María del Rosario León 201423755

Juan Sebastián Millán 201516818

Jorge Andrés Gómez 201618492

Juan Camilo Ruiz 201617394

Andrés Felipe López 201531595

**Entrega Experimento 1 Final**

**Escenarios de pruebas**

Para el desarrollo de las pruebas de carga, se pobló el sistema con los siguientes datos:

Unidad Residencial:

{ "nombreUnidadResidencial": "Miracolina",

"id": "52121"

}

Residencia:

{ "nombreResidencia": "APTO401 T2",

"id": "1245", "propietario": "Jorge Gomez"

}

Hub:

{

"id": "41351"

}

Cerradura:

{

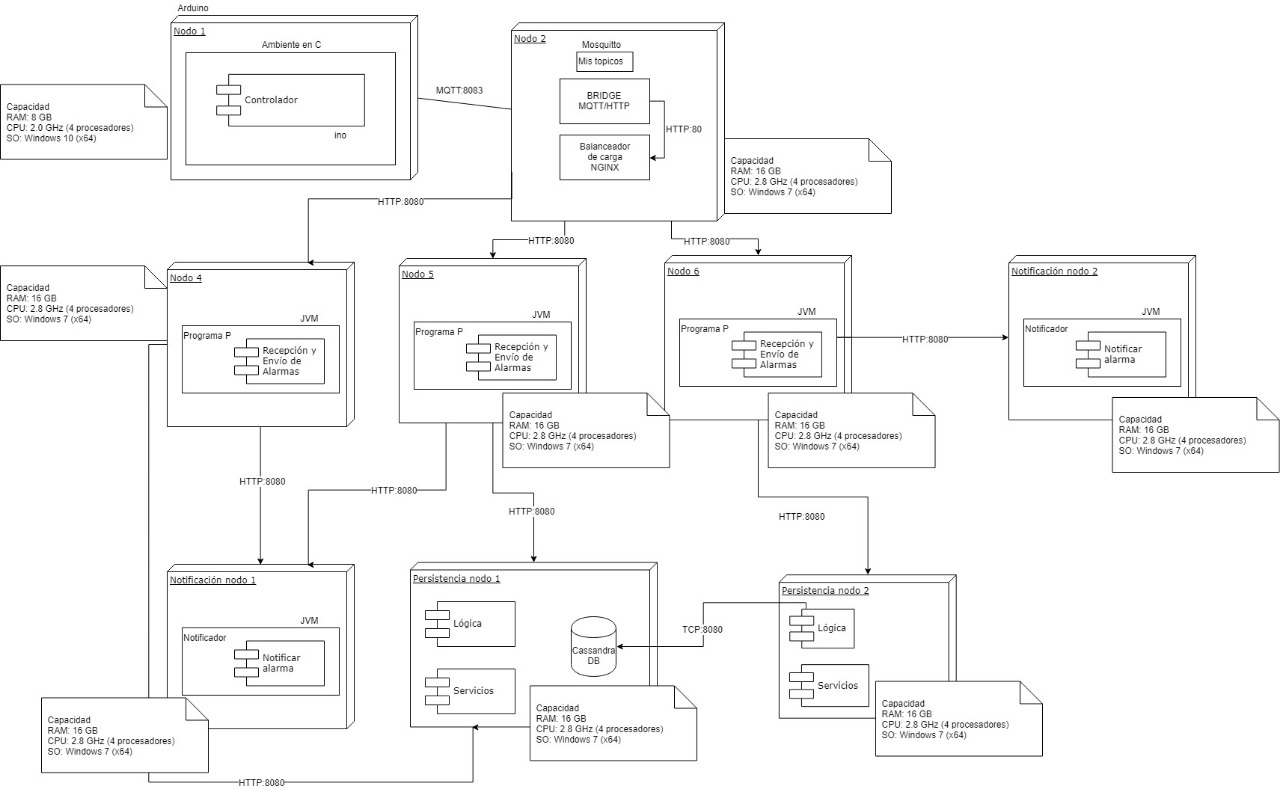
"id":"51451"

}

De igual forma es importante mencionar que cada vez que ocurría una alarma de alguna unidad residencial, residencia, hub o cerradura que no estuviera en la base de datos. Esta entidad se agregaba automáticamente.

**Diagrama de Despliegue**

En el diagrama que se presenta a continuación se denota la arquitectura utilizada donde se evidencian aspectos tales como protocolos de comunicación, ambientes de ejecución, dispositivos junto con sus especificaciones de hardware, y artefactos. Esta arquitectura fue la que se utilizó explícitamente en el proyecto. En este diagrama se pueden visualizar 3 nodos que fueron utilizados para balancear la carga en los programas P que se encargaban de la recepción y envío de alarmas. Sin embargo, para lograr el resultado esperado con respecto a rendimiento y escalabilidad se requiere de una mayor cantidad de nodos realizando estas mismas funciones.



**Resultados pruebas de carga**

Al realizar las pruebas de carga a la funcionalidad de notificación de alarmas, se obtuvieron los siguientes resultados:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Notificación Alarmas** | | | | | |
| Número de muestras | Tiempo respuesta medio (ms) | Min (ms) | Max (ms) | Porcentaje error (%) | Rendimiento |
| 500 | 5 | 2 | 22 | 0,00 | 8,3/sec |
| 1000 | 5 | 1 | 40 | 0,00 | 16,6/sec |
| 2000 | 5 | 1 | 43 | 0,00 | 33,1/sec |
| 5000 | 5 | 1 | 238 | 0,00 | 81,9/sec |
| 10000 | 12 | 1 | 1396 | 0,00 | 160,0/sec |
| 20000 | 44 | 1 | 3621 | 0,00 | 267,5/sec |
| 40000 | 77 | 1 | 4543 | 0,00 | 451,5/sec |
| 50000 | 457 | 1 | 10618 | 0,00 | 515,7/sec |
| 60000 | 1266 | 1 | 18831 | 19,55 | 430,2/sec |
| 70000 | 4133 | 1 | 25690 | 34,43 | 495,2/sec |
| 80000 | 6174 | 1 | 72297 | 37,82 | 418,8/sec |

**Análisis Resultados**

Al observar los resultados obtenidos se puede apreciar que la arquitectura soporta correctamente 50000 notificaciones de alarmas en una ventana de 60 segundos con un tiempo de respuesta inferior a 1 segundo. Sin embargo, una vez el número de alarmas supera los 50000, el tiempo de respuesta promedio pasa a ser mayor a 1 segundo y aumenta proporcionalmente con el número de alarmas notificadas. De igual forma, si las alarmas son inferiores a 50000, todas estas son notificadas sin problema y el porcentaje de error es 0,0%. Pero cuando las alarmas superan 50000, empiezan a ocurrir errores y el porcentaje de error aumenta proporcionalmente con el número de notificaciones.

Teniendo en cuenta que los escenarios de calidad establecidos son:

“En el escenario más crítico, la arquitectura debe ser capaz de manejar el envío de alarmas provenientes de los 200 inmuebles ubicados en las 1500 unidades residenciales que Yale espera alcanzar para 2020. Las peticiones son enviadas en una ventana de 1 minuto con 0% de error y tiempos de respuesta inferiores a 1 segundo. Esto aplica para los diferentes tipos de emergencia.”

Se concluye que la arquitectura actual no cumple con los escenarios de calidad solicitados por Yale, por lo que se propone realizar los siguientes cambios para cumplirlos en un futuro. En primer lugar, aumentar la redundancia de los programas P que se encargan de enviar las notificaciones de las alarmas a los a las máquinas que simulan los servidores de correo. Esto con el propósito de aumentar la capacidad computacional de toda la arquitectura. En segundo lugar, introducir concurrencia en los programas P para repartir las tareas entre los procesadores de las distintas máquinas y así mejorar el desempeño. En tercer lugar, sería útil implementar la táctica de desempeño que trata de incrementar los recursos disponibles de los dispositivos para que cada se puedan tener una mayor cantidad de procesadores lógicos en las maquinas que permiten el funcionamiento del programa P. Finalmente, también aplicaremos la táctica de desempeño que consiste en reducir el número de intermediarios en la arquitectura, para así mejorar los tiempos de respuesta y cumplir con los escenarios de calidad.