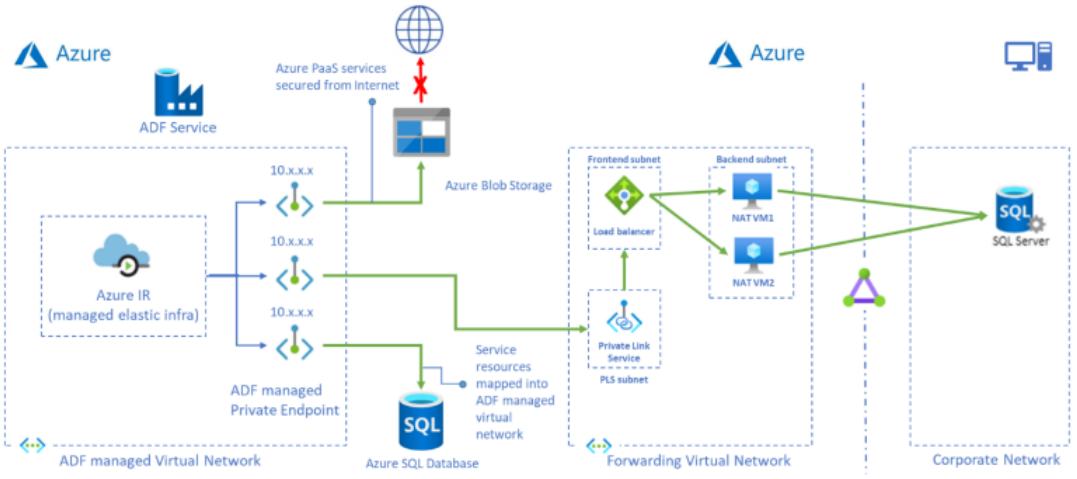
Third party private endpoint - troubleshoot on forwarding VM

Last updated by | Ruoyu Li | Apr 2, 2023 at 7:44 PM PDT

Contents

Background

For data store does not provide first party private endpoint support and customer want to use Managed VNet IR to establish private connection, then we have to follow below solution to achieve that. Connect to SQL Server from MVNet IR is the most common scenario when customer has to setup below topology.



It's a relatively complex setup. To diagnose a test connection failure, there are multiple areas to check as failure could happen anywhere in between **Managed VNet - Forwarding VNet (PLS, forwarding VM) - SQL Server**. It will be helpful if we could narrow down a little bit from above topology.

Narrow down: Managed VNet or Forwarding VNet?

Managed VNet troubleshoot

For this part, it's quite similar with those first party managed private endpoint. Please refer to this TSG xxxxx.

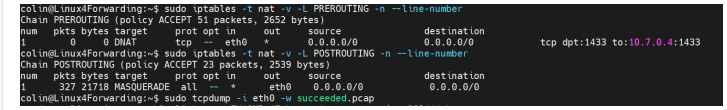
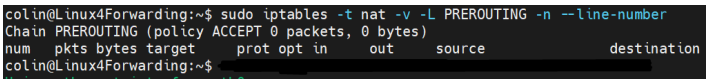
Forwarding VNet troubleshoot

Inside forwarding VNet, there is a Linux backend VM used for traffic forwarding. The good thing is that this VM is owned by customer, so we can perform several troubleshooting steps on it.

Check SNAT and DNAT rules are present and correct

On forwarding VM, traffic forwarding is achieved by configuring iptables rules. So it's important to ensure those forwarding rules are present and correct. You can run below two commands to check it.

```
sudo iptables -t nat -v -L PREROUTING -n --line-number
sudo iptables -t nat -v -L POSTROUTING -n --line-number
```

expected result	non-expected result (DNAT rule is missing)
	

Collect TCPDump trace to check network traffic

TCPDump collection Steps:

1. Install TCPdump (Debian/Ubuntu):
   `$ sudo apt-get install tcpdump`

2. Start capturing, if you are using default interface eth0:
   `$ sudo tcpdump -i eth0 -w dump.pcap`

3. Reproduce your issue.

4. In the former terminal, press **Ctrl + C** to stop capture.

```
colin@Linux4Forwarding:~$ sudo tcpdump -i eth0 -w succeeded.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C252 packets captured
253 packets received by filter
0 packets dropped by kernel
colin@Linux4Forwarding:~$ ^C
```

Example #1 : network trace for working scenario

Below shows network packets collected for a working scenario.

Frame Summary - tcp.port==1433

Find

Frame Number	Time Date Local Adjusted	Time Offset	Source	Destination	Protocol Name	Description
43	11:18:25 AM 2/8/2023	6.2196210	10.7.0.6	10.7.0.7	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87377945, Ack=0, Win=64860 (Negotiating scale factor 0x8) = 64860
46	11:18:25 AM 2/8/2023	6.2204690	10.7.0.7	10.7.0.4	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87377945, Ack=0, Win=64860 (Negotiating scale factor 0x8) = 64860
47	11:18:25 AM 2/8/2023	6.2242670	10.7.0.4	10.7.0.7	TCP	TCP:Flags=.E.A..S., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032551812, Ack=87377946, Win=65535 (Negotiated scale factor 0x8) = 16776960
48	11:18:25 AM 2/8/2023	6.2242930	10.7.0.7	10.7.0.6	TCP	TCP:Flags=.E.A..S., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032551812, Ack=87377946, Win=65535 (Negotiated scale factor 0x8) = 16776960
49	11:18:25 AM 2/8/2023	6.2249690	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87377946, Ack=3032551813, Win=49173 (scale factor 0x8) = 12588288
50	11:18:25 AM 2/8/2023	6.2249740	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87377946, Ack=3032551813, Win=49173 (scale factor 0x8) = 12588288
51	11:18:25 AM 2/8/2023	6.2421480	10.7.0.6	10.7.0.7	TDS	TDS:Prelogin, Version = Undefined TDS version(0x74000004), SPID = 0, PacketID = 1, Flags=...AP..., SrcPort=1025, DstPort=1433, PayloadLen=94, Seq=87377946 -...
52	11:18:25 AM 2/8/2023	6.2421790	10.7.0.7	10.7.0.4	TDS	TDS:Prelogin, Version = Undefined TDS version(0x74000004), SPID = 0, PacketID = 1, Flags=...AP..., SrcPort=1025, DstPort=1433, PayloadLen=94, Seq=87377946 -...
53	11:18:25 AM 2/8/2023	6.2429990	10.7.0.4	10.7.0.7	TDS	TDS:Response, Version = Undefined TDS version(0x74000004), SPID = 0, PacketID = 1, Flags=...AP..., SrcPort=1433, DstPort=1025, PayloadLen=54, Seq=30325518...
54	11:18:25 AM 2/8/2023	6.2430040	10.7.0.7	10.7.0.6	TDS	TDS:Response, Version = Undefined TDS version(0x74000004), SPID = 0, PacketID = 1, Flags=...AP..., SrcPort=1433, DstPort=1025, PayloadLen=54, Seq=30325518...
55	11:18:25 AM 2/8/2023	6.2541280	10.7.0.6	10.7.0.7	TLS	TLS:TLS Rec Layer-1 HandShake: Client Hello.
56	11:18:25 AM 2/8/2023	6.2541550	10.7.0.7	10.7.0.4	TLS	TLS:TLS Rec Layer-1 HandShake: Client Hello.
57	11:18:25 AM 2/8/2023	6.2582020	10.7.0.4	10.7.0.7	TLS	TLS:TLS Rec Layer-1 HandShake: Server Hello. Certificate. Server Key Exchange. Server Hello Done.
58	11:18:25 AM 2/8/2023	6.2582200	10.7.0.7	10.7.0.6	TLS	TLS:TLS Rec Layer-1 HandShake: Server Hello. Certificate. Server Key Exchange.
59	11:18:25 AM 2/8/2023	6.2582220	10.7.0.7	10.7.0.6	TCP	TCP:[Continuation to #58] [Bad CheckSum]Flags=...AP..., SrcPort=1433, DstPort=1025, PayloadLen=13, Seq=3032553103 - 3032553116, Ack=87378223, Win=1638...
60	11:18:25 AM 2/8/2023	6.2590030	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378223, Ack=3032553116, Win=49168 (scale factor 0x8) = 12587008
61	11:18:25 AM 2/8/2023	6.2590080	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378223, Ack=3032553116, Win=49168 (scale factor 0x8) = 12587008
62	11:18:25 AM 2/8/2023	6.2618700	10.7.0.6	10.7.0.7	TLS	TLS:TLS Rec Layer-1 HandShake: Client Key Exchange.; TLS Rec Layer-2 Cipher Change Spec; TLS Rec Layer-3 HandShake: Encrypted Handshake Message.
63	11:18:25 AM 2/8/2023	6.2618830	10.7.0.7	10.7.0.4	TLS	TLS:TLS Rec Layer-1 HandShake: Client Key Exchange.; TLS Rec Layer-2 Cipher Change Spec; TLS Rec Layer-3 HandShake: Encrypted Handshake Message.
64	11:18:25 AM 2/8/2023	6.2644040	10.7.0.4	10.7.0.7	TLS	TLS:TLS Rec Layer-1 Cipher Change Spec; TLS Rec Layer-2 HandShake: Encrypted Handshake Message.
65	11:18:25 AM 2/8/2023	6.2644320	10.7.0.7	10.7.0.6	TLS	TLS:TLS Rec Layer-1 Cipher Change Spec; TLS Rec Layer-2 HandShake: Encrypted Handshake Message.
66	11:18:25 AM 2/8/2023	6.2819280	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378389, Ack=3032553175, Win=49168 (scale factor 0x8) = 12587008
67	11:18:25 AM 2/8/2023	6.2819590	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378389, Ack=3032553175, Win=49168 (scale factor 0x8) = 12587008
68	11:18:25 AM 2/8/2023	6.2833470	10.7.0.6	10.7.0.7	TDS	TDS:Data, Version = Undefined TDS version(0x74000004), Reassembled Packet
69	11:18:25 AM 2/8/2023	6.2833580	10.7.0.7	10.7.0.4	TDS	TDS:Data, Version = Undefined TDS version(0x74000004), Reassembled Packet
70	11:18:25 AM 2/8/2023	6.2851480	10.7.0.4	10.7.0.7	TDS	TDS:Response, Version = Undefined TDS version(0x74000004), SPID = 65, PacketID = 1, Flags=...AP..., SrcPort=1433, DstPort=1025, PayloadLen=440, Seq=303255...
71	11:18:25 AM 2/8/2023	6.2851550	10.7.0.7	10.7.0.6	TDS	TDS:Response, Version = Undefined TDS version(0x74000004), SPID = 65, PacketID = 1, Flags=...AP..., SrcPort=1433, DstPort=1025, PayloadLen=440, Seq=303255...
72	11:18:25 AM 2/8/2023	6.2973490	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378819, Ack=3032553615, Win=49173 (scale factor 0x8) = 12588288
73	11:18:25 AM 2/8/2023	6.2973620	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378819, Ack=3032553615, Win=49173 (scale factor 0x8) = 12588288
74	11:18:25 AM 2/8/2023	6.3502110	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...F, SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378819, Ack=3032553615, Win=49173 (scale factor 0x8) = 12588288
75	11:18:25 AM 2/8/2023	6.3502350	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...F, SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378819, Ack=3032553615, Win=49173 (scale factor 0x8) = 12588288
76	11:18:25 AM 2/8/2023	6.3509780	10.7.0.4	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032553615, Ack=87378820, Win=16383 (scale factor 0x8) = 4194048
77	11:18:25 AM 2/8/2023	6.3509780	10.7.0.4	10.7.0.7	TCP	TCP:Flags=...A...F, SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032553615, Ack=87378820, Win=16383 (scale factor 0x8) = 4194048
78	11:18:25 AM 2/8/2023	6.3509870	10.7.0.7	10.7.0.6	TCP	TCP:Flags=...A...., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032553615, Ack=87378820, Win=16383 (scale factor 0x8) = 4194048
79	11:18:25 AM 2/8/2023	6.3509900	10.7.0.7	10.7.0.6	TCP	TCP:Flags=...A...F, SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=3032553615, Ack=87378820, Win=16383 (scale factor 0x8) = 4194048
80	11:18:25 AM 2/8/2023	6.3513950	10.7.0.6	10.7.0.7	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378820, Ack=3032553616, Win=49173 (scale factor 0x8) = 12588288
81	11:18:25 AM 2/8/2023	6.3513980	10.7.0.7	10.7.0.4	TCP	TCP:Flags=...A...., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=87378820, Ack=3032553616, Win=49173 (scale factor 0x8) = 12588288

- 10.7.0.6 - private link service
- 10.7.0.7 - Linux forwarding VM
- 10.7.0.4 - SQL Server on Azure VM, data source

You could see Linux forwarding VM (10.7.0.7) is forwarding every packet it receives from PLS to SQL Server, and same for the other way, forwarding every packet it receives from SQL Server to PLS.

Example #2 : network trace for failing scenario

Time Date Local Adjusted	Time Offset	Source	Destination	Protocol Name	Description
18:08:08 2023/2/11	7.0190500	172.17.4.4	yofeBEServer.jngwhn0h4oi...	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=2637469589, Ack=0, Win=64860 (Negotiating sca...
18:08:08 2023/2/11	7.0190910	yofeBEServer.jngwhn0h4oiet...	172.17.4.4	TCP	TCP:Flags=...A.R.., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=0, Ack=2637469590, Win=0
18:08:09 2023/2/11	7.7444340	172.17.4.4	yofeBEServer.jngwhn0h4oi...	TCP	TCP:Flags=.....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=2637469589, Ack=0, Win=64860 (Negotiating scal...
18:08:09 2023/2/11	7.7444750	yofeBEServer.jngwhn0h4oiet...	172.17.4.4	TCP	TCP:Flags=...A.R.., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=0, Ack=2637469590, Win=0
18:08:10 2023/2/11	8.4782970	172.17.4.4	yofeBEServer.jngwhn0h4oi...	TCP	TCP:Flags=.....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=2637469589, Ack=0, Win=64860 (Negotiating scal...
18:08:10 2023/2/11	8.4783380	yofeBEServer.jngwhn0h4oiet...	172.17.4.4	TCP	TCP:Flags=...A.R.., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=0, Ack=2637469590, Win=0
18:08:31 2023/2/11	29.8334900	172.17.4.4	yofeBEServer.jngwhn0h4oi...	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1659013568, Ack=0, Win=64860 (Negotiating sca...
18:08:31 2023/2/11	29.8335300	yofeBEServer.jngwhn0h4oiet...	172.17.4.4	TCP	TCP:Flags=...A.R.., SrcPort=1433, DstPort=1025, PayloadLen=0, Seq=0, Ack=1659013569, Win=0

- 172.17.4.4 - private link service
- yofeBEServer - linux forwarding VM

You could see for every packet PLS sent to forwarding VM, forwarding VM directly reset it. No traffic forwarding is in place. So this indicates the forwarding rule might not be setup correctly. You can run above mentioned commands to check **SNAT and DNAT rules**.

One known issue is that Ubuntu system does not store iptable rules persistently and every time after forwarding VM reboot, you have to **rerun the iptables configuration scripts**. Please check this TSG for details.

Example #3 : network trace for failing scenario

Time Date Local Adjusted	Time Offset	Source	Destination	Protocol Name	Description
15:06:00 2023/2/10	6.3850450	172.17.4.4	172.17.2.4	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013549, Ack=0, Win=64860 (Negotiating sca...
15:06:00 2023/2/10	6.3851120	172.17.2.4	172.17.0.4	TCP	TCP:Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013549, Ack=0, Win=64860 (Negotiating sca...
15:06:03 2023/2/10	9.3911550	172.17.4.4	172.17.2.4	TCP	TCP:[SynReTransmit #37]Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013549, Ack=0, Win=...
15:06:03 2023/2/10	9.3912040	172.17.2.4	172.17.0.4	TCP	TCP:[SynReTransmit #38]Flags=CE....S., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013549, Ack=0, Win=...
15:06:08 2023/2/10	14.8822370	172.17.4.4	172.17.2.4	TCP	TCP:Flags=...A.R.., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013550, Ack=0, Win=0 (scale factor 0x0) = 0
15:06:08 2023/2/10	14.8823010	172.17.2.4	172.17.0.4	TCP	TCP:Flags=...A.R.., SrcPort=1025, DstPort=1433, PayloadLen=0, Seq=1666013550, Ack=0, Win=0 (scale factor 0x0) = 0

- 172.17.4.4 - private link service
- 172.17.2.4 - forwarding VM
- 172.17.0.4 - SQL Server data source

For this one, you could see the traffic forwarding is working fine however, when SQL Server received the TCP SYN frame, SQL Server did not respond and client has to retransmit SYN. This indicate something between forwarding VM and SQL Server, or SQL Server itself could block the traffic. The next action plan could be:

- on forwarding VM: run **telnet <FQDN_of_SQL_Server> 1433** to check whether 1433 port is reachable
- on SQL Server: 1) run **netstat -anob** to verify SQL Server is listening on port 1433; 2) verify Windows firewall is not blocking SQL traffic via port 1433.
- collect information about how SQL Server VM (on premise) connects to forwarding VNet (Azure), usually it will use either ExpressRoute or VPN gateway. Please check with customer and open collaboration with SAP **Azure/Virtual Network/VPN Gateway** or **Azure/Virtual Network/ExpressRoute**.

Short Summary

To summarize, tcpdump trace is helpful to dive deeply into the network blockage. So please collect it and do some initial investigation to see if it matches with any of the patterns in above examples. It will also make it easier when you need to further collaborate with Azure Network team.

Other Notes

When using this solution to connect to **Azure SQL Managed Instance**, we have a known issue that "Redirect" connection policy on SQL MI is not supported, you need to switch to "**Proxy**" mode.

On the other hand, private endpoint for SQL MI is going to enter GA soon so we are likely to use first party managed private endpoint to connect SQL MI. We may not suggest above solution for SQL MI in the future and move this public doc to deprecation.