ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"IMPLEMENTACIÓN EN LA NUBE DE UN SISTEMA DE MONITOREO DE EVENTOS DE FALLAS PARA INFRAESTRUCTURA DE REDES Y DE SEGURIDAD INFORMÁTICA UTILIZANDO LA INTEGRACIÓN DE ZABBIX, GRAFANA Y ZAMMAD"

EXAMEN DE GRADO

Previo a la obtención del Título de:
MAGISTER EN TELECOMUNICACIONES

Presentado por:
JOSE STALYN CASTRO GONZALEZ

GUAYAQUIL - ECUADOR
AÑO 2020

AGRADECIMIENTO

Primeramente, a DIOS que me ha permitido llegar a esta instancia de un nuevo logro profesional.

A mis padres que a pesar de su ausencia física dejaron una huella invaluable de responsabilidad, dedicación y honestidad.

A mis hermanos, María, Juan y Yessenia, que siempre me han apoyado en las acciones emprendidas y por contar con ellos en cada momento bueno o malo de mi vida, y que me dieron el ánimo suficiente para no claudicar en el momento doloroso de la muerte de nuestro querido padre.

Quiero expresar mi agradecimiento al Ing. Iván Garcia Criollo, un viejo amigo, que me presto una valiosa ayuda en la consecución de este proyecto brindándome recomendaciones y sugerencias al poseer una vasta experiencia en servidores Linux y aplicaciones de código abierto.

Al Ing. Javier Togra Alvarado, gerente de la empresa Segurinfor S.A. y también un gran amigo desde la época de aulas, quien me facilito para la simulación el monitoreo en tiempo real de agentes activos en Internet como fueron su servidor de red y firewall de seguridad, así como permitirme utilizar el nombre de su dominio segurinfor.net para la máquina virtual utilizada en la simulación.

A todos mis compañeros de aula, pero en especial con los que arme la mayoría de las veces grupo a lo largo de este proceso como son Diego Pinguil, Jair Torres, José Flores y Eduardo Puertas que desde el propedéutico siempre me dieron apoyo y ánimo, en especial en aquel momento doloroso de mi vida como fue la muerte de mi amado padre.

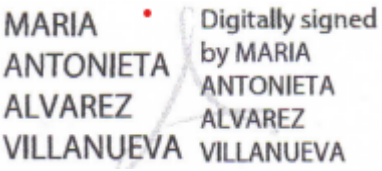
Y, por último, agradezco a los profesores de la MET quienes dejaron una huella de nuevos conocimientos que me han permitido actualizarme en un mundo de tecnología que había abandonado y me hicieron volver con ímpetu nuevamente a la vida profesional y laboral en este campo extraordinario de las Telecomunicaciones.

DEDICATORIA

Dedico a DIOS este logro, que a pesar de las diversas adversidades que se me presentaron me dio la fortaleza para seguir adelante

Con todo mi amor a mis padres amados, Pascual y Fanny, que desde el cielo guían hoy mis pasos en este mundo, por ellos que fueron mis pilares para ser lo que soy, por su infinito amor y apoyo, como me hubiera gustado tenerlos presente, pero sé que desde el cielo sonríen por el logro alcanzado, aún recuerdo las palabras de mi mami diciéndome que vuelva a estudiar, a mi papi que me pedía que vaya a la Universidad cuando su estado de salud decaía día a día. Con lágrimas en los ojos esto es para Uds. mis viejitos queridos, deseo con el alma haberles podido regalar una sonrisa de satisfacción allá en ese mundo que llegado su momento espero ser digno de alcanzar para volverlos abrazar.

TRIBUNAL DE EVALUACIÓN



MSc. Verónica Soto Vera PROFESOR EVALUADOR	PhD. María Antonieta Alvarez 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"

Ing. José Stalyn Castro González

RESUMEN

El objetivo del presente proyecto es integrar, implementar y simular una solución de monitoreo, alertas y presentación gráfica utilizando microservicios.

Para la demostración se ha elegido un sistema operativo y aplicativos de código abierto, así tenemos que como plataforma de gestión y monitoreo la herramienta a utilizar es Zabbix, para la presentación gráfica de resultados se eligió Grafana y como un ERP de atención de casos basado en tickets se utilizó Zammad.

La integración de estos sistemas se la realizo usando el concepto de microservicios instalados sobre contenedores basados en Docker sobre un sistema operativo Linux CentOS versión 7 implementado y configurado en la nube de Google.

INDICE GENERAL

INDICE DE FIGURAS

ABREVIATURAS

SNMP	Simple Network Management Protocol
IPMI	Intelligent Platform Management Interface
JMX	Java Management Extensions
MIB	Management Information Base
JSON	JavaScript Object Notation
API	Application Programming Interface
HTTP	Hypertext Transfer Protocol
YAML	YAML Ain't Markup Language (acrónimo recursivo)
ERP	Enterprise Resource Planning
ICMP	Internet Control Messages Protocol
SLA	Service Level Agreement

CAPITULO 1

1. INTRODUCCIÓN

1.1. Descripción del problema

Hoy en día mantener la disponibilidad de los servicios a los clientes es primordial para el éxito de cualquier negocio. Existen empresas que poseen arquitecturas complejas de comunicación en donde se integran varios aplicativos, sistemas operativos, bases de datos, o una red compleja con muchas sucursales en donde para detectar una falla o anomalía de algún elemento de la infraestructura de operación no poseen una plataforma adecuada para alertar sobre este tipo de eventos imprevistos, provocando el malestar o inconformidad a los usuarios de un determinado servicio lo que genera pérdida de imagen, incumplimiento de SLA, multas de entes estatales y otros.

Actualmente existen en el mercado softwares de gestión, monitoreo u administración de tipo comercial como PRTG o Solarwind que resultan muy costosos en su adquisición por que requieren de licencia renovable cada cierto periodo de tiempo, así como también de gastos adicionales cuando se desea incrementar el número de dispositivos o aplicativo debido a esto algunas empresas no se encuentran en la capacidad financiera de invertir en estos sistemas por lo que normalmente buscan sistemas más económicos o de tipo gratuito o código abierto.

1.2. Descripción de la propuesta

Para solucionar el problema descrito se presenta como alternativa la integración de varias plataformas de código abierto con propósito diferente que al unirlas forman un sistema coordinado en respuestas a eventos de fallas o indisponibilidades. Los aplicativos que se plantea integrar son herramientas que han sido desarrolladas para cumplir un objetivo o meta definida, que explicamos a continuación.

Zabbix: Un sistema de gestión para infraestructura de hardware y software incluyendo sistemas de networking (routers o switches), de seguridades informática(firewalls), servidores, máquinas virtuales, aplicativos de internet (correo, web), bases de datos, que permite administrar y monitorear de manera integral los diferentes elementos de una red de datos. [1]

Se puede observar en la Figura 1.1 un resumen de las capacidades que tiene Zabbix para monitorear.

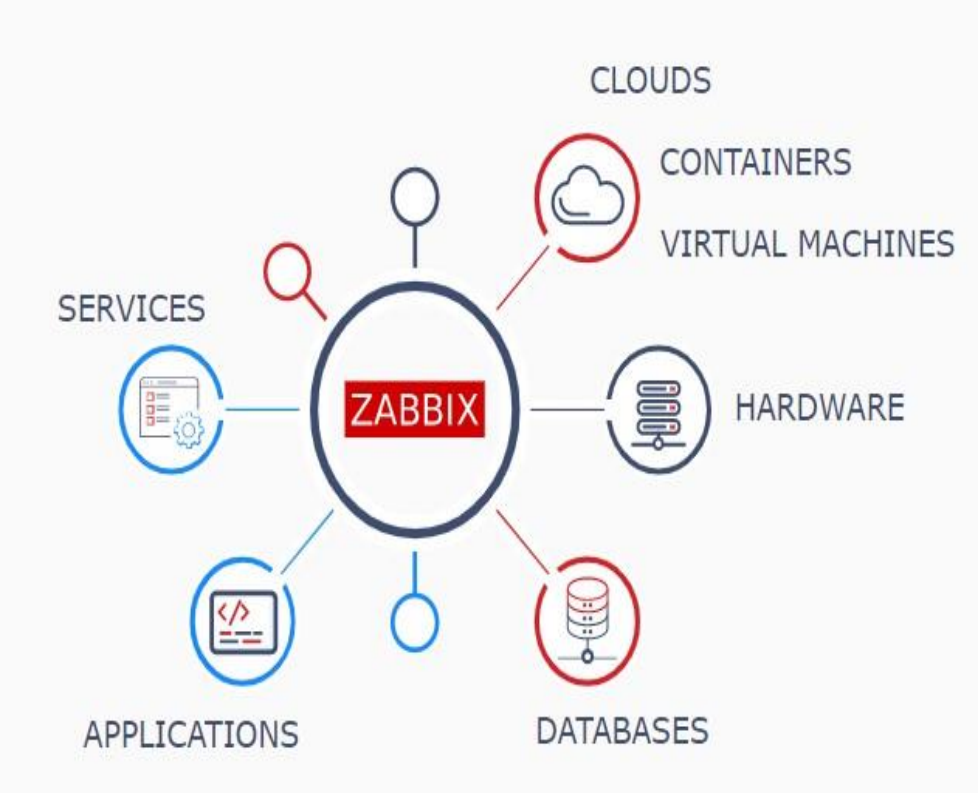


Figura 1.1. Características de Gestión de Zabbix

Grafana: Un poderoso presentador de resultados gratuito que complementa a Zabbix en el sentido que sus reportes gráficos obtenidos son más ejecutivos y se pueden personalizar de acuerdo con las necesidades del usuario final.[2]

Zammad: Es un sistema de tickets basado en la web de código abierto y con todas las funciones para el servicio de asistencia técnica o atención al

cliente. Se integra con una multitud de funciones para manejar la comunicación con el cliente a través de varios canales, como redes sociales, chat en vivo, correos electrónicos y teléfono. El objetivo de usar Zammad es que al generarse una alerta en Zabbix inmediatamente genere un ticket de atención que será derivado al responsable del servicio para la solución del problema, esto permitirá primeramente un correcto registro de los cambios de estado de los diversos elementos supervisados y la notificación de estos eventos para ser atendidos.[3]

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar e Implementar un sistema integrado de monitoreo y gestión de infraestructura de redes y seguridad informática de código abierto en la nube que nos muestre de manera personalizada los resultados de gestión de las plataformas y que además permita escalar de manera proactiva ticket de atención de servicio al presentarse un cambio de estado en los elementos o aplicativos gestionados.

1.3.2. Objetivos Específicos

- Implementar una herramienta de gestión y monitoreo de infraestructura de redes, servidores y aplicaciones.
- Implementar la configuración de un sistema de tickets de servicios para asignar un requerimiento de soporte alertando sobre novedades que se presenten en los sistemas o dispositivos a monitorear.
- Presentar resultados visuales del monitoreo de una manera ejecutiva y entendible para el usuario.
- Integrar la plataforma de gestión con el sistema de notificación y presentación de resultados.

1.4. Alcance del Proyecto

- Se presentará la implementación de la solución en un servidor en alguna de las plataformas de la nube como (Google, AWS o Azure), en donde se

instalará las plataformas anteriormente descritas (Zabbix, Grafana, Zammad) en un servidor Linux cada una de ellas en contenedores diferentes.

- Se mostrará un ejemplo real de la solución en donde se configurará elementos de redes y seguridad que se conecten al sistema de monitoreo y muestren de manera gráfica el estado de los dispositivos o aplicativos a monitorear.
- Se simulará como un cambio de estado de un sistema o dispositivo genera una alerta al administrador y la asignación de un ticket de atención en Zammad.

CAPITULO 2

2. DISEÑO DE IMPLEMENTACION DE LA SOLUCION

2.1. Descripción de arquitectura de las herramientas o plataformas utilizadas en la solución

El presente proyecto tiene por la finalidad de integrar Zabbix contra las herramientas Grafana y Zammad de acuerdo con la arquitectura mostrada en la Figura 2.1.

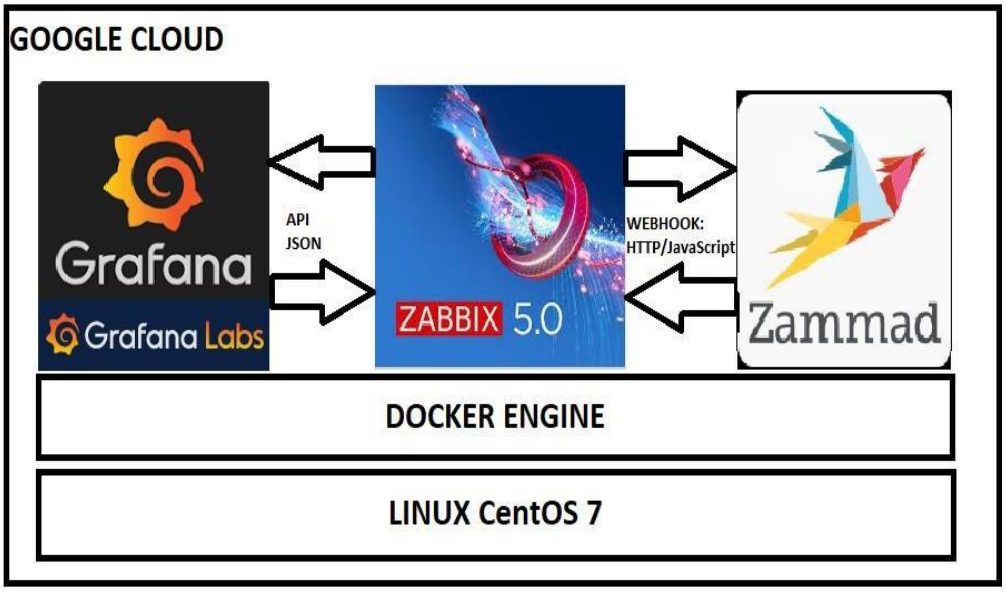


Figura 2.1. Arquitectura Comunicación Zabbix - Grafana y Zabbix-Zammad en Linux usando Docker

Para el presente proyecto usaremos las siguientes versiones de cada herramienta:

- Zabbix: Zabbix 5.0.2
- Grafana: Grafana v7.1.1
- Zammad: Zammad versión 3.4

Las plataformas serán instaladas en contenedores individuales dentro de una maquina Docker en un sistema Linux de tipo CentOS Versión 7.

El server usado se encontrará alojado en una máquina virtual en Google Cloud.

2.2. Instalación de Linux con Docker en la nube de Google

Para iniciar la implementación del proyecto se procede primeramente a configurar en la nube de Google una máquina virtual con el sistema operativo CentOS 7[4], como se muestra en la Figura 2.2.

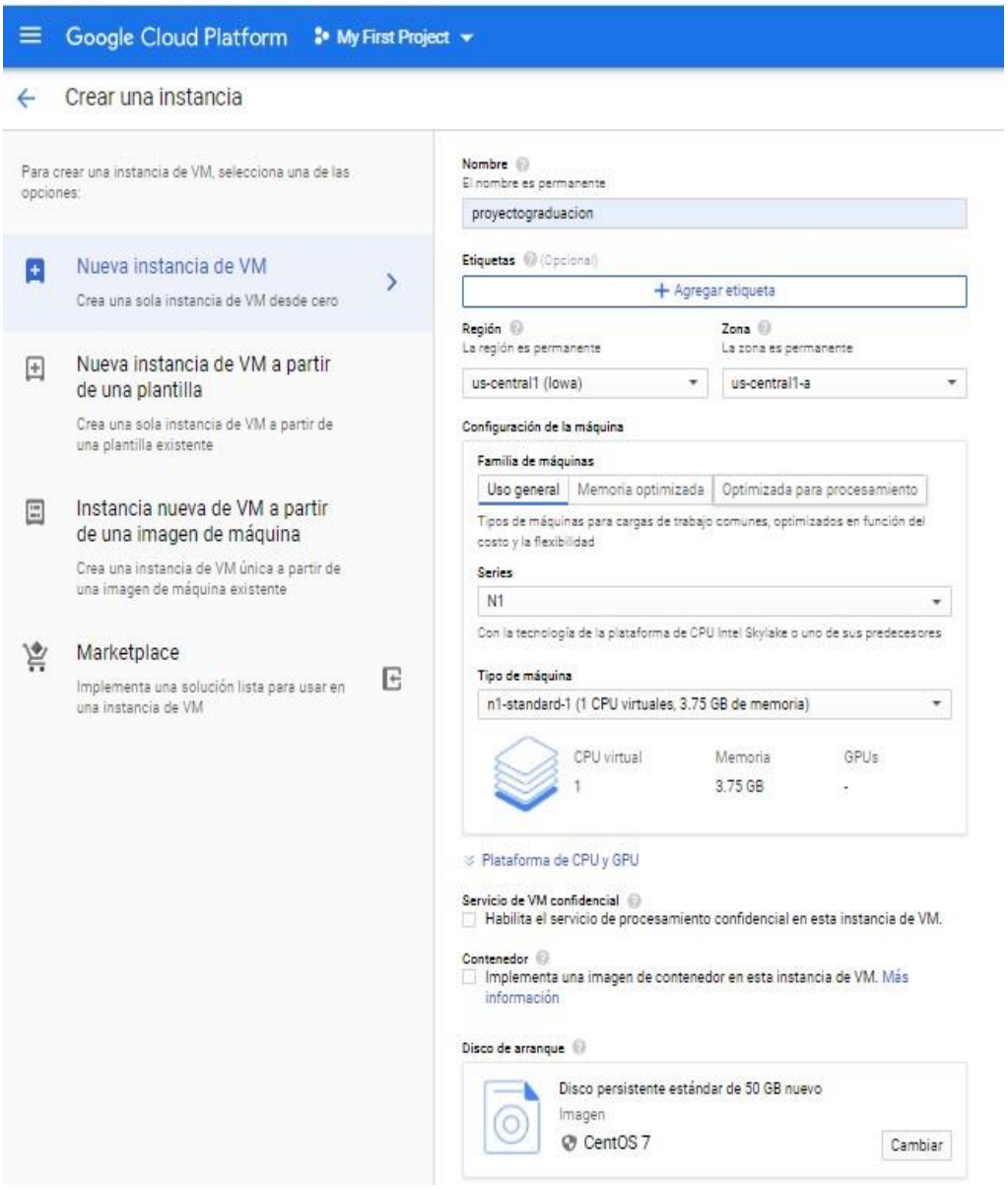


Figura 2.2. Creación de Server Linux con CentOS 7 en la nube de Google

Como se observa en la gráfica, la máquina virtual es creada con las características básicas con un solo CPU, con 3.75 GB de memoria, por la cantidad de aplicaciones a instalarse se crea imagen con 50GB de capacidad. El principal motivo de usar las características básicas es que Google asigna a los usuarios un demo gratuito de un año o $300, lo primero que se cumpla, este abono de $300 se consume de acuerdo con las características de configuración de la máquina virtual, mientras mayores características se utiliza el valor por hora será mayor lo que ocasionaría que el abono de gracia proporcionado se consuma con más rapidez.

Google asigna una dirección IP interna (10.128.0.9) dentro de su rango privado y adicional a esta característica se le configura una dirección IP v4 de tipo pública para poder ser accedida desde cualquier parte del mundo, a ella se le configura un nombre de host en un DNS para un más fácil acceso con el nombre zabbix.segurinfor.net con IP 35.208.18.97. En la Figura 2.3 se puede mostrar lo indicado anteriormente.

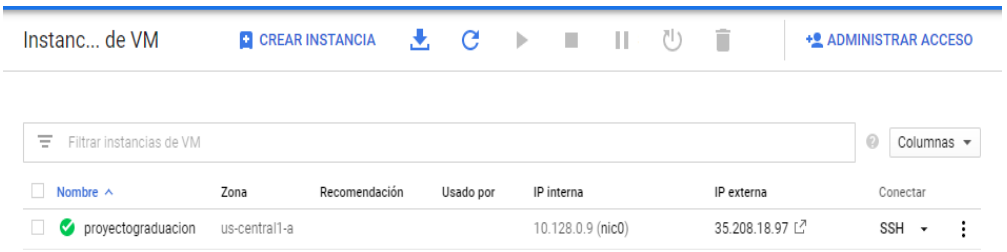


Figura 2.3. Presentación de máquina virtual en la nube de Google

Una vez creada la máquina virtual se procede a la instalación del sistema Docker para poder crear los contenedores individuales de cada una de las herramientas a integrar en la presente solución.[5]

En el proyecto se utilizó la máquina Docker y una herramienta de esta denominada Docker-Compose, que permite mediante archivos YAML poder instruir al Docker Engine a realizar tareas de manera programada, obteniendo una serie de instrucciones, que luego fácilmente pueden ser realizadas en otros ambientes de Sistemas Operativos.

Se usaron las 2 diferentes formas de instalación por la facilidad en la implementación: Zabbix y Zammad existían ya versiones propiamente con Docker Compose en donde de manera separada instalaba lo que requería para su funcionamiento como servidor web, base de datos y otros requerimientos adicionales, independiente del sistema operativo, en el caso de Grafana se decidió utilizar ya una imagen diseñada propiamente para CentOS 7.

Los pasos para instalar Docker [6] son los siguientes:

i. Actualizar el repositorio de donde se descargará Docker en su última versión estable de acuerdo con los comandos de la Figura 2.4.

```
$ sudo yum install -y yum-utils

$ sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
```

Figura 2.4. Comandos para actualizar repositorio Docker

ii. Instalar la última versión estable de Docker, como se muestra en la Figura 2.5

```
$ sudo yum install docker-ce docker-ce-cli containerd.io
```

Figura 2.5. Comando para Instalar Docker

iii. Revisión de versión de Docker. Se observa en la Figura 2.6 que la última versión estable instalada tanto del servidor y el cliente Docker es la 19.03.12.

```
[josec24091971@proyectograduacion ~]$ docker version
Client: Docker Engine - Community
 Version:           19.03.12
 API version:       1.40
 Go version:        go1.13.10
 Git commit:        48a66213fe
 Built:             Mon Jun 22 15:46:54 2020
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.12
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.10
  Git commit:       48a66213fe
  Built:            Mon Jun 22 15:45:28 2020
  OS/Arch:          linux/amd64
  Experimental:     false
```

Figura 2.6. Revisión de versión de Docker instalada

iv. Para habilitar e iniciar Docker, se ejecuta los comandos de acuerdo con lo mostrado en la Figura 2.7.

```
systemctl enable docker
```

```
systemctl start docker
```

Figura 2.7. Habilitar e Iniciar Docker

Los pasos para instalar Docker-Compose [7] son los siguientes:

i. Descargamos con Curl la última versión de Docker-Compose estable 1.26.2, ejecutando el comando como se muestra en la Figura 2.8.

```
curl -L "https://github.com/docker/compose/releases/download/ \
1.26.2/docker-compose-$(uname -s)-$(uname -m)" -o \
/usr/local/bin/docker-compose
```

Figura 2.8. Descargar Docker-Compose

ii. De acuerdo con los comandos indicados en la Figura 2.9 se asignan los permisos de ejecución al Docker Compose binario.

```
chmod +x /usr/local/bin/docker-compose
```

Figura 2.9. Asignar permisos a Docker-Compose

iii. Verificamos la versión de Docker-Compose, en donde como se muestra en la Figura 2.10 es la 1.26.2.

```
[josec24091971@proyectograduacion ~]$ docker-compose -v
docker-compose version 1.26.2, build eefe0d31
```

Figura 2.10. Verificar versión de Docker-Compose instalada

2.3. Instalación de Herramientas Zabbix, Grafana y Zammad en Linux

2.3.1. Instalación de Zabbix en Linux

Para la instalación de Zabbix en Linux CentOS utilizamos la imagen disponible de Docker-Compose.[8][9]

Zabbix proporciona archivos de composición para definir y ejecutar componentes de varios contenedores en Docker. Estos archivos de redacción están disponibles en el repositorio oficial de Docker de Zabbix en github.com: https://github.com/zabbix/zabbix-docker .

El archivo de composición utilizado en el presente proyecto de acuerdo como se muestra en la Figura 2.11 fue el docker-compose_v3_alpine_mysql_latest.yaml.

Nombre del archivo	Descripción
docker-compose_v3_alpine_mysql_latest.yaml	El archivo de redacción ejecuta la última versión de los componentes Zabbix 5.0 en Alpine Linux con soporte de base de datos MySQL.

Figura 2.11. Nombre de archivo de composición utilizando Alpine y MySQL

Se implemento la imagen utilizando como sistema operativo base el Linux Alpine por ser un sistema que no consume muchos recursos, se eligió MySQL como base de datos. Así mismo los componentes de servidor web se utilizó Nginx y se instalaron los componentes de Java

Gateways para monitoreo de aplicaciones que utilicen JMX y SNMPTraps para dispositivos que envían información cuando se presenta un evento como por ejemplo un sensor.

A continuación, en la Figura 2.12 se muestra una pantalla de los componentes instalados para la ejecución de Zabbix:

```
[josec24091971@proyectograduacion ~]$ docker ps |grep zabbix
d76578b7b8dc        zabbix/zabbix-web-nginx-mysql:alpine-5.0-latest             "docker-entrypoint.sh"
 9 days (healthy)   0.0.0.0:80->8080/tcp, 0.0.0.0:443->8443/tcp   zabbix-docker_zabbix-web-nginx-mysql_1
215dc9c6ceaa        zabbix/zabbix-server-mysql:alpine-5.0-latest                "/sbin/tini -- /usr/…"
 9 days             0.0.0.0:10051->10051/tcp                    zabbix-docker_zabbix-server_1
755c8d83e0b4        zabbix/zabbix-snmptraps:alpine-5.0-latest                   "/usr/bin/supervisor…"
 9 days             0.0.0.0:162->1162/udp                       zabbix-docker_zabbix-snmptraps_1
1908c81c3c77        mysql:8.0                                                   "docker-entrypoint.s…"
 9 days                                                         zabbix-docker_mysql-server_1
d5aff88e9282        zabbix/zabbix-java-gateway:alpine-5.0-latest                "docker-entrypoint.s…"
 9 days                                                         zabbix-docker_zabbix-java-gateway_1
```

Figura 2.12. Componentes Zabbix instalados usando Docker-Compose

El procedimiento para su instalación fue el siguiente:

i. Se descarga los archivos de composición de la última versión de Zabbix haciendo una clonación del repositorio desde GitHub a nivel local con los siguientes comandos indicados en la Figura 2.13

```
# get zabbix-docker repo
git clone https://github.com/zabbix/zabbix-docker.git
cd zabbix-docker
```

Figura 2.13. Comandos para descargar archivos de composición para Zabbix desde GitHub

ii. Se ingresa al directorio donde se encuentran los archivos descargados y se edita el archivo de configuración “docker-compose_v3_alpine_mysql_latest.yaml” en donde se realiza la customización de lo requerido a ejecutar para la funcionalidad de este proyecto como es elegir Nginx como servidor Web, el direccionamiento de los puertos a utilizar como el 80 para conexiones web, el 10051 para conexiones propias de Zabbix, y

demás configuraciones. En la Figura 2.14 se observa brevemente el aspecto del archivo YAML utilizado.

```
[josec24091971@proyectograduacion zabbix-docker]$ more docker-compose_v3_alpine_mysql_latest.yaml
version: '3.5'
services:
 zabbix-server:
  image: zabbix/zabbix-server-mysql:alpine-5.0-latest
  ports:
   - "10051:10051"
  volumes:
   - /etc/localtime:/etc/localtime:ro
   - /etc/timezone:/etc/timezone:ro
   - ./zbx_env/usr/lib/zabbix/alertscripts:/usr/lib/zabbix/alertscripts:ro
   - ./zbx_env/usr/lib/zabbix/externalscripts:/usr/lib/zabbix/externalscripts:ro
   - ./zbx_env/var/lib/zabbix/export:/var/lib/zabbix/export:rw
   - ./zbx_env/var/lib/zabbix/modules:/var/lib/zabbix/modules:ro
   - ./zbx_env/var/lib/zabbix/enc:/var/lib/zabbix/enc:ro
   - ./zbx_env/var/lib/zabbix/ssh_keys:/var/lib/zabbix/ssh_keys:ro
   - ./zbx_env/var/lib/zabbix/mibs:/var/lib/zabbix/mibs:ro
   - snmptraps:/var/lib/zabbix/snmptraps:rw
  links:
   - mysql-server:mysql-server
   - zabbix-java-gateway:zabbix-java-gateway
  ulimits:
   nproc: 65535
   nofile:
    soft: 20000
    hard: 40000
  deploy:
   resources:
    limits:
      cpus: '0.70'
      memory: 1G
    reservations:
      cpus: '0.5'
      memory: 512M
  env_file:
   - .env_db_mysql
   - .env_srv
  secrets:
   - MYSQL_USER
   - MYSQL_PASSWORD
   - MYSQL_ROOT_PASSWORD
```

Figura 2.14. Edición de archivo YAML para configurar servicios de Zabbix.

Se procede a descargar las imágenes requeridas lo cual es realizado en la primera ejecución y se levanta cada uno de los servicios necesarios, con el comando mostrado en la Figura 2.15

```
[josec24091971@proyectograduacion zabbix-docker]$ docker-compose -f docker-compose_v3_alpine_mysql_latest.yaml up -d
```

Figura 2.15. Iniciar servicios de Zabbix usando Docker-Compose

iii. Luego de esto quedan levantado los servicios, pudiendo ingresar al sistema de Zabbix accediendo al sitio

http://zabbix.segurinfor.net/index.php, con lo que ahora se inicia la integración y configuración de acuerdo con los objetivos planteados. En la Figura 2.16 se detalla la página de inicio de Zabbix.

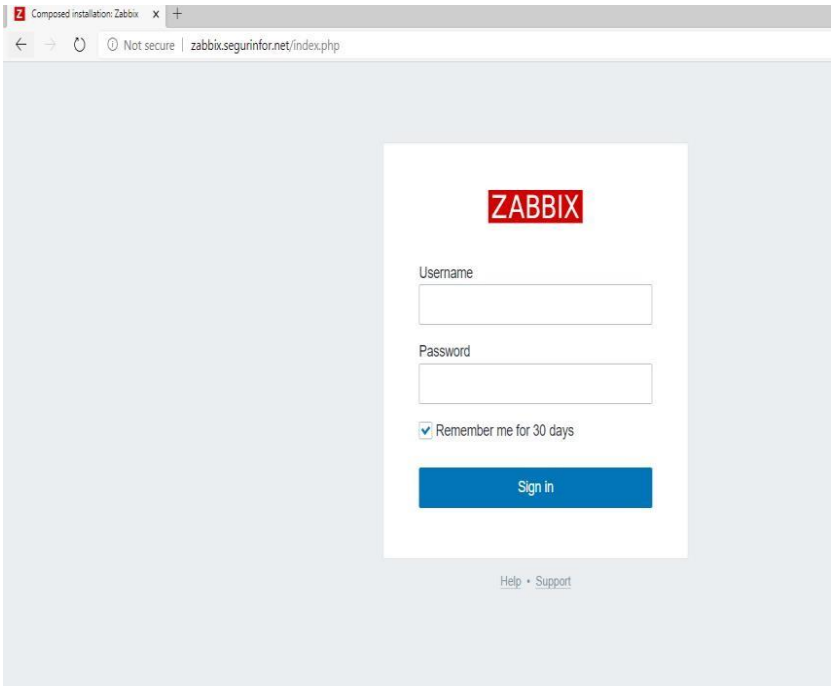


Figura 2.16. Página de Inicio de Zabbix

2.3.2. Instalación de Grafana en Linux

Para la instalación de Grafana es necesario seguir los siguientes pasos[10]:

i. Ejecutar el comando de acuerdo con la Figura 2.17 que descargara la última versión de Grafana si es que no ha sido aun descargado, y lo ejecutará en el puerto 3000.

```
docker run -d --name=grafana -p 3000:3000 grafana/grafana
```

Figura 2.17. Instalación y ejecución de Grafana en Linux

ii. Para que se ejecute de manera automática cuando el sistema se inicializa se debe ejecutar el comando de acuerdo con la Figura 2.18.

```
docker update --restart always grafana
```

Figura 2.18. Comando para ejecutar manera automática Grafana al iniciar el sistema

iii. Ingresamos a la dirección: http://zabbix.segurinfor.net:3000/ que nos mostrara la página de Inicio de Grafana, tal como se muestra en la Figura 2.19.

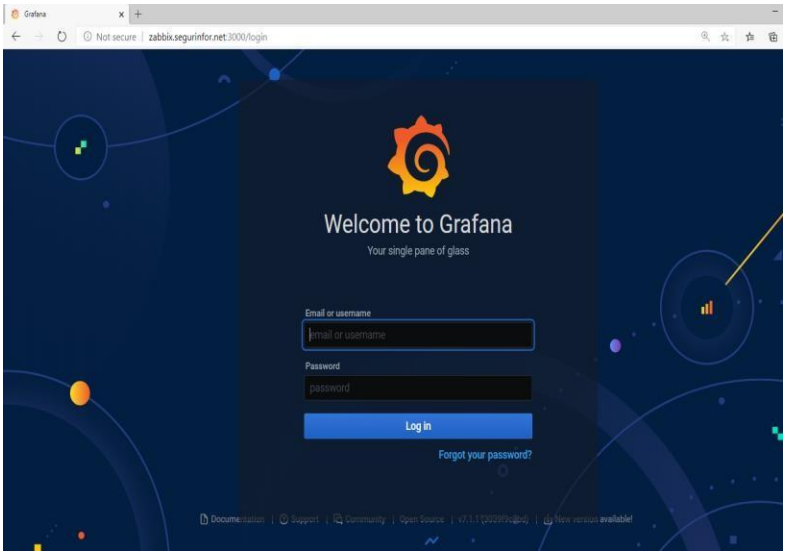


Figura 2.19. Página de Inicio de Grafana

2.3.3. Instalación de Zammad en Linux

La instalación de Zammad fue similar a la de Zabbix, se usó la técnica de Docker-Compose por la necesidad de levantar algunos servicios adicionales que son necesarios para el funcionamiento de Zammad como son el servidor web que también estará basado en Nginx, Elasticsearch, la base de datos Postgresql, y otros requerimientos adicionales.[11]

En la Figura 2.20, se muestran los componentes instalados para el funcionamiento de Zammad:

```
[josec24091971@proyectograduacion zammad-docker-compose]$ docker ps | grep zammad
65e5090d08a3        zammad/zammad-docker-compose:zammad-3.4.0-16                "/docker-entrypoint.…"
:8070->80/tcp                           zammad-docker-compose_zammad-nginx_1
64133940ec5e        zammad/zammad-docker-compose:zammad-3.4.0-16                "/docker-entrypoint.…"
                                        zammad-docker-compose_zammad-scheduler_1
37f33d899c25        zammad/zammad-docker-compose:zammad-3.4.0-16                "/docker-entrypoint.…"
                                        zammad-docker-compose_zammad-websocket_1
85d61b49b07b        zammad/zammad-docker-compose:zammad-postgresql-3.4.0-16     "/usr/local/bin/back…"
p                                       zammad-docker-compose_zammad-backup_1
eae28f9ca659        zammad/zammad-docker-compose:zammad-3.4.0-16                "/docker-entrypoint.…"
                                        zammad-docker-compose_zammad-railsserver_1
1ea2271c9935        memcached:1.5.22-alpine                                     "docker-entrypoint.s…"
cp                                      zammad-docker-compose_zammad-memcached_1
fb6eb3d68666        zammad/zammad-docker-compose:zammad-elasticsearch-3.4.0-16  "/usr/local/bin/dock…"
p, 9300/tcp                             zammad-docker-compose_zammad-elasticsearch_1
146142299c91        zammad/zammad-docker-compose:zammad-postgresql-3.4.0-16     "docker-entrypoint.s…"
p                                       zammad-docker-compose_zammad-postgresql_1
```

Figura 2.20. Componentes de Zammad instalados a través de Docker-Compose

El procedimiento para su instalación fue el siguiente:

i. Se descarga los archivos de composición de la última versión de Zammad haciendo una clonación del repositorio desde GitHub a nivel local con los comandos mostrados en la Figura 2.21.

```
git clone https://github.com/zammad/zammad-docker-compose.git

cd zammad-docker-compose/
```

Figura 2.21. Comandos para descargar archivos de composición para Zammad desde GitHub

ii. Obtenemos el numero de la última versión de Zammad desde https://hub.docker.com/r/zammad/zammad-docker-compose/tags. En donde se obtuvo la versión: 3.4.0-16

iii. Se ingresa al directorio donde se encuentran los archivos descargados y se edita el archivo de configuración “.env” que es donde se definen las variables de ambiente de Zammad y se establecen el user y password que Zammad utilizara para conectarse a la base de datos PostgreSQL, que levante cada vez que el sistema inicie y la versión que se utilizara, en donde se procede a actualizar con la última versión estable. Esto se refleja en la Figura 2.22:

```
[josec24091971@proyectograduacion zammad-docker-compose]$ more .env
IMAGE_REPO=zammad/zammad-docker-compose
POSTGRES_PASS=zammad
POSTGRES_USER=zammad
RESTART=always
# don't forget to add the minus before the version
VERSION=-3.4.0-16
```

Figura 2.22. Configuración de variables de inicio para Zammad

iv. Luego de esto editamos el archivo de composición “docker-compose.override.yml” para cambiar el direccionamiento de puerto de 80:80 a 8070:80, esto debido al que el puerto 80 ya está siendo utilizado por Zabbix, como se muestra en la Figura 2.23.

```
[josec24091971@proyectograduacion zammad-docker-compose]$ more docker-compose.override.yml
version: '2'
services:

  zammad-nginx:
    ports:
      - "8070:80"
```

Figura 2.23. Edición de archivo configuración para definir puerto comunicación de Zammad

v. Por último, procedemos a iniciar el servicio con el siguiente comando ejecutado desde dentro del directorio “zammad-docker-compose” como se muestra en la Figura 2.24.

```
docker-compose up -d
```

Figura 2.24. Inicialización del servicio Zammad

vi. Se procede a revisar el correcto funcionamiento ingresando al sitio **http://zabbix.segurinfor.net:8070** donde se visualizará la pantalla inicial de Zammad como se muestra en la Figura 2.25.

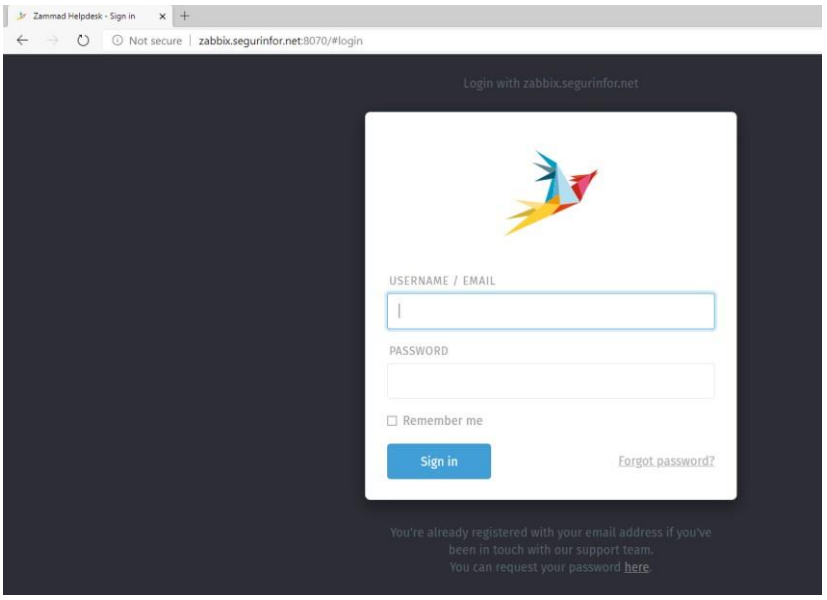


Figura 2.25. Página de Inicio de Zammad

2.4. Integración de las Herramientas: Zabbix-Grafana y Zabbix-Zammad

2.4.1. Integración Zabbix-Grafana

La integración de Zabbix con Grafana se la realiza a través de la instalación de un plugin que debe instalarse en la máquina de Grafana de acuerdo con el procedimiento que se explica a continuación[12][13]:

i. Primeramente, debemos identificar el nombre del contenedor que contiene la imagen de Grafana de acuerdo con el comando “Docker ps” como se muestra en la Figura 2.26, en donde se determina que la imagen utilizada es “e136901337fb”

Figura 2.26. Identificar nombre de Imagen del Contenedor de Grafana

ii. Nos conectamos a la interfaz de esta imagen en modo root con el comando mostrado en la Figura 2.27

```
[josec24091971@proyectograduacion zammad-docker-compose]$ docker exec -u 0 -ti e136901337fb /bin/bash
bash-5.0#
```

Figura 2.27. Ingresando a la Interfaz de comando de Grafana

iii. Se instala el plugin de integración entre Grafana y Zabbix a través del comando mostrado en la Figura 2.28.

```
bash-5.0# grafana-cli plugins install alexanderzobnin-zabbix-app
```

Figura 2.28. Instalación de plugin para integración de Zabbix-Grafana

iv. Validamos en la sección de plugin en Grafana si el procedimiento anterior instalo el plugin Zabbix. Y de acuerdo con lo observamos en la Figura 2.29, este se muestra la instalación de plugin que aún no se encuentra verificado en Grafana por lo que aparece como no firmado (unsigned).

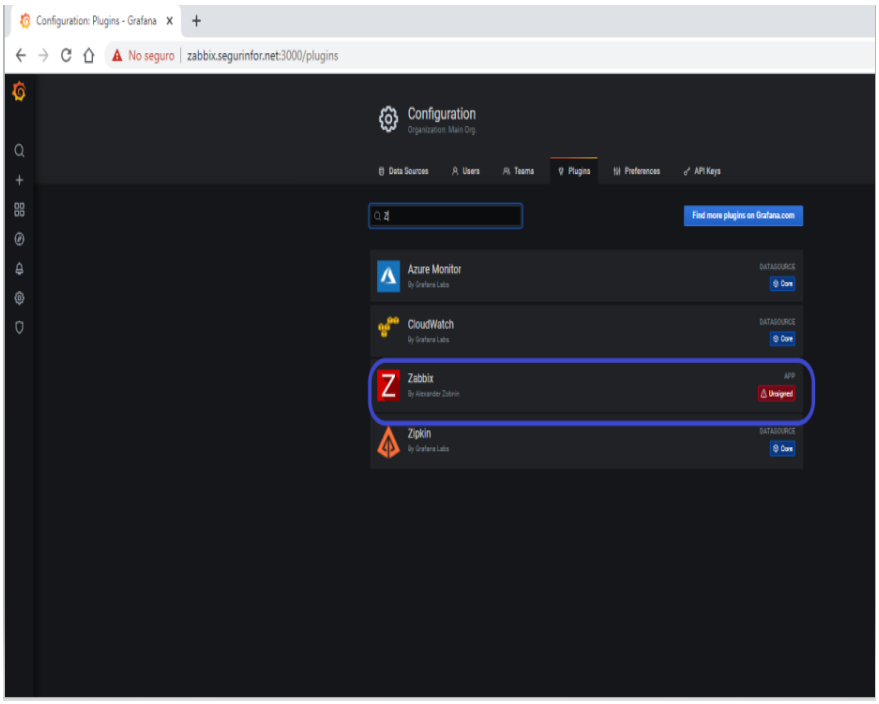


Figura 2.29. Validación de instalación de plugin de Grafana con Zabbix

v. Ingresamos al Plugin para habilitarlo de acuerdo como se muestra en la Figura 2.30.

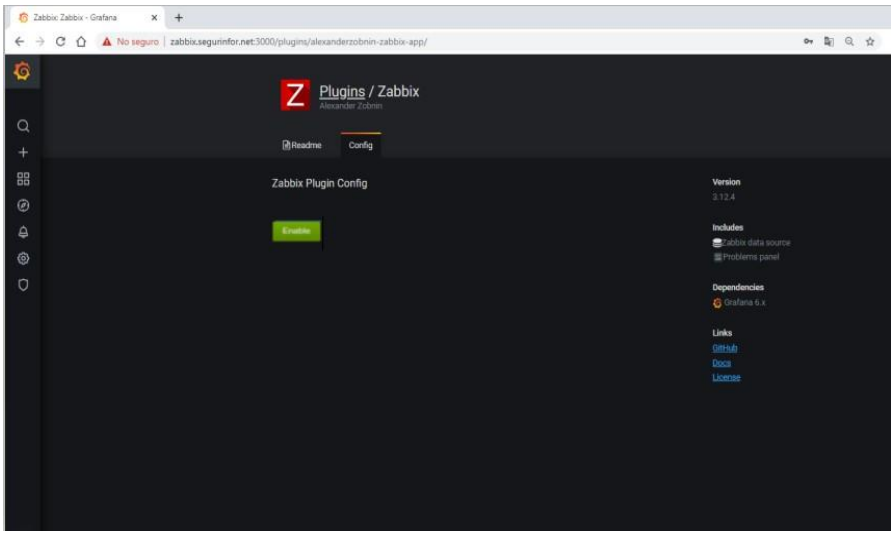


Figura 2.30. Habilitar Plugin de Zabbix en Grafana

vi. Después de habilitarlo se puede añadir la fuente de datos de Zabbix. Y se debe conectar con un username y password creado en la base de datos para Zabbix, en este caso para el proyecto se utilizó el usuario de administración “Admin”, y se lo apunto al servidor de Zabbix instalado en el dominio http://zabbix.segurinfor.net conectándolo al API de Zabbix a través del conector “api_jsonrpc.php”. En la Figura 2.31 se específica las configuraciones a realizar para configurar la fuente de datos.

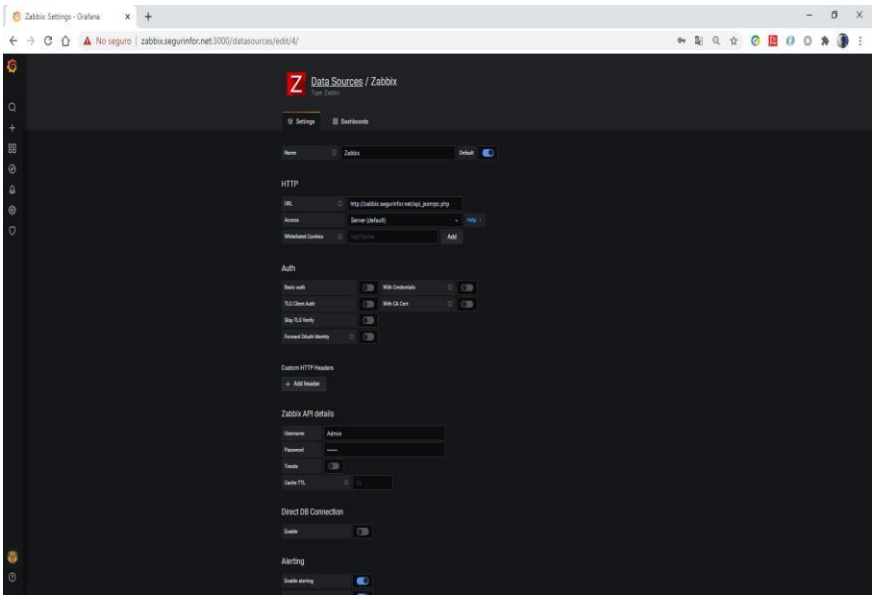


Figura 2.31. Configurar fuente de datos del Server Zabbix

vii. Por último, se valida la correcta conexión de Grafana con la base de datos de Zabbix, lo cual puede ser observado en la Figura 2.32.

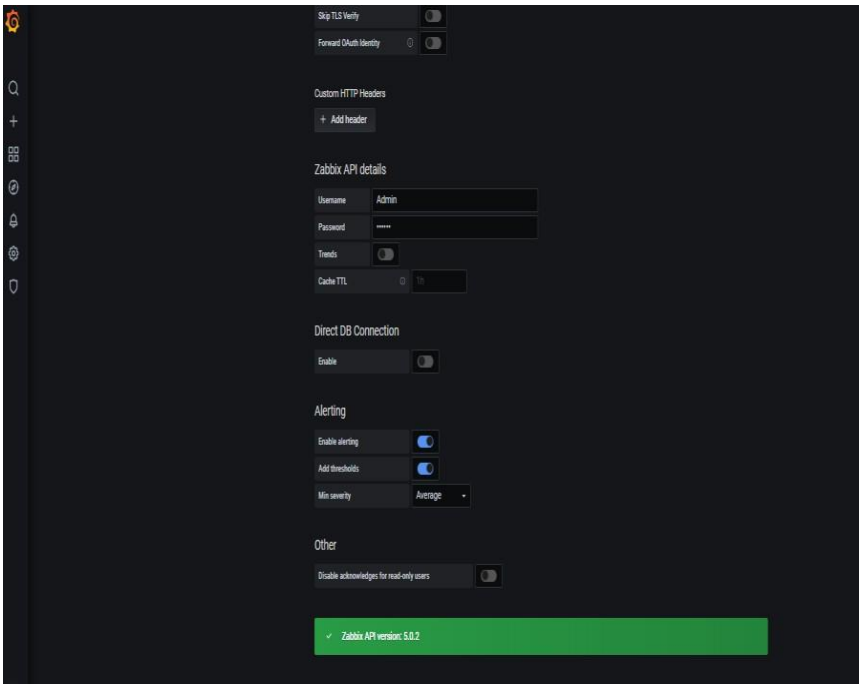


Figura 2.32. Validar conexión de base de datos de Zabbix con Grafana

De esta manera se ha logrado la integración de Zabbix con Grafana en la próxima sección de Presentación de Resultados se mostrará la elaboración de dashboard con servidores, aplicaciones, routers y switches en como presentar las métricas de cada servicio.

2.4.2. Integración Zabbix-Zammad

A diferencia de la integración de Zabbix con Grafana en donde se la realizaba a través de un API que se conectaba a la base de datos de Zabbix en donde se debía autenticar con un usuario para poderacceder a los datos, en Zammad la autenticación se la realiza a travésde un token, adicional que se debe hacer configuraciones en ambos sistemas para lograr la integración.[14]

Zabbix utiliza Webhook para esta integración con Zammad. Webhook es un sistema de comunicación automático entre aplicaciones, a pesar de que son similares en funcionalidad a las API, se diferencian en la forma que se reciben los datos en donde los API lo hacen a través de un proceso denominado “sondeo”, los Webhook permite enviar datos a la aplicación en tiempo real. Los Webhook son herramientas ideales para conocer que transacciones ocurren dentro de una aplicación, en nuestro caso conocer los eventos que ocurren en Zabbix y así poder generar un ticket.[15][16]

Los pasos por seguir para configurar la integración son los siguientes:

ZAMMAD:

i. Se debe acceder a la configuración global en la sección de API para habilitar el acceso al token, esto es mostrado en la Figura 2.33.

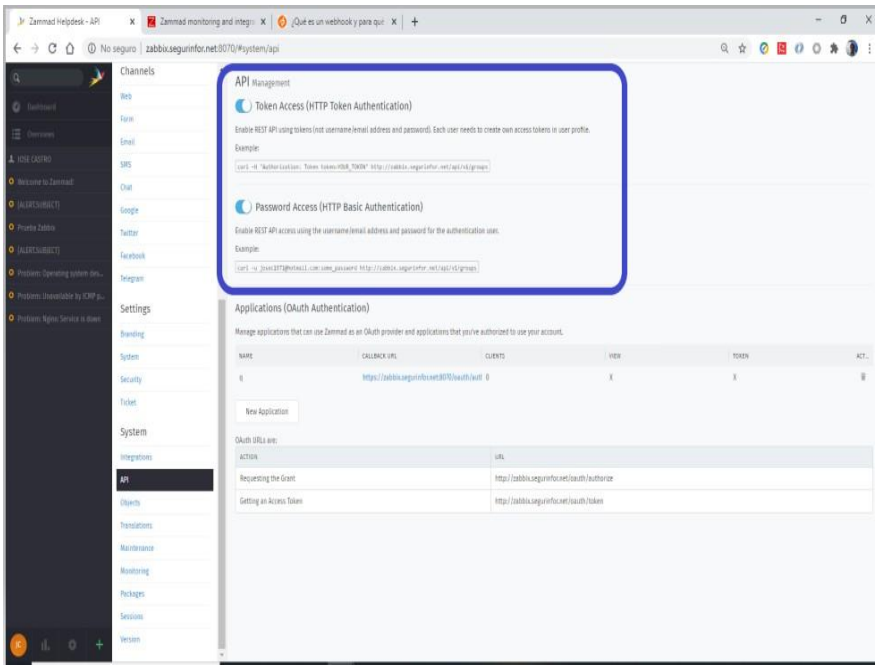


Figura 2.33. Habilitar el acceso al Token del API en Zammad

ii. Crear un nuevo usuario para una alerta de Zabbix con una dirección de correo electrónico y cree un token de usuario personal con permisos de ticket.agent, esto se detalla en la Figura2.34.

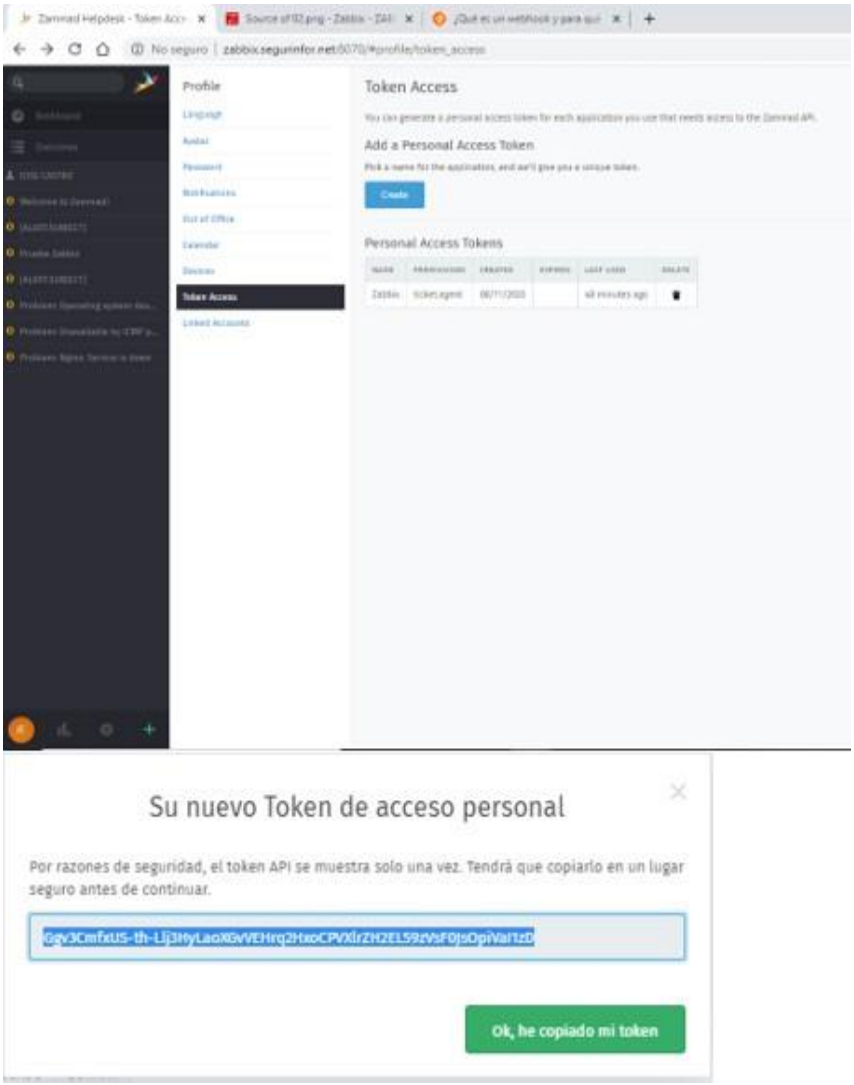


Figura 2.34. Generación de Token para integración con Zabbix

ZABBIX:

i. Configurar la macro global {$ ZABBIX.URL}, que debe contener la URL de la interfaz de Zabbix que es definida como http://zabbix.segurinfor.net como se observa en la Figura 2.35

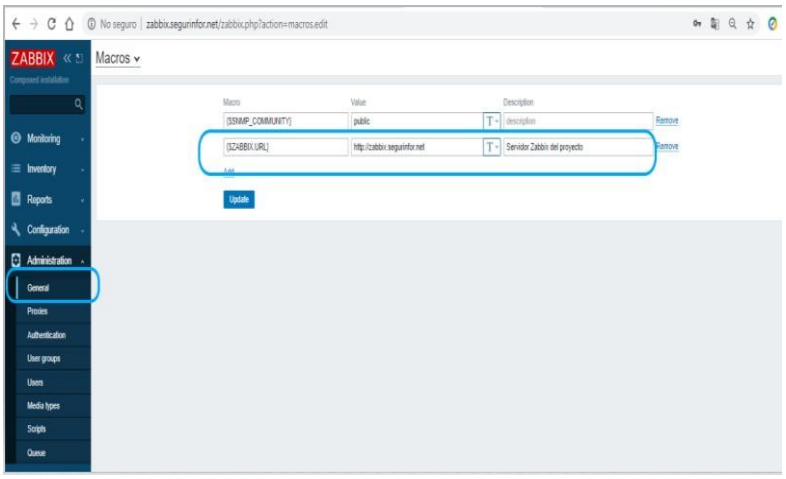


Figura 2.35. Configurar macro global de URL de Zabbix

ii. Importar el archivo “media_zammad.xml” en la sección de configuración de tipos de medios: Media types. Esto crea el medio Zammad de tipo Webhook como se observa en la Figura 2.36 a continuación.

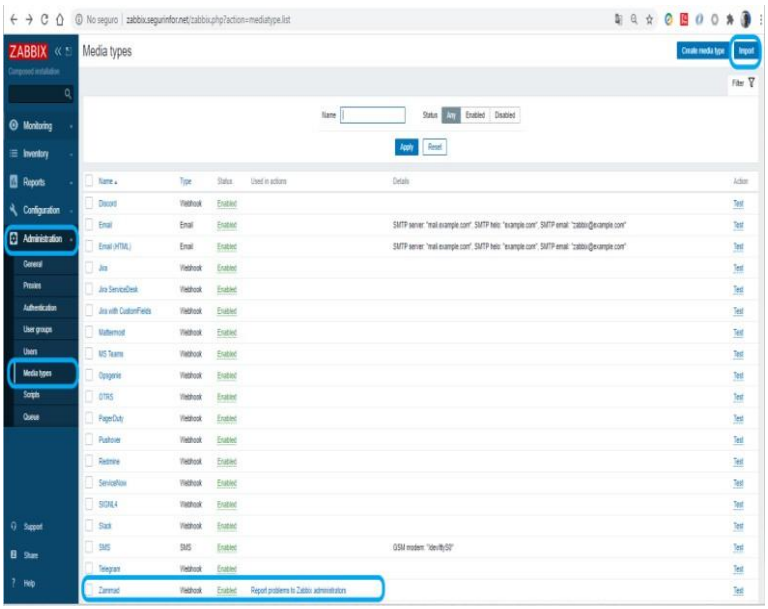


Figura 2.36. Importar archivo media_zammad.xml desde GitHub

iii. Abrir el tipo de medio Zammad creado en el paso anterior y proceder a configurar los siguientes campos, que observamos también en la Figura 2.37

- zammad_access_token con el token proporcionado anteriormente en la parte de configuración de Zammad.
- zammad_url con la dirección o dominio del Zammad instalado, en nuestro caso http://zabbix.segurinfor.net:8070
- zammad_customer con la dirección email configurada en Zammad en nuestro caso **josec1971@hotmail.com**
- zammad_enable_tags se elige como verdadero o falso dependiendo si deseamos o no habilitar las etiquetas de activación.

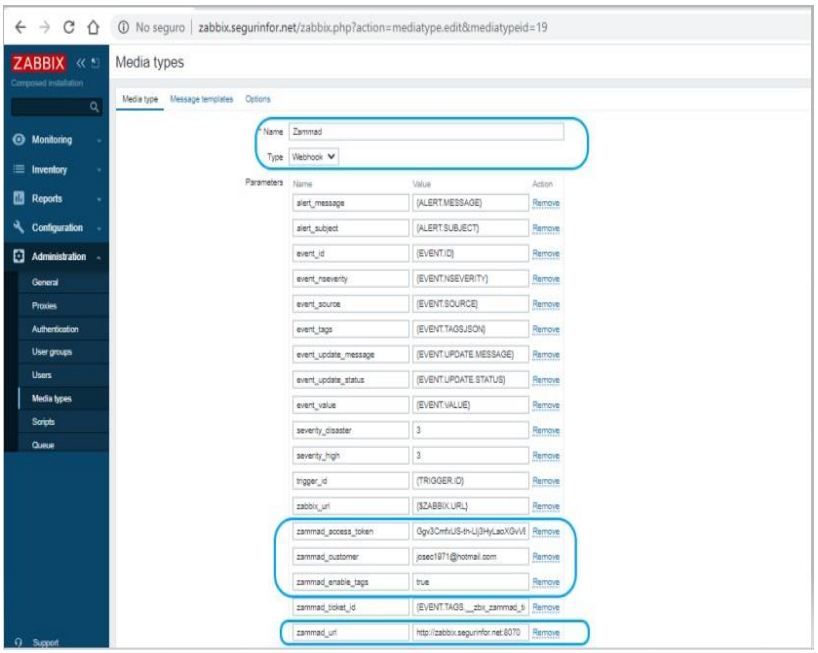


Figura 2.37. Configurar parámetros necesarios para el Webhook de Zammad dentro de Zabbix

iv. Si desea priorizar los problemas de acuerdo con los valoresde gravedad en Zabbix, puede definir parámetro de mapeo:

severity_ \ <name> : ID de prioridad de Zammad, como se observa en la Figura 2.38.

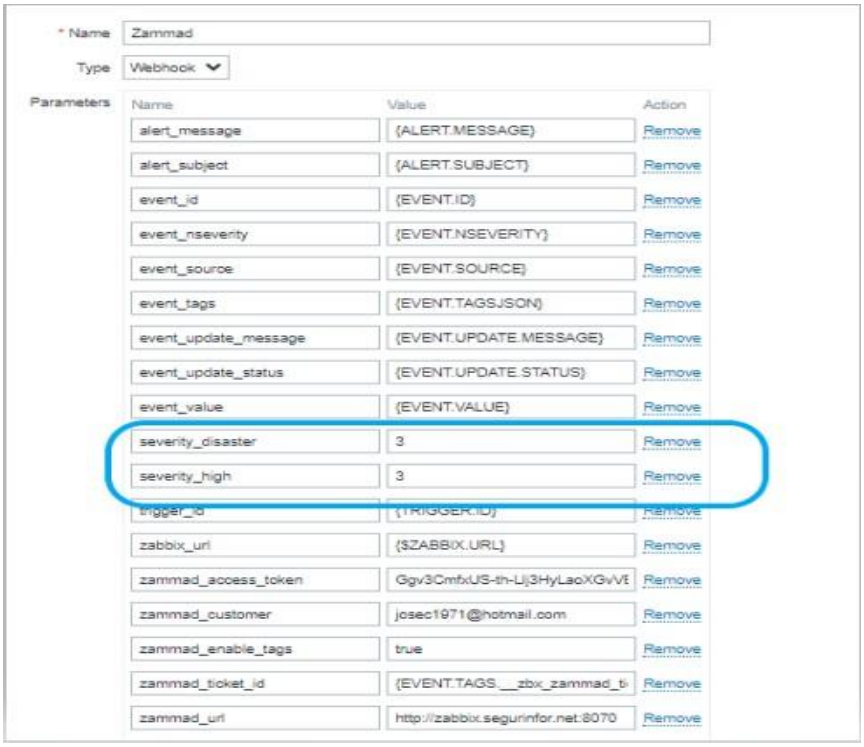


Figura 2.38. Configurar prioridad de casos de acuerdo con severidad definida en Zabbix

v. Guardamos las configuraciones realizadas presionando la acción Update.

vi. Para recibir las notificaciones en Zammad es requisito crear un usuario Zabbix y añadir el medio de tipo Zammad. De acuerdo con los pasos mostrados en la Figura 2.39 y Figura 2.40.

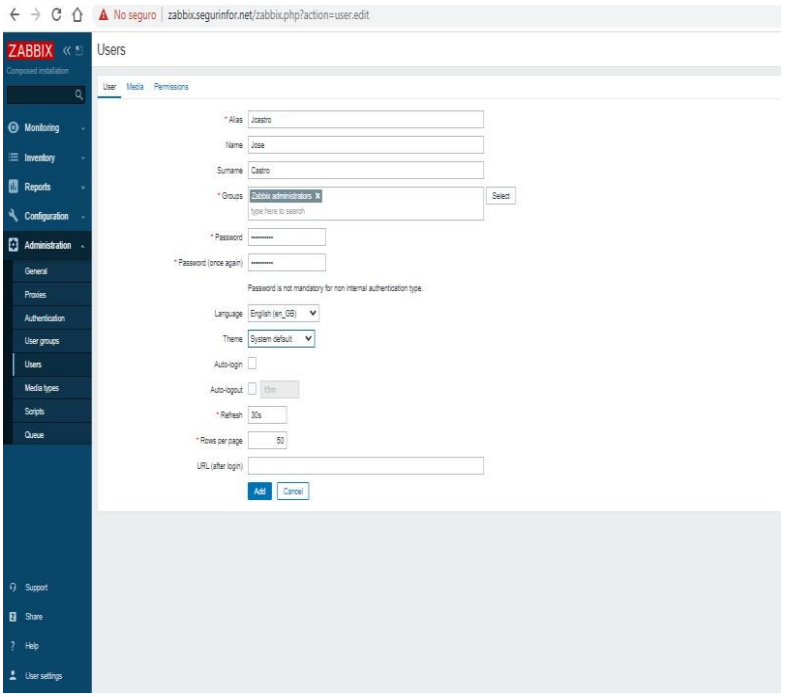


Figura 2.39. Creación de Usuario en Zabbix

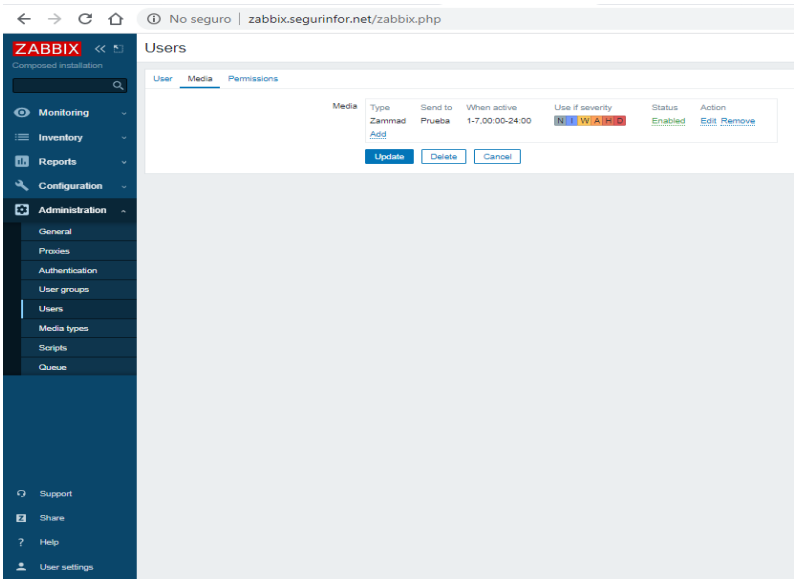


Figura 2.40. Definiendo tipo de medio Zammad al usuario Zabbix

vii. Por último, se debe definir la acción para que se genere el ticket al usuario o grupo de usuarios que deben recibir las notificaciones de

alertas en Zammad. En la Figura 2.41 se observa como configurar el tipo de acción.

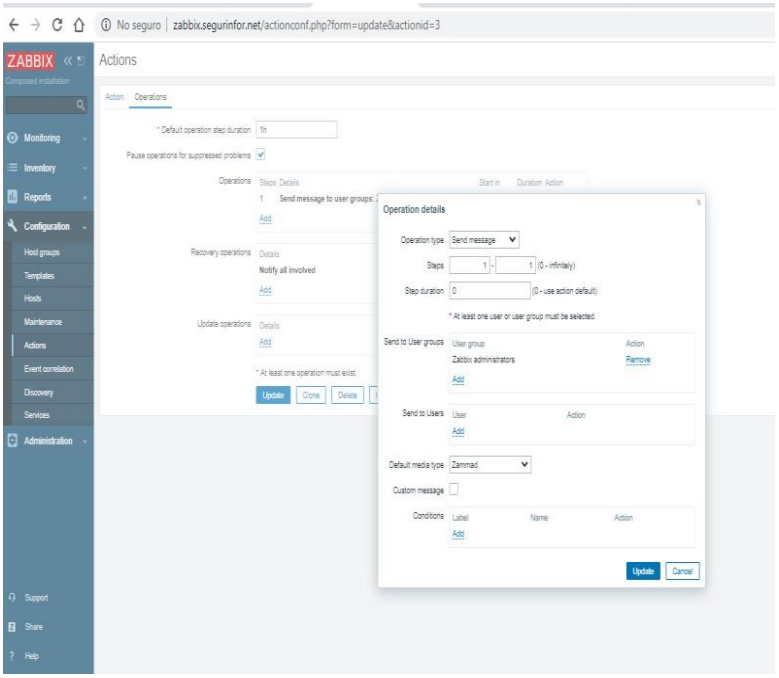


Figura 2.41. Definir tipo de acción para envío de notificaciones a Zammad

CAPITULO 3

3. Presentación de Resultados

3.1. Presentación Visual de Herramientas Integradas

3.1.1. Configuración de Hosts en Zabbix

Para la presente simulación se consideraron varios elementos a monitorear entre los que se incluyen: Monitoreo de 2 Servidores Linux con agente Zabbix, Firewall Sophos usando SNMP, Servidor Web en Nginx, Servidor Linux con SNMP. De acuerdo con la Figura 3.1 podemos observar los hosts configurados en Zabbix para objeto de la presente demostración.

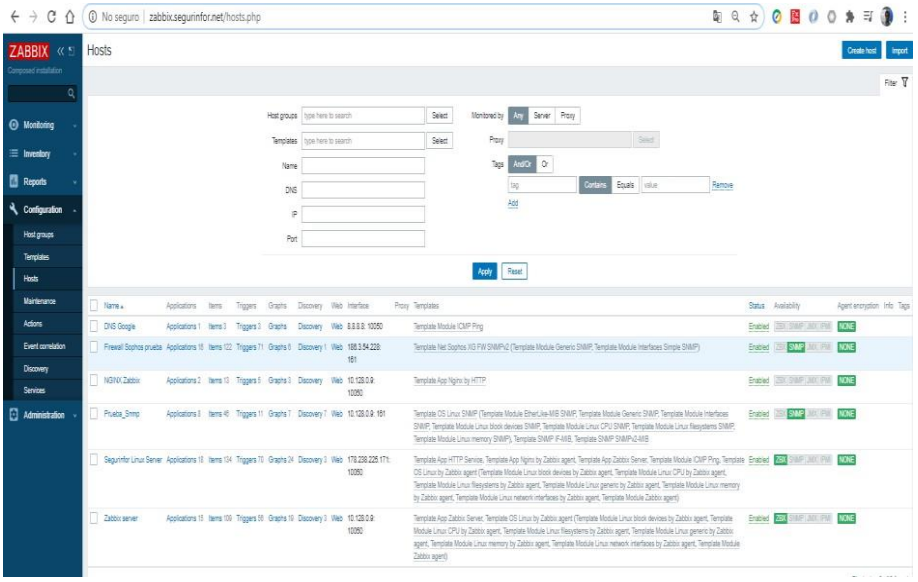


Figura 3.1. Hosts configurados en Zabbix

Al no ser este proyecto un manual de uso de Zabbix se ha omitido lo que corresponde el procedimiento para configurar un host[17], para ello pueden revisar el manual online en donde se explica de manera detallada los pasos a realizar y se encuentra ubicado en el sitio **https://www.zabbix.com/documentation/current/manual/config/hosts/host**

3.1.2. Presentación de Hosts monitoreados por Zabbix en Grafana

En el capítulo anterior se presentó la integración de Zabbix con Grafana, en donde lo que se estableció es la comunicación de la base de datos de Zabbix con Grafana a través de API instalado, y así poder acceder a la información registrada en tiempo real.

Ahora corresponde interpretar la información que nos proporciona la base de datos para mostrarla de manera gráfica y de acuerdo con las necesidades del usuario final para que la información sea de utilidad.

En Grafana los usuarios pueden diseñar sus propios esquemas de visualización o dashboard particularizado a sus necesidades, pero también se puede utilizar templates ya elaborados por personal de desarrollo de Grafana o por usuarios en la red que presentaron alguna necesidad y como colaboración proporcionaron sus desarrollos de manera gratuita para que sean utilizados por otros usuarios. Estos templates o dashboard se pueden encontrar en el sitio: **https://grafana.com/grafana/dashboards.**

En muchos casos estos dashboards sirven de base para crear nuestras propias visualizaciones de resultados y se los personaliza de acuerdo con nuestras necesidades de presentación.

En el caso particular de este proyecto, se ha utilizado como base templates ya diseñados y se ha realizado customizaciones sobre los mismos adaptándolo a los elementos que se están monitoreando.

Por ejemplo, en el caso de monitoreo de Firewall Sophos se ha utilizado como base los templates ubicados en el sitio oficial de Grafana:

- **https://grafana.com/grafana/dashboards/12475**
- **https://grafana.com/grafana/dashboards/12332**

La vista de resultados del Firewall con las personalizaciones basada en estos 2 templates es la que se observa en la Figura 3.2.

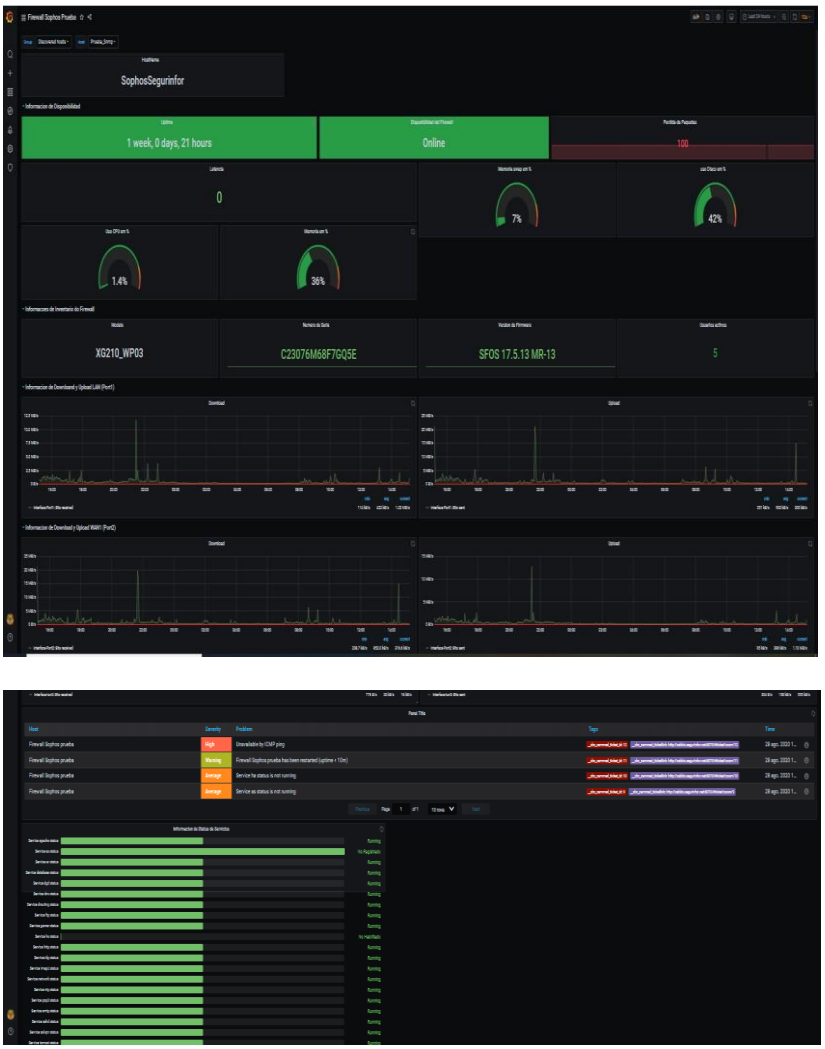


Figura 3.2. Vista de Monitoreo de Firewall Sophos en Grafana

En esta representación se puede observar el monitoreo de la salud del Firewall en sus diferentes componentes como son CPU, Memoria, Disco, Trafico a nivel de la red LAN y WAN, se observan los problemas que en el momento de la verificación posee el equipo con una escala de colores que indica su severidad y adicional se observa una etiqueta indicando que genero un ticket a Zammad.

Otra representación gráfica que se adaptó en este trabajo fue un dashboard que nos permite visualizar de manera general el estado de todos los dispositivos o aplicativos monitoreados en estademostración, en donde se muestra cada dispositivo y las alarmas quepodría tener presente de acuerdo con su severidad, y un detalle de todos los problemas que se tienen al momento de la verificación, así como también un histórico de los problemas que han surgido. En la misma también se muestra una etiqueta que nos muestra la notificación que realizo Zabbix a Zammad sobre las alertas presentadas debido a problemas que han ocurrido en el sistema de monitoreo implementado. A continuación, podemos observar en la Figura 3.3 la vista descrita.

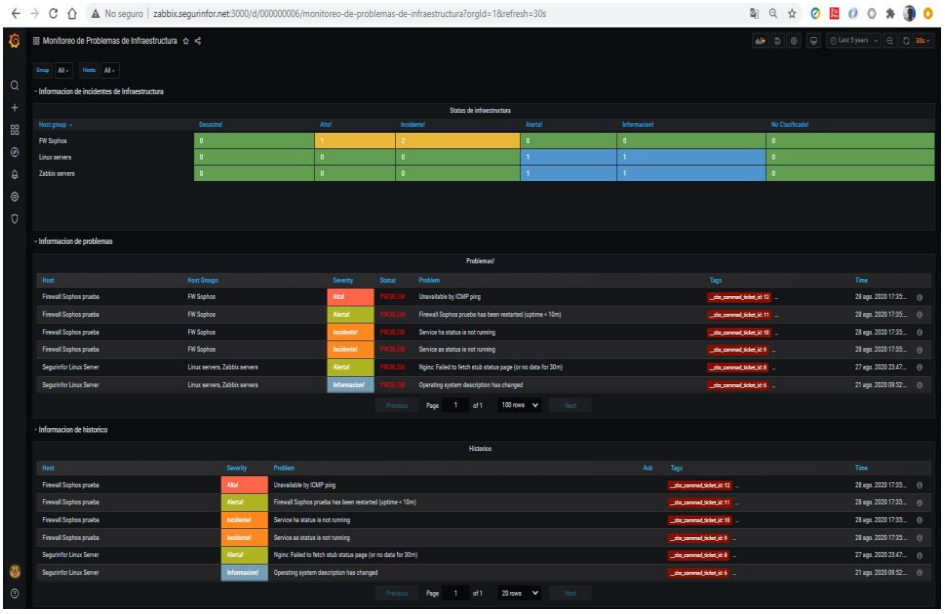


Figura 3.3. Vista de Monitoreo de Dispositivos integrados

3.1.3. Presentación de un ticket generado en Zammad por problema en elemento monitoreado en Zabbix.

Cuando se presenta un problema en algún elemento monitoreado en Zabbix a través de la comunicación Webhook se notifica de esta alerta a Zammad quien de manera automática genera un ticket que es entregado al administrador o al grupo designado a recibir estas notificaciones así

podemos ver en la Figura 3.4 como es el formato de un ticket generado por el aviso de un evento ocurrido en los elementos gestionados.

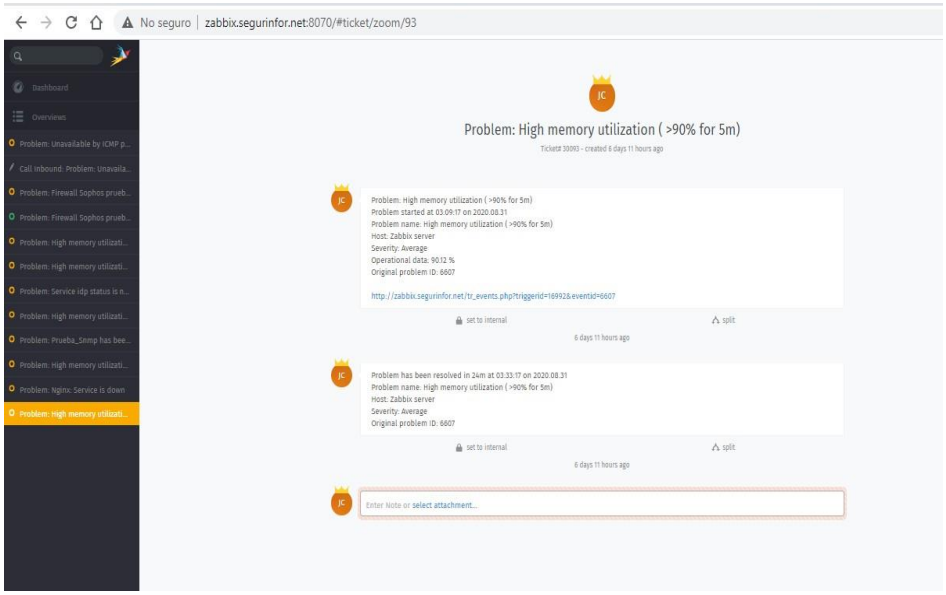


Figura 3.4. Ejemplo de ticket en Zammad por evento o alerta de un problema detectado

3.2. Simulación de uso de la solución con agentes activos

En el presente espacio se simulará un evento de indisponibilidad en los sistemas monitoreados, en donde se mostrará reflejado el comportamiento de cada una de las herramientas integradas con el cambio de estado de uno de sus elementos.

La simulación consistirá en la perdida de conectividad hacía el DNS de Google cuya IP pública es la 8.8.8.8, en donde procederemos a colocar una IP privada que no es alcanzable desde Internet, en este caso utilizaremos la dirección privada 192.168.1.1.

Antes de iniciar la simulación las pantallas en los diversos sistemas es la siguiente:

Zabbix:

A continuación, en la Figura 3.5, se muestra el estado de monitoreo en el sistema de Zabbix antes de iniciar el demo.

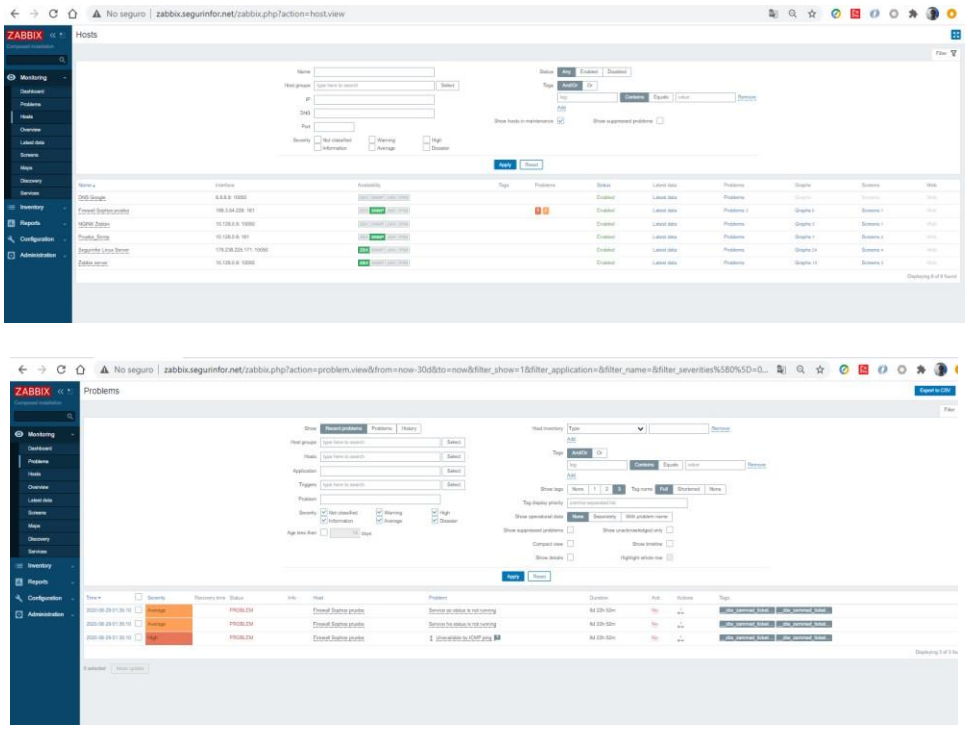


Figura 3.5. Estado de Monitoreo de Zabbix previo a la simulación

Grafana:

Estado del dashboard de “monitoreo de Problemas de Infraestructura en Grafana”, en donde como se observa en la Figura 3.6 muestra los problemas presentes al momento de la captura, previo al inicio de la prueba de indisponibilidad.

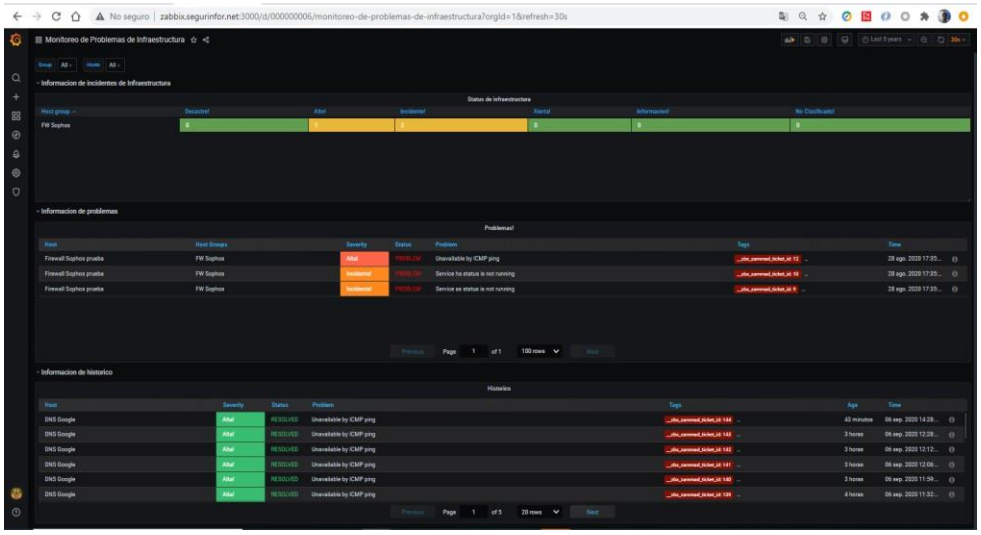


Figura 3.6. Estado de Monitoreo en Grafana previo a simulación

Zammad:

En Zammad, se presenta el dashboard de los tickets abiertos, en el cual aparecen los tickets que permanecen abierto al momento de la simulación, como se presenta en la Figura 3.7.

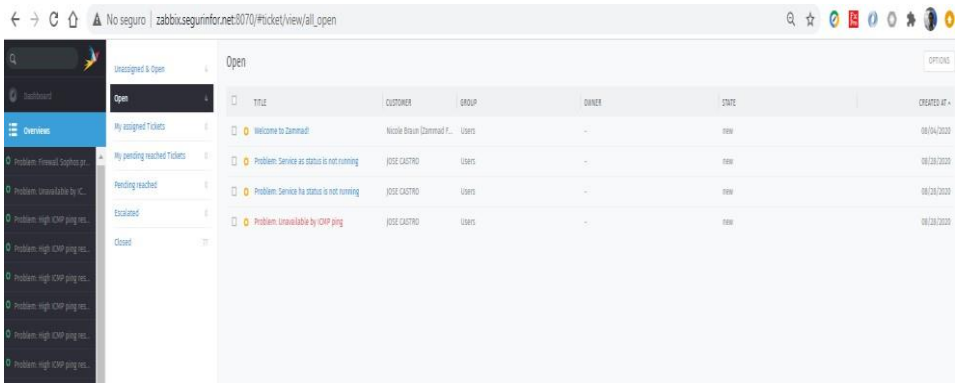


Figura 3.7. Estado de Tickets abiertos en Zammad previo al inicio de simulación

Ahora para la simulación procedemos a iniciar la simulación siguiendo los pasos a continuación:

- En Zabbix, procedemos a cambiar la dirección IP de DNS de Google a una dirección IP privada que no puede ser alcanzada en Internet,

como se muestra en la Figura 3.8 en donde vemos el host alarmado luego del cambio.

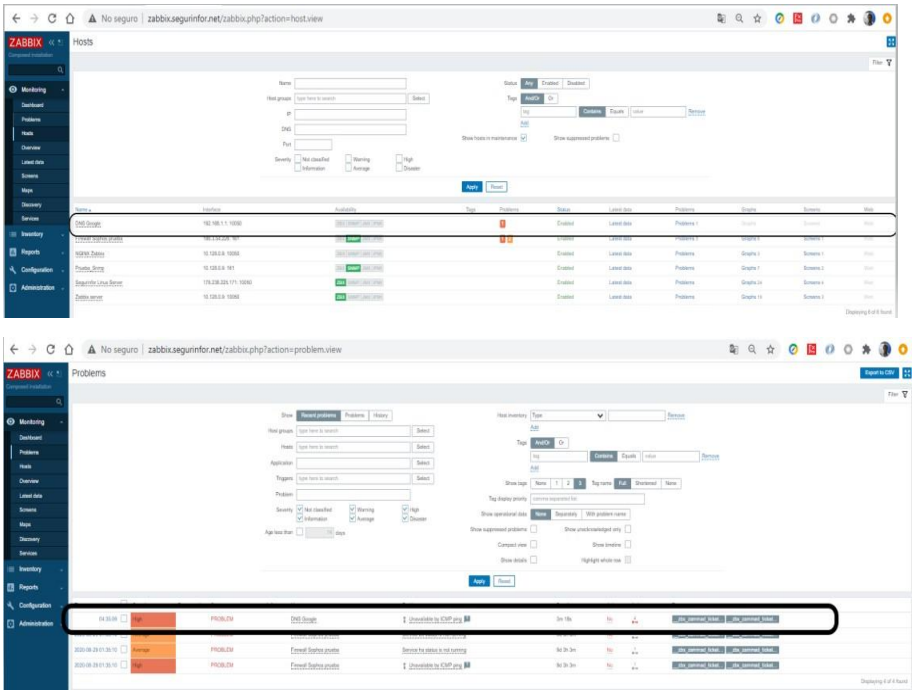


Figura 3.8. Host DNS Google en Zabbix alarmado luego de cambio de Dirección IP

- Ahora procedemos a revisar en Grafana el Host alarmado, lo que se encuentra remarcado en la Figura 3.9.

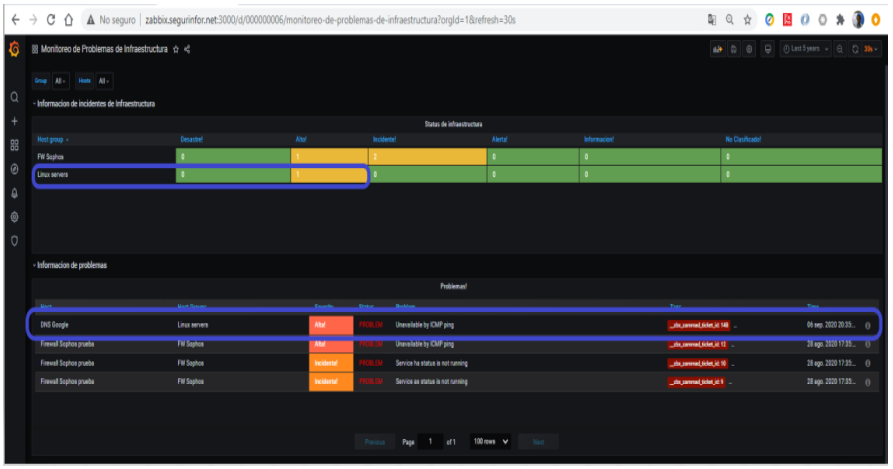


Figura 3.9. Host DNS Google alarmado en Grafana

- En Zammad revisamos la generación del ticket tanto en la pantalla principal, Figura 3.10, y observamos en detalle cómo fue generado, Figura 3.11. El ticket escalado posee la identificación: Ticket#30148

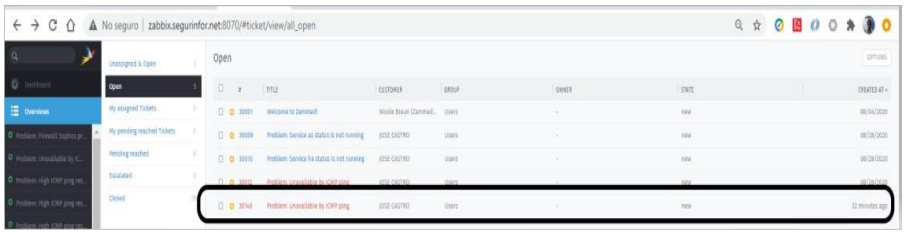


Figura 3.10. Alarma generada en Zammad por alerta generada por Host DNS Google

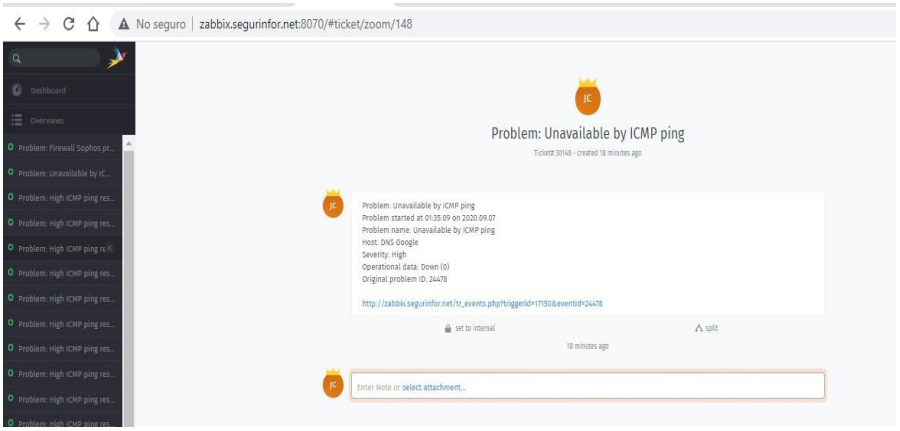


Figura 3.11. Detalle del ticket creado por pérdida de servicio de Host DNS Google

- Es de mencionar, a pesar de que no era objetivo de este proyecto, se configuro en Zammad la opción de enviar por medio de correo electrónico el detalle del ticket creado al agente o agentes que atenderán la notificación de alerta. Un ejemplo de la notificación vía correo es mostrado en la Figura 3.12.

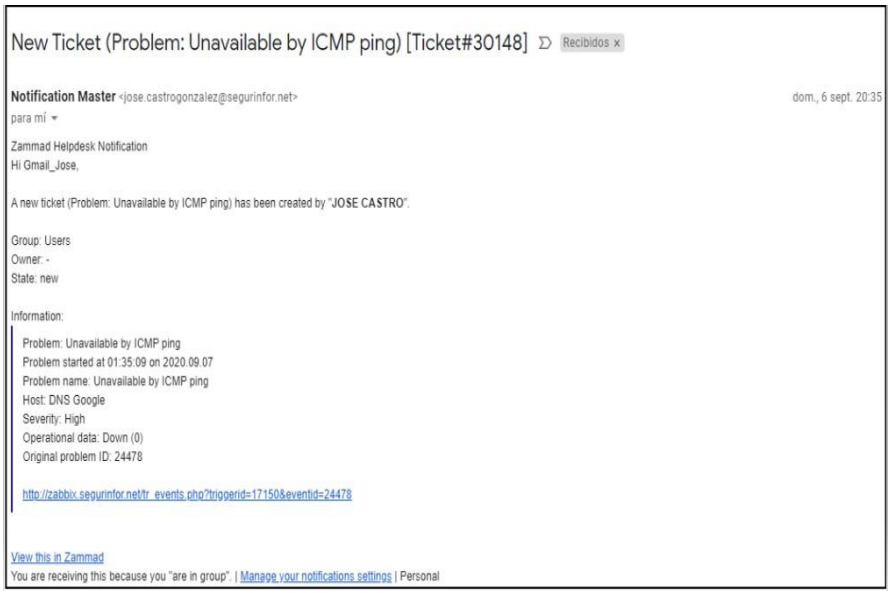


Figura 3.12. Notificación de alerta a través de Correo Electrónico sobre problema de Host DNS Google

Ahora procederemos a volver al sistema a la normalidad, procediendo a colocar la dirección IP correcta del host DNS Google. Luego de esto las alertas graficas en Zabbix y Grafana desaparecerán, y el ticket creado en Zammad se actualizará indicando la resolución como se muestra en la Figura 3.13.

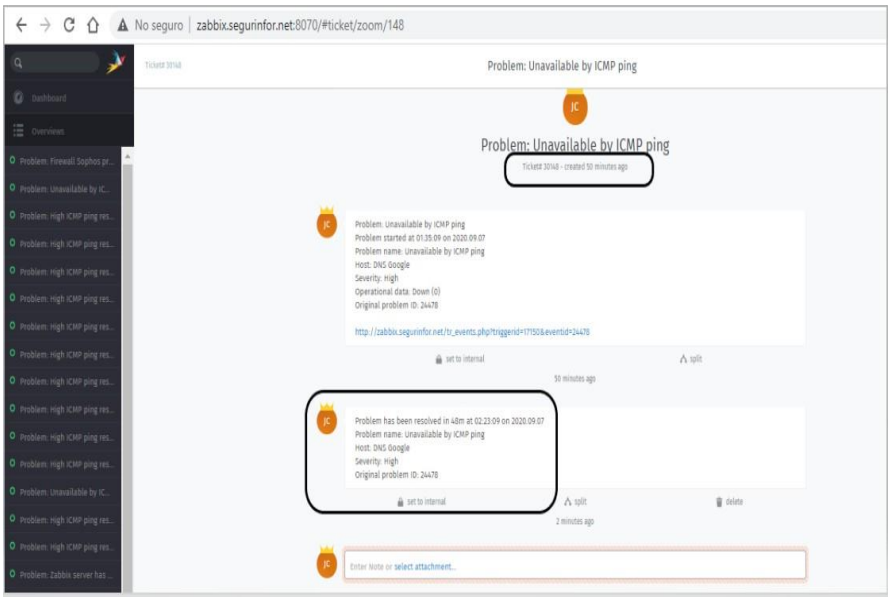


Figura 3.13. Actualización de Ticket #30148 luego de recuperación de servicio de Host DNS Google

El ticket deberá ser cerrado luego manualmente por el administrador o agente que lo atendió, una vez que haga las revisiones del caso y confirme que la alerta generada no persiste más en el host o dispositivo que presento la alarma.

CAPITULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

A lo largo de mi trayectoria profesional he podido observar que una de las principales problemáticas para los administradores de aplicativos, plataformas o infraestructuras es el conocer con anticipación o en el menor tiempo posible cuando uno de los elementos que conforman su sistema de gestión presentan un cambio de estado que derive a un problema o incidencia.

A pesar de la importancia de tener que contar con un sistema de gestión y monitoreo, la mayoría de los responsables dentro de las empresas ponen barreras en adquirirlo principalmente porque normalmente sus costos de implementación suelen ser alto, y en otra por la dificultad de los responsables técnicos de transmitir de manera financiera lo que representa en pérdidas económicas el tiempo que un servicio podría permanecer por fuera hasta ser atendido y resuelto.

La finalidad de este proyecto fue demostrar la capacidad de utilizar aplicaciones de código abierto e integrarlas para brindar una solución global a la gestión de problemas e incidencias orientado a dispositivo de red o de seguridad informática tratando de solventar lo indicado en los párrafos anteriores, y así logramos implementar una solución que permita administrar y monitorear los elementos dentro del sistema, presentar de manera ejecutiva y personalizada las gráficas de resultados y registrar y escalar las incidencias de manera automática a los responsables del servicio.

La integración de Zabbix, Grafana y Zammad demuestra también que hoy en día la utilización de API o WebHook permiten una interacción sencilla entre plataformas con propósitos diferentes, y que funcionan como una unidad coordinada en el momento de ocurrir un evento que requieran comunicar entre ellas.

El utilizar la nube como punto de desarrollo nos permitió monitorear sistemas o infraestructura de redes o seguridades independiente del lugar físico donde se encuentre, otra ventaja de tener el sistema en la nube es el ahorro del costo de hardware y otros gastos asociados al mantenimiento de este.

Se utilizo Docker sobre un sistema operativo Linux con el fin de aplicar el concepto de contenedor, y así implementar cada plataforma en un ambiente separado con sus propias características y de esta forma evitar conflictos de versiones de base de datos, tipo de web servers u otras similitudes que podrían existir entre los requisitos que requieran los aplicativos a integrar.

Se ha cumplido con los objetivos de la propuesta realizada, quedando la satisfacción de haber desarrollado una solución que va a ayudar en la prevención y detección de fallas en dispositivos de redes y seguridades contribuyendo en las empresas que lo adopten en un aumento del porcentaje de disponibilidad de los servicios prestados a los usuarios finales.

4.2. Recomendaciones Futuras

En el campo de gestión o administración de redes usando plataformas de código abierto existen muchas oportunidades de nuevos desarrollos o integraciones.

En el caso particular de este desarrollo mejoras posibles que se pueden desarrollar e implementar son que los tickets generados sean automáticamente delegados a algún agente de acuerdo con su nivel de experiencia. A nivel de integración, que Zammad automáticamente cierre el ticket si la alerta desaparece.

Un tema que me apasiona es la inteligencia artificial, y a nivel de sistemas de gestión los avances han sido mínimo por lo que recomiendo es desarrollar o integrar una plataforma de gestión que pueda tener la capacidad de tomar decisiones basadas en el aprendizaje con el fin de aplicar de manera automática una solución que permita volver al sistema a un estado normal.

BIBLIOGRAFÍA

[1] “WHAT IS ZABBIX.” [Online]. Available: https://www.zabbix.com/documentation/current/manual/introduction/about.

[2] “What is Grafana?” [Online]. Available: https://grafana.com/docs/grafana/latest/getting-started/what-is-grafana/.

[3] “Zammad.” [Online]. Available: https://docs.zammad.org/en/latest/about/zammad.html.

[4] “Guía de inicio rápido sobre cómo usar una VM de Linux.” [Online]. Available: https://cloud.google.com/compute/docs/quickstart-linux?hl=es-419.

[5] “¿Qué es DOCKER?” [Online]. Available: https://www.redhat.com/es/topics/containers/what-is-docker.

[6] “Install Docker Engine on CentOS.” [Online]. Available: https://docs.docker.com/engine/install/centos/.

[7] “Install Docker Compose.” [Online]. Available: https://docs.docker.com/compose/install/.

[8] F. Lee, “Zabbix: Using Docker Compose to install and upgrade Zabbix.” [Online]. Available: https://fabianlee.org/2019/10/06/zabbix-using-docker-compose-to-install-and-upgrade-zabbix/.

[9] “DOCKER COMPOSE.” [Online]. Available: https://www.zabbix.com/documentation/current/manual/installation/containers#docker_compose.

[10] “Grafana Docker image.” [Online]. Available: https://hub.docker.com/r/grafana/grafana/.

[11] “Install with Docker-Compose Zammad.” [Online]. Available: https://docs.zammad.org/en/latest/install/docker-compose.html.

[12] A. Zobnin, “Zabbix plugin for Grafana,” 2020. [Online]. Available: https://grafana.com/grafana/plugins/alexanderzobnin-zabbix-app.

[13] A. Zobnin, “Grafana-Zabbix Documentation: Configuration.” [Online]. Available: https://alexanderzobnin.github.io/grafana-zabbix/configuration/.

[14] “Zabbix + Zammad.” [Online]. Available: https://www.zabbix.com/integrations/zammad?fbclid=IwAR0XiQ_jjUEeQvCVXsvPQd4dGDA_mKujjrv02Y7i0TqoVkMVLMQN4Ldtreg.

[15] NewsMDirector, “¿Qué es un webhook y para qué sirve?,” 2020. [Online]. Available: https://www.mdirector.com/email-marketing/que-es-un-webhook.html.

[16] “WEBHOOKS.” [Online]. Available: https://www.zabbix.com/documentation/guidelines/webhooks.

[17] Zabbix SIA, “CONFIGURING A HOST,” 2020. [Online]. Available: https://www.zabbix.com/documentation/current/manual/config/hosts/host.