**VM Azure a Base de Datos en VNet Diferente**

El escenario es una VM (que aloja un servicio) en la VNet A intentando conectarse a una Base de Datos en la VNet B, ambas en la misma suscripción de Azure, y la conexión falla.

**Pasos para Diagnosticar el Problema de Conectividad:**

**Verificación de Conectividad Básica (Desde la VM del Servicio):**

* + **Ping (ICMP):**
    - ping <IP\_privada\_de\_la\_DB>: Intentar hacer ping desde la VM del servicio a la IP privada de la base de datos.
    - **Resultado:** Si el ping falla, indica un problema a nivel de red (firewall, routing). Si el ping funciona, la conectividad básica de red está establecida, y el problema podría ser el puerto específico de la DB o un firewall a nivel de aplicación.
    - **Consideración:** Azure NSG por defecto no permite ICMP entre subredes, por lo que un fallo de ping no es siempre concluyente por sí solo.
  + **Telnet / Netcat (TCP Connectivity):**
    - telnet <IP\_privada\_de\_la\_DB> <PUERTO\_DB> (Ej: telnet 10.0.2.4 5432 para PostgreSQL): Esto es crucial. Intenta establecer una conexión TCP directa al puerto de la base de datos.
    - **Resultado:**
      * Si se conecta (aparece una pantalla en blanco o un mensaje de conexión), significa que el puerto está abierto y es alcanzable desde la VM del servicio. El problema no es de red ni de NSG, sino probablemente de la aplicación (cadena de conexión, credenciales, firewall de la propia DB).
      * Si falla (timeout, "Connection refused"), el problema está en la red o en un firewall que bloquea el puerto.

**Revisión de Configuraciones de Red Virtual en Azure (Azure Portal / Azure CLI / Azure PowerShell):**

* + **VNet Peering:**
    - **Concepto:** Para que dos redes virtuales (VNet\_A y VNet\_B) en la misma suscripción se comuniquen directamente, deben estar interconectadas mediante **VNet Peering**. Es la causa más común de problemas de conectividad entre VNets.
    - **Acción:** Ir al Azure Portal, navegar a cada VNet (VNet\_A y VNet\_B), y en la sección **"Peerings"**, verificar que el peering esté configurado en **ambas direcciones** y que su estado sea "Conectado".
    - **Configuraciones a Revisar en el Peering:**
      * Allow Virtual Network Access: Debe estar habilitado en *ambos lados* del peering.
      * Allow Forwarded Traffic: Si el tráfico se enruta a través de un Network Virtual Appliance (NVA) o un firewall central, esto debe estar habilitado.
      * Allow Gateway Transit / Use Remote Gateways: Si se espera que el tráfico use un Gateway VPN/ExpressRoute en la VNet remota.
  + **Espacios de Direcciones IP de las VNets:**
    - **Acción:** Asegurarse de que los rangos de direcciones IP de las subredes en VNet\_A y VNet\_B **no se solapen**. El solapamiento de IPs es una causa común de fallos de peering.
  + **Network Security Groups (NSGs):**
    - **Concepto:** Los NSGs actúan como firewalls a nivel de interfaz de red (NIC) o subred. Pueden bloquear el tráfico incluso si el VNet Peering está bien configurado.
    - **Acción:**
      * **NSG(s) de la VM de Servicio (VNet A):**
        + **Reglas de Salida (Outbound Security Rules):** Verificar que exista una regla que permita el tráfico **al puerto de la base de datos** (e.g., 5432 para PostgreSQL, 1433 para SQL Server) hacia el rango IP de la subred donde reside la DB (o su IP específica).
      * **NSG(s) de la DB (VNet B):**
        + **Reglas de Entrada (Inbound Security Rules):** Verificar que exista una regla que permita el tráfico **desde el rango IP de la subred de la VM de servicio** (o su IP específica) hacia el puerto de la base de datos.
      * **Prioridad de Reglas:** Los NSGs procesan las reglas por número de prioridad (más bajo = mayor prioridad). Asegurarse de que una regla de "denegar todo" de baja prioridad no esté bloqueando el tráfico.
      * **Reglas de Flujo Efectivas (Effective Security Rules):** Utilizar la opción "Effective security rules" en el Azure Portal (dentro de la configuración de la interfaz de red o subred) para ver las reglas finales que se están aplicando. Esto es extremadamente útil para depurar.
  + **Rutas Definidas por el Usuario (UDRs) / Tablas de Rutas:**
    - **Concepto:** Si hay Network Virtual Appliances (NVAs) como firewalls de terceros o un Azure Firewall en la topología, o si el tráfico debe pasar por un gateway, las UDRs pueden desviar el tráfico de su ruta predeterminada.
    - **Acción:** Verificar si las subredes de la VM de servicio y la DB tienen tablas de rutas (Route Tables) asociadas. Si las tienen, asegurarse de que haya rutas que dirijan el tráfico correctamente hacia la otra VNet o a través del NVA/firewall central. Si hay un NVA/firewall, también debe tener las reglas de firewall configuradas para permitir el tráfico.
  + **Firewall Interno de la Base de Datos (si aplica):**
    - **Concepto:** Si la base de datos es un servicio gestionado de Azure (ej., Azure SQL Database, Azure Database for PostgreSQL/MySQL), estos servicios tienen su propio firewall a nivel de servidor/base de datos.
    - **Acción:** En la configuración del servicio de base de datos en Azure Portal, verificar las **"Reglas de firewall de red"** y asegurarse de que el rango IP de la subred de la VM del servicio (o la IP específica de la VM) esté permitido para acceder a la base de datos.

**Herramientas de Diagnóstico de Azure (Network Watcher):**

* + **IP Flow Verify:** Herramienta en Network Watcher que permite simular un flujo de tráfico (origen, destino, puerto, protocolo) y verificar si el tráfico es permitido o denegado por las reglas de NSG o UDRs. Proporciona la regla exacta que lo bloquea.
  + **Next Hop:** Muestra el siguiente salto para un paquete que sale de una VM, útil para entender el enrutamiento.
  + **Connection Troubleshoot:** Realiza un diagnóstico completo de conectividad entre dos máquinas virtuales o un punto de origen y destino, analizando NSGs, UDRs, etc.

**Verificación a Nivel de Aplicación y Credenciales:**

* + Si todos los pasos anteriores indican que la red está bien, el problema reside en la capa de la aplicación.
  + **Logs de la Aplicación:** Revisar los logs de la aplicación en la VM del servicio para buscar errores de conexión a la base de datos, credenciales inválidas, problemas con el driver de la base de datos, o cadenas de conexión incorrectas.
  + **Credenciales:** Asegurarse de que las credenciales de la base de datos sean correctas y estén actualizadas.

**Resolución DNS en Máquina Linux (api.empresa.local)**

El problema es que una máquina Linux no puede resolver el nombre de un dominio interno (api.empresa.local).

**Pasos para Diagnosticar y Resolver:**

**Verificación de Conectividad Básica al Servidor DNS:**

* + **Identificar Servidor DNS Interno:** Determinar la IP del servidor DNS interno que debería resolver empresa.local.
  + ping <IP\_del\_servidor\_DNS\_interno>: Asegurarse de que la máquina Linux pueda alcanzar el servidor DNS interno a nivel de red. Si esto falla, el problema es de conectividad de red (firewall, routing), no de DNS directamente.
  + ip a o ifconfig: Verificar la configuración IP de la interfaz de red de la máquina Linux (dirección IP, máscara de subred, gateway).

**Revisión de la Configuración DNS del Cliente Linux:**

* + **/etc/resolv.conf:** Este archivo es la fuente principal de la configuración DNS para la mayoría de los sistemas Linux.
    - cat /etc/resolv.conf:
      * nameserver <IP\_del\_servidor\_DNS\_interno>: Asegurarse de que la primera entrada nameserver apunte a la IP correcta del servidor DNS interno. Si hay varias entradas, se prueban en orden.
      * search empresa.local: La directiva search es crucial para la resolución de nombres cortos (no FQDN). Si no está presente, cuando la aplicación intenta resolver api, no le agregará automáticamente empresa.local, por lo que no buscará api.empresa.local.
  + **Gestores de Red:** Si la distribución usa systemd-resolved (Ubuntu recientes), NetworkManager, o netplan (Ubuntu Server), estos servicios gestionan /etc/resolv.conf dinámicamente. A veces, el archivo es un symlink y las configuraciones se hacen en otro lugar (ej., /etc/netplan/\*.yaml para netplan, o a través de los comandos de nmcli para NetworkManager).
    - Para systemd-resolved: systemd-resolve --status mostrará los servidores DNS actuales.

**Pruebas de Resolución DNS:**

* + dig api.empresa.local: Herramienta potente para probar la resolución DNS. Proporciona mucha información sobre la respuesta del DNS.
    - Si devuelve un NXDOMAIN (Non-Existent Domain) o un timeout, significa que el servidor DNS no pudo resolverlo o no es accesible.
    - Verificar la sección ANSWER SECTION para la IP esperada.
    - dig @<IP\_del\_servidor\_DNS\_interno> api.empresa.local: Probar la resolución directamente contra el servidor DNS interno para descartar problemas en la configuración de resolv.conf del cliente.
  + nslookup api.empresa.local: Otra herramienta para probar la resolución DNS.
  + host api.empresa.local: Similar a dig/nslookup.

**Diagnóstico del Servidor DNS Interno (Si el Problema Persiste):**

* + Si las pruebas de resolución fallan incluso cuando se apunta directamente al servidor DNS interno, el problema podría estar en el propio servidor DNS:
    - **Estado del Servicio DNS:** SSH al servidor DNS y verificar que el servicio DNS (ej., bind9 en Linux, DNS Server en Windows Server) esté en ejecución. (sudo systemctl status bind9 o Get-Service DNS).
    - **Registros DNS:** Confirmar que el registro A (o CNAME) para api.empresa.local existe y es correcto dentro de la zona empresa.local en el servidor DNS.
    - **Firewall del Servidor DNS:** Asegurarse de que el firewall del servidor DNS (ej., ufw, firewalld en Linux, Windows Firewall) permita el tráfico entrante en el puerto 53 (UDP y TCP).
    - **Reenvío de Zonas (Forwarders):** Si el servidor DNS interno reenvía consultas a otros servidores DNS (ej., DNS público, otro DNS interno), verificar que esos reenvíos estén configurados correctamente y que los servidores de reenvío sean accesibles.
    - **Logs del Servidor DNS:** Revisar los logs del servidor DNS para errores de resolución, problemas de zona o advertencias.

**Pasos para la Resolución:**

1. **Corregir /etc/resolv.conf:**
   * Si la IP del nameserver es incorrecta o falta, corregirla.
   * Añadir la directiva search empresa.local si no está presente.
   * Si un gestor de red está sobrescribiendo /etc/resolv.conf, configurar el DNS a través de ese gestor (ej., sudo netplan apply después de editar el archivo .yaml, o nmcli con mod <connection\_name> ipv4.dns <ip\_dns>).

Ejemplo de /etc/resolv.conf corregido:  
 nameserver 192.168.1.10 # IP de tu servidor DNS interno

nameserver 8.8.8.8 # (Opcional) DNS público de fallback

search empresa.local # Dominio de búsqueda interno

1. **Reiniciar el Servicio de Red:** Después de modificar la configuración de red, a menudo es necesario reiniciar el servicio de red o la interfaz para que los cambios surtan efecto (ej., sudo systemctl restart networking o sudo systemctl restart NetworkManager).
2. **Corregir Registros DNS (en el servidor DNS):** Si el problema es que el registro api.empresa.local no existe o es incorrecto en el servidor DNS, crearlo o corregirlo.
3. **Ajustar Reglas de Firewall:** Si el firewall del servidor DNS o del cliente está bloqueando el tráfico DNS (puerto 53), ajustar las reglas para permitirlo.

**Herramientas y Comandos Frecuentes para Diagnosticar Problemas de Red entre Contenedores, VMs o Nodos Kubernetes**

Estas herramientas son esenciales para el troubleshooting de red en diferentes capas y entornos complejos:

* **Comandos de Conectividad Básica:**
  + ping <destino\_ip\_o\_hostname>: Verificación rápida de la conectividad a nivel de capa 3 (ICMP). Útil para descartar problemas de enrutamiento o accesibilidad básica.
  + traceroute <destino\_ip\_o\_hostname> (Linux) / tracert <destino\_ip\_o\_hostname> (Windows): Muestra la ruta que toma un paquete para llegar a un destino, identificando cuellos de botella o puntos de fallo en la red.
  + telnet <destino\_ip> <puerto> / nc -zv <destino\_ip> <puerto> (Netcat): Fundamental para verificar la conectividad TCP a un puerto específico. Si telnet conecta, el puerto está abierto y el servicio está escuchando. Si falla, hay un bloqueo (firewall) o el servicio no está activo/escuchando.
* **Información de Interfaces y Rutas:**
  + ip a / ifconfig (Linux): Muestra todas las interfaces de red, sus direcciones IP, estado (UP/DOWN), y detalles como direcciones MAC.
  + ip route / netstat -r (Linux): Muestra la tabla de enrutamiento del sistema. Es crucial para verificar si los paquetes se están dirigiendo al gateway o destino correcto, especialmente en entornos con múltiples interfaces o redes.
  + arp -a (Linux/Windows) / ip n (Linux): Muestra la tabla ARP (Address Resolution Protocol), que mapea IPs a direcciones MAC. Útil para problemas de capa 2.
* **Conexiones y Puertos:**
  + netstat -tulnp (Linux): Muestra todas las conexiones de red (TCP/UDP), puertos en estado LISTENING, y los IDs de proceso (PID) que están utilizando esos puertos. Muy útil para ver qué servicios están escuchando y qué conexiones activas existen.
  + ss -tunap (Linux): Una alternativa más rápida y moderna a netstat, especialmente útil en sistemas con muchas conexiones.
  + lsof -i (Linux): Lista todos los archivos abiertos por procesos, incluyendo sockets de red. Permite ver qué procesos están usando qué puertos y conexiones.
* **Resolución DNS:**
  + dig <dominio>: La herramienta más potente para consultar servidores DNS y ver respuestas detalladas (registros A, CNAME, MX, etc.), así como el servidor DNS que respondió.
  + nslookup <dominio>: Otra herramienta para la resolución DNS, más sencilla.
  + host <dominio>: Similar a dig/nslookup.
  + cat /etc/resolv.conf (Linux): Muestra los servidores DNS configurados para el cliente. Es importante verificar que apunten a los servidores DNS correctos (internos o externos).
* **Análisis de Paquetes:**
  + tcpdump (Linux): Herramienta de línea de comandos para capturar y analizar el tráfico de red en una interfaz específica. Permite ver los paquetes que entran y salen, útil para determinar si el tráfico está llegando o si el servicio está respondiendo.
    - Ejemplo: tcpdump -i eth0 host <IP\_destino> and port <PUERTO>
  + Wireshark (o tshark para CLI): Un analizador de protocolos de red gráfico/CLI. Se usa para un análisis más profundo de capturas de tráfico (.pcap generados por tcpdump).
* **Herramientas HTTP/Aplicación:**
  + curl -v <URL>: Realiza una solicitud HTTP/S y muestra información detallada sobre el proceso de conexión, encabezados, códigos de estado HTTP y mensajes de error. Muy útil para depurar servicios web y API.
  + wget <URL>: Para descargar contenido de una URL, útil para probar la accesibilidad de recursos web.
* **Herramientas Específicas de Contenedores y Kubernetes:**
  + docker exec -it <container\_id> bash / kubectl exec -it <pod\_name> -- bash: Permite acceder al shell dentro de un contenedor o pod para ejecutar las herramientas de red mencionadas anteriormente en su contexto de red aislado.
  + docker inspect <container\_id>: Muestra detalles de la configuración de red de un contenedor (IP, puertos mapeados, red a la que pertenece).
  + kubectl describe pod <pod\_name>: Proporciona información detallada sobre un pod de Kubernetes, incluyendo su IP, eventos y volúmenes montados.
  + kubectl logs <pod\_name>: Para ver los logs de la aplicación dentro del pod, que a menudo contienen errores de conexión de red.
  + kubectl get svc / kubectl get ingress: Para verificar la configuración de los servicios e ingresses de Kubernetes, que exponen las aplicaciones y controlan el enrutamiento del tráfico.
  + kubectl get networkpolicy: Si se están utilizando Network Policies en Kubernetes, es crucial revisar si alguna política está bloqueando el tráfico deseado.
  + **Imágenes de Depuración:** Es común usar imágenes de Docker o Kubernetes específicas para depuración (ej., alpine/helm-network-tools, nicolaka/netshoot) que vienen con herramientas de red preinstaladas para facilitar el troubleshooting dentro de los contenedores.
  + **Logs de kube-proxy:** Monitorear los logs de kube-proxy en los nodos para problemas relacionados con el enrutamiento del servicio de Kubernetes.

La combinación y el uso sistemático de estas herramientas permiten diagnosticar y resolver la mayoría de los problemas de red en infraestructuras complejas.