Matter Large Network Performance

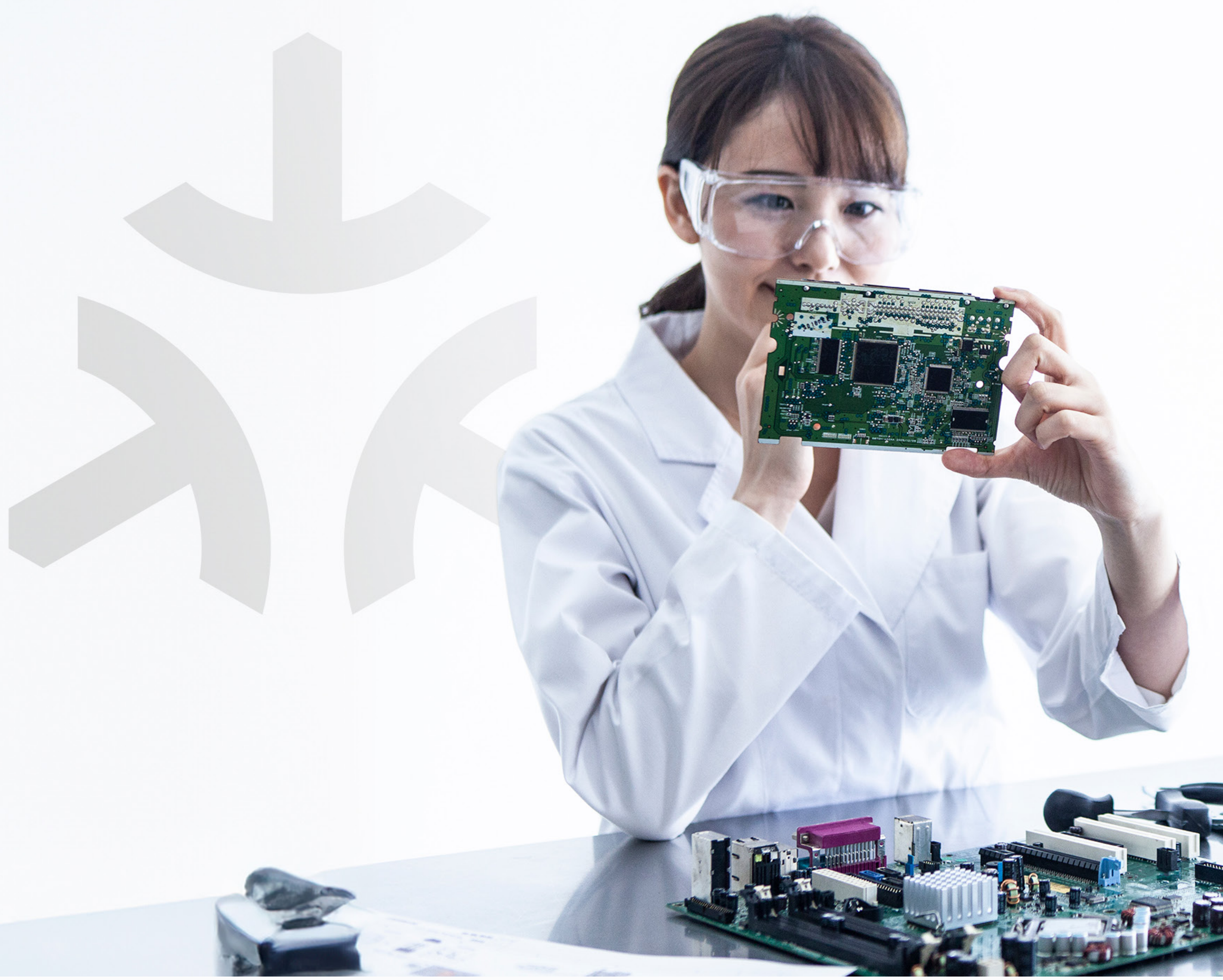
Version	Date	Changes
1.0	11-7-2025	Initial Version

Table of Contents

Overview

Matter is a popular, emerging technology for the home IoT. It is based on common communications transport such as IP, Wi-Fi, Thread, and BLE. Thread in particular is designed to be scalable to large networks that can be more common in commercial environments. As Matter's use grows we anticipate many vendors will expand their use of Matter Thread devices and deploy these in larger and larger installations. With that in mind Silicon Labs sought to analyze its viability and robustness.

This whitepaper discusses Silicon Labs' work on creating and measuring the performance of large networks of Matter Thread devices.



Summary

Silicon Labs conducted a series of tests to evaluate the performance and reliability of large Matter-over-Thread networks. The main intent was to construct a lighting network for a large building. The key findings from these tests are summarized below:

- **Scalability and Stability:** A network comprising 200 nodes was successfully created and operated without any significant issues.
- **Multicast Command Performance:** Matter commands were multicast to all nodes in the network and were processed with acceptable latency. Large lighting networks traditionally uses multicasts, which are network wide broadcasts at the Thread IEEE 802.15.4 radio layer, in order to control sets of nodes in rooms or other logical groupings.
- **Commissioning Consistency:** The process of on-network commissioning for a 200-node setup showed consistent and reliable results across multiple test iterations. Given the size of a commercial building network and the associated installation, it may be common for administrators to add, replace, or reconfigure nodes on a more frequent basis than a home network and thus trigger a commissioning event.
- **Latency Observations:** As anticipated, the latency for receiving a Matter command sent to a multicast group address was higher than that of an ICMP multicast transmitted directly over a Thread network. This aligns with expectations due to the additional processing overhead associated with Matter protocol handling.

In addition, a test of unicast performance was carried out in a controlled environment designed to emulate multi-hop forwarding within a large-scale Matter-over-Thread network. The test measured two key performance metrics: the latency of CASE (Certificate Authenticated Session Establishment) security session setup and the round-trip time for application-layer command processing. These measurements were taken at various hop distances and across a range of payload sizes. CASE session setup should be an infrequent operation but is required in order to properly process the application layer messaging.

General Test Methodology

The performance evaluation was conducted using the **Performance Testing Application**, a custom-built tool derived from the *Matter - SoC Lighting over Thread example.* This application includes a **custom cluster** that defines a command with a single argument of type **octet_string**. By varying the length of this string the test can simulate different payload sizes to assess performance under various data loads. By itself a Matter cluster command with no arguments takes up approximately 40 bytes of UDP payload.

It should be noted that one of the goals of this testing was to measure latency specifically at the Matter application layer. The Performance Testing Application was implemented to address this use case.

Multicast Test

This test measures the **one-way latency** of receiving a Matter application cluster command sent to a **group address (GroupId)**. It closely mirrors real-world scenarios such as a single light switch controlling multiple lights simultaneously.

We make use of the **Silicon Labs Advanced Connectivity Cluster** that has 6 nodes per cluster that are deployed in the Silicon Labs Boston office throughout the floor. For more details see the "Open-Air Environment" part of the "Test Setup and Topology" section of this document.

Test Setup and Execution

- The command transmission and reception are instrumented using a Silicon Labs **Simplicity SDK utility** which outputs debug information via the **Packet Trace Interface (PTI)**. The **Packet Trace Interface** is a **hardware-level debugging interface** provided by **Silicon Labs** that allows the **capture of live radio traffic** and debug data from a wireless device. Silicon Labs SoCs expose PTI over dedicated GPIOs, a debug adapter (like a **Silicon Labs WSTK**) connects PTI to a PC.
- Each debug print includes a **timestamp** and a **sequence number** embedded in the command payload, enabling precise correlation between the time of transmission and reception.
- Since the PTI output is triggered directly from the **application-layer command handler, the measured latency reflects application-layer performance to within 1 µS accuracy.**
    - This was done to accurately capture the time it takes for a smart light to respond to an on/off message so that the user can see the light change state.
- The Matter network is created at the beginning of the test and stays up throughout a test run.

Hardware and Configuration

Tests were conducted on **BRD4187C** development boards using an EFR32MG24 SOC, all commissioned onto a **Matter fabric over Thread**. See the Test Setup and Topology section for details on commissioning.

- A group is configured on all nodes.
- One node sends the custom cluster command to the group, while the remaining nodes receive it.
- Timestamps are collected from the debug output to calculate latency.
- The test is repeated with varying payload sizes by adjusting the length of the octet_string parameter.

Unicast Test

This test evaluates the **round-trip latency** of a Matter application cluster command and its corresponding response when sent to a **unicast address (NodeId)** across multiple hops in the Thread network. This simulates typical use cases such as sending a command to a single device and the controller changing a state on the UI or a controller reading an attribute on a device and displaying the result.

- The test is run on Silicon Labs **BRD4187C boards** within a Matter-over-Thread network. See the Test Setup and Topology section for details on commissioning.
- The sending node calculates the time delta between sending the command and receiving the response and outputs it to the **Matter Shell**. This measures the latency all the way up through the **application layer**.
- As with the multicast test, the payload size is varied by adjusting the octet_string parameter.
- The latency of the first command sent to a given node is tracked and reported separately since the exchange involves establishing a CASE session.

Test Setup and Topology

Device Configuration

All nodes are configured as Full Thread Devices (FTDs). While it may be common for a network to have a mix of end nodes and routers, we wanted to stress the system's ability to handle more traffic from routers performing typical network maintenance. All Matter and Thread configuration parameters are the defaults used in the Silicon Labs' Matter - SoC Lighting over Thread sample application.

The Matter – SoC Lighting over Thread application is running FreeRTOS and has 13 OS threads for servicing various operations. The Silicon Labs Bluetooth stack is running but is not used during our testing.

Test Parameters

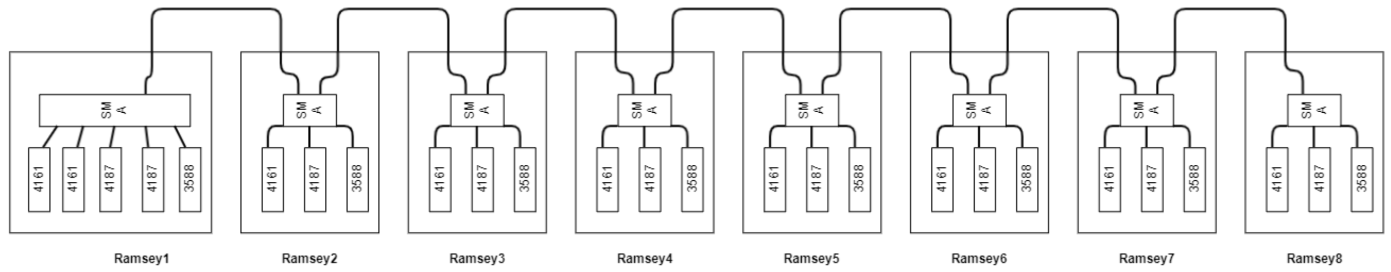
- Network size: 50, 100, 150 or 200 devices
- Packet size: 8, 16, 32 or 64 bytes

Commissioning

Devices first join the Thread network using the network dataset provided by the OTBR (OpenThread Border Router). This is done using the OpenThread CLI. Once connected, the on-network Matter commissioning is used. The commissioning step took 7 seconds per node on average.

Controlled Environment for Unicast Testing

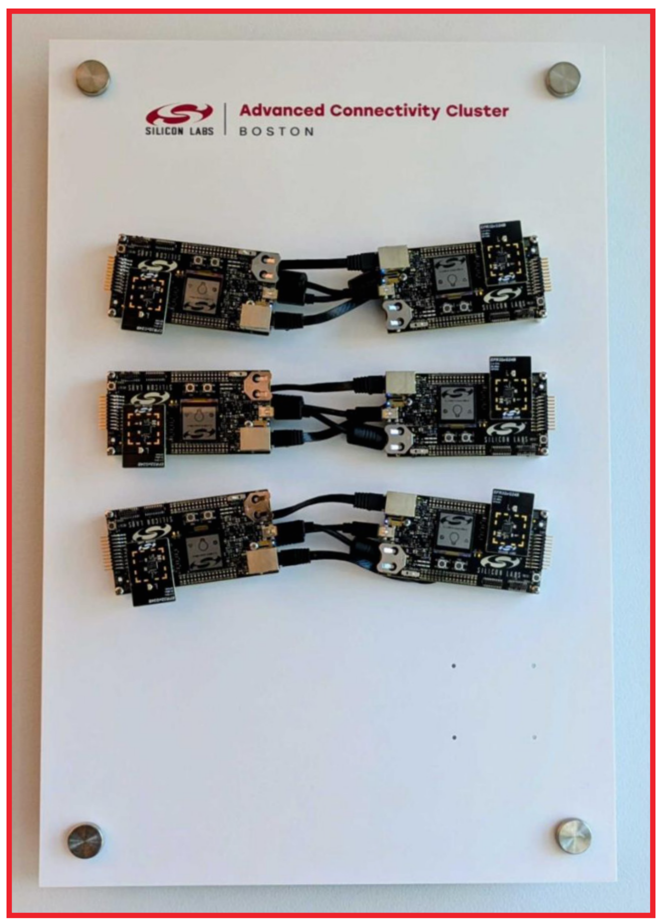
Ramsey isolation boxes (Ramsey Electronics, Model STE330) are conductively connected via wires forming a controlled isolated multi-hop network for precise measurements in unicast testing.



Open-Air Environment for Multicast Testing

Wireless Matter devices are placed across office walls in Silicon Labs' Boston to allow testing of scaled scenarios in order to evaluate performance in the open-air environment. Devices are grouped into 40 clusters, a cluster contains 6 Wireless Started Kits (WSTK) each hosting a BRD4187C. The maximum distance between two clusters is approximately 50 meters. The environment includes interference from a large number of office and test devices running Thread, Wi-Fi and Bluetooth; no measures are taken to isolate the test network from other devices. At our location we typically see 12 or more Wi-Fi networks running. At the same time, no artificial congestion is introduced into the environment.

Connectivity Cluster:



Placement of Connectivity Clusters in the office:

Test Results

Each of the tests was run multiple times across several releases of Silicon Labs Matter and Simplicity SDK. No significant differences were found across multiple runs. See the Software and Hardware section for the software version info.

Multicast Results

The figures below present multicast latency results for different network sizes varying from 50, 100, 150 to 200 nodes. Each histogram illustrates the distribution of latency values (in milliseconds), showing how frequently each value occurs. This analysis is repeated for different packet sizes to highlight performance variations. This test was performed multiple times throughout different Matter releases, and the results were similar. See the Hardware and Software section below on the info of the software release used in the test results reported in this document.

Notes: **The latency value of -10ms represents failure to receive at the application layer including all retry attempts.**

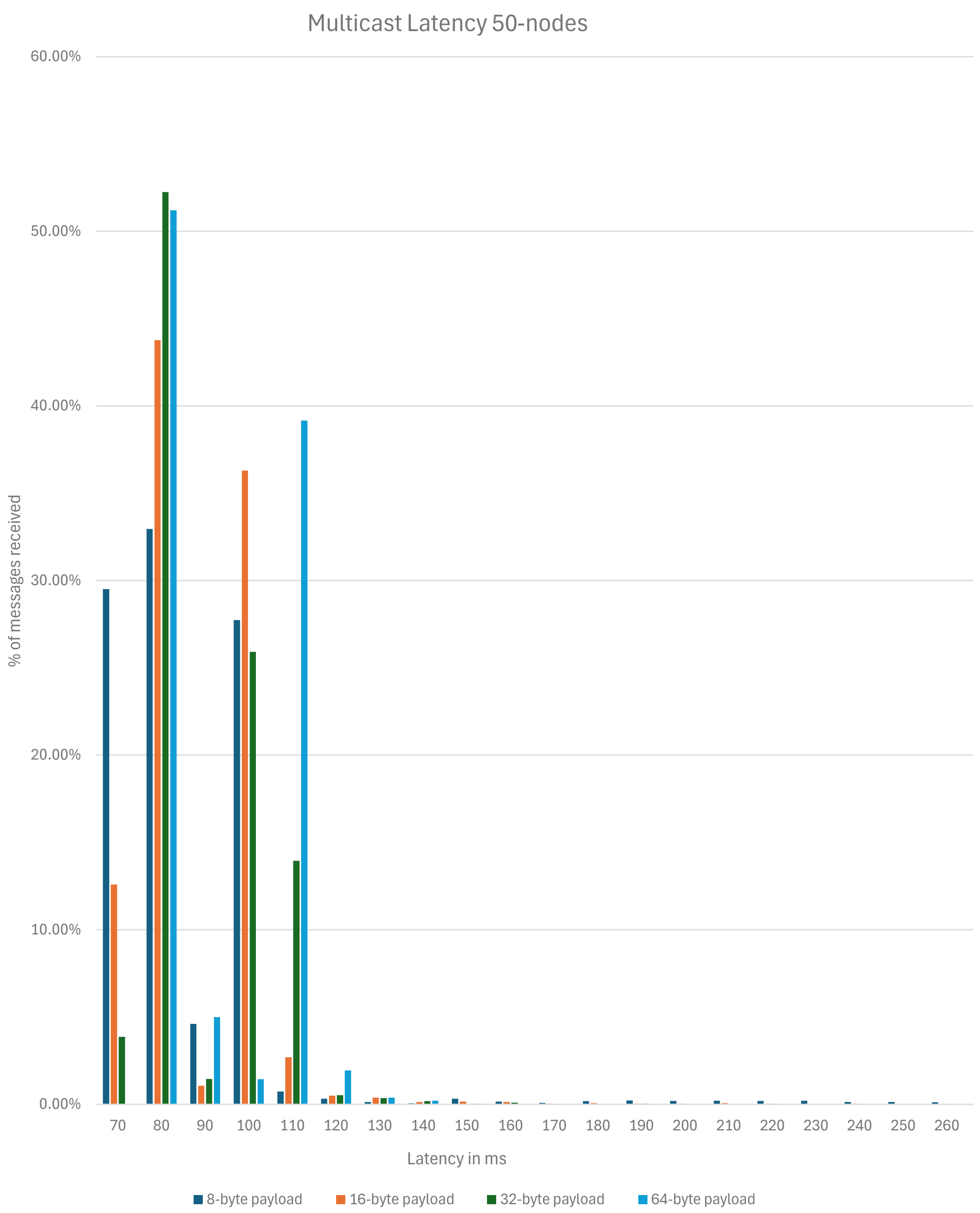
Multicast Results

Summary

Average Latency across all packet sizes: 91ms

The 95th percentile latency (95% of the commands are processed within this time frame): 110ms

Data

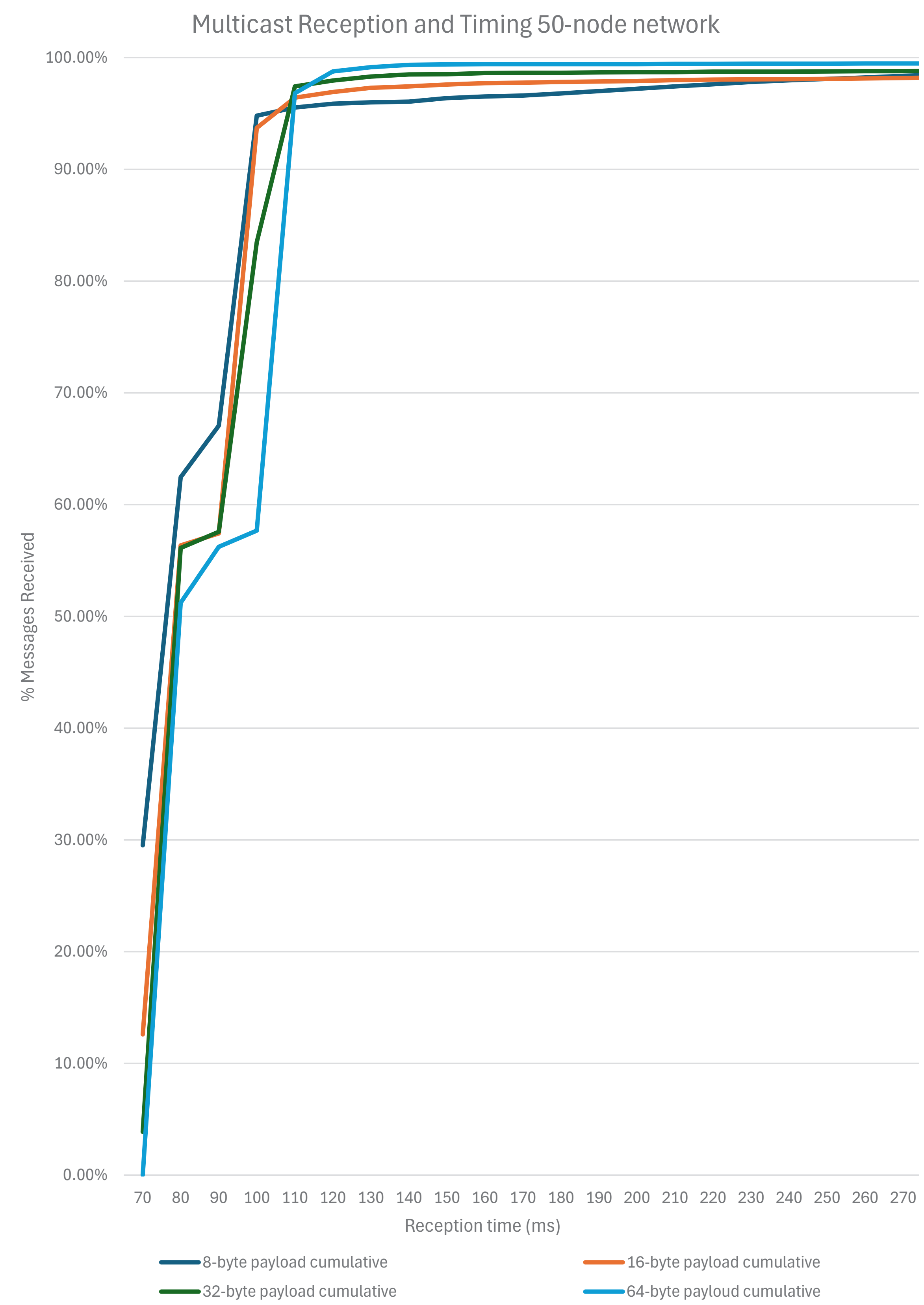


Cumulative Reception

When we aggregate the data across multiple buckets we get the following table of results and the graph below.

50 nodes				
Latency in ms	8-byte payload	16-byte payload	32-byte payload	64-byte payload
70	29.51%	12.59%	3.87%	0.02%
80	62.46%	56.35%	56.12%	51.22%
90	67.07%	57.41%	57.57%	56.22%
100	94.80%	93.71%	83.48%	57.66%
110	95.54%	96.41%	97.43%	96.82%
120	95.86%	96.91%	97.95%	98.77%
130	95.99%	97.29%	98.31%	99.15%
140	96.05%	97.42%	98.49%	99.36%
150	96.37%	97.58%	98.52%	99.40%
160	96.53%	97.71%	98.62%	99.41%
170	96.61%	97.75%	98.63%	99.42%
180	96.79%	97.82%	98.64%	99.42%
190	97.01%	97.87%	98.68%	99.42%
200	97.21%	97.91%	98.69%	99.42%
210	97.42%	97.98%	98.71%	99.43%
220	97.62%	98.02%	98.74%	99.44%
230	97.83%	98.04%	98.75%	99.45%
240	97.97%	98.08%	98.75%	99.46%
250	98.10%	98.10%	98.76%	99.46%
260	98.22%	98.13%	98.78%	99.47%
270	98.36%	98.17%	98.79%	99.48%
280	98.46%	98.21%	98.80%	99.48%
290	98.57%	98.24%	98.81%	99.48%
300	98.67%	98.29%	98.83%	99.49%
310	98.73%	98.34%	98.84%	99.52%
320	98.84%	98.38%	98.85%	99.53%
330	98.91%	98.42%	98.87%	99.53%
340	99.00%	98.44%	98.89%	99.57%
350	99.07%	98.48%	98.91%	99.57%
360	99.13%	98.51%	98.93%	99.57%
370	99.16%	98.55%	98.95%	99.57%
380	99.20%	98.60%	98.97%	99.58%
390	99.25%	98.65%	98.99%	99.58%
400	99.29%	98.68%	99.02%	99.58%
410	99.35%	98.71%	99.03%	99.59%
420	99.41%	98.74%	99.05%	99.60%
430	99.46%	98.77%	99.07%	99.60%
440	99.51%	98.82%	99.10%	99.61%
450	99.54%	98.85%	99.13%	99.62%
460	99.57%	98.89%	99.16%	99.63%
470	99.60%	98.93%	99.18%	99.64%
480	99.63%	98.96%	99.21%	99.65%
490	99.66%	99.00%	99.23%	99.67%
500	99.66%	99.00%	99.23%	99.67%
More	**0.32%**	**1.01%**	**0.73%**	**0.29%**
Mean Latency:	**89.80**	**89.88**	**90.82**	**94.18**

Graph of Cumulative Results



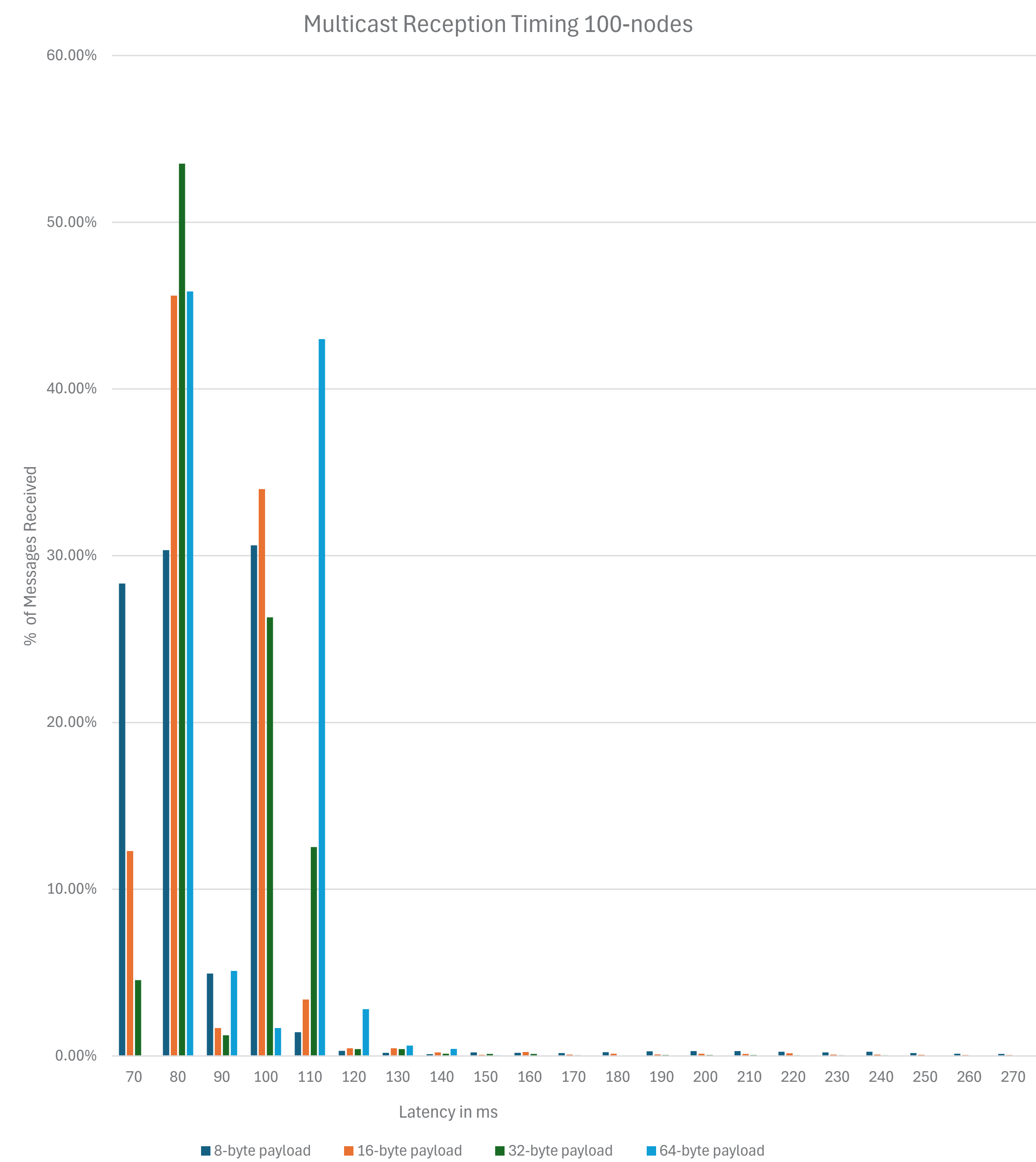
100-Node Network

Summary

Average Latency across all packet sizes: 91ms

The 95th percentile latency (95% of the commands are processed within this time frame): 110ms

Data

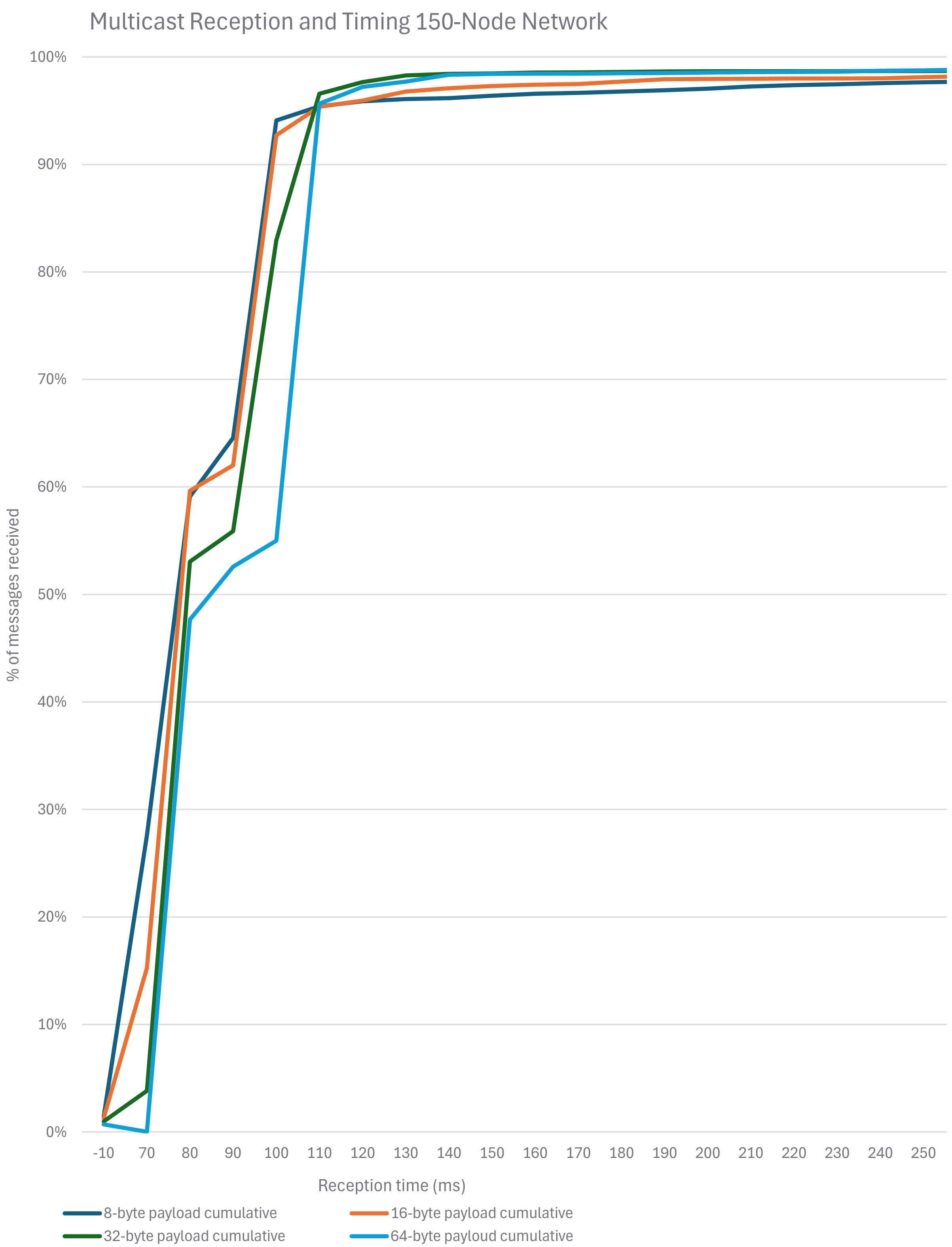


Cumulative Reception

When we aggregate the data across multiple buckets we get the following table of results and the graph below.

100 nodes				
Latency in ms	**8-byte payload**	**16-byte payload**	**32-byte payload**	**64-byte payload**
70.00	28.33%	12.29%	4.55%	0.01%
80.00	58.66%	57.89%	58.06%	45.86%
90.00	63.60%	59.56%	59.30%	50.96%
100.00	94.23%	93.56%	85.60%	52.63%
110.00	95.65%	96.95%	98.13%	95.62%
120.00	95.96%	97.42%	98.54%	98.43%
130.00	96.15%	97.88%	98.95%	99.05%
140.00	96.26%	98.10%	99.09%	99.47%
150.00	96.47%	98.17%	99.21%	99.51%
160.00	96.66%	98.41%	99.33%	99.52%
170.00	96.83%	98.49%	99.37%	99.53%
180.00	97.06%	98.63%	99.40%	99.55%
190.00	97.34%	98.72%	99.45%	99.57%
200.00	97.63%	98.85%	99.51%	99.59%
210.00	97.93%	98.97%	99.56%	99.62%
220.00	98.19%	99.13%	99.60%	99.65%
230.00	98.41%	99.21%	99.64%	99.67%
240.00	98.67%	99.29%	99.68%	99.70%
250.00	98.85%	99.36%	99.71%	99.72%
260.00	98.99%	99.41%	99.74%	99.73%
270.00	99.11%	99.47%	99.76%	99.76%
280.00	99.23%	99.50%	99.79%	99.78%
290.00	99.33%	99.53%	99.81%	99.79%
300.00	99.40%	99.58%	99.83%	99.80%
310.00	99.47%	99.61%	99.86%	99.81%
320.00	99.55%	99.64%	99.87%	99.89%
330.00	99.60%	99.67%	99.88%	99.91%
340.00	99.64%	99.69%	99.89%	99.92%
350.00	99.67%	99.71%	99.89%	99.93%
360.00	99.70%	99.73%	99.89%	99.94%
370.00	99.73%	99.74%	99.90%	99.94%
380.00	99.77%	99.76%	99.90%	99.95%
390.00	99.80%	99.77%	99.91%	99.95%
400.00	99.81%	99.78%	99.93%	99.96%
410.00	99.82%	99.79%	99.93%	99.96%
420.00	99.84%	99.80%	99.94%	99.97%
430.00	99.86%	99.81%	99.94%	99.97%
440.00	99.87%	99.82%	99.94%	99.97%
450.00	99.88%	99.83%	99.94%	99.97%
460.00	99.89%	99.84%	99.94%	99.97%
470.00	99.90%	99.85%	99.94%	99.97%
480.00	99.91%	99.86%	99.94%	99.97%
490.00	99.92%	99.87%	99.94%	99.98%
500.00	99.92%	99.87%	99.94%	99.98%
More	**0.07%**	**0.12%**	**0.04%**	**0.03%**
Mean Latency:	**90.14**	**89.96**	**90.38**	**96.35**

Graph of Cumulative Reception



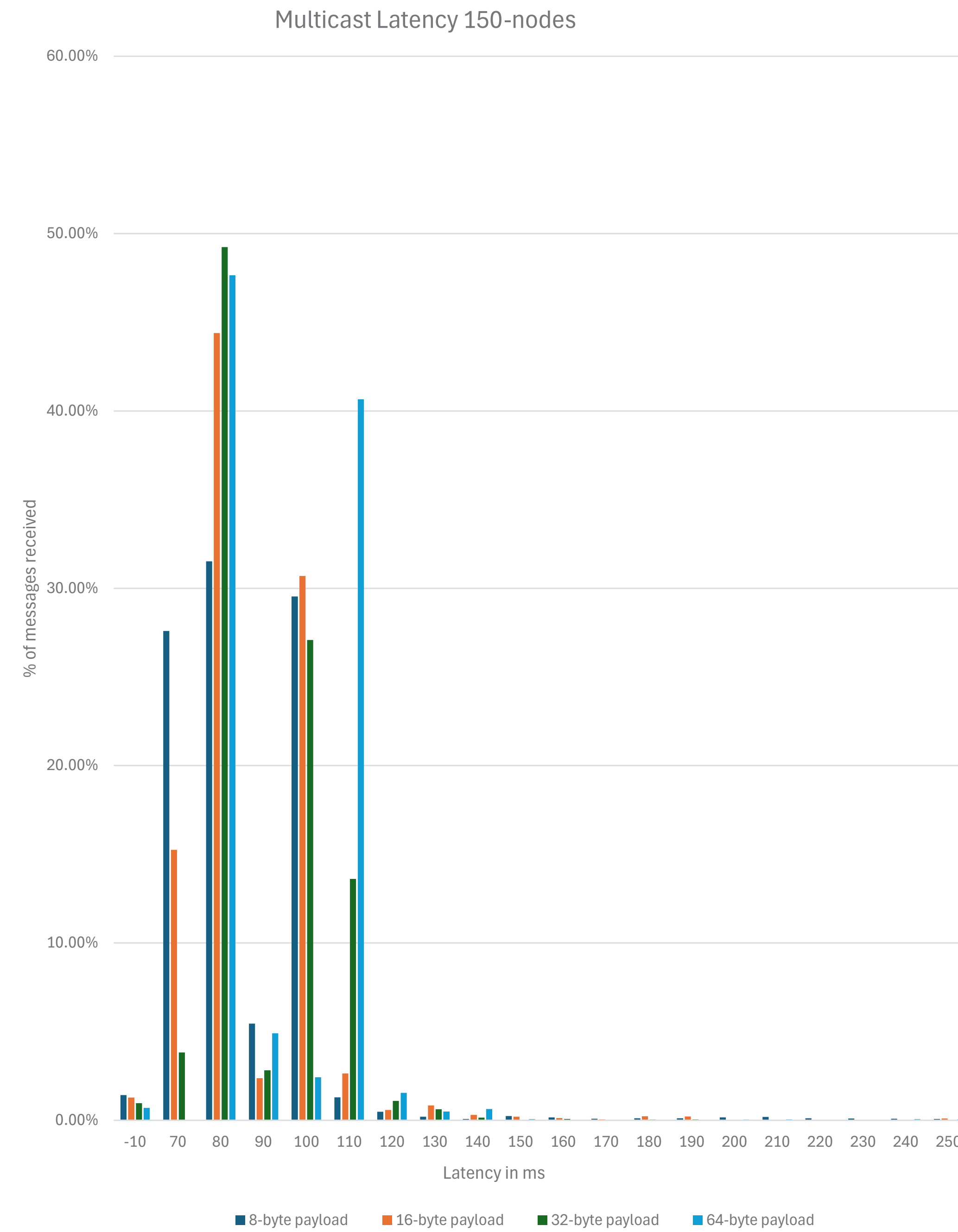
150-Node Network

Summary

Average Latency across all packet sizes: 91ms

The 95th percentile latency (95% of the commands are processed within this time frame): 110ms

Data

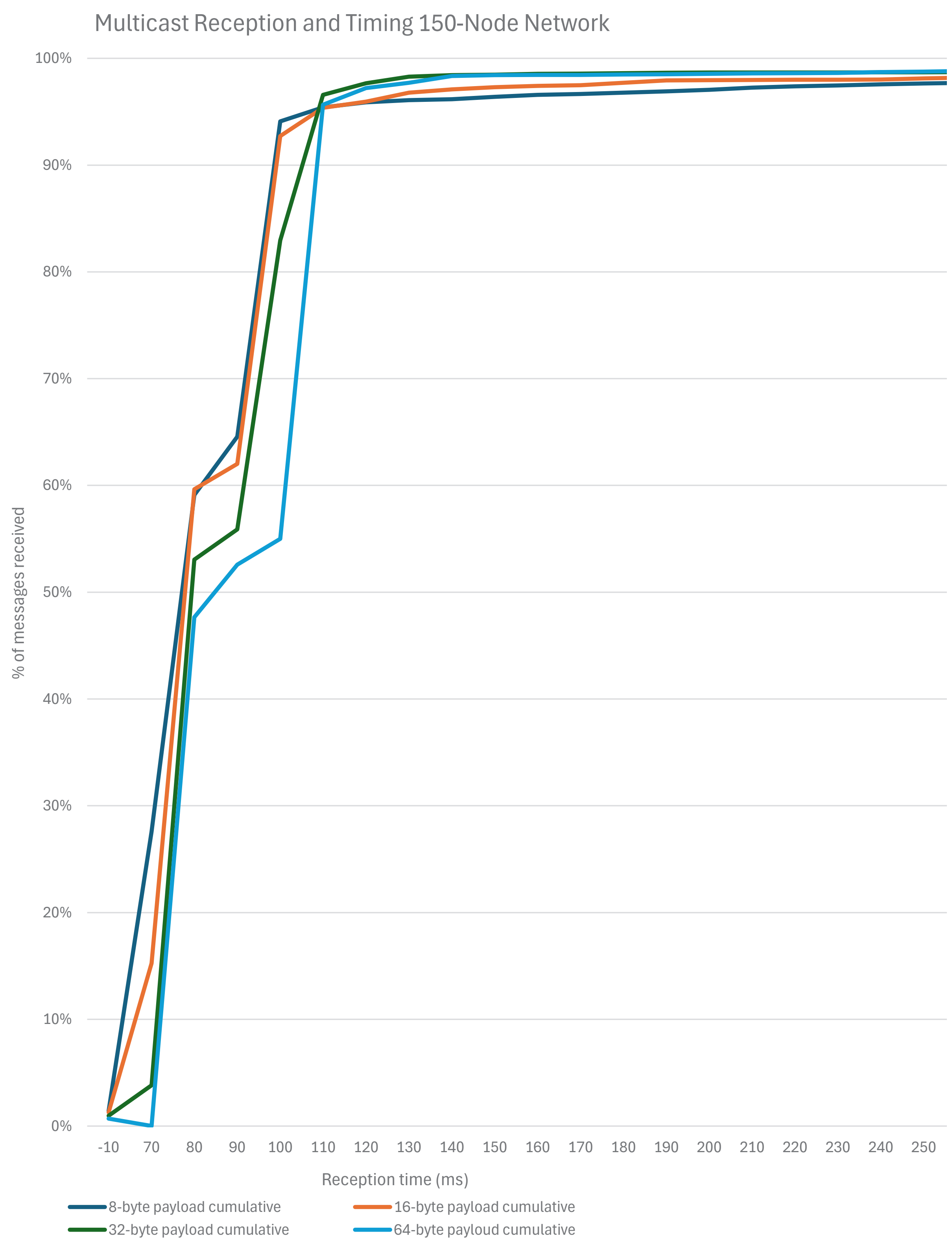


Cumulative Reception

When we aggregate the data across multiple buckets we get the following table of results and the graph below.

150 nodes				
Latency in ms	8-byte payload	16-byte payload	32-byte payload	64-byte payload
-10	1.42%	1.29%	0.97%	0.70%
70	27.59%	15.25%	3.82%	0.01%
80	59.11%	59.65%	53.06%	47.66%
90	64.56%	62.03%	55.88%	52.57%
100	94.10%	92.72%	82.96%	55.00%
110	95.40%	95.37%	96.57%	95.66%
120	95.88%	95.95%	97.66%	97.21%
130	96.08%	96.79%	98.28%	97.71%
140	96.16%	97.10%	98.43%	98.35%
150	96.40%	97.30%	98.46%	98.42%
160	96.57%	97.43%	98.54%	98.44%
170	96.66%	97.48%	98.57%	98.45%
180	96.78%	97.71%	98.61%	98.48%
190	96.90%	97.93%	98.66%	98.50%
200	97.06%	97.96%	98.67%	98.54%
210	97.25%	97.98%	98.67%	98.59%
220	97.37%	97.99%	98.68%	98.61%
230	97.47%	98.00%	98.68%	98.64%
240	97.56%	98.01%	98.69%	98.71%
250	97.64%	98.11%	98.69%	98.76%
260	97.70%	98.21%	98.72%	98.81%
270	97.75%	98.23%	98.78%	98.82%
280	97.79%	98.25%	98.79%	98.83%
290	97.86%	98.26%	98.86%	98.88%
300	97.92%	98.27%	98.86%	98.92%
310	97.95%	98.28%	98.86%	98.95%
320	98.02%	98.29%	98.86%	99.01%
330	98.07%	98.31%	98.86%	99.05%
340	98.11%	98.35%	98.86%	99.06%
350	98.14%	98.38%	98.92%	99.07%
360	98.17%	98.40%	98.93%	99.08%
370	98.19%	98.46%	98.93%	99.09%
380	98.21%	98.47%	98.94%	99.12%
390	98.23%	98.48%	98.95%	99.13%
400	98.24%	98.49%	98.96%	99.14%
410	98.25%	98.50%	98.96%	99.14%
420	98.27%	98.51%	98.96%	99.14%
430	98.28%	98.52%	98.96%	99.15%
440	98.29%	98.52%	98.96%	99.16%
450	98.30%	98.53%	98.96%	99.16%
460	98.31%	98.54%	98.96%	99.16%
470	98.32%	98.54%	98.96%	99.16%
480	98.33%	98.55%	98.96%	99.16%
490	98.34%	98.56%	98.96%	99.16%
500	98.34%	98.56%	98.96%	99.16%
More	**0.23%**	**0.14%**	**0.04%**	**0.11%**
Mean Latency:	**86.64**	**87.57**	**90.37**	**95.36**

Graph of Cumulative Results



200-Node Network

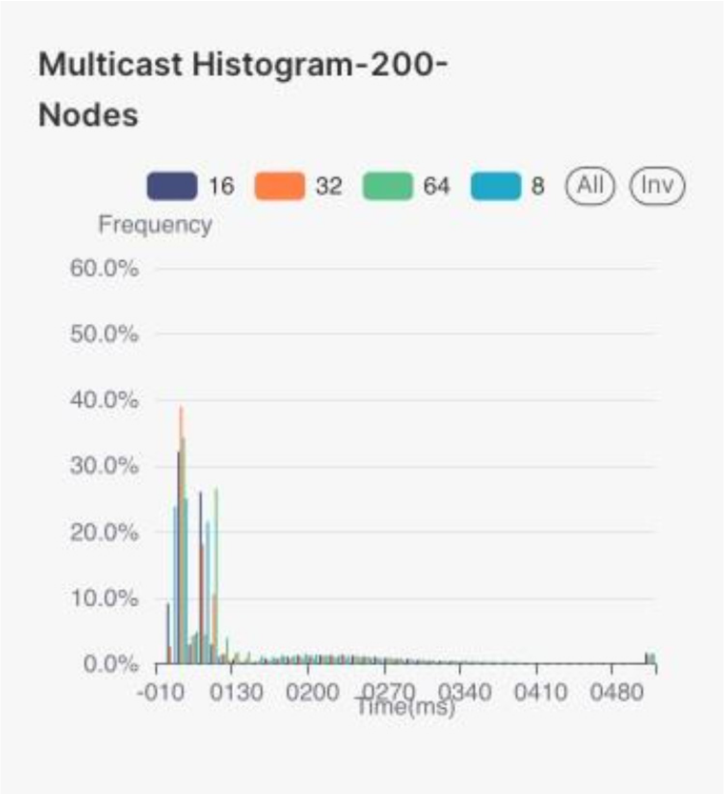
Summary

Average Latency across all packet sizes: 91ms

The 95th percentile latency for each payload size (95% of the commands are processed within this time frame):

- 8 bytes: 320ms
- 16 bytes: 350ms
- 32 bytes: 330ms
- 64 bytes: 340ms

Data

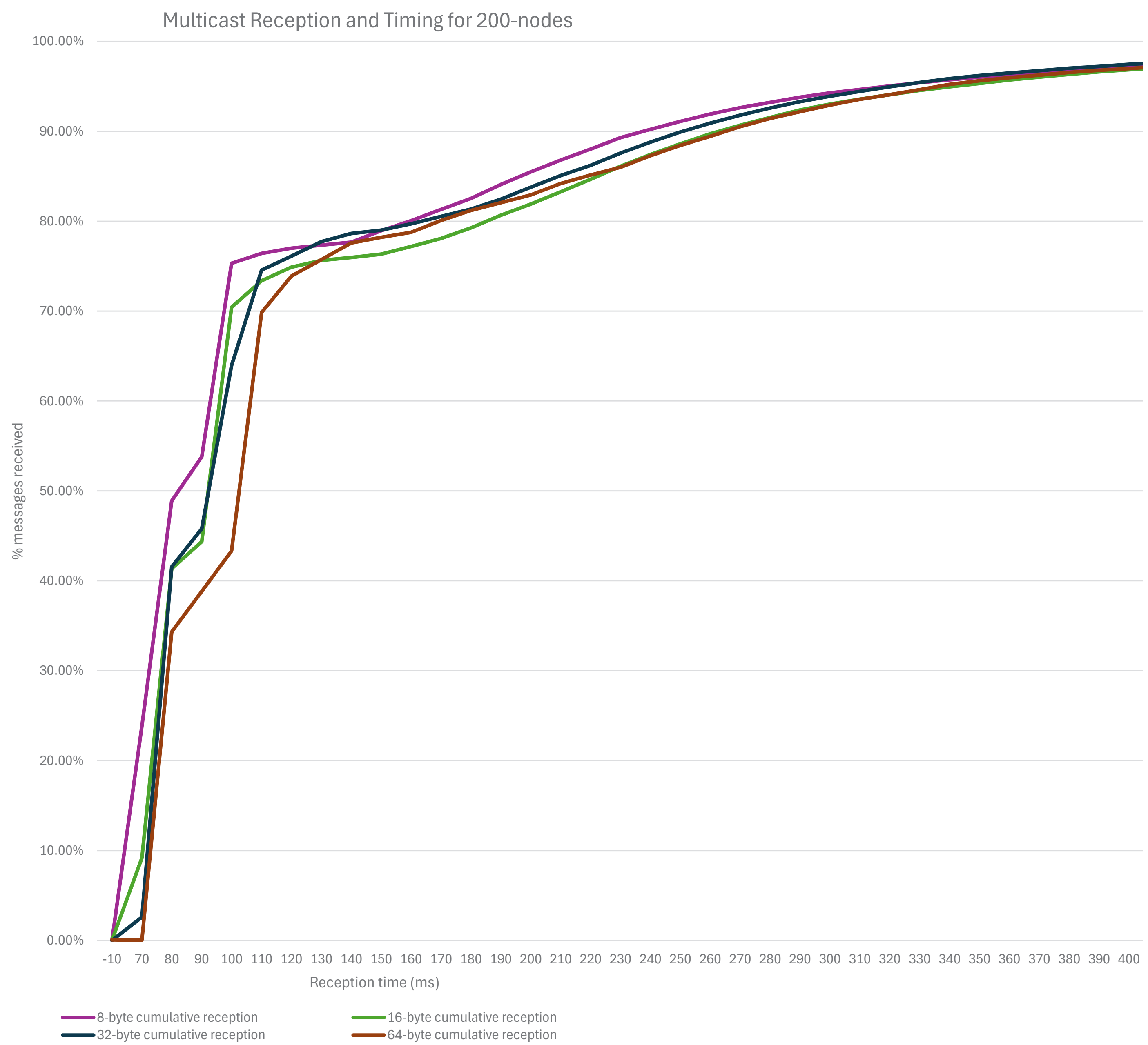


Cumulative Reception

When we aggregate the data across multiple buckets we get the following table of results and the graph below.

200 nodes				
Latency in ms	**8-byte payload**	**16-byte payload**	**32-byte payload**	**64-byte payload**
measurement	**8**	**16**	**32**	**64**
-10	0.01%	0.02%	0.01%	0.06%
70.00	23.85%	9.17%	2.60%	0.02%
80.00	48.90%	41.37%	41.55%	34.32%
90.00	53.79%	44.35%	45.79%	38.82%
100.00	75.32%	70.43%	63.96%	43.32%
110.00	76.41%	73.39%	74.57%	69.85%
120.00	76.99%	74.88%	76.11%	73.89%
130.00	77.35%	75.64%	77.72%	75.72%
140.00	77.66%	75.96%	78.64%	77.57%
150.00	78.93%	76.33%	79.01%	78.21%
160.00	80.06%	77.20%	79.70%	78.77%
170.00	81.30%	78.07%	80.53%	80.08%
180.00	82.53%	79.25%	81.34%	81.20%
190.00	84.08%	80.65%	82.44%	82.06%
200.00	85.49%	81.91%	83.77%	82.91%
210.00	86.79%	83.26%	85.07%	84.18%
220.00	88.02%	84.66%	86.22%	85.14%
230.00	89.30%	86.10%	87.57%	86.00%
240.00	90.21%	87.41%	88.80%	87.28%
250.00	91.11%	88.60%	89.91%	88.43%
260.00	91.91%	89.72%	90.90%	89.44%
270.00	92.62%	90.67%	91.79%	90.51%
280.00	93.22%	91.53%	92.59%	91.43%
290.00	93.80%	92.36%	93.29%	92.18%
300.00	94.26%	93.01%	93.89%	92.91%
310.00	94.65%	93.57%	94.43%	93.56%
320.00	95.05%	94.06%	94.95%	94.08%
330.00	95.40%	94.54%	95.43%	94.63%
340.00	95.72%	94.95%	95.86%	95.22%
350.00	96.02%	95.33%	96.21%	95.62%
360.00	96.30%	95.70%	96.48%	95.97%
370.00	96.54%	96.02%	96.75%	96.25%
380.00	96.75%	96.33%	97.02%	96.54%
390.00	97.00%	96.62%	97.22%	96.78%
400.00	97.21%	96.84%	97.45%	96.97%
410.00	97.38%	97.05%	97.62%	97.22%
420.00	97.56%	97.24%	97.80%	97.40%
430.00	97.71%	97.41%	97.97%	97.56%
440.00	97.82%	97.57%	98.11%	97.75%
450.00	97.95%	97.71%	98.25%	97.90%
460.00	98.08%	97.85%	98.36%	98.03%
470.00	98.20%	98.00%	98.45%	98.16%
480.00	98.31%	98.13%	98.55%	98.26%
490.00	98.42%	98.26%	98.65%	98.35%
500.00	98.43%	98.28%	98.66%	98.36%
More	**1.56%**	**1.70%**	**1.27%**	**1.58%**
Mean Latency:	**116.51**	**125.45**	**124.11**	**129.76**

Graph of Cumulative Results



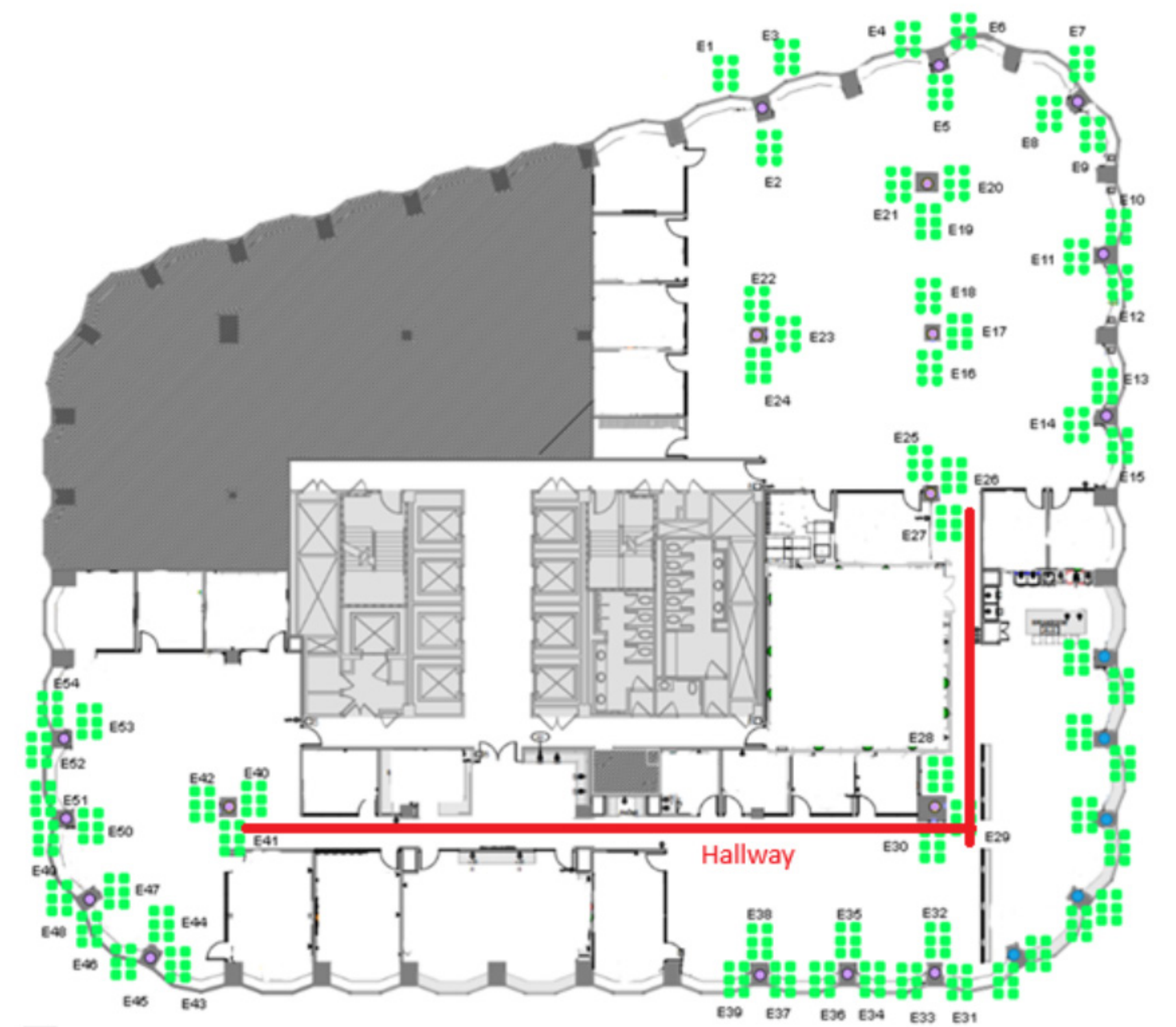
Difference in Results

It is clear that the 200-node network has a greatly different latency curve than the other network sizes. This was fairly unusual at first but as we dug into the results we quickly saw a weakness in the devices that became routers and the physical topology of the Boston office.

When the network is formed, this is done via automated scripts that bring up the nodes based on the ordering of their assigned test network IP addresses. The cluster number goes from E1 to E50 and they are commissioned in increasing number from E1, E2, E3, E4 up to E50.

When building up the 200-node network from the 150-node the additional 50 nodes were placed in a location in the bottom left-hand corner of the building that has a single line of sight to the rest of the network, thus forcing multi-hop forwarding/flooding for the furthest-away nodes. As expected, this resulted in a significant latency increase at the tail of the distribution -- this can be seen in the 95-percentile latency readings for the 200-node network.

In addition, due to the position of nodes on different sides of columns it triggers an extra hop to reach the node on the perpendicular or opposite side.



Combined Results Summary

Mean Latency

	8-byte payload	16-byte payload	32-byte payload	64-byte payload
50 nodes	89.80ms	89.88ms	90.82ms	94.18ms
100 nodes	90.14ms	89.96ms	90.38ms	96.35ms
150 nodes	86.64ms	87.57ms	90.37ms	95.36ms
200 nodes	116.51ms	125.45ms	124.11ms	129.76ms

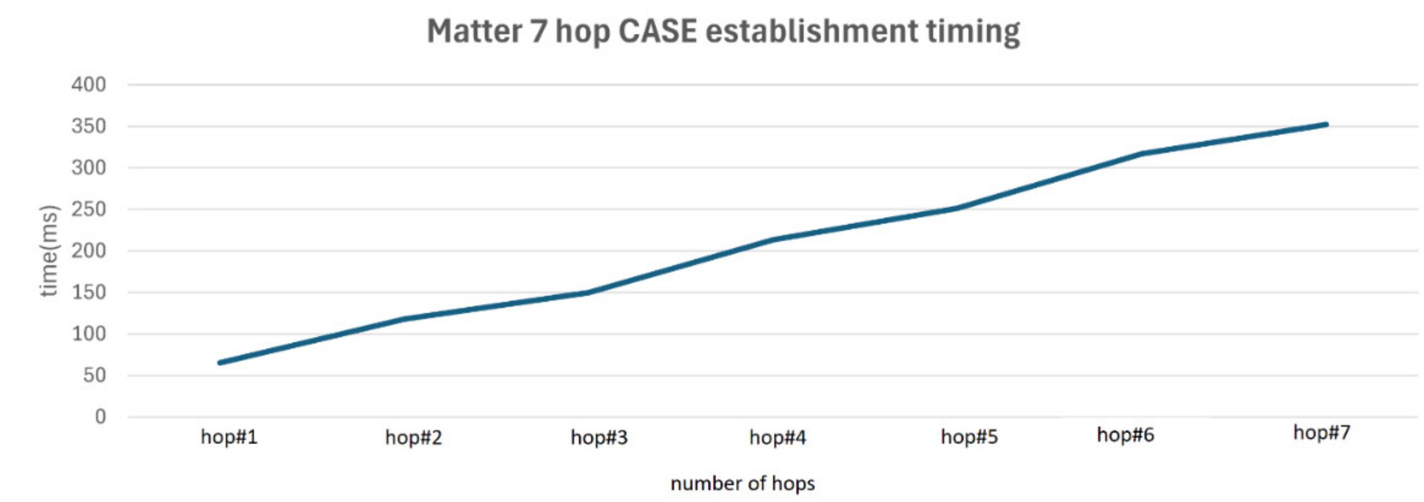
95th percentile latency (95% of the commands are processed within this time frame):

	8-byte payload	16-byte payload	32-byte payload	64-byte payload
50 nodes	110ms	110ms	110ms	110ms
100 nodes	110ms	110ms	110ms	110ms
150 nodes	110ms	110ms	110ms	110ms
200 nodes	320ms	350ms	330ms	340ms

See the Analysis chapter for a discussion of the layout of the 200-node network and the latency numbers.

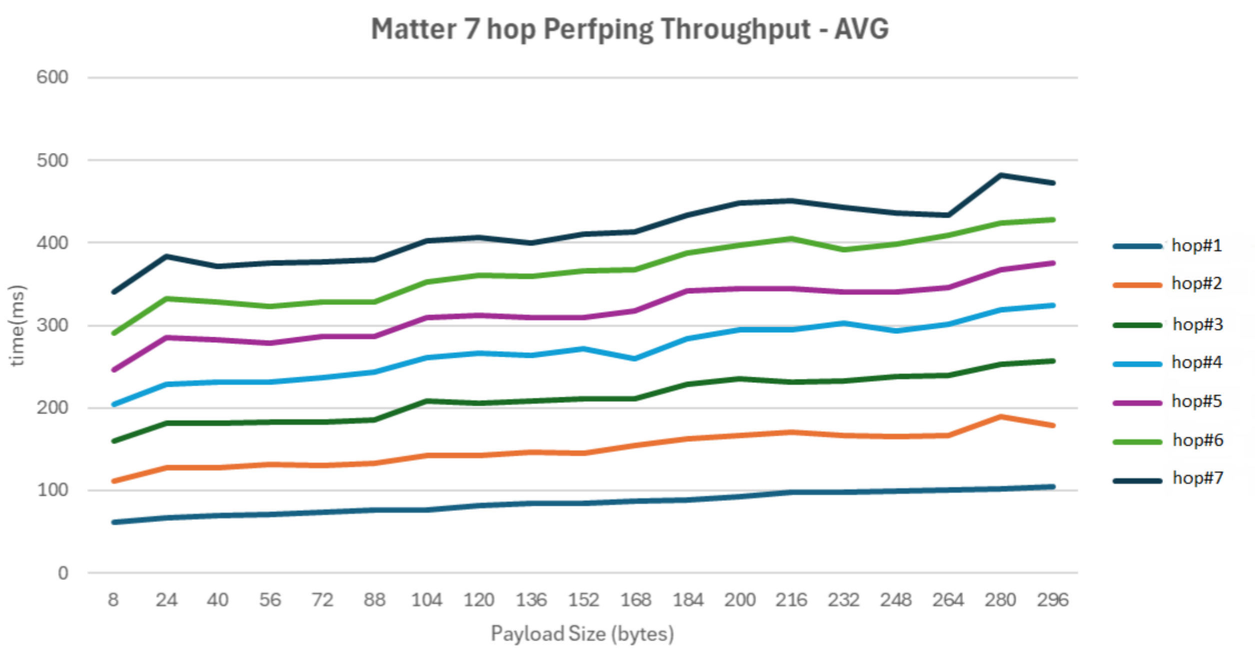
Unicast Results

The latency of the first command sent to a given node is tracked and reported separately since the exchange involves establishing a CASE session. The charts below display the unicast latency values (in milliseconds) of the first command sent to a given node across different payload sizes, highlighting the incremental latency introduced by each network hop.



Hop Count	Payload Size	Packets Transmitted	Packets Received	Packet Loss	Minimum-Time	Average Time	Maximum Time	Time Units
1	8	1	1	0	65	65	65	ms
2	8	1	1	0	118	118	118	ms
3	8	1	1	0	150	150	150	ms
4	8	1	1	0	213	213	213	ms
5	8	1	1	0	251	251	251	ms
6	8	1	1	0	317	317	317	ms
7	8	1	1	0	352	352	352	ms

The charts below display unicast latency values (in milliseconds) across different payload sizes, highlighting the incremental latency introduced by each network hop.



Hop Count	Payload Size	Packets Transmitted	Packets Received	Packet Loss	Minimum-Time	Average Time	Maximum-Time	Time Units
1	8	10	10	0	55	62	72	ms
1	120	10	10	0	76	82	86	ms
1	200	10	10	0	84	92	107	ms
1	296	10	10	0	96	105	115	ms
2	8	10	10	0	97	112	125	ms
2	120	10	10	0	132	143	154	ms
2	200	10	10	0	146	166	173	ms
2	296	10	10	0	166	179	188	ms
3	8	10	10	0	146	160	173	ms
3	120	10	10	0	187	206	221	ms
3	200	10	10	0	225	235	261	ms
3	296	10	10	0	238	257	274	ms
4	8	10	10	0	185	204	214	ms
4	120	10	10	0	234	266	275	ms
4	200	10	10	0	277	295	310	ms
4	296	10	10	0	312	324	337	ms
5	8	10	10	0	225	246	261	ms
5	120	10	10	0	275	312	331	ms
5	200	10	10	0	318	345	372	ms
5	296	10	10	0	359	376	393	ms
6	8	10	10	0	270	291	304	ms
6	120	10	10	0	336	360	381	ms
6	200	10	10	0	370	397	426	ms
6	296	10	10	0	398	428	469	ms
7	8	10	10	0	310	341	359	ms
7	120	10	10	0	364	407	442	ms
7	200	10	10	0	407	448	479	ms
7	296	10	10	0	446	473	526	ms

Set up and longevity Test

A general test was conducted to evaluate the stability and longevity of a large-scale Matter-over-Thread network under real-world conditions, with the goal of understanding how such a network performs over extended periods in environments that closely resemble commercial deployments. This test aimed to validate whether Matter-over-Thread could maintain consistent commissioning, low packet loss, and reliable communication in such conditions. Additionally, it sought to uncover any long-term degradation in responsiveness or connectivity that might emerge after hours of operation.

Test Environment

The test was performed in a typical office setting, as described in the Open-Air Environment section of this document. The environment included natural interference from numerous devices operating on Thread, Wi-Fi and Bluetooth. Importantly:

- **No isolation measures** were taken to shield the test network from surrounding wireless activity.
- **No artificial congestion** was introduced to simulate interference; the test relied solely on ambient network traffic.
- This setup was intended to reflect realistic deployment conditions for Matter-over-Thread networks.
- The network consisted of **200 nodes**, each running the **Performance Testing Application** based on the Matter - *SoC Lighting over Thread example.*

Commissioning Test

- Each device joined the Thread network using a CLI-triggered command that applied the network dataset provided by the **OpenThread Border Router.**
- After joining the Thread network, each device was **commissioned onto the Matter fabric** using the on-network commissioning process.
    - **Average commissioning time:** ~7 seconds per node
    - **Success rate:** 100%

Longevity Test

- A **group** was configured across all 200 nodes.
- One node repeatedly sent a custom cluster command (defined in the Performance Testing Application) to the group address.
- Packet drop rate:
    - ~0.01% for 8, 16,and 32-byte payloads
    - ~1% for 64-byte payloads
- The network remained fully operational with consistent performance and packet delivery rates after 3 continuous hours of operation.

Challenges Faced

BLE Commissioning Limitations and the Shift to On-Network Commissioning

During our testing, we encountered limitations with BLE commissioning, particularly in terms of range and the number of devices it could reliably support. These constraints often prevented successful commissioning of all devices in a single test run, and doing this with automation.

To overcome these challenges, we adopted an on-network commissioning approach. This method leverages the existing Thread network infrastructure. Devices first join the Thread network using the network dataset provided by the OTBR (OpenThread Border Router). Once connected, they register an SRP (Service Registration Protocol) service.

Following this, the on-network commissioning command is used to commission the devices onto the Matter-over-Thread network. This approach eliminates the need for BLE-based interactions, thereby bypassing BLE's inherent limitations and enabling more scalable and reliable device commissioning.

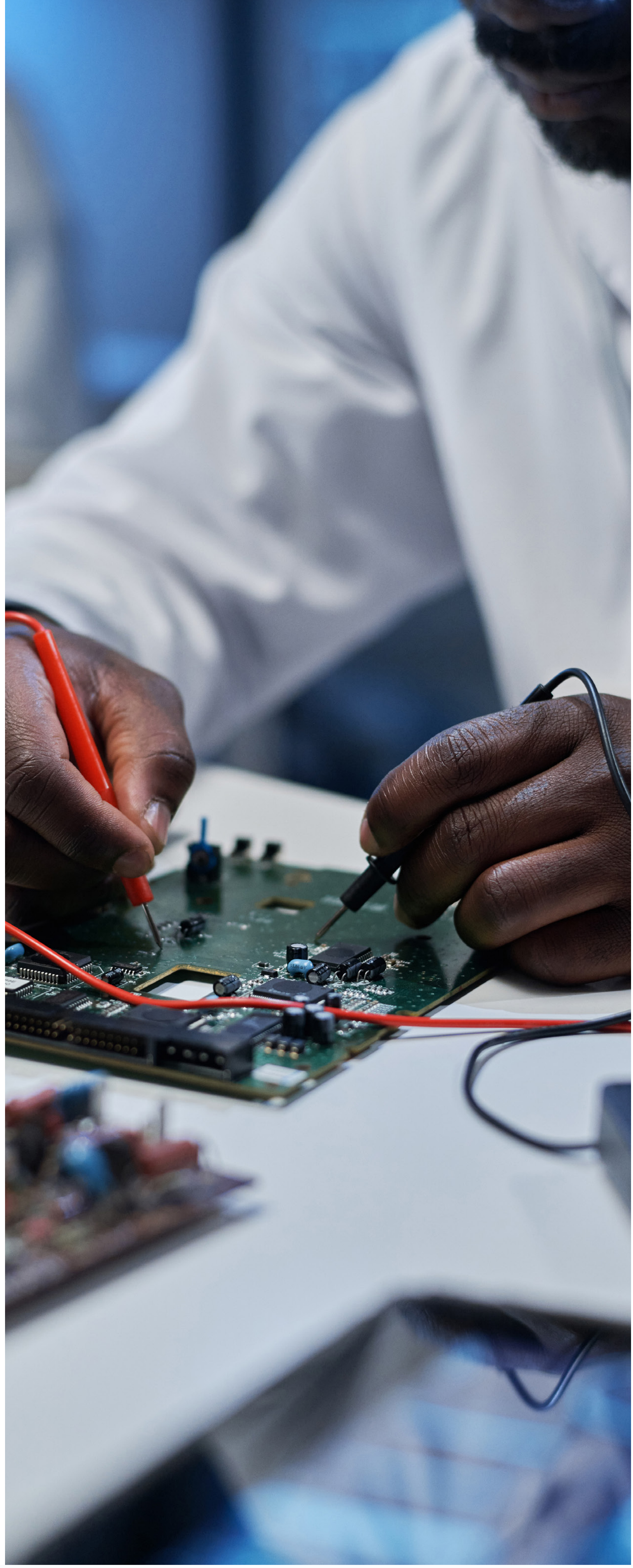
Precision Timestamps

The initial method for measuring multicast latency—relying on CLI printouts or logging—proved to be inadequate. The primary goal of the test was to assess the user experience, specifically how quickly a device responds to a Matter command, such as a light turning on in response to a switch press. However, the additional delay introduced by the CLI and logging mechanisms distorted the latency measurements, making them unreliable.

To address this issue, the test methodology was revised to use Silicon Labs' Packet Trace Interface (PTI). This approach allowed for precise timestamping of message transmission and reception events. The timing data was then extracted through the WSTK backchannel, enabling accurate measurement of application-layer latency without the interference of logging overhead.

CLI Logging

Initial tests on the default image used in development showed unexpectedly high latency. Disabling Matter logging reduced latency by approximately 50%.



Analysis

In performance testing, Matter-over-Thread consistently exhibited higher latency compared to ICMP over Thread (i.e. a ping test). This outcome is expected and can be attributed to several contributing factors:

- **Application-Layer Latency:** In both unicast and multicast scenarios, latency is measured all the way up through the application layer. This introduces additional overhead due to the need to decode cryptographic elements, parse Matter messaging structures, and process cluster commands. These operations are inherently more complex than the lightweight processing required for ICMP messages.

- **Group Message Processing Overhead:** Matter's group messaging functionality is known to be computationally intensive. Processing group commands involves more complex logic and resource usage than individual unicast messages. To address this, Silicon Labs has initiated Matter WG efforts to optimize group message handling. These improvements are expected to reduce latency and enhance overall network responsiveness.

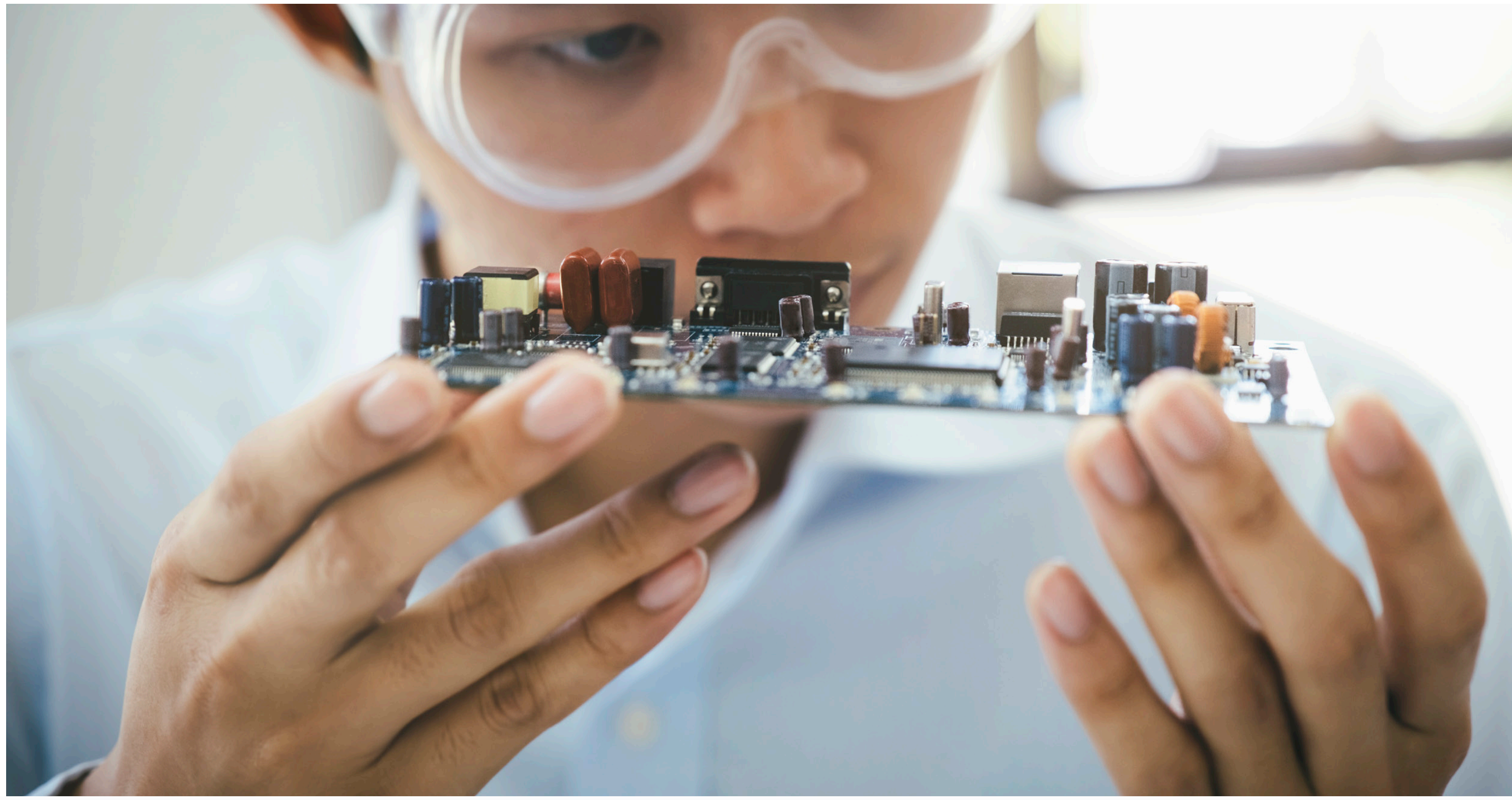
- **Multithreaded Execution Environment:** Matter operates within a multithreaded software architecture which introduces additional processing costs. Each Matter node must manage multiple concurrent tasks, such as message handling, security processing, and service discovery. This is in contrast to devices running native IP over Thread, which typically have a simpler execution model of running on bare metal with no RTOS. The added complexity affects not only the nodes sending and receiving messages but also those involved in message forwarding, contributing to increased end-to-end latency.

Other observations:

- Average latency increases with the increasing payload sizes. This is expected due to larger transmission time and fragmentation at the Thread layer.

- Average latency increases with network size. This is expected due to the forwarding delays from the extra hops.

- When building up the 200-node network from the 150-node the additional 50 nodes were placed in a location within the building that has a single line of sight to the rest of the network, thus forcing multi-hop forwarding/flooding for the furthest-away nodes. As expected, this resulted in a significant latency increase at the tail of the distribution -- this can be seen in the 95-percentile latency readings for the 200-node network.



Plans for Future Testing

There are many additional scenarios and stress conditions that can be explored to further evaluate network behavior, resilience, and operational efficiency. Below are several ideas for future testing that could help deepen understanding and guide improvements in Matter deployments.

Large Network Reboot Recovery

- Objective: Measure the time it takes for a 200-node (or larger) network to resume full operation after a complete power cycle.
- Key Aspects:
    - How long it takes for nodes to rejoin the Thread network.
    - Time until all nodes are reachable by the Matter controller.
    - Recovery of group communication functionality and CASE sessions.
- Motivation: In commercial environments (e.g., offices, retail, factories), power disruptions can occur. Rapid recovery ensures minimal downtime and operational continuity.

CASE Session Recovery

- Objective: Assess the reliability and speed of re-establishing CASE (Certificate Authenticated Session Establishment) sessions after session loss or device reboot.
- Key Aspects:
    - Success rate of session re-establishment.
    - Time required for the new session to become usable.
    - Behavior under repeated disruptions.
- Motivation: CASE sessions are central to secure communication. Their resilience directly impacts network stability and responsiveness as they must be re-established prior to any functional Matter messaging to occur.

Wi-Fi and Thread Coexistence

- Objective: Investigate how Wi-Fi traffic influences Thread network performance under real-world conditions.
- Approach:
    - Introduce multiple high-throughput Wi-Fi networks.
    - Vary packet sizes and test interference on different RF channels.
- Key Aspects: Identify thresholds where Wi-Fi congestion impacts Thread reliability or latency, and explore channel planning or prioritization techniques.

Congested Network Stress Testing

- Objective: Simulate high-traffic scenarios by having all nodes periodically report to a central controller.
- Key Aspects:
    - Effect on commissioning success rate and time.
    - Impact on unicast and multicast latencies.
    - Packet drop rates and retries.
- Motivation: Validates the network's performance under data-heavy workloads common in real-world IoT scenarios like environmental monitoring, building automation, or security systems.

Sensor Reporting Use Case

- Objective: Emulate frequent status updates, such as periodic temperature or occupancy reports, and measure delivery latency to the controller.
- Key Aspects:
    - Varying report intervals (1s, 10s, 30s).
    - Payload sizes and concurrent multicast activity.
- Motivation: Helps establish recommended configurations for sensors operating in constrained environments.

Testing with Real-World Controllers

- Objective: Replace the chip-tool CLI with real commercial Matter controllers (e.g., voice assistants, smart home hubs).
- Key Aspects:
    - Controller behavior under load.
    - Compatibility and feature completeness.
    - Responsiveness of mobile UIs or dashboards when commanding large device groups.
- Motivation: Gauges interoperability and readiness for consumer deployment beyond lab-grade tools.

Hardware and Software

SOFTWARE

- Silicon Labs Matter Extension 2.6.1-1.4
- Simplicity SDK 2025.6.1
- Simplicity Network Analyzer
- ot-br-posix commit (7d327005e)
- OpenThread RCP binary for BRD4187C, SiSDK 2025.6.1
- chip-tool built with the Matter 1.4 released version of Matter SDK

HARDWARE

- BRD4187C (EFR32MG24B220F1536IM48-B)
- Wireless Starter Kit BRD4001A Rev. A01
- 4GB RaspberryPi running Ubuntu 22.01.2 LTS OS

Conclusions

1. Scalability and Robustness:

The results of Silicon Labs' performance testing of Matter-over-Thread networks affirm that this technology is well-suited for large-scale and multihop deployments. Matter-over-Thread demonstrated strong scalability, with successful deployment and sustained operation of a 200-node network in a mixed-use, interference-prone office environment. No special topology engineering or manual router assignments were needed, validating the protocol's ability to form resilient mesh networks naturally.

2. Multicast Performance:

Multicast/group command handling was shown to be effective. Even under real-world conditions with 200 nodes, the system achieved acceptable application-layer latencies, making it suitable for scenarios such as group lighting control. The majority of nodes responded within an expected latency envelope, and reliability held across multiple payload sizes. As expected, placing nodes in remote or low-reachability locations increases latency due to the additional overhead of multicast flooding.

3. Commissioning at Scale:

The testing demonstrated that on-network commissioning is the preferred approach for large-scale Matter-over-Thread deployments. Across hundreds of nodes the commissioning process maintained high reliability and consistent timing with a 100% success rate. This method leverages the existing Thread mesh and removes the dependency on Bluetooth Low Energy (BLE), which showed critical limitations in large-scale setups. While BLE commissioning may remain viable for small network adjustments—such as adding or replacing a few nodes, it is not recommended for larger installations. BLE's limited range and capacity make it unsuitable for bulk operations. In contrast, on-network commissioning ensures broader reach, better automation potential, and smoother integration.

4. Unicast Stability and Latency Trends:

Unicast testing revealed predictable and incremental increases in latency with both payload size and hop count. While latency increases are expected due to cryptographic and transport-layer overhead, the system maintained low packet loss rates and consistent round-trip behavior. CASE session establishment times, though slower than subsequent command interactions, were still within acceptable limits.

5. Diagnostic and Debugging Limitations:

The study surfaced a key challenge in large-scale network diagnostics: many real-world devices lack sufficient on-device debugging tools. CLI-based methods were shown to significantly distort latency measurements, and while Packet Trace Interface (PTI) delivered high-fidelity timing data, it requires dedicated hardware and developer-level access. For production environments, solutions like remote telemetry, OTA log capture, or embedded diagnostic clusters will be essential for troubleshooting at scale.

SILICON LABS

CONNECTED INTELLIGENCE