τα οποία παρουσιάζουν πληροφορίες σχετικά με την δικτυακή κίνηση, όπως είναι η δικτυακή χρήση σε Bytes/sec είτε το πλήθος των λανθασμένων δικτυακών πακέτων.

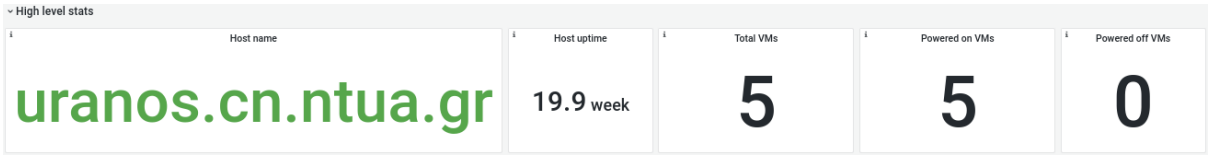


Στιγμιότυπο 4.6.10: Δικτυακή χρήση σε Bytes/sec

Στιγμιότυπο 4.6.11: Πλήθος λανθασμένων δικτυακών πακέτων

Έχοντας λάβει μια εικόνα για την οπτικοποίηση των μετρικών που αφορούν στον ESXi host του VMware, επόμενο βήμα αποτελεί η παρουσίαση των dashboards που αφορούν στα στοιχεία των εικονικών μηχανημάτων (VMs).

Αρχικά, στο dashboard “High level stats” απεικονίζονται πληροφορίες, όπως είναι το σύνολο των εικονικών μηχανημάτων ενός host που βρίσκονται σε λειτουργία ή όχι, τα ονόματα αυτών, καθώς και ο χρόνος λειτουργίας τους.

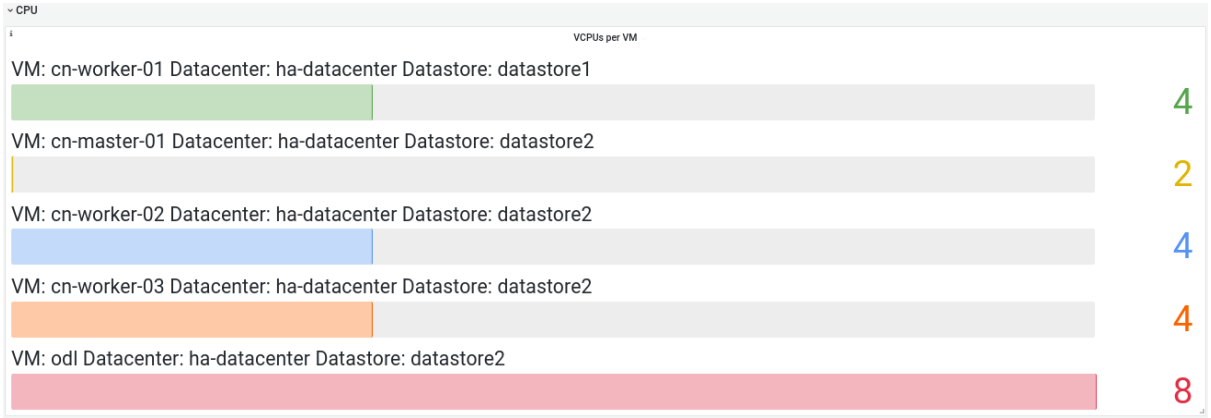


Στιγμιότυπο 4.6.12: Όνομα host, χρόνος λειτουργίας και πλήθος εικονικών μηχανημάτων του

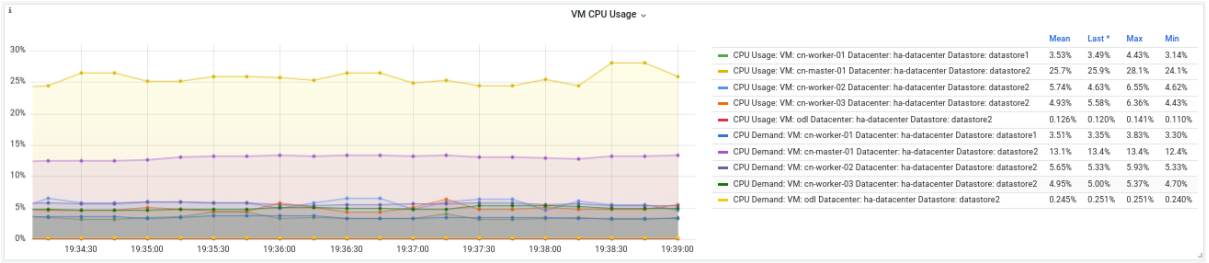
Στιγμιότυπο 4.6.13: Εικονικά μηχανήματα σε χρήση, Datastore και Datacenter που χρησιμοποιούν

Στιγμιότυπο 4.6.14: Μέσος χρόνος λειτουργίας κάθε ενεργού εικονικού μηχανήματος

Αναφορικά με την κατηγορία “CPU”, παρουσιάζονται πληροφορίες σχετικά με την εικονική CPU κάθε εικονικού μηχανήματος, όπως είναι το πλήθος των πυρήνων, αλλά και η μέση χρήση της επεξεργαστικής ισχύος.

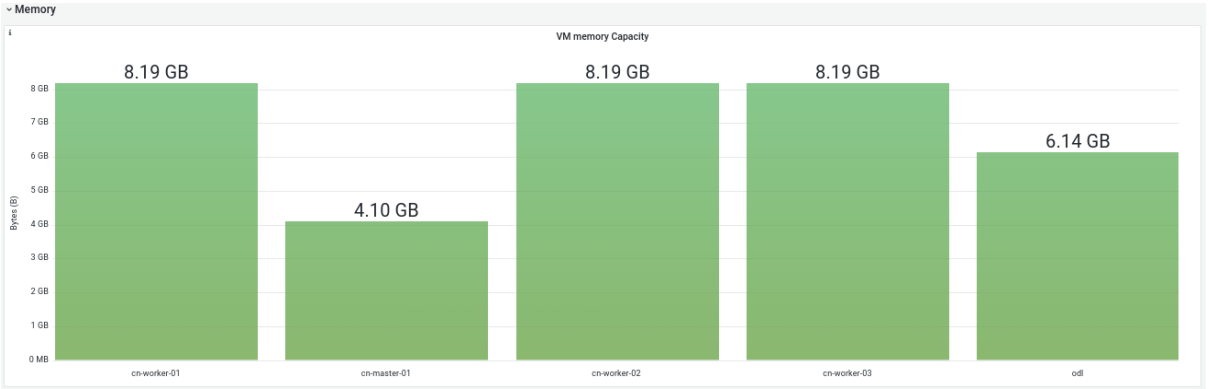


Στιγμιότυπο 4.6.15: Πλήθος εικονικών CPUs ανά εικονικό μηχάνημα

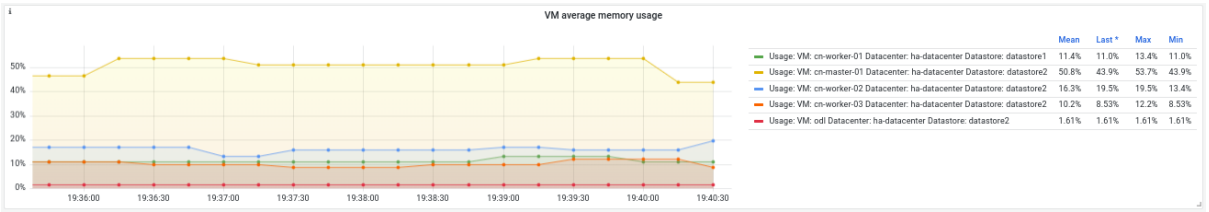


Στιγμιότυπο 4.6.16: Μέση ποσοστιαία χρήση της επεξεργαστικής ισχύος ανά εικονικό μηχάνημα

Στην κατηγορία “Memory”, ενδεικνύονται στοιχεία που αφορούν στην μνήμη των εικονικών μηχανημάτων, όπως είναι το πλήθος της συνολικής χωρητικότητας μνήμης, είτε η μέση ποσοστιαία της χρήση ανά εικονικό μηχάνημα.

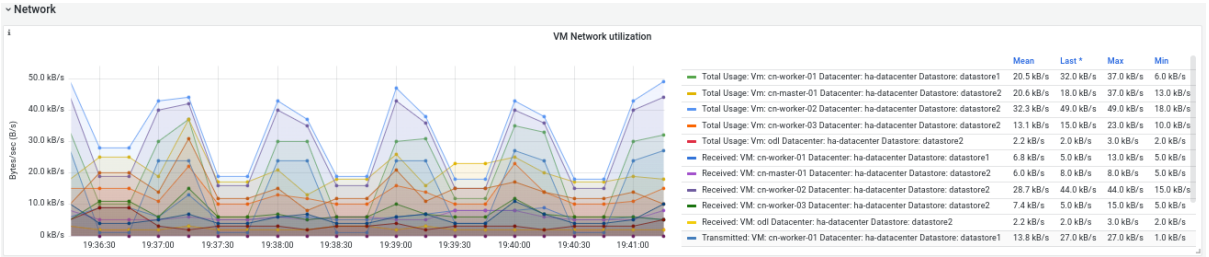


Στιγμιότυπο 4.6.17: Συνολική χωρητικότητα μνήμης ανά ενεργό εικονικό μηχάνημα



Στιγμιότυπο 4.6.18: Μέση χρήση μνήμης ανά ενεργό εικονικό μηχάνημα

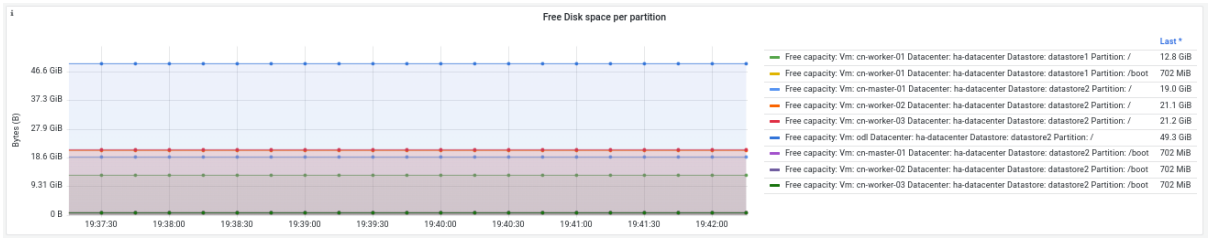
Μια ακόμη κατηγορία αποτελεί η “Network”, στην οποία απεικονίζονται μέσω γραφικών παραστάσεων στατιστικά που αφορούν στην δικτυακή κίνηση κάθε εικονικού μηχανήματος, όπως είναι η δικτυακή κίνηση με βάση την προέλευση (transmitted ή received).



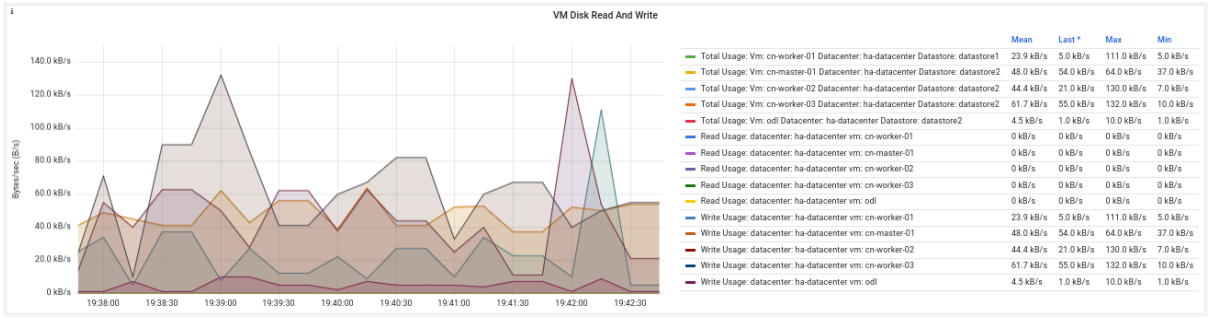
Στιγμιότυπο 4.6.19: Δικτυακή χρήση ανά ενεργό εικονικό μηχάνημα

Τέλος, στην κατηγορία “Disk”, αποτυπώνονται στατιστικά περί της χρήσης του δίσκου ανά εικονικό μηχάνημα. Όπως διακρίνεται και από τα επόμενα στιγμιότυπα, διατίθενται panel που αφορούν στη συνολική χωρητικότητα αποθηκευτικού χώρου ανά εικονικό μηχάνημα, τον υπολειπόμενο ελεύθερο χώρο, αλλά και τον ρυθμό ανάγνωσης ή εγγραφής σε αυτόν.

Στιγμιότυπο 4.6.20: Συνολική χωρητικότητα δίσκου ανά partition και ενεργό εικονικό μηχάνημα



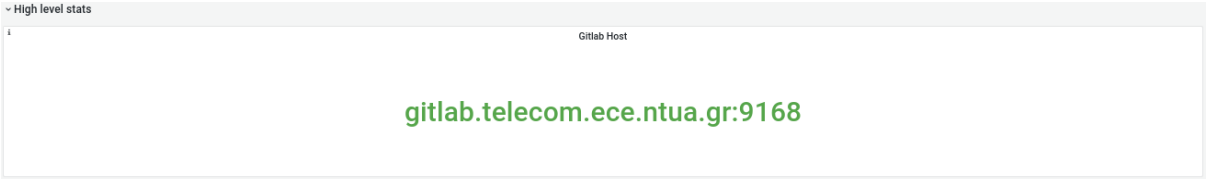
Στιγμιότυπο 4.6.21: Χωρητικότητα ελεύθερου δίσκου ανά partition και ενεργό εικονικό μηχάνημα



Στιγμιότυπο 4.6.22: Ρυθμός ανάγνωσης και εγγραφής δίσκου ανά ενεργό εικονικό μηχάνημα

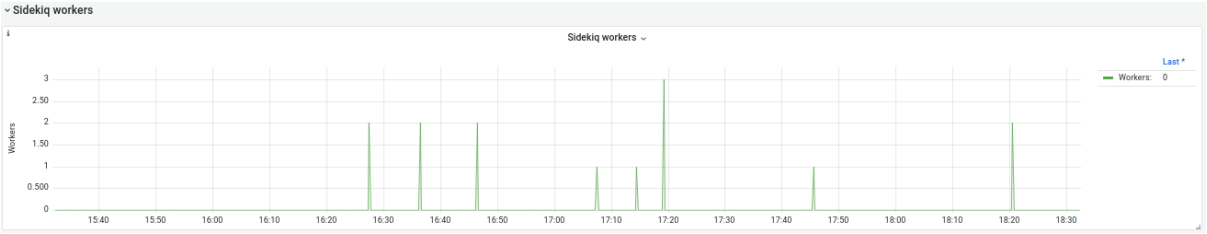
4.7: Grafana και Gitlab exporter

Ομοίως και με τα προηγούμενα dashboards, στο dashboard που αφορά στον Gitlab exporter και συγκεκριμένα στην ομάδα “high level stats”, αναδεικνύεται ο host στον οποίο φιλοξενείται η υπηρεσία του Gitlab.

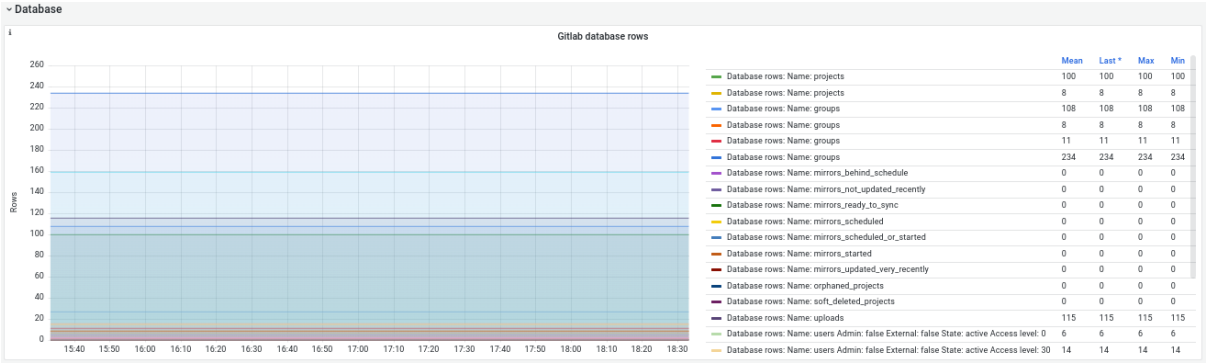


Στιγμιότυπο 4.7.1: Gitlab host

Εκτός τούτων, διατίθενται διάφορα dashboards που αφορούν στο “sidekiq” του Gitlab, αλλά και στο “Gitlab Database rows”.



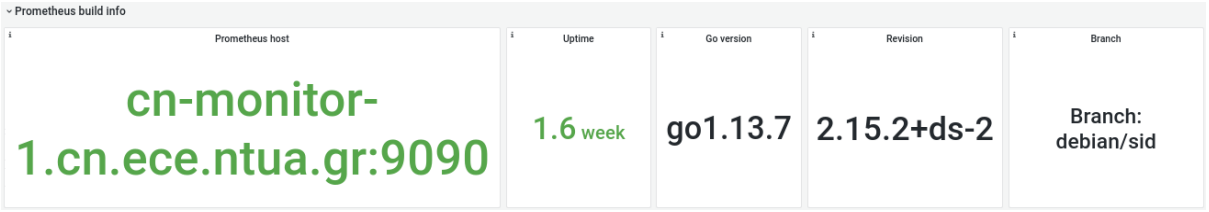
Στιγμιότυπο 4.7.2: Πλήθος sidekiq workers



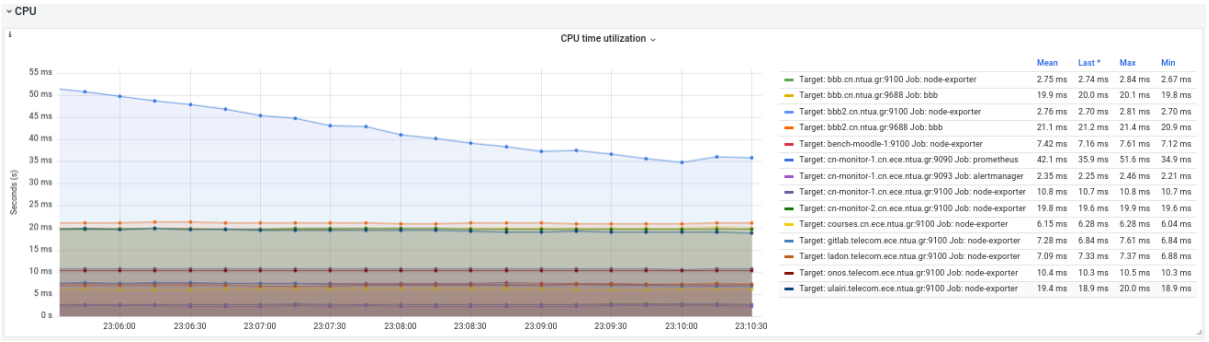
Στιγμιότυπο 4.7.3: Gitlab Database rows

4.8: Grafana και Prometheus

Εκτός των exporters, δύναται κανείς να παρακολουθήσει μέσω του Grafana, το ίδιο το Prometheus και ορισμένα χαρακτηριστικά του, όπως είναι η χρονική χρήση της CPU ανά target, η χρονική διάρκεια κάθε Scrape, πληροφορίες σχετικά με την βάση δεδομένων, τη μνήμη κ.α.



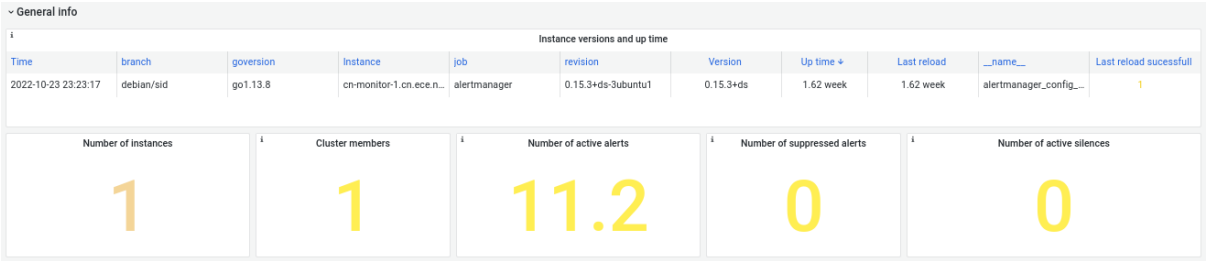
Στιγμιότυπο 4.8.1: Γενικά στοιχεία του Prometheus



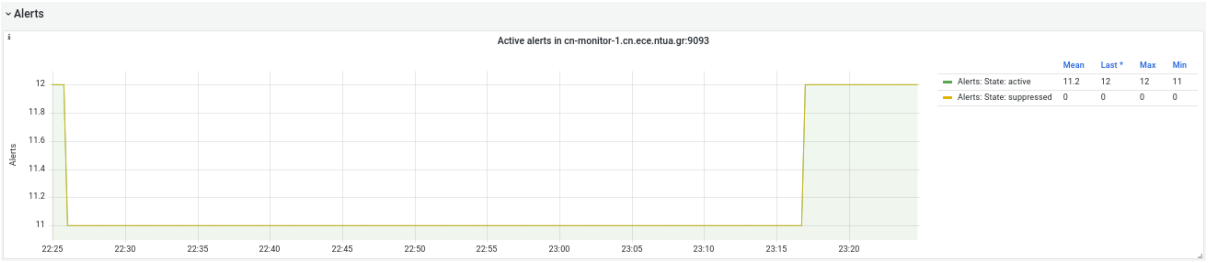
Στιγμιότυπο 4.8.2: Χρονική χρήση της CPU του Prometheus ανά “job”

4.9: Grafana και Prometheus Alertmanager

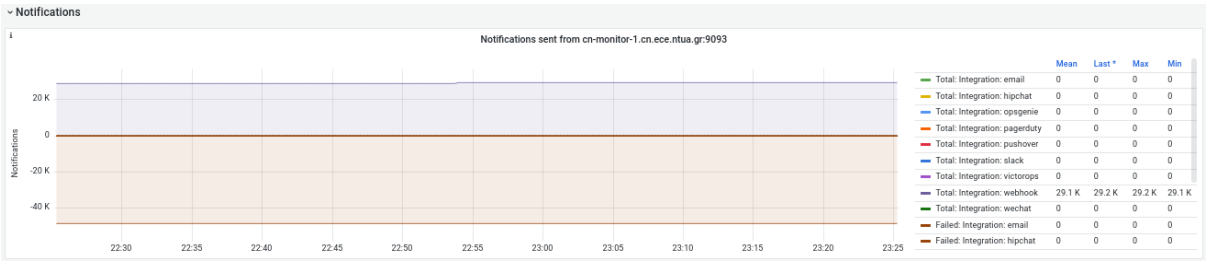
Ένα ακόμη dashboard που έχει δημιουργηθεί αφορά στον Prometheus Alertmanager. Σε αυτό, αποτυπώνονται γενικά στοιχεία του, όπως είναι ο αριθμός των ενεργών ή σιγασμένων ειδοποιητικών συναγερμών, το πλήθος των ειδοποιήσεων ανά κατηγορία, καθώς και διάφορα στοιχεία που αφορούν στη λειτουργία του.



Στιγμιότυπο 4.9.1: Γενικά στοιχεία του Prometheus Alertmanager



Στιγμιότυπο 4.9.2: Στατιστικά περί ειδοποιητικών συναγερμών



Στιγμιότυπο 4.9.3: Πλήθος ειδοποιητικών συναγερμών ανά τύπο

Κεφάλαιο 5: Οπτικοποίηση και αρχεία καταγραφής

5.1: Elasticsearch

Αναφορικά με την ανάλυση δεδομένων, χρησιμοποιήθηκε στα πλαίσια της παρούσης διπλωματικής εργασίας η σουίτα λογισμικού Elasticsearch [7]. Ειδικότερα, περιλαμβάνει συγκεκριμένα εργαλεία για την συλλογή, την εικονική αναπαράσταση αλλά και μεταφορά των δεδομένων. Η χρήση του Elasticsearch είναι ιδιαίτερα σημαντική καθώς επιτρέπει στους διαχειριστές να έχουν όλα τα αρχεία καταγραφής των διαφόρων συστημάτων σε ένα κεντρικό σημείο. Η αρχιτεκτονική του Elasticsearch επιτρέπει την αποδοτική διαχείριση μεγάλου πλήθους αρχείων καταγραφής σε πραγματικό χρόνο. Παράλληλα, μπορεί να δώσει τη δυνατότητα ανάπτυξης αλγορίθμων για τον αυτόματο εντοπισμό ανωμαλιών στη λειτουργία των συστημάτων καθώς και περιστατικών ασφαλείας.

Στιγμιότυπο 5.1: Λογότυπο Elasticsearch

Η επιλογή του Elasticsearch ως εργαλείου, πραγματοποιήθηκε χάριν της ευελιξίας που προσφέρει. Από τη μία πλευρά, πρόκειται για ένα λογισμικό ανοιχτού κώδικα, το οποίο δίνει έμφαση στην ταχύτητα, αλλά και στην ευελιξία των επιλογών και εργαλείων διαφόρων εφαρμογών που προσφέρει.
Από την άλλη πλευρά, η οπτικοποίηση, αλλά και η ευκολία ανάλυσης των δεδομένων είναι εκείνη που το καθιστά μια από τις πιο ευρέως χρησιμοποιούμενες πλατφόρμες αυτού του είδους.

Ορισμένες από τις κύριες χρήσεις του Elasticsearch διακρίνονται στην παρακάτω λίστα:

- Application, Website και Enterprise search
- Logging και log analytics
- Παρακολούθηση από μετρικές και container
- Παρακολούθηση επίδοσης Application
- Γεωχωρική ανάλυση και οπτικοποίηση δεδομένων
- Εφαρμογές analytics για εφαρμογές Cyber Security
- Εφαρμογές Business analytics

Με βάση τα παραπάνω, για την διπλωματική εργασία, η σουίτα του Elasticsearch συνεισφέρει στην διαδικασία του Logging και των log analytics.

Εφόσον έχουν συλλεχθεί τα ενδιαφέροντα “unstructured data”, επόμενο στάδιο αποτελεί η διαδικασία, κατά την οποία τα δεδομένα του Elasticsearch επιδέχονται γραμματική ανάλυση (parsing), κανονικοποίηση (normalization), αλλά και εμπλουτισμό (enrichment), η οποία ονομάζεται Data ingestion.

Στη συνέχεια, η πληροφορία επιδέχεται “indexing”, το οποίο είναι το χαρακτηριστικό που προσφέρει τη δυνατότητα διαχείρισης και επεξεργασίας των δεδομένων, μιας και τα δεδομένα είναι πλέον είναι οργανωμένα.
Αυτό, επιτυγχάνεται προσδίδοντας σε κάθε δεδομένο ένα σετ από από “keys”, το οποίο δεν είναι παρά μια αντιστοιχία ενός ονόματος ή κάποιας ιδιότητας με τις αντίστοιχες τιμές των δεδομένων που μπορεί να είναι αριθμοί, Boolean αριθμοί, ημερομηνίες, πίνακες από τιμές, γεωγραφικές τοποθεσίες ή άλλου τύπου πληροφορίες.

Ο ρόλος του Elasticsearch αποτελεί η διαδικασία του “indexing”, αλλά και η αποθήκευση των παραπάνω δεδομένων, καθώς πρόκειται για μία “document oriented” βάση δεδομένων.

5.2: Logstash

Το λογισμικό Logstash, είναι εκείνο που συγκεντρώνει και επεξεργάζεται τα δεδομένα προτού τα διοχετεύσει στο Elasticsearch [39].

Στιγμιότυπο 5.2: Λογότυπο Logstash

Πιο συγκεκριμένα, πρόκειται για ένα “data processing pipeline”, το οποίο επιτρέπει τη δυνατότητα ταυτόχρονης συλλογής δεδομένων από διαφορετικές πηγές. Τα δεδομένα αυτά, μπορεί να είναι οποιουδήποτε τύπου, όπως μετρικές, αρχεία καταγραφής, δεδομένα από εφαρμογές ιστού κ.α. Οι πιο διαδεδομένοι μέθοδοι συλλογής δεδομένων είναι μέσω του παραδοσιακού “Syslog” και του νεότερου “Filebeat” που εξασφαλίζει την αξιόπιστη παράδοση των αρχείων καταγραφής.

Στη συνέχεια, η επεξεργασία αφορά στην διαδικασία του “Data ingestion”, δηλαδή της γραμματικής ανάλυσης (parsing), κανονικοποίησης (normalization), αλλά και του εμπλουτισμού (enrichment) των δεδομένων. Με αυτό τον τρόπο, τα δεδομένα είναι πλέον έτοιμα να αποσταλούν στο Elasticsearch.

5.3: Filebeat

Αναφορικά με τη συλλογή αρχείων καταγραφής, γίνεται χρήση του λογισμικού Filebeat [40].

Στιγμιότυπο 5.3: Λογότυπο Filebeat και της σουίτας Beats

Πρόκειται για ένα "data shipper", που ανήκει στη σουίτα Beats [41], το οποίο διοχετεύει τα δεδομένα που έχει συγκεντρώσει, σε μια βάση δεδομένων, όπως είναι το Elasticsearch.

Το πλεονέκτημα του Filebeat, έγκειται στο γεγονός, ότι μπορεί να εγκατασταθεί σε οποιονδήποτε εξυπηρετητή και μέσω παραμετροποιήσεων δύναται να προσφέρει τη δυνατότητα παρακολούθησης συγκεκριμένων αρχείων που ο διαχειριστής επιθυμεί.
Συμπερασματικά, το Filebeat συλλέγει δεδομένα τύπου αρχείων καταγραφής από τα αρχεία που έχει προσδιορίσει ο διαχειριστής ως σημαντικά προς ανάλυση και στη συνέχεια τα αποστέλλει στη μονάδα επεξεργασίας των δεδομένων αυτών, προτού αποσταλούν σε μια βάση δεδομένων. Η μονάδα αυτή στα πλαίσια της διπλωματικής εργασίας είναι το Logstash ενώ η βάση δεδομένων, το Elasticsearch.

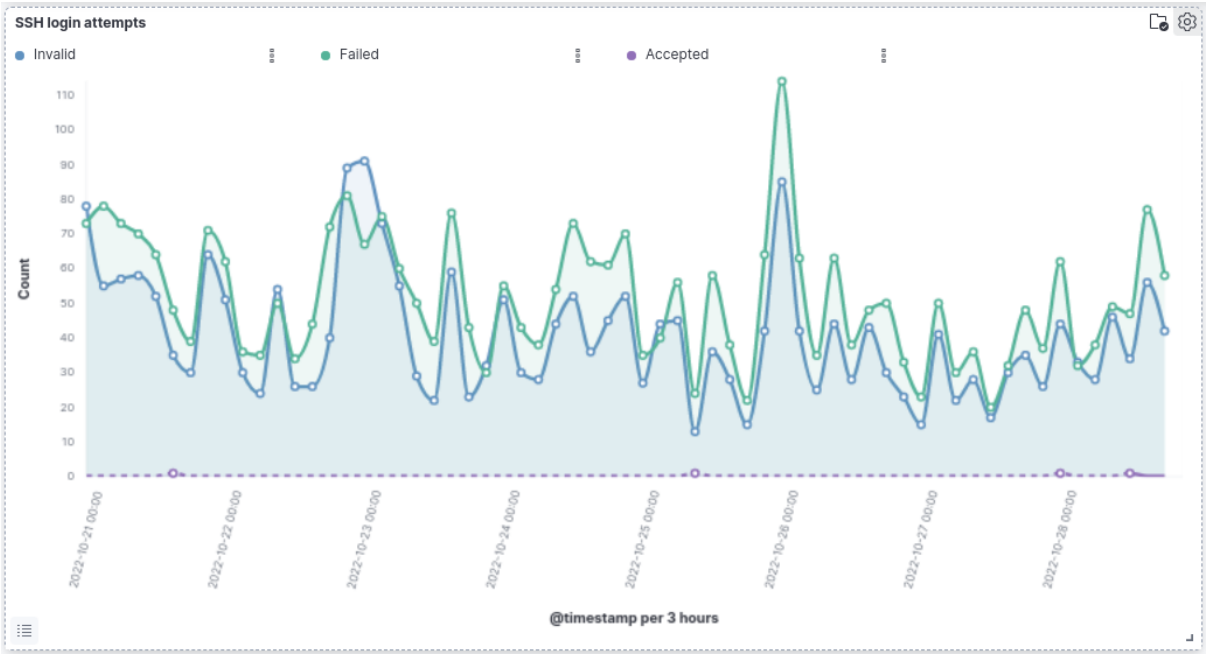
5.4: Kibana

Όπως προαναφέρθηκε και στο κεφάλαιο περί Elasticsearch, ύστερα από την διαδικασία του “indexing”, τα δεδομένα είναι πλέον διαθέσιμα για οποιαδήποτε επεξεργασία.

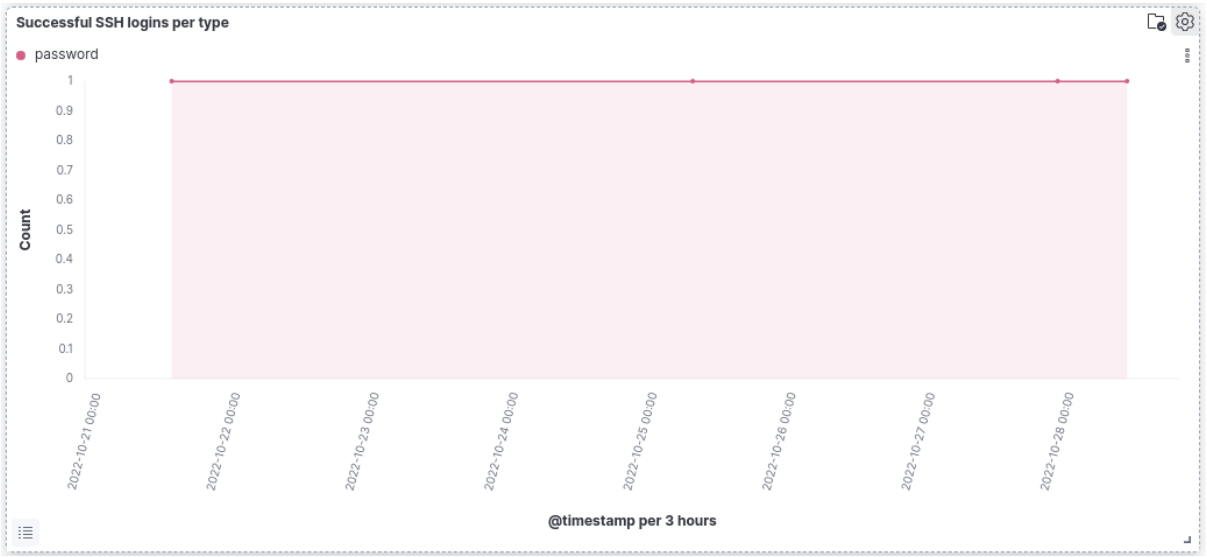
Στιγμιότυπο 5.4.1: Λογότυπο Kibana

Το Kibana, αποτελεί το τελευταίο στάδιο στην διαχείριση των δεδομένων, καθότι παρέχει τη δυνατότητα αναζήτησης (searching), αλλά και οπτικοποίησης αυτών μέσω μιας γραφικής διεπαφής, το οποίο εξυπηρετεί τη διαδικασία της παρακολούθησης και του Querying των δεδομένων [42].

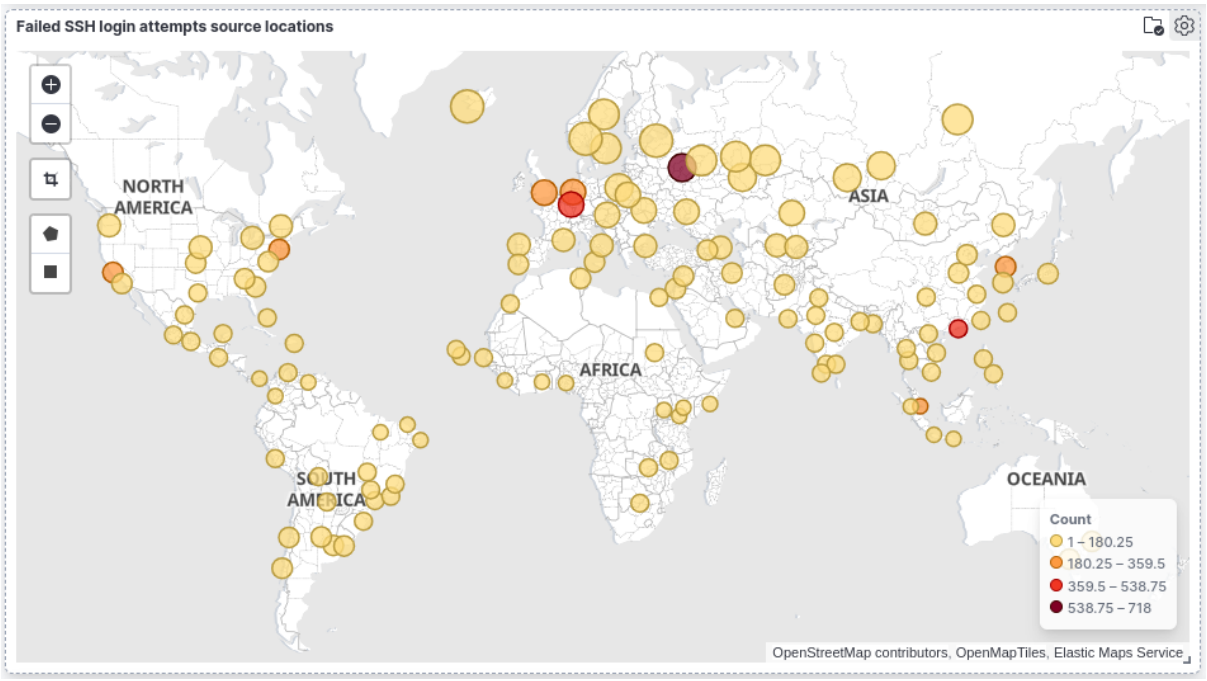
Ακόμη, διατίθενται διάφορα dashboards για κάθε χρήση, το οποίο διευκολύνει ιδιαίτερα την διαδικασία της ανάλυσης της πληροφορίας. Τα dashboards δύναται να είναι γραφήματα, χάρτες, ιστογράμματα και άλλα, όπως ακριβώς συμβαίνει και με τα dashboard του Grafana. Στις επόμενες εικόνες, απεικονίζονται ορισμένα panel από τα διάφορα dashboards. Συγκεκριμένα στο dashboard “SSH login”, διακρίνονται πληροφορίες σχετικά με τις προσπάθειες σύνδεσης μέσω SSH (SSH login attempts) και τα χαρακτηριστικά τους, το πλήθος των επιτυχημένων SSH συνόδων ανά τύπο, αλλά και τις γεωγραφικές τοποθεσίες των αποτυχημένων SSH συνόδων.



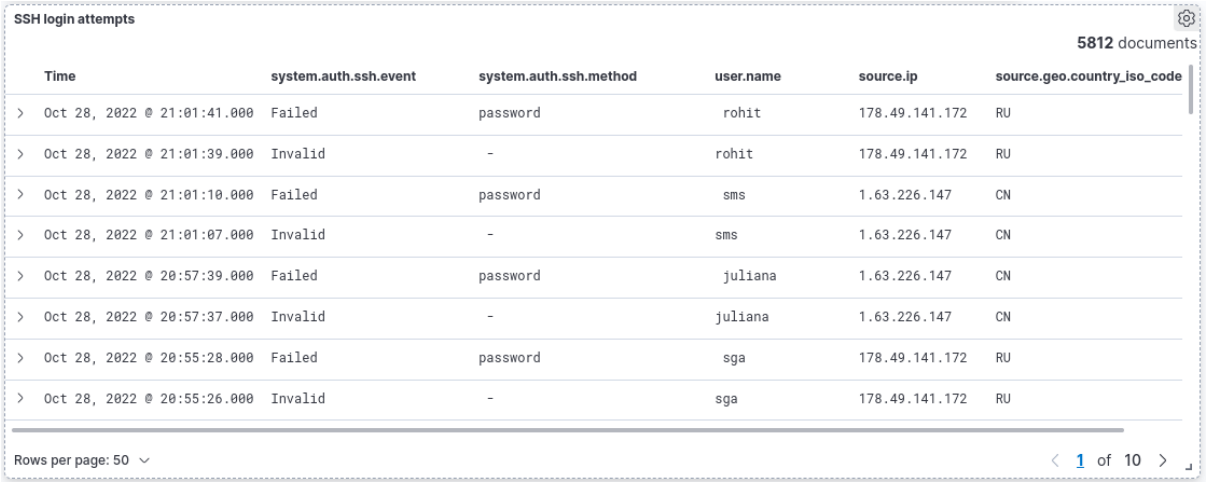
Στιγμιότυπο 5.4.2: Στατιστικά προσπαθειών SSH συνόδων



Στιγμιότυπο 5.4.3: Πλήθος επιτυχημένων SSH συνόδων ανά τύπο

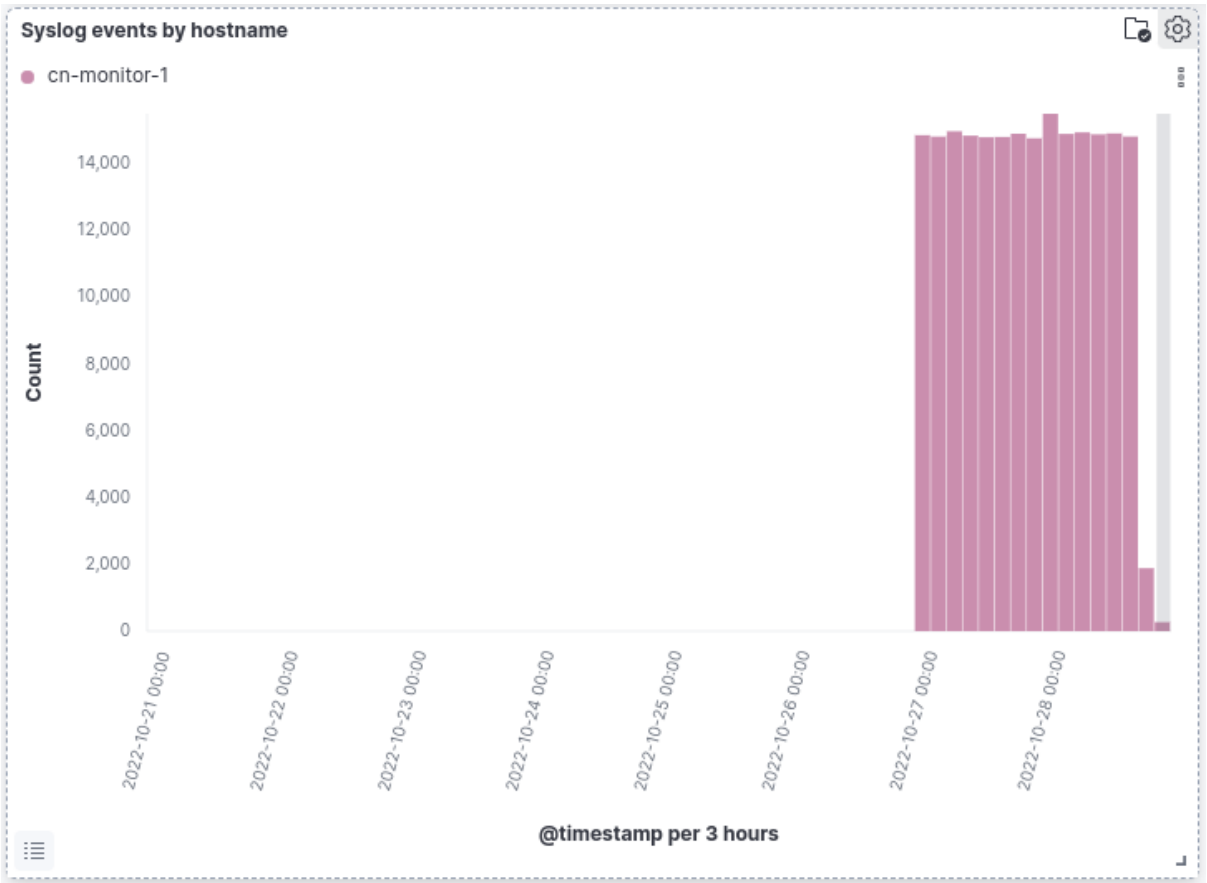


Στιγμιότυπο 5.4.4: Γεωγραφικές τοποθεσίες αποτυχημένων SSH προσπαθειών σύνδεσης

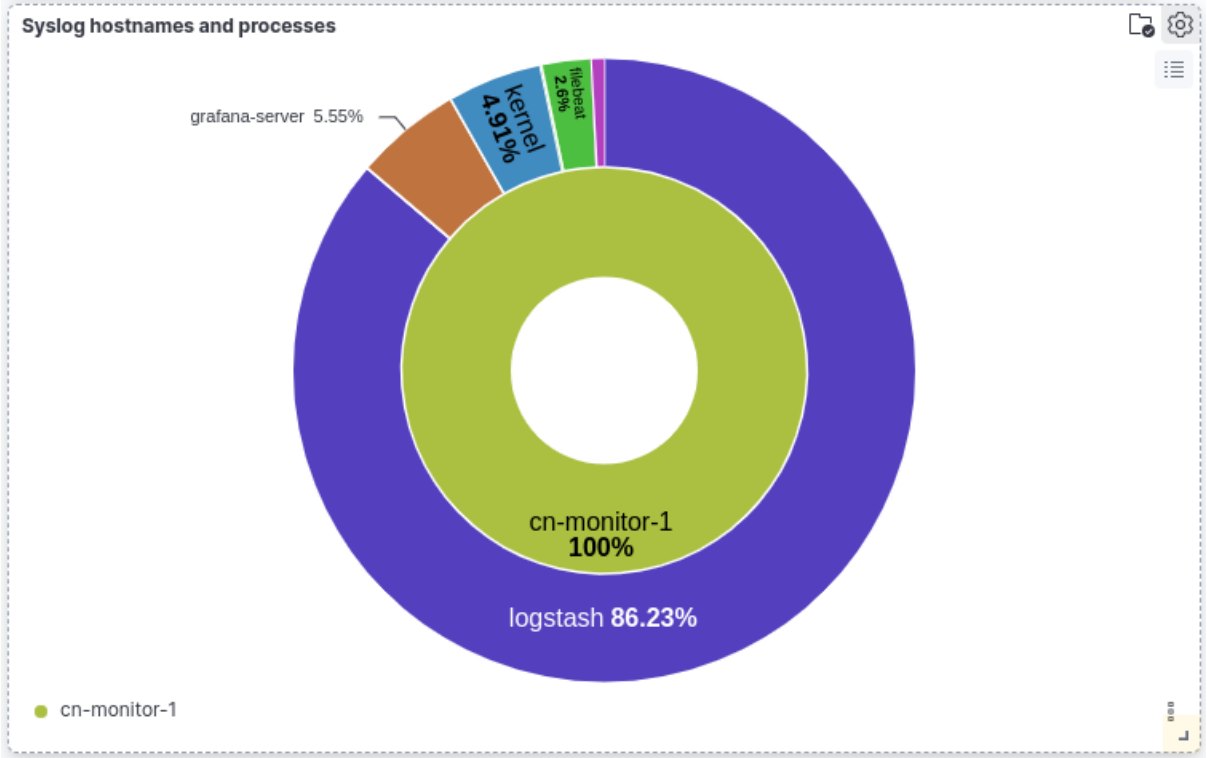


Στιγμιότυπο 5.4.5: Χαρακτηριστικά προσπαθειών SSH σύνδεσης

Ακόμη, έχει δημιουργηθεί ένα dashboard ονόματι “Syslog”, το οποίο παρουσιάζει διάφορα panel αναφορικά με τα αρχεία καταγραφής του Syslog, όπως είναι τα συμβάντα του Syslog ανά hostname, ανά διεργασία, αλλά και οποιοδήποτε αρχείο καταγραφής του Syslog και τα χαρακτηριστικά του.



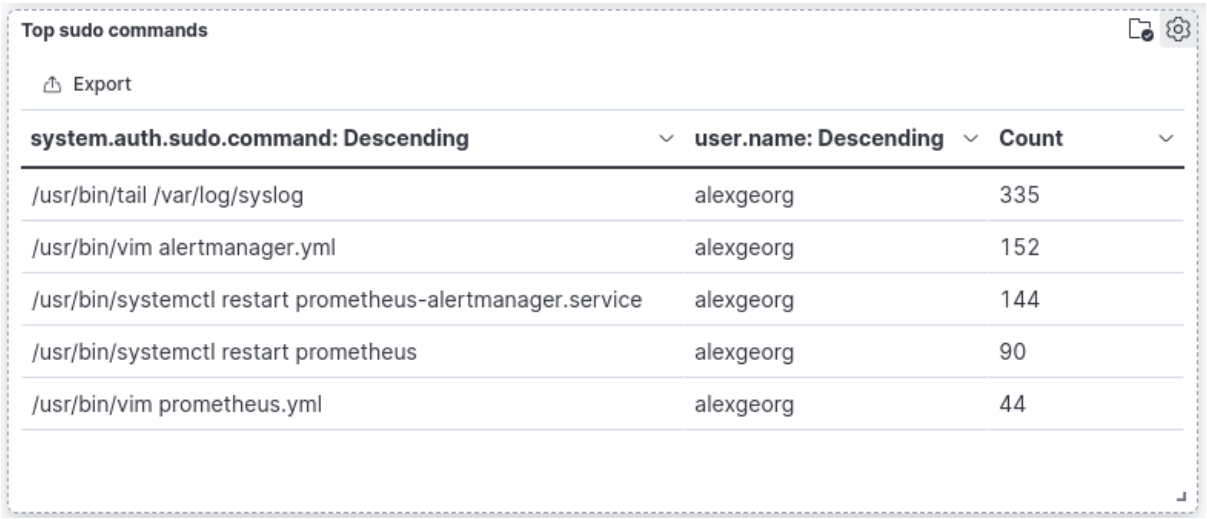
Στιγμιότυπο 5.4.6: Συμβάντα του Syslog ανά hostname



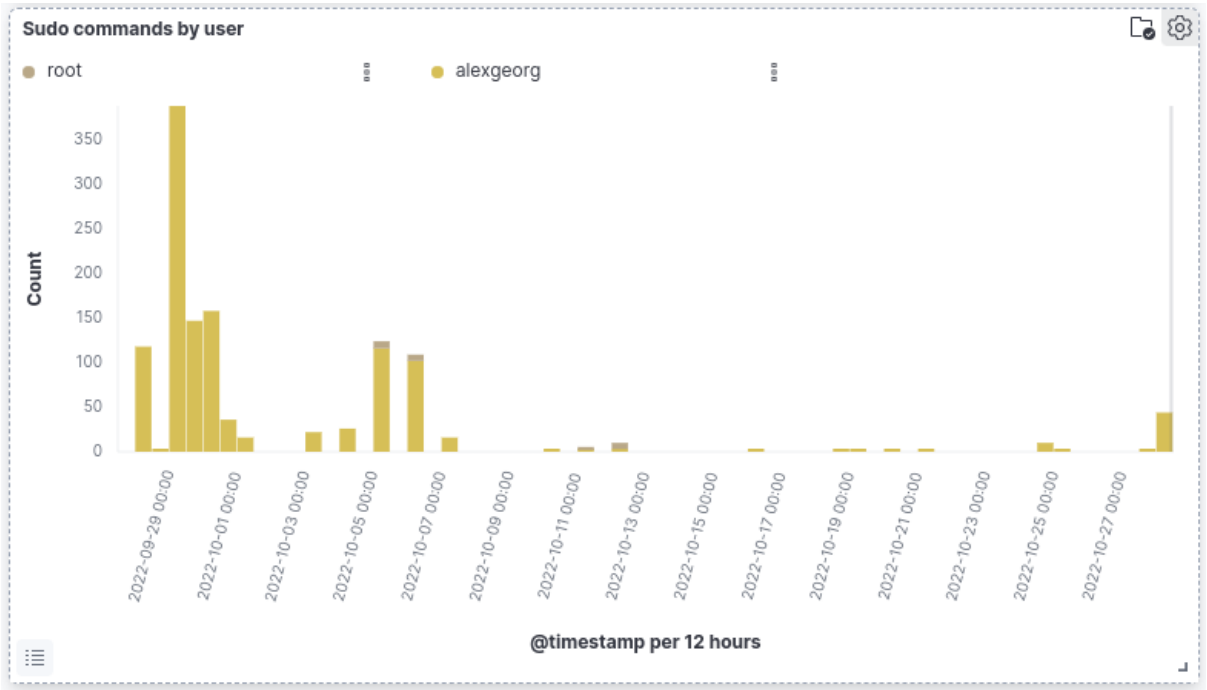
Στιγμιότυπο 5.4.7: Hostname και διεργασίες του Syslog

Στιγμιότυπο 5.4.8: Αρχεία καταγραφής του Syslog

Τέλος, ένα ακόμη dashboard αποτελεί το “Sudo commands”, στο οποίο παρουσιάζονται πληροφορίες σχετικά με τις εντολές “sudo”.



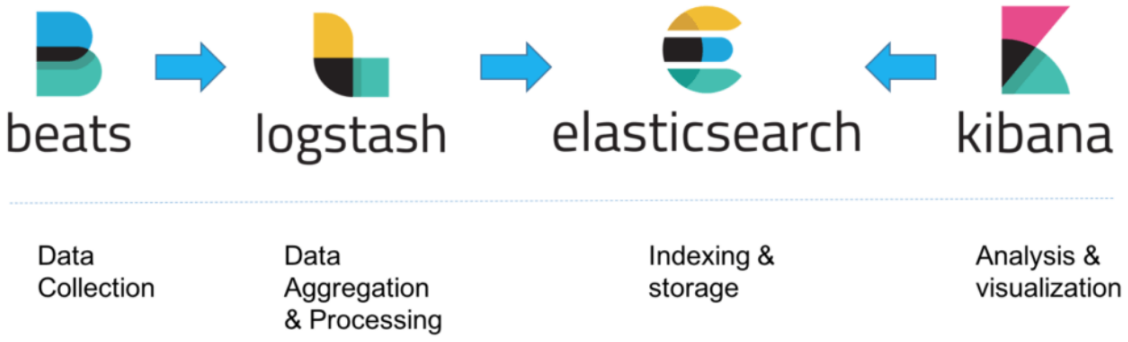
Στιγμιότυπο 5.4.9: Δημοφιλέστερες "sudo" εντολές



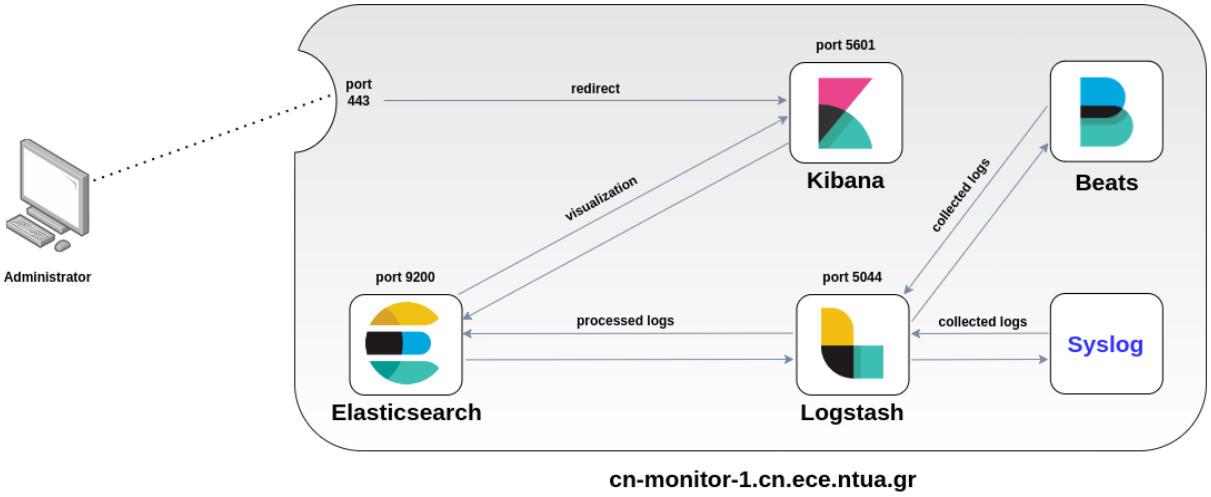
Στιγμιότυπο 5.4.10: Δημοφιλέστερες "sudo" εντολές ανά χρήστη

Παρατηρούμε, ότι το Kibana είναι ένα ιδιαίτερα ισχυρό εργαλείο για την ανάλυση των αρχείων καταγραφής, καθότι με τους ποικίλους τρόπους απεικόνισης τους, εξασφαλίζεται η βέλτιστη ανάλυσης της πληροφορίας.

Συμπερασματικά, επισυνάπτονται οι παρακάτω σχηματικές αναπαραστάσεις, οι οποίες παρουσιάζουν, αφενός τον τρόπο με τον οποίο λειτουργεί η ροή της πληροφορίας για την οπτικοποίηση της μέσω του Kibana (Στιγμιότυπο 5.4.11) και αφετέρου τη λειτουργία του reverse proxy για την χρήση του Kibana από τον διαχειριστή (Στιγμιότυπο 5.4.12).



Στιγμιότυπο 5.4.11: Διαδικασία επεξεργασίας πληροφορίας στη σουίτα ELK και Beats [43]



Στιγμιότυπο 5.4.12: Reverse proxy και διαδικασία ροής δεδομένων προς εικονοποίηση μέσω Kibana

Όπως επιβεβαιώνεται και από το παραπάνω στιγμιότυπο, το Kibana είναι διαθέσιμο μέσω reverse proxy, στη θύρα “5601”.

Μέρος 2ο

Κεφάλαιο 6: Εγκατάσταση λογισμικών-παραμετροποιήσεις

Σημείωση: Οποιαδήποτε εγκατάσταση λογισμικού πραγματοποιείται για τους σκοπούς της διπλωματικής εργασίας, επιτελείται σε περιβάλλον Ubuntu.

6.1: Εγκατάσταση των Prometheus και Node exporter

Για να εγκατασταθεί ο Prometheus, κατεβάζουμε το κατάλληλο apt-package και το με την εντολή:

```
sudo apt install prometheus
```

Ομοίως, να εγκατασταθεί ο Node exporter, κατεβάζουμε το κατάλληλο apt-package και το με την εντολή:

```
sudo apt install prometheus-node-exporter
```

Υπάρχει και η δυνατότητα να εγκατασταθεί μέσω **wget**, ωστόσο επειδή σε αυτή τη περίπτωση θα πρέπει χειροκίνητα να εγκατασταθούν νέες αναβαθμίσεις (updates) μελλοντικά, επιλέγουμε τον τρόπο που υποδείχθηκε προηγουμένως, ο οποίος λύνει αυτό το πρόβλημα ευκολότερα και τυχόν αναβαθμίσεις μπορούν να γίνουν αυτόματα για όλες τις εφαρμογές με χρήση της εντολής:

```
sudo apt-get update
```

Για να μπορέσει ο Node exporter να τεθεί σε λειτουργία, θα πρέπει το αρχείο τύπου “service” του Node exporter, το οποίο βρίσκεται στο path “/lib/systemd/system”, να περιλαμβάνει τα παρακάτω:

```
[Unit]
Description=Prometheus exporter for machine metrics
Documentation=https://github.com/prometheus/node_exporter

[Service]
Restart=always
User=prometheus
EnvironmentFile=/etc/default/prometheus-node-exporter
```

```
ExecStart=/usr/bin/prometheus-node-exporter $ARGS
ExecReload=/bin/kill -HUP $MAINPID
TimeoutStopSec=20s
SendSIGKILL=no

[Install]
WantedBy=multi-user.target
```

Στη συνέχεια, οι παρακάτω εντολές είναι ιδιαίτερα χρήσιμες για την διαχείριση του exporter. Αρχικά, για να ενεργοποιείται αυτόματα ο Node exporter κατά την εκκίνηση του συστήματος θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable prometheus-node-exporter
```

Στη συνέχεια, για να επανεκκινηθούν τα αρχεία τύπου "unit", θα πρέπει να εκτελεστεί η εντολή:

```
sudo systemctl daemon-reload
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Node exporter, αντίστοιχα.

```
sudo systemctl start prometheus-node-exporter
sudo systemctl stop prometheus-node-exporter
sudo systemctl status prometheus-node-exporter
```

Για να προστεθεί ένας νέος κόμβος στον οποίο συλλέγονται δεδομένα μέσω του Node exporter, για παρακολούθηση στον Prometheus, θα πρέπει να τροποποιηθεί κατάλληλα το αρχείο παραμετροποίησης "**prometheus.yml**". Αυτό, βρίσκεται στο μηχάνημα που έχει εγκατασταθεί ο Prometheus, εν προκειμένω στον εξυπηρετητή "cn-monitor-1".

Επόμενο βήμα, αποτελεί να επεξεργαστεί κανείς με κάποιον Text-Editor το αρχείο αυτό, ώστε να προστεθούν οι προς παρακολούθηση κόμβοι. Ανοίγοντας με κάποιον Text-Editor το αρχείο, προσθέτουμε στο τμήμα του "**scarpe_configs**" τα παρακάτω:

```
- job_name: 'node-exporter'
  static_configs:
    - targets: ['cn-monitor-1.cn.ece.ntua.gr:9100', 'ladon.telecom.ece.ntua.gr:9100',
'ulairi.telecom.ece.ntua.gr:9100', 'cn-monitor-2.cn.ece.ntua.gr:9100',
```

'courses.cn.ece.ntua.gr:9100', 'bbb.cn.ntua.gr:9100', 'bbb2.cn.ntua.gr:9100',
'gitlab.telecom.ece.ntua.gr:9100', 'bench-moodle-1:9100', 'onos.telecom.ece.ntua.gr:9100']

Ως "job_name" μπορεί να προστεθεί οποιοδήποτε όνομα, ώστε να γνωρίζει ο διαχειριστής ότι πρόκειται για τον Node exporter.
Στο τμήμα "**static_configs**" και συγκεκριμένα στο "**targets**", θα πρέπει να οριστούν τα μηχανήματα τύπου UNIX, τα οποία επιθυμεί κανείς να παρακολουθήσει και στα οποία προηγουμένως έχει εγκατασταθεί ο Node exporter.

Διακρίνεται από την παραπάνω ρύθμιση, ότι σε περίπτωση που απαιτείται η παρακολούθηση πάνω από ενός μηχανήματος, τότε αυτό επιτυγχάνεται με τον διαχωρισμό τους με κόμμα μέσα στην επιλογή "targets".

Από προεπιλογή, ο Node exporter "ακούει" στην θύρα "9100", εάν επιθυμεί κανείς να αλλάξει την επιλογή αυτή θα πρέπει να τροποποιήσει κατάλληλα το αρχείο τύπου "service" [44].

6.2: Εγκατάσταση του BigBlueButton exporter

Για την εγκατάσταση του BigBlueButton exporter, μπορεί να ακολουθήσει κανείς τις οδηγίες που παρέχονται στην ιστοσελίδα του BigBlueButton [45], [46]. Παρέχονται αναλυτικά βήματα για την εγκατάσταση, τόσο για "Docker" όσο και για "Systemd Installation". Στην προκειμένη περίπτωση, ακολουθείται η μέθοδος του "Systemd Installation".

Καθότι απαιτείται η εντολή "python3-pip", θα πρέπει να εγκατασταθεί ως εξής:

```
sudo apt install python3-pip -y
```

Στη συνέχεια, μετακινούμαστε στον φάκελο "/opt", ώστε να εγκαταστήσουμε εκεί το repository του BigBlueButton exporter από το Github.

```
cd /opt
```

Θα πρέπει να ελεγχθεί εάν είναι εγκατεστημένο το εργαλείο Git, ώστε να μπορεί να χρησιμοποιηθεί η εντολή "git clone". Αυτό επιτυγχάνεται με την εντολή:

```
sudo apt policy git
```

Εφόσον αναγράφεται "Installed: none", θα πρέπει να εγκατασταθεί εκ νέου.

```
sudo apt install git -y
```

Έχοντας εγκατεστημένο το εργαλείο Git, το χρησιμοποιούμε για να εγκαταστήσουμε τον BigBlueButton exporter ως εξής:

```
sudo git clone https://github.com/greenstatic/bigbluebutton-exporter.git
```

Ύστερα, αφού μετακινηθούμε στον φάκελο "/opt/bigbluebutton-exporter", εγκαθιστούμε τα απαραίτητα dependencies, που περιγράφονται στο αρχείο "requirements.txt", ως εξής:

```
sudo pip3 install -r requirements.txt
```

Έπειτα, δημιουργούμε έναν "non-privileged" χρήστη για τον exporter και ορίζουμε τα κατάλληλα δικαιώματα:

```
sudo useradd -r -d /opt/bigbluebutton-exporter -s /usr/sbin/nologin bbb-exporter
sudo chown -R bbb-exporter:bbb-exporter /opt/bigbluebutton-exporter
```

Στη συνέχεια, αντιγράφουμε το αρχείο “bigbluebutton-exporter.service”, το οποίο αποτελεί το service του BigBlueButton exporter, στον φάκελο “/lib/systemd/system/”.

```
sudo cp /opt/bigbluebutton-exporter/extras/systemd/bigbluebutton-exporter.service
/lib/systemd/system/
```

Έπειτα, δημιουργούμε τον παρακάτω φάκελο:

```
sudo mkdir /etc/bigbluebutton-exporter
```

και αντιγράφουμε μέσα σε αυτόν, τα περιεχόμενα του φακέλου “/opt/bigbluebutton-exporter/extras/systemd/bigbluebutton-exporter” με την παρακάτω εντολή:

```
sudo cp /opt/bigbluebutton-exporter/extras/systemd/bigbluebutton-exporter/*
/etc/bigbluebutton-exporter
```

Επιπλέον, χρησιμοποιώντας κάποιον Text-Editor, ορίζουμε τις παραμέτρους “API_BASE_URL” και “API_SECRET”, στο αρχείο “/etc/bigbluebutton-exporter/settings.env”.

Αμέσως μετά, για να φορτώσουμε εκ νέου τα αρχεία τύπου “service”, συμπεριλαμβανομένου και του “BigBlueButton service”, ώστε να συμπεριληφθεί το “service” που δημιουργήθηκε, εκτελούμε:

```
sudo systemctl daemon-reload
```

Και αφού εκκινήσουμε τον BigBlueButton exporter ως:

```
sudo systemctl start bigbluebutton-exporter
```

με παρακάτω εντολή ενεργοποιούμε τον exporter, ώστε να εκκινεί αυτόματα μόλις ο εξυπηρετητής τίθενται σε λειτουργία (boot).

```
sudo systemctl enable bigbluebutton-exporter
```

Σε αυτό το σημείο, τονίζεται ότι εάν στον εξυπηρετητή που φιλοξενεί το BigBlueButton και συνακόλουθα τον BigBlueButton exporter, δεν έχει ενεργοποιηθεί το HTTPS, τότε το scheme στο αρχείο παραμετροποίησης του Prometheus: "prometheus.yml", θα είναι HTTP και υπάρχει η δυνατότητα χρήσης "basic HTTP authentication".

Για το σκοπό αυτό, θα πρέπει να οριστεί η δυνατότητα HTTP "basic authentication στον Nginx". Ωστόσο, για να επιτευχθεί αυτό απαιτείται να εγκατασταθεί το πακέτο (package) apache2-utils, όπως διακρίνεται παρακάτω:

```
sudo apt-get install apache2-utils
```

Στη συνέχεια, με την εντολή:

```
sudo htpasswd -c /etc/nginx/.htpasswd alexgeorg
```

δημιουργούμε ένα όνομα χρήστη ονόματι "**alexgeorg**" και στην προτροπή που εμφανίζεται πληκτρολογούμε τον επιθυμητό κωδικό πρόσβασης.

Ύστερα, προαιρετικά και για λόγους αξιοπιστίας, δημιουργούμε έναν πρόχειρο φάκελο, ώστε να γράψουμε τα απαραίτητα που θα χρειαστούν για την παραμετροποίηση του Nginx.

```
mkdir /etc/nginx/bigbluebutton
touch /etc/nginx/bigbluebutton/monitoring.nginx
```

και με κάποιον Text-Editor προσθέτουμε τα εξής:

```
# BigBlueButton exporter (metrics)
location /metrics/ {
    auth_basic "BigBlueButton exporter";
    auth_basic_user_file /etc/nginx/.htpasswd;
    proxy_pass http://127.0.0.1:9688/;
    include proxy_params;
}
```

Ύστερα, μετακινούμαστε στον φάκελο "etc/nginx/sites-enabled" και προσθέτουμε στο κατάλληλο αρχείο, τα παραπάνω που μόλις γράψαμε.

Για να λάβουν χώρα οι αλλαγές που επιτελέσαμε, εκτελούμε με τη σειρά τις παρακάτω εντολές:

```
sudo systemctl daemon-reload
sudo systemctl restart nginx
sudo systemctl status nginx
```

Στη συνέχεια, στο αρχείο παραμετροποίησης του Prometheus, στον εξυπηρετητή που φιλοξενείται το Prometheus, εν προκειμένω στον "cn-monitor-1", προσθέτουμε τα παρακάτω:

```
# Monitoring BigBlueButton
- job_name: 'bbb'
  scrape_interval: 5s
  static_configs:
  - targets: ['bbb2.cn.ntua.gr:9688', 'bbb.cn.ntua.gr:9688']
```

Θα πρέπει να τονιστεί ότι, για λόγους αξιοπιστίας, η διαδικασία της εγκατάστασης πραγματοποιήθηκε σε δοκιμαστικούς εξυπηρετητές και όχι απευθείας στους εξυπηρετητές που φιλοξενούν το BigBlueButton για τα πλαίσια τηλεκπαίδευσης του Ε.Μ.Π. Γι' αυτό το λόγο, σε περίπτωση που επιθυμεί κανείς να ακολουθήσει την ίδια διαδικασία και να διαγράψει τον BigBlueButton exporter, θα πρέπει να ακολουθήσει τα παρακάτω βήματα [47].

Αρχικά, θα πρέπει να απενεργοποιηθεί το "service" του BigBlueButton, αλλά και να διαγραφεί σε επόμενο στάδιο:

```
sudo systemctl stop bigbluebutton-exporter
sudo systemctl disable bigbluebutton-exporter
sudo rm /usr/lib/systemd/system/ bigbluebutton-exporter
```

Επιπλέον, χρήσιμη είναι και η εκτέλεση των εντολών:

```
sudo systemctl daemon-reload
sudo systemctl reset-failed
```

Στη συνέχεια, απαιτείται η διαγραφή του φακέλου "/opt/bigbluebutton-exporter", καθώς και του "/etc/bigbluebutton-exporter" που είχε δημιουργηθεί.

Ομοίως, θα πρέπει να διαγραφεί ό,τι παραμετροποίηση πραγματοποιήθηκε στο αρχείο "/etc/nginx/sites-enabled".

6.3: Εγκατάσταση του Blackbox exporter

Ομοίως με προηγούμενες εγκαταστάσεις, κατεβάζουμε το κατάλληλο apt-package με την εντολή:

```
sudo apt-get install prometheus-blackbox-exporter
```

Στη συνέχεια, το μόνο που απαιτείται είναι η ρύθμιση του αρχείου παραμετροποίησης "prometheus.yml" του Prometheus. Προσθέτουμε λοιπόν τα παρακάτω στο αρχείο.

```
#Monitoring HTTP(S) Servers
- job_name: 'blackbox'
  metrics_path: /probe
  params:
    module: [http_2xx]  # Look for a HTTP 200 response.
  static_configs:
    - targets:
     - https://courses.cn.ntua.gr   # Target no.1
      - https://www.cn.ntua.gr       # Target no.2
      - https://students.ece.ntua.gr # Target no.3
      - https://helios.ntua.gr       # Target no.4
      - https://www.ece.ntua.gr      # Target no.5
      - http://users.ntua.gr         # Target no.6
      - https://bbb2.cn.ntua.gr      # Target no.7
      - https://bbb.cn.ntua.gr       # Target no.8
      - http://onos.telecom.ece.ntua.gr # Target no.9
      - https://cn-monitor-1.cn.ece.ntua.gr:3000 # Target no.10
      - http://cn-monitor-1.cn.ece.ntua.gr:9090 # Target no.11
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
            replacement:  cn-monitor-2.cn.ntua.gr:9115    # The blackbox exporter's real hostname:port.
```

6.4: Εγκατάσταση του SNMP exporter

Ομοίως με προηγούμενες εγκαταστάσεις, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install prometheus-snmp-exporter
```

Στη συνέχεια, για τη ρύθμιση του αρχείου παραμετροποίησης “prometheus.yml”, προσθέτουμε τα παρακάτω:

```
# Monitoring SNMP devices
- job_name: 'snmp'
  static_configs:
    - targets:
      - alderaan.cn.ece.ntua.gr
  metrics_path: /metrics
  params:
    module: [if_mib]
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: cn-monitor-2.cn.ece.ntua.gr:9116  # The SNMP exporter's real
hostname:port.
  scrape_interval: 10s
```

Αμέσως μετά, για να φορτώσουμε εκ νέου τα αρχεία τύπου “service”, συμπεριλαμβανομένου και του “prometheus-snmp-exporter.service”, ώστε να συμπεριληφθεί το “service” που δημιουργήθηκε, εκτελούμε:

```
sudo systemctl daemon-reload
```

Τέλος, εκκινούμε το “prometheus-snmp-exporter service” και ενεργοποιούμε τον exporter, ώστε να εκκινεί αυτόματα όταν ο εξυπηρετητής τίθενται σε λειτουργία (boot).

```
sudo systemctl start prometheus-snmp-exporter
sudo systemctl enable prometheus-snmp-exporter
```

6.5: Εγκατάσταση του VMware exporter

Για την εγκατάσταση του VMware exporter στον εξυπηρετητή “cn-monitor-2”, αρχικά μετακινούμαστε στον φάκελο “/opt”, ώστε να εγκαταστήσουμε εκεί το repository του VMware exporter από το Github.

Έχοντας εγκατεστημένο το εργαλείο Git, το χρησιμοποιούμε για να εγκαταστήσουμε το repository του VMware exporter ως εξής:

```
git clone https://github.com/pryorda/vmware_exporter/
```

Ύστερα, αφού μετακινηθούμε στον φάκελο “/opt/vmware_exporter”, εγκαθιστούμε τον exporter ως εξής:

```
python3 setup.py install
```

Ύστερα, δημιουργούμε το αρχείο παραμετροποίησης “config.yml” και προσθέτουμε τα παρακάτω:

```
default:

#esx:
  vsphere_host: 147.102.7.3
  vsphere_user: '*****'
  vsphere_password: '********'
  ignore_ssl: True
  specs_size: 5000
  fetch_custom_attributes: False
  fetch_tags: False
  fetch_alarms: True
  collect_only:
    vms: True
    vmguests: True
    datastores: True
    hosts: True
    snapshots: True
```

Στη συνέχεια, στο αρχείο παραμετροποίησης του Prometheus, προσθέτουμε τα παρακάτω:

```
# Monitoring Vmware (Esx, Vcenter etc.)
- job_name: 'vmware_vcenter'
  metrics_path: '/metrics'
  static_configs:
    - targets:
      - 'atlas.telecom.ece.ntua.gr'
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: 147.102.40.79:9272
```

Ύστερα, ο VMware exporter μπορεί να ενεργοποιηθεί με την εντολή:

```
vmware_exporter -c /opt/vmware_exporter/config.yml
```

Ωστόσο, για να είναι δυνατή η καλύτερη διαχείριση του VMware exporter, θα δημιουργηθεί ένα αρχείο τύπου "service" [48]. Για το σκοπό αυτό, δημιουργούμε το αρχείο "vmware-exporter.service" στον φάκελο "/etc/systemd/system". Στη συνέχεια, με τη βοήθεια ενός Text-Editor, επεξεργαζόμαστε το αρχείο προσθέτοντας τα παρακάτω:

```
[Unit]
Description=Vmware-exporter

[Service]
ExecStart=vmware_exporter -c /opt/vmware_exporter/config.yml

[Install]
WantedBy=multi-user.target
```

Ύστερα, για να φορτώσουμε εκ νέου τα αρχεία τύπου "service", ώστε να περιληφθεί το service που φτιάξαμε εκτελούμε:

```
sudo systemctl daemon-reload
```

Για να ενεργοποιήσουμε το "service", ώστε να εκκινεί αυτόματα όταν ο εξυπηρετητής τίθενται σε λειτουργία (boot), εκτελούμε την παρακάτω εντολή:

```
sudo systemctl enable vmware-exporter.service
```

Τέλος, για να εκκινήσουμε το “service”, αλλά και να απεικονίσουμε την κατάστασή του αντίστοιχα, εκτελούμε τις εντολές:

```
sudo systemctl start vmware-exporter.service
sudo systemctl status vmware-exporter.service
```

Όπως προαναφέρθηκε, στο κεφάλαιο περί Blackbox exporter, ο VMware exporter υποστηρίζει το **multi-target exporter pattern** που υποστηρίζει το Prometheus. Γι’ αυτό το σκοπό, παρομοίως μέσω “curl” μπορούμε να λάβουμε για παράδειγμα τις μετρικές ενός ESXi host ονόματι “uranos” ως εξής:

```
curl -vvv "http://cn-monitor-2.cn.ntua.gr:9272/metrics?section="uranos""
```

6.6: Εγκατάσταση του Gitlab

Αρχικά, για να πραγματοποιηθεί επιτυχώς η εγκατάσταση του “Gitlab”, θα πρέπει να προστεθούν τα προαπαιτούμενα πακέτα ως εξής:

```
sudo apt install tzdata curl ca-certificates openssh-server
```

Στη συνέχεια, εγκαθιστούμε το “GPG key” μέσω wget στην “APT installation list” από “trusted keys”, το οποίο θα μας επιτρέψει να κατεβάσουμε το κατάλληλο πακέτο:

```
gpg_key_url="https://packages.gitlab.com/gitlab/gitlab-ce/gpgkey"
curl -fsSL $gpg_key_url| sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/gitlab.gpg
```

Ύστερα, προσθέτουμε το Gitlab repository για σύστημα Ubuntu, στο αρχείο “sources.list.d”:

```
sudo tee /etc/apt/sources.list.d/gitlab_gitlab-ce.list<<EOF

deb https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/ focal main

deb-src https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/ focal main

EOF
```

Τέλος, εγκαθιστούμε το Gitlab ως εξής:

```
sudo apt update && sudo apt install gitlab-ce
```

6.7: Ενεργοποίηση του Gitlab exporter

Για την ενεργοποίηση του Gitlab exporter, θα πρέπει στο αρχείο παραμετροποίησης "/etc/gitlab/gitlab.rb" του Gitlab, να προστεθούν οι εξής ρυθμίσεις:

```
gitlab_exporter['enable'] = true
 gitlab_exporter['log_directory'] = "/var/log/gitlab/gitlab-exporter"
 gitlab_exporter['home'] = "/var/opt/gitlab/gitlab-exporter"
 gitlab_exporter['listen_address'] = '0.0.0.0'
 gitlab_exporter['listen_port'] = '9168'
 node_exporter['enable'] = true
 node_exporter['listen_address'] = '0.0.0.0:9100'
```

Οι παραπάνω επιλογές ενεργοποιούν τον exporter, ορίζουν τον "home" φάκελο, καθώς και τον φάκελο στον οποίο αποθηκεύονται τα αρχεία καταγραφής του, αλλά και την διεύθυνση IP και την θύρα που εκείνος "ακούει".
Οι τελευταίες δύο γραμμές αφορούν στην ενεργοποίηση του Node exporter.

Τέλος, στον εξυπηρετητή που φιλοξενεί το Prometheus, εν προκειμένω στο μηχάνημα "cn-monitor-1", θα πρέπει να προστεθούν στο αρχείο παραμετροποίησης "prometheus.yml", ώστε να υπάρχει επικοινωνία με τον Prometheus, τα εξής:

```
# Monitoring Gitlab server
- job_name: gitlab_exporter_metrics
  metrics_path: "/metrics"
  static_configs:
    - targets:
      - gitlab.telecom.ece.ntua.gr:9168
```

6.8: Εγκατάσταση του Fail2Ban

Λαμβάνοντας υπόψη την δικτυακή ασφάλεια της υποδομής που χρησιμοποιείται, αποτελεί σημαντικό βήμα η χρήση του λογισμικού Fail2ban [49].

Στιγμιότυπο 6.8: Λογότυπο Fail2Ban

Το λογισμικό αυτό, επιβλέπει συνεχώς ορισμένα αρχεία καταγραφής του συστήματος, ώστε να διαπιστώσει τυχόν ανωμαλίες που μπορεί να σχετίζονται με κάποια επίθεση.

Μια από τις πιο χρήσιμες δυνατότητες του, είναι η δημιουργία συγκεκριμένων και παραμετροποιημένων κανόνων τείχους προστασίας (firewall) που περιορίζουν συγκεκριμένες διευθύνσεις IP ύστερα από έναν αριθμό αποτυχημένων προσπαθειών ταυτοποίησης.

Στο αρχείο "/var/log/fail2ban.log", μπορεί να ενημερωθεί κανείς σχετικά με τα αρχεία καταγραφής του Fail2ban και συγκεκριμένα για το ποιες διευθύνσεις IP είναι σε κατάσταση: αποκλεισμού (ban), απελευθέρωσης (unban) ή εντοπισμού (found) με την εντολή "sudo tail /var/log/fail2ban.log", το αποτέλεσμα της οποίας διακρίνεται στον παρακάτω πίνακα.

fail2ban.filter	[22098]: INFO	[sshd] Found	67.169.127.118
fail2ban.filter	[22098]: INFO	[sshd] Found	67.169.127.118
fail2ban.actions	[22098]: NOTICE	[sshd] Ban	67.169.127.118
fail2ban.filter	[22098]: INFO	[sshd] Found	67.169.127.118
fail2ban.actions	[22098]: NOTICE	[sshd] Unban	82.66.145.150
fail2ban.filter	[22098]: INFO	[sshd] Found	211.199.177.129
fail2ban.filter	[22098]: INFO	[sshd] Found	211.199.177.129
fail2ban.filter	[22098]: INFO	[sshd] Found	179.60.147.122
fail2ban.actions	[22098]: NOTICE	[sshd] Unban	67.169.127.118
fail2ban.filter	[22098]: INFO	[sshd] Found	92.255.85.70

Ομοίως με προηγούμενες εγκαταστάσεις, εγκαθιστούμε το πακέτο που μας ενδιαφέρει:

```
sudo apt install fail2ban -y
```

Ο φάκελος "/etc/fail2ban" περιλαμβάνει κάποια αρχεία, τα οποία θα χρειαστούν παραμετροποίηση, ώστε να επιτευχθούν οι επιθυμητές ενέργειες του διαχειριστή.

Τα αρχεία, τα οποία θα παραμετροποιηθούν, είναι τα "fail2ban.conf" και "jail.conf". Γι' αυτό το λόγο, για μια πιο ασφαλή παραμετροποίηση και για την αποφυγή πιθανών λαθών, δημιουργούμε τα αντίγραφα αρχεία στα οποία θα προσθέσουμε τις αλλαγές και στη συνέχεια θα τα αντιγράψουμε στα αυθεντικά αρχεία [50]. Αυτό επιτυγχάνεται ως εξής:

```
sudo cp fail2ban.conf fail2ban.local
sudo cp jail.conf jail.local
```

Κάποιες από τις παραμετροποιήσεις που έγιναν στο αρχείο "jail.conf" είναι οι εξής:

```
# "bantime" is the number of seconds that a host is banned.
bantime = 120m

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime = 120m

# "maxretry" is the number of failures before a host gets banned.
maxretry = 4
```

Οι παραπάνω ρυθμίσεις αφορούν στις εξής λειτουργίες:

- bantime: ο χρόνος σε δευτερόλεπτα για τον οποίο αποκλείεται (ban) κάποια IP διεύθυνση.
- maxretry: ο μέγιστος αριθμός αποτυχημένων προσπαθειών, προτού αποκλειστεί μια IP διεύθυνση.
- findtime: ο χρόνος στον οποίο μπορεί κανείς να προσπαθήσει να συνδεθεί στο σύστημα έχοντας το πολύ "maxretry" προσπάθειες. Ο χρόνος μετρά από την πρώτη ανεπιτυχή προσπάθεια.

Ακόμη, διατίθεται και η δυνατότητα για αποστολή ειδοποιητικών συναγερμών μέσω email σε περίπτωση που ο διαχειριστής επιθυμεί να ενημερώνεται. Οι ρυθμίσεις αυτές, γίνονται στα πεδία "destemail", "sender" και "mta", όπως διακρίνεται παρακάτω.

```
destemail = root@localhost

# Sender email address used solely for some actions
sender = root@<fq-hostname>
```

```
# E-mail action. Since 0.8.1 Fail2Ban uses sendmail MTA for the
# mailing. Change mta configuration parameter to mail if you want to
# revert to conventional 'mail'.
mta = sendmail
```

Όπως είναι αναμενόμενο, ο κάθε διαχειριστής μπορεί να πραγματοποιήσει όποιες ρυθμίσεις επιθυμεί.

Μια πολύ χρήσιμη εντολή που απεικονίζει ποιες υπηρεσίες είναι ενεργοποιημένες στην "Jail list" του Fail2ban είναι η "sudo fail2ban-client status", το αποτέλεσμα της οποίας αναγράφει:

```
Status
- Number of jail: 1
- Jail list: sshd
```

Παρατηρούμε, ότι στη λίστα περιλαμβάνεται η "sshd", επομένως το Fail2ban θα λειτουργήσει για το πρωτόκολλο SSH.

Για να ενεργοποιείται αυτόματα το Fail2ban κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable fail2ban
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Fail2ban αντίστοιχα.

```
sudo systemctl start fail2ban
sudo systemctl stop fail2ban
sudo systemctl status fail2ban
```

Σε αυτό το σημείο, θα πρέπει να σημειωθεί, ότι το Fail2ban εγκαθίσταται σε όσους εξυπηρετητές χρησιμοποιούνται στην παρούσα διπλωματική εργασία.

6.9: Εγκατάσταση του UFW

Ένα περαιτέρω βήμα για την ασφάλεια των συστημάτων που χρησιμοποιούνται, αποτελεί η εγκατάσταση ενός λογισμικού τείχους προστασίας, το οποίο θα ρυθμίζει και θα θέτει ορισμένα φίλτρα, αναφορικά με την δικτυακή κίνηση.
Το UFW (Uncomplicated Firewall) [51], καθιστά την παραπάνω διαδικασία εύκολη σε αντίθεση με άλλα εργαλεία όπως "iptables" ή "nftables", καθώς και άμεση, χρησιμοποιώντας τη γραμμή εντολών.

Όπως ισχύει και για τα περισσότερα λογισμικά αυτού του είδους, έτσι και στο UFW, οι κανόνες τοποθετούνται σε έναν πίνακα κατά σειρά προτεραιότητας γραμμής.
Κάθε γραμμή ορίζει και έναν κανόνα και ο αριθμός γραμμής είναι εκείνος που χρησιμοποιείται σε περίπτωση διαγραφής ή περαιτέρω ρύθμισης ενός κανόνα.
Για τον λόγο αυτό, θα πρέπει ορισμένοι αυστηρότεροι και πιο εξειδικευμένοι κανόνες να βρίσκονται στις πρώτες γραμμές του πίνακα.

Κάποιες από τις κυριότερες επιλογές που διαθέτει, αποτελούν οι παρακάτω εντολές:

- allow: επιτρέπει συγκεκριμένη κίνηση.
- deny: αποτρέπει συγκεκριμένη κίνηση.
- reject: απορρίπτει συγκεκριμένη κίνηση.
- limit: περιορίζει συγκεκριμένη κίνηση θέτοντας επιπλέον περιορισμούς.
- status: προβάλλει την κατάσταση των κανόνων.
- show: προβάλλει πληροφορίες για το τρεχούμενο firewall.
- delete: διαγράφει κάποιον κανόνα.
- reload: επανεκκινεί το firewall.
- enable: ενεργοποιεί το firewall αυτόματα σε κάθε εκκίνηση του μηχανήματος.
- disable: απενεργοποιεί το firewall αυτόματα σε κάθε εκκίνηση του μηχανήματος.

Αναφορικά με την εγκατάσταση του UFW, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install ufw
```

Για να ενεργοποιείται αυτόματα το UFW κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable ufw.service
```

Με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του UFW αντίστοιχα.

```
sudo systemctl start ufw.service
sudo systemctl stop ufw.service
sudo systemctl status ufw.service
```

Στη συνέχεια, ενεργοποιούμε το default policy για τις εξερχόμενες κινήσεις, ώστε να υπάρχει πρόσβαση στο διαδίκτυο.

```
sudo ufw default allow outgoing
```

Σε αυτό το σημείο, θα θέσουμε κάποιους κανόνες (firewall rules), ώστε να επιτρέπεται συγκεκριμένη κίνηση **από** και **προς** ορισμένα δίκτυα και θα τεθούν ως ανοιχτές συγκεκριμένες θύρες.

Ύστερα, στον εξυπηρετητή "cn-monitor-1" ενεργοποιούμε τις θύρες, τις οποίες θέλουμε να έχουμε ανοιχτές. Αυτές είναι οι: 22 που αφορά στο πρωτόκολλο SSH, η 9090 που αφορά στη γραφική διεπαφή του Prometheus, η 80 στο HTTP, η 443 στο HTTPS και η 3000 στη γραφική διεπαφή του Grafana αντίστοιχα. Οι ενέργειες αυτές, επιτυγχάνονται με τις επόμενες εντολές.

```
sudo ufw allow 22
sudo ufw allow 'Nginx Full'
sudo ufw allow 3000
sudo ufw allow 9090
```

Θα πρέπει να σημειωθεί, ότι οι θύρες "9090" και "3000" είναι "ανοιχτές" και εκτός του δικτύου του Πολυτεχνείου, ώστε να υπάρχει πρόσβαση και τοπικά από web browser.
Στη συνέχεια, θα ρυθμιστούν τα δίκτυα και το εύρος των IP διευθύνσεων με το οποίο επιθυμούμε να έχουμε επικοινωνία.

Αναφορικά με τις διευθύνσεις IP του Ε.Μ.Π, επιθυμούμε να υπάρχει πρόσβαση μόνο από και προς τα δίκτυα: 147.102.7.0/24, 147.102.39.0/24 και 147.102.40.0/24.

Αυτό επιτυγχάνεται με τις παρακάτω εντολές:

```
sudo ufw allow from 147.102.7.0/24 to any
sudo ufw allow from 147.102.39.0/24 to any
sudo ufw allow from 147.102.40.0/24 to any
```

Τα αποτελέσματα των παραπάνω εντολών απεικονίζονται με τη εντολή "sudo ufw status", όπως διακρίνεται παρακάτω.

```
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
3000                       ALLOW       Anywhere
9090                       ALLOW       Anywhere
Nginx Full                 ALLOW       Anywhere
Anywhere                   ALLOW       147.102.40.0/24
Anywhere                   ALLOW       147.102.7.0/24
Anywhere                   ALLOW       147.102.39.0/24
22/tcp (v6)                ALLOW       Anywhere (v6)
3000 (v6)                  ALLOW       Anywhere (v6)
9090 (v6)                  ALLOW       Anywhere (v6)
Nginx Full (v6)            ALLOW       Anywhere (v6)
```

Ας σημειωθεί επίσης πως σε περίπτωση που επιθυμεί κανείς να διαγράψει κάποιον κανόνα, αυτό επιτυγχάνεται πολύ εύκολα με χρήση της εντολής:

```
sudo ufw delete <line>
```

όπου “<line>”, είναι η γραμμή που επιθυμεί κανείς να διαγράψει.

Αναφορικά με τον εξυπηρετητή “cn-monitor-2” στον οποίο είναι εγκατεστημένοι συγκεκριμένοι exporters για την συγκέντρωση εξειδικευμένων μετρικών, οι θύρες που επιθυμούμε να είναι “ανοιχτές” είναι οι 22, οι 80 και 443, ενώ τα δίκτυα με τα οποία επιθυμούμε να έχουμε επικοινωνία είναι τα 147.102.7.0/24, 147.102.39.0/24 και 147.102.40.0/24 (υποδίκτυα Ε.Μ.Π). Αντίστοιχα, ο πίνακας με τους κανόνες του τείχους προστασίας για το “cn-monitor-2” απεικονίζεται στην επόμενη εικόνα.

```
Status: active

To                         Action      From
--                         ------      ----
22                         ALLOW       Anywhere
Nginx Full                 ALLOW       Anywhere
147.102.40.79              ALLOW       147.102.7.0/24
147.102.40.79              ALLOW       147.102.39.0/24
147.102.40.79              ALLOW       147.102.40.0/24
22 (v6)                    ALLOW       Anywhere (v6)
Nginx Full (v6)            ALLOW       Anywhere (v6)
```

6.10: Εγκατάσταση SSL πιστοποιητικού σε Nginx

Για λόγους αξιοπιστίας και ασφάλειας σε επίπεδο εξυπηρετητή ιστού, θα ενεργοποιηθεί το πρωτόκολλο HTTPS στον εξυπηρετητή "cn-monitor-1", ώστε να υπάρχει κρυπτογράφηση του περιεχομένου που επεξεργαζόμαστε. Ο εξυπηρετητής ιστού που χρησιμοποιείται, είναι ο Nginx αντί του Apache [52]. Και οι δύο επιλογές δύναται να χρησιμοποιηθούν, ωστόσο λόγω ταχύτητας αλλά και ευελιξίας, επιλέχθηκε ο Nginx.

Στιγμιότυπο 6.10.1: Λογότυπο Nginx

Για τη χρήση του HTTPS στον εξυπηρετητή "cn-monitor-1", θα πρέπει σε πρώτο στάδιο να δημιουργηθεί ένα SSL πιστοποιητικό [53].
Για το σκοπό αυτό, χρησιμοποιείται η Let's Encrypt η οποία είναι μια δωρεάν υπηρεσία Certificate Authority (CA) [54].

Στιγμιότυπο 6.10.2: Λογότυπο Let's Encrypt

Το Let's Encrypt περιλαμβάνει έναν "software client" τον Certbot, ο οποίος απλοποιεί την παραπάνω διαδικασία.

Σε περίπτωση που είναι ήδη εγκατεστημένος ο Apache και επιθυμεί κανείς να τον διαγράψει, ώστε να χρησιμοποιήσει τον Nginx, μπορεί να ακολουθήσει τα παρακάτω απλά βήματα.

Για να απενεργοποιηθεί ο apache server:

```
sudo systemctl stop apache2
```

Για να διαγραφούν όλα τα στοιχεία του πληκτρολογούμε:

```
sudo apt-get remove --purge apache2 apache2-utils
```

```
sudo rm -rf /etc/apache2
sudo apt-get autoremove
```

Έχοντας διαγράψει τον Apache, εγκαθιστούμε το κατάλληλο apt-package που μας ενδιαφέρει ως εξής:

```
sudo apt install nginx
```

Ομοίως, για την εγκατάσταση του Certbot εκτελούμε:

```
sudo apt install certbot python3-certbot-nginx -y
```

Στη συνέχεια, θα χρησιμοποιηθεί η παρακάτω εντολή για να δημιουργηθεί το SSL πιστοποιητικό για το domain "**cn-monitor-1.cn.ece.ntua.gr**":

```
sudo certbot -d cn-monitor-1.cn.ece.ntua.gr
```

Στην προτροπή που εμφανίζεται, πληκτρολογούμε το επιθυμητό email, για την διαχείριση των ειδοποιήσεων σχετικά με την ανανέωση των πιστοποιητικών.

```
Enter email address (used for urgent renewal and security notices) (Enter 'c' to cancel):
my_email@ntua.gr
```

Εκτός τούτων, υποστηρίζεται η δυνατότητα για **ανακατεύθυνση** (redirect) της HTTP κίνησης σε HTTPS, αφαιρώντας το HTTP. Παρ' όλα αυτά, το HTTP χρησιμοποιείται μέσω reverse proxy στην θύρα "5601" για την χρήση της υπηρεσίας Kibana.
Επομένως, η παραμετροποίηση αυτή γίνεται ξεχωριστά και λεπτομέρειες για την διαδικασία αυτή παρέχονται στο κεφάλαιο 6.20.b του Kibana.

Για να οριστεί ένα "HTTP basic authentication" στον Nginx, απαιτείται η εγκατάσταση του πακέτου "apache2-utils", όπως διακρίνεται παρακάτω:

```
sudo apt-get install apache2-utils
```

Στη συνέχεια με την εντολή:

```
sudo htpasswd -c /etc/nginx/.htpasswd user
```

δημιουργούμε ένα όνομα χρήστη “user” και στην αντίστοιχη προτροπή που εμφανίζεται πληκτρολογούμε τον επιθυμητό κωδικό πρόσβασης.

Τέλος, το SSL πιστοποιητικό αποθηκεύεται στον φάκελο “/etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr” με όνομα **fullchain.pem**, ενώ το κλειδί του πιστοποιητικού (certificate key) στον ίδιο φάκελο με όνομα **privkey.pem**.

Έχοντας δημιουργήσει το κλειδί και το SSL πιστοποιητικό, το Certbot αναλαμβάνει την κατάλληλη τροποποίηση των αρχείων:

/etc/nginx/sites-available/cn-monitor-1.cn.ece.ntua.gr
/etc/nginx/sites-enabled/cn-monitor-1.cn.ece.ntua.gr

Με άλλα λόγια, προστίθενται αυτόματα οι γραμμές:

```
listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;
```

Οι οποίες είναι απαραίτητες για να ορισθούν τα κατάλληλα paths για το πιστοποιητικό και το κλειδί του πιστοποιητικού και να είναι διαθέσιμο το HTTPS.

Σε αυτό το σημείο, θα πρέπει να δοθεί ιδιαίτερη προσοχή στο γεγονός, ότι ενδέχεται λόγω δικαιωμάτων των αρχείων που δημιουργήθηκαν να **μην** υπάρχει πρόσβαση από τον Nginx στο πιστοποιητικό και στο κλειδί του πιστοποιητικού.

Πιο συγκεκριμένα, στα πλαίσια της παραπάνω διαδικασίας στον εξυπηρετητή “cn-monitor-1”, παρουσιάστηκε μετά τη δημιουργία του πιστοποιητικού και την εκτέλεση της εντολή “sudo sudo nginx -t” το παρακάτω μήνυμα:

```
nginx: [alert] could not open error log file: open() "/var/log/nginx/error.log" failed (13:
Permission denied)
[warn] 3803147#3803147: the "user" directive makes sense only if the master process runs
with super-user privileges, ignored in /etc/nginx/nginx.conf:1
[emerg] 3803147#3803147: cannot load certificate
"/etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem": BIO_new_file() failed
(SSL: error:0200100D:system library:fopen:Permission
denied:fopen('/etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem','r')
error:2006D002:BIO routines:BIO_new_file:system lib)
nginx: configuration file /etc/nginx/nginx.conf test failed
```

σύμφωνα με το οποίο δεν υπάρχει πρόσβαση στον Nginx, ώστε να χρησιμοποιήσει το SSL πιστοποιητικό, το οποίο βρίσκεται στον φάκελο "/etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr".

Το πρόβλημα αυτό, διορθώνεται εύκολα εάν οριστεί ο κατάλληλος χρήστης στο αρχείο παραμετροποίησης "/etc/nginx/nginx.conf". Πιο συγκεκριμένα, από χρήστης "www-data" που έχει ορισθεί από προεπιλογή, θα πρέπει να τεθεί ως χρήστης ο "root".

Τέλος, μέσω της εντολής "sudo nginx -t", ελέγχονται τυχόν λάθη στο αρχείο παραμετροποίησης του Nginx και εφόσον δεν υπάρχει κάποιο πρόβλημα, φορτώνουμε το "service" του Nginx:

```
sudo systemctl reload nginx.service
```

Εκτός τούτων, ας σημειωθεί ότι για την χρήση του HTTPS, θα πρέπει να είναι ενεργοποιημένη η θύρα "443" στο UFW.

Επιπλέον, δε θα πρέπει να παραβλεφθεί το γεγονός ότι η ισχύς ενός πιστοποιητικού "Let's Encrypt" διαρκεί για διάστημα **90 ημερών** ύστερα από τη δημιουργία του. Μετά το πέρας αυτού του διαστήματος, το πιστοποιητικό χάνει την ισχύ του και θα πρέπει να ανανεωθεί.

Μέσω της εντολής "sudo certbot renew" εμφανίζεται η κατάσταση ενός πιστοποιητικού, καθώς και η ημερομηνία λήξης του, όπως διακρίνεται παρακάτω.

```
Saving debug log to /var/log/letsencrypt/letsencrypt.log

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Processing /etc/letsencrypt/renewal/cn-monitor-1.cn.ece.ntua.gr.conf
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Cert not yet due for renewal

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

The following certs are not due for renewal yet:
  /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem expires on 2023-01-03 (skipped)
No renewals were attempted.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Λαμβάνοντας υπόψη τα παραπάνω, η ανανέωση ενός πιστοποιητικού ανεξάρτητα του εάν το πιστοποιητικό πρόκειται να λήξει σύντομα ή όχι, επιτυγχάνεται με την εντολή:

```
sudo certbot renew –force-renewal --nginx
```

Σε περίπτωση που η διαδικασία της ανανέωσης είναι επιτυχής, τότε εμφανίζεται το παρακάτω αποτέλεσμα:

```
Saving debug log to /var/log/letsencrypt/letsencrypt.log

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Processing /etc/letsencrypt/renewal/cn-monitor-1.cn.ece.ntua.gr.conf
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Plugins selected: Authenticator nginx, Installer nginx
Renewing an existing certificate

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
new certificate deployed with reload of nginx server; fullchain is
/etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Congratulations, all renewals succeeded. The following certs have been renewed:
  /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem (success)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Παρ’ όλα αυτά, για λόγους ευελιξίας και αυτοματοποίησης, προτιμάται η μέθοδος της αυτόματης ανανέωσης του πιστοποιητικού. Αυτό αναλαμβάνεται αυτόματα από τον Certbot. Πιο συγκεκριμένα, για συστήματα που δεν χρησιμοποιούν το “systemd” ως “init system”, ο cron ελέγχει δύο φορές την ημέρα εάν έχει παρέλθει η ημερομηνία λήξης του πιστοποιητικού και σε αυτή τη περίπτωση ανανεώνει αυτόματα το SSL πιστοποιητικό.

Σε περίπτωση που χρησιμοποιείται ο “systemd” ως “init system”, τότε η αυτόματη ανανέωση επιτυγχάνεται με παρόμοιο τρόπο μέσω των:

- certbot.timer: ως “systemd timer” ελέγχει την ημερομηνία λήξης του πιστοποιητικού
- certbot.service: ως “systemd service” επιτελεί την ανανέωση του πιστοποιητικού.

Ωστόσο, είναι υποχρεωτικό να έχει ενεργοποιηθεί ο certbot timer, ώστε να εκκινείται σε κάθε επανεκκίνηση του συστήματος μέσω της εντολής:

```
sudo systemctl enable certbot.timer
```

6.11: Εγκατάσταση του Prometheus Alertmanager

Για να εγκαταστήσουμε το κατάλληλο apt-package εκτελούμε την εντολή:

```
sudo apt install prometheus-alertmanager
```

Στη συνέχεια, για καλύτερη οργάνωση των αρχείων που αφορούν στον Prometheus Alertmanager, κρίνεται σκόπιμο να δημιουργήσουμε τον φάκελο “alertmanager” εντός του φακέλου “/etc/prometheus”, με δικαιώματα “root” μέσω της εντολής:

```
sudo mkdir /etc/prometheus/alertmanager
```

Ύστερα, δημιουργούμε το αρχείο τύπου yml: “alertmanager.yml”.

```
sudo touch /etc/prometheus/alertmanager/alertmanager.yml
```

6.12: Κανόνες αποστολής ειδοποιητικών συναγερμών-σύνδεση με Prometheus

Για τον ορισμό κανόνων, κρίνεται σκόπιμο για την καλύτερη οργάνωση, να δημιουργηθούν ξεχωριστά αρχεία κανόνων για την κάθε υπηρεσία που μας ενδιαφέρει. Πιο συγκεκριμένα, δημιουργούνται τα αρχεία "node_exporter_rules.yml", "alertmanager_rules.yml", "vmware_rules.yml", "blackbox_rules.yml", "prometheus_rules.yml" για τον μηχανισμό αποστολής ειδοποιητικών συναγερμών (Alerting) σχετικά με τον Node exporter, τον Prometheus Alertmanager, τον VMware exporter, τον Blackbox exporter, αλλά και τον Prometheus αντίστοιχα.
Μια σημαντική λεπτομέρεια είναι, πως τα αρχεία αυτά θα πρέπει να βρίσκονται στον ίδιο φάκελο με το αρχείο "prometheus.yml" και συγκεκριμένα στον φάκελο "/etc/prometheus/".

Στη συνέχεια, για να ενσωματωθούν οι κανόνες στην λειτουργία του Prometheus, θα πρέπει να σημειωθούν τα αρχεία αυτά στο πεδίο "rule_files" του αρχείου παραμετροποίησης του Prometheus, prometheus.yml, όπως διακρίνεται παρακάτω:

```
# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets: ['cn-monitor-1.cn.ece.ntua.gr:9093']

rule_files:
  - prometheus_rules.yml
  - node_exporter_rules.yml
  - blackbox_rules.yml
  - vmware_rules.yml
  - alertmanager_rules.yml
```

Επιπλέον, προστίθεται και ένα job ονόματι "alertmanager", το οποίο είναι απαραίτητο για την διαχείριση και παρακολούθηση των μετρικών που συλλέγονται σχετικά με τον Prometheus Alertmanager.

```
- job_name: alertmanager
  static_configs:
        - targets: ['cn-monitor-1.cn.ece.ntua.gr:9093']
```

Για λόγους πληρότητας, παρακάτω παρουσιάζεται ένας κανόνας που αφορά στον Node exporter και βρίσκεται στο αρχείο "node_exporter_rules.yml":

```
groups:
- name: Node_exporter
  rules:
  - alert: HostOutOfMemory
    expr: node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes * 100 < 10
    for: 2m
    labels:
      severity: warning
    annotations:
      summary: Host out of memory (instance {{ $labels.instance }})
        description: "Node memory is filling up (< 10% left)\n  VALUE = {{ $value }}\n
LABELS = {{ $labels }}"
```

Παρατηρεί κανείς, ότι ο κανόνας ονομάζεται “HostOutOfMemory” και τίθεται σε κατάσταση “firing”, όταν ο ελεύθερος χώρος σε bytes της μνήμης του συγκεκριμένου host βρίσκεται κάτω από το 10% του συνολικού διαθέσιμου χώρου. Το γεγονός αυτό, υπολογίζεται στην μεταβλητή “expr”.

Πιο συγκεκριμένα, επιτελείται η πράξη: “node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes * 100 < 10”, το οποίο δεν είναι παρά ένας έλεγχος του εάν το ποσοστό των διαθέσιμων bytes της μνήμης είναι μικρότερο του 10% των συνολικών bytes.
Σε αυτό το σημείο, θα πρέπει να τονιστεί, ότι το στοιχείο “node_memory_MemAvailable_bytes”, όπως και το “node_memory_MemTotal_bytes” δεν είναι παρά **μετρικές**, οι οποίες έχουν συλλεχθεί από τον Node exporter.

Η μεταβλητή “for”, όπου ισούται με 2 λεπτά (2m), είναι μια προαιρετική επιλογή, η οποία ορίζει στον Prometheus να αναμείνει για μια ορισμένη διάρκεια, έως ότου αλλάξει την κατάσταση του ειδοποιητικού συναγερμού σε “firing”.

Σε αυτήν την περίπτωση, ο Prometheus θέτει τον συγκεκριμένο ειδοποιητικό συναγερμό σε κατάσταση “pending” για ένα διάστημα 2 λεπτών και σε περίπτωση που ο συναγερμός εξακολουθεί να είναι ενεργός ακόμη και μετά το πέρας αυτού του διαστήματος, τότε μεταβάλλεται η ειδοποίηση σε κατάσταση “firing”.

Από τον παραπάνω κανόνα, παρατηρεί κανείς ότι το “severity” είναι τύπου “warning”, ενώ στο πεδίο των “annotations”, η μεταβλητή “summary” φέρει το μήνυμα, το οποίο αποτυπώνεται είτε στον Prometheus, είτε στο Alerta σε περίπτωση που ενεργοποιηθεί ο συγκεκριμένος συναγερμός.

6.13: Prometheus Alertmanager και σύνδεση με Slack

Για την αποτύπωση των ειδοποιητικών συναγερμών μέσω της πλατφόρμας Slack, θα πρέπει να δημιουργηθεί ένα "Slack Workspace". Στη συνέχεια, από τις ρυθμίσεις "Administration"→"Manage Apps" αναζητούμε την επιλογή "**Incoming WebHooks**", την οποία προσθέτουμε στο "workspace" που διαθέτουμε.

Στη συνέχεια, θα πρέπει να επιλεχθεί το κανάλι, το οποίο θα φιλοξενήσει τους εισερχόμενους ειδοποιητικούς συναγερμούς. Στη προκειμένη περίπτωση, δημιουργήθηκε το κανάλι **#alerts**. Επιπλέον, θα πρέπει να αντιγραφεί το "WebHook URL", το οποίο απαιτείται για την κατάλληλη παραμετροποίηση του αρχείου "alertmanager.yml" και συγκεκριμένα για την παράμετρο "api_url".

Τέλος, για να ενεργοποιηθεί ο ειδοποιητικός μηχανισμός αποστολής συναγερμών μέσω του Prometheus Alertmanager και να συνδεθεί με το κανάλι #alerts του Slack, το αρχείο παραμετροποίησης "alertmanager.yml" θα πρέπει να τροποποιηθεί όπως διακρίνεται παρακάτω:

```
global:
 resolve_timeout: 1m
route:
  receiver: 'slack-notifications'

receivers:
- name: 'slack-notifications'
  slack_configs:
  - channel: '#alerts'
    api_url: 'https://hooks.slack.com/services/Here_Is_The_Secret_Webhook_URL'
    send_resolved: true
    icon_url: https://avatars3.githubusercontent.com/u/3380462
    title: |-
     [{{ .Status | toUpper }}{{ if eq .Status "firing" }}:{{ .Alerts.Firing | len }}{{ end }}] {{
.CommonLabels.alertname }} for {{ .CommonLabels.job }}
      {{- if gt (len .CommonLabels) (len .GroupLabels) -}}
        {{" "}}(
        {{- with .CommonLabels.Remove .GroupLabels.Names }}
          {{- range $index, $label := .SortedPairs -}}
            {{ if $index }}, {{ end }}
            {{- $label.Name }}="{{ $label.Value -}}"
          {{- end }}
        {{- end -}}
        )
```

```
    {{- end }}
  text: >-
    {{ range .Alerts -}}
    *Alert:* {{ .Annotations.title }}{{ if .Labels.severity }} - `{{ .Labels.severity }}`{{ end }}

    *Description:* {{ .Annotations.description }}

    *Details:*
      {{ range .Labels.SortedPairs }} • *{{ .Name }}:* `{{ .Value }}`
      {{ end }}
    {{ end }}
```

Σε αυτό το σημείο θα πρέπει να σημειωθεί ότι, το πεδίο “title” είναι προαιρετικό και αφορά στην καλύτερη παρουσίαση του κειμένου του μηνύματος ειδοποίησης στο Slack [55].

6.14: Εγκατάσταση του Alerta

Η εγκατάσταση του Alerta πραγματοποιείται σε ξεχωριστό εξυπηρετητή, ο οποίος θα φιλοξενεί και τη γραφική διεπαφή του Alerta. Αρχικά, θα πρέπει να εγκατασταθεί η βάση δεδομένων **MongoDB 4.0** ακολουθώντας τα παρακάτω βήματα.

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv
9DA31620334BD75D9DCB49F368818C72E52529D4

sudo echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu
bionic/mongodb-org/4.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.0.list

sudo apt-get update
sudo apt-get install -y mongodb-org
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση της MongoDB, αντίστοιχα.

```
sudo systemctl start mongod
sudo systemctl status mongod
sudo systemctl enable mongod
```

Για να είναι εφικτό να χρησιμοποιηθεί το Alerta, θα πρέπει να εγκατασταθούν τα dependencies της “Python 3”. Αυτό, πραγματοποιείται με τις επόμενες εντολές.

```
sudo apt-get install -y python3 python3-setuptools python3-pip python3-dev python3-venv
sudo apt-get install -y nginx uwsgi-plugin-python3
```

Στη συνέχεια, ώντας στον φάκελο “/opt”, εγκαθιστούμε το “alerta command-line tool”, σε ένα “Python 3 virtual environment”.

```
sudo  python3 -m venv alerta
sudo alerta/bin/pip install --upgrade pip wheel alerta-server alerta uwsgi
```

Έπειτα, για να μπορεί να γίνει χρήση μιας “web console”, απαιτούνται οι επόμενες εντολές.

```
sudo  wget
https://github.com/alerta/alerta-webui/releases/latest/download/alerta-webui.tar.gz
sudo tar zxvf alerta-webui.tar.gz
sudo cp dist /var/www/html
```

Έχοντας επιτελέσει τα παραπάνω επιτυχώς, επόμενο βήμα αποτελεί η δημιουργία ενός "wsgi" αρχείου python, και ενός "systemd" αρχείου που θα ελέγχει το αρχείο αυτό. Το WSGI (Web Server Gateway Interface) πρόκειται για ένα framework, το οποίο προωθεί τα αιτήματα ενός εξυπηρετητή ιστού, σε ένα backend Python web application [56].

Αρχικά, δημιουργούμε ένα αρχείο τύπου "wsgi" (python) και προσθέτουμε τις κατάλληλες ρυθμίσεις:

```
sudo vim /var/www/wsgi.py
from alerta import create_app
app = create_app()
```

Ύστερα, δημιουργούμε ένα αρχείο τύπου "ini", που αφορά στον uwsgi εξυπηρετητή και προσθέτουμε τις κατάλληλες ρυθμίσεις που θέτουν το Alerta API στο path /api, τα αρχεία καταγραφής στο Syslog, αλλά και ρυθμίζουν τη χρήση ενός UNIX socket για την επικοινωνία με τον Nginx αντίστοιχα.

```
sudo vim /etc/uwsgi.ini
```

```
[uwsgi]
chdir = /var/www
mount = /api=wsgi.py
callable = app
manage-script-name = true
env = BASE_URL=/api

master = true

processes = 5
logger = syslog:alertad

socket = /tmp/uwsgi.sock
chmod-socket = 664
uid = www-data
gid = www-data
vacuum = true

die-on-term = true
```

Στη συνέχεια, δημιουργούμε ένα αρχείο τύπου "service", το οποίο θα διαχειρίζεται τον uwsgi εξυπηρετητή:

```
sudo vim /etc/systemd/system/uwsgi.service
```

```
[Unit]
Description=uWSGI service

[Service]
ExecStart=/opt/alerta/bin/uwsgi --ini /etc/uwsgi.ini

[Install]
WantedBy=multi-user.target
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του uwsgi εξυπηρετητή, αντίστοιχα.

```
sudo service start uwsgi
sudo service status uwsgi
sudo service enable uwsgi
```

Σε αυτό το σημείο, θα πρέπει να τονιστεί, ότι στην παρούσα περίπτωση, χρησιμοποιείται ο Nginx ως εξυπηρετητής ιστού.

Γι' αυτόν τον λόγο, θα πρέπει να ρυθμιστεί το αρχείο "/etc/nginx/sites-enabled/default", ώστε να εξυπηρετεί το Alerta ως ένα "uWsgi application" στο path "/api".

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        location /api { try_files $uri @api; }
        location @api {
           include uwsgi_params;
           uwsgi_pass unix:/tmp/uwsgi.sock;
           proxy_set_header Host $host:$server_port;
           proxy_set_header X-Real-IP $remote_addr;
           proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        }

        location / {
              root /var/www/html;
        }
}
```

Τέλος, για να λάβουν χώρα οι αλλαγές, επανεκκινούμε τον Nginx και παραμετροποιούμε κατάλληλα το αρχείο “/var/www/html/config.json”, ώστε να οριστεί το endpoint στο “/api”:

```
sudo systemctl restart nginx
sudo vim /var/www/html/config.json
{"endpoint": "/api"}
```

6.15: Σύνδεση Alerta και Prometheus Alertmanager

Για να επιτευχθεί η σύνδεση του Prometheus Alertmanager και του Alerta, θα πρέπει να τροποποιηθεί το αρχείο παραμετροποίησης "alertmanager.yml" στον εξυπηρετητή που φιλοξενεί το Prometheus. Παρακάτω, διακρίνονται τα στοιχεία που προστέθηκαν για τον σκοπό αυτό.

```
route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 3m
  repeat_interval: 1m
  receiver: 'alerta'
receivers:
  - name: 'alerta'
    webhook_configs:
      - url: 'http://onos.telecom.ece.ntua.gr/api/webhooks/prometheus'
inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```

Σε αυτό το σημείο, για την κατανόηση της παραμετροποίησης του αρχείου, διευκρινίζονται κάποιες από τις μεταβλητές του.

- route: ορίζει έναν κόμβο στη ιεραρχία του routing tree.
- group_by: ορίζει την ομαδοποίηση των ειδοποιητικών συναγερμών με βάση την ετικέτα τους (label).
- group_wait: ο χρόνος αναμονής, έως ότου αποσταλεί μια ειδοποίηση για μια ομάδα συναγερμών. Η τιμή της συγκεκριμένης παραμέτρου κυμαίνεται συνήθως από 0 έως 30 δευτερόλεπτα.
- group_interval: ο χρόνος αναμονής, έως ότου αποσταλεί μια ειδοποίηση για νέους συναγερμούς, οι οποίοι προηγουμένως έχουν ομαδοποιηθεί σε ομάδες και έχουν σταλεί για αυτά ειδοποιήσεις.
- repeat_interval: ο χρόνος αναμονής, έως ότου αποσταλεί ξανά μια ειδοποίηση για την οποία έχει σταλεί προηγουμένως επιτυχώς κάποιος ειδοποιητικός συναγερμός.
- receiver: Ορίζει το όνομα του παραλήπτη.
- webhook_configs: επιτρέπει τον ορισμό ενός παραλήπτη με τη χρήση Webhook URL.
- inhibition_rules: κανόνες, οι οποίοι σιγούν ειδοποιητικούς συναγερμούς, που έχουν την ίδια ετικέτα στο πεδίο "equal".

6.16.a: Εγκατάσταση του Grafana

Στο παρόν κεφάλαιο, πραγματοποιείται μια παρουσίαση του τρόπου με τον οποίο εγκαθίσταται το λογισμικό Grafana, καθώς και ο τρόπος με τον οποίο χρησιμοποιείται μέσω HTTPS [57].

Ώντας στον εξυπηρετητή που επιθυμούμε να εγκαταστήσουμε το Grafana, εν προκειμένω στον "cn-monitor-1", στον οποίο φιλοξενείται και το Prometheus, εγκαθιστούμε το Grafana "GPG key" μέσω wget, το οποίο προσθέτει το key στην "APT installation list" από "trusted keys" και θα μας επιτρέψει να κατεβάσουμε το κατάλληλο πακέτο:

```
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```

Στην παραπάνω εντολή, η παράμετρος "-q" απενεργοποιεί το ειδοποιητικό μήνυμα της wget, ενώ η "-o" εκτυπώνει το αρχείο που μόλις εγκαταστάθηκε στη κονσόλα. Ύστερα, προσθέτουμε το Grafana repository στα "APT sources":

```
sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"
```

Στη συνέχεια, εγκαθιστούμε το κατάλληλο apt-package που μας ενδιαφέρει ως εξής:

```
sudo apt install grafana
```

Επιπλέον, ενεργοποιούμε τον Grafana εξυπηρετητή:

```
sudo systemctl start grafana-server
```

Για να ενεργοποιείται αυτόματα ο Grafana εξυπηρετητής κατά την εκκίνηση του συστήματος πληκτρολογούμε:

```
sudo systemctl enable grafana-server
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να πάψει, αλλά και να απεικονίσει την κατάσταση του Grafana εξυπηρετητή αντίστοιχα.

```
sudo systemctl stop grafana-server
sudo systemctl status grafana-server
```

6.16.b: Ενεργοποίηση HTTPS σε Grafana-σύνδεση με Prometheus

Έχοντας δημιουργήσει το κατάλληλο SSL πιστοποιητικό, αλλά και το κλειδί του πιστοποιητικού, για την καλύτερη διαχείριση των αρχείων αυτών, αντιγράφονται στον φάκελο "/etc/grafana" και προστίθενται τα κατάλληλα δικαιώματα ως εξής:

```
sudo cp  /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem /etc/grafana
sudo cp /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/privkey.pem /etc/grafana

sudo chown grafana:grafana fullchain.pem
sudo chown grafana:grafana privkey.pem
sudo chown 760 privkey.pem fullchain.pem
```

Στη συνέχεια, για την ενεργοποίηση του HTTPS στο Grafana, θα πρέπει να προστεθούν στο αρχείο "/etc/grafana/grafana.ini" τα εξής:

```
[server]
# Protocol (http, https, h2, socket)
protocol = https

root_url = https://cn-monitor-1.cn.ntua.gr:3000

# https certs & key file
cert_file = /etc/grafana/fullchain.pem
cert_key = /etc/grafana/privkey.pem
```

Από το παραπάνω αρχείο παρατηρεί κανείς, ότι το πρωτόκολλο που χρησιμοποιείται είναι το HTTPS, ενώ το "root_url" φέρει την διεύθυνση IP, καθώς και την θύρα στο οποίο θα είναι προσβάσιμο το Grafana.
Οι παράμετροι "cert_file" και "cert_key" ορίζουν το path για το SSL πιστοποιητικό, αλλά και το ιδιωτικό κλειδί αντίστοιχα.

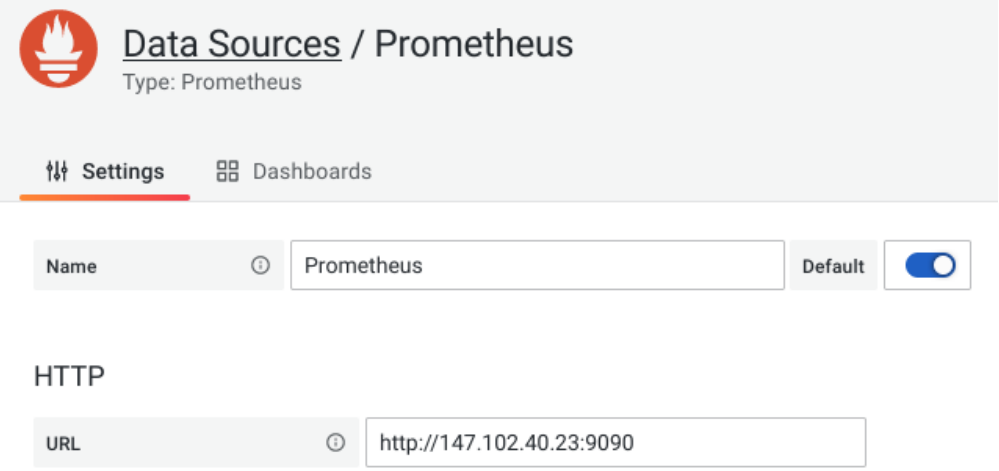
Στη συνέχεια, για να ενεργοποιηθούν οι παραπάνω αλλαγές, επανεκκινούμε το Grafana:

```
sudo systemctl restart grafana-server
```

Έχοντας ενεργοποιήσει το Grafana σε HTTPS, τίθεται ως διαθέσιμο στην διεύθυνση "https://cn-monitor-1.cn.ece.ntua.gr:3000". Στην login σελίδα που εμφανίζεται η προτροπή για την είσοδο, πληκτρολογούμε ως όνομα χρήστη: admin και κωδικό πρόσβασης: admin.

Μετά την είσοδο, κρίνεται απαραίτητη η δημιουργία ενός χρήστη με δικαιώματα διαχειριστή, αλλά και ενός ισχυρού κωδικού πρόσβασης. Αυτό, επιτυγχάνεται από το μενού με την επιλογή: “Configuration”⟶“Users” και στη συνέχεια επιλέγεται το κατάλληλο πεδίο για την δημιουργία χρήστη με δικαιώματα διαχειριστή.

Ακόμη, για να είναι προσβάσιμα τα δεδομένα του Prometheus, θα πρέπει να ρυθμιστεί η επικοινωνία μεταξύ Grafana και Prometheus. Αυτό επιτυγχάνεται μέσω της γραφικής διεπαφής του Grafana από το μενού στην επιλογή: “Configuration”⟶Data sources”⟶“Add data source”, όπου επιλέγουμε το Prometheus. Ύστερα, ορίζουμε το URL στο οποίο είναι προσβάσιμα τα δεδομένα του Prometheus, όπως διακρίνεται παρακάτω:



Στιγμιότυπο 6.16.b: Επιλογή του Prometheus ως “Data source” στο Grafana

6.17: Εγκατάσταση του Elasticsearch

Στο παρόν κεφάλαιο, πραγματοποιείται μια παρουσίαση του τρόπου με τον οποίο εγκαθίσταται το λογισμικό Elasticsearch, καθώς και ο τρόπος με τον οποίο χρησιμοποιείται μέσω HTTPS [58].

Καθότι τα στοιχεία που απαιτούνται για την εγκατάσταση του Elasticsearch δεν περιλαμβάνονται στα "default package repositories" του Ubuntu, θα πρέπει να προστεθούν, ώστε να γίνουν διαθέσιμα και στη συνέχεια να εγκατασταθεί η υπηρεσία μέσω APT.

Αυτό, διότι τα πακέτα που αφορούν στο Elasticsearch, απαιτούν ένα κλειδί, ώστε να αποφευχθεί οποιοδήποτε "package spoofing". Τα πακέτα τα οποία έχουν πιστοποιηθεί από αυτό το κλειδί, θεωρούνται ως έμπιστα από τον "package manager" του συστήματος.

Για το σκοπό αυτό, λαμβάνεται μέσω "curl" το "public GPG key" του Elasticsearch:

```
curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
```

Η επιλογή "-fsSL" σιγεί το progress και επιτρέπει αν χρειαστεί, να πραγματοποιηθεί κάποιο redirected request.

Στη συνέχεια, προσθέτουμε την λίστα με τα στοιχεία που απαιτούνται για την εγκατάσταση του Elasticsearch στο αρχείο "sources.list.d", από το οποίο το APT θα μπορεί πλέον να αναζητήσει για νέες πηγές πακέτων:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

Ύστερα, για να λάβουν χώρα οι αλλαγές, ενημερώνουμε την λίστα του APT ως εξής:

```
sudo apt update
```

Ομοίως με προηγούμενες εγκαταστάσεις, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install elasticsearch
```

Έχοντας εγκαταστήσει το Elasticsearch, θα πρέπει να παραμετροποιηθεί κατάλληλα το αρχείο "elasticsearch.yml" στον φάκελο "/etc/elasticsearch".

```
network.host: localhost
```

Η παραπάνω μεταβλητή στο αρχείο αυτό, ορίζει στο Elasticsearch να “ακούει” σε όλες τις διεπαφές.

Για να ενεργοποιείται αυτόματα το Elasticsearch κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable elasticsearch.service
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Elasticsearch, αντίστοιχα.

```
sudo systemctl start elasticsearch.service
sudo systemctl stop elasticsearch.service
sudo systemctl status elasticsearch.service
```

Αναφορικά με την χρήση του πρωτοκόλλου HTTPS για την κρυπτογράφηση της πληροφορίας δεν απαιτείται κάποια περαιτέρω ρύθμιση, ωστόσο θα πρέπει ο εξυπηρετητής ιστού που φιλοξενεί το Kibana να έχει SSL πιστοποιητικό. Η διαδικασία αυτή, αναλύεται στο κεφάλαιο 6.20.b, περί HTTPS σε Nginx.

6.18: Εγκατάσταση του Logstash

Για την εγκατάσταση του Logstash, ομοίως με προηγούμενες εγκαταστάσεις, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install logstash
```

Για να ενεργοποιείται αυτόματα το Logstash κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable logstash.service
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Logstash, αντίστοιχα.

```
sudo systemctl start logstash.service
sudo systemctl stop logstash.service
sudo systemctl status logstash.service
```

Στη συνέχεια, θα παραμετροποιηθεί κατάλληλα το Logstash, ώστε να επικοινωνεί με το Filebeat και το Elasticsearch. Για το σκοπό αυτό, αρχικά θα πρέπει να δημιουργηθεί ένα νέο αρχείο παραμετροποίησης αναφορικά με το Logstash.

Έχοντας δημιουργήσει το αρχείο με όνομα "/etc/logstash/conf.d/02-beats-input.conf", προσθέτουμε ως παραμέτρους τα παρακάτω:

```
input {
  beats {
    port => 5044
  }
}
```

Με αυτόν τον τρόπο, ορίζεται στο Logstash, να λαμβάνει δεδομένα από το Beats στην TCP θύρα "5044".

Αντίστοιχα, δημιουργώντας το αρχείο: "/etc/logstash/conf.d/30-elasticsearch-output.conf", προσθέτουμε στο αρχείο αυτό τα παρακάτω:

```
output {
  if [@metadata][pipeline] {
        elasticsearch {
        hosts => ["http://localhost:9200"]
        manage_template => false
        index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"
        pipeline => "%{[@metadata][pipeline]}"
        }
  } else {
        elasticsearch {
        hosts => ["http://localhost:9200"]
        manage_template => false
        index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"
        }
  }
}
```

Η παραπάνω παραμετροποίηση ορίζει στο Logstash να αποθηκεύει τα δεδομένα που λαμβάνει από το Beats (από την θύρα "5044"), στο Elasticsearch το οποίο ακούει στην θύρα "9200". Η επιτυχής ρύθμιση των παραπάνω βημάτων, μπορεί να ελεγχθεί με την εντολή:

```
sudo -u logstash /usr/share/logstash/bin/logstash --path.settings /etc/logstash -t
```

η οποία σε περίπτωση επιτυχούς παραμετροποίησης, αποτυπώνει το μήνυμα: **"**Config Validation Result: OK. Exiting Logstash".

Σε περίπτωση που αποτυπωθούν ειδοποιητικά μηνύματα σχετικά με το OpenJDK, μπορούν να αγνοηθούν, καθότι δεν προκαλούν κάποιο λειτουργικό πρόβλημα.

6.19: Εγκατάσταση του Filebeat

Για την εγκατάσταση του Filebeat, ομοίως με προηγούμενες εγκαταστάσεις, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install filebeat
```

Για να ενεργοποιείται αυτόματα το Filebeat κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable filebeat.service
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Filebeat αντίστοιχα.

```
sudo systemctl start filebeat.service
sudo systemctl stop filebeat.service
sudo systemctl status filebeat.service
```

Στη συνέχεια, θα παραμετροποιηθεί κατάλληλα το Filebeat, ώστε να επικοινωνεί με το Logstash. Για το σκοπό αυτό, θα πρέπει να τροποποιηθεί το αρχείο "/etc/filebeat/filebeat.yml". Πιο συγκεκριμένα, απαιτείται να προστεθούν τα παρακάτω:

```
output.logstash:
  # The Logstash hosts
  hosts: ["localhost:5044"]
```

Σε αυτό το σημείο, τονίζεται ότι το Filebeat υποστηρίζει διαφόρων τύπων modules [59], ωστόσο θα ενεργοποιηθεί μονάχα το module system.

```
sudo filebeat modules enable system
```

Στη συνέχεια θα πρέπει να ενεργοποιηθεί το pipeline του Filebeat, το οποίο αναλαμβάνει το "data parse", προτού μεταφέρει τα δεδομένα μέσω του Logstash στο Elasticsearch:

```
sudo filebeat setup --pipelines --modules system
```

Σε αυτό το βήμα, θα φορτωθεί το "index template" στο Elasticsearch. Ένα Elasticsearch "index" αποτελεί μια συλλογή από δεδομένα με όμοια χαρακτηριστικά.

Κάθε “index” διαφέρει με βάση το όνομά του, το οποίο χρησιμοποιείται σε χρήσεις που απαιτείται να εκτελεστεί κάποια λειτουργία. Το “index template” φορτώνεται ως εξής:

```
sudo filebeat setup --index-management -E output.logstash.enabled=false -E
'output.elasticsearch.hosts=["localhost:9200"]'
```

Το Filebeat περιλαμβάνει κάποια δείγματα από dashboards για το Kibana, τα οποία επιτρέπουν την οπτικοποίηση των δεδομένων μέσω του Kibana. Για να χρησιμοποιηθούν τα dashboards αυτά, θα πρέπει να δημιουργηθεί ένα “index pattern” και να ενσωματωθούν στο Kibana. Για να πραγματοποιηθεί αυτό, χρησιμοποιείται η παρακάτω εντολή:

```
sudo filebeat setup -E output.logstash.enabled=false -E
output.elasticsearch.hosts=['localhost:9200'] -E setup.kibana.host=localhost:5601
```

6.20.a: Εγκατάσταση του Kibana

Για την εγκατάσταση του Kibana, είναι απαραίτητο να έχει προηγηθεί η εγκατάσταση του Elasticsearch. Έχοντας εγκατεστημένο το Elasticsearch, εγκαθιστούμε το πακέτο που μας ενδιαφέρει ως εξής:

```
sudo apt install kibana
```

Για να ενεργοποιείται αυτόματα το Kibana κατά την εκκίνηση του συστήματος, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo systemctl enable kibana.service
```

Ύστερα, με τις παρακάτω εντολές, μπορεί κανείς να εκκινήσει, να πάψει, αλλά και να απεικονίσει την κατάσταση του Kibana, αντίστοιχα.

```
sudo systemctl start kibana.service
sudo systemctl stop kibana.service
sudo systemctl status kibana.service
```

6.20.b: Χρήση του Kibana μέσω HTTPS και reverse proxy

Σε αυτό το βήμα, θα πρέπει αρχικά να οριστεί μέσω του εργαλείου openssl, ένας Kibana χρήστης, ώστε να είναι προσβάσιμη η γραφική διεπαφή του Kibana μέσω αυθεντικοποίησης (Authentication):

```
echo "user:`openssl passwd -apr1`" | sudo tee -a /etc/nginx/htpasswd.users
```

Στην παραπάνω εντολή, όπου "user" ορίζουμε το όνομα χρήστη που επιθυμούμε και στην προτροπή που εμφανίζεται, πληκτρολογούμε τον επιθυμητό κωδικό πρόσβασης. Τα αποτελέσματα αποθηκεύονται στο αρχείο "/etc/nginx/htpasswd.users".

Στη συνέχεια, θα χρησιμοποιηθεί ο Nginx ως reverse proxy, ώστε η HTTP(S) κίνηση όταν πληκτρολογεί κανείς σε έναν web browser "https://cn-monitor-1.cn.ece.ntua.gr", να ανακατευθύνεται (redirect) εσωτερικά στην υπηρεσία του Kibana που "ακούει" στην θύρα "5601". Αυτό, επιτυγχάνεται παραμετροποιώντας κατάλληλα το αρχείο "/etc/nginx/sites-available/cn-monitor-1/cn-monitor-1.cn.ece.ntua.gr" ως εξής:

```
server {

    server_name cn-monitor-1.cn.ece.ntua.gr;
    auth_basic "Restricted Access";
    auth_basic_user_file /etc/nginx/htpasswd.users;

    location / {
        proxy_pass http://localhost:5601;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    listen 443 ssl; # managed by Certbot
            ssl_certificate    /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/fullchain.pem;    # managed by Certbot
          ssl_certificate_key  /etc/letsencrypt/live/cn-monitor-1.cn.ece.ntua.gr/privkey.pem;    # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
```

Στο παραπάνω αρχείο, η παράμετρος “auth_basic_user_file” ορίζει η πρόσβαση στη γραφική διεπαφή να γίνεται με Αυθεντικοποίηση και συγκεκριμένα με τα στοιχεία που βρίσκονται στο αρχείο “/etc/nginx/htpasswd.users” που δημιουργήθηκαν προηγουμένως.

Επιπλέον, η παράμετρος “proxy_pass” είναι εκείνη που ενεργοποιεί το reverse proxy, ώστε όλη η κίνηση να κατευθύνεται στην θύρα “5601”.

Όσον αφορά την παράμετρο “listen 443 ssl”, μέσω αυτής ορίζεται η κίνηση να πραγματοποιείται με χρήση του πρωτοκόλλου SSH. Οι υπόλοιπες εντολές που ακολουθούν στο αρχείο, έχουν ορισθεί αυτόματα από το Certbot κατά τη διαδικασία της δημιουργίας του SSL πιστοποιητικού.

Στη συνέχεια, δημιουργούμε ένα “symbolic link” για τον φάκελο “/etc/nginx/sites-enabled”, ώστε να μην χρειαστεί να προσθέσουμε και σε εκείνον τον φάκελο τις παραμετροποιήσεις που επιτελέσαμε.

```
sudo ln -s /etc/nginx/sites-available/cn-monitor-1.cn.ece.ntua.gr
/etc/nginx/sites-enabled/cn-monitor-1.cn.ece.ntua.gr
```

Τέλος, με χρήση της εντολής “sudo nginx-t”, είναι δυνατό να ελεγχθεί το αρχείο παραμετροποίησης του Nginx και σε περίπτωση που δεν υπάρχει κάποιο λάθος, για να λάβουν χώρα οι αλλαγές επανεκκινούμε τον Nginx.

```
sudo systemctl reload nginx
```

6.21. Παράδειγμα dashboard του Grafana σε JSON μορφή

Παρακάτω επισυνάπτεται ως παράδειγμα σε αρχείο τύπου JSON το dashboard: “Blackbox exporter” του Grafana. Στο αρχείο αυτό, περιλαμβάνονται όλες οι παραμετροποιήσεις που πραγματοποιήθηκαν στα panel του dashboard.


```
{
"annotations": {
  "list": [
    {
      "builtIn": 1,
      "datasource": "-- Grafana --",
      "enable": true,
      "hide": true,
      "iconColor": "rgba(0, 211, 255, 1)",
      "name": "Annotations & Alerts",
      "target": {
        "limit": 100,
        "matchAny": false,
        "tags": [],
        "type": "dashboard"
      },
      "type": "dashboard"
    }
  ]
},
"description": "",
"editable": true,
"fiscalYearStartMonth": 0,
"gnetId": 5345,
"graphTooltip": 0,
"id": 72,
"iteration": 1667035672056,
"links": [],
"liveNow": false,
"panels": [
  {
    "collapsed": false,
    "gridPos": {
      "h": 1,
      "w": 24,
      "x": 0,
      "y": 0
    },
    "id": 40,
    "panels": [],
    "title": "High level stats",
    "type": "row"
  },
  {
    "description": "Target of probing.",
    "fieldConfig": {
      "defaults": {
        "color": {
          "mode": "thresholds"
        },
        "mappings": [],
        "thresholds": {
          "mode": "absolute",
          "steps": [
            {
              "color": "green",
              "value": null
            },
            {
              "color": "red",
              "value": 80
            }
```

```
        ]
       }
      },
      "overrides": []
     },
     "gridPos": {
      "h": 4,
      "w": 9,
      "x": 0,
      "y": 1
     },
     "id": 2,
     "links": [],
     "options": {
      "colorMode": "value",
      "graphMode": "area",
      "justifyMode": "auto",
      "orientation": "auto",
      "reduceOptions": {
       "calcs": [
        "lastNotNull"
       ],
       "fields": "/^instance$/",
       "values": false
      },
      "textMode": "auto"
     },
     "pluginVersion": "8.4.4",
     "repeatDirection": "h",
     "targets": [
      {
       "datasource": {
        "type": "prometheus",
        "uid": "ljhTj8Enk"
       },
       "exemplar": true,
       "expr": "probe_success{instance=~\"$instance\"}",
       "format": "table",
       "instant": false,
       "interval": "5s",
       "intervalFactor": 1,
       "legendFormat": "",
       "refId": "A"
      }
     ],
     "title": "Target",
     "type": "stat"
    },
    {
     "description": "Indication whether the IP protocol is IPv4 or IPv6. The value 4 designates the IPv4 protocol and the value 6 the IPv6 protocol.",
     "fieldConfig": {
      "defaults": {
       "mappings": [],
       "thresholds": {
        "mode": "absolute",
        "steps": [
         {
          "color": "green",
          "value": null
         },
         {
          "color": "red",
          "value": 80
         }
        ]
       },
       "unit": "short"
      },
      "overrides": []
     },
     "gridPos": {
      "h": 4,
      "w": 6,
```

```
    "x": 9,
    "y": 1
   },
   "id": 30,
   "options": {
    "colorMode": "value",
    "graphMode": "none",
    "justifyMode": "auto",
    "orientation": "horizontal",
    "reduceOptions": {
     "calcs": [
      "lastNotNull"
     ],
     "fields": "",
     "values": false
    },
    "text": {},
    "textMode": "value_and_name"
   },
   "pluginVersion": "8.4.4",
   "targets": [
    {
     "datasource": {
      "type": "prometheus",
      "uid": "ljhTj8Enk"
     },
     "exemplar": true,
     "expr": "probe_ip_protocol{instance=~\"$instance\"}",
     "interval": "",
     "legendFormat": "IP version",
     "refId": "A"
    }
   ],
   "title": "IP Protocol",
   "type": "stat"
  },
  {
   "description": "Version of HTTP.",
   "fieldConfig": {
    "defaults": {
     "mappings": [],
     "thresholds": {
      "mode": "absolute",
      "steps": [
       {
        "color": "green",
        "value": null
       },
       {
        "color": "red",
        "value": 80
       }
      ]
     },
     "unit": "short"
    },
    "overrides": []
   },
   "gridPos": {
    "h": 4,
    "w": 5,
    "x": 15,
    "y": 1
   },
   "id": 28,
   "options": {
    "colorMode": "value",
    "graphMode": "none",
    "justifyMode": "auto",
    "orientation": "auto",
    "reduceOptions": {
     "calcs": [
      "lastNotNull"
```

```
      ],
      "fields": "",
      "values": false
    },
    "textMode": "value_and_name"
  },
  "pluginVersion": "8.4.4",
  "targets": [
    {
      "datasource": {
        "type": "prometheus",
        "uid": "ljhTj8Enk"
      },
      "exemplar": true,
      "expr": "probe_http_version{instance=~\"$instance\"}",
      "instant": false,
      "interval": "",
      "legendFormat": "Version",
      "refId": "A"
    }
  ],
  "title": "HTTP Version",
  "type": "stat"
},
{
  "description": "Response HTTP status code.",
  "fieldConfig": {
    "defaults": {
      "color": {
        "mode": "thresholds"
      },
      "decimals": 0,
      "mappings": [
        {
          "options": {
            "0": {
              "text": "NO"
            },
            "1": {
              "text": "YES"
            }
          },
          "type": "value"
        },
        {
          "options": {
            "match": "null",
            "result": {
              "text": "N/A"
            }
          },
          "type": "special"
        }
      ],
      "thresholds": {
        "mode": "absolute",
        "steps": [
          {
            "color": "#299c46",
            "value": null
          },
          {
            "color": "orange",
            "value": 200
          },
          {
            "color": "#d44a3a",
            "value": 299
          }
        ]
      },
      "unit": "none"
    },
```

```
    "overrides": []
   },
   "gridPos": {
    "h": 4,
    "w": 4,
    "x": 20,
    "y": 1
   },
   "id": 20,
   "links": [],
   "maxDataPoints": 100,
   "options": {
    "colorMode": "none",
    "graphMode": "none",
    "justifyMode": "auto",
    "orientation": "horizontal",
    "reduceOptions": {
     "calcs": [
      "lastNotNull"
     ],
     "fields": "",
     "values": false
    },
    "textMode": "auto"
   },
   "pluginVersion": "8.4.4",
   "repeatDirection": "h",
   "targets": [
    {
     "datasource": {
      "type": "prometheus",
      "uid": "ljhTj8Enk"
     },
     "exemplar": true,
     "expr": "probe_http_status_code{instance=~\"$instance\"}",
     "format": "time_series",
     "interval": "",
     "intervalFactor": 1,
     "legendFormat": "",
     "refId": "A"
    }
   ],
   "title": "HTTP Status Code",
   "type": "stat"
  },
  {
   "collapsed": true,
   "gridPos": {
    "h": 1,
    "w": 24,
    "x": 0,
    "y": 5
   },
   "id": 42,
   "panels": [
    {
     "description": "Indicates if SSL was used for the final redirect.",
     "fieldConfig": {
      "defaults": {
       "color": {
        "mode": "thresholds"
       },
       "mappings": [
        {
         "options": {
          "0": {
           "text": "NO"
          },
          "1": {
           "text": "YES"
          }
         },
         "type": "value"
```

```
      },
      {
       "options": {
        "match": "null",
        "result": {
         "text": "N/A"
        }
       },
       "type": "special"
      }
     ],
     "thresholds": {
      "mode": "absolute",
      "steps": [
       {
        "color": "#d44a3a"
       },
       {
        "color": "rgba(237, 129, 40, 0.89)",
        "value": 0
       },
       {
        "color": "#299c46",
        "value": 1
       }
      ]
     },
     "unit": "none"
    },
    "overrides": []
   },
   "gridPos": {
    "h": 4,
    "w": 11,
    "x": 0,
    "y": 6
   },
   "id": 18,
   "links": [],
   "maxDataPoints": 100,
   "options": {
    "colorMode": "value",
    "graphMode": "none",
    "justifyMode": "auto",
    "orientation": "horizontal",
    "reduceOptions": {
     "calcs": [
      "lastNotNull"
     ],
     "fields": "",
     "values": false
    },
    "textMode": "auto"
   },
   "pluginVersion": "8.4.4",
   "repeatDirection": "h",
   "targets": [
    {
     "datasource": {
      "type": "prometheus",
      "uid": "ljhTj8Enk"
     },
     "exemplar": true,
     "expr": "probe_http_ssl{instance=~\"$instance\"}",
     "format": "time_series",
     "interval": "",
     "intervalFactor": 1,
     "legendFormat": "",
     "refId": "A"
    }
   ],
   "title": "SSL",
   "type": "stat"
```

```
},
{
 "description": "Earliest SSL cert expiry.",
 "fieldConfig": {
  "defaults": {
   "color": {
    "mode": "thresholds"
   },
   "decimals": 2,
   "mappings": [
    {
     "options": {
      "0": {
       "text": "NO"
      },
      "1": {
       "text": "YES"
      }
     },
     "type": "value"
    },
    {
     "options": {
      "match": "null",
      "result": {
       "text": "N/A"
      }
     },
     "type": "special"
    }
   ],
   "thresholds": {
    "mode": "absolute",
    "steps": [
     {
      "color": "#d44a3a"
     },
     {
      "color": "rgba(237, 129, 40, 0.89)",
      "value": 0
     },
     {
      "color": "#299c46",
      "value": 1209600
     }
    ]
   },
   "unit": "dtdurations"
  },
  "overrides": []
 },
 "gridPos": {
  "h": 4,
  "w": 13,
  "x": 11,
  "y": 6
 },
 "id": 19,
 "links": [],
 "maxDataPoints": 100,
 "options": {
  "colorMode": "none",
  "graphMode": "none",
  "justifyMode": "auto",
  "orientation": "horizontal",
  "reduceOptions": {
   "calcs": [
    "lastNotNull"
   ],
   "fields": "",
   "values": false
  },
  "textMode": "auto"
```

```
      },
      "pluginVersion": "8.4.4",
      "repeatDirection": "h",
      "targets": [
       {
        "datasource": {
         "type": "prometheus",
         "uid": "ljhTj8Enk"
        },
        "exemplar": true,
        "expr": "probe_ssl_earliest_cert_expiry{instance=~\"$instance\"}-time()",
        "format": "time_series",
        "interval": "",
        "intervalFactor": 1,
        "legendFormat": "",
        "refId": "A"
       }
      ],
      "title": "SSL Cert Expiry",
      "type": "stat"
     }
    ],
    "title": "SSL",
    "type": "row"
   },
   {
    "collapsed": true,
    "gridPos": {
     "h": 1,
     "w": 24,
     "x": 0,
     "y": 6
    },
    "id": 48,
    "panels": [
     {
      "description": "Duration of HTTP request by phase, summed over all redirects.",
      "fieldConfig": {
       "defaults": {
        "color": {
         "mode": "palette-classic"
        },
        "custom": {
         "axisLabel": "Seconds (s)",
         "axisPlacement": "auto",
         "barAlignment": 0,
         "drawStyle": "line",
         "fillOpacity": 10,
         "gradientMode": "hue",
         "hideFrom": {
          "legend": false,
          "tooltip": false,
          "viz": false
         },
         "lineInterpolation": "linear",
         "lineWidth": 1,
         "pointSize": 5,
         "scaleDistribution": {
          "type": "linear"
         },
         "showPoints": "auto",
         "spanNulls": true,
         "stacking": {
          "group": "A",
          "mode": "none"
         },
         "thresholdsStyle": {
          "mode": "off"
         }
        },
        "mappings": [],
        "thresholds": {
         "mode": "absolute",
```

```
          "steps": [
            {
              "color": "green"
            },
            {
              "color": "red",
              "value": 80
            }
          ]
        },
        "unit": "s"
      },
      "overrides": []
    },
    "gridPos": {
      "h": 10,
      "w": 24,
      "x": 0,
      "y": 7
    },
    "id": 38,
    "options": {
      "legend": {
        "calcs": [
          "mean",
          "lastNotNull",
          "max",
          "min"
        ],
        "displayMode": "table",
        "placement": "right"
      },
      "tooltip": {
        "mode": "single",
        "sort": "none"
      }
    },
    "targets": [
      {
        "datasource": {
          "type": "prometheus",
          "uid": "ljhTj8Enk"
        },
        "exemplar": true,
        "expr": "probe_http_duration_seconds{instance=~\"$instance\"}",
        "interval": "",
        "legendFormat": "Duration: Phase: {{phase}}",
        "refId": "A"
      }
    ],
    "title": "HTTP request duration by phase",
    "type": "timeseries"
  }
 ],
 "title": "Requests",
 "type": "row"
},
{
 "collapsed": true,
 "gridPos": {
  "h": 1,
  "w": 24,
  "x": 0,
  "y": 7
 },
 "id": 46,
 "panels": [
  {
   "description": "The number of HTTP redirects.",
   "fieldConfig": {
    "defaults": {
     "color": {
      "mode": "palette-classic"
```

```
    },
    "custom": {
      "axisLabel": "",
      "axisPlacement": "auto",
      "barAlignment": 0,
      "drawStyle": "line",
      "fillOpacity": 10,
      "gradientMode": "hue",
      "hideFrom": {
        "legend": false,
        "tooltip": false,
        "viz": false
      },
      "lineInterpolation": "linear",
      "lineWidth": 1,
      "pointSize": 5,
      "scaleDistribution": {
        "type": "linear"
      },
      "showPoints": "auto",
      "spanNulls": true,
      "stacking": {
        "group": "A",
        "mode": "none"
      },
      "thresholdsStyle": {
        "mode": "off"
      }
    },
    "mappings": [],
    "thresholds": {
      "mode": "absolute",
      "steps": [
        {
          "color": "green"
        },
        {
          "color": "red",
          "value": 80
        }
      ]
    }
  },
  "overrides": []
},
"gridPos": {
  "h": 9,
  "w": 24,
  "x": 0,
  "y": 8
},
"id": 26,
"options": {
  "legend": {
    "calcs": [
      "mean",
      "lastNotNull",
      "max",
      "min"
    ],
    "displayMode": "table",
    "placement": "right"
  },
  "tooltip": {
    "mode": "single",
    "sort": "none"
  }
},
"pluginVersion": "8.4.4",
"targets": [
  {
    "datasource": {
      "type": "prometheus",
```

```
          "uid": "ljhTj8Enk"
        },
        "exemplar": true,
        "expr": "probe_http_redirects{instance=~\"$instance\"}",
        "format": "time_series",
        "interval": "",
        "legendFormat": "Redirects:",
        "refId": "A"
       }
     ],
     "title": "HTTP redirects",
     "type": "timeseries"
    }
   ],
   "title": "Redirects",
   "type": "row"
  },
  {
   "collapsed": true,
   "gridPos": {
    "h": 1,
    "w": 24,
    "x": 0,
    "y": 8
   },
   "id": 50,
   "panels": [
    {
     "description": "Boolean value which indicates if the probe was a success. The result 1 (true) designates successful probe.",
     "fieldConfig": {
      "defaults": {
       "color": {
        "mode": "palette-classic"
       },
       "custom": {
        "axisLabel": "",
        "axisPlacement": "auto",
        "barAlignment": 0,
        "drawStyle": "line",
        "fillOpacity": 10,
        "gradientMode": "hue",
        "hideFrom": {
         "legend": false,
         "tooltip": false,
         "viz": false
        },
        "lineInterpolation": "linear",
        "lineWidth": 1,
        "pointSize": 5,
        "scaleDistribution": {
         "type": "linear"
        },
        "showPoints": "auto",
        "spanNulls": true,
        "stacking": {
         "group": "A",
         "mode": "none"
        },
        "thresholdsStyle": {
         "mode": "off"
        }
       },
       "mappings": [],
       "max": 1,
       "min": 0,
       "thresholds": {
        "mode": "absolute",
        "steps": [
         {
          "color": "green"
         },
         {
          "color": "red",
```

```
          "value": 80
        }
       ]
      }
     },
     "overrides": []
    },
    "gridPos": {
     "h": 8,
     "w": 12,
     "x": 0,
     "y": 9
    },
    "id": 32,
    "options": {
     "legend": {
      "calcs": [
       "lastNotNull"
      ],
      "displayMode": "table",
      "placement": "right"
     },
     "tooltip": {
      "mode": "single",
      "sort": "none"
     }
    },
    "targets": [
     {
      "datasource": {
       "type": "prometheus",
       "uid": "ljhTj8Enk"
      },
      "exemplar": true,
      "expr": "probe_success{instance=~\"$instance\"}",
      "interval": "",
      "legendFormat": "Probe success:",
      "refId": "A"
     }
    ],
    "title": "Probe success (boolean)",
    "type": "timeseries"
   },
   {
    "description": "Boolean value which indicates if the probe failed due to regex. The result 1 (true) designates unsuccessful probe due to regex.",
    "fieldConfig": {
     "defaults": {
      "color": {
       "mode": "palette-classic"
      },
      "custom": {
       "axisLabel": "",
       "axisPlacement": "auto",
       "barAlignment": 0,
       "drawStyle": "line",
       "fillOpacity": 10,
       "gradientMode": "hue",
       "hideFrom": {
        "legend": false,
        "tooltip": false,
        "viz": false
       },
       "lineInterpolation": "linear",
       "lineWidth": 1,
       "pointSize": 5,
       "scaleDistribution": {
        "type": "linear"
       },
       "showPoints": "auto",
       "spanNulls": true,
       "stacking": {
        "group": "A",
        "mode": "none"
```

```
      },
      "thresholdsStyle": {
       "mode": "off"
      }
     },
     "mappings": [],
     "thresholds": {
      "mode": "absolute",
      "steps": [
       {
        "color": "green"
       },
       {
        "color": "red",
        "value": 80
       }
      ]
     }
    },
    "overrides": []
   },
   "gridPos": {
    "h": 8,
    "w": 12,
    "x": 12,
    "y": 9
   },
   "id": 34,
   "options": {
    "legend": {
     "calcs": [
      "mean",
      "lastNotNull",
      "max",
      "min"
     ],
     "displayMode": "table",
     "placement": "right"
    },
    "tooltip": {
     "mode": "single",
     "sort": "none"
    }
   },
   "targets": [
    {
     "datasource": {
      "type": "prometheus",
      "uid": "ljhTj8Enk"
     },
     "exemplar": true,
     "expr": "probe_failed_due_to_regex{instance=~\"$instance\"}",
     "interval": "",
     "legendFormat": "Probe regex failure:",
     "refId": "A"
    }
   ],
   "title": "Probe failure due to regex",
   "type": "timeseries"
  },
  {
   "description": "Duration in seconds of the probe completion.",
   "fieldConfig": {
    "defaults": {
     "color": {
      "mode": "palette-classic"
     },
     "custom": {
      "axisLabel": "Seconds (s)",
      "axisPlacement": "auto",
      "barAlignment": 0,
      "drawStyle": "line",
      "fillOpacity": 10,
```

```
      "gradientMode": "hue",
      "hideFrom": {
        "legend": false,
        "tooltip": false,
        "viz": false
      },
      "lineInterpolation": "linear",
      "lineWidth": 1,
      "pointSize": 5,
      "scaleDistribution": {
        "type": "linear"
      },
      "showPoints": "auto",
      "spanNulls": true,
      "stacking": {
        "group": "A",
        "mode": "none"
      },
      "thresholdsStyle": {
        "mode": "off"
      }
    },
    "mappings": [],
    "thresholds": {
      "mode": "absolute",
      "steps": [
        {
          "color": "green"
        },
        {
          "color": "red",
          "value": 80
        }
      ]
    },
    "unit": "s"
  },
  "overrides": []
},
"gridPos": {
  "h": 9,
  "w": 24,
  "x": 0,
  "y": 17
},
"id": 17,
"links": [],
"options": {
  "legend": {
    "calcs": [
      "mean",
      "lastNotNull",
      "max",
      "min"
    ],
    "displayMode": "table",
    "placement": "right"
  },
  "tooltip": {
    "mode": "multi",
    "sort": "none"
  }
},
"pluginVersion": "8.4.4",
"targets": [
  {
    "datasource": {
      "type": "prometheus",
      "uid": "ljhTj8Enk"
    },
    "exemplar": true,
    "expr": "probe_duration_seconds{instance=~\"$instance\"}",
    "format": "time_series",
```

```
      "interval": "",
      "intervalFactor": 1,
      "legendFormat": "Duration:",
      "refId": "A"
     }
    ],
    "title": "Probe Duration",
    "type": "timeseries"
   },
   {
    "description": "Length of HTTP content response.",
    "fieldConfig": {
     "defaults": {
      "color": {
       "mode": "palette-classic"
      },
      "custom": {
       "axisLabel": "",
       "axisPlacement": "auto",
       "barAlignment": 0,
       "drawStyle": "line",
       "fillOpacity": 10,
       "gradientMode": "hue",
       "hideFrom": {
        "legend": false,
        "tooltip": false,
        "viz": false
       },
       "lineInterpolation": "linear",
       "lineStyle": {
        "fill": "solid"
       },
       "lineWidth": 1,
       "pointSize": 5,
       "scaleDistribution": {
        "type": "linear"
       },
       "showPoints": "auto",
       "spanNulls": true,
       "stacking": {
        "group": "A",
        "mode": "none"
       },
       "thresholdsStyle": {
        "mode": "off"
       }
      },
      "mappings": [],
      "thresholds": {
       "mode": "absolute",
       "steps": [
        {
         "color": "green"
        },
        {
         "color": "red",
         "value": 80
        }
       ]
      }
     },
     "overrides": []
    },
    "gridPos": {
     "h": 11,
     "w": 24,
     "x": 0,
     "y": 26
    },
    "id": 36,
    "options": {
     "legend": {
      "calcs": [
```

```
          "mean",
          "lastNotNull",
          "max",
          "min"
         ],
         "displayMode": "table",
         "placement": "right"
        },
        "tooltip": {
         "mode": "single",
         "sort": "none"
        }
       },
       "targets": [
        {
         "datasource": {
          "type": "prometheus",
          "uid": "ljhTj8Enk"
         },
         "exemplar": true,
         "expr": "probe_http_content_length{instance=~\"$instance\"}",
         "interval": "",
         "legendFormat": "Length:",
         "refId": "A"
        }
       ],
       "title": "HTTP content response length",
       "type": "timeseries"
      }
     ],
     "title": "HTTP probing",
     "type": "row"
    },
    {
     "collapsed": false,
     "gridPos": {
      "h": 1,
      "w": 24,
      "x": 0,
      "y": 9
     },
     "id": 44,
     "panels": [],
     "title": "DNS",
     "type": "row"
    },
    {
     "description": "Duration in seconds of the completion of the DNS Lookup.",
     "fieldConfig": {
      "defaults": {
       "color": {
        "mode": "palette-classic"
       },
       "custom": {
        "axisLabel": "Seconds (s)",
        "axisPlacement": "auto",
        "barAlignment": 0,
        "drawStyle": "line",
        "fillOpacity": 10,
        "gradientMode": "hue",
        "hideFrom": {
         "legend": false,
         "tooltip": false,
         "viz": false
        },
        "lineInterpolation": "linear",
        "lineWidth": 1,
        "pointSize": 5,
        "scaleDistribution": {
         "type": "linear"
        },
        "showPoints": "never",
        "spanNulls": true,
```

```
        "stacking": {
          "group": "A",
          "mode": "none"
        },
        "thresholdsStyle": {
          "mode": "off"
        }
      },
      "mappings": [],
      "thresholds": {
        "mode": "absolute",
        "steps": [
          {
            "color": "green",
            "value": null
          },
          {
            "color": "red",
            "value": 80
          }
        ]
      },
      "unit": "s"
    },
    "overrides": []
  },
  "gridPos": {
    "h": 9,
    "w": 24,
    "x": 0,
    "y": 10
  },
  "id": 21,
  "links": [],
  "options": {
    "legend": {
      "calcs": [
        "mean",
        "lastNotNull",
        "max",
        "min"
      ],
      "displayMode": "table",
      "placement": "right"
    },
    "tooltip": {
      "mode": "multi",
      "sort": "none"
    }
  },
  "pluginVersion": "8.4.4",
  "targets": [
    {
      "datasource": {
        "type": "prometheus",
        "uid": "ljhTj8Enk"
      },
      "exemplar": true,
      "expr": "probe_dns_lookup_time_seconds{instance=~\"$instance\"}",
      "format": "time_series",
      "interval": "",
      "intervalFactor": 1,
      "legendFormat": "Duration:",
      "refId": "A"
    }
  ],
  "title": "DNS Lookup",
  "type": "timeseries"
 }
],
"refresh": false,
"schemaVersion": 35,
"style": "dark",
```

```
"tags": [
  "blackbox",
  "prometheus"
],
"templating": {
  "list": [
    {
      "current": {
        "selected": false,
        "text": "https://courses.cn.ntua.gr",
        "value": "https://courses.cn.ntua.gr"
      },
      "hide": 0,
      "includeAll": false,
      "label": "site",
      "multi": false,
      "name": "instance",
      "options": [
        {
          "selected": false,
          "text": "https://helios.ntua.gr",
          "value": "https://helios.ntua.gr"
        },
        {
          "selected": true,
          "text": "https://courses.cn.ntua.gr",
          "value": "https://courses.cn.ntua.gr"
        },
        {
          "selected": false,
          "text": "https://www.ece.ntua.gr",
          "value": "https://www.ece.ntua.gr"
        },
        {
          "selected": false,
          "text": "https://www.cn.ntua.gr",
          "value": "https://www.cn.ntua.gr"
        },
        {
          "selected": false,
          "text": "https://students.ece.ntua.gr",
          "value": "https://students.ece.ntua.gr"
        },
        {
          "selected": false,
          "text": "http://users.ntua.gr",
          "value": "http://users.ntua.gr"
        },
        {
          "selected": false,
          "text": "http://onos.telecom.ece.ntua.gr",
          "value": "http://onos.telecom.ece.ntua.gr"
        },
        {
          "selected": false,
          "text": "https://cn-monitor-1.cn.ece.ntua.gr:3000",
          "value": "https://cn-monitor-1.cn.ece.ntua.gr:3000"
        },
        {
          "selected": false,
          "text": "http://cn-monitor-1.cn.ece.ntua.gr:9090",
          "value": "http://cn-monitor-1.cn.ece.ntua.gr:9090"
        },
        {
          "selected": false,
          "text": "https://bbb2.cn.ntua.gr",
          "value": "https://bbb2.cn.ntua.gr"
        },
        {
          "selected": false,
          "text": "https://bbb.cn.ntua.gr",
          "value": "https://bbb.cn.ntua.gr"
        }
```

```
      ],
      "query": "https://helios.ntua.gr, https://courses.cn.ntua.gr, https://www.ece.ntua.gr, https://www.cn.ntua.gr, https://students.ece.ntua.gr, http://users.ntua.gr,
http://onos.telecom.ece.ntua.gr, https://cn-monitor-1.cn.ece.ntua.gr:3000, http://cn-monitor-1.cn.ece.ntua.gr:9090,  https://bbb2.cn.ntua.gr,  https://bbb.cn.ntua.gr",
      "queryValue": "",
      "skipUrlSync": false,
      "type": "custom"
    }
  ]
 },
 "time": {
  "from": "now-1h",
  "to": "now"
 },
 "timepicker": {
  "refresh_intervals": [
   "5s",
   "10s",
   "30s",
   "1m",
   "5m",
   "15m",
   "30m",
   "1h",
   "2h",
   "1d"
  ],
  "time_options": [
   "5m",
   "15m",
   "1h",
   "6h",
   "12h",
   "24h",
   "2d",
   "7d",
   "30d"
  ]
 },
 "timezone": "",
 "title": "Blackbox Exporter",
 "uid": "xtkCtBkiz2",
 "version": 91,
 "weekStart": ""
}
```

Επίλογος

Σύνοψη

Σκοπός της παρούσας διπλωματικής εργασίας αποτελεί η παρακολούθηση και επίβλεψη της διαθεσιμότητας και των επιδόσεων ενός κέντρου δεδομένων. Η τοπολογία του δικτύου περιλαμβάνει ποικίλες δικτυακές συσκευές, όπως είναι εξυπηρετητές τύπου UNIX, εξυπηρετητές ιστού, εξυπηρετητές που φιλοξενούν λογισμικά, όπως είναι το BigBlueButton, το Gitlab, αλλά και εξυπηρετητές και εικονικές μηχανές της σουίτας VMware, όπως είναι οι ESXi hosts. Η διαδικασία της παρακολούθησης (monitoring) που ακολουθείται, αποτελείται, αφενός από την συλλογή και απεικόνιση μετρικών και αφετέρου από την συγκέντρωση και οπτικοποίηση αρχείων καταγραφής.
Σε επόμενο βήμα, υποστηρίζεται και η δυνατότητα της αποστολής ειδοποιητικών συναγερμών (Alerting), με την οποία οι διαχειριστές του συστήματος είναι σε θέση να ενημερώνονται εγκαίρως μέσω ειδοποιητικών μηνυμάτων για οποιαδήποτε αστοχία έχει επέλθει.
Η χρήση από τη μία πλευρά του Prometheus ως Time Series βάση δεδομένων και από την άλλη πλευρά του Elasticsearch ως document oriented βάση δεδομένων, καθιστά την αποθήκευση και προσβασιμότητα των μετρικών και των αρχείων καταγραφής αντίστοιχα, εύχρηστη και ευέλικτη. Αυτό, διότι τα δεδομένα όλων των συσκευών προς παρακολούθηση είναι οργανωμένα και συγκεντρωμένα σε έναν κόμβο, με αποτέλεσμα να είναι προσβάσιμα μέσω αυτού. Ομοίως, η οπτικοποίηση και ανάλυση της πληροφορίας καθίσταται άμεσα διαθέσιμη μέσω της γραφικής διεπαφής του Grafana όσον αφορά τις μετρικές, αλλά και του Kibana αναφορικά με τα αρχεία καταγραφής.

Συμπεράσματα-Μελλοντικές επεκτάσεις

Όπως γίνεται αντιληπτό, η δημιουργία μιας οργανωμένης τοπολογίας που θα είναι σε επικοινωνία με ποικιλόμορφες δικτυακές συσκευές, αποτελεί ιδιαίτερο πλεονέκτημα στην βέλτιστη αξιοποίηση των πόρων μιας υποδομής. Είναι γεγονός, ότι στις μέρες μας αυξάνεται όλο και περισσότερο η ανάγκη για τον έλεγχο και την διαχείριση των πληροφοριών.
Με την υλοποίηση που υποδεικνύεται στην παρούσα διπλωματική εργασία, καθίσταται ευέλικτη η κλιμάκωση μιας τοπολογίας. Αφενός, σε περίπτωση προσθήκης νέων συσκευών, απαιτείται μονάχα η σύνδεση αυτών με τις βάσεις δεδομένων που θα αποθηκεύουν τα νέα δεδομένα, εν προκειμένω το Prometheus και το Elasticsearch. Αφετέρου, η οπτικοποίηση της πληροφορίας αναλαμβάνεται άμεσα από το Grafana και το Kibana.
Σε μελλοντική χρήση, στα πλαίσια της κυβερνοασφάλειας, θα μπορούσε να πραγματοποιηθεί τόσο η ανάλυση αρχείων καταγραφής, όσο και μετρικών, με απώτερο σκοπό την έγκαιρη αναγνώριση cyber attacks, όπως είναι οι DDOS (Distributed Denial of Service). Αυτό, θα μπορούσε να επιτευχθεί και σε πραγματικό χρόνο, με την χρήση ειδοποιητικών συναγερμών, οι οποίοι θα προειδοποιούν για οποιαδήποτε ασυνήθιστη και απότομη σε σχέση με τη μέση τιμή, μεταβολή.

Παραπομπές

[1] Prometheus, “Prometheus metric types”. [Online].
Available: https://prometheus.io/docs/concepts/metric_types/
[Accessed October 2022].

[2] Vivek Sharma, “Managing Multi-Cloud Deployments on Kubernetes with Istio, Prometheus and Grafana”, In proc. 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2022. [Online].
Available: https://ieeexplore.ieee.org/document/9785124
[Accessed October 2022].

[3] Alessandro Di Stefano *et al.*, “Prometheus and AIOps for the orchestration of Cloud-native applications in Ananke”, In proc. 2021 IEEE 30th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Bayonne, France, 2021. [Online].
Available: https://ieeexplore.ieee.org/document/9680514
[Accessed October 2022].

[4] Lei Chen *et al.*, “Monitoring System of OpenStack Cloud Platform Based on Prometheus”, In proc. 2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL), Chongqing, China, 2020. [Online].
Available: https://ieeexplore.ieee.org/document/9270544
[Accessed October 2022].

[5] Bin Lv, *et al.*, “Network Traffic Monitoring System Based on Big Data Technology”, In proc. International Conference on Big Data and Computing (ICBDC), 2018. [Online].
Available: https://www.researchgate.net/publication/326499385
[Accessed October 2022].

[6] Prometheus, “What is Prometheus?”. [Online].
Available: https://prometheus.io/docs/introduction/overview/#what-is-prometheus
[Accessed October 2022].

[7] Elastic, “What is Elasticsearch”. [Online].
Available: https://www.elastic.co/what-is/elasticsearch
[Accessed October 2022].

[8] Prometheus, “What are metrics?”. [Online].
Available: https://prometheus.io/docs/introduction/overview/#what-are-metrics
[Accessed October 2022].

[9] Prometheus, “Comparison to alternatives”. [Online].
Available: https://prometheus.io/docs/introduction/comparison/
[Accessed October 2022].

[10] Timescale, “How TimescaleDB works”. [Online].
Available: https://www.timescale.com/products#how-it-works
[Accessed October 2022].

[11] Prometheus, “Text-based format”. [Online].
Available: https://prometheus.io/docs/instrumenting/exposition_formats/#text-based-format
[Accessed October 2022].

[12] Prometheus, “Client libraries”. [Online].
Available: https://prometheus.io/docs/instrumenting/clientlibs/
[Accessed October 2022].

[13] Prometheus, “Metric types”. [Online].
Available: https://prometheus.io/docs/concepts/metric_types/
[Accessed October 2022].

[14] Brian Brazil, *Prometheus: Up & Running, INFRASTRUCTURE AND APPLICATION PERFORMANCE MONITORING*. Gravenstein Highway North, Sebastopol, CA: O’Reilly Media, 2018. [Online].
Available: https://www.oreilly.com/library/view/prometheus-up/9781492034131/
[Accessed October 2022].

[15] Prometheus, “Querying Prometheus”. [Online].
Available: https://prometheus.io/docs/prometheus/latest/querying/basics/#querying-prometheus
[Accessed October 2022].

[16] Prometheus, “Monitoring Linux host metrics with the Node Exporter”. [Online].
Available: https://prometheus.io/docs/guides/node-exporter/#monitoring-with-node-exporter
[Accessed October 2022].

[17] Alan Storm, “What are Prometheus Exporters?”, May, 12, 2020. [Online].
Available: https://alanstorm.com/what-are-prometheus-exporters/
[Accessed October 2022].

[18] Nancy Chauhan, “Building a Prometheus Exporter”, May, 3, 2021. [Online].
Available: https://levelup.gitconnected.com/building-a-prometheus-exporter-8a4bbc3825f5
[Accessed October 2022].

[19] Github project, “Windows exporter”, Aug, 21, 2016. [Source code].
Available: https://github.com/prometheus-community/windows_exporter
[Accessed October 2022].

[20] BigBlueButton Exporter, “Overview of BigBlueButton Exporter”. [Online].
Available: https://bigbluebutton-exporter.greenstatic.dev/
[Accessed October 2022].

[21] Github project, Prometheus, “Blackbox exporter”. [Source code].
Available: https://github.com/prometheus/blackbox_exporter
[Accessed October 2022].

[22] Github project, “Apache exporter for Prometheus”. [Source code].
Available: https://github.com/Lusitaniae/apache_exporter
[Accessed October 2022].

[23] Github project, Pedro Gomes, “Nginx Prometheus exporter”. [Source code].
Available: https://github.com/nginxinc/nginx-prometheus-exporter
[Accessed October 2022].

[24] Prometheus, “Understanding and using the Multi-Target exporter pattern”. [Online].
Available: https://prometheus.io/docs/guides/multi-target-exporter/
[Accessed October 2022].

[25] Github project, Prometheus, “Blackbox exporter configuration”. [Source code].
Available: https://github.com/prometheus/blackbox_exporter/blob/master/CONFIGURATION.md
[Accessed October 2022].

[26] Github project, Prometheus, “Prometheus SNMP exporter”. [Source code].
Available: https://github.com/prometheus/snmp_exporter
[Accessed October 2022].

[27] Paessler, “SNMP, MIBs and OIDs - an overview”. [Online].
Available: https://www.paessler.com/info/snmp_mibs_and_oids_an_overview
[Accessed October 2022].

[28] Influxdata, “SNMP Agent Protocol Monitoring”. [Online].
Available: https://www.influxdata.com/integration/snmp/
[Accessed October 2022].

[29] Github project, Daniel Pryor, “Vmware exporter”. [Source code].
Available: https://github.com/pryorda/vmware_exporter
[Accessed October 2022].

[30] Github project, Kugathasan Janarthanan, “ESXI Monitoring”. [Source code].
Available: https://github.com/kujalk/ESXI_Monitoring
[Accessed October 2022].

[31] Github project, CloudChef, “Vmware-exporter”. [Source code].
Available: https://github.com/CloudChef/vmware-exporter/blob/master/README.md
[Accessed October 2022].

[32] Gitlab project, “Gitlab exporter”, Nov, 14, 2017. [Source code].
Available: https://gitlab.com/gitlab-org/gitlab-exporter
[Accessed October 2022].

[33] Github project, Prometheus, “Prometheus Alertmanager”, Jul, 13, 2013. [Source code].
Available: https://github.com/prometheus/alertmanager
[Accessed October 2022].

[34] Alerta, “Alerta monitoring system”. [Online].
Available: https://docs.alerta.io/
[Accessed October 2022].

[35] Grafana, “What is Grafana”. [Online].
Available: https://grafana.com/oss/grafana/
[Accessed October 2022].

[36] Grafana dashboards, “Node Exporter Full”. [Online].
Available: https://grafana.com/grafana/dashboards/1860-node-exporter-full/
[Accessed October 2022].

[37] Github project, Gregor Krmelj, “Bigbluebutton exporter dashboards”, Feb, 21, 2021. [Source code].
Available: https://github.com/greenstatic/bigbluebutton-exporter/tree/master/extras/dashboards
[Accessed October 2022].

[38] Grafana dashboards, “Alertmanager”. [Online].
Available: https://grafana.com/grafana/dashboards/9578-alertmanager/
[Accessed October 2022].

[39] Elastic, “Centralize, transform & stash your data”. [Online].
Available: https://www.elastic.co/logstash/
[Accessed October 2022].

[40] Elastic, “Filebeat overview”. [Online].
Available: https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-overview.html
[Accessed October 2022].

[41] Elastic, “Lightweight data shippers”. [Online].
Available: https://www.elastic.co/beats/
[Accessed October 2022].

[42] Elastic, “What is Kibana”. [Online].
Available: https://www.elastic.co/what-is/kibana
[Accessed October 2022].

[43] Knoldus, “What is the ELK Stack?”, Jun, 4, 2020. [Online].
Available: https://medium.com/@knoldus/what-is-the-elk-stack-ad8398dd265e
[Accessed October 2022].

[44] KevinSummersill, “How to change the Node exporter port if it is already in use”, Dec, 13, 2021. [Online].
Available: https://aws.plainenglish.io/changing-the-node-exporter-port-if-already-in-use
[Accessed October 2022].

[45] Github project, Gregor Krmelj, “BigBlueButton exporter Systemd installation”. [Source code].
Available: https://github.com/greenstatic/bigbluebutton-exporter/systemd-installation
[Accessed October 2022].

[46] BigBlueButton, “Installation of Bigbluebutton Exporter”. [Online].
Available: https://bigbluebutton-exporter.greenstatic.dev/installation/bigbluebutton_exporter/
[Accessed October 2022].

[47] Superuser, “How to remove systemd services. [Online].
Available: https://superuser.com/questions/513159/how-to-remove-systemd-services
[Accessed October 2022].

[48] Dr. Shubham Dipt, “How to create a Systemd service in Linux”, Jan, 28, 2020. [Online].
Available: https://www.shubhamdipt.com/blog/how-to-create-a-systemd-service-in-linux/
[Accessed October 2022].

[49] Fail2ban, “Main Page”. [Online].
Available: https://www.fail2ban.org/wiki/index.php/Main_Page
[Accessed October 2022].

[50] Linode, “Using Fail2ban to Secure Your Server”. [Online].
Available: https://www.linode.com/docs/guides/using-fail2ban-to-secure-your-server-a-tutorial/
[Accessed October 2022].

[51] Ubuntu wiki, “Uncomplicated Firewall”. [Online].
Available: https://wiki.ubuntu.com/UncomplicatedFirewall
[Accessed October 2022].

[52] Nginx, “Nginx”. [Online].
Available: https://nginx.org/en/
[Accessed October 2022].

[53] DigitalOcean, “How To Secure Nginx with Let's Encrypt on Ubuntu 20.04”. [Online].
Available: https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-letsencrypt
[Accessed October 2022].

[54] Let’s Encrypt, “About Let’s Encrypt”. [Online].
Available: https://letsencrypt.org/about/
[Accessed October 2022].

[55] Juliusv, “PromSlack”. [Online].
Available: https://juliusv.com/promslack/
[Accessed October 2022].

[56] Liquid Web, “What is WSGI”, Sep, 25, 2019. [Online].
Available: https://www.liquidweb.com/kb/what-is-wsgi/
[Accessed October 2022].

[57] DigitalOcean, “How To Install and Secure Grafana on Ubuntu 20.04”. [Online].
Available: https://www.digitalocean.com/community/tutorials/how-to-install-grafana
[Accessed October 2022].

[58] DigitalOcean, “How To Install Elasticsearch, Logstash, and Kibana (Elastic Stack) on Ubuntu 20.04”. [Online].
Available: https://www.digitalocean.com/install-elasticsearch-logstash-and-kibana-elastic
[Accessed October 2022].

[59] Elastic, “Modules”. [Online].
Available: https://www.elastic.co/guide/en/beats/filebeat/7.6/filebeat-modules.html
[Accessed October 2022].