ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

PROPUESTA METODOLÓGICA Y SIMULACIÓN DE LA IMPLEMENTACIÓN DE UN SISTEMA (PROMETHEUS) DE CÓDIGO ABIERTO, QUE PERMITE REALIZAR EL MONITOREO DE APLICATIVOS

EXAMEN DE GRADO

Previo a la obtención del Título de:

MAGISTER EN TELECOMUNICACIONES

LORENA VANESSA CHASI MONTAÑO

GUAYAQUIL – ECUADOR

AÑO 2020

AGRADECIMIENTO

A mis padres, en especial a mi madre y mi tía Daysi quienes, con su amor, paciencia y ayuda me alentaron a seguir con esta meta. A mis compañeros y amigos de aula.

Y al amor, porque siempre estuvo presente en mi corazón.

DEDICATORIA

Este trabajo está dedicado a Dios, por darme las fuerzas y la sabiduría para seguir adelante hasta conseguir mi propósito y anhelo. A mi familia en especial a mi madre que a pesar de la distancia siempre estuvo alentándome, a mis hermanos y a mi gran amor por estar siempre presente.

TRIBUNAL DE EVALUACIÓN

MARIA ANTONIETA ALVAREZ VILLANUEVA Digitally signed by MARIA ANTONIETA ALVAREZ VILLANUEVA

…….…….....

MSc. Verónica Soto

PROFESOR EVALUADOR

…….…….....

María Antonieta Álvarez, PhD

PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, me corresponde exclusivamente; y doy mi consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual".

RESUMEN

Este proyecto presenta la implementación de una aplicación open Source denominada Prometheus el mismo que permitirá realizar el monitoreo de aplicativos (Base de datos, Correo electrónico, aplicativos webs, servidores, entre otros).

Es difícil intentar monitorear una aplicación sin una tecnología que permita detectar fallas en tiempo real con el fin de solucionar problemas y tener el control en forma oportuna y continua.

Este sistema en conjunto con otros aplicativos tiene la particularidad de presentar mediante graficas la situación actual de las métricas a monitorear para este caso (CPU, memoria, procesador entre otros), enviando notificaciones a correos electrónicos y al sistema de mensajería (Telegram).

INDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ABREVIATURA

HTML	Lenguaje de marca de Salida de Hipert Texto
HTTP	Protocolo de Transferencia de Hipertexto
SSH	Protocolo de Transferencia de Archivo Seguro
URL	Localizador de Fuente Uniforme

CAPÍTULO 1

1. INTRODUCCIÓN

1.1. Descripción del problema

En la actualidad la mayoría de las empresas pequeñas y medianas, no cuentan con herramientas básicas que permitan realizar el monitoreo de los diferentes aplicativos existentes en las empresas, como es el caso de aplicaciones web, servicios de correo, base de datos, servidores entre otros. Por ello la importancia del uso de herramientas para el monitoreo, permitiendo determinar el estado de los servicios y la detección temprana de problemas desde cualquier lugar o dispositivo.

Los aplicativos cumplen una función importante dentro de las empresas ya sea para las comunicaciones entre entidades u ofrecer diferentes servicios a clientes y es imperante el correcto funcionamiento y gestión de los mismos.

En aquellas empresas donde sus actividades son interrumpibles (24/7), es importante contar con un sistema que permita realizar el monitoreo de los diferentes aplicativos más relevantes para la institución y que estos puedan operar eficientemente y se le brinde un control oportuno y continuo.

1.2 Descripción de la propuesta

Prometheus es un sistema de monitoreo Open Source basado en series de tiempo y alertas, que permiten monitorear aplicativos o equipos donde se encuentren alojados estos aplicativos. Los aplicativos serán identificados o establecidos por el administrador de seguridad de la información, donde indicará cuales son los aplicativos (Base de datos, Correo electrónico, aplicativos webs, entre otros) más críticos de una empresa y aquellos que serán monitoreados. La creación de esta metodología de implementación permitirá realizar un monitoreo y análisis de datos de la empresa con el fin de anticiparse a posibles contratiempos. Contar con un sistema de monitorización es indispensable para optimizar la administración de

sistemas IT, ganando así eficiencia en los servicios ofrecidos tanto a los usuarios internos como a clientes.

Después de un análisis de varias herramientas de monitoreo de aplicaciones, se implementará Prometheus el mismo que estará alojado en un repositorio en la nube, se monitoreará el servidor o un equipo que cuenten con varios servicios y por medio del conector Node_Exporter se extraerán las métricas.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar una propuesta de metodología y simulación de la implementación de un Sistema (Prometheus) de código abierto, que permite realizar el monitoreo de aplicativos.

1.3.2 Objetivos Específicos

- ✓ Monitorear equipos o aplicativos críticos de la empresa.
- ✓ Obtener un desarrollo de un enfoque preventivo.
- ✓ Garantizar el cumplimiento de todos los servicios y estándares, a través del monitoreo se podrá recolectar información la misma que servirá para detectar comportamientos anormales e identificar posibles amenazas a la seguridad de los datos.
- ✓ Construir gráficas de visualización de los servicios del rendimiento de los activos de TI.

1.4 Marco Teórico

Los sistemas informáticos en los últimos años han evolucionado de manera significativa, su avance tecnológico se ve reflejado en el progreso de los servicios, los mismos que permiten mejorar la eficiencia y continuidad de los procesos de una empresa. Es muy común que las organizaciones utilicen herramientas básicas, con el fin de garantizar el correcto funcionamiento de la misma, de una manera ordenada y ágil. Cada organización cuenta con una

infraestructura de redes y telecomunicaciones ya sea física o en la nube compuesta por una serie de dispositivos o equipos donde se encuentran instalados aplicativos como (Base de datos, Correo electrónico, aplicativos webs, entre otros), que componen la cadena de valores de la organización.

Las redes de telecomunicaciones soportan cada vez más aplicativos y servicios valiosos de las organizaciones, por lo que el análisis y monitoreo de los aplicativos se ha transformado en una actividad de carácter importante para evitar problemas a futuro dentro de las organizaciones. Prometheus es un sistema de monitoreo Open Source basado en series de tiempo y alertas, que sirve para monitorear servicios en estado down, memoria, procesador, disco entre otros, además admite enviar vía correo electrónico alertas sobre eventualidades presentadas en los aplicativos monitoreados, permitiendo resolver con rapidez y eficacia los problemas suscitados.

Los analistas de seguridad son conscientes de la importancia de monitorear eventos en aquellos equipos o aplicativos de mayor interés para la organización, para esto es necesario plantearse las siguientes preguntas:

¿Qué tecnología permite monitorear aplicativos o equipos?

¿Existe alguna norma que se ajuste al proceso de monitoreo de eventos?

Para despejar las dudas es importante tener los siguientes conceptos claros, conceptos relacionados con la utilización de las herramientas a usarse para la implementación del proyecto como son:

1.4.1. Que es Prometheus

Prometheus es una base de series de tiempo y un sistema de monitoreo y alertas. Las series de tiempo se encargan de almacenar datos ordenados de manera cronológica, permitiendo medir variables a lo largo del tiempo, de igual forma las bases de datos enfocadas a series de tiempo son eficientes en almacenamiento y consultas de datos [1].

Características principales:

1. **Modelo de datos multidimensional:** Trabaja con datos de series de tiempo que están identificados por nombres métricos y claves /valores pares.
2. **Lenguaje flexible de consultas:** Por medio de la multidimensionalidad, se puede consultar datos de una manera sencilla y flexible.
3. **Nodos y autónomos de servidor único:** Por medio de HTTP, el modelo de extracción realiza la colección de series de tiempo
4. **Configuración estática**: Se determina a través del descubrimiento de servicios o configuraciones estáticas.

1.4.2. Arquitectura de Prometheus

La arquitectura permite almacenar las muestras de manera local, ejecutando reglas sobre estos datos, para añadir y registrar nuevas series temporales partiendo de los datos existentes o generando alertas. En la Figura 1.1 se puede visualizar la Arquitectura.

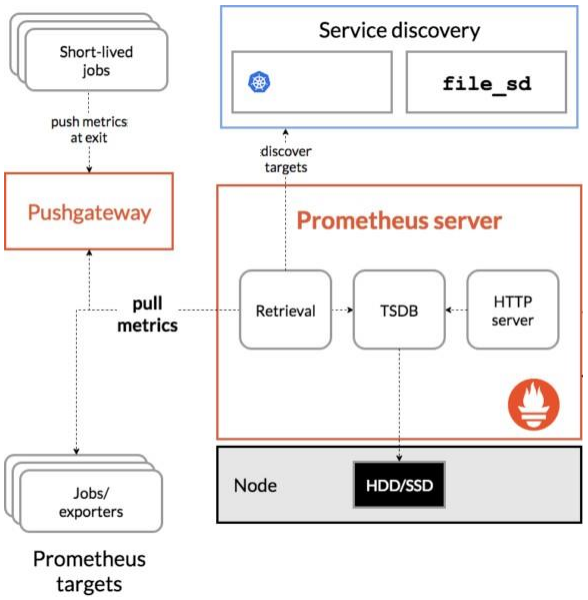


Figura 1.1 Arquitectura Prometheus

Prometheus utiliza el modelo HTTP pull, es decir que es el que va por las métricas a las aplicaciones expuestas o también llamadas exporters.

1.4.3. Node_Exporter

Es el encargado de obtener las métricas y brindarlas en un formato entendible para Prometheus, con Node_Exporter estas y entre otras son las métricas que puede usar Prometheus [2], ver tabla 1.:

Name	Enaled by Default	Description
conntrack	Yes	Muestra estadísticas de Conntrack (no hace nada si no /proc/sys/net/netfilter/ present).
diskstats	Yes	Expone las métricas de I/O de los discos.
edac	Yes	Estadísticas de detección y corrección de errores.
entropy	Yes	Expone la entropía disponible (muy importante para la criptografía).
filefd	Yes	Estadísticas del descriptor de archivos de /proc/sys/fs/file-nr.
filesystem	Yes	Expone las estadísticas de los sistemas de ficheros, como el espacio en disco usado, etc.
hwmon	Yes	Expone los sensores de hardware de /sys/class/hwmon/.
infiniband	Yes	Estadísticas de red específicas para configuraciones InfiniBand.
loadavg	Yes	Expone el promedio de carga del nodo.., Dragonfly, FreeBSD, Linux, NetBSD, OpenBSD, Solaris
mdadm	Yes	Estadísticas sobre dispositivos en /proc/mdstat (does nothing if no /proc/mdstat present).
meminfo	Yes	Expone las estadísticas de la RAM. , Dragonfly, FreeBSD, Linux
netdev	Yes	Más estadísticas de red, como los bytes transferidos, etc.
netstat	Yes	Estadísticas de red de /proc/net/netstat. Esta es la misma información que netstat -s.
sockstat	Yes	Varias estadisticas de /proc/net/sockstat.
stat	Yes	Varias estadisticas de /proc/stat. Esto incluye uso de CPU, tiempo de arranque, bifurcaciones e interrupciones

Tabla 1. Métricas Node_Exporter

Existen varios módulos o exportes para monitorizar diferentes servicios remotos, cada exporter cuenta con sus métricas respectivas. En la Tabla 2 se describe algunos exportes:

Exporter	Descripción	Métrica	Descripción de la métrica
statsd_exporter	Exportadora de métricas de estadísticas a Prometheus	StatsD	Traduce métricas de StatsD a Prometheus a través de reglas de mapeo configurables
		Tagging Extensions	El exportador admite etiquetas de estilo Librato, InfluxDB, DogStatsD y SignalFX, que se convertirán en etiquetas Prometheus.
consul_exporter	Exportador de métricas de Consul	consul_up	¿Fue exitosa la última consulta del Cónsul?
		consul_raft_peers	Cuántos pares (servidores) hay en el clúster de Raft
		consul_serf_lan_members	Cuántos miembros hay en el clúster
		consul_serf_lan_member_status	Estado del miembro en el clúster. 1 = vivo, 2 = saliendo, 3 = izquierda, 4 = fallido.

Exporter	Descripción	Métrica	Descripción de la métrica
mysqld_exporter	Exportador de métricas de servidor MySQL	collect.auto_increment.columns	Recopila columnas auto_increment y valores máximos de information_schema.
		collect.binlog_size	Recopila el tamaño actual de todos los archivos binlog registrados
		collect.global_status	Recopila y MUESTRA ESTADO GLOBAL (habilitado de forma predeterminada)
		collect.info_schema.tables	Recopila métricas de information_schema.tables.
		collect.slave_hosts	Colecciona de SHOW SLAVE HOSTS
		config.my-cnf	ruta al archivo .my.cnf para leer las credenciales de MySQL. (defecto: ~/.my.cnf)
		exporter.lock_wait_timeout	Se establece un lock_wait_timeout en la conexión para evitar un bloqueo prolongado de metadatos. (predeterminado: 2 segundos)

Exporter	Descripción	Métrica	Descripción de la métrica
memcached_exporter	Exporta métricas de servidores memcached para que las consuma Prometheus		
haproxy_exporter	Servidor simple que extrae las estadísticas de HAProxy y las exporta a través de HTTP para el consumo de Prometheus		
graphite_exporter	Servidor que acepta métricas a través del protocolo Graphite y las exporta como métricas de Prometheus		
blackbox_exporter	Exportador de prueba de caja negra		

Tabla 2. Métricas de Prometheus

1.4.4. Grafana

Grafana es un software de análisis y visualización de código abierto. Le permite consultar, visualizar, alertar y explorar sus métricas sin importar dónde estén almacenadas. En un lenguaje sencillo, le proporciona herramientas para convertir los datos de su base de datos de series de tiempo (TSDB) en gráficos y visualizaciones [3].

Figura 1.2 Grafana

Grafana (ver Figura 1.2) permite observar datos meteorológicos y estadísticos, explorar métricas y logs y anotaciones gráficas con eventos.

1. Alejar rango de tiempo.
2. Menú desplegable del selector de tiempo. Aquí puede acceder a opciones de rango de tiempo relativo, opciones de actualización automática y establecer rangos de tiempo absolutos personalizados.
3. Botón de actualización manual. Hará que todos los paneles se actualicen (obtengan nuevos datos).
4. Panel del tablero. Haga clic en el título del panel para editar paneles.
5. Leyenda del gráfico. Puede cambiar los colores de la serie, el eje y, y la visibilidad de la serie directamente desde la leyenda.

1.4.5. Definición Open Source

Open Source o código abierto se denomina así a Software's que son distribuidos y desarrollados de manera libre donde el código fuente es compartido libremente, a diferencia del software libre que son programas que se pueden descargar y distribuirse de manera gratuita (sin código fuente). En la actualidad se han incrementado de una manera acelerada el uso de este tipo de programas [4].

1.4.6. Infraestructura de Red y Redes de comunicaciones

Se denomina así aquellos elementos básicos y necesarios para una organización ya sea de tipo pública o privada que haga uso de servicios tecnológicos como: impresoras, cámaras de seguridad y vigilancia, climatización, ordenadores entre otros. Como se ilustra en la figura 1.3.

Los diferentes elementos que conforman una infraestructura de red son:

- ✓ Seguridad y control
- ✓ Climatización
- ✓ Cableado estructurado
- ✓ SAI: Sistema de Alimentación Ininterrumpida de equipos de IT
- ✓ Redundancia eléctrica
- ✓ Cuarto de comunicaciones.

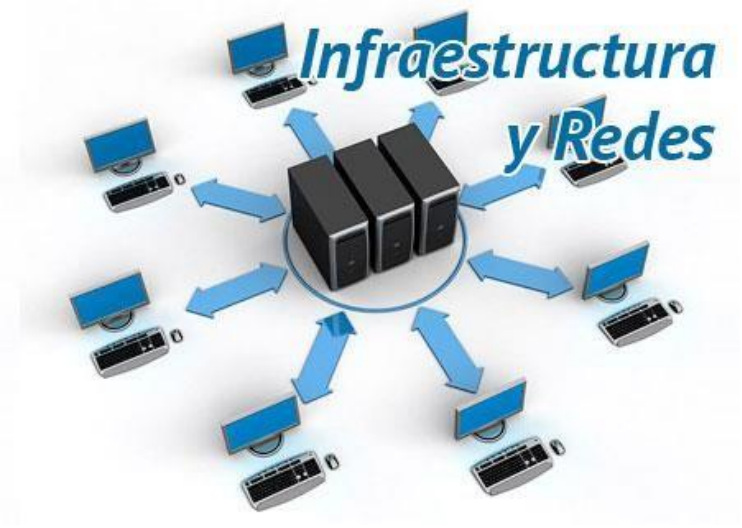


Figura 1.3 Infraestructura y Redes

A las redes de comunicaciones se la conoce como el intercambio de información entre dos o más entidades. Aquella información que viaja como paquete permitiendo lograr una comunicación entre entidades haciendo uso de los diferentes protocolos de comunicación.

1.4.7. Monitoreo de Redes

Definición de Monitoreo

Se determina así al sistema que monitorea constantemente los servicios en una red, en busca de posibles eventualidades adicionalmente permitirle al administrador de la red recibir estas notificaciones vía correo, mensajes de texto u otros tipos de alarmas, con el fin de resolver los problemas en el menor tiempo posible.

Herramientas de monitoreo

En el mercado existen un sin número de herramientas de monitoreo como son:

- ✓ **Zabbix:** permite monitorear cualquier dispositivo en la red, usa SNMP, posee interfaz web, provee plantillas genéricas para servidores Linux, correo, web entre otros [5].

- ✓ **Solarwinds:** permite la administración de archivos de configuración de la red a través de una interfaz web [6].
- ✓ **Cacti:** es una herramienta para monitorear, presentar y archivar estadísticas de redes y servidores [7].
- ✓ **Nagios:** permite determinar la conectividad entre los hosts y garantizar la funcionabilidad de los servicios de la red [8].
- ✓ **Pandora FMS:** es una herramienta de código abierto que permite monitorear la infraestructura de una red, servidores plataformas virtualizadas y aplicativos [9].

1.4.8. Computación en la Nube y soluciones de (Cloud Computing)

El Instituto Nacional de Estándares y Tecnología (NIST) en la publicación No. 800-145 define Cloud Computing como el modelo que permite el acceso ubicuo, conveniente y bajo demanda a una gran cantidad de servicios configurables (ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser provistos eficientemente y manejados con un bajo esfuerzo e interacción mínima del proveedor del servicio [10]. La siguiente figura 1.4, representa la definición y esquema de Cloud Computing.

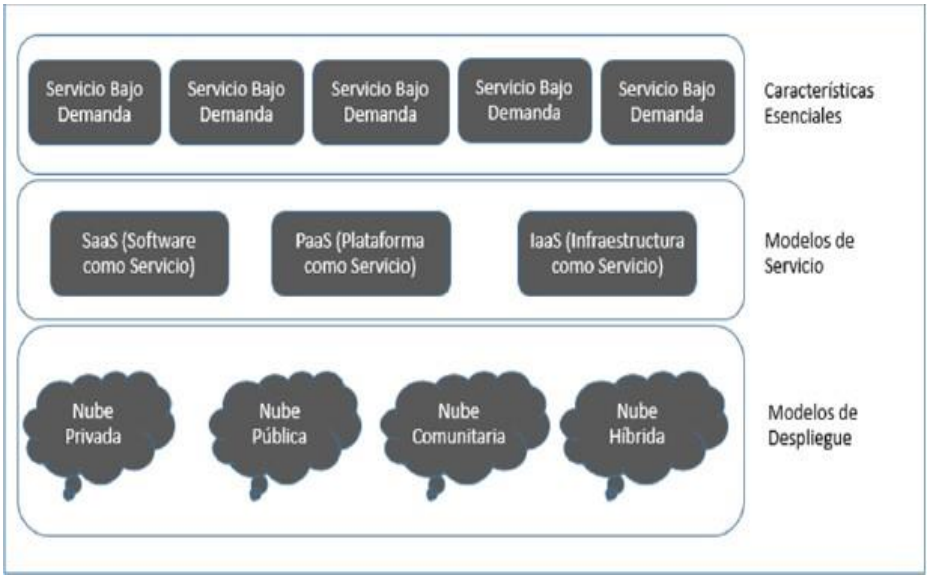


Figura 1.4 Esquema de Cloud Computing

Existen varias organizaciones que ofrecen el servicio en la Nube cada una de ellas ha ido evolucionando según las necesidades y exigencias del mercado.

En la siguiente figura 1.5, se muestra el posicionamiento de las Organizaciones según el cuadrante de Gartner de fecha febrero 2020 [11].

Figura 1.5 Cuadrante mágico de Gartner CLOUD 2020

1.4.9. Norma ISO 27001 y Ciclo de Deming

La ISO 27001, es una norma Internacional que permite establecer, implementar, operar, monitorear, revisar, mantener y mejorar un SGSI (Sistema de Gestión de Seguridad de la Información). Adoptar un SGSI es una decisión importante para la organización y debe de basarse en las necesidades, objetivos y cadena de valor de la organización [12].

Dentro de una organización es necesario identificar y administrar sus actividades con el fin de que todo marche correctamente. Realizar un sistema de procesos en una organización, con la identificación de sus actividades, procesos y gestión, se considera un enfoque del proceso. Se debe de identificar claramente los requisitos de seguridad de la información, realizar una correcta implementación de controles con políticas de seguridad aplicados a aquellos activos de información más valiosos dentro de la organización y realizar el monitorear del desempeño del SGSI.

El ciclo de Deming es considerado como una estrategia para implantar un sistema de mejora continua aplicables a cualquier proceso de una organización [13]. En la figura 1.6 se visualiza el ciclo de Deming.

El circulo de Deming consiste en 4 etapas:

1. Planear: Se plantean los requerimientos y las estrategias a usar para cumplir con los objetivos.
2. Hacer: En esta fase se acciona los pasos planificados anteriormente.
3. Verificar: Se realiza la verificación de la planeación y se da seguimiento.
4. Actuar: Los resultados obtenidos en la etapa anterior, se analizan y se tomará acciones correctivas y preventivas.

Figura 1.6 Ciclo de Deming

CAPÍTULO 2

2. METODOLOGÍA PARA MONITOREAR EQUIPOS (SERVIDORES)

El presente trabajo de investigación corresponde a un estudio de implementación o proyecto factible, realizable dentro de los plazos establecidos y haciendo uso de plataformas libres. La investigación a realizarse se basa en como implementar un sistema (Open Source) que monitoree aplicativos o equipos que mejoren la disponibilidad de los sistemas, permitiendo reducir los errores y así mantener la continuidad de las operaciones en una organización.

El Manual de Tesis de Grado y Especialización y Maestría y Tesis Doctorales de la Universidad Pedagógica Libertador, (2003), manifiesta que un proyecto factible: "Consiste en la investigación, elaboración y desarrollo de un modelo operativo viable para solucionar problemas, requerimientos necesidades de organizaciones o grupos sociales que pueden referirse a la formulación de políticas, programas, tecnologías, métodos, o procesos. El proyecto debe tener el apoyo de una investigación de tipo documental, y de campo, o un diseño que incluya ambas modalidades [14].

Así mismo Arias [15, p. 134]**,** señala: "Que se trata de una propuesta de acción para resolver un problema práctico o satisfacer una necesidad. Es indispensable que dicha propuesta se acompañe de una investigación, que demuestre su factibilidad o posibilidad de realización".

Según Labrador y Otros, [16, p. 186] las fases o etapas de la factibilidad son: diagnóstico, factibilidad y diseño, por lo que expresan: "El diagnóstico es una reconstrucción del objeto de estudio y tiene por finalidad, detectar situaciones donde se ponga de manifiesto la necesidad de realizarlo".

De todas las definiciones citadas podemos concluir que cuando hablamos de un Proyecto factible nos referimos a una propuesta viable, cuyo fin es atender las necesidades fijadas a partir de un diagnóstico.

Este trabajo tomará como referencia los documentos” Metodologías para crear metodologías”, en directa relación con la información de las metodologías, de las Normas ISO270001 y el ciclo de Deming.

2.1 Diseño de Metodología

La “Metodología para crear metodologías” [17], es la manera de crear metodologías de una forma ordenada aplicables en cualquier ámbito, facilitando las tareas complejas en pasos secuenciales y lógicos. A continuación, se detallan las actividades de la “Metodología para crear metodologías” (Tabla 3):

Metodología para crear Metodologías		
#	**Actividades**	**Descripción**
1	Definición de Alcance	Se identifican los objetivos de los procesos
2	Puntos clave de la problemática	Determinar los puntos claves de los subprocesos que forman parte de la implementación
3	Poner el proceso en contexto	Provee una visión para determinar el alcance, el camino y el uso de los procesos
4	Criterios para la elaboración de los subprocesos	Establece criterios para evaluar los subprocesos y sus resultados
5	Inventario de información y recursos	Se determina cuales son los insumos a utilizar.
6	Orden lógico de los recursos	Organiza el proceso en pasos permitiendo incluir la retroalimentación
7	Ejecución de la metodología	Implica la prueba de la metodología, utilizándola como guía, mas no como reglas
8	Evaluación de cada paso	Recopila datos y mide el rendimiento en "tiempo real" para mejorar a un futuro el rendimiento
9	Facilitación del proceso	Los participantes se instruyen sobre el aprendizaje de los procesos por medio de habilidades de facilitación.
10	Evaluación del rendimiento	Se determina cambios necesarios en la metodología analizando los resultados

Tabla 3. Resumen de Metodología para crear Metodologías

2.1.1 Definición de Alcance

La creación de esta metodología empezará determinado el modelo de negocios de TI, donde se identifique los activos de información, que información es la que se debe salvaguardar y dar un constante monitoreo de la misma y que la implementación a realizarse cubra las necesidades.

Como resultado de esta implementación se obtendrá una plataforma de monitoreo, herramienta utilizada para la administración, visualización y gestión de eventos. Es decir que esta sería una guía para los profesionales TIC, donde estarán en la capacidad de implementar un sistema de monitoreo.

2.1.2 Puntos clave de la Problemática

Se considera un punto clave y crítico la integración de los componentes: software, hardware y conectividad, de aquí la investigación realizada para este proyecto se seleccionó las metodologías “Top Dow [18]” (tecnología orientada a la implementación de redes) y “MSF [19]” (metodología orientada a la implementación de software). Donde se elaboró una comparación de las metodologías donde se determina lo siguiente (Tabla 4):

Metodologías		
#	**Top Down**	**MSF**
1	Análisis de Requerimientos	Visión y Alcance
2	Diseño Lógico	Planificación
3	Desarrollar Diseño Físico	Desarrollo
4	Probar, optimizar y documentar diseño	Estabilización
5	Implementar y probar la red	Implementación o Despliegue
6	Monitorear y optimizar la red	

Tabla 4. Comparación de metodologías Top Down y MSF

Solo la tecnología “TOP DOWN”, propone realizar un monitoreo y optimización, considerándolo importante para este proyecto de implementación. Este proceso se subdivide de la siguiente manera:

#	**Subprocesos**
1	Requerimientos
2	Planificación
3	Selección de la tecnología
4	Pruebas
5	Implementación
6	Monitoreo

Tabla 5. Metodología Top Down

2.1.3 Poner el proceso en contexto

Se refiere a tener una visión sistemática del uso de cada proceso.

2.1.4 Criterios para la elaboración de los Subprocesos

Los criterios pertenecen a cada una de las actividades que deben de desarrollarse dentro de cada subproceso. Definiendo las actividades a realizarse nos basamos en las metodologías “Top Dow [18]” y “MSF [19]”, según lo muestra el cuadro comparativo en la tabla 6:

METODOLOGÍAS			
TOP DOWN		**MSF**	
Subprocesos	**Actividades**	**Subprocesos**	**Actividades**
Análisis de Requerimientos	Analizar metas del negocio	Visión y Alcance	Elaboración y aprobación documento visión y alcance
	Metas Técnicas		Equipo de trabajo, competencias y responsabilidades
	Análisis de red Existente		Elaboración de plan de trabajo
	Análisis de Trafico Existente		Elaboración de matriz de riesgo

METODOLOGÍAS			
TOP DOWN		MSF	
Subprocesos	**Actividades**	**Subprocesos**	**Actividades**
Diseño Lógico	Diseño de topología de red	Planificación	Elaboración de planificación y diseño de arquitectura
	Diseño de modelos de direccionamiento y “hostnames”		
	Selección de protocolos para “Switching” y ”Routing”		Laboratorio (pruebas de concepto)
	Desarrollo de estrategias de seguridad		
	Desarrollo de estrategias de administración de red		
Desarrollo de Diseño Físico	Selección de tecnologías y dispositivos para redes	Desarrollo	Construcción de los componentes de la infraestructura a implementar
Prueba, optimización y documentación del diseño	Probar el diseño de red	Estabilización	Planificación de los casos de pruebas que se ejecutarán
	Optimizar el diseño de red		“Bugtracker”: documentación de errores a ser corregidos
	Documentar del diseño		

METODOLOGÍAS			
TOP DOWN		**TOP DOWN**	
Subprocesos	**Subprocesos**	**Subprocesos**	**Subprocesos**
Implementación y prueba de la red	Realizar de cronograma de implementación	Despliegue o Liberación	Implementación de la plataforma
	Implementación del diseño de red		Análisis de nuevos procesos
			Análisis personal extra
	Realizar pila de pruebas		Matriz de impacto
			Lista de chequeo
			Capacitación a usuarios
Monitoreo y optimización de la red	Operación de la red en producción		
	Monitoreo y optimización de la red		

Tabla 6. Comparativa de las actividades de las metodologías “Top Down” y “MSF”

2.1.5 Inventario de información y recursos

En este punto se determinan las entradas y salidas que son las mismas que ejecutan cada uno de los subprocesos, definidos anteriormente en la tabla Comparativa de las actividades de las metodologías “Top Down [18]” y “MSF [19]”.

A continuación, se resumen los subprocesos de la implementación de un sistema de monitoreo de aplicativos o equipos, haciendo uso del Sistema (Prometheus con Node_Exporter y Grafana) Tabla 7:

#	Subprocesos	Recursos/insumos	Resultado
1	Requerimientos	Inventario de activos de información	Activos y eventos de seguridad
		Documento análisis de riesgo	
2	Planificación	Activos y eventos de seguridad	Diseño de arquitectura de Prometheus (Plataforma Virtual)
		Diagrama de la red	
3	Selección de la tecnología	Activos y eventos de seguridad	Conector para extraer métricas según la necesidad (Node_Exporter)
		Diseño de arquitectura de Prometheus	
4	Pruebas	Activos y eventos de seguridad	Resultado de pruebas y correcciones (simulación)
		Diseño de arquitectura de Prometheus	
		Sistema Prometheus a utilizar	
5	Implementación de la arquitectura Prometheus (Física o Virtual)	Resultado de pruebas y correcciones	Implementación y configuración de la arquitectura Prometheus con Node_Exporter
		Diagrama de la red	
		Activos y eventos de seguridad	
		Diseño de arquitectura Prometheus	
		Sistema Prometheus a utilizar	
6	Implementación de agentes	Activos y eventos de seguridad	Implementación de envío de alertas vía correo.
		Implementación y configuración de la arquitectura Prometheus y Grafana	Implementación de Grafana para visualización de eventos de manera gráfica
7	Monitoreo	Diagrama de la red	Prometheus con Node_Exporter
		Activos y eventos de seguridad	Reporte y visualización de eventos a través de Grafana
		Implementación de agentes	
		Implementación y configuración de la arquitectura de Prometheus con Node_Exporter	

Tabla 7. Subprocesos Recursos y Resultados

Para realizar la implementación de Prometheus con Node_Exporter y Grafana, se lo ha realizado de manera virtual a través de la plataforma GOOGLE CLOUD [20] y para las respectivas pruebas se ha implementado en AWS (Amazon Web Services) [21], un servidor con el fin de ser monitoreado.

2.1.6 Orden lógico de los recursos

Todos los subprocesos descritos en la tabla de Subprocesos Recursos y Resultados son los componentes del proceso de implementación del Sistema de Monitoreo, conservando el orden y un paso a paso según como se visualiza en la siguiente figura.

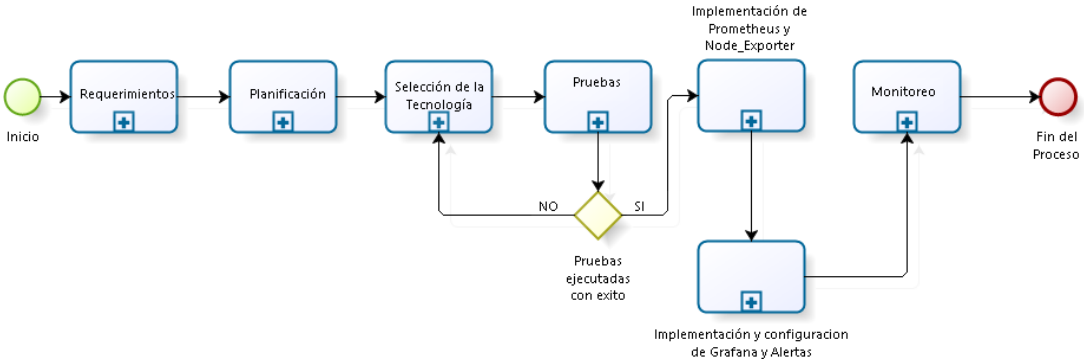


Figura 2.1 Flujo de procesos de la implementación de un Sistema de Monitoreo de aplicativos o equipos

Las actividades mencionadas en la tabla de Resumen de Metodología para crear Metodologías 1 – 6 ya fueron abordadas; la actividad 7 se desarrollará en la fase de ejecución, los puntos 8 y 10 en la Fase de Resultados y Discusión y el punto 9 no aplica para el desarrollo de este proyecto.

2.2 Fase de ejecución

El objetivo de esta fase es la validación de la metodología descrita en los anteriores puntos, y para este trabajo la simulación se desarrollará en una red de tamaño mediano implementando el sistema de Monitoreo (Prometheus con Node_Exporter y Grafana) sobre la misma, donde a continuación se muestra la Arquitectura de red a usar (Figura 2.2):

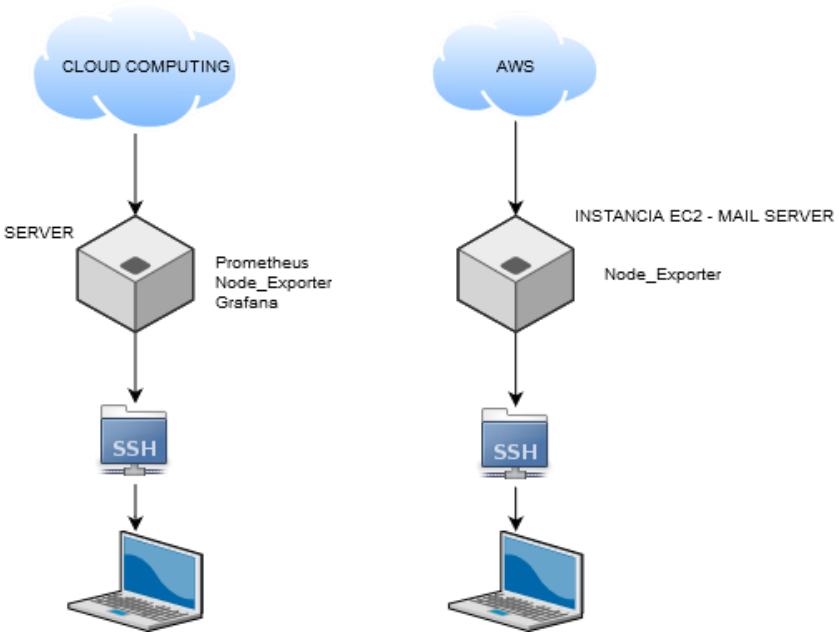


Figura 2.2 Topología de red

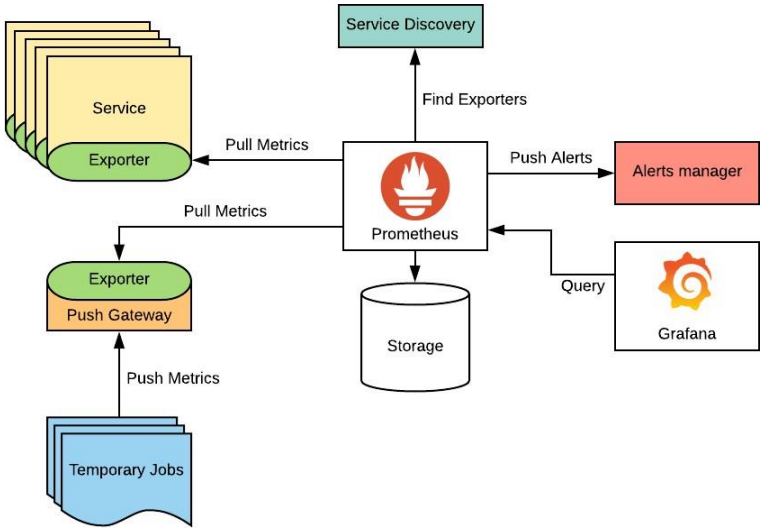


Figura 2.3 Arquitectura General de Prometheus, Node_Exporter y Grafana

2.2.1 Implementación del ambiente de simulación

En la implementación del sistema de monitoreo de aplicativos o equipos se utilizaron las plataformas de CLOUD COMPUTING, como se refleja en la siguiente tabla:

Nombre	Descripción	Dirección Ip Pública
AWS CLOUD	Configuración del servidor de correo para ser monitoreado por medio de Node_Exporter	54.207.89.182
GOOGLE CLOUD	Configuración del servidor que comprende: Prometheus, Node_Exporter, Grafana y Alertas	35.193.246.169

Tabla 8. Tecnología CLOUD

Hardware

Para la implementación se utilizó las siguientes características de Hardware (Tabla 9):

Equipo	Características	
AWS CLOUD [21]	CPU	Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
	RAM	4GB
	DISCO	30 GB
	S.O.	AMAZON LINUX 2 AMI
GOOGLE CLOUD [20]	CPU	Intel(R) Xeon(R) CPU @ 2.30GHz
	RAM	4GB
	DISCO	10 GB
	S.O.	Ubuntu 18.04

Tabla 9. Requerimientos de Hardware

Software

Contar con un ambiente de simulación y con la infraestructura adecuada es muy importante para llegar a realizar esta implementación. La tabla 10. muestra los requisitos del software:

Nombre	Descripción	Versión
Node_Exporter	Sirve como conector para extraer las métricas de los equipos o aplicativos a monitorear	node_exporter-1.0.1.linux-amd64
Prometheus	Presenta las métricas extraídas en tiempo real	prometheus-2.20.0.linux-amd64
Grafana	Permite presentar de manera gráfica la información en tiempo real	grafana-6.3.6.linux-amd64

Tabla 10. Requerimientos de Software

2.2.2 Configuración del Sistema de monitoreo

Basados que a este punto las máquinas virtuales a usar en la implementación ya están configuradas partimos:

Implementación de Node_Exporter

Para realizar la implementación se debe de ejecutar los siguientes pasos:

En la línea de consola ejecutamos: wget https://github.com/prometheus/node_exporter/releases/download/v1.0.1/node_exporter-1.0.1.linux-amd64.tar.gz

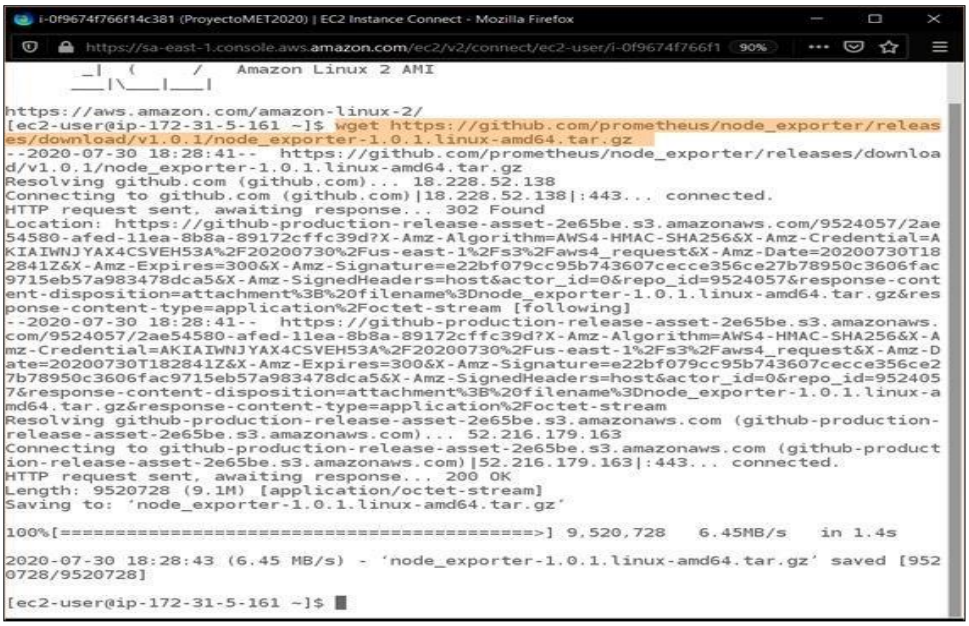


Figura 2.4 Descarga de Node_Exporter

Descomprimir con el comando: tar -xzf node_exporter-1.0.1.linux-amd64.tar.gz y levantar el servicio por medio del comando "./node_exporter". Ingresamos a la dirección IP y haciendo uso del puerto 9100 para corroborar el levantamiento del servicio lo verificamos de la siguiente manera:

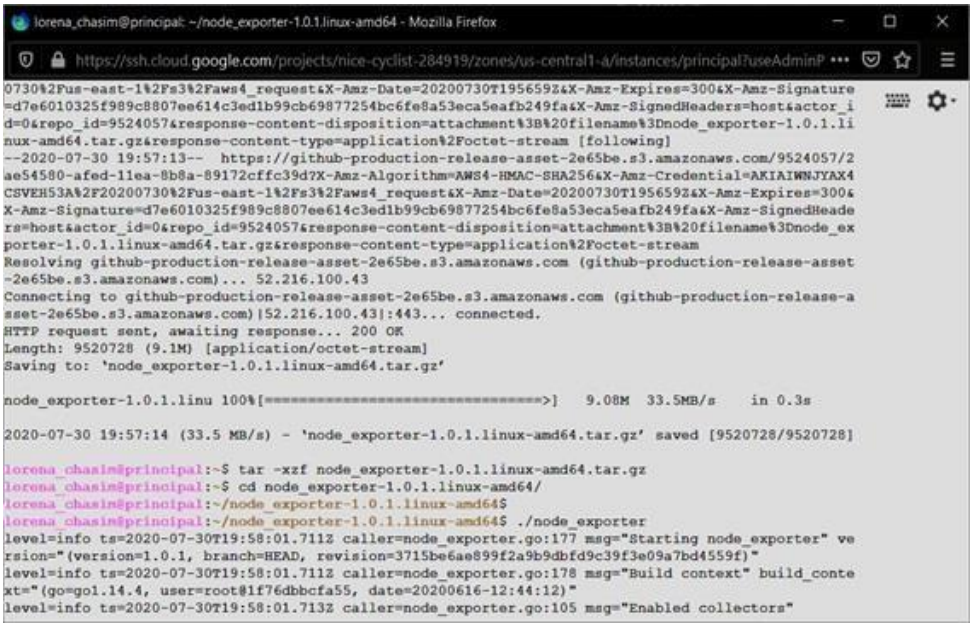


Figura 2.5 Extracción y ejecución de Node_Exporter

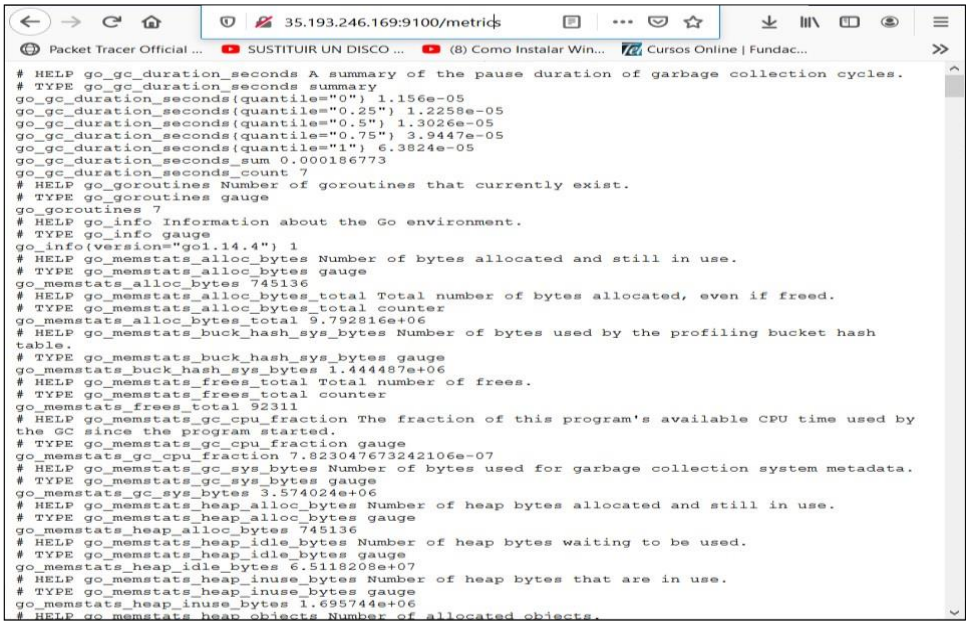


Figura 2.6 Visualización de Métricas

Importante **Node_Exporter** debe de estar instalando tanto en el servidor como en los clientes que se vayan a monitorear.

Implementación de Prometheus

Para realizar la implementación se debe de ejecutar los siguientes pasos

En la línea de consola ejecutamos: wget https://github.com/prometheus/node_exporter/releases/download/v2.20.0/

prometheus-2.20.0.linux-amd64.tar.gz

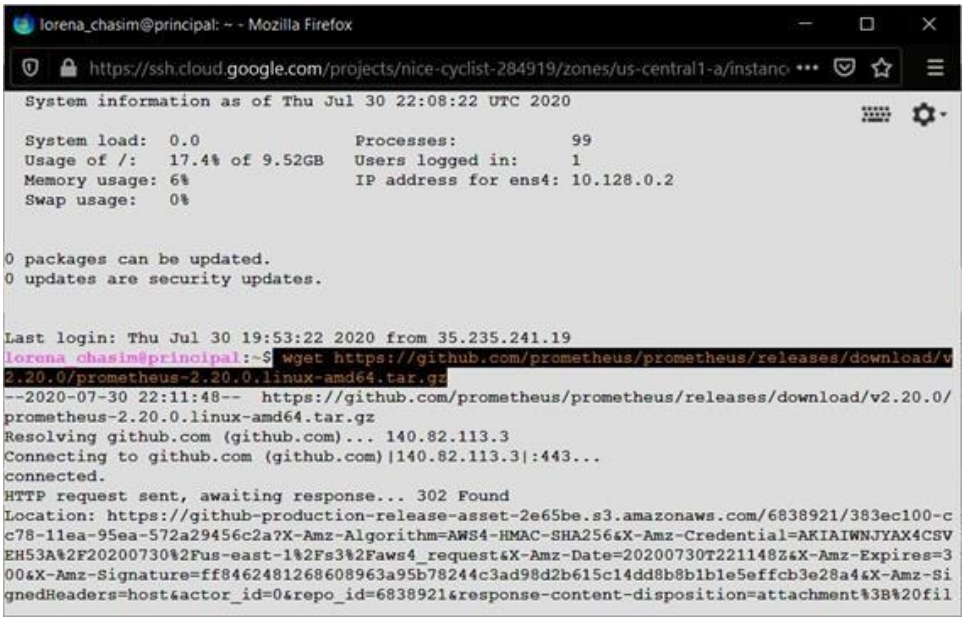


Figura 2.7 Descarga de Prometheus

Descomprimir con el comando tar -xzf prometheus-2.20.0.linux-amd64.tar.gz y levantar el servicio por medio del comando “./Prometheus”. Ingresamos a la dirección IP y haciendo uso del puerto 9090 para corroborar el levantamiento del servicio lo verificamos de la siguiente manera:

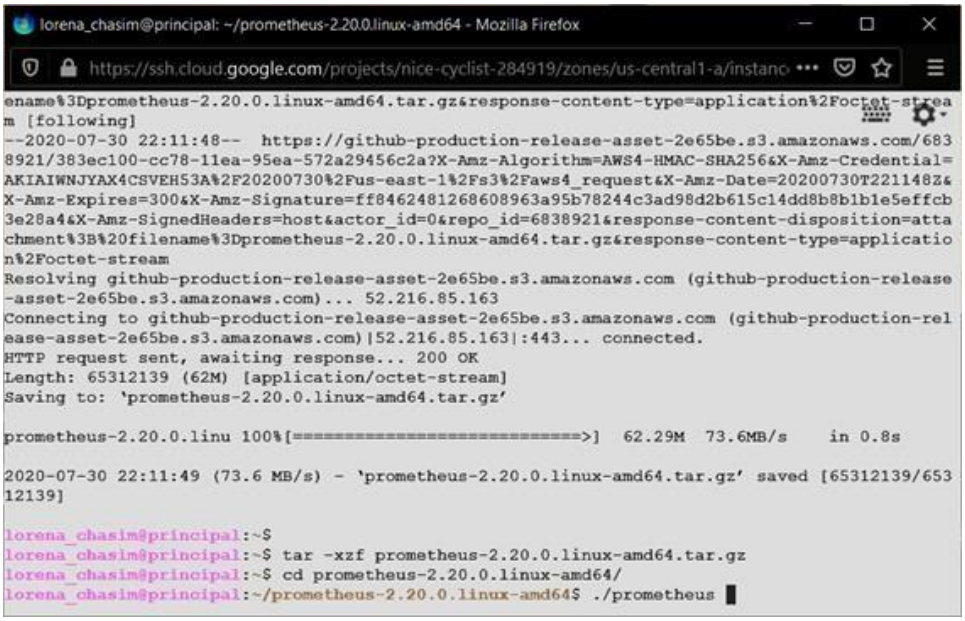


Figura 2.8 Extracción y ejecución de Prometheus

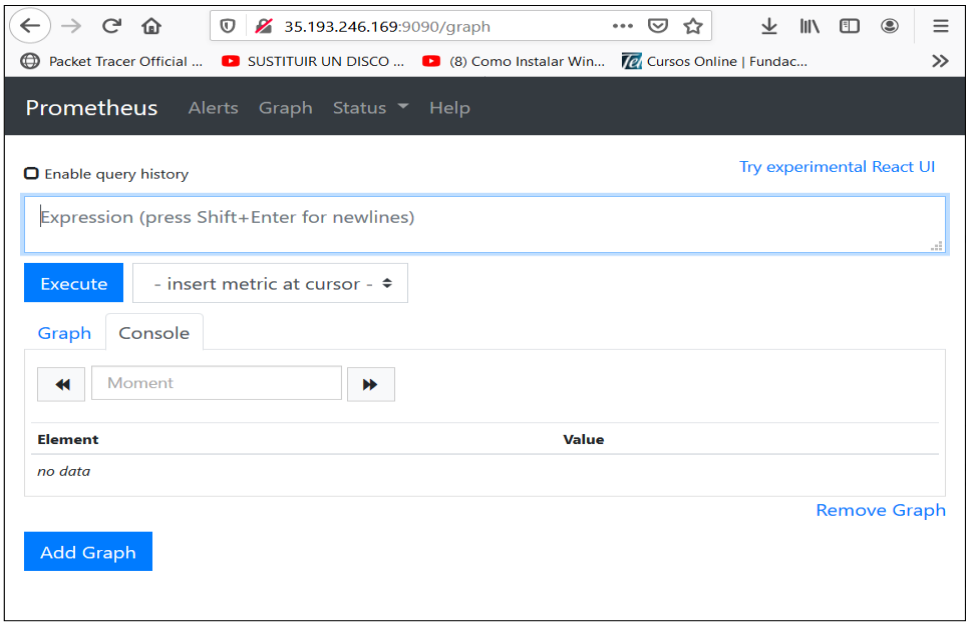


Figura 2.9 Visualización de Prometheus

Integración de node_exporter con Prometheus

Ubicar y editar el archivo “prometheus.yml”, en el cual se configurarán los trabajos y las direcciones de los equipos a monitorear con el respectivo puerto.

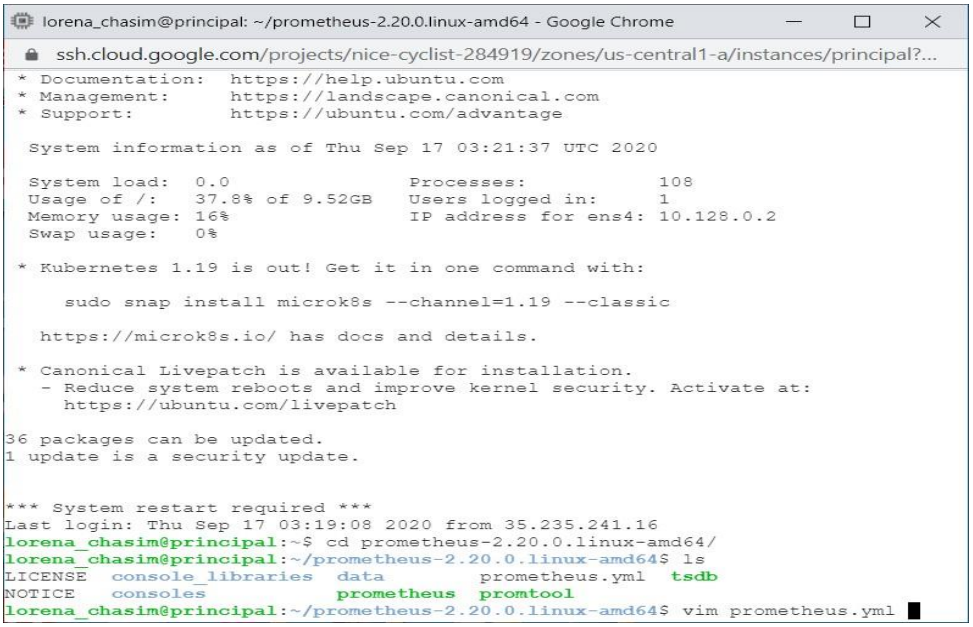


Figura 2.10 Integración de Node_Exporter con Prometheus

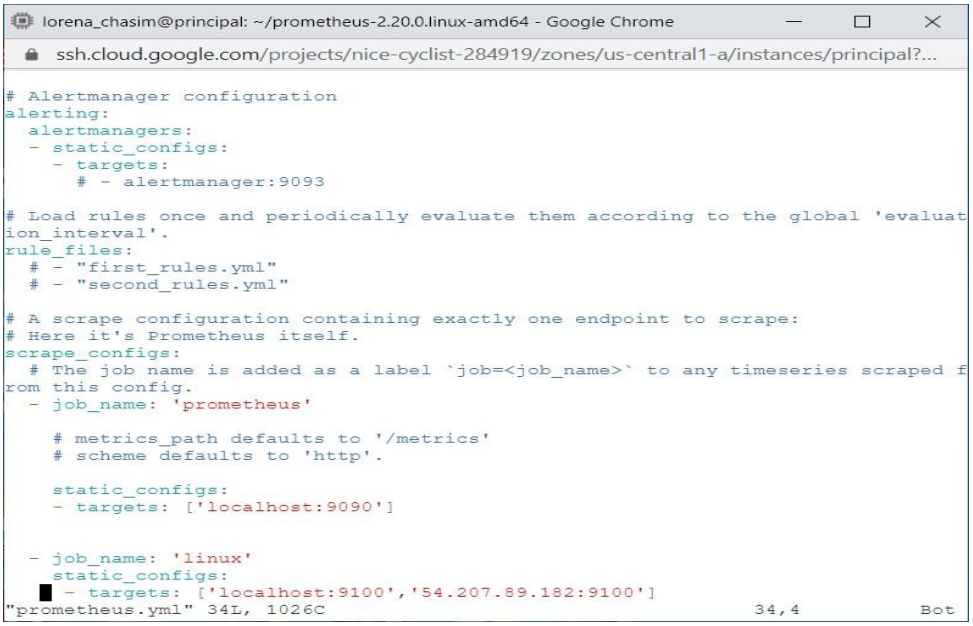


Figura 2.11 Configuración de “Jobs” en Prometheus

Formato para ingresar equipos a monitorear

```
- job_name: 'linux'
  static_configs:
   - targets: ['localhost:9100']
```

Una vez ingresados todos los equipos a monitorear se realiza el reinicio de Prometheus con el siguiente comando: pkill -HUP Prometheus, y se puede visualizar que ya se encuentra integrado Prometheus con Node_Exporter.

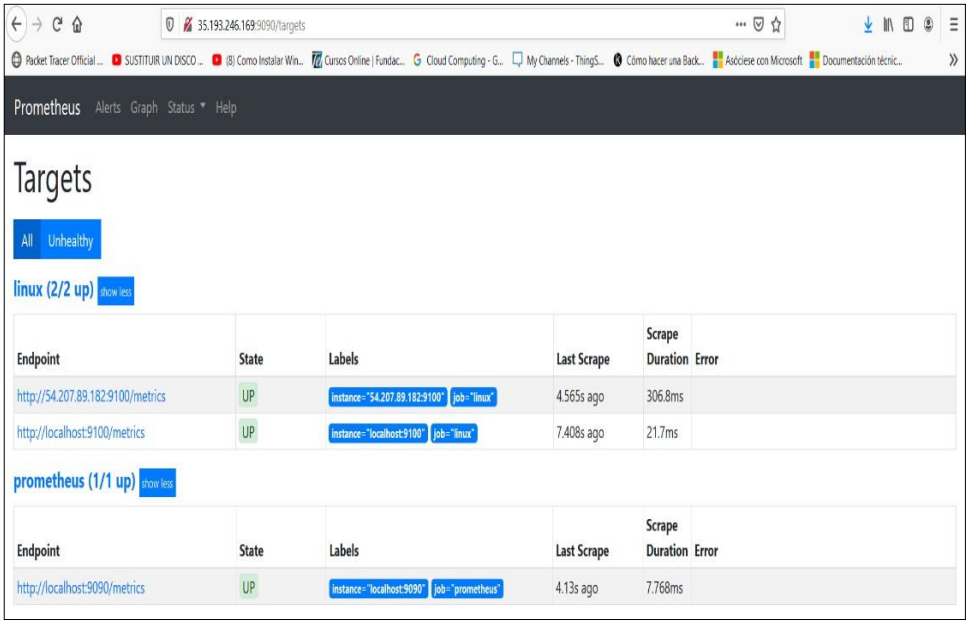


Figura 2.12 Visualización de “Jobs” y equipos integrados

Implementación de Grafana

En la línea de consola ejecutamos: wget https://dl.grafana.com/oss/release/grafana-6.3.6.linux-amd64.tar.gz

Descomprimir con el comando tar -zxvf grafana-6.3.6.linux-amd64.tar.gz y levantar el servicio por medio del comando “./bin/grafana-server”. Ingresamos a la dirección IP y haciendo uso del puerto 3000 para

corroborar el levantamiento del servicio, lo verificamos de la siguiente manera:

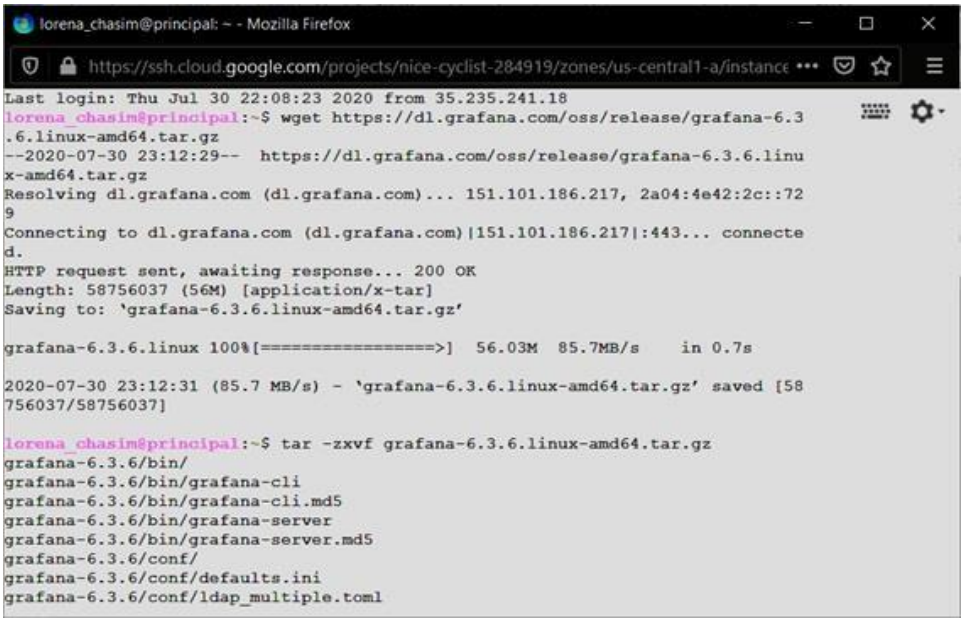


Figura 2.13 Extracción y ejecución de Grafana

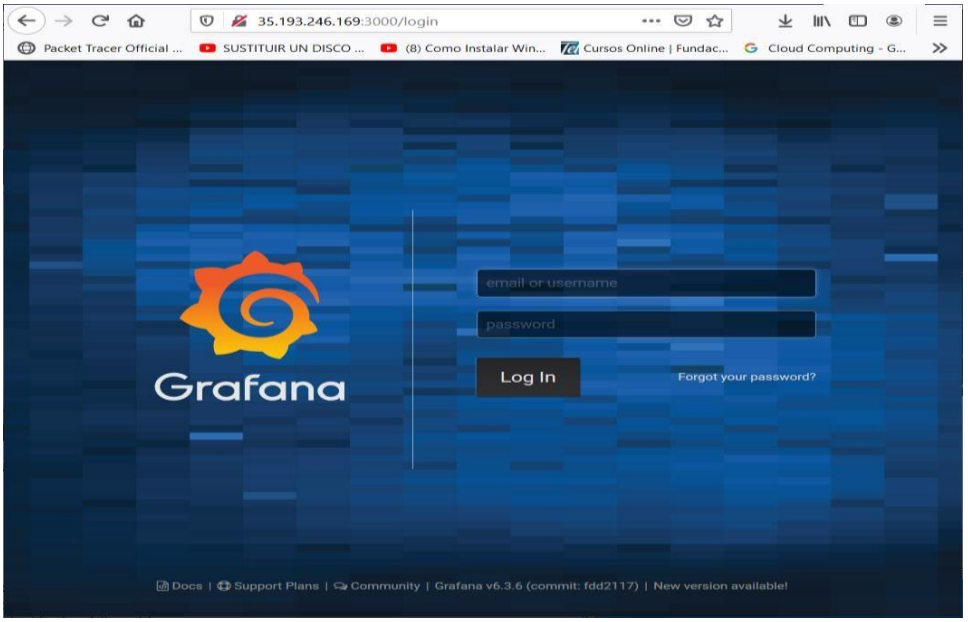


Figura 2.14. Inicio de Grafana

El usuario y clave “admin” ingresamos a la administración de Grafana.

Integración de Grafana con Prometheus

Luego de ingresar con las credenciales descritas anteriormente se procede a integrar Prometheus con Grafana ingresando a la opción de “CREATE - DASHBOARD”

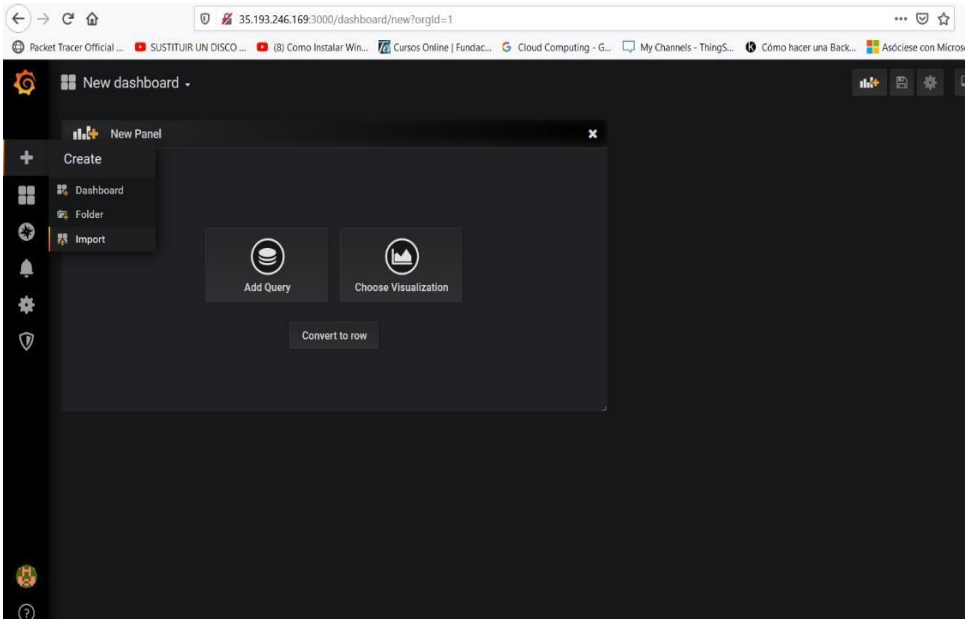


Figura 2.15 Creación de Dashboards

Se realiza la selección de Prometheus.

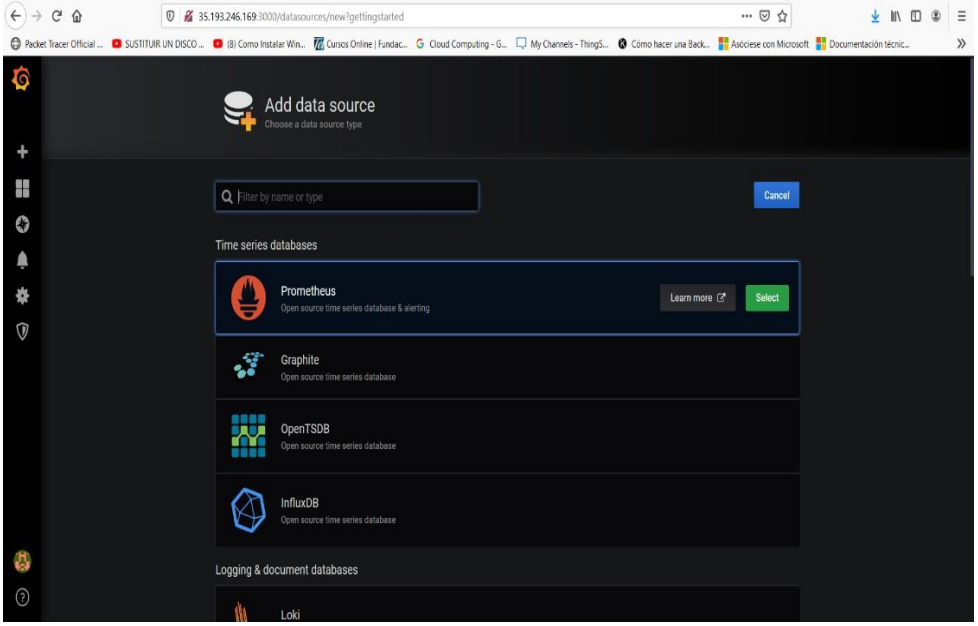


Figura 2.16 Selección de Prometheus

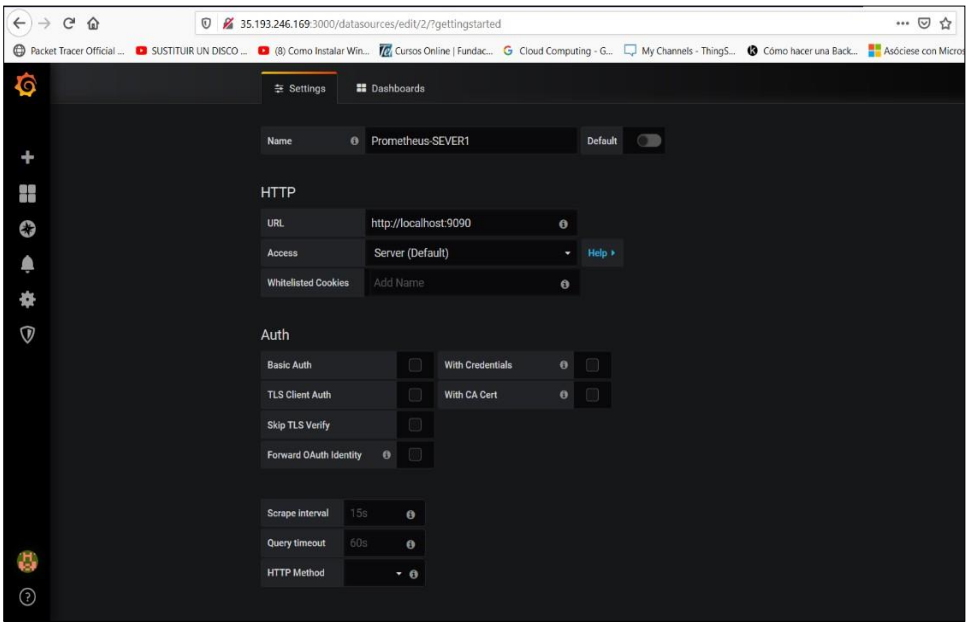


Figura 2.17 Configuración de parámetros de DataSource

Se le asigna un nombre al Dashboards, se selecciona la URL y se procede a guardar.

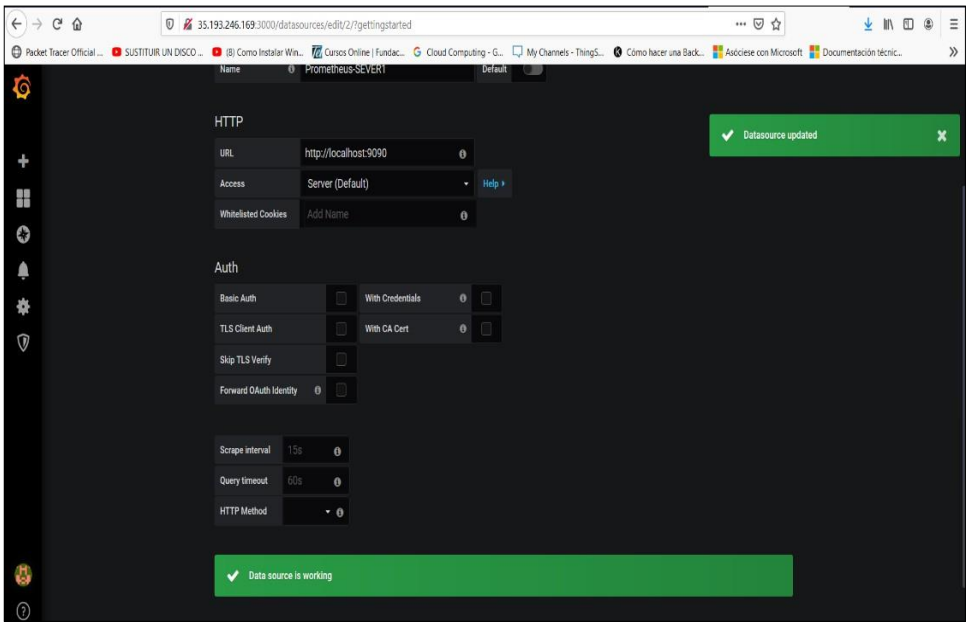


Figura 2.18 Información completa en DataSource

Existen un sin número de estilos de Dashboards (son diferentes tipos de visualizaciones de la información a presentar), para efectos de este proyecto se han utilizado el 1860 y 450 donde se los puede importar de la siguiente manera:

Importación de Dashboard

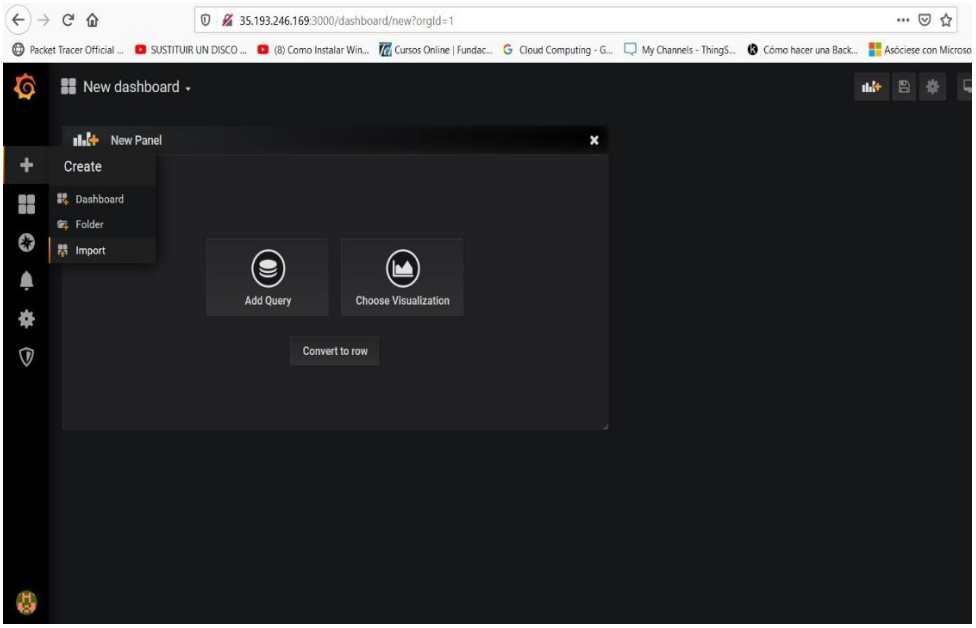


Figura 2.19 Importación de Dashboards

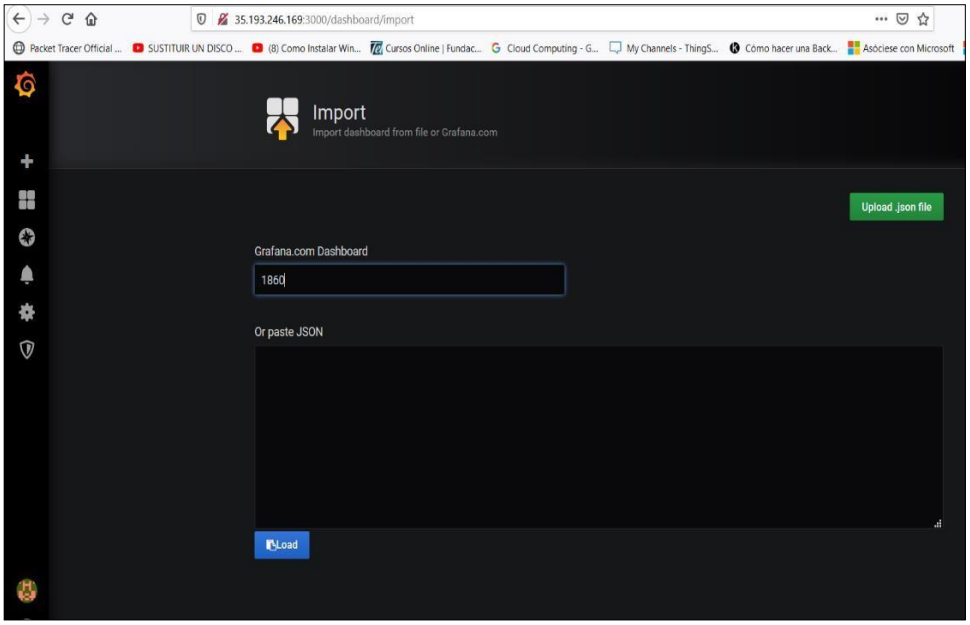


Figura 2.20 Selección de Dashboards

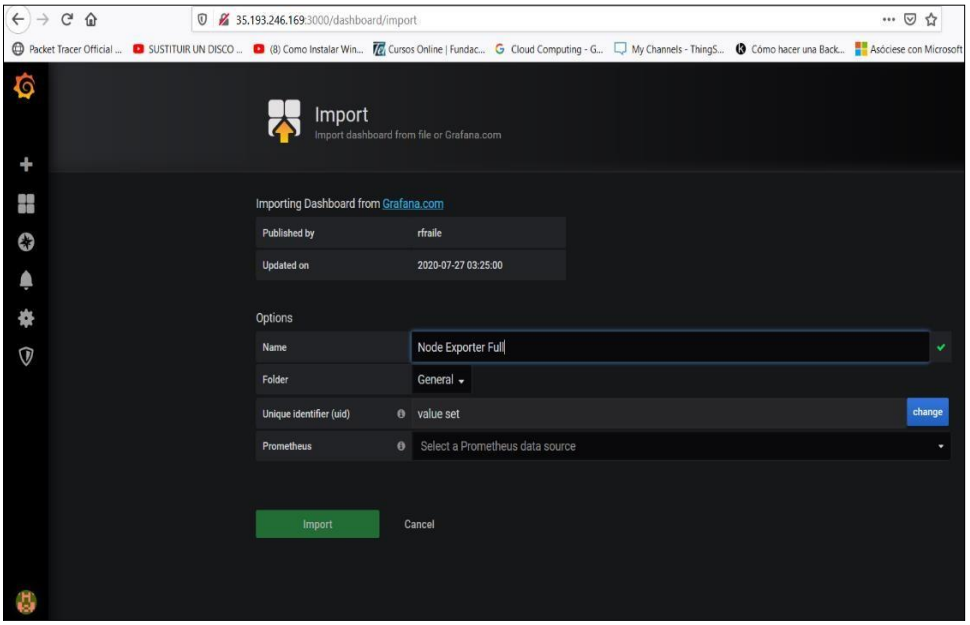


Figura 2.21 Importación de Dashboards

Todos los datos de los campos quedan por defecto a excepción del ultimo campo “Prometheus”, seleccionamos el nombre de nuestro DataSource creado anteriormente.

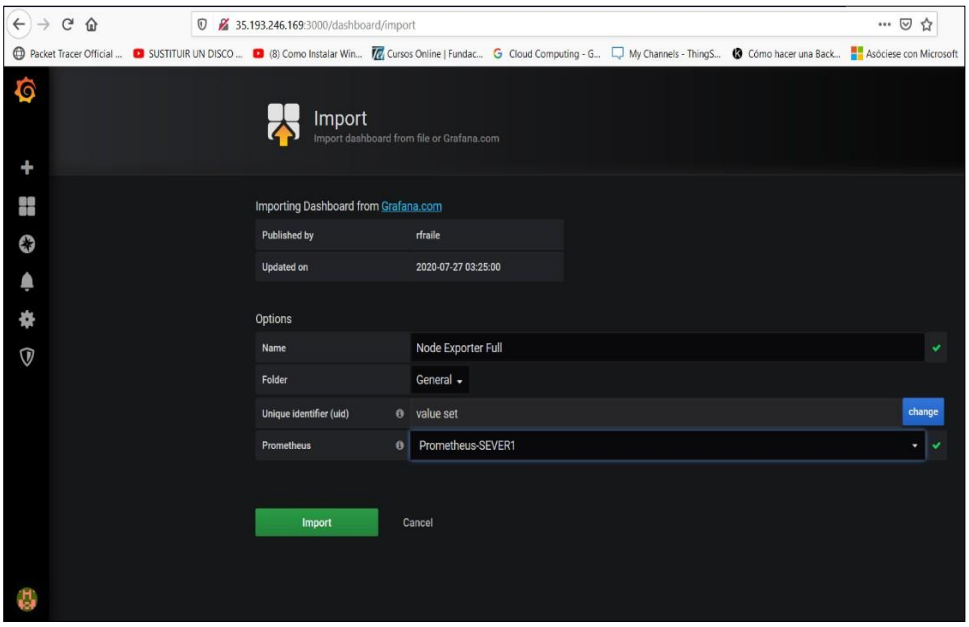


Figura 2.22 Integración del Dashboards con el DataSource

Configuración de alertas en Grafana enviadas vía correo

Para configurar las alertas hemos usado SMMTP, para el envío de correos a cuentas de Gmail entrelazado con Grafana, por lo cual se deben de seguir los siguientes pasos:

Ingresar a nuestra cuenta de correo Gmail

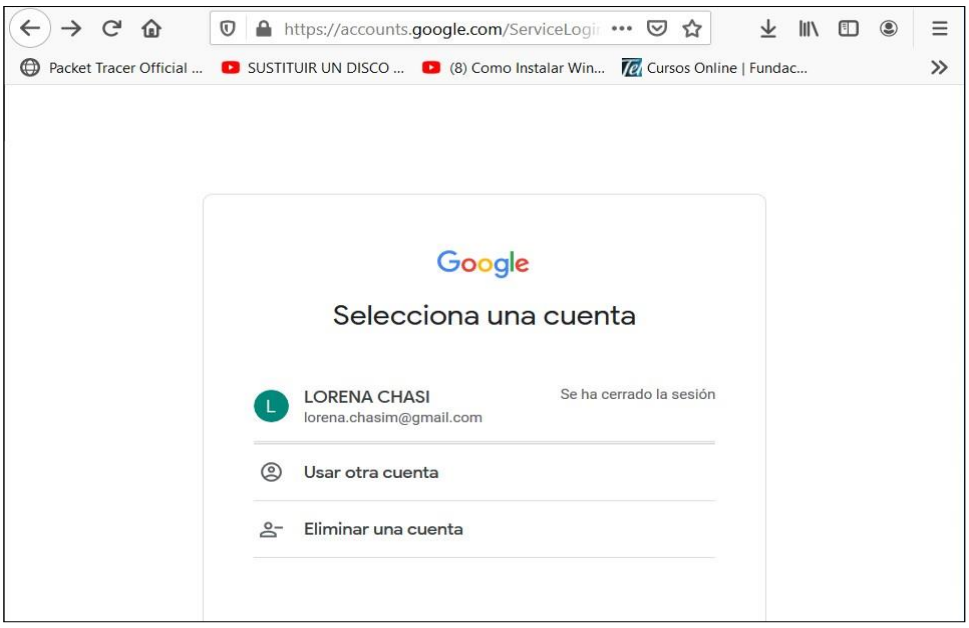


Figura 2.23 Ingreso a cuenta del correo electrónico Gmail

Ingresamos al siguiente link donde habilitaremos (A SI) la opción en nuestra cuenta de correo : https://myaccount.google.com/lesssecureapps

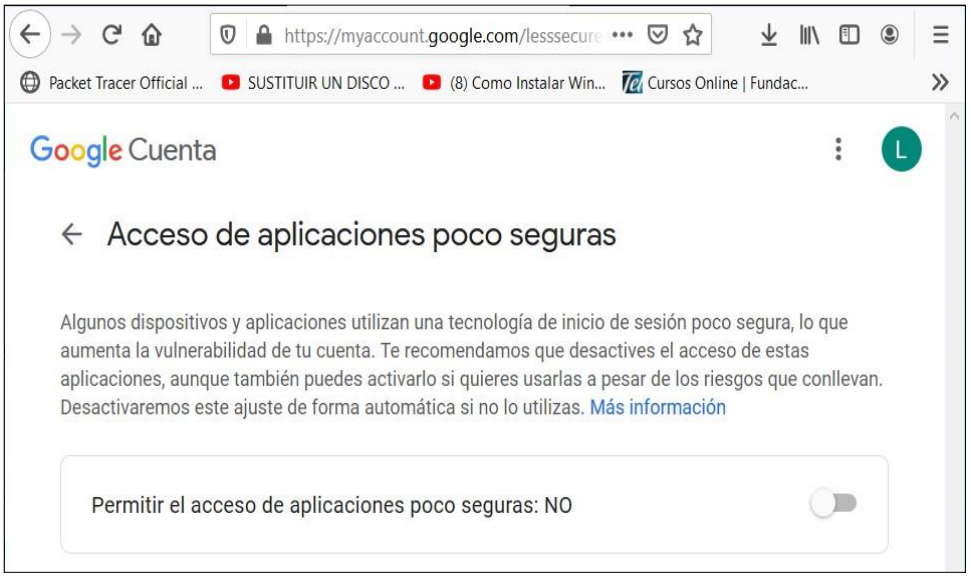


Figura 2.24 Acceso de aplicaciones poco seguras

Configuración SSMTP

Para lo cual ejecutamos las siguientes líneas en la consola

sudo apt-get update

sudo apt-get install ssmtp

Editamos el archive ssmtp.conf con el siguiente comando #sudo vi /etc/ssmtp/ssmtp.conf

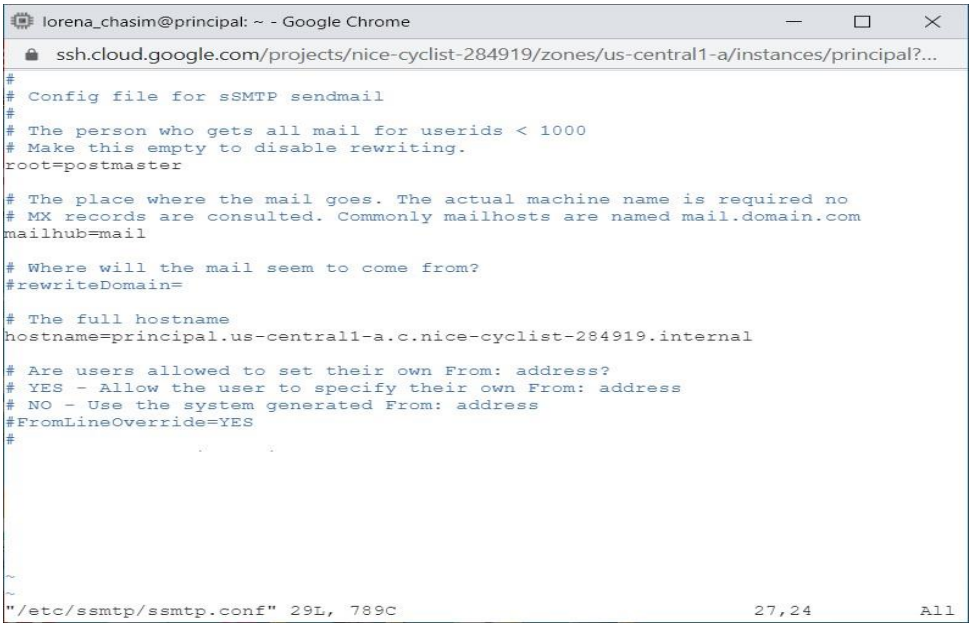


Figura 2.25 Instalación de SSMTP

Y Agregamos las siguientes líneas:

root=virtualcoin.videos@gmail.com

mailhub=smtp.gmail.com:587

FromLineOverride=YES

AuthUser=lorena.chasim@gmail.com

AuthPass=”CLAVE”

UseTLS=YES

UseSTARTTLS=YES

En la línea AuthPass= “se debe ingresar la contraseña de la cuenta de correo”

Figura 2.26 Configuración de SSMTP

Usaremos el siguiente comando para enviar un correo a manera de ejemplo

echo "E-Mail using the command-line" | ssmtp lorenachasi@gmail.com

Si el envío no tuvo éxito y vía consola muestra un mensaje de la siguiente manera:

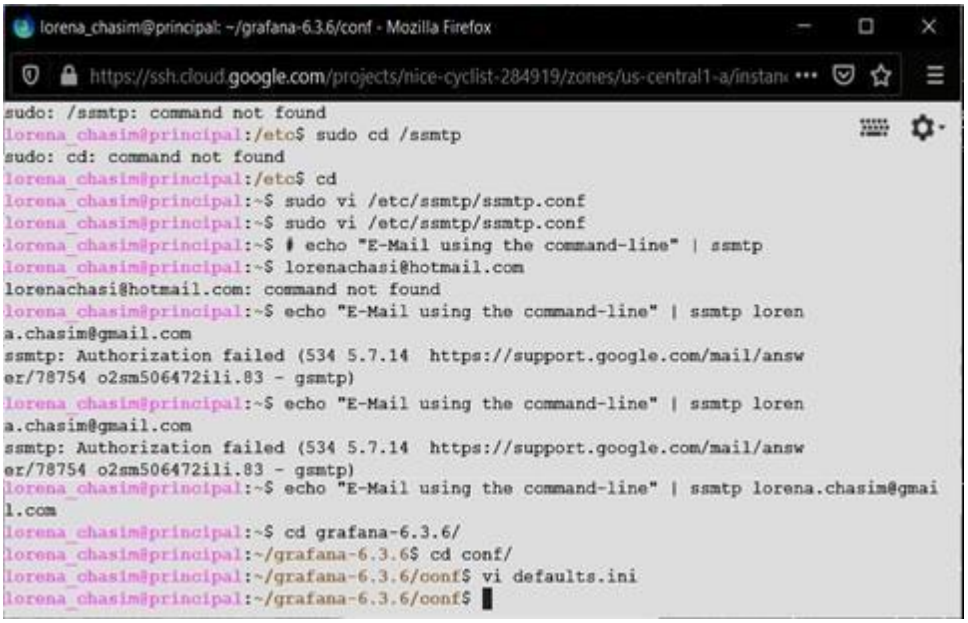


Figura 2.27 Error de envío de correo vía consola

SSMTP: Authorization failed (534 5.7.14 https://support.google.com/mail/answer/78754 d1sm484300ila.67 - gsmtp)

Para resolver este problema ingresamos a la siguiente URL: https://accounts.google.com/DisplayUnlockCaptcha

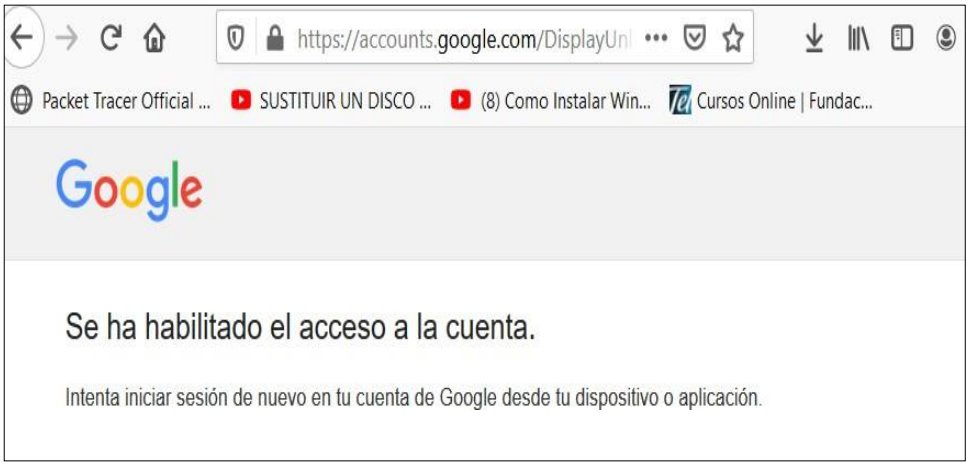


Figura 2.28 Habilitar acceso a la cuenta de Gmail

Volvemos a reenviar el correo y verificamos en la bandeja del correo electrónico la recepción del mismo.

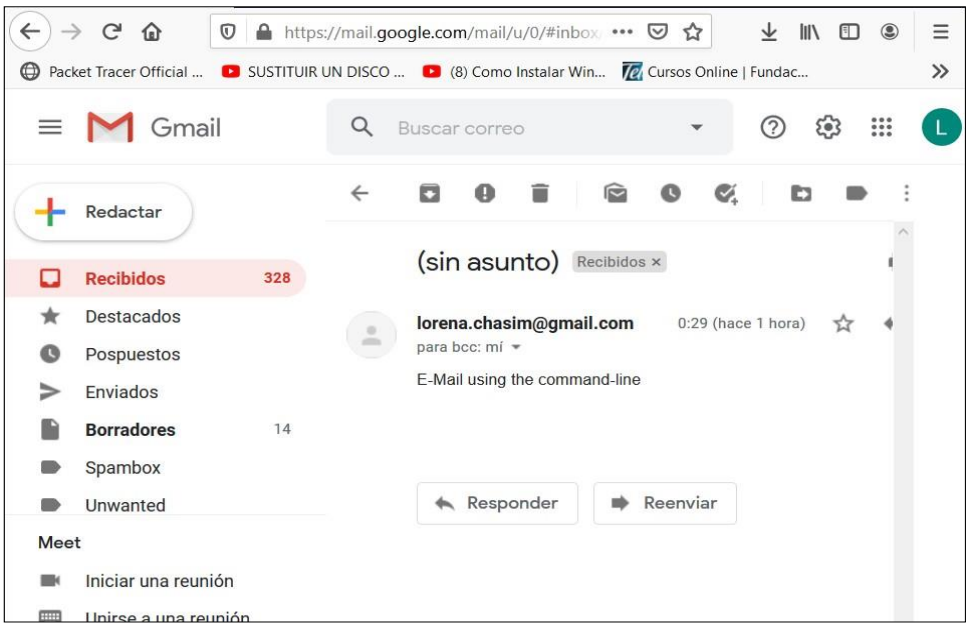


Figura 2.29 Recepción de correo de prueba

Configuración en Grafana para envío de correos

Para realizar esta configuración se debe hacer lo siguiente:

Editar el archivo defaults.ini por medio del comando vi conf/ defaults.ini

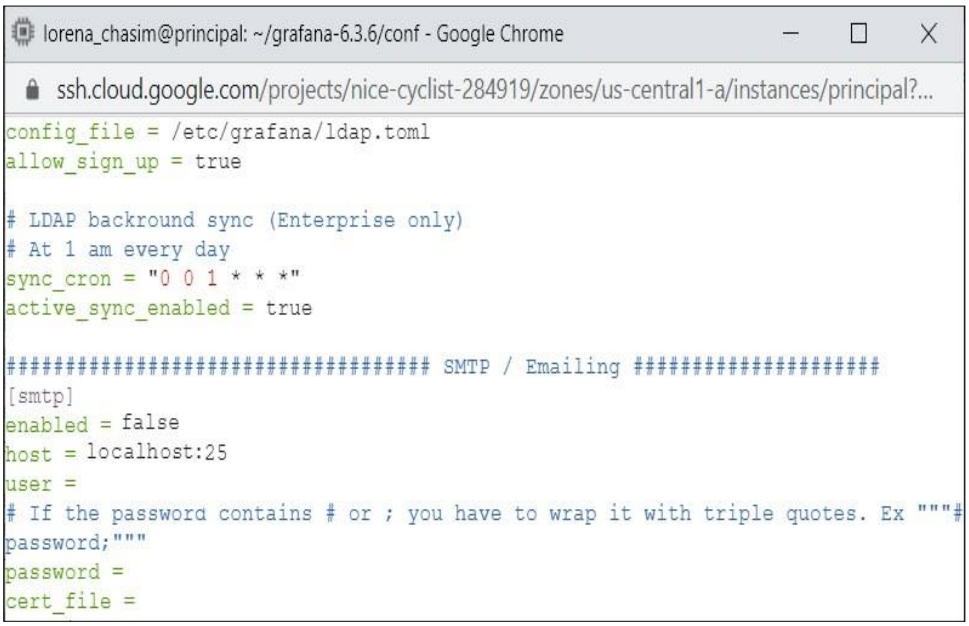


Figura 2.30 Archivo defaults original

En el archivo se debe de configurar las siguientes opciones:

[smtp]

enabled = true

host = smtp.gmail.com:587

user = Lorena.chasim@gmail.com

password = “CLAVE”

;cert_file =

;key_file =

skip_verify = true

from_address = Lorena.chasim@gmail.com

from_name = Grafana

EHLO identity in SMTP dialog (defaults to instance_name)

;ehlo_identity = dashboard.example.com

Adicionalmente es importante en la línea de password = “colocar la clave del correo electrónico”

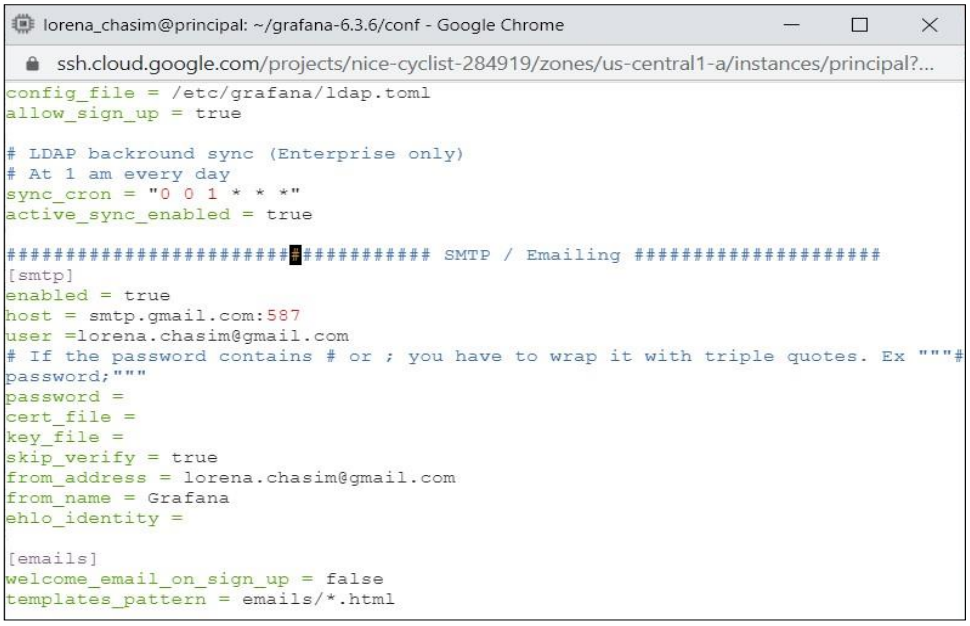


Figura 2.31 Archivo default.ini configurado

Una vez configurado los parámetros se procede a guardar y reiniciar Grafana con el comando “./bin/ grafana-server restart”

Configuración de alertas en Grafana

Para realizar esta configuración se debe hacer lo siguiente:

Damos clic en la campana (Alerting) y elegimos Notificacion Channel

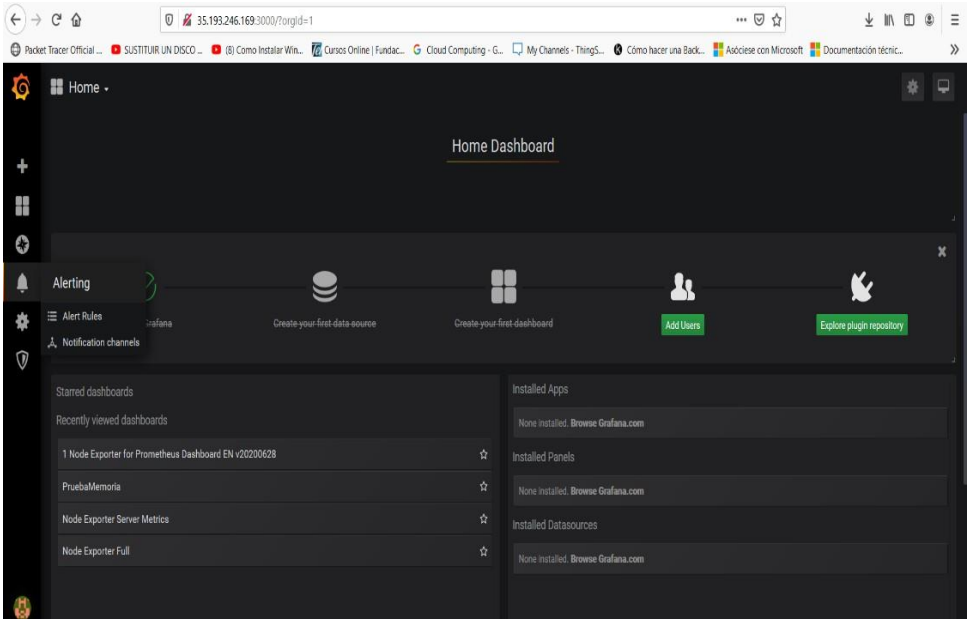


Figura 2.32 Creación de notificaciones

Elegimos notification channels y clic en Add channel

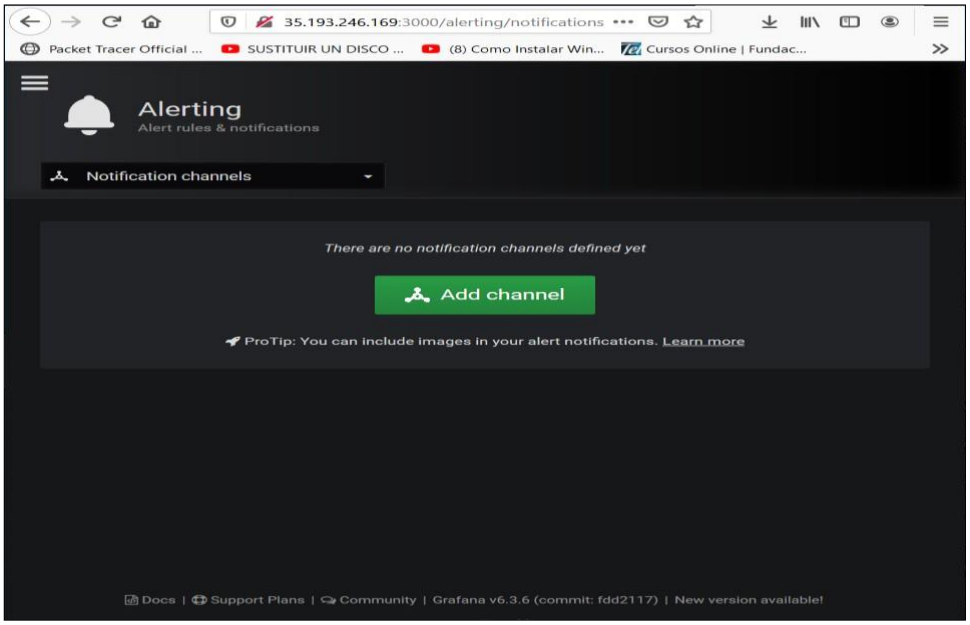


Figura 2.33 Agregación de Canal

Al elegir Add channel nos muestra la siguiente pantalla, se le asigna un nombre a la alarma, seleccionamos el “Type” y elegimos Email, activamos

default (send on all alerts), include image y en el casillero de Email addresses ingresamos el correo electrónico del cual ya fue configurado en los pasos anteriores.

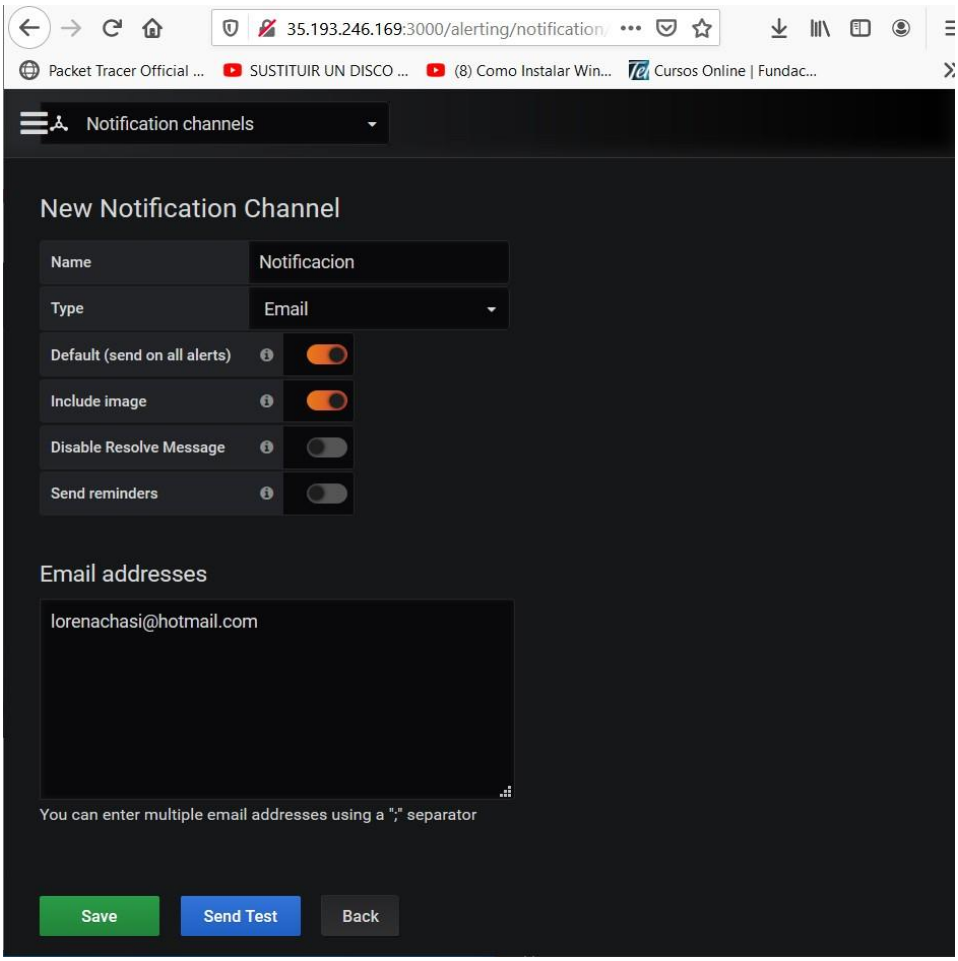


Figura 2.34 Creación de notificación en Grafana

Enviamos un test y si no existe algún problema (esta verificación se la hace vía consola), revisamos la bandeja de correo y comprobamos la recepción.

Configuración de alertas en telegram con Grafana

Para realizar esta configuración se debe hacer lo siguiente:

Elegimos notification channels y clic en Add channel

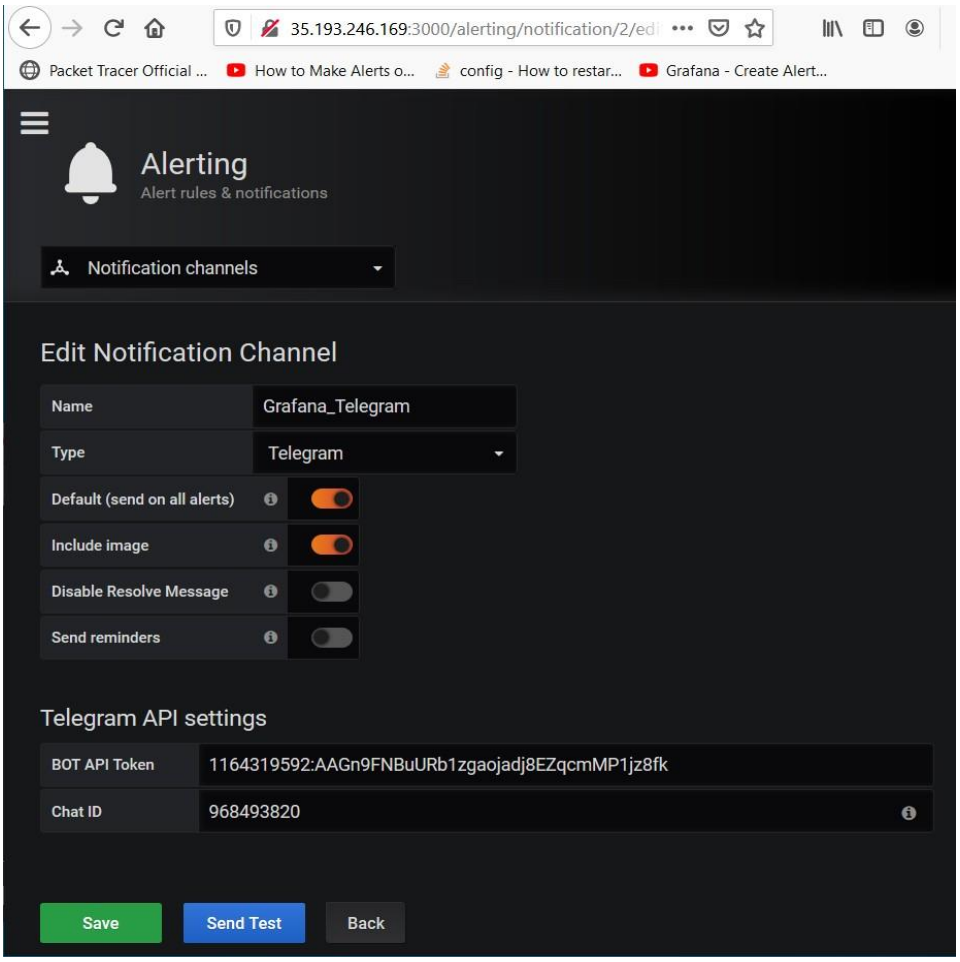


Figura 2.35 Creación de notificación de Telegram en Grafana

Los campos agregados en Telegram API settings se los obtiene de la siguiente forma:

En la aplicación Telegram buscamos a BotFather digitamos /newbot, luego nos indica que ingresemos el nombre del usuario bot deseado posterior nos genera un token para acceso a HTTP API, ese token es el que usaremos en el campo “BOT API Token”

Creamos un grupo donde se agregue al usuario bot en el paso anterior creado posterior enviamos un mensaje digitando la palabra bot en el usuario creado.

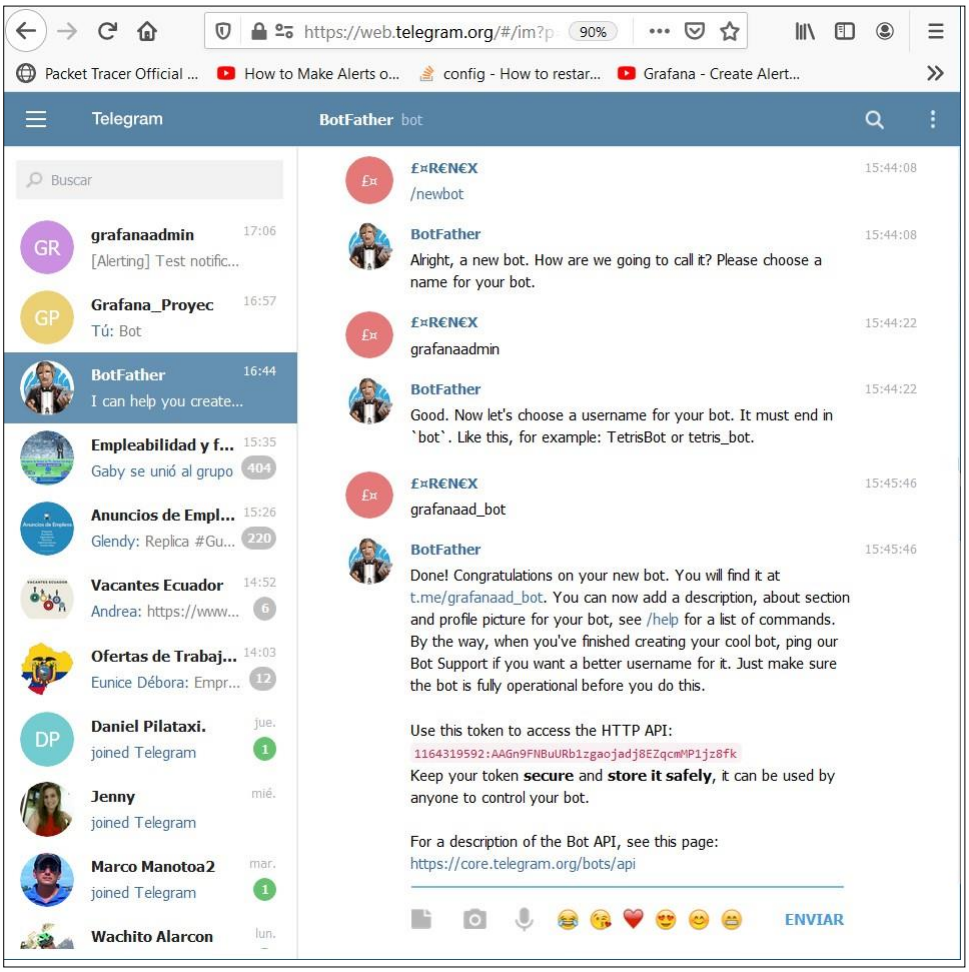


Figura 2.36 Creación de bot en Telegram

Nota: Diferenciar que grafanaadmin es el nombre del usuario creado en telegram y grafanaad_bot es el usuario para el bot [22].

Luego de realizar esto con el siguiente formato vamos al browser para poder obtener el chat ID :

https://api.telegram.org/bot<token>/METHOD_NAME

Para nuestro caso queda así

https://api.telegram.org/bot1164319592:AAGn9FNBuURb1zgaojadj8EZqcmMP1jz8fk/getupdates

Figura 2.37 Actualización de Token y obtención del Chat ID

Y visualizamos el chat ID “968493820” el mismo que colocamos en la creación de la alerta (ver Figura 2.35).

Tanto para Gmail como Grafana revisamos los mensajes recibidos (test de prueba).

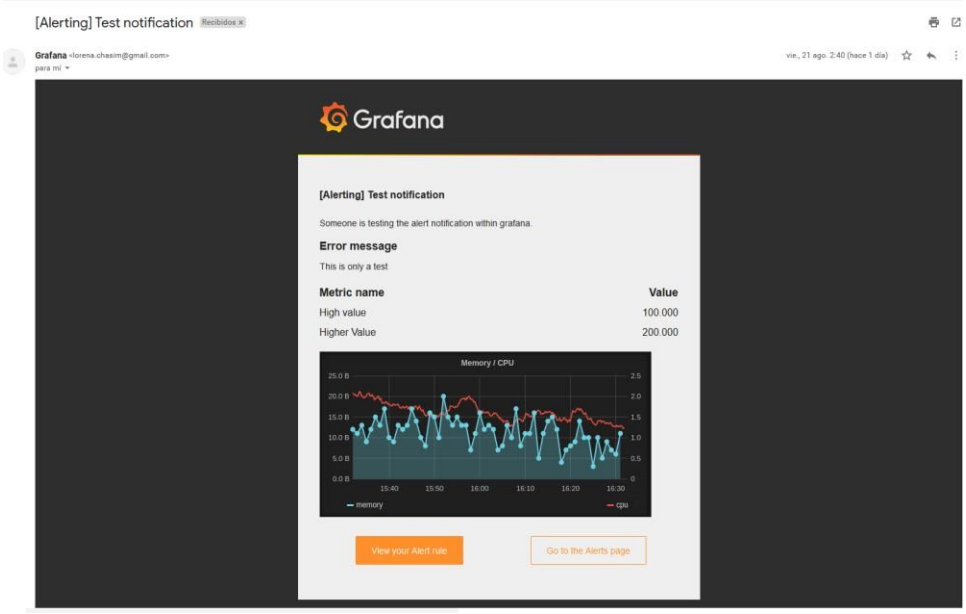


Figura 2.38 Recepción de alertas vía correo

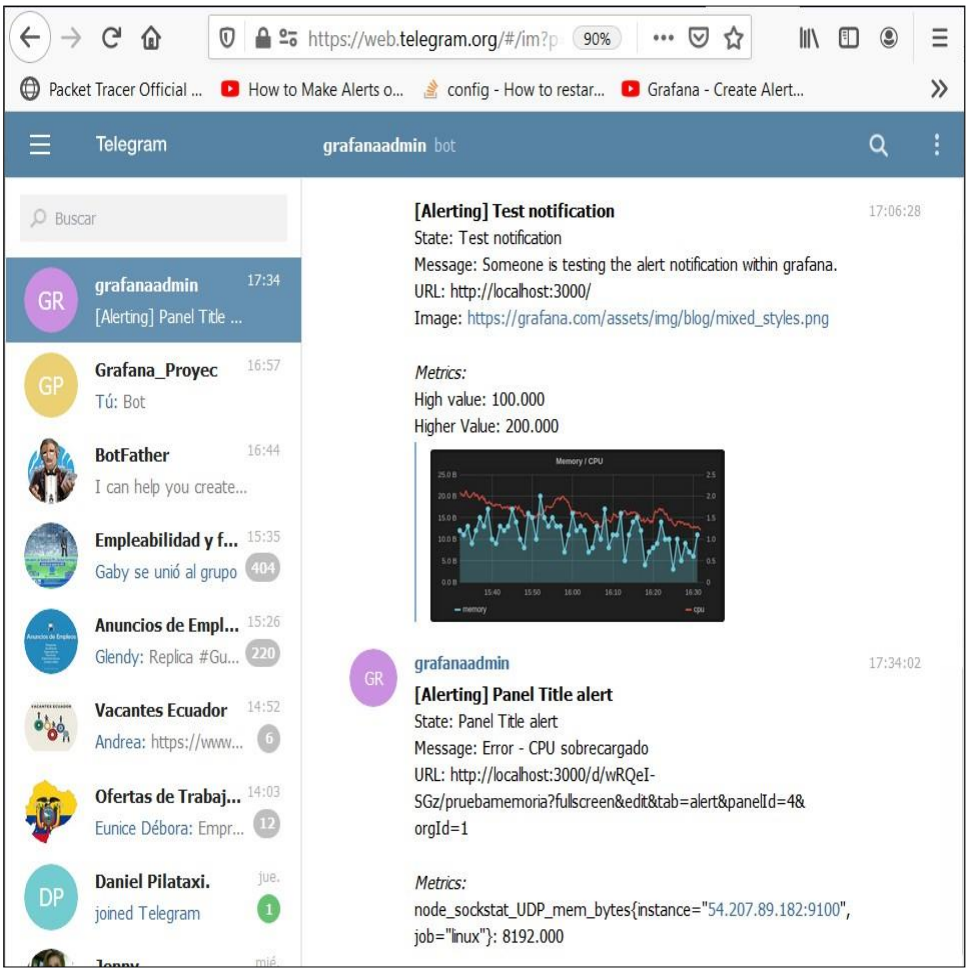


Figura 2.39 Recepción de alertas vía telegram

Configuración de métrica y envío de alertas vía correo y telegram

Es importante considerar este punto ya que se puede personalizar la data a ser monitoreada, es decir que se puede elegir varias métricas específicas para que solo de esas muestre información y de existir algún tipo de evento, este sea enviado vía correo o telegram dependiendo de las condiciones que el administrador de la red crea considerable.

Para esto realizamos lo siguiente:

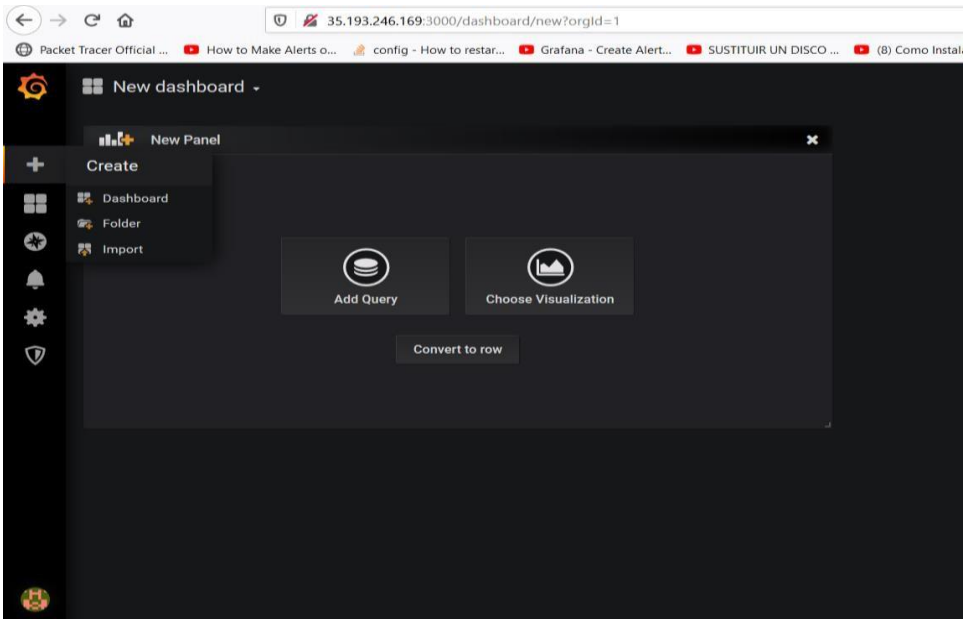


Figura 2.40 Añadir Consulta

Elegimos” Add Query”

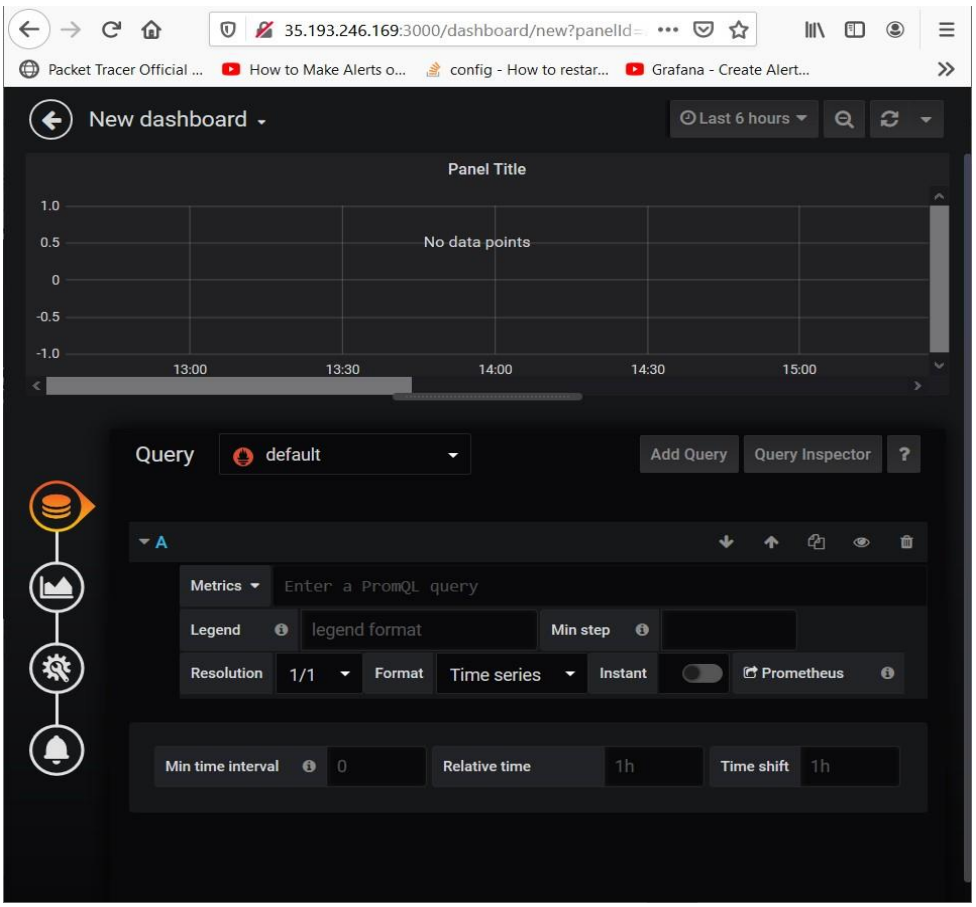


Figura 2.41 Configuración parametrizable de monitoreo

En el campo métrica se digita la palabra node y automáticamente se despliegan todas las métricas que deseemos visualizar tal como se muestra en la siguiente figura:

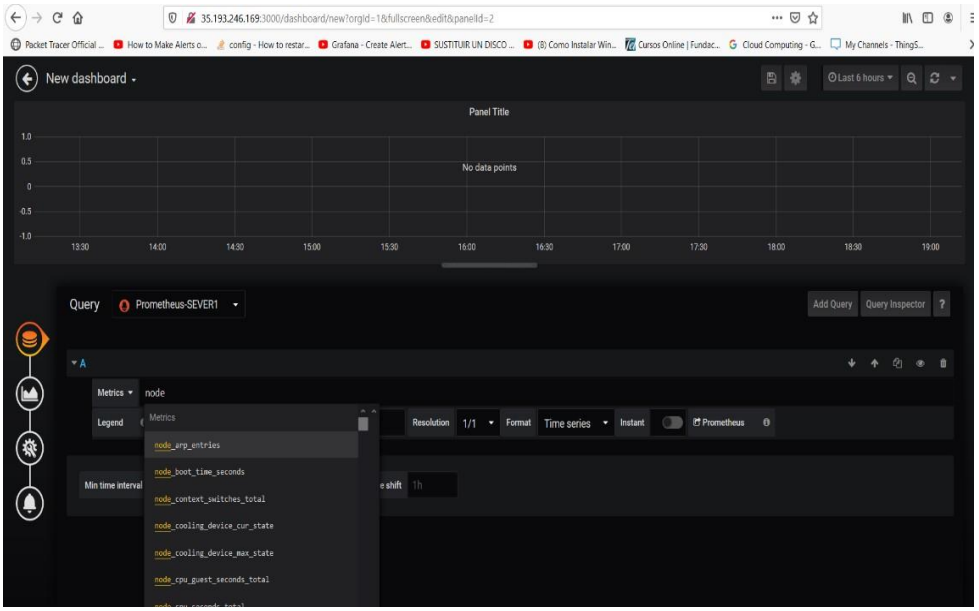
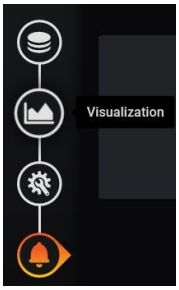


Figura 2.42 Configuración de Métrica

De lado izquierdo contamos con varias opciones que en base a los requerimientos del administrador de red pueden ser modificadas



Damos clic en la campana para crear la alerta

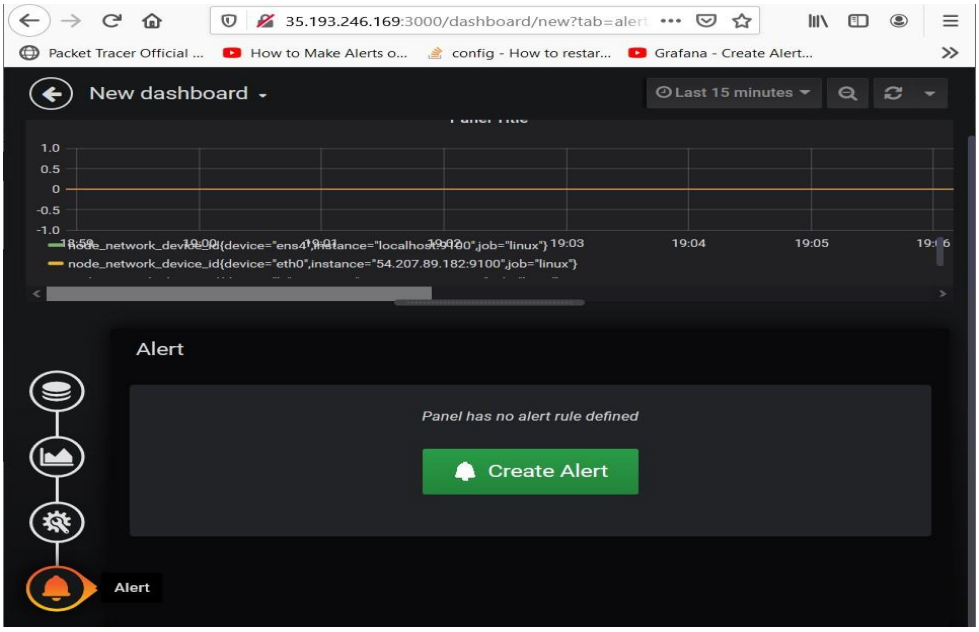


Figura 2.43 Creación de alerta parametrizable

Los siguientes parámetros son ingresados en base a las necesidades a monitorear. El cuadro que está marcado elegimos las “notifications” que fueron creadas con anterioridad y son las permiten enviar estas alertas ya sea por telegram o por correo electrónico (Figura 2.44)

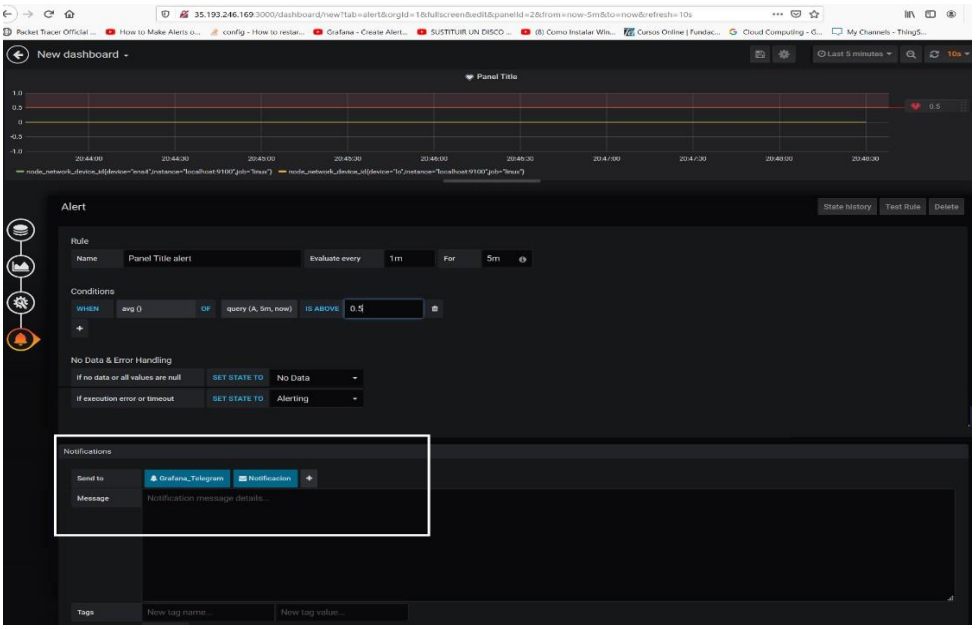


Figura 2.44 Configuración de alerta parametrizable

CAPÍTULO 3

3. ANÁLISIS DE RESULTADOS Y DISCUSIÓN

Esta sección se basa en la evaluación de los resultados obtenidos, a partir de los subprocesos antes descritos en la metodología, los mismos que son: Requerimientos, Planificación, Selección de Tecnología, Pruebas, Implementación de la arquitectura Prometheus (Física o Virtual), Implementación de agentes y Monitoreo.

3.1 Resultados

La siguiente Tabla 11. muestra los resultados obtenidos por cada uno de los subprocesos en conjunto con sus respectivas actividades, donde fueron todos exitosos.

#	Subprocesos	Actividad	Resultado	Logro obtenido
1	Requerimientos	Activos a monitorear	Actividades con éxito	Activos de información examinados
		Revisión de documentos		Activos determinados
2	Planificación	Análisis y Diseño de la Infraestructura de Prometheus	Actividad con éxito	Infraestructura definida
3	Selección de la tecnología	Compatibilidad de programas a usar	Actividad con éxito	Tecnología identificada
4	Pruebas	Pruebas del diseño	Actividad con éxito	Diseño de Pruebas
5	Implementación de la arquitectura Prometheus (Física o Virtual)	Implementación de Prometheus con Node_Exporter	Actividades con éxito	Integración configurada
		Implementación de Grafana		configuración realizada
		Implementación de Grafana con Prometheus		Integración configurada
		Implementación de Notificaciones y alertas		Servidor SSMTPy Telegram configurado

#	Subprocesos	Actividad	Resultado	Logro obtenido
6	Implementación de agentes	Configuración de envío de eventosvía correo y Telegram	Actividades con éxito	Agentes implementados
		Conexión de un Server externo por medio de Node_Exporter para ser monitoreado		Agentes conectados
7	Monitoreo	Verificación de recepción de correos de alertas	Actividades con éxito	Envío de información vía correo de alertas
		Visualización de gráficas en tiempo real según métricas establecidas		Visualización de graficas definidas según el Dashboards

Tabla 11. Resultados

Todas las actividades fueron realizadas con éxito, pero se evidenció que al momento de la recepción vía correo o telegram presentada por alguna eventualidad en el monitoreo, la gráfica como tal del evento no se visualiza (En el correo electrónico).

Se logró obtener los procedimientos, configuración e instalación de los aplicativos con la finalidad de definir la mejor alternativa que permita el correcto monitoreo de los aplicativos.

3.2 Discusión

En este punto se puede acotar por cada subproceso lo siguiente:

Requerimientos: Este punto es muy importante, permitió determinar que activos son los más cruciales para una organización

Planificación: Basándonos en las metodologías Top Down y MSF, este punto nos permitió definir las características de la infraestructura y lo que se desea implementar.

Selección de la tecnología: La mayor parte de información para realizar esta implementación se obtuvo de las paginas oficiales de cada fabricante. Información importante para llevar a cabo este proyecto.

Pruebas: En este punto las pruebas del diseño fueron adaptadas a la arquitectura de Prometheus

Implementación de la arquitectura Prometheus (Física o Virtual): La integración de cada componente fue importante permitió continuar con la implementación de Prometheus.

Implementación de agentes: La implementación de estos agentes permitió monitorear un servidor externo y obtener información vía correo y Telegram de alertas presentadas en el monitoreo

Monitoreo: Para este subproceso se consideran las referencias de las paginas oficiales, que permitieron realizar una correcta integración de cada agente para poder visualizar la información en tiempo real de manera gráfica.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- ✓ La Propuesta metodológica y simulación de la implementación de un Sistema (Prometheus) de código abierto, que permite realizar el monitoreo de aplicativos, fue el resultado del trabajo de titulación.
- ✓ La implementación de Prometheus permitirá monitorear servidores, donde por medio de gráficas se podrá visualizar en tiempo real cada servicio monitoreado (CPU, memoria, procesador entre otros), enviando un mensaje de alerta vía correo o telegram de alguna eventualidad suscitada en el monitoreo.
- ✓ Prometheus no solo permite monitorear el estado de un servidor, sino que cuenta con un sin número de métricas que permiten obtener diferentes tipos de data de otros aplicativos, como es el caso statsd_exporter, consul_exporter, mysqld_exporter, entre otros; cada una descritas en el cuadro de Métricas de Prometheus.
- ✓ Las metodologías Top Down y MSF, permitieron conocer los procesos de las actividades para implementar este proyecto.
- ✓ La instalación de los agentes permitió la recepción de notificaciones vía correo y telegram de eventuales presentadas en el monitoreo
- ✓ La implementación del sistema de monitoreo Prometheus se adapta fácilmente a organizaciones pequeñas, medianas y grandes empresas.

4.2 Recomendaciones

- ✓ Se recomienda instalar la métrica de Prometheus a usar en el cliente basándose en lo que el administrador de la red desea monitorear, con el fin de poder extraer la información y presentarla por medio de Prometheus.
- ✓ Se recomienda crear alertas parametrizables según los datos a monitorear y dichas alertas sean enviadas por algún tipo de medio ya sea correo u otro tipo soportado por el sistema.

BIBLIOGRAFÍA

[1] Prometheus. [En línea]. Available: https://prometheus.io/docs/introduction/overview/.

[2] Prometheus - Node Exporter, [En línea]. Available: https://prometheus.io/docs/guides/node-exporter/.

[3] Grafana, [En línea]. Available: https://grafana.com/docs/grafana/latest/.

[4] Martínez, Santos, «Que es el Open Source y como puede ayudarte?,» [En línea]. Available: https://openexpoeurope.com/es/open-source-puede-ayudarte/.

[5] Zabbix, [En línea]. Available: https://www.zabbix.com/.

[6] Solaris, «Solaris,» [En línea]. Available: https://www.solarwinds.com/es/.

[7] Cacti, «Cacti,» [En línea]. Available: https://www.cacti.net/download_cacti.php.

[8] Nagios. [En línea]. Available: https://www.nagios.org/.

[9] PandoraFms, [En línea]. Available: https://pandorafms.org/es/.

[10] P. M. &. G. T. Mell, «The NIST Definition of Cloud Computing. National Institute of Standards & Technology,» 2011. [En línea]. Available: http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf.

[11] Gartner, «Reviews for Cloud Infrastructure as a Service (IaaS),» [En línea]. Available: https://www.gartner.com/reviews/market/public-cloud-iaas.

[12] Iso 27000 - SGSI, [En línea]. Available: http://www.iso27000.es/sgsi.html. [Último acceso: 01 08 2020].

[13] J. Jimeno, «El ciclo de Deming o círculo PDCA,» 2013. [En línea]. Available: https://www.pdcahome.com/5202/ciclo-pdca/.

[14] C. Gómez, Proyectos Factibles, Valencia: Editorial Predios, 2000.

[15] F. Arias, El proyecto de investigación: Introducción a la metodología científica. (5º.ed.), Caracas - Venezuela, 2006.

[16] L. y. Otros, Proyectos Factibles - Metodología, Valencia, 2002.

[17] D. K. A. Peter Smith, «Methodology for Creating Methodologies,» [En línea]. Available: http://pcrest.com/research/2_4_16%20Methodology%20for%20Creating%20Methodologies.pdf.

[18] «Metodología Top Down,» [En línea]. Available: https://www.scribd.com/doc/242870887/2-Metodologia-Top-Down-espanol-pdf .

[19] «Descripcion de la Metodologia: MSF,» [En línea]. Available: https://sites.google.com/site/aessl13g314/practica-2/2-1.

[20] Google Cloud, «Documentación Google Cloud,» [En línea]. Available: https://cloud.google.com/docs. [Último acceso: 01 08 2020].

[21] Amazon Web Services, «Documentación AWS,» [En línea]. Available: https://docs.aws.amazon.com/.

[22] Telegram, «Telegram Bot,» [En línea]. Available: https://core.telegram.org/bots.