nitred / **optimal_mtu.md**

Last active 2 days ago

Wireguard Optimal MTU

optimal_mtu.md

About

- I faced bandwidth issues between a WG Peer and a WG server. Download bandwidth when downloading from WG Server to WG peer was reduced significantly and upload bandwidth was practically non existent.
- I found a few reddit posts that said that we need to choose the right MTU. So I wrote a script to find an optimal MTU.
- Ideally I would have liked to have run all possible MTU configurations for both WG Server and WG Peer but for simplicity I choose to fix the WG Server to the original 1420 MTU and tried all MTUs from 1280 to 1500 for the WG Peer.

Testing

- On WG server, I started an `iperf3` server
- On WG peer, I wrote a script that does the following:
  - `wg-quick down wg0`
  - Edit MTU in the `/etc/wireguard/wg0.conf` file
  - `wg-quick up wg0`
  - `iperf3 -c 172.123.0.1 -J -t 5 -i 5`
    - This runs an `iperf3` client that connects to `172.123.0.1` which is the WG Server gateway
    - The `iperf3` client runs for 5 seconds and stops and dumps a JSON output

Setup

- Max bandwidth provided by my ISP (1000Mbps Download, 50Mbps Upload)

- WG Server is a VPS running Ubuntu 20.04 on a cloud provider.
- WG Peer is a PC running Ubuntu 20.04 locally at home.

Config

I followed this tutorial to setup my Wireguard configurations.

WG-server

```
# /etc/wireguard/wg0.conf
[Interface]
Address = 172.123.0.1/24
MTU = 1420
SaveConfig = true
PostUp = iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A
POSTROUTING -o eth0 -j MASQUERADE; ip6tables -A FORWARD -i wg0 -j ACCEPT;
ip6tables -t nat -A POSTROUTING -o eth0 -j MASQUERADE; iptables -t nat -A
POSTROUTING -o ens10 -j MASQUERADE; ip6tables -t nat -A POSTROUTING -o
ens10 -j MASQUERADE
PostDown = iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D
POSTROUTING -o eth0 -j MASQUERADE; ip6tables -D FORWARD -i wg0 -j ACCEPT;
ip6tables -t nat -D POSTROUTING -o eth0 -j MASQUERADE; iptables -t nat -D
POSTROUTING -o ens10 -j MASQUERADE; ip6tables -t nat -D POSTROUTING -o
ens10 -j MASQUERADE
ListenPort = 51820
PrivateKey = xxxxxxxxxxxxxxxxx

[Peer]
PublicKey = xxxxxxxxxxxxxxxxx
AllowedIPs = 172.123.0.2/32
Endpoint = X.X.X.X:61426
```

WG-peer

```
# /etc/wireguard/wg0.conf
[Interface]
PrivateKey = xxxxxxxxxxxxxxxxx
ListenPort = 51820
Address = 172.123.0.2/24
MTU = 1384

[Peer]
PublicKey = xxxxxxxxxxxxxxxxx
AllowedIPs = 172.123.0.0/24, 10.1.0.0/24
```

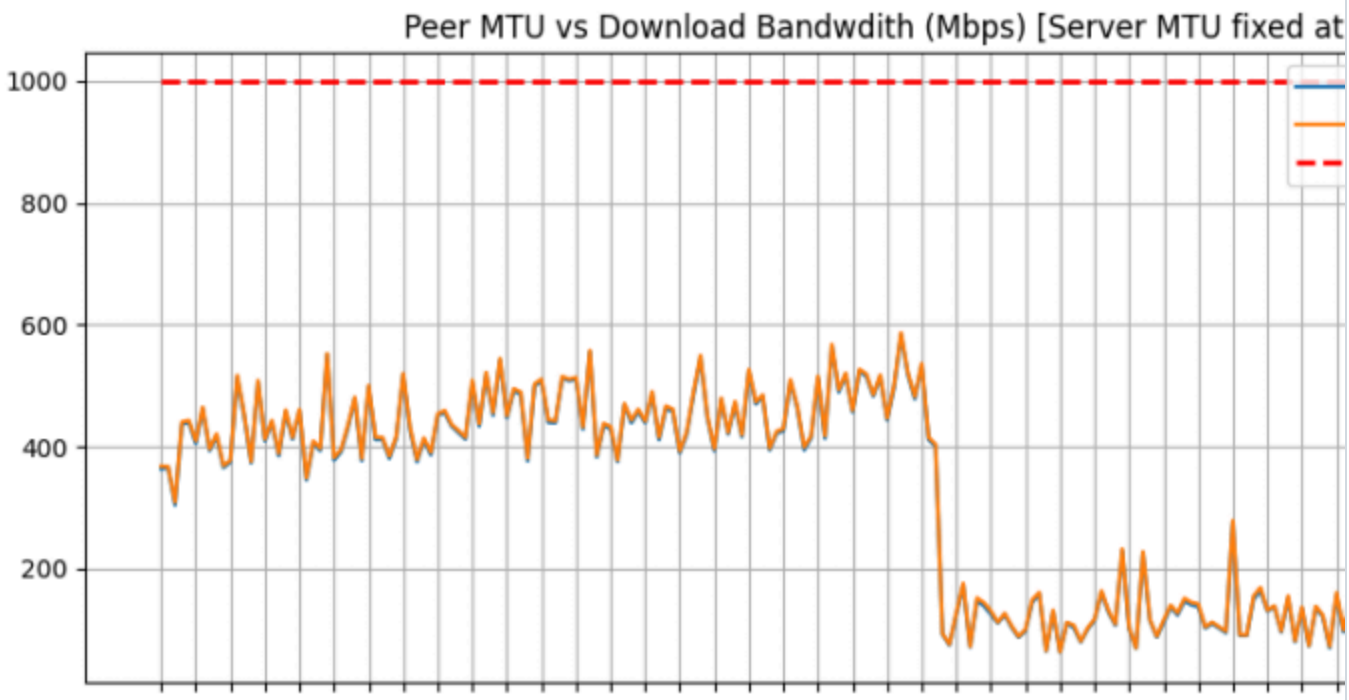
```
Endpoint = Y.Y.Y.Y:51820
PersistentKeepalive = 5
```

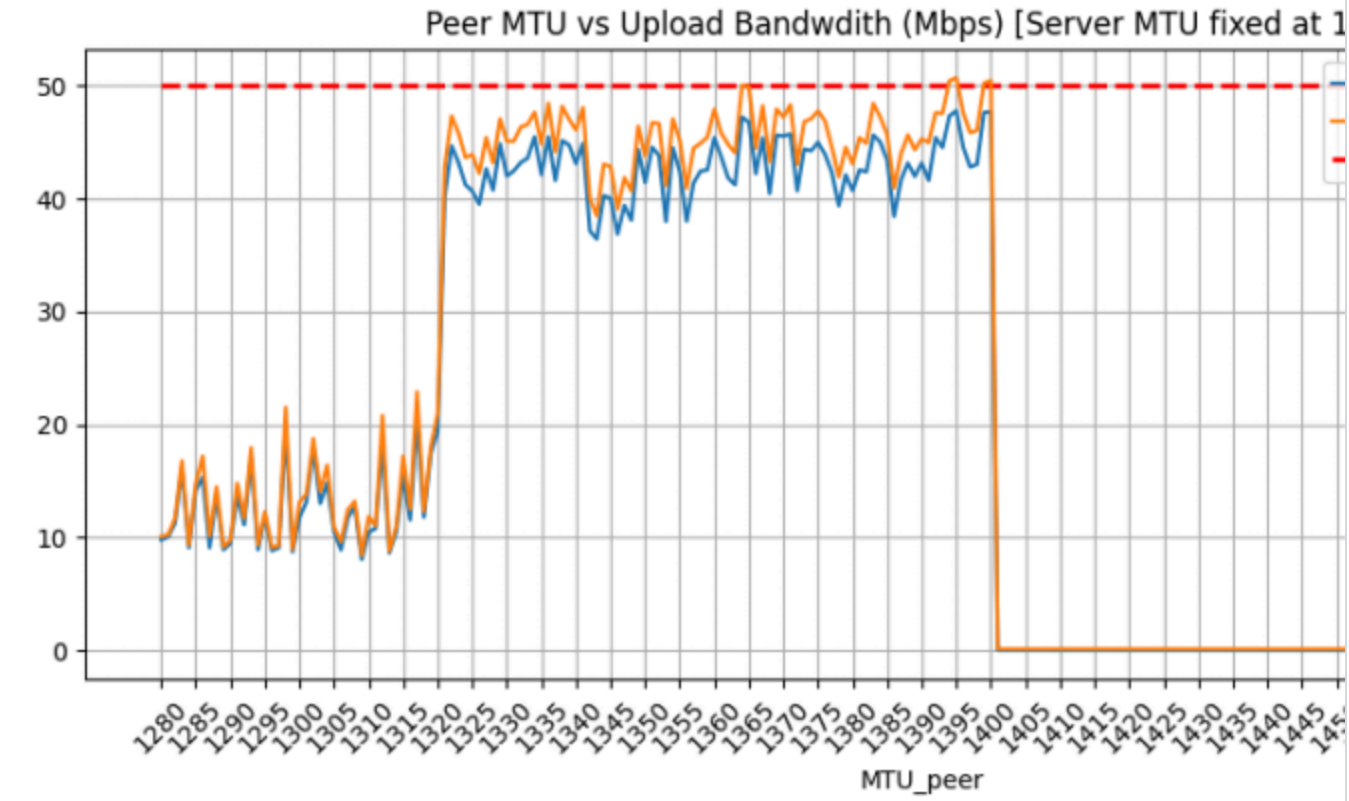
Conclusions

- As you can see in the image, the original MTU setting of 1420 for both peer and server gives abysmal bandwdith.
- I found that that MTU 1384 on the WG peer with 1420 on the WG server seems to almost have the best bandwidth.
- For WG Peer MTU 1384, the max upload bandwidth of 50Mbps of my ISP connection is achieved but I was only able to hit 550 Mbps for the download bandwidth where the max download bandwidth of my ISP connection is 1000 Mbps. This reduction in download bandwidth might be due to other factors but 550 Mbps was sufficient for my use cases so I stopped testing it further.

In case any one has any explanations for this behavior or have found some mistakes in my configurations or tests, please let me know.

peer_mtu_vs_bandwidth.png





wireguard_peer_mtu.csv

Search this file…

MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
1420	1280	9.779	10.085	363.282	367.822
1420	1281	10.037	10.221	365.3	367.071
1420	1282	11.266	11.709	304.065	309.068
1420	1283	16.128	16.773	436.708	440.421
1420	1284	9.085	9.32	439.456	442.93
1420	1285	14.175	14.784	405.159	409.575
1420	1286	15.324	17.204	461.383	464.782
1420	1287	9.089	10.135	393.274	396.265
1420	1288	13.843	14.456	417.028	420.83
1420	1289	8.903	9.154	365.598	368.779
1420	1290	9.465	9.741	374.178	377.272
1420	1291	13.716	14.789	513.172	516.836
1420	1292	11.127	11.686	449.897	452.991
1420	1293	17.062	17.937	373.172	375.754
1420	1294	8.913	9.336	506.806	508.484
1420	1295	11.947	12.288	409.137	412.493
1420	1296	8.825	9.102	439.877	442.88
1420	1297	9.073	9.256	384.711	388.869
1420	1298	19.563	21.517	456.428	460.047
1420	1299	8.732	8.942	412.162	415.124
1420	1300	11.78	13.12	458.38	460.836
1420	1301	13.143	13.835	345.283	348.746
1420	1302	17.922	18.753	405.418	409.175
1420	1303	13.009	14.061	393.397	397.085
1420	1304	14.858	16.404	550.447	552.679
1420	1305	10.502	10.884	377.632	381.924
1420	1306	8.892	9.62	391.268	394.983
1420	1307	11.758	12.459	430.463	434.699
1420	1308	12.74	13.193	478.767	481.101
1420	1309	8.037	8.403	376.72	380.583

MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
1420	1310	10.473	11.855	497.612	500.3
1420	1311	10.782	11.032	411.316	415.38
1420	1312	18.371	20.796	411.914	415.025
1420	1313	8.627	8.813	380.383	384.415
1420	1314	10.49	10.996	413.854	416.949
1420	1315	15.525	17.196	517.402	519.832
1420	1316	11.548	12.51	426.626	431.252
1420	1317	20.41	22.89	374.859	378.885
1420	1318	11.807	12.317	410.784	414.048
1420	1319	17.487	18.05	386.323	390.693
1420	1320	19.413	21.036	451.24	453.66
1420	1321	40.135	42.82	455.563	459.274
1420	1322	44.686	47.313	433.582	436.076
1420	1323	43.2	45.824	422.772	426.286
1420	1324	41.241	43.648	412.187	415.485
1420	1325	40.672	43.91	504.611	509.149
1420	1326	39.525	42.257	433.56	437.778
1420	1327	42.659	45.424	518.56	521.709
1420	1328	40.767	43.171	451.313	455.999
1420	1329	44.862	47.07	541.91	544.553
1420	1330	42.012	45.087	446.963	450.593
1420	1331	42.473	45.097	491.676	495.083
1420	1332	43.203	46.347	485.328	489.416
1420	1333	43.597	46.58	376.498	381.03
1420	1334	45.479	47.653	499.392	502.72
1420	1335	42.135	44.841	507.46	510.794
1420	1336	45.487	48.443	439.534	443.768
1420	1337	41.627	44.141	438.988	442.439
1420	1338	45.14	48.169	512.471	515.087
1420	1339	44.749	47.02	507.913	510.595

MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
1420	1340	43.1	46.089	510.18	513.517
1420	1341	44.901	48.07	428.675	432.339
1420	1342	37.157	39.991	554.82	557.78
1420	1343	36.424	38.485	382.8	386.083
1420	1344	40.27	43.063	434.334	438.187
1420	1345	40.031	42.848	428.99	432.84
1420	1346	36.87	39.092	375.337	378.224
1420	1347	39.407	41.866	468.454	470.649
1420	1348	38.104	40.695	438.86	443.512
1420	1349	44.343	46.408	457.244	461.109
1420	1350	41.43	43.773	439.033	442.085
1420	1351	44.566	46.729	486.283	490.222
1420	1352	43.826	46.653	411.147	415.675
1420	1353	37.991	41.169	462.502	466.677
1420	1354	44.502	47.079	457.439	461.368
1420	1355	42.396	45.164	389.4	392.718
1420	1356	38.012	40.918	419.988	423.808
1420	1357	41.4	44.477	485.799	490.087
1420	1358	42.438	44.904	545.914	549.605
1420	1359	42.562	45.459	447.518	450.806
1420	1360	45.411	47.928	393.015	396.515
1420	1361	43.693	45.798	476.769	479.518
1420	1362	41.835	44.81	420.56	423.304
1420	1363	41.241	44.067	469.51	474.006
1420	1364	47.234	50.056	416.02	419.452
1420	1365	46.766	49.922	522.7	526.512
1420	1366	42.228	44.471	469.826	473.419
1420	1367	45.386	48.233	481.835	484.83
1420	1368	40.473	43.29	394.326	397.456
1420	1369	45.624	47.933	421.451	424.758

	MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
92	1420	1370	45.554	47.232	426.004	429.556
93	1420	1371	45.72	48.316	505.968	509.855
94	1420	1372	40.715	43.042	463.434	466.39
95	1420	1373	44.388	46.817	394.342	399.014
96	1420	1374	44.234	47.081	414.499	417.601
97	1420	1375	44.987	47.772	511.849	515.524
98	1420	1376	43.949	46.898	413.523	417.787
99	1420	1377	42.328	44.548	564.274	567.8
100	1420	1378	39.379	41.955	489.432	493.637
101	1420	1379	42.114	44.552	517.783	520.464
102	1420	1380	40.725	43.098	456.699	459.31
103	1420	1381	42.542	45.402	523.891	526.872
104	1420	1382	42.395	44.906	514.792	518.786
105	1420	1383	45.63	48.434	482.853	485.493
106	1420	1384	45.023	47.23	514.259	517.485
107	1420	1385	43.41	45.675	443.337	446.342
108	1420	1386	38.452	40.952	496.755	500.859
109	1420	1387	41.681	43.994	582.79	586.792
110	1420	1388	43.156	45.636	517.781	521.409
111	1420	1389	41.981	44.336	478.165	481.941
112	1420	1390	43.159	45.298	533.484	536.138
113	1420	1391	41.629	44.981	411.492	415.005
114	1420	1392	45.42	47.612	400.268	403.638
115	1420	1393	44.537	47.562	91.884	93.101
116	1420	1394	47.322	50.432	74.137	75.698
117	1420	1395	47.792	50.713	124.13	126.563
118	1420	1396	44.564	47.602	171.668	175.504
119	1420	1397	42.817	45.856	70.946	72.115
120	1420	1398	43.047	46.05	146.641	151.438
121	1420	1399	47.634	50.21	138.423	143.805

MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
1420	1400	47.693	50.47	124.784	130.186
1420	1401	0.002	0.112	110.135	111.815
1420	1402	0.002	0.112	123.79	126.142
1420	1403	0.002	0.112	103.124	105.616
1420	1404	0	0.112	86.977	88.483
1420	1405	0.002	0.112	97.243	99.735
1420	1406	0.002	0.113	144.839	148.922
1420	1407	0.002	0.113	157.802	160.497
1420	1408	0.002	0.113	64.273	65.018
1420	1409	0	0.113	128.295	131.228
1420	1410	0.002	0.113	62.276	63.044
1420	1411	0.002	0.113	109.99	111.325
1420	1412	0.002	0.113	102.435	106.584
1420	1413	0	0.113	79.216	80.371
1420	1414	0.002	0.113	100.332	102.214
1420	1415	0.002	0.113	114.64	116.9
1420	1416	0.002	0.113	160.194	163.28
1420	1417	0.002	0.113	128.546	131.178
1420	1418	0	0.114	106.562	108.683
1420	1419	0.002	0.114	228.267	231.989
1420	1420	0.002	0.114	100.311	102.456
1420	1421	0.002	0.114	68.612	69.437
1420	1422	0	0.114	222.877	227.281
1420	1423	0	0.114	114.47	116.175
1420	1424	0.002	0.114	87.183	88.719
1420	1425	0.002	0.114	111.124	115.439
1420	1426	0.002	0.114	136.311	139.502
1420	1427	0	0.114	123.442	126.936
1420	1428	0	0.114	146.415	151.041
1420	1429	0.002	0.114	140.015	144.193

1	MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
152	1420	1430	0.002	0.114	137.163	141.625
153	1420	1431	0	0.114	101.918	104.045
154	1420	1432	0.002	0.114	109.205	111.651
155	1420	1433	0.002	0.114	102.198	104.472
156	1420	1434	0.002	0.114	94.156	96.912
157	1420	1435	0	0.114	274.923	278.557
158	1420	1436	0	0.114	89.887	91.313
159	1420	1437	0.002	0.114	89.854	90.785
160	1420	1438	0.002	0.114	149.419	154.718
161	1420	1439	0	0.114	162.98	168.52
162	1420	1440	0	0.114	129.754	131.415
163	1420	1441	0.002	0.114	136.865	138.569
164	1420	1442	0.002	0.114	95.295	97.197
165	1420	1443	0.002	0.114	150.988	154.865
166	1420	1444	0	0.114	79.417	81.118
167	1420	1445	0.002	0.114	134.379	136.362
168	1420	1446	0.002	0.114	72.408	73.436
169	1420	1447	0.002	0.114	134.539	137.73
170	1420	1448	0	0.114	121.354	123.645
171	1420	1449	0.002	0.114	69.386	72.186
172	1420	1450	0.002	0.114	155.168	160.252
173	1420	1451	0.002	0.114	95.625	98.829
174	1420	1452	0	0.114	154.402	158.912
175	1420	1453	0.002	0.114	77.676	81.093
176	1420	1454	0	0.114	151.933	155.661
177	1420	1455	0.002	0.114	127.309	132.258
178	1420	1456	0.002	0.114	148.996	153.324
179	1420	1457	0.002	0.114	92.367	95.644
180	1420	1458	0.002	0.114	144.232	148.894
181	1420	1459	0.002	0.114	77.765	80.336

	MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
182	1420	1460	0.002	0.114	133.286	138.711
183	1420	1461	0	0.114	129.757	135.053
184	1420	1462	0.002	0.114	203.826	209.096
185	1420	1463	0.002	0.114	71.752	74.063
186	1420	1464	0.002	0.114	159.222	164.221
187	1420	1465	0	0.114	119.811	124.256
188	1420	1466	0	0.114	105.249	110.749
189	1420	1467	0.002	0.114	60.189	62.376
190	1420	1468	0.002	0.114	213.514	217.745
191	1420	1469	0	0.114	64.601	66.069
192	1420	1470	0.002	0.114	138.102	142.246
193	1420	1471	0	0.114	148.872	152.626
194	1420	1472	0.002	0.114	212.529	216.044
195	1420	1473	0.002	0.114	104.654	108.83
196	1420	1474	0	0.114	137.298	142.16
197	1420	1475	0.002	0.114	93.042	96.909
198	1420	1476	0.002	0.114	150.276	154.835
199	1420	1477	0.002	0.114	80.563	85.272
200	1420	1478	0	0.114	114.828	119.55
201	1420	1479	0.002	0.114	65.511	67.377
202	1420	1480	0.002	0.114	141.206	146.038
203	1420	1481	0.002	0.114	210.533	215.735
204	1420	1482	0.002	0.114	133.742	138.278
205	1420	1483	0	0.114	127.18	130.657
206	1420	1484	0.002	0.114	169.513	174.231
207	1420	1485	0.002	0.114	155.057	158.817
208	1420	1486	0.002	0.114	112.984	117.463
209	1420	1487	0	0.114	122.723	127.679
210	1420	1488	0	0.114	127.9	132.621
211	1420	1489	0.002	0.114	131.65	136.616

1	MTU_server	MTU_peer	upload_rcv_mbps	upload_send_mbps	download_rcv_mbps	download_send_mbps
212	1420	1490	0.002	0.114	60.408	62.315
213	1420	1491	0.002	0.114	107.486	111.7
214	1420	1492	0	0.114	113.105	116.995
215	1420	1493	0.002	0.114	161.789	167.387
216	1420	1494	0.002	0.114	87.811	90.931
217	1420	1495	0.002	0.114	96.189	99.908
218	1420	1496	0	0.114	83.761	86.526
219	1420	1497	0	0.114	80.308	82.488
220	1420	1498	0.002	0.114	124.105	128.196
221	1420	1499	0.002	0.114	120.256	124.763
222	1420	1500	0	0.114	184.46	189.222

probablypablito commented on Mar 10, 2022

thanks!

nitred commented on Mar 16, 2022 Author

FYI, there's better data and plots that can be found on a very related project of mine. The plot above is only for a fixed server MTU of 1420 and different peer MTUs. The plot in the other repo is 2D which covers different values of both server and peer MTUs.

project: nr-wg-mtu-finder
example csv and png folder: examples

skydiablo commented on Mar 18, 2022

@**nitred** thx for input! script is running and will test my WG env...

nitred commented on May 2, 2022 Author

@**skydiablo** Did you happen to finish running the test? I'm quite curious to know if the tool helped and if you were able to generate a plot?

skydiablo commented on May 2, 2022

I'm really unsure to make it sense on a different MTU on Server and client Side. I have calc my MTU by real logical facts.

nitred commented on May 2, 2022 Author

@**skydiablo** That's alright, I was just curious. If you were able to calculate the MTU and it worked for you, that's great!

Harliff commented on Sep 1, 2022

@**nitred** Wow! Great work work!

Have you tried to reproduce your test?

Harliff commented on Sep 1, 2022

Have anyone tried to do this test in controlled environment (e.g. in LAN)?

Harliff commented on Sep 1, 2022

Just tested bandwidth between router at work and cloud VPS server.
I haven't faced any issues using MTU 1420 at both sides.
I believe the issues with MTU larger than 1400 is related to @**nitred**'s ISP.

nitred commented on Sep 6, 2022 Author

> Have you tried to reproduce your test?

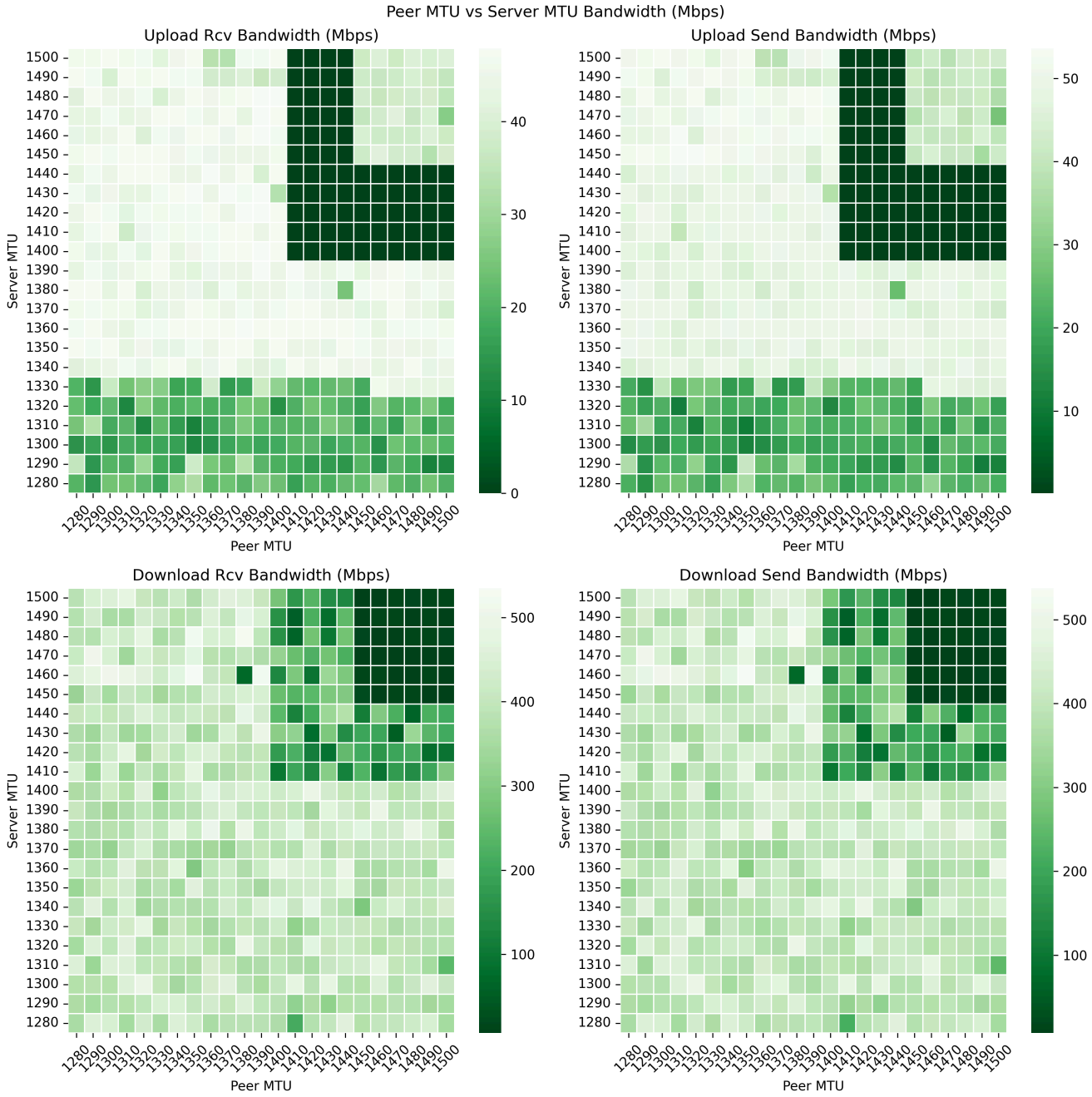
> Have anyone tried to do this test in controlled environment (e.g. in LAN)?

@**Harliff** I was able to get the same results (i.e. cutoff at 1400) when I repeated the test on a new cloud VM which acted as a WG server while my PC acted as WG peer. I also did a test where my Raspberry Pi (in LAN) acted as the WG server while my PC acted as a WG peer and it gave different results to the one above which didn't have any hard drop outs.

> I believe the issues with MTU larger than 1400 is related to @**nitred**'s ISP.

I think you're right that there's an issue somewhere between my router and cloud VPS server and very likely with my ISP.

However the point wasn't to say an MTU of 1420 is bad for everyone, it's just that there's an optimal MTU for every configuration of WG Server-Peer setups. Here's a image with a more extensive test which plots the bandwidths when WG Peer's and WG Server's MTUs are altered. I created a repo which one can use to create such plots - [nitred/nr-wg-mtu-finder]



XenGi commented on Nov 7, 2022

AFAIK it makes no sense to set different MTUs for both ends of a connection. The MTU is basically limited by the headers of your ethernet frames. if you haven't enabled jumbo frames (MTU = 9000) for all your connections this is fixed to 1500. For example, I dropped mine to 1400 because I send IPv6 wireguard headers inside an IPv6 internet connnection. I needed the lower MTU so both fit into each other I could go a little higher if I used IPv4 for both because of the smaller header size of IPv4 frames. You pretty much always want to use the biggest possible MTU to maximaize your throughput and efficiency. It's very easy to calculate that. Just look at your headers.
There are very few factors out of your control that could have an effect on your MTU choice, like if you're connecting via a DSL line you have to make space for the PPPOE overhead or if your connecting through an overlay network like an MPLS cloud.

But for normal everyday connections you know all the components involved or have no control over them anyway. SO just calculate for what you know and use the biggest value possible.

mikalai-t commented on Jan 6, 2023

@**nitred** Great job anyway! I'm not really familiar with all the tricks and calculations over MTUs, but page gave me more understanding how to fix this or that stuff, than reading tons of theoretical articles.

Svaag commented on Feb 11, 2023 • edited

I would argue that the whole approach of trying to "tune" MTU is wrong to start with and any guide that you can find, even a good one, is both hard to know if it's applicable to your own circumstances and very prone to breakage as any network change can completely invalidate your configuration.
Not to mention that you (at most) only control your own half of the pipe and you will be communicating with a host that has it's own whole set of unknown and variable circumstances.

What you instead should try to do is make sure that the built in mechanism for auto-discovering MTU (Path MTU Discovery, or PMTUD) is enabled and functioning. Mainly this simply involves allowing ICMP traffic through your firewall and letting it take care of controlling your traffic. The functionality is even in the name, Internet Control Message Protocol.

However because of a long history of misinformation, and a false sense of security, most people are actively blocking all ICMP in their firewalls which can actually do quite a lot of damage to the quality of connections.

The other thing is to clamp MSS for TCP connections to PMTU, look up the --clamp-mss-to-pmtu option in iptables. This helps correct misbehaving clients however if PTMUD is truly broken you will still experience issues. And of course in the best case this is still only valid for TCP.

So my general advice for any budding network operator would be to do the following:

1. Allow ICMP and ICMPv6 in firewalls to allow PMTUD to work. If you desperately want to "hide", block only ICMP Echo/Reply types (ping). This is VERY important and if you only choose to do one thing, do this one.
2. Clamp MSS to PTMU in firewall.

3. Stop trying to micro-optimize your connection for performance,as it is extremely unlikely you are limited by Packets Per Second, which is what you are really tuning for by changing MTU. Only tinker with defaults if you are actually having problems.
4. If you are having problems, set a reasonably low value without being too extreme. I'd even recommend choosing the same value for IPv4 and IPv6, despite the header differences. IPv6 has 1280 default MTU, which is a MSS of 1260. Go with that for both protocols. You are extremely unlikely to ever notice the slightest difference in performance in a small network. However "MTU blackholing" should largely be a thing of the past so, again, it is very unlikely you will need to touch your MTU beyond defaults.

This is a very good article for anyone who finds this discussion interesting:
https://blog.apnic.net/2019/07/31/tcp-mss-values-whats-changed/
It suggests that staying

chayleaf commented on Aug 6, 2023

> block only ICMP Echo/Reply types

Ping is the part of ICMP that doesn't need blocking. There are much more dangerous features of ICMP though, such as type 5 (redirect).

y0nei commented on Sep 27, 2023

I was getting ~35 Mbits/sec inside of my tunnel with having a MTU of 1500 both on client and server (chosen at random), after testing your values (i know everyone should find out what their own are) im getting ~240 Mbits/sec
I had no idea this little setting changes so much, thank you!

nitred commented on Sep 28, 2023 Author

@**y0nei** I'm glad it helped! I know what you mean, it's a single parameter that I was able to tweak and gain significant bandwidth improvement. In my case it went from ~0 Mbps to ~50Mbps. Then I was able to move on to what I really was using the VPN for.

pirmanji commented on Sep 30, 2023

Wow, I'm sitting 3000km from home right now with a very poor internet connection, and after reading this thread, I just reduced the MTU on both sides of my WireGuard connection to 1400 (as my understanding is that this is the size WireGuard will give the packets that go through the tunnel, right?). And now, after doing so, my VPN has become workable and feels quite fast, whereas it wasn't usable at all before.

So my understanding is that the VPN MTU should be below the MTU of the underlying real network connection so that the VPN packets "fit" into the network packets and won't be split up, which would cause a reduction in usable bandwidth. Is that correct?

Svaag commented on Oct 1, 2023 • edited

> block only ICMP Echo/Reply types
>
> Ping is the part of ICMP that doesn't need blocking. There are much more dangerous features of ICMP though, such as type 5 (redirect).

Yeah I would personally never block Echo/Reply because it makes troubleshooting a big pain in the ass. But that sentence was more a comment on IF you want to block ICMP to "hide", only block (or rather, drop, so your computer wont reply at all) those two types and not all of ICMP (which is was most people do). Blocking ICMP entirely will break your connection in various ways.

As for redirects, most operating systems nowadays don't really accept them:
[svag@tuxxx] sysctl net.ipv4.conf.all.accept_redirects
net.ipv4.conf.all.accept_redirects = 0
[svag@tuxxx] sysctl net.ipv4.conf.default.accept_redirects
net.ipv4.conf.default.accept_redirects = 0

But blocking them in firewall anyway is good for a layered approach to security.

> So my understanding is that the VPN MTU should be below the MTU of the underlying real network connection so that the VPN packets "fit" into the network packets and won't be split up, which would cause a reduction in usable bandwidth. Is that correct?

When MTU is not aligned to the actual MTU of the link, then every connection you make has to rely on Path MTU Discovery to work. But PMTUD is often broken due to firewalls and middleboxes. That results in packets sized above the actual MTU not reaching their target at all thus creating packetloss - especially with UDP; which is very noticable with DNS traffic and of course also with gaming.

localhost commented on Oct 2, 2023

> block only ICMP Echo/Reply types
>
> Ping is the part of ICMP that doesn't need blocking. There are much more dangerous features of ICMP though, such as type 5 (redirect).

Asuka, ping is primarily a utility...

mattpr commented on Nov 2, 2023 • edited

Wireguard does not default to MTU 1500. It defaults to `1500 - 80` ...but only if all other attempts to detect your connection MTU fail. I'm talking about `wg-quick` helper script here. MTU config setting doesn't mean anything to `wg` (which expects you to manage ip/link/route/addr stuff yourself).

This is because wireguard has 80 byte overhead for ipv6 and 60 byte overhead for ipv4. You may also find this page helpful.

Relevant code is here.

Here is the MTU setting logic from `wg-quick`:

- if an mtu is set in the config file, the link is brought up with that mtu and we are done. 80 is NOT subtracted from the value.**
- otherwise, we move on to "auto detection" of MTU
- wg-quick grabs all endpoints (IPs of peers) from the wireguard interface (using wg0 as example)
  - `wg show wg0 endpoints`
- attempts to parse mtu **for each endpoint** by running `ip route get <endpoint address>` and looking for `mtu [0-9]+` in output.
  - if no mtu is found in the output, then the device for the route (`dev [^ ]+`) is taken instead and used in `ip link show dev <dev-for-route>` ...and the mtu (`mtu [0-9]+`) is extracted from that output.
- the **largest** of all endpoint mtus found are kept***
- if at the end of checking all wg peer endpoints (routes/devices) we still did not find ANY set `mtu` (on output of `ip route get <addr>` or `ip link show dev <endpoint-dev>` ...then wg-quick tries to get the mtu from `ip route show default`
- if we still have not found any mtu, wireguard defaults to 1500.
- whatever mtu value we have at this point...wg-quick brings up the link with `mtu - 80`.

** This means if your eth0 MTU is 1500 and you explicitly set MTU to 1500 in wg-quick config, 1500 will be used as-is (not 1420)...so you will have problems because your wg packets will end up larger than what is supported on the underlying device/route and so there will be fragmentation (slow because packets have to be broken up).

*** Is this right? If we have a `peer endpoint A` with `1200 MTU` on `eth1` and another `peer endpoint B` with `1500 MTU` on `eth2`, it appears to me that wg-quick keeps the `1500 MTU` from `B` (minus 80 at the end)...meaning packets for peer `A` may be larger than the `1200 MTU` supported on `eth1` causing fragmentation and slow bandwidth to peer `A`? I would think we would want to take the smallest MTU...which reduces bandwidth (smaller packets) for peers that are connected over links/routes that support larger MTU...but avoiding fragmentation on the routes/links that have lower MTU support. Or did I misunderstand something? Maybe small packets are slower than fragmentation?

You can manually repeat the wg-quick MTU detection process to see how it works. But ultimately it is relying on the MTUs the networking stack reports for routes/links to the endpoints. So if these MTUs are wrong or not-detectable...then there is a more general problem that will impact your networking performance for more than just wireguard. I think the suggestions here are good to make sure you are doing what you can to not prevent your networking stack from detecting MTUs correctly.

Of course if your ISP is doing stupid things, then you may need to resort to empirical testing.

Biggest footgun here is setting MTU manually in config rather than letting wg-quick do it's auto-detection. You can also let wg-quick detect/set the mtu and then look at what MTU was set by wg-quick (`ip link show dev wg0`) and then further reduce that to see if it helps. The MTU tradeoff is: bigger packets mean higher throughput...until it doesn't: if your packets are too big for the link/route then you have fragmentation and things get slow. If your packets are too small then it is also slow. Ideal MTU (largest packet without fragmentation) is: actual supported MTU by the route/device minus wg overhead. You can use `mtu - 60` for instance if you know you will only use ipv4.

If you have a mix of MTUs to your various peers...I would expect that choosing the SMALLEST would be the best compromise because I would expect throughput decrease of smaller packets to be less of a slowdown than fragmentation of too-large packets. But I'm not an expert. Interested for feedback on this point.

jduan00 commented on Jan 18 • edited

@**nitred** : thank you so much! It is very helpful. Just an observation. It seems that you don't need to put MTU in the `wg0.conf` for server, as it is automatically done. Per @**mattpr** .

```
# ip a
...
3: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default
qlen 1000
```

Only on the client they should use a smaller MTU size via wg conf.

Thanks to @**Svaag** , I am adding this to the iptables firewall on the server, and the client's test confirmed it worked without making the conf mod for MTU size.

```
iptables -t mangle -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss
1384
```

It seems for some clients, 1400 (MTU 1420-20) is not good enough, thus I am not using: `--clamp-mss-to-pmtu`

Ref: https://blog.vitlabuda.cz/2022/10/15/clamping-tcp-mss-using-iptables.html

KoloKush commented last month

Thank you. This discussion has helped me out a lot.

ShipkaChalk commented 3 weeks ago

Just an FYI, you can save a step, according to this: https://serverfault.com/questions/1116070/very-high-probably-mtu-being-set-automatically-on-wireguard-interface

You can just run:

`ip link set wg0 mtu 1420` as `root` on the active connection, without having to bring it up / down for your testing.

adamk commented 2 weeks ago

How about running `sudo iptables -t mangle -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu` ?