Measuring Library Broadband Networks Training Manual - Final

MLBN
MEASURING LIBRARY BROADBAND NETWORKS

Published May 1, 2021

Simmons UNIVERSITY

CS&S Code for Science & Society

MLAB

INTERNET2

Table of Contents

Overview of Section 1: Welcome to Measuring Library Broadband Networks (MLBN) 8

I Just Want To Plug Things In! 8

How to Use This Manual 8

Project Description 9

Towards Gigabit Libraries Toolkit 9

Participating in MLBN 10

What is Measuring Library Broadband Networks for the National Digital Platform (MLBN) ? 10

What is being measured? 11

How are the MLBN tests different from other Internet speed tests? 11

Privacy and Security 12

Addressing Privacy Concerns 12

Measurement System Security 13

Why Measure Broadband in Public Libraries? 13

Benefits of Measuring Broadband 13

What is not being measured – and why not? 14

Overview of Section 2: Measurement Computer Setup Instructions 16

Pre-Installation 17

Wired Measurement Computer Installation 18

WiFi Measurement Computer Installation 24

Software Running on MLBN Measurement Computers 30

Data Visualization Platform 30

Overview of Section 3: Technical Detail 31

MLBN Broadband Measurement Platform 32
Tests Run from MLBN Measurement Computers 33
Measurement Computers 33
Hardware 34
Device Management and Data Collection 34
The Role of M-Lab Tests and Servers 36
Data Visualization Platform 36
Pre-Installation 37
Configuring Your Library Data Network for the Measurement Computers 37
Troubleshooting 39
Introduction: Setting up the MLBN Measurement System 41
Overview - Required Infrastructure, Services, and Knowledge 41
MLBN Measurement System Components 42
Required & optional services, accounts, hardware, and infrastructure 43
Balena Cloud - device management and configuration 43
GitHub - code and application deployment management 43
Measurement Computer Hardware 43
Computer & Helpful Accessories for Setup and Configuration 44
SCP Server and/or Google Cloud Storage for Test Data Archive (optional) 44
Server or Virtual Machine to Host Data Visualization Service (optional) 45
Create a Measurement System Deployment Plan 45
Setup Measurement System Infrastructure 52
Setup Your Development Computer(s) 52
Setup a Test Unit and Alternate Operating System SD Card 52
Obtaining MAC Addresses Using an Ubuntu Testing Device 52

Setup a GitHub Account 54

Review Options for Data Storage Location(s) of Test Data 56

Set All Application Settings 56

Overall Application Settings 56

Test Runner Settings 57

Exporter Settings 57

Local Exporter 58

SCP Exporter 58

GCS Exporter 59

HTTP Exporter 60

Setup a Balena Cloud Account & Application 61

Create an account 61

Create a Balena Cloud Application 62

Connect Your Balena Application to Your Local Copy of Murakami 63

Configuring Defaults for the Balena Cloud Application 63

Prepare to Setup Measurement Devices 64

Name the New Measurement Computer - Device Location, Network Type, and Connection Type Settings 65

Identifying the MAC addresses of measurement computers network interfaces 66

Label the Device 67

Setup Your Measurement Devices 68

Setup New Device Using the Balena Cloud Website 68

Setup New Device Using the Balena CLI 71

Obtain a Balena OS image for your device type 71

Register a new device in your application 72

Generate a configuration file for this device 72
Flash the Balena OS image to an SD card 73
Apply your generated configuration file for this device to the SD card 73
Special Configurations for Specific Network Conditions 74
Special Configuration for WiFi Connected Devices 74
Special Configuration to Support Custom DNS Servers 75
Add Your DNS servers to the Balena OS configuration file. 75
Add Your DNS servers to the System Connections Configuration File(s) 76
Configure a Static IP Address for Ethernet Connected Device 77
Configure a Static IP Address for WiFi Connected Device 78
Summary of Balena CLI Commands / Scripting with Balena CLI 79
Push a Murakami Release to Balena Cloud 80
Prepare and Push a Release 81
Deploy Measurement Devices in Intended Locations 85
Troubleshooting Tips 86
Setup the Measurement System Data Visualization Service 86
Installing Murakami Viz 87
System Requirements 87
Structure 87
Configuration 87
Deployment 88
Standalone 88
Docker 88
Configuring Murakami Viz After Installation 89
Administrator Setup Tasks 89

Add About, Contact, FAQ, or Glossary content if desired 90
Add Locations and Measurement Devices 90
Whitelist IP Address(es) for the Location 94
Add Users for each location if desired 95
Accessing and Using Test Data 96
Using the Optional Data Visualization Service 96
Accessing Data stored in remote locations defined by Murakami GCS or SCP exporters 100
Accessing data on each device 100
Accessing data via the Balena Dashboard 100
Accessing data locally via the device's SD card 103
Converting Test Result Data from JSON to CSV 104
Sending Previously Collected Data to a Murakami Viz Instance 104
How to Use murakami-upload 105
Obtain test data from missing dates 106
On-Device Data 106
Data from GCS or SCP Server Archive 107
Transferring Balena Managed Devices to Another Application/Owner 108
Overview of Section 5: Glossary 109
Appendix A: The MLBN Team 113
Board of Advisors 113
Simmons University 113
Code for Science and Society / Measurement Lab 114
Internet2 115
Institute of Museum and Library Services 116
Pilot Libraries - Year One 117

Participating Libraries - Year Two 117

Appendix B: Example of data generated by the Measurement Computers 119

NDT (version 5) 119

NDT (version 7) 120

Speedtest-cli (single and multi-stream) 121

Appendix C: Pre-Installation Questions 123

MLBN Measurement Device Installation 123

Network Management Practices 124

Appendix D: Using MLBN Measurement Devices After the Program Ends 125

Re-purpose the measurement devices for your own use 125

Continue using the measurement devices to measure Internet service 125

Managing your Devices with Balena.io 125

Running the Measurement Software on the Device without Balena.io 125

Section I: Welcome to Measuring Library Broadband Networks (MLBN)

Overview of Section 1: Welcome to Measuring Library Broadband Networks (MLBN)

Welcome to Measuring Library Broadband Networks for the National Digital Platform (MLBN)!

I Just Want To Plug Things In!

Do you just want to start installing now and read the rest of this section later? Please see the setup instructions starting on page 15.

Please be sure to read all of the instructions. Installation success comes from the work you perform before you receive your measurement computers.

How to Use This Manual

This manual is divided into the following sections:

- **Section 1** (this section) provides an overview of the MLBN Project and basic equipment setup instructions for the Measurement Computers.
- **Section 2** provides **setup instructions** for the Measurement Computers
- **Section 3** provides additional **technical information** about the program
- **Section 4** provides instructions for setting up and running all components of the MLBN measurement system
- **Section 5** is a glossary of technical terms used in this project
- **Appendix A** has descriptions of the **MLBN Team**
- **Appendix B** contains example data generated by the **Measurement Computers**
- **Appendix C** contains **Pre-Installation Questions**
- **Appendix D** contains options for libraries to continue using MLBN Measurement Computers

Materials published by the MLBN team in this manual are available under a Creative Commons Attribution-NonCommercial-ShareAlike license: https://creativecommons.org/licenses/by-nc-sa/3.0/us/

Attribution-NonCommercial-ShareAlike 3.0 United States (CC BY-NC-SA 3.0 US)

Project Description

Simmons University, Measurement Lab, and Internet2 are working together with public libraries across the United States to measure library broadband networks. The purpose of this research project is to better understand how libraries leverage broadband to serve the "National Digital Platform" (https://americanlibrariesmagazine.org/2015/06/11/the-national-digital-platform-for-libraries-and-museums/). Through May 2020, the project team will work with up to 60 libraries in rural, suburban, and urban communities across the United States to gather quantitative and qualitative data using participatory design approaches.

The project aims to:

- Understand the broadband speeds and quality of service that public libraries receive;
- Assess how well broadband service and infrastructure are supporting libraries' communities' digital needs;
- Understand broadband network usage and capacity, along with additional data that would be useful to public libraries in providing their communities with online software applications and social and technical infrastructure; and
- Increase libraries' knowledge of networked services and connectivity needs.

The research is funded by a grant (#LG-71-18-0110-18) from the U.S. Institute of Museum and Library Services (IMLS) National Leadership Grant program.

To learn more, please visit the MLBN project home page at http://slis.simmons.edu/blogs/mlbn

Towards Gigabit Libraries Toolkit

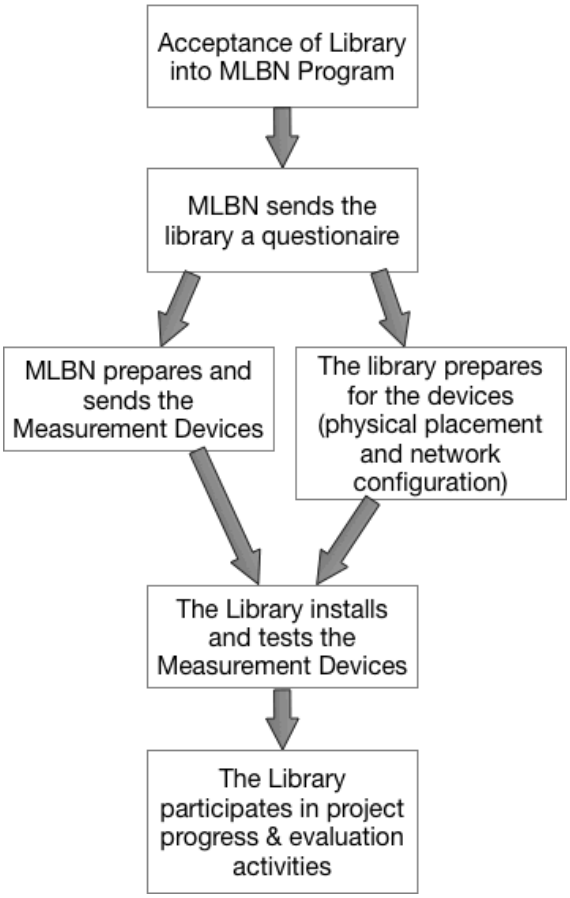
The MLBN project builds upon the work piloted in the Towards Gigabit Libraries grant and toolkit funded by the Institute for Museum and Library Services Laura Bush 21st Century Librarian Program (award# RE-00-15-0110-15), the grant created a "Broadband Toolkit" and customized "Broadband Improvement Plan" designed to help rural and tribal librarians learn about their current broadband infrastructure and internal information technology (IT) environment.

Through the use of the "Broadband Toolkit" and "Broadband Improvement Plan," librarians are better equipped to improve their broadband services and become stronger advocates for their libraries' broadband infrastructure needs.

The Toward Gigabit Libraries toolkit is available at http://www.internet2.edu/tgl/.

Participating in MLBN

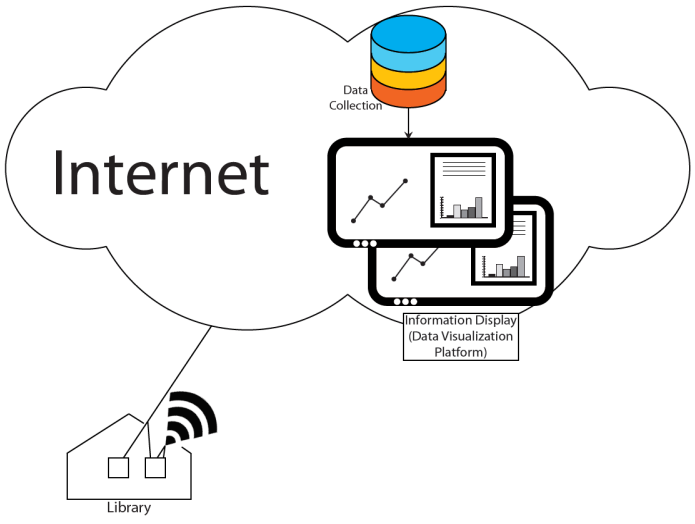
This flowchart illustrates the major steps you'll experience as an MLBN program participant. Further details about the installation process can be found in Section 2 of this training manual.



What is Measuring Library Broadband Networks for the National Digital Platform (MLBN) ?

The purpose of the MLBN Project is to provide public libraries with a way to accurately measure the **speed** and **quality** of their internet connections by taking standardized equipment and measures. This approach provides a **prototype broadband measurement system,** which includes automated measurement of connection speeds and quality of service metrics using a variety of Internet measurement tests.

The MLBN Project is using participatory design research techniques. This means that each public library that participates plays an important role in helping to ensure that the broadband measurement system is useful and effective for public libraries across the U.S. The prototype broadband measurement system will be refined and expanded over time based on the experiences reported and needs expressed by the library community.



How the System Works

1. One or more small computers are connected to an institution's network, and the code that they run is configured to run a selection of Internet measurement tests, including: M-Lab's Network Diagnostic Tool (NDT), additional M-Lab hosted tests, and third party tests such as speedtest.net.
2. Service used to manage, provision, administer, and update all of the measurement computers in

the field.

3. Service that receives test data from small computers, and provides an analysis and visualization, where project and library staff can access.

What is being measured?

The Measurement Computers are pre-configured to perform the following tests of the **speed** and **quality** of the library's broadband connection:

Test	Description
NDT (Network Diagnostic Tool)	A bandwidth and latency test provided by M-Lab that measures upload and download speeds to the nearest M-Lab server hosted in a peering location where networks connect with one another. Two protocol versions of the NDT test are included, ndt5 and ndt7. Both NDT protocols measure the bulk transport capacity of a connection using a single TCP stream. NDT 7 improves on the previous version by using the latest TCP features.
Speedtest.net (speedtest-cli)	Speedtest.net is a widely used commercial bandwidth and latency test platform provided by Ookla. This test attempts to measure the full capacity of the Internet connection using multiple TCP streams. We are including an open source, community-developed client, speedtest-cli, which uses the speedtest.net platform to provide this popular test.

How are the MLBN tests different from other Internet speed tests?

There are many Internet speed tests available. Often people assume that because these tests provide the same metrics or variables, like upload and download speed, their results should be similar. The Internet is not one network - it's an interconnected network of smaller networks and larger ones. Different tests have different methods, and they measure different parts of the path between people and Internet content.

The MLBN measurement system incorporates multiple Internet speed and quality of service tests and provides a comparison of the results of each test. M-Lab's NDT test is conducted between the Measurement Computer and the nearest M-Lab server in data centers where consumer ISP networks connect their networks to the Internet. The servers used by the Speedtest.net test are usually hosted by consumer ISPs, near the edge of their networks, but still inside the last mile. Both of these tests provide useful information about different parts of the networks that provide you Internet service.

Let's use an analogy to illustrate why having measurements from NDT and Speedtest.net are an important part of the MLBN project. Imagine for a moment that you're planning a trip to a nearby city, and you want to know how long it might take to drive there from your home in the suburbs. To get there you take a county maintained road until you get to the interstate, which takes you the rest of the way. If we measured the time it took to get to the point where the county road meets the interstate, this would be similar to running a Speedtest.net test, where the server is at the edge of the "county road" network. If we measured from your home to the final destination, this is more like running an M-Lab NDT test. The path is longer and the endpoint is not within the county road network. Having an understanding of both provides you better information about your entire trip.

The MLBN measurement system also standardizes how tests are conducted, which is important for consistency of the results. By using a small but powerful computer to run the tests automatically, the MLBN system addresses issues that can arise when tests are run from an assortment of computers with different ages and system capabilities. By placing measurement computers inside a public library, the MLBN system assesses both types of connections independently and continuously, providing an automated way to observe speed and quality of service trends, rather than one or more single measurement snapshots.

Privacy and Security

As librarians and technologists ourselves, the project team shares the perspective of the library community to ensure that patron privacy is protected and that documentation of our broadband measurement system clearly and effectively addresses those concerns. The security of our measurement system is paramount to the success of this project.

Addressing Privacy Concerns

In the Internet measurement community, there are generally two types of measurement methods: active and passive. Concerns about privacy and security generally arise when the use of passive testing or monitoring is in use. This approach might place a "pass-through" device between a router and a switch to watch traffic as it "passes by." ***Passive measurement or monitoring methods are not used by any of our measurement system devices or tests.***

Active tests are initiated by a person or automated by a person or organization. The full consent of the person or organization running the tests is required. The measurement computers to be placed in your library will serve as a proxy for patrons using the Internet. Our Measurement Computers do not monitor or inspect traffic on your networks or have any way of knowing the content your patrons are requesting. Each test generates its own random data solely for the purpose of measuring the device's connection to the Internet. Please see Appendix B to this document showing the data generated by each test.

Measurement System Security

Allowing our team to place a computer in your network that you're not managing directly requires trust that the device is secure. The MLBN team takes the security of our systems very seriously and will work with you to confirm that trust. Our prototype broadband measurement platform uses the Balena.io service (https://balena.io) to build containerized services which run on our devices. The Balena service security model is quite good and is described in detail in their documentation: https://www.balena.io/docs/learn/welcome/security/#security.

One of our project objectives is to produce a completely **open source** system at the lowest cost possible, and this may lead us to use something other than Balena. Our approach, in that case, will follow similar security requirements such as regular system and security updates, following the principles of least privilege and process isolation, as well as recognized security best practices for the system's cloud services components.

Why Measure Broadband in Public Libraries?

The MLBN Project relies heavily on technology, but its purpose goes beyond the technology itself. In conversations with public library professionals, we learned that access to reliable broadband measurement was lacking and prevented them from providing the best services possible. Having access to data and the skills needed to interpret the data would help them and their staff to build confidence; whether that would be for helping patrons with the internet or using broadband data to advocate with funders for internet service that meets the needs of libraries. Library professionals know they need information. Broadband measurement gives them confidence in that information and a way to communicate it. The MLBN team hopes that each participant grows in their knowledge of how technology - as a service - is being used in its library to respond to their community's digital demands.

This section is written to help participants understand the need for an open source broadband measurement system.

Benefits of Measuring Broadband

Our data networks, as part of a public library's internet infrastructure, are often *invisible* and unnecessarily mysterious. Measurements performed in the MLBN Project make those network visible to library staff and patrons as an important library resource

Measuring and evaluating these measures may help each participating library in the following ways:

Performance and Advocacy

Measures may lead to:

- Getting better performance of internal library data networks and Wifi by establishing baseline performance data; allowing for a means to troubleshoot problems; and being able to monitor the results of improvements.
- Establishing a basis for the library to advocate for faster and higher quality broadband speed.
- Having richer conversations with broadband providers, including opportunities to verify rated performance of Internet connections; ensure library (and in many cases, taxpayer) value for dollars spent; and opportunities for libraries and connectivity providers to collaborate on performance improvements.

Accurate and Consistent Comparative Data Over Time

Documenting where the library is now in terms of quality of its broadband connection may help plan for the future in the following areas:

- Speed and quality of the Internet connection
- Readiness for future services, especially those requiring robust connectivity
- Scalability planning for the library's Internet connection

Technology Confidence for Library Staff

In terms of understanding its technology, many library workers don't know where to start. By performing broadband measurement tests and reviewing the resulting information, library workers may gain greater confidence in understanding certain aspects of their technology services.

What is not being measured – and why not?

The MLBN measurement system is designed to measure the **performance of the network to the internet**, and is not designed as a general data network monitoring infrastructure.

Your library may also be interested in measuring other things that are not measured by the MLBN measurement system, including the number of uses of the WiFi system, information about the content of data traffic (without compromising privacy or confidentiality), or the ability to limit the amount of speed given to certain services (such as streaming media when a library has a very slow Internet connection). **These types of data are not provided by the MLBN measurement system,** but many libraries can obtain this data from their local systems or IT staff. Future enhancements to the MLBN system could support the ability to import these data sources.

Participating libraries are of all different sizes and capacities. The MLBN team understands that your library may want to take a deeper dive into measuring these and other aspects of your library's internet connectivity. The data generated by the MLBN project will have much deeper context if paired with data that the library is already collecting, such as WiFi and wired patron session reports. For those interested in this type of deeper work, the MLBN team would like to encourage libraries to use the data they already have in combination with other measurements. Further, at the conclusion of your involvement

with the MLBN Project, your library may want to learn even more about your network performance and traffic.

Section 2: Measurement Computer Setup Instructions

Overview of Section 2: Measurement Computer Setup Instructions

In this section you will learn how to install your broadband Measurement Computers.

Most program participants will have 2 Measurement Computers:

- A "**Wired Measurement Computer**" to measure the speed and quality of your library's primary data connection.
- A "**WiFi Measurement Computer**" to measure the speed and quality that an access point in your library that provides WiFi to multiple patrons.

For each installation, there are two essential steps:

1. Pre-installation (activities that you will perform prior to installing the device)
2. Physical installation (activities you will perform after you receive the device)

For public libraries of all sizes, it's a winning strategy to gather together people with the most technical knowledge -- especially about the library network -- to perform pre-installation and installation tasks. In large organizations, it's likely that the network configuration has some complexity, so anyone with detailed knowledge of your library's network configuration (whether it's a staff member, an outsourcer, or anyone else) should be invited to help with the pre-installation and installation activities for this project.

The MLBN Project team will prove assistance throughout your participation in the research. In addition to the information in this training manual, the project team encourages you to participate via the Discourse website (joining other project participants in the project; log-in instructions are sent to participants separately). You may also send an email to mlbn@measurementlab.net for any needs you might have at any point during the project

Pre-Installation

Pre-Installation

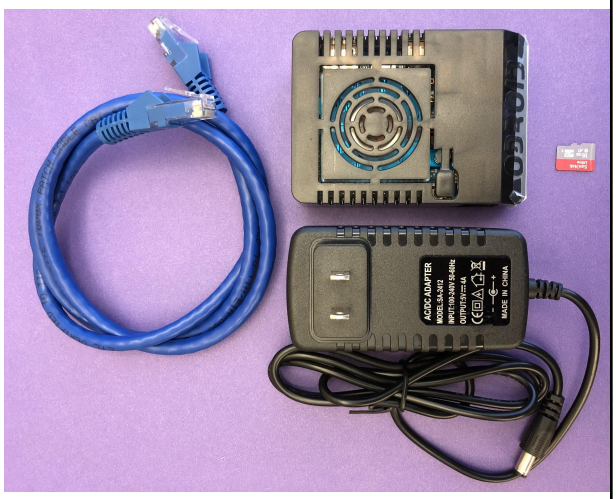
Before you receive your Measurement Computers, you will receive a questionnaire from the MLBN team asking some questions about your technology environment and making suggestions for configuring your library data network to allow the Measurement Computers to work properly.

If you haven't already, it's time to gather together people in your library who have the best understanding of your technology, especially the way your network is set up, to assist with the pre-installation and installation steps.

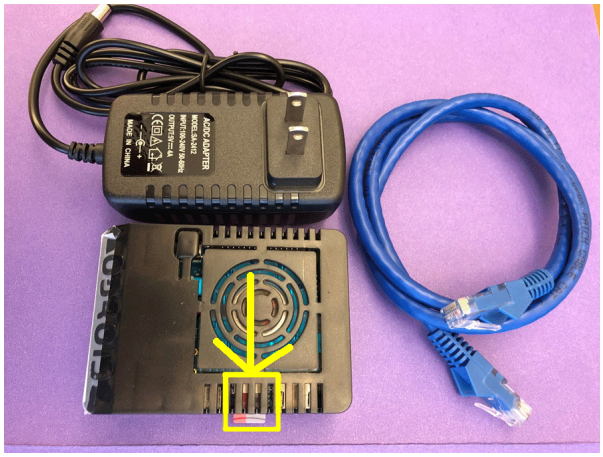
Wired Measurement Computer Installation

Wired Measurement Computer Installation

The Wired Measurement Computer is designed to measure the speed and quality of your primary data connection. For some libraries, this connection might be direct to the Internet; for others, this connection might be to a **Wide Area Network (WAN)** or another data network.

1. Wired Measurement Computer Components	
The components of the Wired Measurement Computer are (from top center): • Measurement Computer • Micro SD memory card (which should already be plugged into the ODROID Computer) • Power Supply • Ethernet Cable *Please note that the Wired Measurement Computer must contain all of these components to function properly. If any components are missing, please contact the MLBN team.*	

2. Device Location	
Find a suitable location for the Wired Measurement Computer. A suitable location will include: ● Easy access to a standard power outlet ● Somewhere near the network switch closest to your connection to the Internet (or for some, the connection to your Wide Area Network) ● An area safe from damage to the Wired Measurement Computer or other equipment	

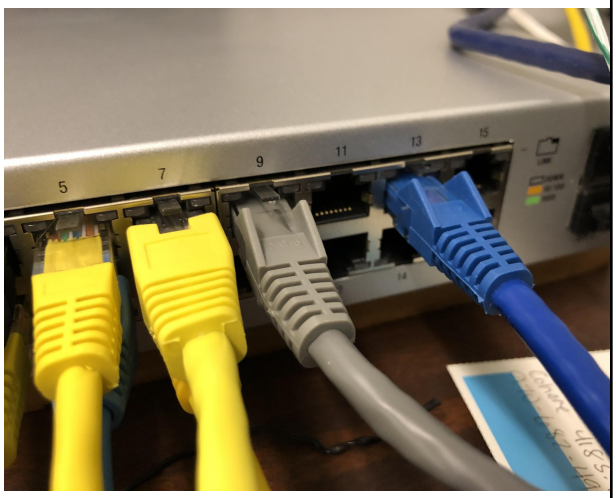
3. Micro SD Card	
Ensure the Micro SD card is plugged into its port on the Measurement Computer.	

4. Power Supply	
Connect the power supply to the ODROID computer.	

5. Ethernet Connection	
Connect the Ethernet cable to the Measurement Computer.	

Connect the other end of the Ethernet cable to an appropriate switch port.

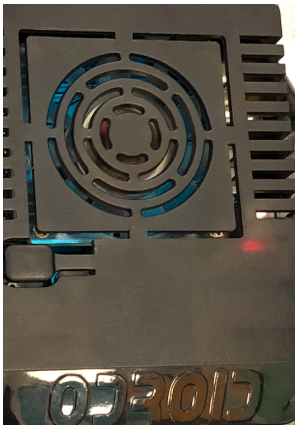
If you have a simple or small network, your switch ports may look something like this.

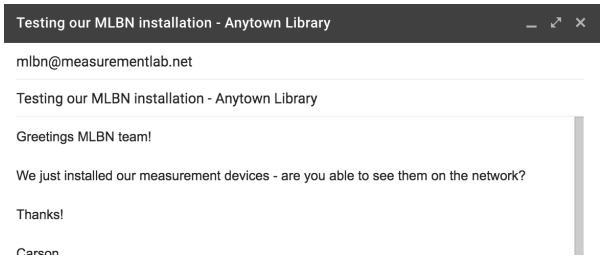


If you have a large or more complex network, your switch ports may look like this.

In this photo, one end of a blue ethernet cable is being plugged in to a spare switch port.

In this photo, the person performing the installation is connecting the other end of the ethernet cable to the Measurement Computer.	

6. Power Connection	
Connect the ODROID computer to power. Immediately upon plugin, you will see a red light indicating that the power is on.	

7. Test	
Once the devices are installed, please send an email to the project team at mlbn@measurementlab.net to ensure the devices are showing up on the network! Please use the following subject line in your email: "Testing our MLBN Installation - *(your library or location)*"	

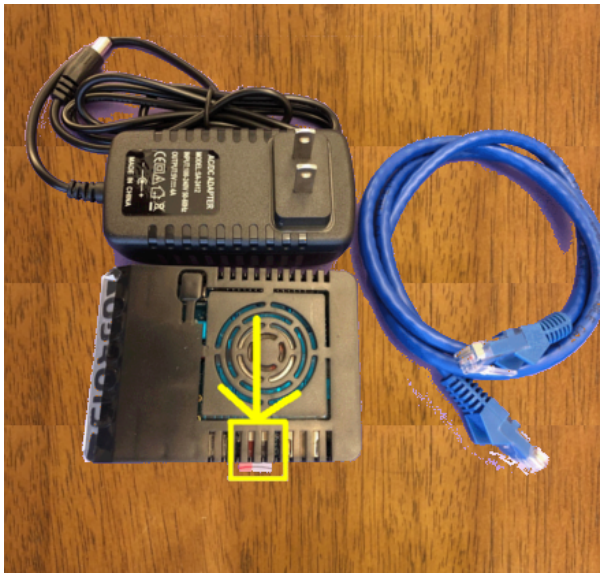
WiFi Measurement Computer Installation

WiFi Measurement Computer Installation

You will also have a Measurement Computer to measure the performance of your library's WiIFi network. This measurement computer will also be connected using an ethernet cable, but to a switch port or wall jack that would otherwise connect a WiFi access point in your library.

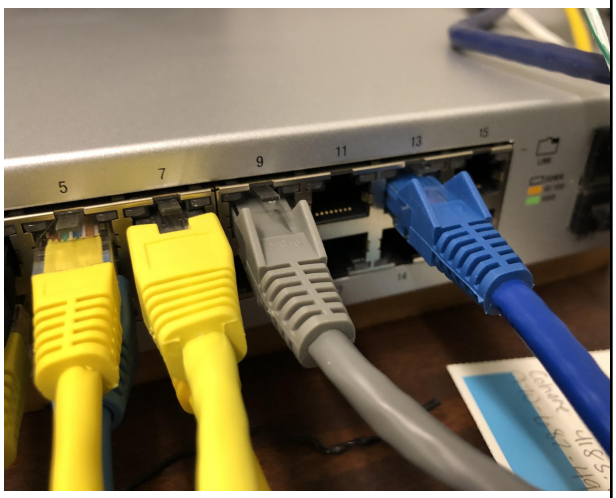
1. WiFi Measurement Computer Components	
The components of the WiFi Measurement Computer are (from top right): • ODROID Computer • Power Supply • Micro SD memory card (which should already be plugged into the Measurement Computer) • Ethernet Cable *Please note that the WiFi Measurement Computer must contain all of these components to function properly. If any components are missing, please contact the MLBN team.*	

2. Device Location	
Find a suitable location for the WiFi Measurement Computer. A suitable location will include: • Easy access to a standard power outlet • An ethernet port that would otherwise be used for a WiFi access point. This port could be on any **network switch** or wall jack that would otherwise connect an access point. • An area safe from damage to the Wired Measurement Computer or other equipment	

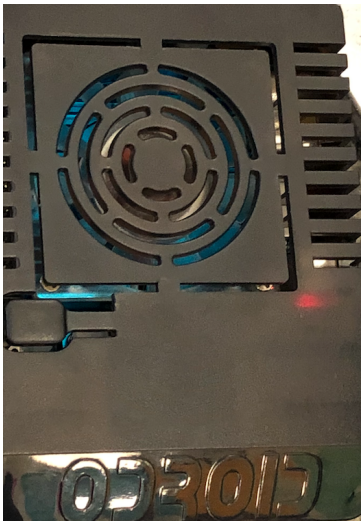
3. Micro SD Card	
Ensure the Micro SD card is plugged into its port on the Measurement Computer.	

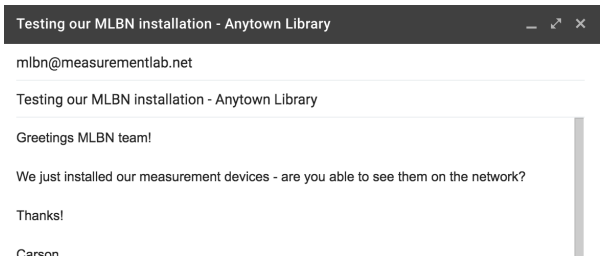
4. Power Supply	
Connect the power supply to the Measurement Computer.	

5. Ethernet Connection	
Connect the Ethernet cable to the Measurement Computer.	

Connect the other end of the Ethernet cable to an appropriate switch port. If you have a simple or small network, your switch ports may look something like this.	
	
If you have a large or more complex network, your switch ports may look like this. In this photo, one end of a blue ethernet cable is being plugged in to a spare switch port.	

In this photo, the person performing the installation is connecting the other end of the ethernet cable to the Measurement Computer.	

6. Power Connection	
Connect the ODROID computer to power. Immediately upon plugin, you will see a red light indicating that the power is on.	

6. Test	
Once the devices are installed, please send an email to the project team at **mlbn@measurementlab.net** to ensure the devices are showing up on the network. Please use the following subject line in your email: "Testing our MLBN Installation - *(your library or location)*"	

Software Running on MLBN Measurement Computers

The software running on the MLBN program's measurement computers is called *Murakami*, which enables regular, randomized tests to be conducted automatically. This software is flexible enough to support measurement initiatives of different scales, from one device to thousands. Supported "test runners" provide a modular approach to running any type of network measurement, both from M-Lab or from others. Data files from each test result can be saved on each device, or uploaded to a variety of cloud locations, where it can be archived, analyzed, and visualized.

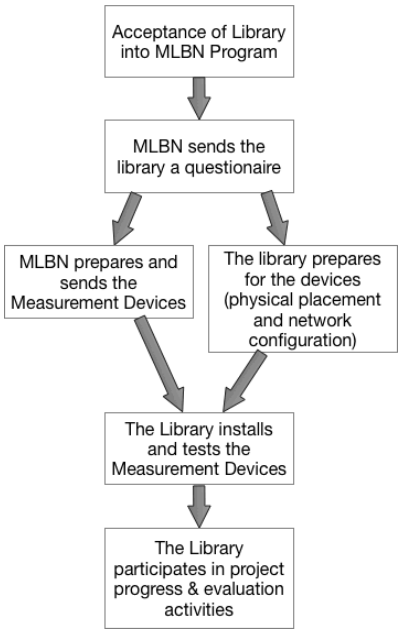
Data Visualization Platform

The data generated by each test can be archived centrally and used to build visualizations and reports in a variety of ways. Organizations with data science and visualization capacities might choose to integrate this data into other systems that they use already. To provide a complete system for all types of organizations, the MLBN project supported the development of a data visualization web application called *Murakami-Viz* that can receive data from measurement devices, and provide a way for users to review, annotate, and download it. The service is run on a standard web server that supports Docker.

Section 3: Technical Detail

Overview of Section 3: Technical Detail

This section of the document provides additional technical details to note when installing the measurement computers in your library. This section is aimed at staff with technical knowledge, but everyone is welcome to dig in here!

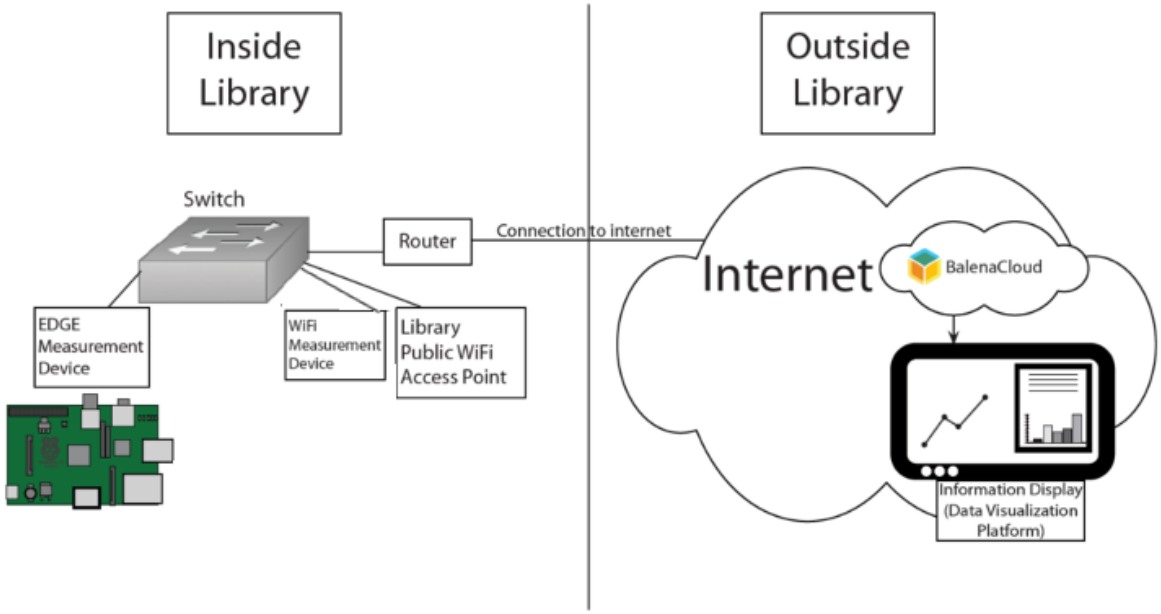


For libraries of all sizes, it's a winning strategy to gather together people with the most technical knowledge -- especially about the library network -- to perform pre-installation and installation tasks. In large organizations, it's likely that the network configuration has some complexity, so anyone with detailed knowledge of your library's network configuration (whether it's a staff member, an outsourcer, or anyone else) should be invited to help with pre-installation and installation activities.

Assistance is available throughout your participation in the MLBN project. Please send an email to mlbn@measurementlab.net for any needs you might have.

MLBN Broadband Measurement Platform

The current broadband measurement system provides a baseline: automated measurement of connection speeds and quality of service metrics using a variety of Internet measurement tests. The system will be refined and expanded based on the needs expressed by the library community, using participatory design research methods with this grant's research panel of library stakeholders.



The current measurement system consists of three main components:

1. One or more small Measurement Computers, are connected to an institution's network, and run code that is configured to run a selection of Internet measurement tests, including: M-Lab's Network Diagnostic Tool (NDT), additional M-Lab hosted tests, and third party tests such as speedtest-cli.
2. BalenaCloud service used to manage, provision, administer, and update all of the measurement computers in the field.
3. Visualization service that receives test data from small computers and provides an analysis and visualization that project and library staff can access.

Tests Run from MLBN Measurement Computers

NDT (Network Diagnostic Tool) NDT (Network Diagnostic Tool) is a speed and diagnostic test that 8reports upload and download speeds and attempts to determine what problems limit speeds. It also provides detailed diagnostics. Two versions of NDT are being used: ndt5 and ndt7.

Speedtest.net Speedtest.net is a widely used commercial bandwidth and latency test provided by Ookla. Internet service providers and network operators host Ookla servers in their networks as partners with the company. This "on-net" architecture for server placement differentiates Ookla/Speedtest from M-Lab and other commercial measurement services such as Akamai. Our broadband measurement platform has integrated a community developed command line interface for running tests to the speedtest.net platform (https://github.com/sivel/speedtest-cli), allowing our system to save Ookla test results for each device, and to directly compare between M-Lab and Ookla measurements. Our system runs both single TCP stream and multi TCP stream versions of this test.

Measurement Computers

- **Wired Measurement Computer, connected to an ethernet port on the router/switch**
  - One device will be labelled using the pattern: *<library name>-egress-wired*.
  - This device should be connected to an available port on the main switch or router where Internet service comes into the facility.
  - The port should provide our device with an IP address in the same range as those assigned to computers in your libraries that patrons use.
  - It is recommended that you identify a suitable port in advance, and confirm that a computer connected to it receives an IP address in the expected range.

- **WiFi Measurement Computer, connected to an ethernet port that would otherwise connect a WiFi access point**
  - One device will be labelled using the pattern: *<library name>-public-wifiAP*.
  - This device should be connected to an ethernet port in your library that would otherwise connect a WiFi access point..
  - The port should provide our device with an IP address in the same range as those assigned to other WiFi access points in the library.

- It is recommended that you identify a suitable port in advance, and confirm that a computer connected to it receives an IP address in the expected range.

Hardware

The Measurement Computers are small ODROID XU-4 computers running a Linux operating system and the measurement software, *Murakami*, from M-Lab. Wikipedia ODROID computer description: *"The **ODROID** is a series of single-board computers and tablet computers created by Hardkernel Co., Ltd. Even though the name ODROID is a portmanteau of open + Android, the hardware is not actually open because some parts of the design are retained by the company. Many ODROID systems are capable of running not only Android, but also regular Linux distributions."*[1]

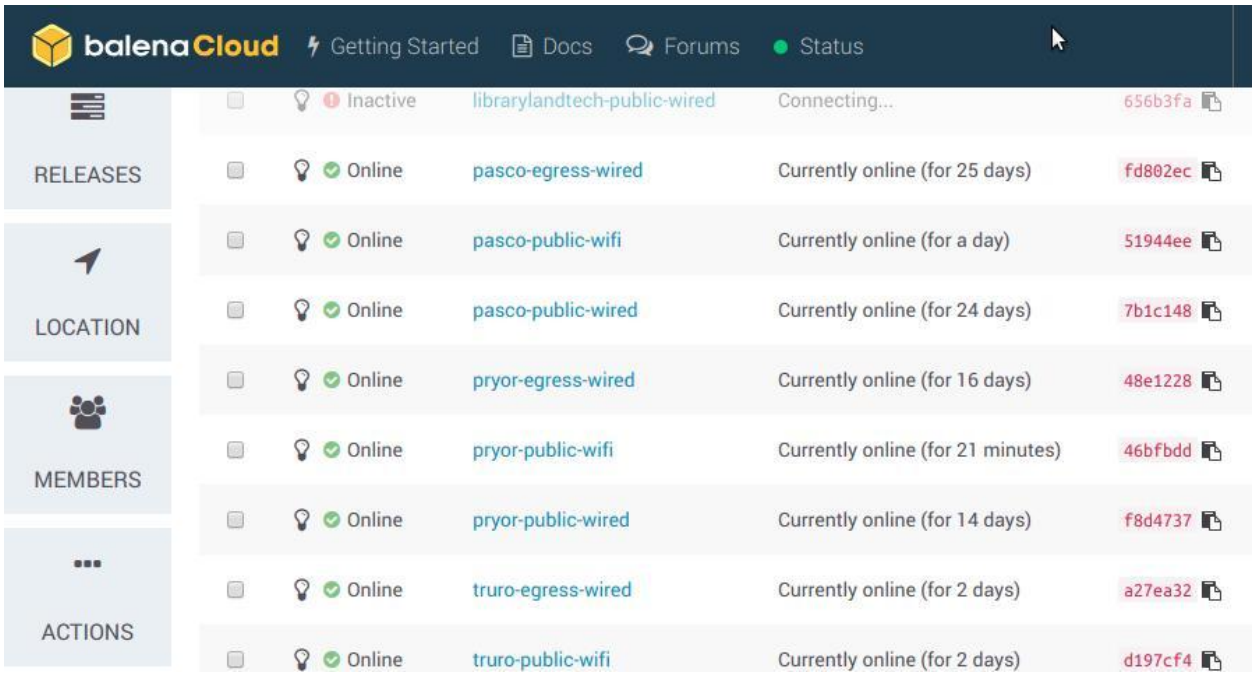
More information on ODROID XU-4 is available at the company website: https://www.hardkernel.com/.

The operating system being used is BalenaOS, a linux distribution created by Balena.io. The configuration information for each Measurement Computer is created by the MLBN team and written to a micro-SD card inserted into the card reader on the Measurement Computer.

Device Management and Data Collection

To provide scalable device setup and management of our Measurement Computers, we selected Balena Cloud, an "Internet of Things" (IoT) virtualization and device management software as a service (SaaS) provider with a commitment to open source tools.

[1] https://en.wikipedia.org/wiki/ODROID



Balena allows us to setup and pre-configure a wide array of single board computers, register them in a project connected to our code, provide remote access to them, and enables the ability to push updates to an entire fleet of devices. Balena has also recently released an open source server edition, Open Balena (https://www.balena.io/open/), that should provide a means for self-hosting the tools used in their supported Balena Cloud service. These options fit well with other open source services used by the library community such as LibLime's (http://www.liblime.com/) support for the open source Integrated Library System, Koha (https://koha-community.org/). For this research, we've subscribed to the Balena Cloud service.

We wanted our measurement system to be easy to set up and manage, and using Balena has made that possible. But also knew that we would need to coordinate with both library staff and administrators, as well as with their IT support staff and possibly with their system vendors, for a couple of reasons. Depending on the library's network management practices, firewall exceptions to allow our Measurement Computers to communicate with Balena services would need to be added. Also, the ports currently used by version 5 of the NDT test (NDT5) are sometimes blocked by firewalls. In the second year of the MLBN program, support the new version 7 of the NDT test (NDT7) was added to the measurement system, which removes these port requirements, running either over port 80 or port 443. The MLBN measurement devices will use port 80 for all NDT tests. NDT7 uses the latest TCP compression

algorithm, BBR, which stands for Bandwidth, Bottleneck, and Round Trip Time. NDT7 using BBR (https://queue.acm.org/detail.cfm?id=3022184) estimates the end-to-end performance of a connection.

The Role of M-Lab Tests and Servers

M-Lab servers (https://www.measurementlab.net/status/) through which tests are conducted are hosted within Internet exchange points, or IXPs, and have built in redundancy and testing infrastructure. Each M-Lab server "pod" consists of four servers and a high capacity switch connected to a single transit provider with an uplink speed between 1 Gbps and 10Gbps.

Currently, all M-Lab's US servers have 10Gbps links. In large metro areas, M-Lab hosts between 4 and 8 server pods, each connected to different providers. Three of the servers in each pod are used for production traffic, and the fourth is used for testing.

Data Visualization Platform

The data generated by each test can be archived centrally and used to build visualizations and reports in a variety of ways. Organizations with data science and visualization capacities might choose to integrate this data into other systems that they use already. To provide a complete system for all types of organizations, the MLBN project supported the development of a data visualization web application called *Murakami-Viz* that can receive data from measurement devices, and provide a way for users to review, annotate, and download it. The service is run on a standard web server that supports Docker.

Pre-Installation

Pre-Installation

Before you receive your Measurement Computers, you will receive a questionnaire from the MLBN team (called the *Information Sheet and Questionnaire - Measuring Library Broadband Networks - Year 2 Libraries*) asking some **questions about your technology environment**, and making **suggestions for configuring your library data network** to allow the Measurement Computers to work properly. Please see Appendix C for the full document.

Configuring Your Library Data Network for the Measurement Computers

To help you or your staff in adding firewall rules, exceptions, whitelisting, etc. above, MLBN can:

- Provide you with the MAC address for each Measurement Computer.
- Configure any Measurement Computer with a static IP address that you provide.

If you need assistance, please get in touch with the MLBN team: mlbn@measurementlab.net

Firewall Exceptions Rules (to add if needed)

This section contains network configuration information to ensure that the Measurement Computers work correctly in your library.

1. **For all devices, add firewall exceptions/rules if needed**
   - The management software used by MLBN requires several URLs and web ports to be open to function properly.
   - Some of MLBN's measurement tests also require a range of ports to be open bi-directionally. These are listed in the table below.

Ports Required:	***Reason:***
3001-3010, 32768-65535 TCP	Required by M-Lab's NDT5 test protocol. Firewall rules need to be bidirectional.
22 TCP	SSH protocol for device communication with our central data repository server.

443 TCP	Allows devices to connect to the Balena.io VPN and the web terminal, and many web endpoints using TLS
123 UDP	For NTP time synchronisation with the following servers: 0.resinio.pool.ntp.org 1.resinio.pool.ntp.org 2.resinio.pool.ntp.org 3.resinio.pool.ntp.org
53 UDP	For DNS name resolution
Domains / URLs that need to be allowed for our devices:	
*.measurement-lab.org *.balena-cloud.com *.docker.com *.docker.io	

Providing Additional Data on Network Sessions

Our research seeks to understand public internet service at your library. If you can provide any of the information below, please send it to the MLBN team: mlbn@measurementlab.net

1. **Data to compare the overall upload and download bandwidth used by your network over time with our speed and quality of service measurements.** This data is usually available via network switches through the Simple Network Management Protocol (SNMP). When we visit your location, we would like to discuss with your IT staff whether SNMP data is available and how we might access it.

   Can you provide us with SNMP data or reports of your overall network bandwidth usage?

2. **Understanding the number of public computers and WiFi sessions, and their length.**

Can you provide us with reports showing session start time, end time, and date for public computer sessions and/or WiFi sessions?

If you can provide any of the information above, please send it to the MLBN team: mlbn@measurementlab.net

Troubleshooting

In the process of the pilot phase of the project, the MLBN team learned that there are very few identical network setups in the libraries visited. Even between similar libraries, the team encountered significant variations in any issues encountered as well as the solutions to overcome these difficulties.

Most common measurement computer installation issues:

Issue	Possible Solution or Troubleshooting Options
Measurement computers are connected to the network, but do not register with Balena remote management service for MLBN staff	Are the requested network ports and URLs bi-directionally open for the measurement computers?
Measurement computers connect to the network, but MLBN staff find that some tests are not able to run	Are the requested network ports bi-directionally open for the measurement computers?

A key success factor is each library working with the MLBN team in advance of shipping the Measurement Computers. For the smoothest installation at your library, please work through the steps carefully; review the information in this section of the guide, and stay in close communication with the MLBN team at milb@measurementlab.net

Section 4: Setting up the MLBN Measurement System

Section 4 Table of Contents

- Introduction: Setting up the MLBN Measurement System
  - Overview - Required Infrastructure, Services, and Knowledge
    - MLBN Measurement System Components
    - Required & optional services, accounts, hardware, and infrastructure
  - Create a Measurement System Deployment Plan
    - Where do you want to deploy the measurement system?
    - What do you want to measure?
    - What networks do you want to measure at each location?
      How many measurement devices?
    - Can the network support the measurement system?
    - What are the network parameters needed for each device?
    - Which tests do you wish to run from measurement devices?
    - Do you plan to host the Measurement System's Data Visualization Service?
    - Where do you want measurement tests to be saved?
  - Setup Measurement System Infrastructure
    - Setup Your Development Computer(s)
    - Setup a Test Unit and Alternate Operating System SD Card
      - Obtaining MAC Addresses Using an Ubuntu Testing Device
    - Setup a GitHub Account
      - Make a Copy or "Fork" of the Murakami Code
      - Pull Your Copy of the Code to Your Computer
    - Review Options for Data Storage Location(s) of Test Data
    - Set All Application Settings
    - Setup the Measurement System Data Visualization Service (optional)
    - Setup a Balena Cloud Account & Application
    - Configuring Defaults for the Balena Cloud Application
    - Prepare to Setup Measurement Devices
    - Setup Your Measurement Devices
      - Setup New Device Using the Balena Cloud Website
      - Setup New Device Using the Provided Setup Script
      - Special Configurations for Specific Network Conditions
        - Special Configuration for WiFi Connected Devices

        - Special Configuration to Support Custom DNS Servers
        - Configure a Static IP Address for Ethernet Connected Device
        - Configure a Static IP Address for WiFi Connected Device
    - Setup New Device Manually using the Balena CLI
    - Push a Murakami Release to Balena Cloud
        - Prepare and Push a Release
    - Deploy Measurement Devices in Intended Locations
    - Troubleshooting Tips
- Accessing and Using Test Data
    - Using the Optional Data Visualization Service
    - Accessing Data stored in remote locations defined by Murakami GCS or SCP exporters
    - Accessing data on each device
        - Accessing data via the Balena Dashboard
        - Accessing data locally via the device's SD card
    - Converting data from JSON to CSV

Introduction: Setting up the MLBN Measurement System

This section of the training manual documents how to setup, configure, and administer the various components of the MLBN Measurement System. The goal of this section is to provide a guide for organizations who may wish to implement the system and use it to measure broadband performance in the same way that the MLBN research project did for the IMLS funded program that helped develop it.

Overview - Required Infrastructure, Services, and Knowledge

While setting up and running the MLBN Measurement System will require some familiarity with computer networking concepts, running commands in a terminal, and some system administration, **planning and organizing how you wish the measurement system to work for you will come first**, and will guide the technical implementation of the system. We recommend that organizations form a small team with expertise in administration, library services, and technology, and together use this guide to create a measurement system deployment plan. This same team could coordinate regularly as the system is deployed and running, and also review and analyze results. Of course, your implementation of the measurement system will need to leverage the staff and resources relevant to your situation. We hope that our guide provides you with information and resources to help you plan and implement the measurement system to suit your local needs.

MLBN Measurement System Components

The table below provides a summary of each component of the measurement system.

Automated Measurement Test Application - Murakami The application running on measurement devices is *Murakami*, and is maintained on GitHub by M-Lab. You will make a copy of the code and push it to your Balena Cloud project to install it onto measurement devices.	**Balena Cloud - IoT Device Management Service** Balena Cloud provides the overall management and provisioning of measurement computers. You will create a Balena Cloud account, create a project, connect your code to the project, download a copy of Balena OS for your devices' base OS, install the Balena cli command line tools, and use them to deploy the *Murakami* code to your measurement devices.
Measurement Computers Measurement Computers are connected to networks you wish to measure in library locations or branches. You will configure each measurement computer by installing Balena OS, configure it, register it with your Balena Cloud project, and install it in the intended location.	**Data Archiving Locations** Your Measurement Computers will be configured to store or send test results to one or more locations. Data may be stored on each measurement computer, may be sent to one or more servers using the Secure Copy Protocol (SCP), or to one or more Google Cloud Storage (GCS) locations.
	Data Visualization Service The Data Visualization Service is a web application that provides an accessible way to view, analyze and export the test data generated by the measurement computers. The service will be installed on a separate server, either inside your network, or on common cloud hosting providers.

Required & optional services, accounts, hardware, and infrastructure

In this section, we outline the services, accounts, and infrastructure required to run the MLBN measurement system. There are some optional components as well, which allow you to plan and implement the system to suit your needs. This section provides an overview of these options to consider as you plan your implementation. Detailed setup instructions are provided in the section, Setup Measurement System Infrastructure.

Balena Cloud - device management and configuration

Balena Cloud is used to manage your measurement computers, provide overall and individual device configurations, and enable remote access and monitoring of all measurement computers you deploy. We also will use Balena's services to build and push code releases to all devices in your system.

GitHub - code and application deployment management

The code used on the measurement computers is called Murakami, and is maintained by Measurement Lab (M-Lab) using the code hosting platform, GitHub. A GitHub account is required to deploy the MLBN Measurement System code to your measurement computers.

Measurement Computer Hardware

The MLBN Measurement System currently uses small, single board computers to run regular, automatic tests from the locations you wish to measure. While other computer system architectures will be supported in the future by this system, we focus here on the hardware used in the initial MLBN program research.

For each measurement computer, you will need the following:

- Odroid XU-4
- 5V/4A Power Supply
- Case
- RTC Battery
- WiFi Module 0 802.11b/g/n (optional - needed to connect directly to WiFi networks)
- Ethernet cable
- Class 10 SD card, 16GB or larger capacity

The links above are to Ameridroid.com, a US distributor for many single board computer manufacturers, including the Odroid manufacturer, HardKernel. Ameridroid will assemble each unit for an additional fee, which is desirable if you are ordering a large number of units.

The Odroid XU-4 was selected because its ethernet interface has been benchmarked to perform up to 1Gbps upload and download speed, and has sufficient processing power and memory to run effectively.

The system architecture of the Odroid is *armv7hf*. If you wish to evaluate other measurement computers, other *armv7hf* systems would work fine with the current code, as long as they are supported by Balena Cloud. Support for other system architectures will be added in the future.

Computer & Helpful Accessories for Setup and Configuration

To set up each measurement computer you will need a desktop or laptop computer to install Balena OS on the SD card and configure it. The MLBN team has used Ubuntu Linux and Mac OS to perform these tasks, however Windows computers should also work just fine. To install or "flash" Balena OS images onto SD cards, we recommend using *Etcher*, a widely used and cross-platform application that is also provided by Balena. You will also need at least one SD card reader. Two example models of SD card readers are shown below.

If you have a large number of measurement computers to set up, we recommend using a label maker to mark each device, so that it can be easily identified over the course of your measurement program. This might include the device name, IP address, MAC address, and/or contact information.

We recommend having a separate monitor and keyboard, and at least one test device with Ubuntu Linux installed on its SD card. Temporarily booting up devices you plan to ship and issuing a single command is a quick method of gathering information from the measurement computers (such as MAC address). We'll cover this process later in this guide.

SCP Server and/or Google Cloud Storage for Test Data Archive (optional)

Tests run from your measurement computers are saved in JSON format, and can be archived in several ways. Test data can be saved on each measurement device, uploaded to a server that you maintain using the Secure Copy Protocol (SCP), or uploaded to a Google Cloud Storage (GCS) bucket. One or more of these "exporter" methods may be used in your measurement system deployment. If you decide to run the optional Data Visualization web service, you can also configure the HTTP exporter to send data to that service's database.

An SCP server could be on-premise at your library, provided by your library system, or hosted in various cloud hosting services, but all of these options require your staff to maintain the SCP server. You may also choose to use this same server to host the measurement system's data visualization service (see next section). You will need a user account and SSH public key for this SCP server, to allow "passwordless" login. The SSH key will be packaged with your code, allowing configured measurement devices to upload test data without being prompted for a password.

If you would like to use a GCS bucket for your test data archive, you will need a Google Cloud Project with billing enabled, a cloud storage bucket, and a service account with an exported key file. The service account key file will be packaged with your code, allowing each measurement computer in your fleet to upload test results to your GCS bucket.

Server or Virtual Machine to Host Data Visualization Service (optional)

The accompanying data visualization service is a web application called *Murakami Viz*, that receives data from your measurement computers, and provides an accessible way to access the data, interact with it, and export it. The visualization service can be installed on the same machine as the SCP Server you may have selected for your test data archive, or it could be installed on a cloud VM or other hosting provider. *Murakami Viz* is deployed using Docker Compose.

Create a Measurement System Deployment Plan

To deploy and run the MLBN Broadband Measurement System, we recommend creating a deployment plan. This section of the manual provides a series of questions that your team should answer concerning which locations and networks where measurement devices will be connected, which tests you wish to run, where you would like to store collected data, and what optional components of the system you wish to run. Throughout this section, we will reference a spreadsheet called the *Measurement System Deployment Plan - Planning Log*, that we have found useful in deploying and running our system for the MLBN research program. Below we present a series of questions and options, which will then be used to configure and deploy your measurement system.

1. Where do you want to deploy the measurement system?

	One branch or independent library
	Multiple branches or libraries

Next add details for each library location in the “Locations/Branches” tab in your copy of the Measurement System Deployment Plan - Planning Log spreadsheet template.

2. What do you want to measure?

If you want to measure the network capacity that an individual patron experiences, select #1 below. You will connect measurement computers either to an available ethernet jack on the desired network, or directly via WiFi in an area of the library near where patrons typically connect with their own laptops, phones, or tablets.

However, if you want to measure the total capacity of your wired and wireless networks to serve multiple patrons, select #2 below. You will connect measurement devices at the main switch nearest where Internet service enters and exits the building, and potentially other locations depending on the networks you wish to discretely measure.

You could also decide to measure both individual patron experience and the total capacity of wired or wireless networks.

	#1 - Measure the individual patron experience using the network
	#2 - Measure the total network capacity of the network to serve many patrons

3. What networks do you want to measure at each location? How many measurement devices?

Depending on what you wish to measure, you may have multiple networks at a location used for different needs. In this section, you'll inventory which networks you wish to measure, at each location, and note the number of devices you'll need. The table below is a shortened example, but you can use your copy of the Measurement System Deployment Plan - Planning Log spreadsheet template.

		For WiFi Networks:			*For all Networks:*	
Location / Branch	**Network Type**	**SSID**	**Password**	**ToS / Splash Page can be disabled?**	**DHCP or Static IP address? (If static, please provide the IP)**	**Static IP (if applicable)**
Branch 1	Public WiFi AP	Not Applicable	Not Applicable	Not Applicable	Static IP	172.16.0.30
Branch 1	Egress Wired	Not Applicable	Not Applicable	Not Applicable	Static IP	172.16.1.50
Branch 2	Public WiFi	library-wifi	welcome!	Yes	DHCP	
Branch 2	Public Wired	Not Applicable	Not Applicable	Not Applicable	DHCP	

In this example, we have two branches in the deployment plan. At Branch 1 we will measure the overall capacity of a WiFi Access Point to serve multiple patrons (Public WiFi AP) and the overall capacity of the network at the branch (Egress Wired). At Branch 2 we will measure the individual patron experience using WiFi (Public WiFi) and the individual patron experience of using a public computer provided by the library (Public Wired). Your deployment plan should reflect the needs of your library, system or consortium. The spreadsheet template will provide additional options for the type of networks and information about how each network expects measurement devices to be configured. This list will also determine how many total measurement devices will need to be purchased. We'll refer to this template throughout this guide.

4. Can the network support the measurement system?

WiFi Networks

Some WiFi networks may not support the MLBN measurement system. The most common reasons we have identified for this are:

- The WiFi network requires a person to read and complete a consent or terms of service form, **AND** the page cannot be bypassed for the measurement devices.

  Library WiFi networks often are configured to require people to view a terms of service or policy page when they first connect to WiFi, and click a button to consent to those terms. If a WiFi network uses this technology, sometimes referred to as a "Splash Page" or "Captive Portal", it must be disabled for the individual measurement devices you plan to use. Typically this is a configuration in the system your library uses to manage the "Captive Portal" service.

- The WiFi network configuration is not compatible with the measurement device's operating system or WiFi card.

  In a very limited number of cases during the MLBN research program, the measurement device was unable to connect via WiFi due to unknown issues. This could have been caused by misconfiguration of the device itself, or on unknown or undiagnosed network conditions or configurations.

- The WiFi network blocks ports needed by the measurement software.

  If any of the following ports or URLs are blocked by the network firewall or filtering service, some tests provided by the measurement system will not be able to run. Additionally, some features of the MLBN measurement system may not function correctly if these ports or URLs are blocked or filtered. Your network administrator will need to add rules to allow bidirectional use of these ports, and unblock any URLs below, using the MAC address for each measurement computer.

3001-3010 TCP, 32768-65535 TCP	Required by NDT 5 test
80 TCP, 443 TCP	Required by NDT 7, speedtest-cli, Balena Cloud VPN
22 TCP	Required by SCP test data exporter
123 UDP, 53 UDP	Required by Balena OS for network time sync and DNS queries

*.measurement-lab.org, *.balena-cloud.com, *.docker.com, *.docker.io	Required by M-Lab tests, Balena services

Suggestions for WiFi Network Measurement

If your network supports it, to measure the individual patron experience using the WiFi network (option 1 in question 2 above):

- One measurement device connected via WiFi, with any terms of service page requirement removed. In this guide we will refer to this example as “public-wifi”.

Wired Networks

There are fewer potential pitfalls to supporting the MLBN measurement devices when they are connected via ethernet. Many libraries have different virtual networks or VLANs to serve staff, the public, WiFi, etc. Each ethernet jack should be active and configured for the network you intend to measure. Additionally, the network ports described above should be allowed for the measurement devices you plan to connect.

Suggestions for Wired Network Measurement

If you wish to measure the total network capacity of the network to serve many patrons (option 2 in question 2 above):

- To measure the total capacity of the library or branch:
  One measurement device connected to the switch nearest where Internet service enters/exits the facility. In this guide we will refer to this example as “wired-egress”.
- To measure the total capacity of one WiFi access point to serve multiple patrons:
  One measurement device connected to an ethernet jack *that would otherwise connect a new WiFi access point.* In this guide we will refer to this example as “wifi-AP”.

If you wish to measure the individual patron experience using the network (option 1 in question 2 above):

- To measure the patron experience using public computers provided by the library:
  One measurement device connected to an ethernet jack that would otherwise connect a library-provided computer. In this guide we will refer to this example as “public-wired”.

5. What are the network parameters needed for each device?

You've identified how you wish to measure (overall capacity, individual patron experience, or a combination of both), the network(s) where you will connect measurement devices, and should have a sense of the total number of devices you need.

Next, you will need to gather information needed to configure each measurement device so they will work as expected when you connect them. The table below describes the network details you'll need for each device with examples. You can use your copy of the Measurement System Deployment Plan - Planning Log spreadsheet template to log this information for your deployment. If the device needs a static IP address, enter the IP, subnet, and gateway details. If the device will receive an IP dynamically, enter "DHCP" for these columns. For all devices that require a static IP and/or non-Google DNS servers, please see the section later in this document, Special Configurations for Specific Network Conditions.

		For WiFi Networks:			***For all Networks:***	
Location / Branch	Network Type	**SSID**	**Password**	**ToS / Splash Page can be disabled?**	**DHCP or Static IP address? (If static, please provide the IP)**	**Static IP (if applicable)**
test library 1	Public WiFi	library-wifi	welcome!	Yes	DHCP	
test library 1	Public Wired			Not Applicable	Static IP	172.16.0.30
test library 2	Public WiFi AP			Not Applicable	Static IP	10.22.8.199
test library 2	Egress Wired			Not Applicable	Static IP	10.22.8.200

For WiFi networks, you will need to provide the exact WiFI network name or SSID, the password if required, and any Terms of Service or splash page must be disabled.

6. Which tests do you wish to run from measurement devices?

Next, you'll decide which of the available tests you wish to run from your measurement devices. You can choose to use all tests on all devices, select a subset of the available tests to run on all devices, or run some tests only on some devices but not others. The available tests are NDT5, NDT7, Speedtest-CLI Single-stream, and Speedtest-CLI Multi-stream. To learn more about each test and what it measures, please review Appendix B.

For each device you add in your copy of the Measurement System Deployment Plan - Planning Log spreadsheet template, indicate which tests to use in columns N - Q.

7. Do you plan to host the Measurement System's Data Visualization Service?

If you plan to host the data visualization service, you will need a server or virtual machine to host it. This could be on a virtual machine or server in your preferred cloud hosting service, on a physical server in your datacenter public to the Internet, or on a virtual machine server in your local network. See the section, Setup the Measurement System Data Visualization Service, found later in this document.

If you decide to NOT host the Measurement System's Data Visualization Service, the Murakami system includes a script called *murakami-convert* that can be used to convert all test results to a file in Comma Separated Values (CSV) format. Details about how to use *murakami-convert* can be found in the section, Accessing and Using Test Data > Converting data from JSON to CSV.

8. Where do you want measurement tests to be saved?

You have multiple options for saving the test data generated by measurement devices:

- Local, on each measurement computer
- Remote, within the Measurement System's Data Visualization Service database
- Local or Remote, on an SSH server
- Remote, in a Google Cloud Storage (GCS) Bucket

These options are called "exporters" in the Measurement System, and will be configured using Environment Variables in your Balena Cloud project. If some measurement devices need to send data to different locations than others, you can override the default exporter-related environment variables for these devices.

You can use one or all of these options at the same time, for example keeping one copy on each device, sending one copy to your own SSH server, another copy to a GCS Bucket, and/or another copy to another GCS Bucket or SSH server.

If you plan to host the Measurement System's Data Visualization Service, you will at least need to configure the HTTP exporter after setting up the service on your server or VM, as described in this section of the manual.

Setup Measurement System Infrastructure

Setup Your Development Computer(s)

The computer(s) you will use to interact with the Balena Cloud and GitHub services need to have a few things installed and working. It's beyond the scope of this guide to instruct you on how to install everything needed for your operating system. The list of requirements is below, along with links to documentation to get you started.

- The Git program or application for your operating system
- An SSH key to securely copy code from GitHub, and push code releases to Balena Cloud - GitHub Guide to SSH Keys
- The Balena CLI (Command Line Interface) to interact with the balena API

These programs may also have additional dependencies which should be installed as a part of following their installation documentation.

Setup a Test Unit and Alternate Operating System SD Card

Even if your measurement system deployment will only have a small number of measurement devices, we recommend setting aside at least one testing device that can be used to gather important information about the units you will deploy in the field, as well as to troubleshoot potential issues.

Setup a testing workstation, with a spare monitor, keyboard, measurement device, and SD card. When you set up your measurement devices, you'll flash their SD cards with Balena OS. But you'll want to gather the MAC addresses of each device's network interfaces to enable easy identification by remote staff, and to be used by IT staff to set firewall rules, for example to allow a WiFi device to bypass the terms of service page, and more.

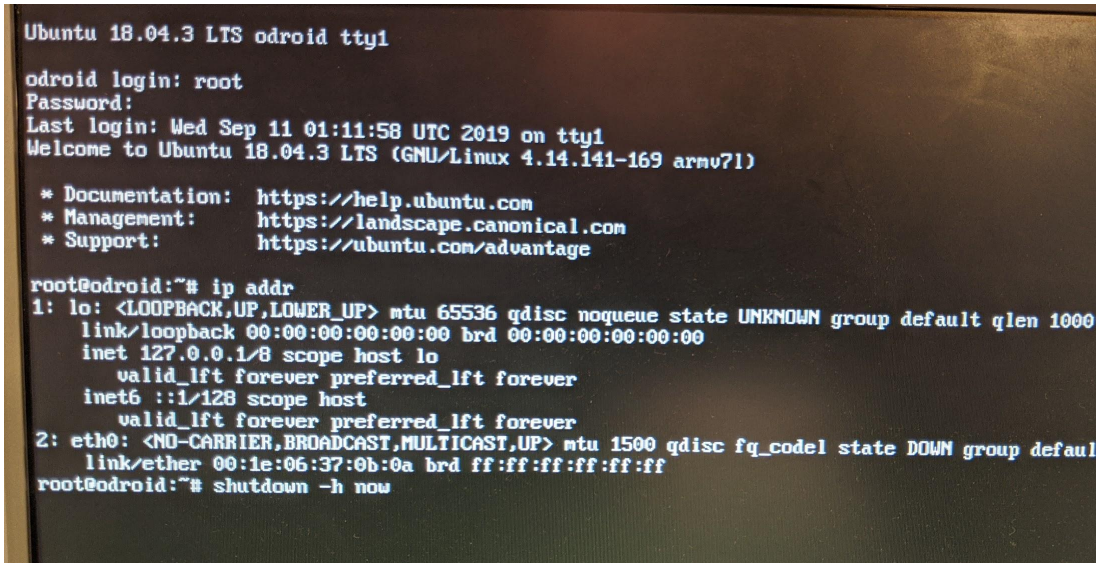
We recommend installing the Ubuntu Linux operating system for your measurement device model, to be used to obtain MAC address information for each device in your system. For example the MLBN program used the ODROID XU-4 model as a measurement device, and installed the Ubuntu 18.04 OS image provided on the Odroid wiki. If you are using a different model device, ensure that your testing device(s) match the model(s) you will deploy in the field, and use an Ubuntu OS image appropriate for your device model(s).

Obtaining MAC Addresses Using an Ubuntu Testing Device

Once you startup your Ubuntu testing device, login using the default username and password (for images downloaded from the ODROID wiki, that is usually: username: **root**
password: **odroid**

We used the “minimal” or “server” image, and will only display images here of the command line terminal. If you are using an Ubuntu image with a GUI desktop environment, just open a terminal window to follow along.

Once you’re logged in at a terminal, you should see something like:



In the image above, we’ve first logged in as “root”, and then typed “ip addr” to get information about the network interfaces on the device. In this example, we have two interfaces:

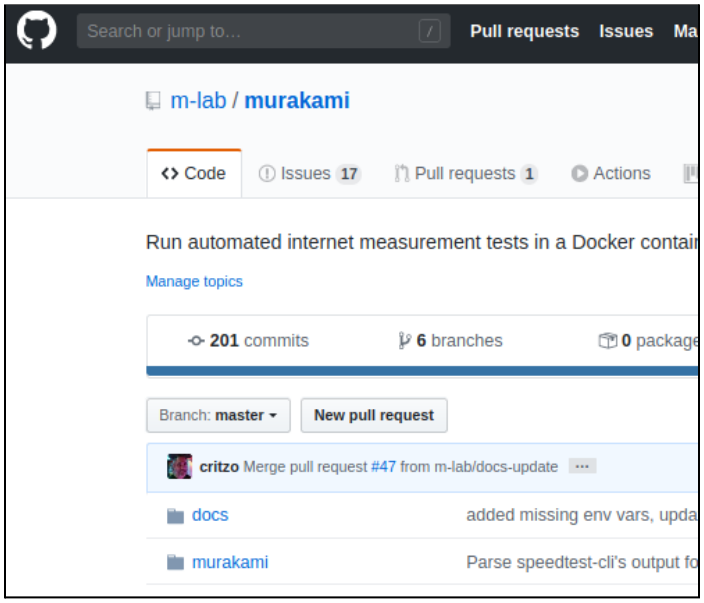
1: lo: …
2: eth0: …

The interface “lo” stands for local or loopback, and will be present on all computers. Our test device only has one ethernet port, so “eth0” is what we’re looking for. If a device also had a WiFi card, a third interface would be listed, “wlan0”. The device’s MAC address is shown after the text “link/ether”, for example here we’ve highlighted the MAC address in gray: link/ether `00:1e:06:37:0b:0a`

The final command in the image above shows how to shut down your test device: `shutdown -h now`

Later in this guide when we set up our measurement devices, you’ll use your test device’s SD card to obtain the MAC addresses of each measurement device you plan to deploy.

Setup a GitHub Account



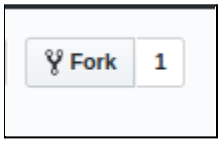
You'll need an account on the popular code version control web service, GitHub, to obtain a copy of the code for your measurement devices, to connect your copy to your Balena Cloud project, and you'll use Git commands to deploy releases of your application.

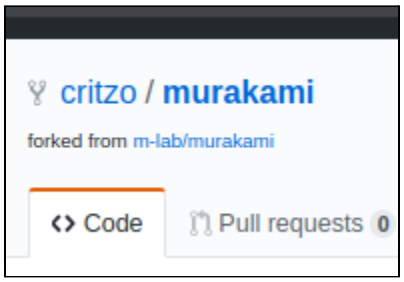
Visit: https://github.com/ and sign up for an account, or use an existing account if you already have one.

Once you have a GitHub account and have logged in, visit the M-Lab code repository for *Murakami*: https://github.com/m-lab/murakami

Make a Copy or "Fork" of the Murakami Code

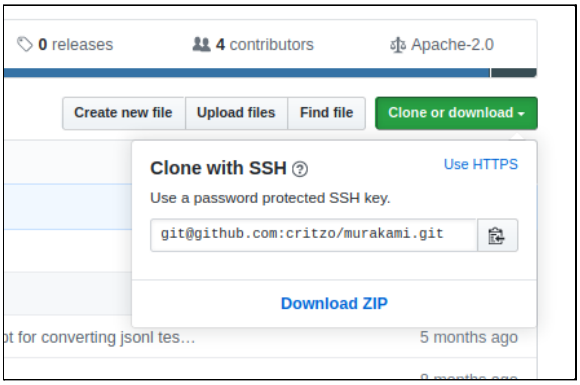
You'll next create a copy of the *Murakami* code repository in your account, by clicking the "Fork" icon in the top right of the page (see images on right). If you use your GitHub account a lot, you may be part of multiple organizations. If so, select the organization where the fork or copy of the code should be made. Once complete, your fork should appear on screen. You can tell it's a fork by looking in the top left part of the page.





Pull Your Copy of the Code to Your Computer

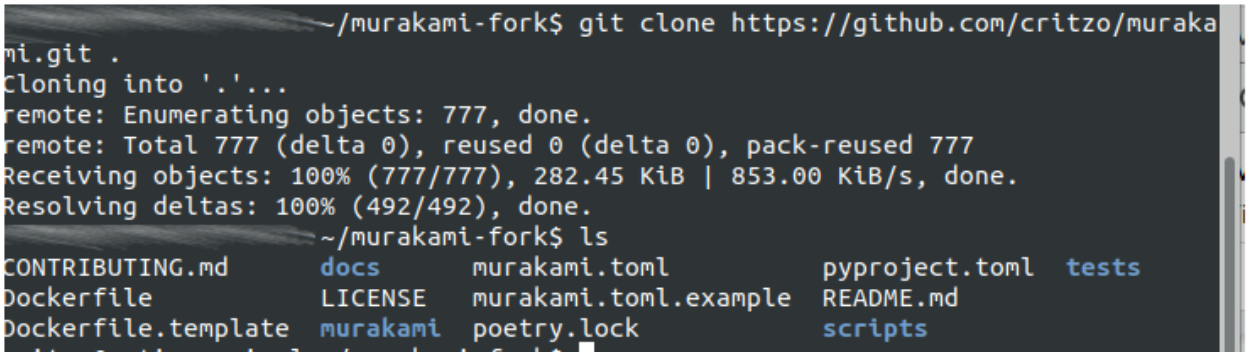
Next, you need to pull a copy of the forked code to your local computer.



Click on the button "Clone or download" and copy the URL listed under the "Clone with SSH" option. If you don't have an SSH key setup on your GitHub account, please review the GitHub Guide to SSH Keys, create your keys, and connect them to your GitHub account.

Next, open a terminal or your Git client program on your computer, and pull a copy of your *Murakami* fork from GitHub. In the screenshot below, you can see that we've created a folder called "murakami-fork" and are cloning

the repository inside it, then listing its contents.



We'll come back to your local copy of the code after setting up the Balena Cloud application, but at this point, you can do a couple of things to stay organized and prepare.

1. Create a folder called "keys". If you use the SCP or GCS exporters, you'll add a file for each to the "keys" folder before pushing a release to your Balena Project.
2. Make a copy of the file, ***env-vars-example.txt***:
   `cp env-vars-example.txt env-vars.txt`

You will update env-vars.txt in the following sections of this guide to define variables that will be the same for ***all measurement devices*** in your deployment, such as where test data is exported and which tests are enabled by default. These variables can be overridden for individual measurement devices. Some environment variables will be specific to each device. We'll cover those later when we set up devices.

Review Options for Data Storage Location(s) of Test Data

You have several options for saving the data generated by tests running on your measurement devices:

- Local on each device
- SSH server
- Google Cloud Storage
- Visualization Service database

Instructions and requirements for setting up an SSH server will be specific to your network. Similarly, using Google Cloud Storage to export data involves setting up a project and other resources in a Google Cloud Project. We recommend consulting with your network administrator or IT support staff if these options are desired.

In the next section you will define the storage options you wish to use in a text file, which you will later use to define environment variables in your Balena Cloud project. Open the file "**env-vars.txt**" that you created in the previous section, and update the environment variable values appropriate for your system. The available environment variables, options, and functions are described in detail in the *Murakami* project README. Below we'll walk through each environment variable option as well.

We'll next look at the contents of "env-vars-example.txt" one section at a time, and set the variables that will be common to all measurement devices in your Measurement System Deployment Plan. Keep in mind that the values we set here will become the defaults for all measurement devices in your deployment. Later, when we set up each new measurement device, we'll set some additional variables that are device-specific.

Set All Application Settings

Overall Application Settings

```
MURAKAMI_SETTINGS_PORT=80
MURAKAMI_SETTINGS_WEBTHINGS=0
MURAKAMI_SETTINGS_LOGLEVEL=DEBUG
MURAKAMI_SETTINGS_IMMEDIATE=1
```

The first set of variables are required and define some application settings that you will not need to change. *Murakami* has experimental support for basic control of which tests run on each device using a Mozilla WebThings Gateway. We recommend disabling this option (as seen above: `MURAKAMI_SETTINGS_WEBTHINGS=0`), since you'll be managing devices using the Balena Cloud application. If it were enabled, Murakami devices would advertise a web service over port 80.

We also set the default LOGLEVEL to DEBUG, so that your measurement devices will provide sufficient log messages so we can know what's happening with each device through the Balena Cloud remote console. Lastly, `MURAKAMI_SETTINGS_IMMEDIATE=1` instructs each device to run it's first set of tests immediately when the device starts up. If you'd rather that your devices always generate a random time for their first test after startup, instead of testing immediately, set this value to zero: `MURAKAMI_SETTINGS_IMMEDIATE=0`

Test Runner Settings

By default, all available test runners are enabled. If you only wish to enable some supported test runners, and not all, you must declare environment variables for each available runner. For example, if you only wish to run the NDT 7 and speedtest-cli multi-stream tests, the following variables would need to be defined:

MURAKAMI_TESTS_NDT5_ENABLED=0
MURAKAMI_TESTS_NDT7_ENABLED=1
MURAKAMI_TESTS_SPEEDTESTMULTI_ENABLED=1
MURAKAMI_TESTS_SPEEDTESTSINGLE_ENABLED=0

Exporter Settings

In the example environment variables file, we define each of the exporter types. You can define multiple SCP, GCS, and HTTP exporters, for example if you wish to send one copy of all test results to a pristine data archive using SCP or GCS, and a second copy to the Data Visualization Service using the HTTP exporter. Below are the contents of the example environment variables file:

```
MURAKAMI_EXPORTERS_LOCAL_TYPE=local
MURAKAMI_EXPORTERS_LOCAL_ENABLED=true
MURAKAMI_EXPORTERS_LOCAL_PATH=/data/
MURAKAMI_EXPORTERS_SCP_TYPE=scp
MURAKAMI_EXPORTERS_SCP_ENABLED=true
MURAKAMI_EXPORTERS_SCP_TARGET=<server url or ip address>:<folder>/
MURAKAMI_EXPORTERS_SCP_PORT=22
MURAKAMI_EXPORTERS_SCP_USERNAME=<username>
MURAKAMI_EXPORTERS_SCP_PRIVATE_KEY=/murakami/keys/<ssh keyfile>
MURAKAMI_EXPORTERS_GCS_TYPE=gcs
MURAKAMI_EXPORTERS_GCS_ENABLED=true
MURAKAMI_EXPORTERS_GCS_TARGET=gs://bucket/name/path/
MURAKAMI_EXPORTERS_GCS_SERVICE_ACCOUNT=<name@gcs-project.iam.gserviceaccount.com>
MURAKAMI_EXPORTERS_GCS_KEY=/murakami/keys/<gcs-serviceaccount>.json
MURAKAMI_EXPORTERS_HTTP0_ENABLED=1
```

```
MURAKAMI_EXPORTERS_HTTP0_TYPE=http
MURAKAMI_EXPORTERS_HTTP0_URL=<domain-name-murakami-viz-service>/api/v1/runs
```

Local Exporter

To use the "Local", on-device storage exporter, no changes are needed to the variables below. We set the LOCAL_PATH variable to "/data/" because Balena provides this special, persistent storage volume for each measurement device. If you wish to disable local storage, set MURAKAMI_EXPORTERS_LOCAL_ENABLED to "false" or "0".

```
MURAKAMI_EXPORTERS_LOCAL_TYPE=local
MURAKAMI_EXPORTERS_LOCAL_ENABLED=true
MURAKAMI_EXPORTERS_LOCAL_PATH=/data/
```

SCP Exporter

To use the SCP exporter, you must first have a server with SSH enabled, an account on that server, and a public/private SSH key pair for that account. You will also need to know the server URL or IP address, and the port used for SSH access. The private SSH key will be distributed with each measurement device.

```
MURAKAMI_EXPORTERS_SCP_TYPE=scp
MURAKAMI_EXPORTERS_SCP_ENABLED=true
MURAKAMI_EXPORTERS_SCP_TARGET=<server url or ip address>:<folder>/
MURAKAMI_EXPORTERS_SCP_PORT=22
MURAKAMI_EXPORTERS_SCP_USERNAME=<username>
MURAKAMI_EXPORTERS_SCP_PRIVATE_KEY=/murakami/keys/<ssh keyfile>
```

In the example above, in the variable MURAKAMI_EXPORTERS_SCP_PRIVATE_KEY, we define the system path where each measurement device can find the SSH private key file: /murakami/keys/<ssh keyfile> We recommend creating a folder in your copy of the *Murakami* code called "keys" where you will save the SSH private key file before deploying a release to your Balena project.

> **Security note:** take care to not commit contents of your /keys directory to Github. Exposing these keys publicly would allow anyone who obtains the key files to write or overwrite data on your SCP server.

You may define multiple SCP exporters if desired. To do so, simply add a copy of the variables above and modify them to define a second SCP location. In the example below, we've simply added "2" to the SCP exporter variable names to define the second SCP server your measurement devices should use.

```
MURAKAMI_EXPORTERS_SCP_TYPE=scp
MURAKAMI_EXPORTERS_SCP_ENABLED=true
MURAKAMI_EXPORTERS_SCP_TARGET=<first SCP server url or ip address>:<folder>/
MURAKAMI_EXPORTERS_SCP_PORT=22
MURAKAMI_EXPORTERS_SCP_USERNAME=<username>
MURAKAMI_EXPORTERS_SCP_PRIVATE_KEY=/murakami/keys/<ssh keyfile>
MURAKAMI_EXPORTERS_SCP2_TYPE=scp
MURAKAMI_EXPORTERS_SCP2_ENABLED=true
MURAKAMI_EXPORTERS_SCP2_TARGET=<second SCP server url or ip address>:<folder>/
MURAKAMI_EXPORTERS_SCP2_PORT=22
MURAKAMI_EXPORTERS_SCP2_USERNAME=<username>
MURAKAMI_EXPORTERS_SCP2_PRIVATE_KEY=/murakami/keys/<ssh keyfile>
```

GCS Exporter

```
MURAKAMI_EXPORTERS_GCS_TYPE=gcs
MURAKAMI_EXPORTERS_GCS_ENABLED=true
MURAKAMI_EXPORTERS_GCS_TARGET=gs://bucket/name/path/
MURAKAMI_EXPORTERS_GCS_SERVICE_ACCOUNT=<name@gcs-project.iam.gserviceaccount.com>
MURAKAMI_EXPORTERS_GCS_KEY=/murakami/keys/<gcs-serviceaccount>.json
```

Google Cloud Storage (GCS) may also be used to export measurement device data. The following is required in order to use the GCS exporter:

- Google Cloud Project with a billing account
- GCS Bucket within your project
- Service account with permissions to write to the GCS bucket
- Service account key file in JSON format, downloaded from your GCP project

In the example above, we set MURAKAMI_EXPORTERS_GCS_TARGET to your GCS bucket. Your GCS Service Account name looks like an email address, in the variable MURAKAMI_EXPORTERS_GCS_SERVICE_ACCOUNT. Finally, the variable MURAKAMI_EXPORTERS_GCS_KEY, should contain the system path to your JSON key file that is distributed to each measurement device.

We recommend creating a folder in your copy of the *Murakami* code called “keys” where you will save your GCS Service Account’s JSON key file before deploying a release to your Balena project. We’ll walk through how to deploy the GCS key with your first release when we setup a new device in the following section.

Security note: take care to not commit contents of your /keys directory to Github. Exposing these keys publicly would allow anyone who obtains the key files to write or overwrite data in your GCS bucket.

You may define multiple GCS exporters if desired. To do so, simply add a copy of the variables above and modify them to define a second GCS location. In the example below, we've simply added "2" to the GCS exporter variable names to define the second GCS bucket where your measurement devices should push test results.

```
MURAKAMI_EXPORTERS_GCS_TYPE=gcs
MURAKAMI_EXPORTERS_GCS_ENABLED=true
MURAKAMI_EXPORTERS_GCS_TARGET=gs://bucket/name/path/
MURAKAMI_EXPORTERS_GCS_SERVICE_ACCOUNT=<name@gcs-project.iam.gserviceaccount.com>
MURAKAMI_EXPORTERS_GCS_KEY=/murakami/keys/<gcs-serviceaccount>.json
MURAKAMI_EXPORTERS_GCS2_TYPE=gcs
MURAKAMI_EXPORTERS_GCS2_ENABLED=true
MURAKAMI_EXPORTERS_GCS2_TARGET=gs://bucket2/name2/path2/
MURAKAMI_EXPORTERS_GCS2_SERVICE_ACCOUNT=<name2@gcs-project.iam.gserviceaccount.com>
MURAKAMI_EXPORTERS_GCS2_KEY=/murakami/keys/<gcs-serviceaccount2>.json
```

HTTP Exporter

The HTTP exporter pushes test results to the visualization service that comes with the MLBN measurement system. To use this exporter, you will need to first setup and configure the Murakami Data Visualization Service, as outlined in the next section.

The HTTP exporter is enabled and configured using three environment variables:

```
MURAKAMI_EXPORTERS_HTTP0_ENABLED=true
MURAKAMI_EXPORTERS_HTTP0_TYPE=http
MURAKAMI_EXPORTERS_HTTP0_URL=<domain-name-murakami-viz-service>/api/v1/runs
```

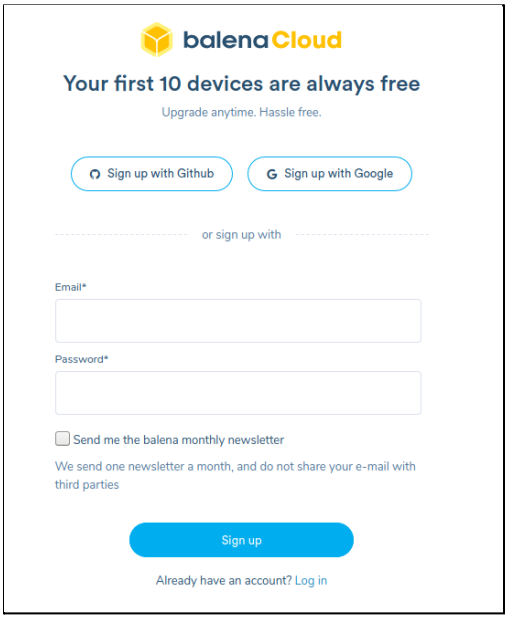
Substitute the domain name where you are hosting the Murakami Viz service in the bold portion of MURAKAMI_EXPORTERS_HTTP0_URL above.

As outlined in the section, *Setup the Measurement System Data Visualization Service*, in order to accept test results from Murakami measurement devices, the Murakami Viz service requires you to whitelist the IP addresses of the networks where you have measurement devices deployed. This interaction works best when the IP address assigned to your customer premise hardware (i.e. your switch, cable or DSL

modem) is static. If you are running a Murakami measurement device from within a network that does not have a static IP, when the IP address changes new tests will be rejected by the Murakami Viz service until you add the new IP in the settings. Additional methods of authenticating a device may be added in the future.

Setup a Balena Cloud Account & Application

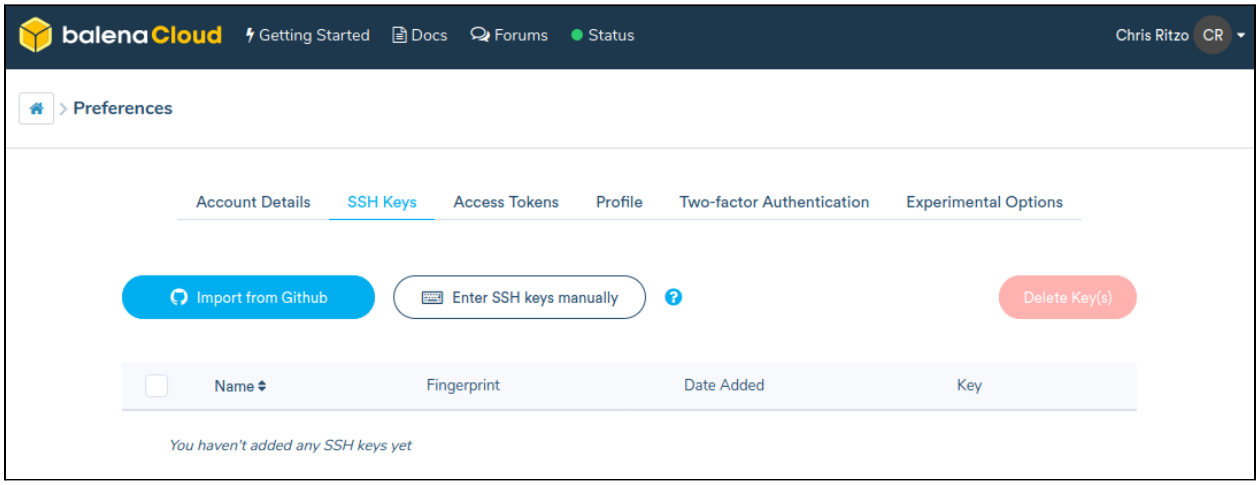
Create an account



To manage your measurement system devices, you will need a Balena Cloud account. Visit https://www.balena.io/ and click “Sign up”. You have the option to use your GitHub or Google accounts, or to sign up using your email and a new password. The image on the right shows the Balena Cloud account signup page.

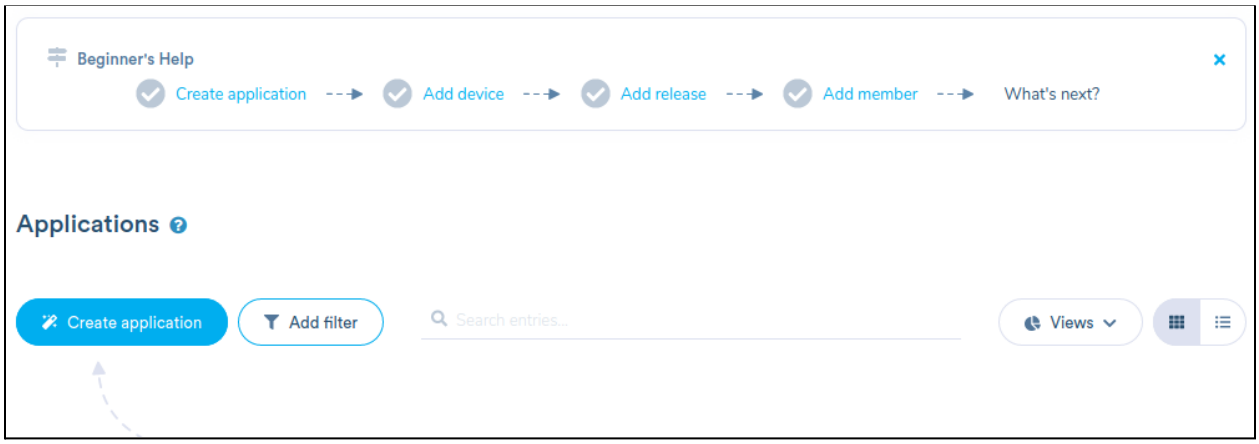
If you plan to use more than 10 measurement devices in your Measurement System deployment, you will need to purchase a service plan from Balena.

Be sure to add your SSH key to your Balena Cloud account preferences. This will ensure you can push code releases from your local computer to your application on Balena Cloud. If you’ve set up the SSH key on GitHub, you have the option to import the same SSH key from GitHub. Access the Balena account preferences by clicking your account name in the upper right corner of the page.

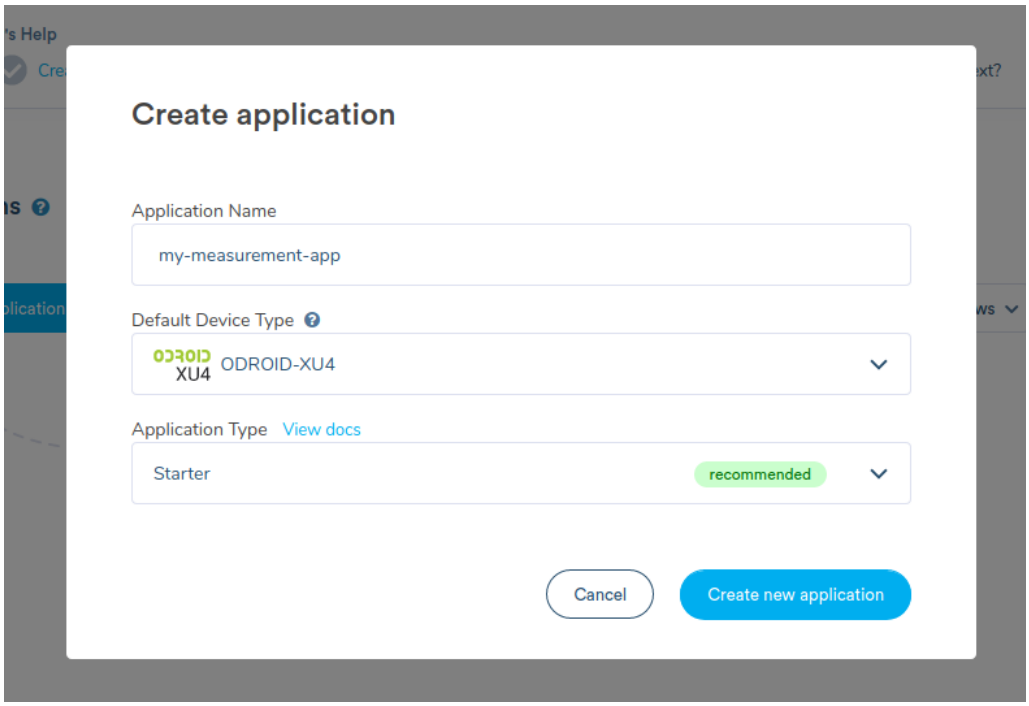


Create a Balena Cloud Application

Once you have a Balena Cloud account, you'll need to create an application. When you first login, after creating your profile, Balena prompts you to create your first application. Below is an image of Balena's "Beginner's Help" page, describing the steps involved in setting up your application on Balena Cloud.

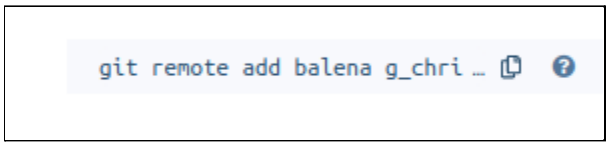


Click "Create application", add your application name, and select "ODROID-XU4" for the default device type (or a different supported device type if you are not using the Odroid-Xu4 model). If you have purchased a Balena Cloud plan, select the "Essentials" application type. Otherwise, select "Starter" as the application type. Finally, click "Create Application".

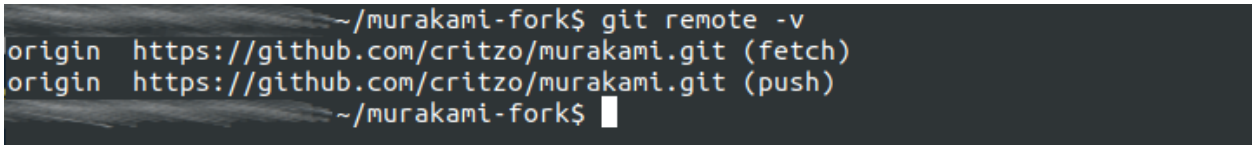


Connect Your Balena Application to Your Local Copy of Murakami

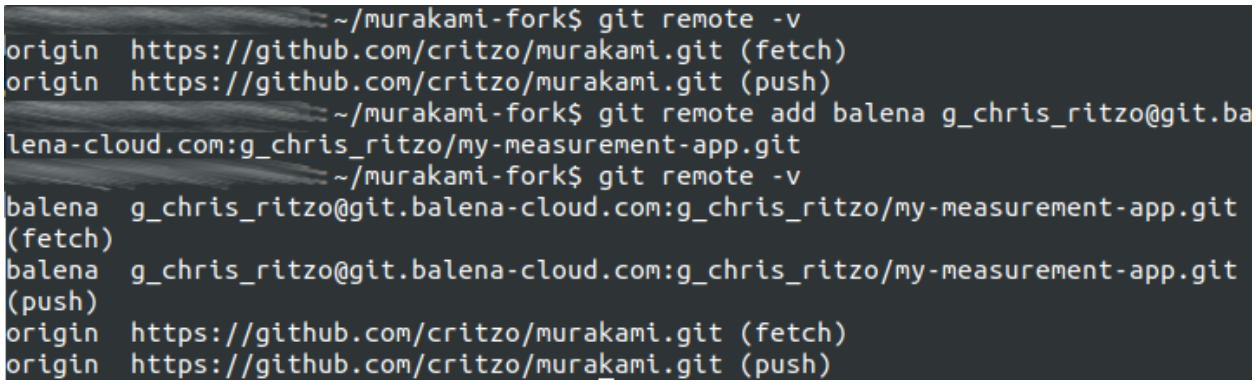
On the main page of our Balena application, you'll see a URL very similar to the GitHub URL you copied earlier when pulling the *Murakami* code to your local computer.



Copy the URL and return to your terminal or Git application. You'll now add a second "remote" for the Git repository for the Balena application. First, you might examine the existing remote using the command: `git remote -v`



Now add the new remote by typing `git remote add balena` , followed by pasting the URL you copied from your balena project above.

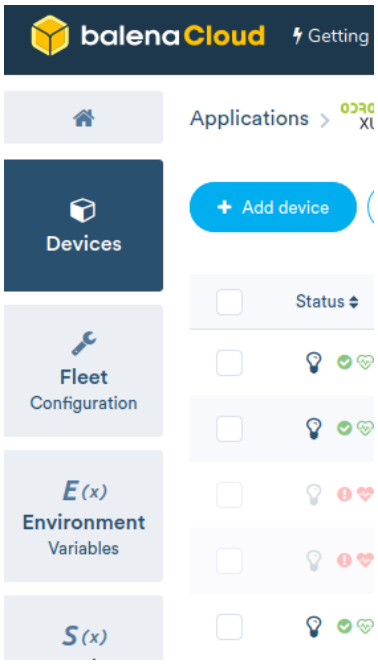


When we again type `git remote -v` there are now two "Git remotes", one called **origin** that points to your GitHub fork, and another called **balena** that points to your Balena application. Later, we'll deploy a release to your Balena application by pushing to your balena Git remote.

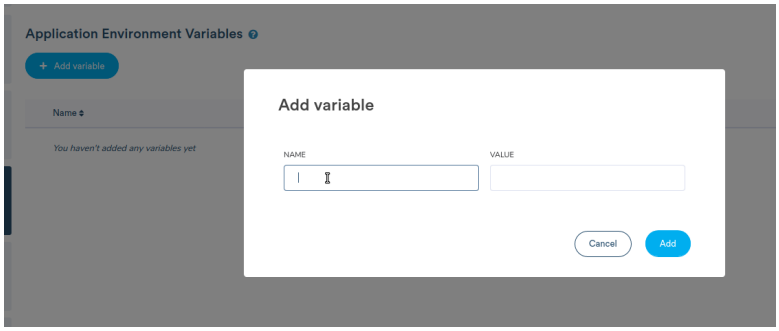
Configuring Defaults for the Balena Cloud Application

Next, you will add environment variables to the Balena application to configure the default behavior of your measurement devices. We'll create these variables in the Balena Cloud Project web console, but you can also use the balena-cli tool to set them. Read more about the Balena CLI tool.

Open your file **env-vars.txt** to refer to, and open your Balena Cloud application in your web browser. On the left side of the page, click on “Environment Variables”. In a brand new application, there should be no variables listed.



- Click on “Add variable” then copy / paste the first variable name and value from **env-vars.txt** into the form.
- Do not include the = sign.
- When done, click “Add”
- Repeat this process for your other system-wide variables



Once you’re done adding the system-wide environment variables, you’re ready to proceed with setting up your first measurement devices!

Prepare to Setup Measurement Devices

You’re now nearly ready to start setting up the measurement devices for your deployment. To recap, so far you’ve done the following:

- Created a Measurement System Deployment Plan
- Identified the locations and networks where you wish to measure, gathered the network details for each device, and decided how many measurement devices needed for your deployment
- (Optional) Setup the Measurement System's Data Visualization Service
- Determined where you wish to save measurement test data
- Setup a test measurement device with the Ubuntu Linux operating system
- Setup a GitHub account, forked/cloned the Murakami code, and have a local copy of the code on your development computer
- Setup a Balena Cloud account and application
- Setup a "git remote" for your Balena application in your local copy of Murakami
- Configured the default environment variables for your Balena application

Next, we'll walk through adding a new measurement device to your Balena application in a couple of different ways. You can use the Balena Cloud website or the Balena CLI from your development computer. In both cases, the steps to add a new measurement device to your application are:

- Identifying the MAC addresses of the measurement computer's network interface(s)
- Download Balena OS image, write it to an SD card
- Pre-provision a new device in the Balena Cloud application, name it with a recognizable name
- Generate and apply a configuration file for the device if it is needed
- Add environment variables specific to the new measurement device
- Apply special configurations required by DNS servers
- Install device in the planned location
- Confirm working status

Name the New Measurement Computer - Device Location, Network Type, and Connection Type Settings

Each measurement computer in your deployment should be named with a combination of three variables that will be used to name the device and as environment variables for the device in the Balena Cloud application. In the MLBN system, we've named these variables:

- MURAKAMI_SETTINGS_**LOCATION**
- MURAKAMI_SETTINGS_**CONNECTION_TYPE**
- MURAKAMI_SETTINGS_**NETWORK_TYPE**

In the MLBN measurement program, we used these variables to name our devices by:

- library/branch location (LOCATION) example(s): Clarkston-MI
- how the device is connected (CONNECTION_TYPE) example(s): wired, wifi, wifiAP
- where the device is connected (NETWORK_TYPE) example(s): egress, public

We recognize that these variable names might not fit your deployment scenarios, but you can use whatever text you'd like for them, to organize your device names so they make sense to you and your team.

The variable names you choose will be used:

- To name each device as they appear in your Balena Cloud app
- In the file names of each test result
- As columns in the each test result file's contents

You can use column K, Device Name, in the "Networks" tab of the Measurement System Deployment Plan to note the device name, for example: **Clarkston-MI-egress-wired**

Sharing an example from the MLBN research program, we used these values for the device measuring the egress at one library:

And an example test result file generated by that device was named:

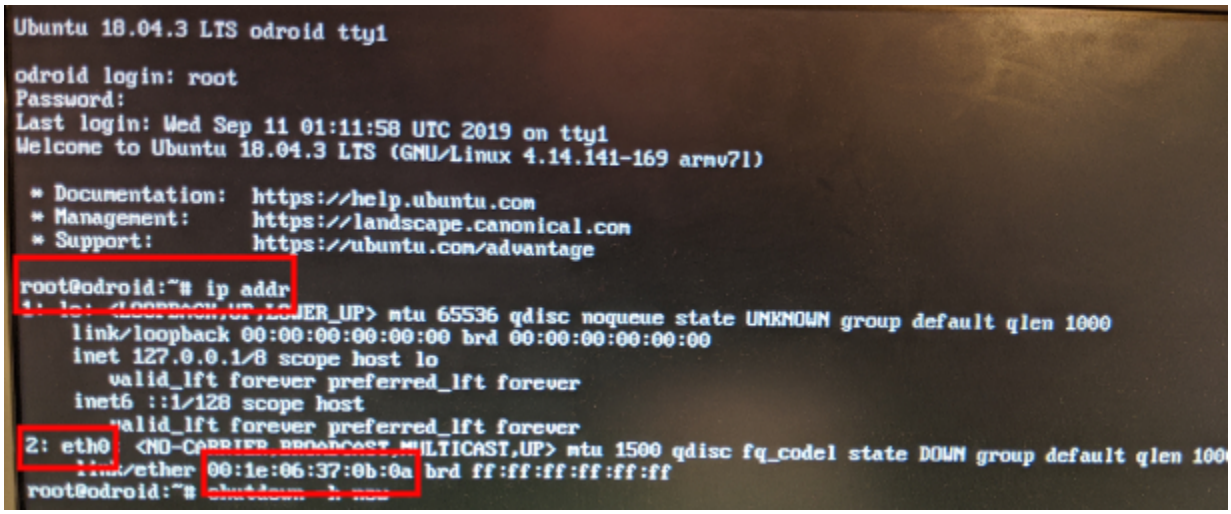
`ndt5-Clarkston-MI-egress-wired-2020-03-08T23:39:18.796318.jsonl`

Identifying the MAC addresses of measurement computers network interfaces

Identifying the media access control address (MAC address) at the time you setup a measurement computer is good practice, and will be needed by many network IT staff to receive a static IP address, to allow the computer to not be presented with a WiFi splash page, or for other reasons. In the MLBN research program, we placed the MAC address and device name on a label attached to each device before sending them out to partner libraries.

Earlier in this guide, you set up a test measurement device with Ubuntu Linux. To obtain the MAC address for each device you plan to add to your measurement system:

- Insert the SD card from your test device into the measurement computer you're about to add to the measurement system.
- Connect the device to your testing workstation's monitor and keyboard.
  You will not need to connect the device to any network at this time.
- Once the device starts up and you have logged in, use this terminal to obtain the MAC address(es) for the device:
  - In the image below you'll see the command, `ip addr`, highlighted in red,
  - Followed by the wired ethernet interface, `2: eth0`,
    and it's MAC address, `00:1e:06:37:0b:0a`
  - If you have a device with a WiFi card as well as an ethernet interface, it will likely be labelled `wlan0`



Make a note of the MAC address(es) for this device in your Measurement System Deployment Plan, in columns K or L in the "Networks" sheet.

The final command in the image above shows how to shut down your test device: `shutdown -h now`

Label the Device

At this point you should know the device's name and the MAC addresses of it's network interfaces. If you're labelling your measurement devices, now is a good time to do that. In the MLBN, program we printed a label with the device name and the device's MAC address to be used for measurement.

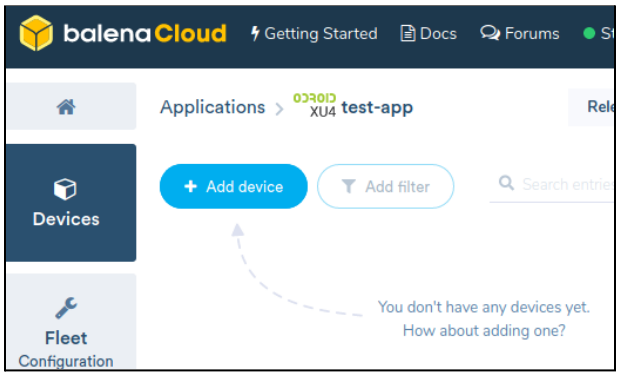
If you're preparing multiple measurement devices, having the labels on the target devices at this point will help keep your work organized as you prepare the SD cards and configurations for each device.

Setup Your Measurement Devices

There are two ways to set up a new measurement device in your system:

- through the Balena Cloud website
- manual setup using the Balena CLI from your development computer

Setup New Device Using the Balena Cloud Website



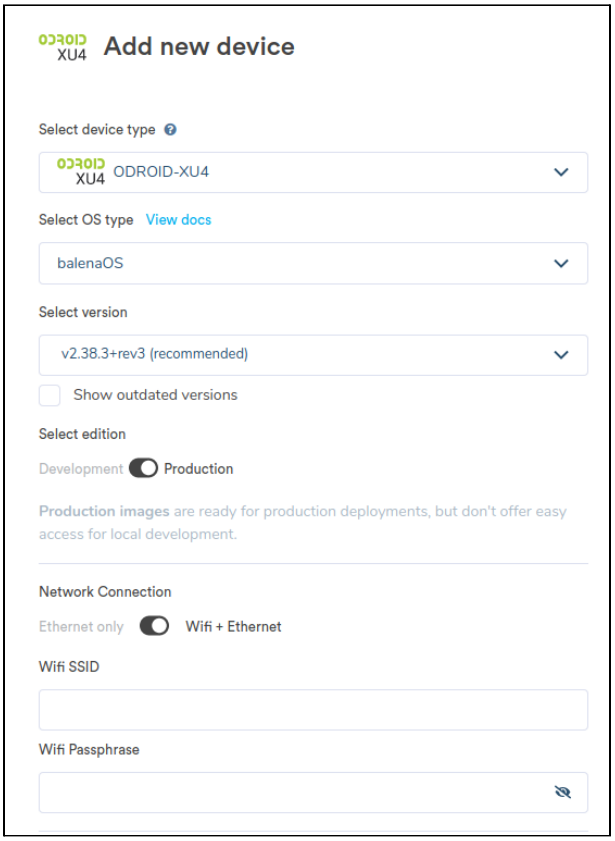
For simple deployments, or for deployments with a small number of devices, you can use the Balena Cloud website to add devices.

"Simple" deployments are those where:

- Devices don't need a static IP addresses
- Networks don't use custom DNS servers and allow Google's DNS

You can still use the Balena website to add new devices for deployments that do have custom DNS servers or required static IPs, but with additional setup tasks. If your measurement devices need static IPs, custom DNS servers, or other custom configurations, please review the section Setup New Device Using the Balena CLI and Special Configurations for Specific Network Conditions.

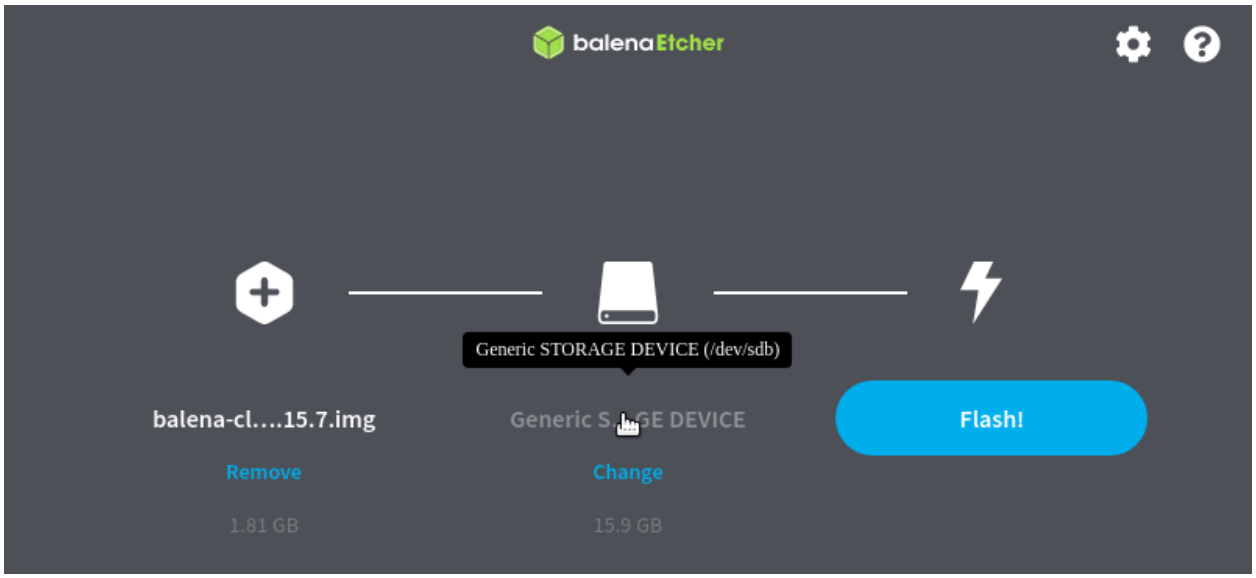
Log into your Balena Cloud application, and click “Add device”.



- Select the armv7 device type you plan to deploy. We used ODROID-XU4.
- For “OS type”, select “balenaOS”
- For “version”, select “v.2.38.3+rev3 (recommended)”.
- Select the “Production” edition
- For “Network Connection” select the connection you plan to use for this device in your deployment.
    - If selecting “Wifi + Ethernet”, enter the Wifi SSID and Passphrase, if applicable.
- Click “Download balenaOS”

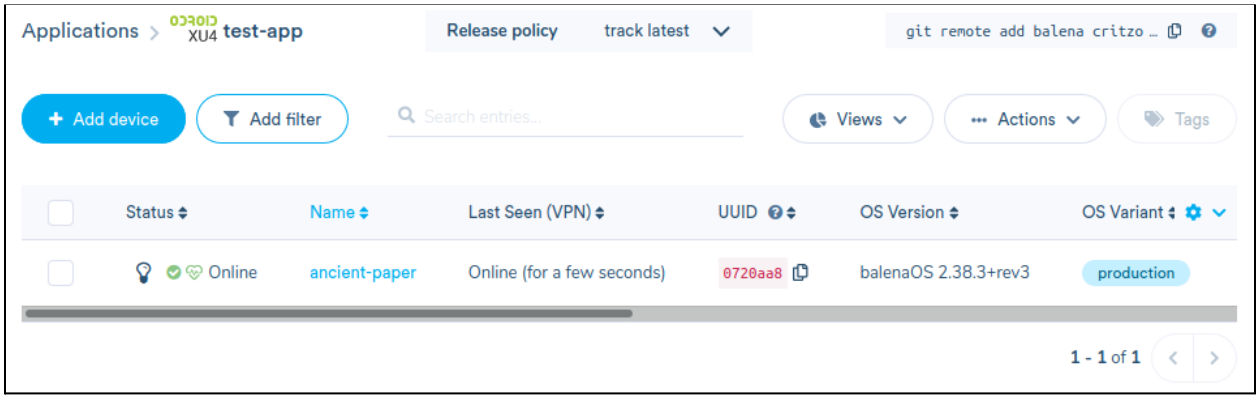
The Balena OS image will download to your computer. At this point, the image is associated with your Balena application and is ready to be installed on one or more measurement device SD cards. **If all your measurement computers will connect to the Internet in the same way, you could use this image for all of them.** However, each location where you wish to place a measurement device is likely to have different network connection requirements. For example one library might use custom DNS servers, all libraries will have different WiFi SSID’s and passphrases, and some library networks may require static IP addresses. We’ll address those custom configurations and setup instructions in the next section.

Insert a fresh SD card into your card slot or adapter, and use Etcher to flash the newly downloaded image to it. Select the image you downloaded, and then select the SD card using the middle button. **Double check that the SD card is selected!** You wouldn’t want to overwrite your computer’s hard drive.



After the image is written to the SD card, insert the card into your target device. It is now ready to be deployed to the intended location.

When the device is deployed, it will first register with your Balena application. If everything is working correctly, the new device should appear in your Balena app dashboard with a random name.

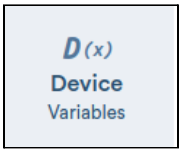


You can now rename the device and add the three environment variables discussed in the previous section, *Name the new measurement computer*.

Click on the device name, then on the device detail page, click the pencil icon to edit the name.

Next, click on the “Device Variables” button in the left menu, and add the environment variables for this device:

- MURAKAMI_SETTINGS_**LOCATION**
- MURAKAMI_SETTINGS_**CONNECTION_TYPE**
- MURAKAMI_SETTINGS_**NETWORK_TYPE**



The device has now been added to your Balena Application, but isn’t doing much yet. We’ll next go over an additional way to add devices, then we’ll deploy our first release.

Setup New Device Using the Balena CLI

The Balena CLI provides many commands to interact with your project and devices. We’ll use it to deploy a new release of Murakami to your devices, but you can also use Balena CLI to setup a new device.

All setup instructions in this section will use the command line terminal on your development machine, and will use the Balena CLI tool. If you have not already, please install the Balena CLI tool, and ensure you are logged in to your Balena Cloud account. To login once you have installed Balena CLI, type `balena login` on your terminal, choose “Web Authorization”, and follow the prompts.

Once you are logged in, you can confirm by listing your apps using the command:

```
balena apps
```

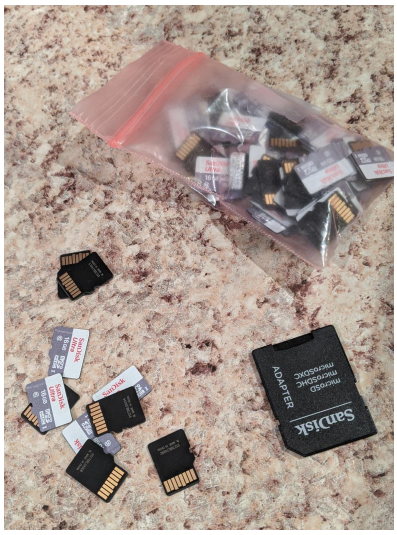
Your apps should be listed along with the number of online devices and total devices.

Obtain a Balena OS image for your device type

Next, you’ll need a copy of Balena OS for the type of device you’re using. You can download an image that will already contain your app’s current release from the website as described in the previous section, but you can also use the Balena CLI to download a generic Balena OS image:

```
balena os download odroid-xu4 -o
images/balena_odroid_xu4-v2.38.3+rev3.img --version 2.38.3+rev3
```

In the command above, we’re downloading the Balena OS image for the Odroid-xu4, version 2.38.3+rev3, and saving it into a folder called “images”. You can write this base image to all of your measurement devices, but will also need to give each of them custom configurations after associating them with your application. **The instructions in the remainder of this section assume that you are using a generic Balena OS image downloaded using the command above.**



Register a new device in your application

Next, register a new device in your application using the command:

```
balena device register <app-name>
```

If successful, you will get a response with the new device's ID in the terminal:

```
Registering to <app-name>:
dd8f9fdf9eeb0c25d33b18910036f35033a8355012022e5a164e50ad875639
```

Next, rename the device to something more meaningful:

```
balena device rename
dd8f9fdf9eeb0c25d33b18910036f35033a8355012022e5a164e50ad875639
<new-name>
```

You should now see the new device in your project on Balena Dashboard.

Generate a configuration file for this device

Next, generate a configuration file by using the command(s) below. Note that the specific text in the commands will be different based on the device ID, and the options you need for the device. We can use the first seven characters of the device ID, and the version should match the Balena OS version you are using.

Example for generating a config for a WiFi connected device:

```
balena config generate --device dd8f9fd --version 2.38.3+rev3
--output configs/<device-name>-config.json --network wifi --wifiSsid
mySsid --wifiKey abcdefgh --appUpdatePollInterval 1
```

Example for generating a config for an ethernet connected device:

```
balena config generate --device dd8f9fd --version 2.38.3+rev3
--output configs/<device-name>-config.json --network ethernet
--appUpdatePollInterval 1
```

In the above examples, we're saving the generated configuration file in a folder called "configs" which must already be present. Naming your configuration files including the device name helps keep track of which configuration is on which device. You'll need to know this if you're setting up more than a few devices.

Flash the Balena OS image to an SD card

Insert a fresh SD card into your card slot or adapter, and use Etcher to flash the newly downloaded image to it. Select the image you downloaded, and then select the SD card using the middle button. **Double check that the SD card is selected!** You wouldn't want to overwrite your computer's hard drive :)

You should also note the drive path for the SD card as you mouse over the middle button. You'll need that exact path in the next section.

After the image is written to the SD card, eject it and reinsert it into your computer so it's re-mounted. You should see several volumes once the SD card has been reinserted

Apply your generated configuration file for this device to the SD card

In your terminal window use the command below to inject the configuration file onto the SD card. If you're using a different device type than the odroid-xu4, substitute the device type, and at the end of the command, replace `/dev/disk2` with the drive path you noted in the previous step.

```
balena config inject configs/<device-name>-config.json --type
odroid-xu4 --drive /dev/disk2
```

Next, review the remaining sections covering additional custom configurations that may apply to your device(s).

- WiFi connected devices may need to have a configuration file added that de-prioritizes the ethernet interface to properly connect via WiFi.
  See the section: **Special Configuration for WiFi Connected Devices**
- Your network requires custom DNS servers and/or does not support Google Public DNS.
  See the section: **Special Configuration to Support Custom DNS Servers**
- The device needs a static IP address. See one of these sections, as appropriate for your device:
  Configure a Static IP Address for Ethernet Connected Device OR
  Configure a Static IP Address for WiFi Connected Device

If none of the situations above apply to your device, it is ready to be deployed in the field. If any of them do, please read on.

Special Configurations for Specific Network Conditions

Setting up your device using the Balenda CLI in the previous step registers the device with your Balena Cloud project and creates a Balena OS configuration file on the device: `/resin-boot/config.json`. For basic network configurations, this is all that's needed. Learn more about the Balena OS configuration file options, please review the Balena Cloud page, *Configuring Balena OS*. To support the device's operation under some network conditions, you may need to edit the Balena Cloud configuration file, and you may also need to create "system connection" files for the device's ethernet and/or wifi interfaces in the folder: `/resin-boot/system-connections`. These files are NetworkManager interface configuration files, which is the connectivity management package in Balena OS 2.0. Learn more about Balena Networking on their Network Setup page.

The following sections cover network configurations encountered in the MLBN research program. The instructions below assume you have just set up a new device using the Balena CLI. Please note that the examples below cover configurations for each special case. The networks where you deploy your measurement devices may require a combination of the settings below from each case, for example a WiFi device with custom DNS servers, or a static IP address on a WiFi device.

Special Configuration for WiFi Connected Devices

In the basic configuration above, you defined the WiFi SSID and Passphrase for the Balena OS configuration file. In addition, the device needs to have a network configuration file for the ethernet interface, to de-prioritize it in favor of the WiFi interface.

- Eject the SD card and reinsert it into your computer to re-mount it
- Navigate to the folder `/resin-boot/system-connections` using either your terminal or a GUI file browser
- Create a new text file to define a custom network configuration for the ethernet interface: `/resin-boot/system-connections/ethernet`. You can actually name the file whatever you'd like, but we recommend naming it for the interface it defines.
- To deprioritize the ethernet interface on a WiFi connected device, paste the configuration below into your new ethernet system connection file. Please change `eth0` if your device's interface is named differently

```
[connection]
id=my-ethernet
```

```
type=ethernet
interface-name=eth0
permissions=
secondaries=

[ethernet]
mac-address-blacklist=

[ipv4]
never-default=true
route-metric=2000
dns-search=

ignore-auto-routes=true
method=auto

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
method=auto
```

Special Configuration to Support Custom DNS Servers

By default, Balena OS will use Google's Public DNS servers (8.8.8.8 8.8.4.4) for name resolution. If the device will be placed in a network that uses custom DNS servers, for example OpenDNS, additional configurations are needed. Usually the network's DHCP server will provide DNS servers to each client that connects. However in the case of Balena OS, we need to manually specify them in each device's Balena OS configuration file and in the system network configuration file for each interface.

Add Your DNS servers to the Balena OS configuration file.

It's very important to double check this edit before deploying your device to make sure there is not a mistype or formatting error. This file is in JSON format, which is a structured format for representing "key": "value" pairs of information, separated by commas, and enclosed in curly braces: { }. In this case used to configure the device's operating system.

Open the file `/resin-boot/config.json` and add your DNS servers to it, separated by spaces. See the example config.json contents below, with the text you need to add in bold (of course please substitute the IP addresses of your DNS servers:

```
{"applicationName":"imls-murakami","applicationId":1555306,"devi
ceType":"odroid-xu4","userId":17807,"username":"critzo","appUpda
tePollInterval":900000,"listenPort":48484,"vpnPort":443,"apiEndp
oint":"https://api.balena-cloud.com","vpnEndpoint":"vpn.balena-c
loud.com","registryEndpoint":"registry2.balena-cloud.com","delta
Endpoint":"https://delta.balena-cloud.com","pubnubSubscribeKey":
"","pubnubPublishKey":"","dnsServers":"10.0.10.10 10.0.10.11",
"mixpanelToken":"9ef939ea64cb6cd8bbc96af72345d70d","deviceApiKey
":"9Z3UIt27Eq7paJKsZkg0cyg0aEC47fqE","registered_at":1582814951,
"deviceId":1913027,"uuid":"2bcb7a8bcabdba8739fcec68b1fab56a1e5ba
28c798424e8bde803164be946"}
```


If you have multiple DNS servers, separate them by a single space as shown above. You can use this line to copy/paste into your device's `/resin-boot/config.json` file: `"dnsServers":""`

Add Your DNS servers to the System Connections Configuration File(s)

If you are using custom DNS servers, you also must define them in the Network Manager configuration file for any network interface on the device that will be connected to the Internet. Typically this is one of the two files, if the device is only connected via one network interface:

- `/resin-boot/system-connections/ethernet` device connected via ethernet
- `/resin-boot/system-connections/wifi` device connected via WiFi

Open a new or existing file in `/resin-boot/system-connections/` to define a system level Network Manager configuration for the interface that will connect this device. The example below shows a configuration for a device connected via ethernet, that receives an IP address via DHCP, and defines two custom DNS servers:

```
[connection]
id=resin-ethernet
type=ethernet
interface-name=eth0

[ethernet]
mac-address-blacklist=

[ipv4]
method=auto
```

```
dns=10.0.10.10,10.0.10.11;
dns-search=

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
method=auto
```

Substitute the IP addresses of your DNS servers, and note that in this file, multiple DNS servers should be separated by commas, not spaces.

Configure a Static IP Address for Ethernet Connected Device

If your ethernet connected device needs to use a static IP address, you should define the IP to be used in a Network Manager interface configuration file, saved in `/resin-boot/system-connections/ethernet`. For this configuration you will need to know 1) the static IP address to be used, 2) the subnet in CIDR notation, and the IP address of the network gateway. Here we provide an example configuration file:

```
[connection]
id=resin-ethernet
type=ethernet
interface-name=eth0
permissions=
secondaries=

[ethernet]
mac-address-blacklist=

[ipv4]
address1=172.16.0.101/24,172.16.0.1
dns=8.8.8.8,8.8.4.4;
dns-search=
method=manual

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
```

```
method=auto
```

In the example above, this line defines the static IP: `address1=172.16.0.101/24,172.16.0.1`

The first value is the device's static IP address: 172.16.0.101
Next we have the size of the subnet: /24
And finally, we have the gateway's IP address: 172.16.0.1

This example also defines the Google Public DNS servers explicitly.

Configure a Static IP Address for WiFi Connected Device

If your WiFi connected device needs to use a static IP address, you should define the IP to be used in a Network Manager interface configuration file, saved in `/resin-boot/system-connections/wifi` . For this configuration you will need to know 1) the static IP address to be used, 2) the subnet in CIDR notation, and the IP address of the network gateway. Of course you will also need to know the WiFi network's SSID and passphrase.

Here we provide an example configuration file for a WiFi device with a static IP address and passphrase:

```
[connection]
id=resin-wifi
type=wifi

[wifi]
mode=infrastructure
ssid=Truro Library

[ipv4]
address1=172.16.0.100/24,172.16.0.1
dns=8.8.8.8,8.8.4.4;
dns-search=
method=manual

[ipv6]
addr-gen-mode=stable-privacy
method=auto
```

```
[wifi-security]
auth-alg=open
key-mgmt=wpa-psk
psk=super_secret_wifi_password
```

Many additional options are supported, which are documented on Balena Cloud's Network Setup page. For example, if your WiFi network uses a different key management algorithm than WPA2, or if the WiFi network SSID is hidden, there are options to support these situations in Balena's documentation.

Summary of Balena CLI Commands / Scripting with Balena CLI

In the previous section we set up a single device using the Balena CLI tool. The Balena CLI offers a many other ways to interact with your app and devices using the Balena API. Advanced users may wish to review Balena's documentation and design their own scripts or applications that interact with it. For advanced users who want to know what's happening "under the hood", this section summarizes the Balena CLI commands used in the previous section. We present the Balena CLI commands used in the left column, and a description of the action that command does on the right. Text in <brackets> indicates a variable name that must be substituted or defined in a script. Upper case variable names are those defined as environment variables specific to the MLBN system. Lower case or camelCase variable names indicate that this value comes from your Balena app.

Balena CLI Command	**Action**
`balena os download odroid-xu4 -o images/balena_odroid_xu4-v2.38.3+rev3.img --version 2.38.3+rev3`	Downloads a copy of BalenaOS version 2.38.3+rev3 for the ODROID XU4 model, saves it in the folder, *images*
`balena device register <balena-app-name>`	Registers a new device with your Balena app.
`balena device rename <deviceId>`	Renames a device that has been registered with your Balena app.
`balena config generate --output configs/<LOCATION>-<NETWORK_TYPE>-<CONNECTION_TYPE>-config.json --device <deviceId> --version 2.38.3 --network`	Generates a configuration file for a device, saving it in the "configs" folder. This example uses three variables that define device names in the MLBN research program.

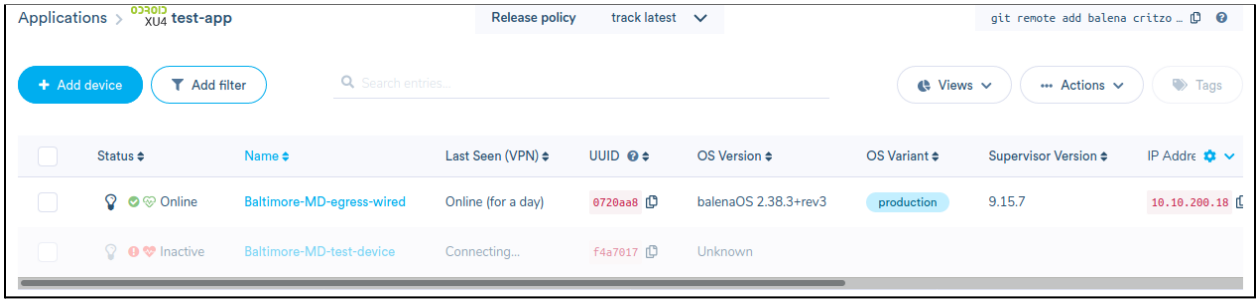
`<CONNECTION_TYPE> --appUpdatePollInterval 1`	
`balena config inject configs/<LOCATION>-<NETWORK_TYPE>-<CONNECTION_TYPE>-config.json --type odroid-xu4 --drive <DRIVE>`	Injects a local BalenaOS configuration file onto an SD card that has been previously flashed with BalenaOS. This allows you to create and backup a device's config.json and add it to an SD card when you're ready to provision the device.
`balena env add MURAKAMI_SETTINGS_HOSTNAME <LOCATION>-<NETWORK_TYPE>-<CONNECTION_TYPE> --device <deviceId>`	Adds an environment variable for a device's hostname to the Balena app.

`balena env add MURAKAMI_SETTINGS_LOCATION <LOCATION> --device <deviceId>`
 	
`balena env add MURAKAMI_SETTINGS_NETWORK_TYPE <NETWORK_TYPE> --device <deviceId>`
 	
`balena env add MURAKAMI_SETTINGS_CONNECTION_TYPE <CONNECTION_TYPE> --device <deviceId>`
	Adds other environment variables for a device.

Push a Murakami Release to Balena Cloud

At this point you likely have one or more Murakami measurement devices added to your Balena app, which are either inactive or online, but are not yet running the Murakami code. You should have at least

one test device registered in your Balena app that is online.

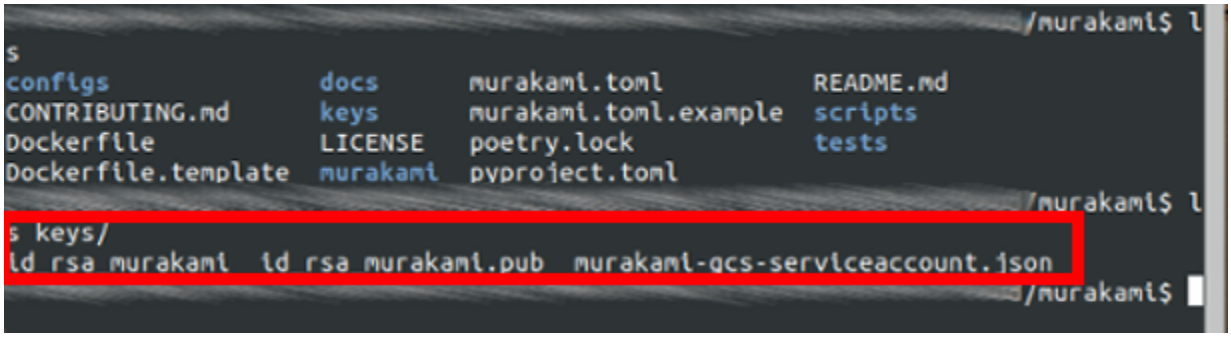


The final step to getting your new devices online and running tests is pushing a release to Balena Cloud. After you push a release to your app, all Murakami measurement devices in your Balena app that are online will automatically download and start running the latest release. New devices added to your app or that come online later will also download and start running the app's latest release.

Prepare and Push a Release

Open a terminal window on your local machine and navigate to the folder where you cloned the Murakami code.

1. **Add key files to support GCS and/or SCP exporters**
   If you are using the GCS or SCP exporters, ensure that you have placed the required key files for these services in the folder `/keys` . For example, in the screenshot below we have three files in the keys folder:

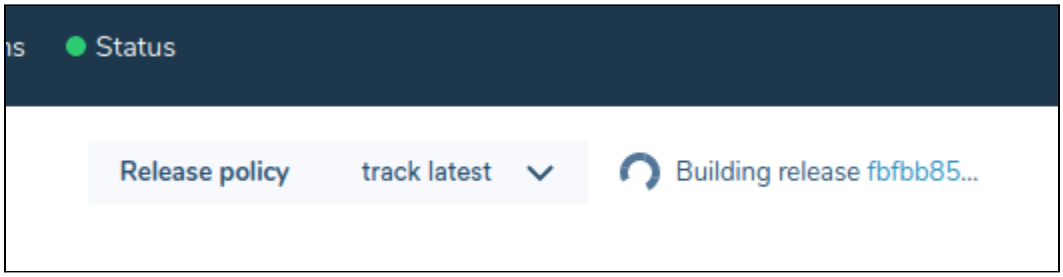


2. **Confirm environment variables for your application have been added**
   Confirm you have added all relevant environment variables for your application and devices. You can always change environment variables and values later if you find one is incorrect.

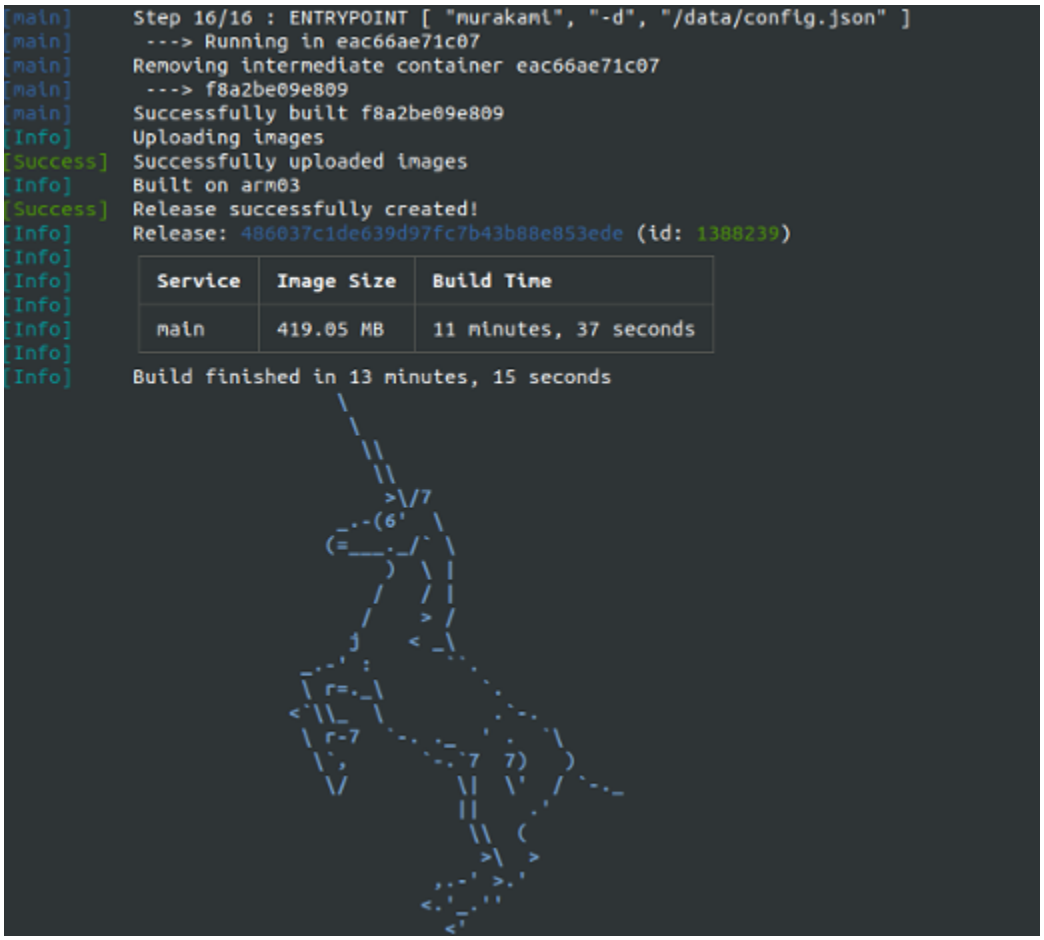
3. **Push code to Murakami to Balena's build servers, and deploy to devices in your Balena App**
   Lastly, use the command below to push your code to Balena's build servers:

```
balena push your-balena-project-name -c
```

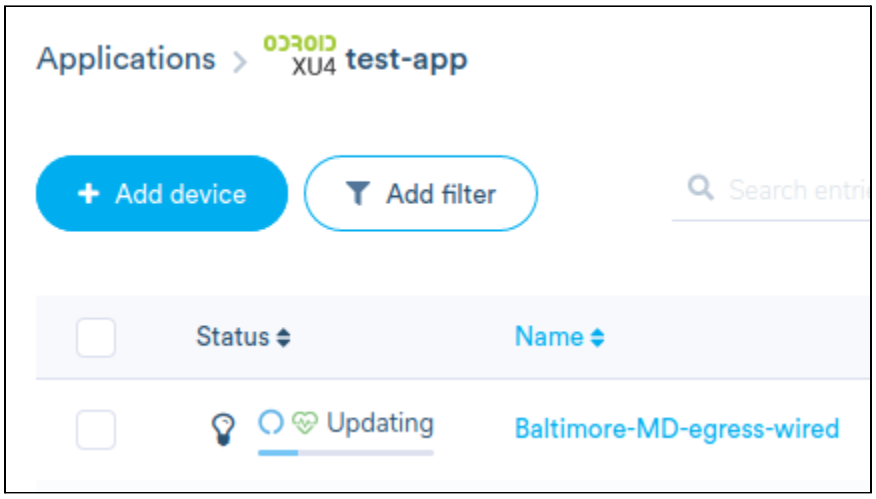
The terminal will display a large number of status messages, and will either finish with success, or display any errors. During this process, Balena's servers are building a Docker container with your Murakami code. While your build is in progress, the Balena Dashboard for your app will provide status for you, as seen in the image below.



If the build is successful, any devices added to your app will automatically download the new container image and start it. When the image has built successfully, you'll see a confirmation message like the one below in your terminal window.

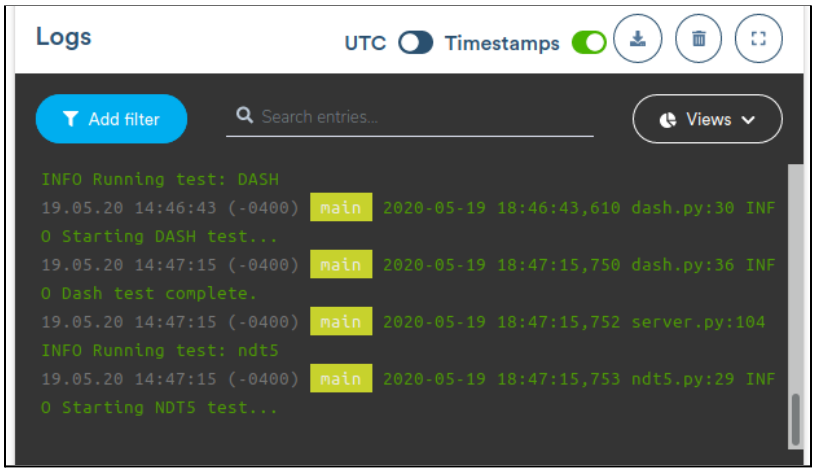


Switch over to your Balena app on the web, and you should see any devices in your app that are online are now in “updating” status.

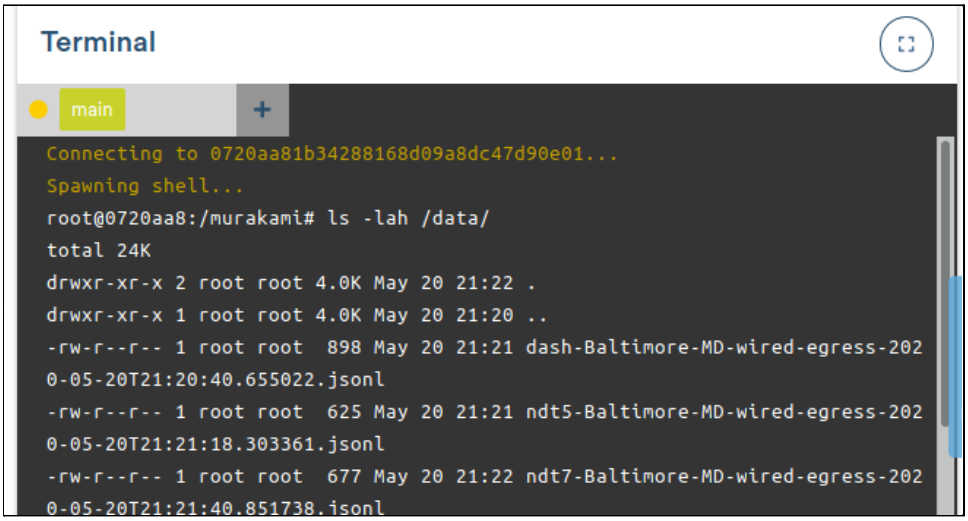


4. **Confirm your Murakami devices are running tests and sending test result data to the intended locations.**

   View a device's details page on the Balena website. It should be providing logs:



   Start a terminal session in the Main container to confirm that test results are being saved locally. If you are using the Local test exporter, listing the contents of the folder `/data/` should show some test results:



If you are using the GCS or SCP exporters, test data should be getting sent to the GCS bucket(s) or SCP server(s) that you defined in your environment variables.

If you are using the Murakami Data Visualization Service and the HTTP exporter, data should be making its way into the Visualization service's database.

At this point you should have at least one working Murakami measurement device registered in your Balena app, running tests, and collecting data. Now you're ready to set up more of them, repeating the process described in this section, and when you're ready, installing them in the remote locations that you planned in your Measurement System Deployment Plan.

We recommend that you set up each device in advance using the details you gathered in your Measurement System Deployment Plan. This allows you to register and configure each device ***before connecting it in the field.*** Each device should be registered in your Balena app as "Inactive", until it is connected in the remote location where you had planned.

Deploy Measurement Devices in Intended Locations

If you've reached this point, you likely have some Murakami measurement devices setup and ready to be deployed in the field. This may involve mailing devices to your colleagues or team members at remote locations, or you and your staff may connect them one at a time.

When you connect a device in a new location, if your configurations are correct it should show up as "Online" in your Balena app dashboard within seconds of connecting to the Internet, and will download and start running the latest code release.

Troubleshooting Tips

If the device does not show as Online, there is likely an issue with its configuration. This can happen if the device has custom or special network configurations, or if incorrect information was used to make the configurations. It's also possible that the device is functioning correctly, but the network may be blocking it from reaching the Internet.

Is the Network Configured to Support Murakami?

First, confirm that the network is configured to allow the ports and services required by the Murakami measurement software. These requirements are outlined above in the section on Pre-Installation. The network administrator may need the MAC addresses of each device in order to support the required ports and services. If a change needed to be made, reboot the measurement device by unplugging it and plugging it back in again. Then check the Balena app dashboard to see if the device comes online.

Are the network configurations for this device correct?

Confirm that you had the device's network configurations correct. Sometimes details get lost in the shuffle. A simple misspelling or extra space in a configuration file might be the culprit. If you have the correct information (for example, WiFi network name & password, static IP address, or DNS servers), and if you do, make sure this information is correct **in the configuration files on the device.**

It's good practice to have a computer with an SD card reader on site when you deploy your Murakami measurement devices, in case you need to confirm configuration details or make changes. This way you can pull the SD card from the Murakami device, and open it on your computer to make changes. Refer to the previous section, *Special Configurations for Specific Network Conditions,* for more details on which files to check.

If you made changes to network configuration files on the device, reinsert the SD card into the device and power it up again. Then check the Balena app dashboard to see if the device comes online. If it doesn't, confirm the IP address, DNS servers, etc. with your network administrator to diagnose issues further.

Setup the Measurement System Data Visualization Service

Murakami Viz is a visualization service that receives tests from Murakami devices and provides access to them. The service can be installed on a cloud provider or server, and configured to receive results from measurement devices running in multiple, but disparate networks. However, it can also be run on a server or virtual machine in a local or wide-area network if desired. In either case, the measurement devices that send measurements to an installation of Murakami Viz must be able to access the Murakami Viz server over the local or wide area network, or over the public Internet.

Installing Murakami Viz

Complete instructions on how to install and configure Murakami Viz can be found in the README on Github, but are repeated here for completeness.

System Requirements

- Node.js version > =12
- npm
- To run the service with TLS support, a fully qualified domain name with DNS entry pointing to your server's public IP address
- To run the service with Docker, your server or virtual machine should have Docker and Docker Compose installed

Structure

Murakami Viz is composed of two different parts:

- A React-based **frontend**
- A Koa-based **backend** that renders & serves the frontend and exposes an API used by the frontend

These parts are located in the code repository:

- src/backend # The backend components
- src/common # Common code and assets
- src/frontend # The React frontend

Configuration

Murakami-viz is configured via variables either specified in the environment or defined in a `.env` file (see `env.example` for an example configuration that may be edited and copied to `.env`).

The backend parses the following configuration variables:

```
MURAKAMI_VIZ_LOG_LEVEL      # Logging level (default: error)
MURAKAMI_VIZ_HOST           # The host Murakami-viz runs on (default: localhost)
MURAKAMI_VIZ_PORT           # The port to bind to (default: 3000)
MURAKAMI_VIZ_ADMIN_USERNAME  # The administrative user (default: 'admin')
MURAKAMI_VIZ_ADMIN_PASSWORD  # The administrative password
MURAKAMI_VIZ_DB_HOST        # Postgres database host (default: localhost)
MURAKAMI_VIZ_DB_PORT        # Postgres port (default: 5432)
MURAKAMI_VIZ_DB_DATABASE      # Postgres database name (default: murakami)
MURAKAMI_VIZ_DB_USERNAME    # Postgres user (default: murakami)
MURAKAMI_VIZ_DB_PASSWORD    # Postgres password
```

```
MURAKAMI_VIZ_DB_POOL_MIN    # Postgres minimum connections (default: 0)
MURAKAMI_VIZ_DB_POOL_MAX    # Postgres max connections (default: 10)
MURAKAMI_VIZ_DB_TIMEOUT     # Postgres connection timeout (default: 0)
```

Additionally, we use the semi-standard `NODE_ENV` variable for defining test, staging, and production.

Deployment

Standalone

First, clone the Murakami Viz repository and from the root of the resulting directory install Murakami-viz's dependencies:

```
npm install
```

Then, build all components:

```
npm run build
```

Create the database:

```
npm run db:migrations
```

and to optionally populate it with test data:

```
npm run db:seeds
```

And start the running processes (with necessary environment variables if not defined in `.env`):

```
npm run start
```

(use npm run start:dev to run in development mode)

Docker

You can deploy this tool using Docker. There is an included `docker-compose.yml` file that will allow you to run it in a production configuration. First, clone the repo and from this directory run docker-compose:

```
docker-compose up --build -d
```

This will build the docker container from the current repository, download the official Postgres docker image, and configure them both (the -d flag will detach from the current shell so that you can leave it running, but you can omit it in order to leave the log output attached).

If this is the first time you've run it on this system, you'll want to run the database migrations to initialize the database:

```
docker-compose run murakami db:migrations
```

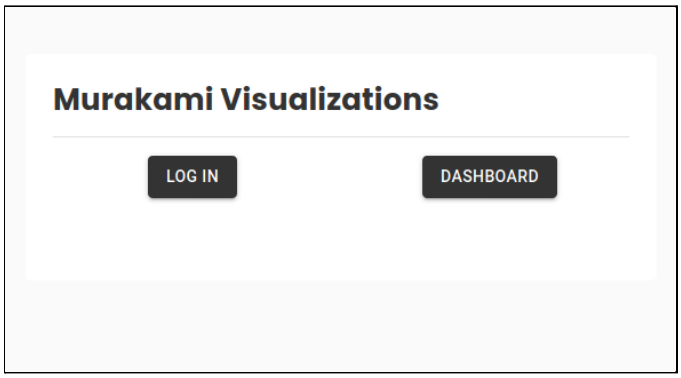
and then optionally seed the database with a default admin user:

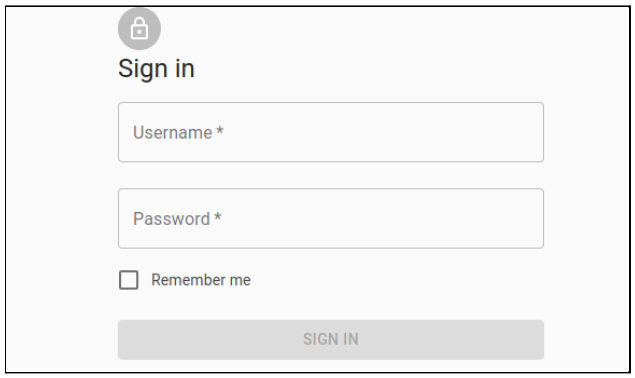
```
docker-compose run murakami db:seeds
```

By default, Murakami Viz runs on http://localhost:3000, but you can place it behind a proxy such as Nginx in order to provide TLS support and other features.

Configuring Murakami Viz After Installation

Once installed on your server, open the server's IP address or domain name in your web browser. You should see the login page for Murakami Visualizations. Login using the default **admin** account that was set up in your environment variables.





The Murakami Viz service allows an administrator to coordinate data collection from multiple locations, and assign user accounts to each location separately. In the case of the MLBN program for example, one administrator added all library locations where measurement devices were placed, added information about each device in each location to the system, and assigned a user account in the Murakami Viz service to their relevant locations. Those users could then login and interact with the data collected by measurement devices in their location, but not the data from other locations.

Administrator Setup Tasks

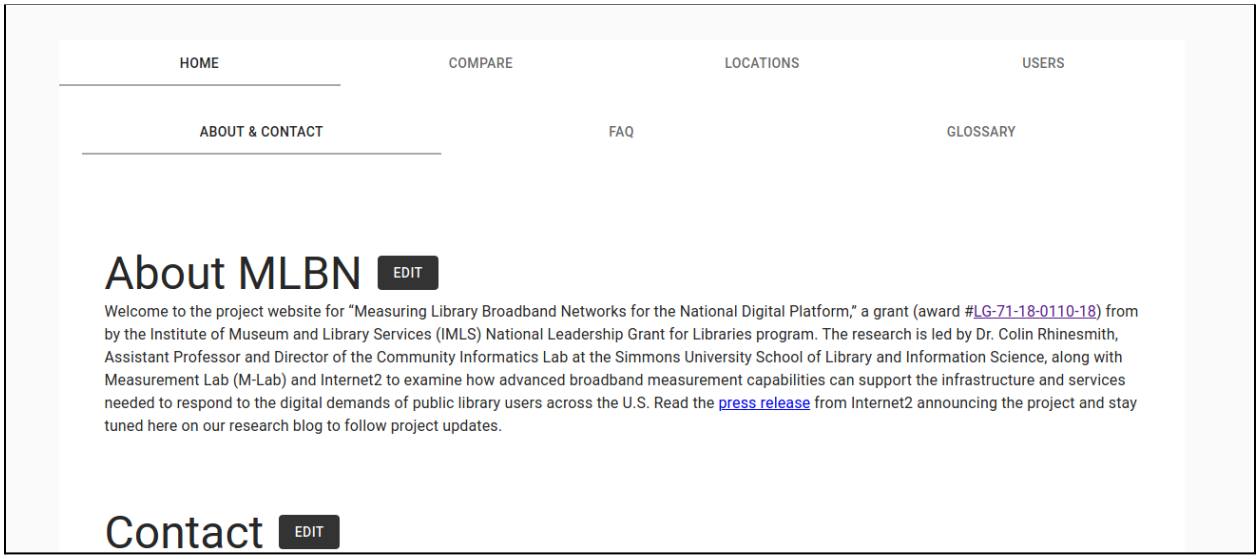
Your primary setup tasks as an administrator of Murakami Viz will be:

- Add About, Contact, FAQ, or Glossary content if desired
- Add Locations and Measurement Devices

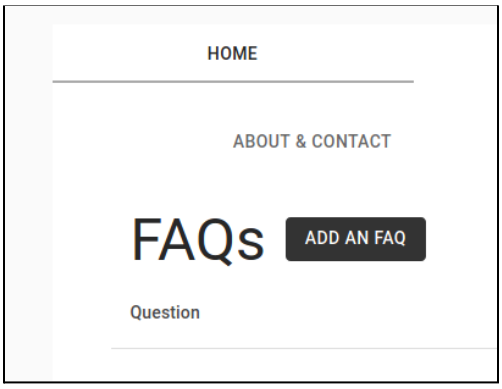
- Add Users for each Location if desired

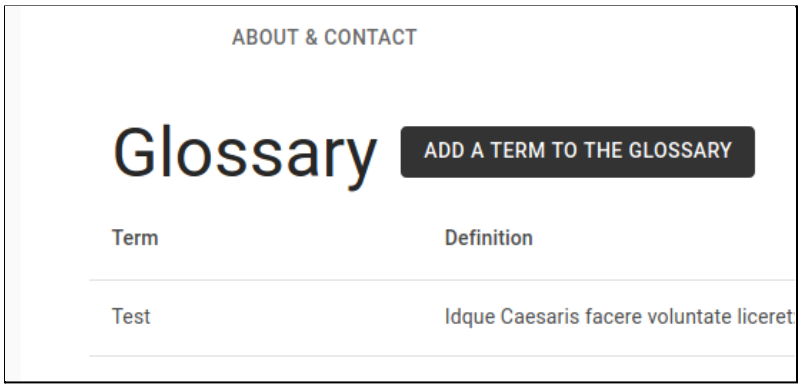
Add About, Contact, FAQ, or Glossary content if desired

When logged in as an administrator, you can add content that other users of the system can see, such as the content on the About & Contact, FAQ, and Glossary pages.



Clicking the buttons on each page labelled: **EDIT , ADD AN FAQ,** or **ADD A TERM TO THE GLOSSARY** will open an overlay window allowing



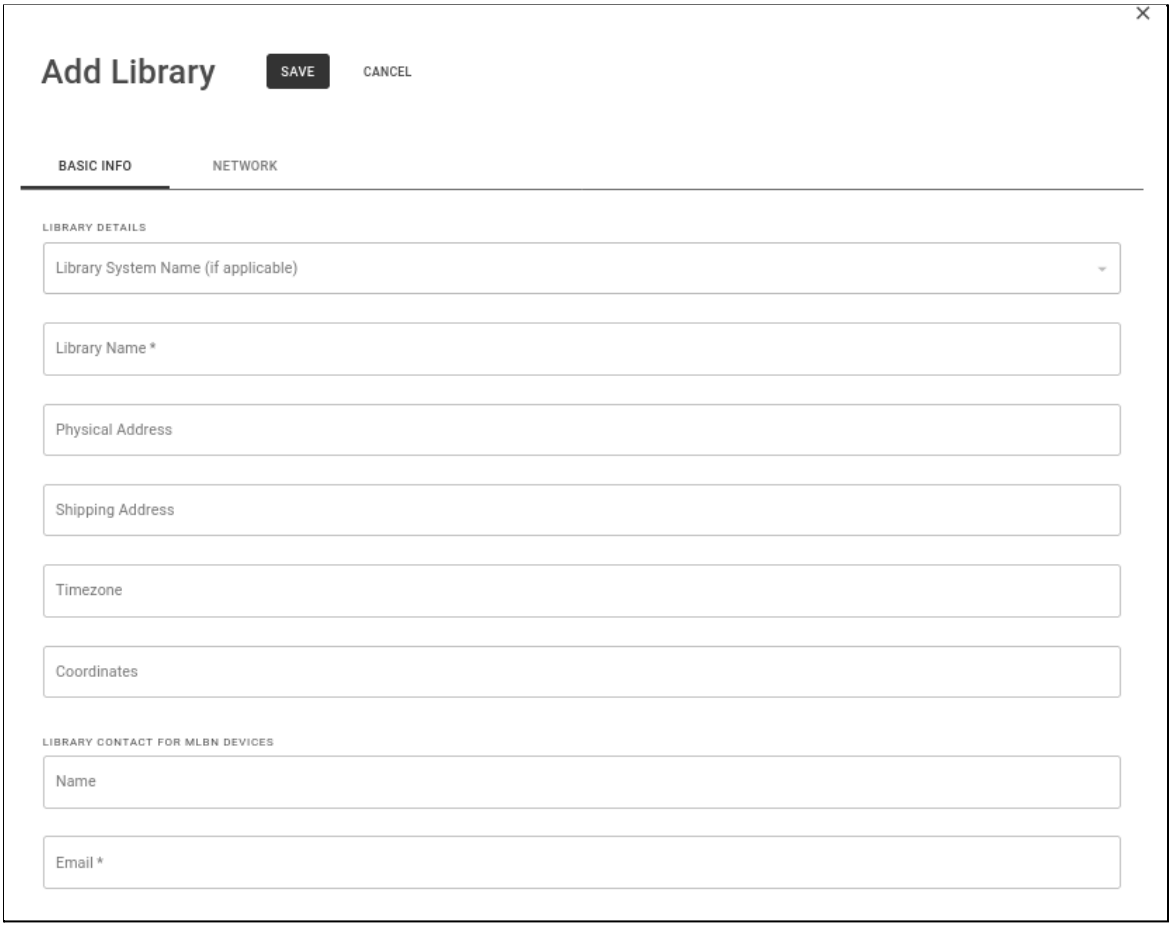


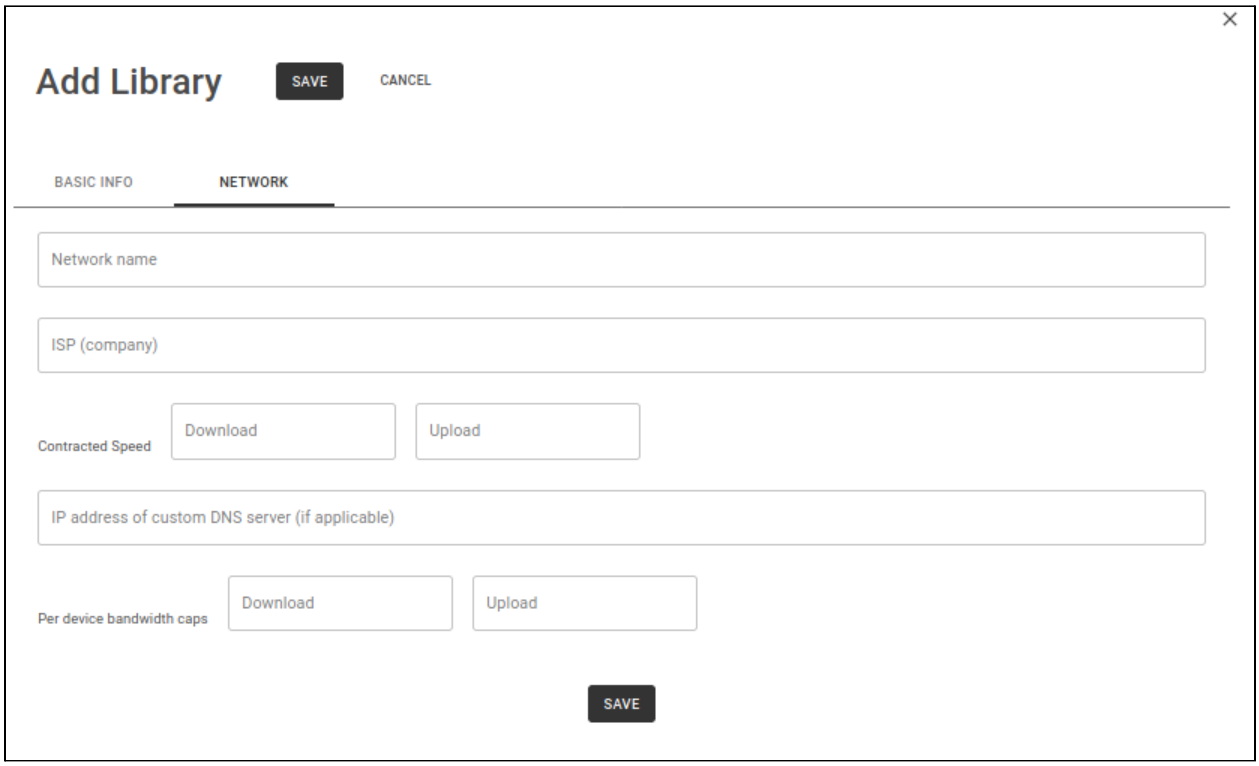
you to edit and save content. HTML is supported in these fields. The intention of these content pages is to support staff at the locations where you are measuring, if they are to login and see the data collected for their location.

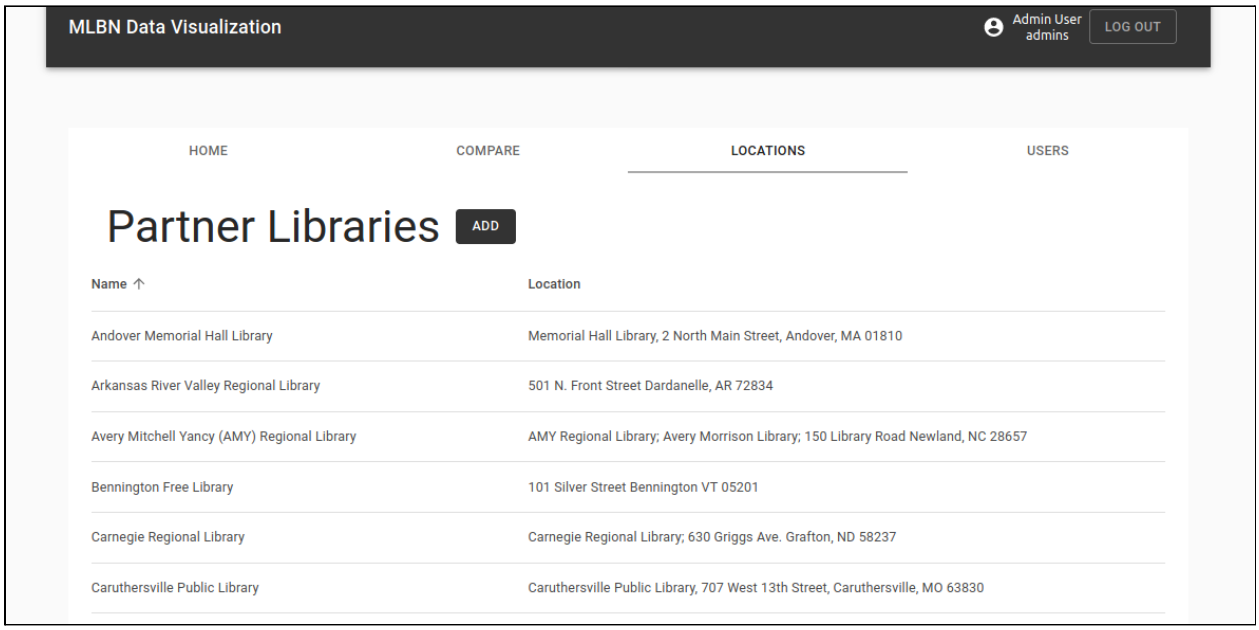
Add Locations and Measurement Devices

Clicking on **LOCATIONS** provides a list of libraries that have been added to Murakami Viz. Click **ADD** to add a new location. Required fields are marked with an asterisk (*). There are two tabs- BASIC INFO and

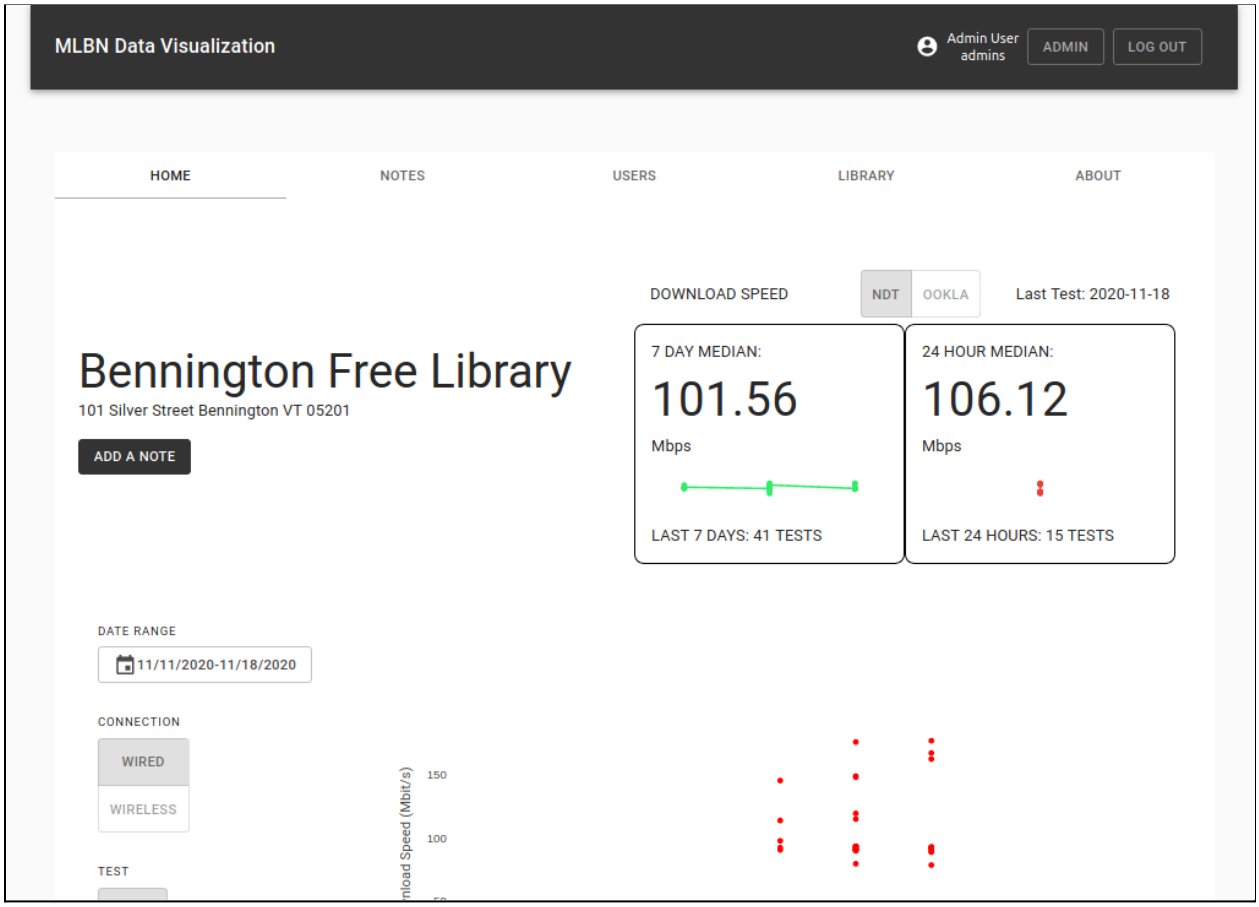
NETWORK. Complete the fields over both tabs and click save. The location should then appear in the list of locations.







Next, click on a Partner Library Location to view it's Home page. This view is what users assigned to the location will see. An example from the MLBN program is shown below:

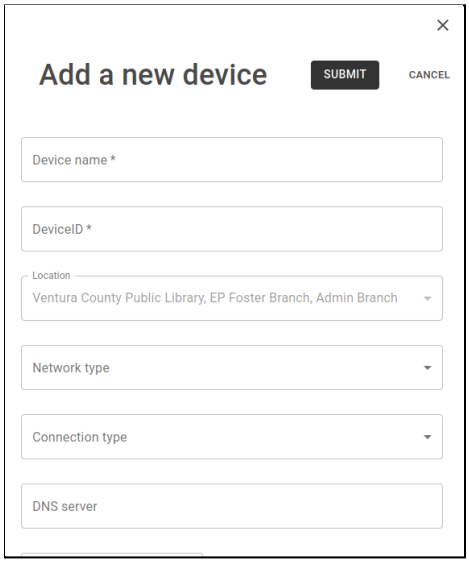


Next, you'll add measurement devices to the location and whitelist the location's public IP address(es).

From the location's home page, click on the **LIBRARY** menu item. You will see the information about that location that you entered before.

Scroll down to the Measurement Devices section and click **ADD A DEVICE**. Add a Device name and DeviceID.

You must use the UUID assigned to your device in your Balena Cloud project for DeviceID.



Select the appropriate value for **Network Type** and **Connection Type**. The values for these two fields are hard-coded, and reflect the types of networks and connections of interest to the MLBN program:

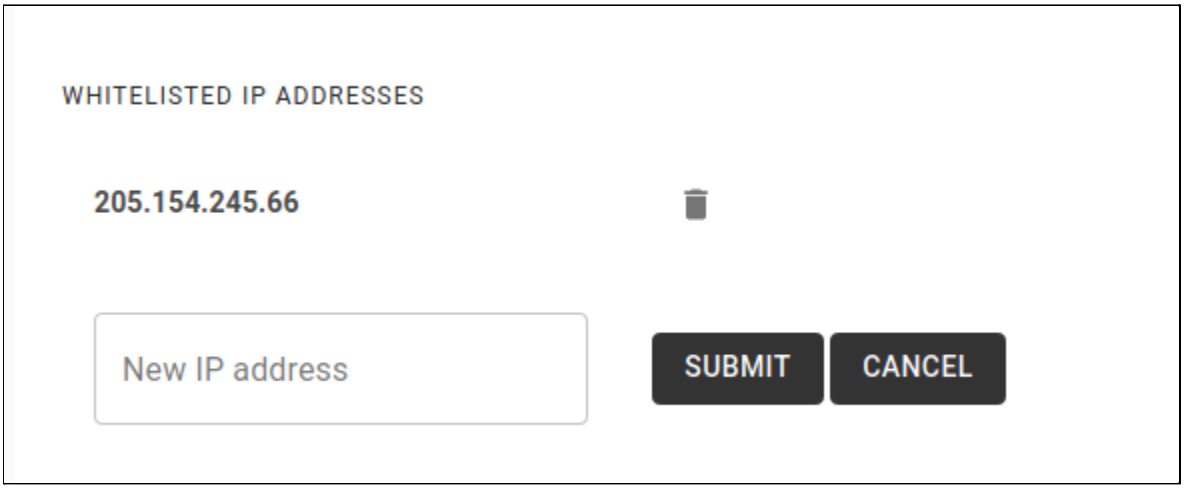
Network Type	*Public*	Identifies devices connected to a network serving library patrons
	Private	Identifies devices connected to a network serving library staff
Connection Type	*Wired*	Identifies devices connected to ethernet
	Wireless	Identifies devices connected via WiFi, or devices connected with ethernet on a VLAN serving WiFi access points

Remaining values are helpful for logging complete information about each device in Murakami Viz, but only **DeviceID** is used

Whitelist IP Address(es) for the Location

To receive test results from Measurement Devices, your instance of Murakami Viz service needs to know which devices are assigned to each location. You did this in the previous step. The service also needs to know the IP addresses from which Measurement Devices will send results.

When editing a location in Murakami Viz, add the public IP address(es) of each location. If your location's IP address is static, adding it here will be a one time task. If your location's IP address is dynamic, you will need to add additional IP addresses as new ones are assigned to your on-premise egress device, such a cable or DSL modem.



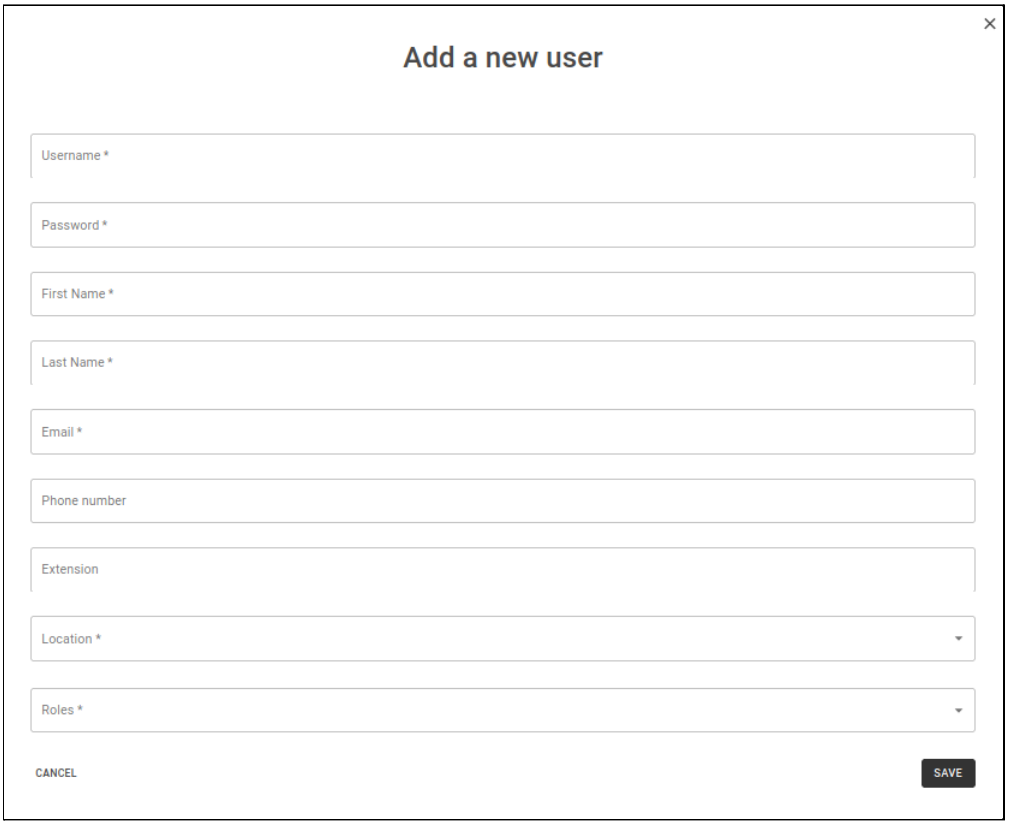
Dynamic addresses may be in use by smaller libraries, where the network more resembles at-home Internet service. In this case, using *Murakami-Viz* will require the addition of new whitelisted IP addresses when the IP at the location changes.

Once you're done, return to the Admin Home page by clicking on the **ADMIN** button in the top right.

Add Users for each location if desired

If you are setting up the Murakami Viz service and wish to provide access to staff or others so they can view and interact with test data being collected by measurement devices in their library, an administrator can add users and assign them to a previously added library location.

The basic user management features of the Murakami Viz allow you to provide people in your measurement program to login and see their location's measurement data. From the Admin Home page, click on Users. Below is an image of the add new user form:



Required fields are labelled with an asterisk (*). Some items to note about users:

- **Username** is not editable once initially saved, but users can change their passwords
- The system does not send emails to users (password reminders, notifications, etc.)

- A user may only be assigned to one Location
- There are three **Roles:** Admin, Editor, and Viewer.
    - The **Admin** role gives the user full administrative permissions to see and edit all information from all locations
    - The **Editor** and **Viewer** roles currently provide the same permissions, and were separated during the software development process to plan for future, separated roles.

  Currently Editors and Viewers can:

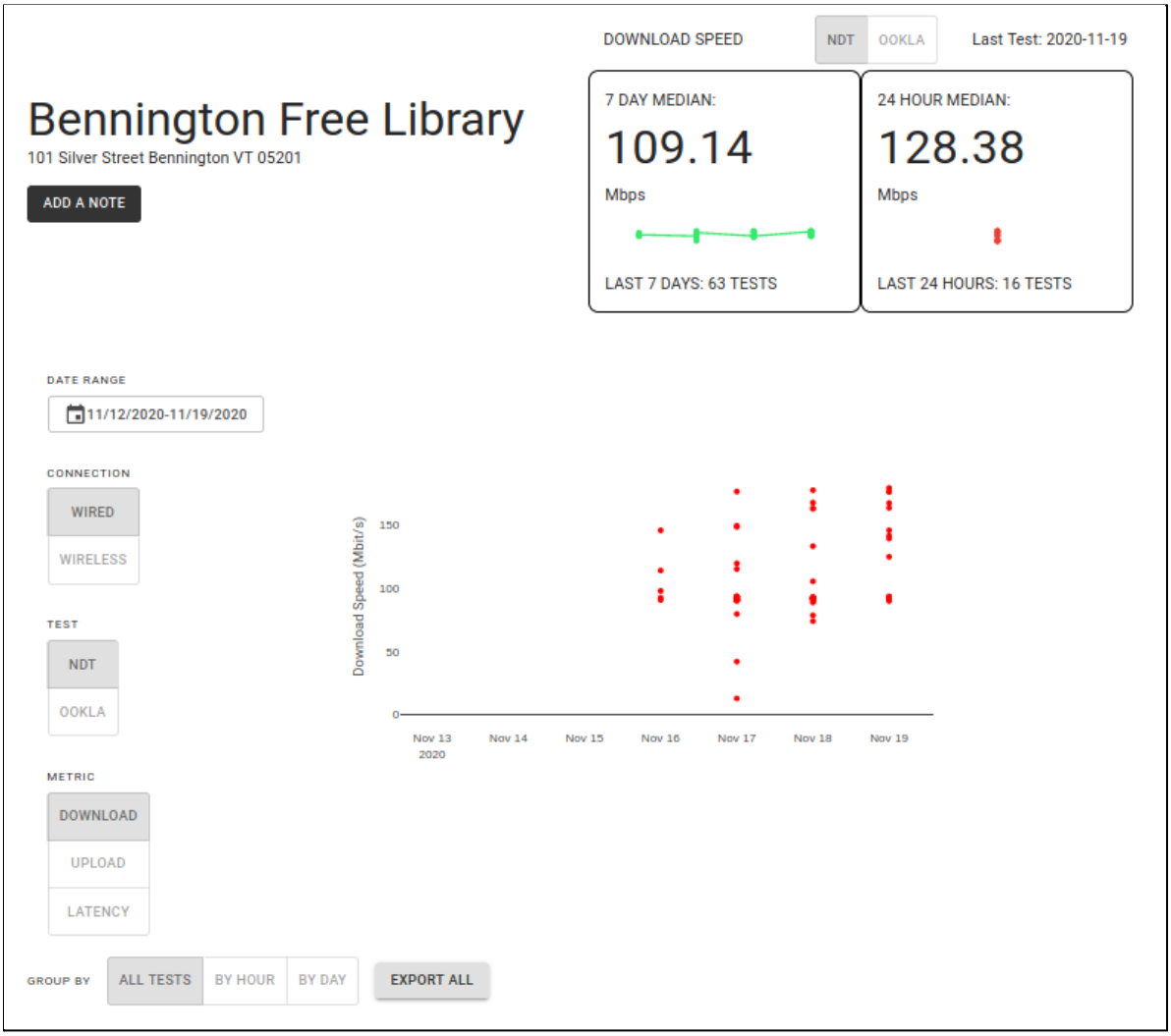
    - Change details about their user account
    - View and interact with the data from their assigned library location
    - Add Notes
    - Edit or remove a defined Network or Device for the location
    - Add or Remove Whitelisted IP Addresses for the location

Accessing and Using Test Data

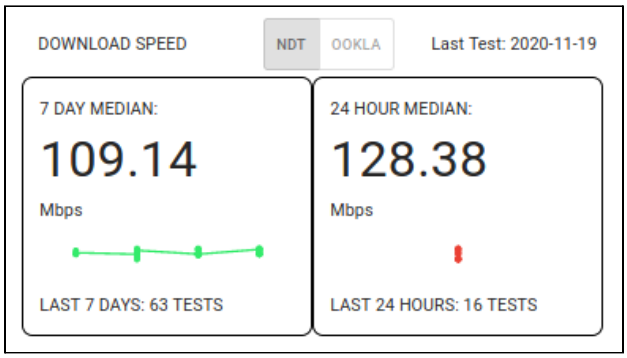
Using the Optional Data Visualization Service

If you have set up the Murakami Viz service as described in the previous section, the service provides a website where you and users you assign can login to access test data generated from measurement devices.

When a user logs into the Murakami Viz service, they will see the Home page for the library location to which they've been assigned. This page provides several ways to view, filter, and download results collected by measurement devices at the location. The image below shows the Home page for one of the libraries in the MLBN research program.

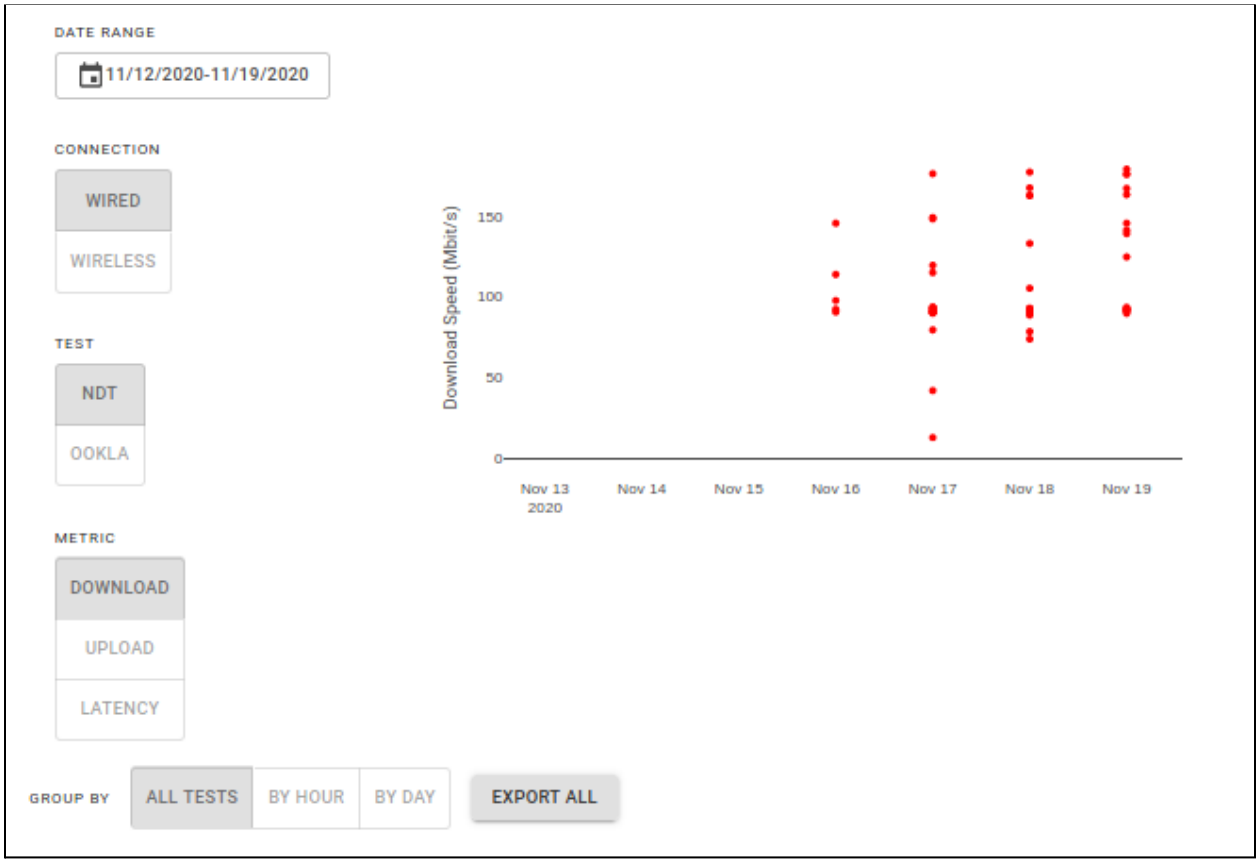


At the top right, two widgets display the 7-day and 24-hour median speeds, along with the number of tests within those time periods. Above the widgets, users can toggle between results of the NDT and Ookla tests, and see the date and time of the last test.



Moving down the page, the larger chart and its filter controls on the left provide more interactivity and allow users to download or export the selected data.

In the **DATE RANGE** filter, select the start and end dates of the test data you wish to view. The default is the current week.



CONNECTION allows a user to filter the data by the type of connection assigned to measurement devices. The default selection is WIRED.

TEST filters the type of test results to display (NDT or Ookla). The default selection is NDT.

METRIC toggles between download speed, upload speed, and latency. The default selection is DOWNLOAD.

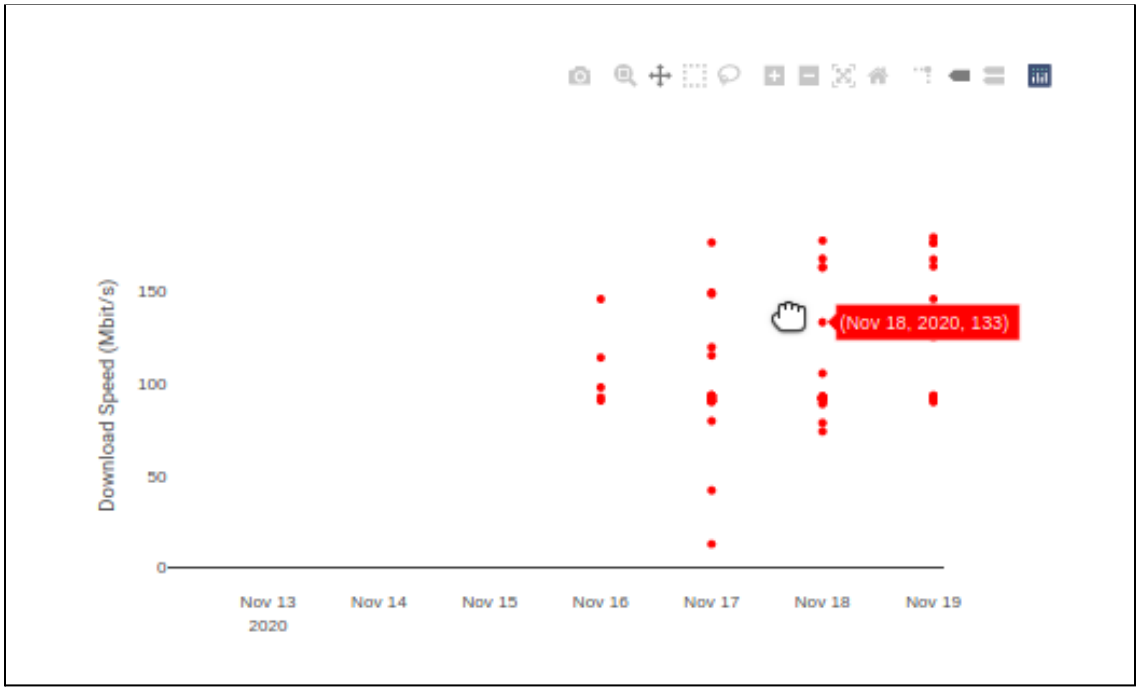
Finally, at the bottom left, users may display test results using the **GROUP BY** filter.

- The default selection is ALL TESTS, which displays all individual test results given the selections in the previous filters.
- Selecting BY HOUR provides a view of tests by hour of the day, over the date range selected. For example if I selected November 1, 2020 - November 20, 2020, and clicked BY HOUR, the X axis of

the chart will display hours of the day from 0 - 23 and display individual tests over that date range that occurred in each hour time block.

- BY DAY is intended to be a future feature to display selected test results by day of the week. This will be added in a future release, and will display results by weekday, i.e. Sunday, Monday, Tuesday, etc.

When viewing the selected data in the main chart, additional information and interactive chart controls are available. Placing your mouse over a test result displays a small window showing the test date, time, and measurement. When your mouse is over any part of the chart, a set of controls appears at the top right. This menu is shown in both images below.



From left to right, these controls are:

- Download plot as a PNG image file
- Zoom in by clicking-dragging
- Pan left or right - views previous or next days or hours, depending on your filter selections
- Box select - draws a box outline around a specific area of the chart to highlight it

- Lasso select - draws a freehand outline around an area of the chart to highlight it
- Zoom in button
- Zoom out button
- Autoscale - shows all data in the system for the location for all days
- Reset axes - resets the chart view to its defaults
- Toggle Spike Lines - displays lines from a point that your mouse touches to the X and Y axes
- Show closest data on hover - when selected shows the data from the closest point to your mouse in the hover area.
- Compare data on hover - intended to show comparison between a data point and other items in the chart. Note: *this feature is not configured in our Murakami Viz charts.*
- Produced with Plot.ly - links to the open source charting library being used, Plotly.

Finally, once you have selected the data of interest to you, click the **EXPORT ALL** button at the bottom of the chart area to download the selected test results as a CSV file.

Accessing Data stored in remote locations defined by Murakami GCS or SCP exporters

If you are using the GCS or SCP exporter options, copies of each test result should be available in JSON format in the remote locations you defined in your environment variables.

In the MLBN program, we used the GCS exporter to send all test data from all measurement devices to a central archive. From there we could load the data into BigQuery for our internal analyses.

Depending on the tools, workflows, and expertise of your staff, you might choose to use the SCP exporter and ingest the test data from there into a Postgresql or other database.

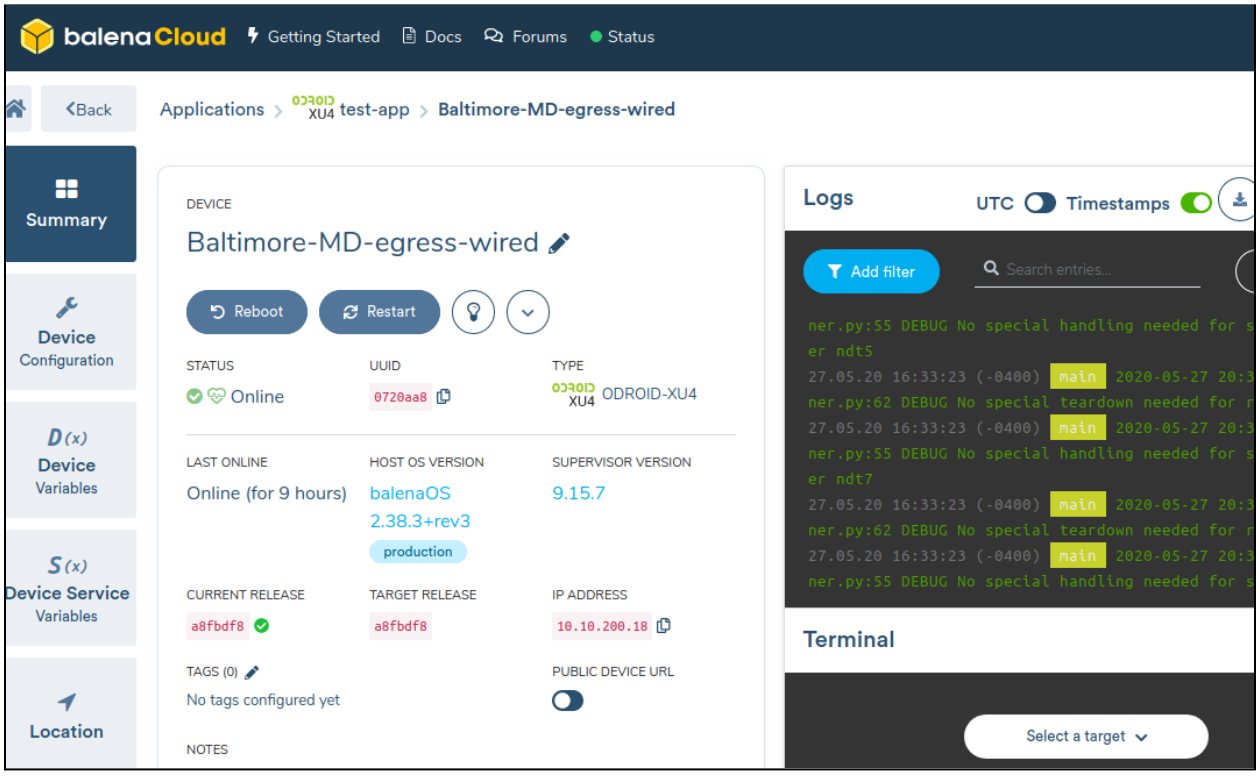
Accessing data on each device

If you have enabled the Local exporter option, copies of each test result will be stored on the filesystem of each Murakami measurement device you have in your system. You can access the data on each device in two ways:

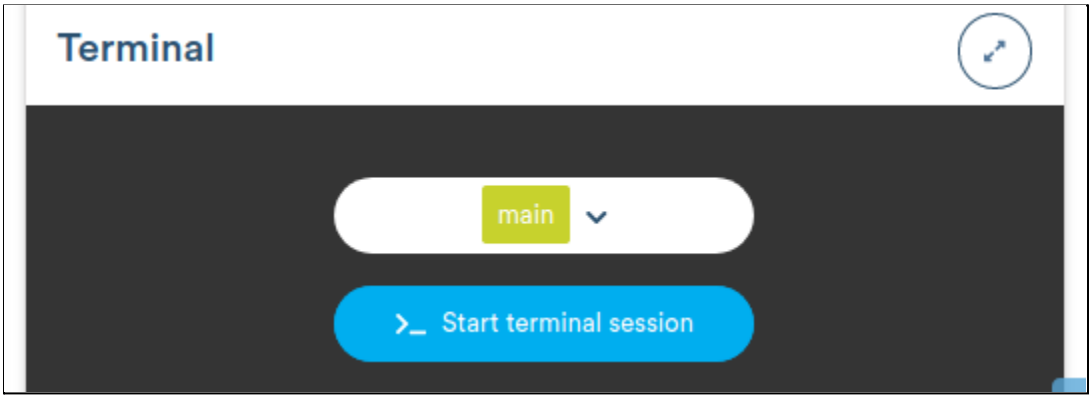
1. Remotely via the Balena Dashboard, if the device is online
2. Locally by accessing each device's SD card directly

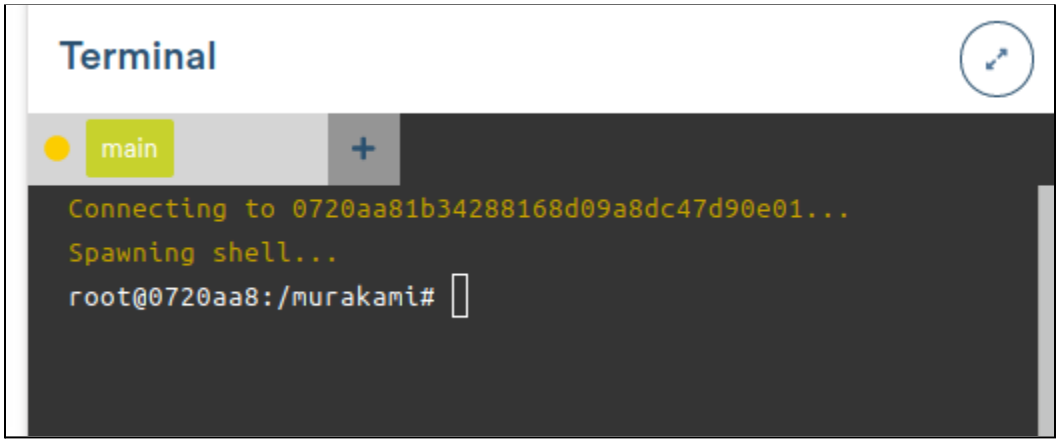
Accessing data via the Balena Dashboard

Navigate to your Balena Cloud application dashboard, and select one of your devices. If you're viewing a page like the one below, but for your device, you're in the right place.



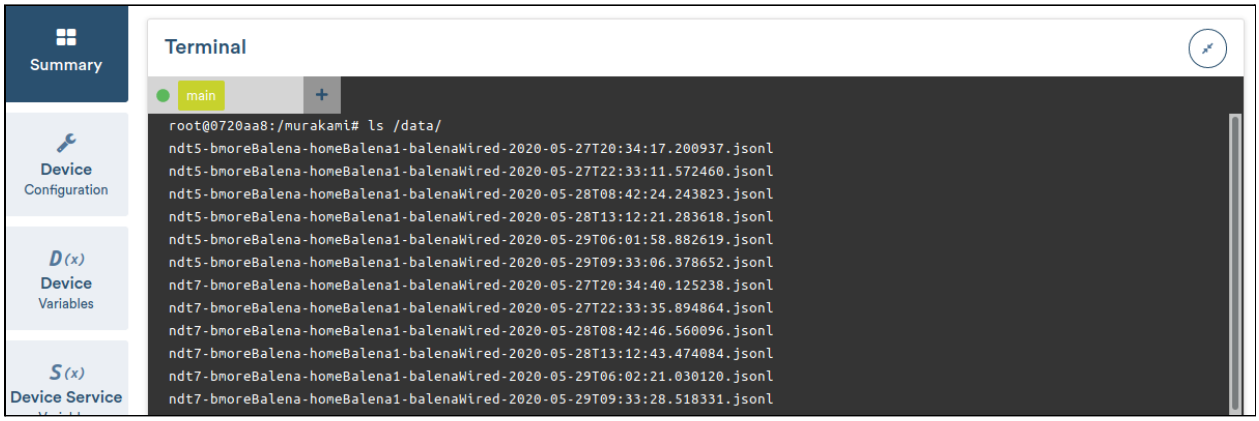
In the section labelled “Terminal”, select “main” to enter the Murakami container on the device, then click “Start terminal session”. Once the terminal session connects, you should see a terminal prompt like the second image below. Use the circle button in the top right to expand the view of the remote terminal.





In the main container, the data your device is collecting is stored by default in `/data/`

If you've defined a different location for your local exporter to store test data, look for it there. Use the command `ls` to show the files in that folder, as seen in the image below:

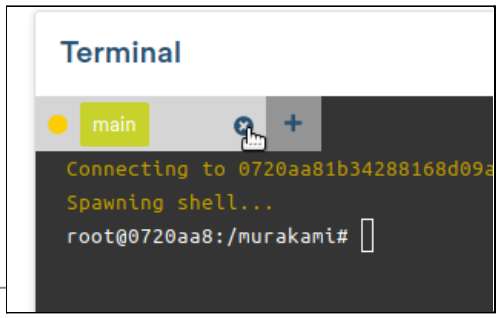


There are a limited number of options for interacting with the test data in the main container context, but viewing it from there is a good way to quickly confirm that test data is being saved to disk.

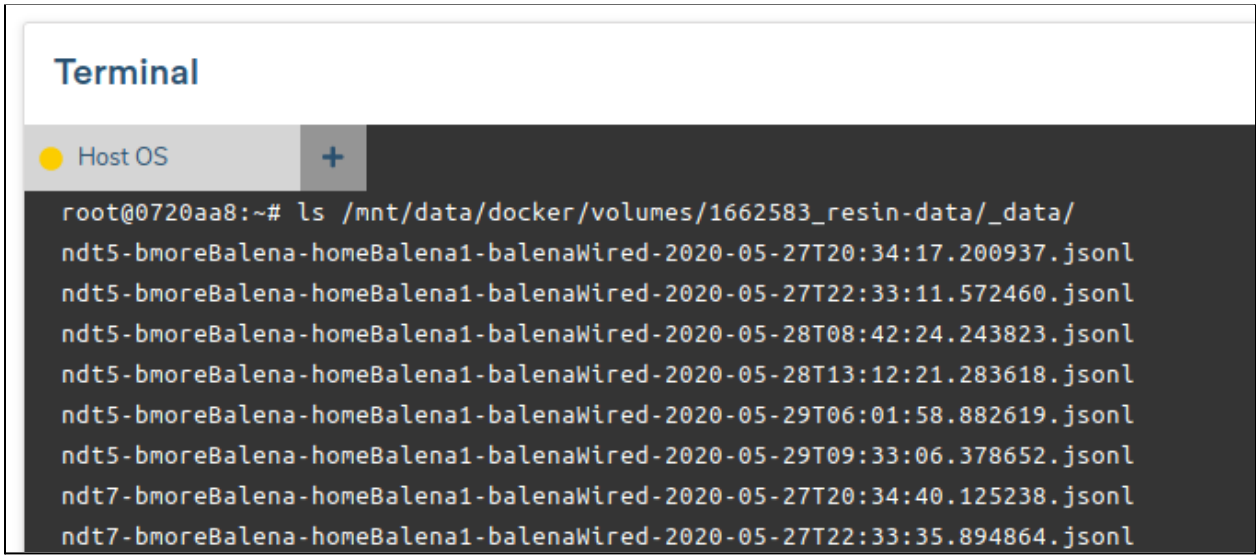
In the "Host OS" context, you will have access to common linux system utilities like `ssh` and `scp`, and can install others that you may commonly use. SCP can be used from the Host OS context to send data to a remote server manually.

To use the Host OS context, close out your "main" container terminal using the X button in the grey area at the top left of the terminal window, as shown in the image on the right.

Then start a new terminal session but select the "Host OS".

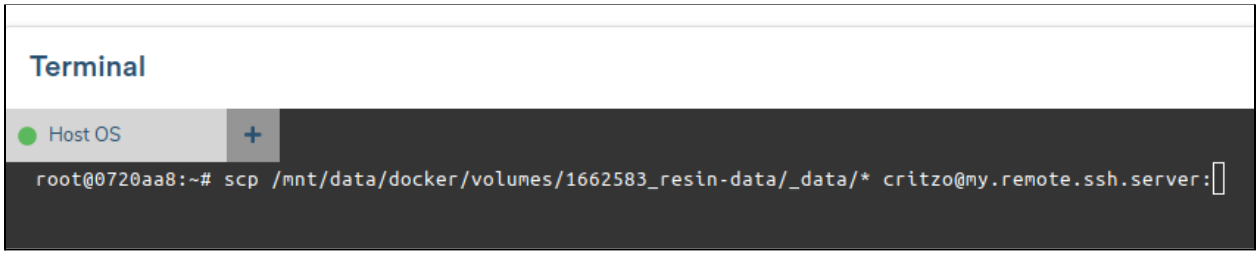


The Host OS context will look very similar, but the path to your data will be different. In the image below, we are listing the test results from one of our MLBN measurement devices.



One part of this path may be different in your Balena device: `1662583_resin-data`
The number, 1662583, is your Balena Application ID.

In the Host OS context, you can use standard linux programs to send data elsewhere, or interact with it on the device. For example you might use the `scp` program to send test data to another computer, as seen in the command below:

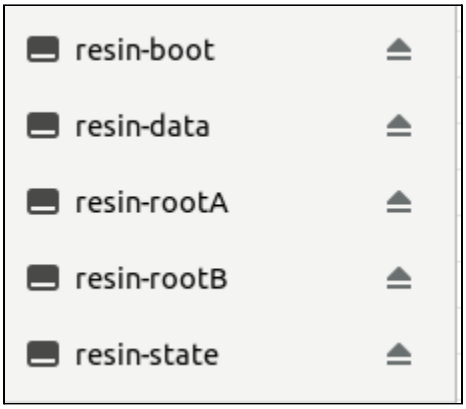


Accessing data locally via the device's SD card

If you have a small scale deployment, or if you prefer to not use some optional Murakami configurations and services like the SCP or GCS exporters, or the Data Visualization Service and HTTP exporter, you can still collect data from each device and optionally convert it to CSV format.

The MLBN program used this method with our first year libraries, before the exporters and visualization service were written. We asked libraries to send back their first year devices, and we then collected the data directly from each device's SD card.

Obtain the SD card from a device that's been collecting test data. Using an SD card reader, open it on your desktop or laptop computer. The SD card has five partitions that will show up on your system. For example, the image on the right shows the SD card partitions on an Ubuntu Linux laptop.



Your test results will be in the resin-data partition. Navigate to your test data files. The path will be similar to the one we reviewed in the section above when accessing the test data from the Balena terminal in the Host OS context. Note that the number preceding `_resin-data` in the path below will be different for each Balena Cloud application:

`resin-data / docker / volumes / 1555306_resin-data / _data /`

You may need to enter your administrator password for your computer to access the `docker` folder.

You should now see your test data files in JSON format and can copy them anywhere you like.

Converting Test Result Data from JSON to CSV

JSON is a commonly used, machine-readable format for exchanging data, configurations, and more. However, you may not be familiar with JSON, and might prefer your data in CSV format.

In this case, the Murakami software includes a script called ***murakami-convert*** that can be used to convert the JSON test result files to CSV format before or after retrieving test files from a device.

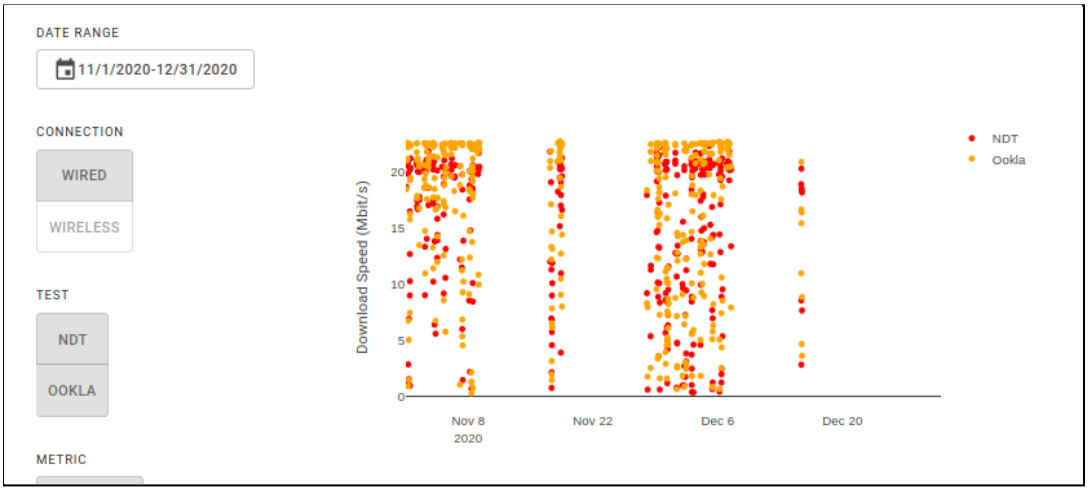
Documentation on how to use *murakami-convert* using Docker is available in the Murakami code repository.

Sending Previously Collected Data to a Murakami Viz Instance

If the Internet connection you're measuring has a dynamic IP address, not a static one, your data in Murakami-Viz will have gaps due to the way Murakami measurement devices and the Murakami-Viz service interact. You also may have run Murakami measurement devices for a bit before setting up an instance of Murakami Viz, and want to import the test data collected into the newly launched instance.

In the image below, you can see what these gaps in measurements look like within Murakami-Viz. Since we know that this connection does not have a static IP address, these gaps likely occurred between the

time when an IP address changed, and the time when the new IP address was whitelisted in Murakami-Viz.



Four gaps in measurements are displayed above: Nov. 9-17, Nov. 19-27, Dec. 7-15, and Dec.16-present. Tests were still conducted by the Murakami measurement device during these times, but were not accepted by Murakami-Viz because the new IP address was not whitelisted.

To enable these use cases, Murakami provides a utility script called ***murakami-upload*** that uses the HTTP exporter and associated environment variables, to upload previously collected results to your instance of Murakami Viz.

Note that for *murakami-upload* to successfully upload results to a Murakami Viz instance:

- all test result files must contain the **MurakamiDeviceID** field
- this device ID must be associated with a measurement device in the Murakami-Viz instance
- the current, public IP address of the computer you'll use to run *murakami-upload* must be whitelisted in the Murakami-Viz instance

How to Use murakami-upload

1. Obtain test data from missing dates
2. Enter or Start an interactive Murakami container containing the missing test data
3. Run *murakami-upload*
4. Confirm data was received by the Murakami-Viz instance

Obtain test data from missing dates

Depending on the export locations configured for your Murakami measurement device, test results may be stored in one or more locations.

On-Device Data

To upload test data that is stored on a Murakami measurement device, first login to the device itself. If this is a Balena Cloud managed device, you can use the terminal through your Balena app or the balena-cli tool to login to the *main* container. If this is a standalone device, SSH to the device. In either case, when you're on the device's terminal, first confirm that *murakami-upload* is a usable program in the Murakami release running on the device.

```
root@d9fa4d8:/murakami# ls scripts/
convert.py  upload.py
root@d9fa4d8:/murakami# which murakami-upload
/usr/local/bin/murakami-upload
```

To learn more about how to use murakami-upload, type: `murakami-upload -h`

```
usage: murakami-upload [-h] [-p PATH] [-u URL]

Uploads JSON results via HTTPExporter
optional arguments:
  -h, --help          show this help message and exit
  -p PATH, -path PATH  input path
  -u URL, -url URL    URL to send JSON data to
```

On a Balena managed device with the local exporter enabled, test result data will be in `/data/`
To upload results from Nov. 9-17, 2020, we might upload all test results from all test runners, one day at a time using a command such as:

```
murakami-upload -p '/data/*2020-11-09*' -u
'https://viz.measuringbroadband.org/api/v1/runs'
```

In the command above, change "https://viz.measuringbroadband.org" to your Murakami-Viz instance's URL or IP address.

When you run the murakami-upload script, the environment variables available on the device are printed in the terminal output, followed by successful status messages such as those listed below:

```
Reading path /data/*2020-11-09*

Files:
['/data/speedtest-cli-multi-stream-mlab-testbed-home-wired-2020-11-09
T17:49:02.633159.jsonl',
'/data/ndt5-mlab-testbed-home-wired-2020-11-09T17:48:09.359179.jsonl'
,
'/data/ndt7-mlab-testbed-home-wired-2020-11-09T17:48:36.805318.jsonl'
,
'/data/speedtest-cli-single-stream-mlab-testbed-home-wired-2020-11-09
T17:49:45.946055.jsonl',
'/data/ndt7-mlab-testbed-home-wired-2020-11-09T17:46:34.017050.jsonl'
]
```

Within a few minutes of completion, the uploaded data should be available in your instance of Murakami-Viz.

Data from GCS or SCP Server Archive

If you've exported your data to GCS or an SCP server, you can use *murakami-upload* from a Docker container on any computer to send it to a running instance of Murakami-Viz.

First, obtain a local copy of the test data you wish to upload to your instance of Murakami-Viz. For demonstration purposes, we have created a folder called `murakami-upload` and a subfolder, `data`, on our local computer. We then copied some test result files from GCS into the data folder.

Next, we will run a Murakami container locally (not on a test device, but on your computer), and mount the `data` folder as a volume within it. The command below is an example which you can customize for the paths to your `data` folder:

```
docker run -it --entrypoint /bin/bash --volume
/path/to/test-results:/data/ measurementlab/murakami:latest
```

The command pulls the latest Murakami image from M-Lab's Dockerhub if needed, and once completed will provide a terminal inside the container. You can then confirm the data you plan to upload is available in the container, and finally upload it to your Murakami-Viz server with the same command as in the previous section. Below is an abbreviated log of terminal output showing these steps.

```
Unable to find image 'measurementlab/murakami:latest' locally
latest: Pulling from measurementlab/murakami
6c33745f49b4: Pull complete
ef072fc32a84: Pull complete
...
e72eb1440e38: Pull complete
Digest:
sha256:30a762371c521b658b75d274c6561481d5244e3bda5e90778279129015673b
0e
Status: Downloaded newer image for measurementlab/murakami:latest
root@0caf99b798d6:/murakami#
root@0caf99b798d6:/murakami# ls /data
ndt5-mlab-testbed-home-wired-2020-11-09T09:27:26.197005.jsonl
ndt5-mlab-testbed-home-wired-2020-11-09T11:06:35.118214.jsonl
ndt5-mlab-testbed-home-wired-2020-11-09T12:48:38.150086.jsonl
...
murakami-upload -p '/data/*2020-11-09*' -u
'https://viz.measuringbroadband.org/api/v1/runs'
```

Within a few minutes of completion, the uploaded data should be available in your instance of Murakami-Viz.

Transferring Balena Managed Devices to Another Application/Owner

You may have need to transfer one or more Balena.io managed measurement devices to another person or organization. For example, at the conclusion of the MLBN research program, the measurement devices placed in partner libraries were transferred to those libraries if desired.

The *new* device administrator must first create a Balena.io account and organization, and aaadd the *current* device administrator as a member of a newly created application with a **Developer** role. Finally, the current administrator must log in, and select **Applications > Transfer This Application** and select the target organization from the list to complete the transfer. More information can be found in the Balena Documentation.

Section 5: Glossary

Overview of Section 5: Glossary

This section of the document contains the technical terms and definitions used in this training manual.

Active tests - Tests that are initiated by a person, or automated by a person or organization. The full consent of the person or organization running the tests is required.

Balena Cloud - Management software for connected devices that are part of the Internet of Things (IoT).

Bandwidth - The capacity for data transfer of an electronic communications system.

BBR - Bandwidth, Bottleneck, and Round trip time.

Beta test - A trial of machinery, software, or other products, in the final stages of its development, carried out by a party unconnected with its development.

Data Visualization Platform - Service or software that allows people to understand the significance of data by placing it in a visual context.

Ethernet Cable - One of the most common types of network cables used to connect devices within a local area network like PCs, routers, or switches.

Firewalls - A part of a computer system or network that is designed to block unauthorized access while permitting outward communication.

Hops - In a packet-switching network, a hop is the trip a data packet takes from one router or intermediate point to another in the network. On the Internet (or a network that uses TCP/IP), the number of hops a packet has taken toward its destination (called the "hop count") is kept in the packet header. A packet with an exceedingly large hop count is discarded.

Internet exchange points, or IXPs - An Internet exchange point (IX or IXP) is a physical infrastructure through which Internet service providers (ISPs) and content delivery networks (CDNs) exchange Internet traffic between their networks.

Internet of Things (IoT) - Interconnection via the Internet of computing devices embedded in everyday objects, enabling them to send and receive data.

OR: the concept of basically connecting any device with an on and off switch to the Internet (and/or to each other). This includes everything from cell phones, coffee makers, washing machines, headphones, lamps, wearable devices, etc. (https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#601f8fec1d09)

Koha - First open-source Integrated Library System (ILS).

Latency - The delay before a transfer of data begins following an instruction for its transfer.

LibLime - Web-based, cloud-hosted library management services.

Local Area Network (LAN) - A computer network that spans a relatively small area.

Micro SD - A type of very small memory card typically used in mobile phones and other portable devices.

Network Diagnostic Tool - A client/server program that provides network configuration and performance testing to a users desktop or laptop computer.

Network switch - A network switch is a central communication device for local area Ethernet networks (AKA switching hub, bridging hub, officially MAC bridge).

ODROID - A series of single-board computers and tablet computers created by Hardkernel Co., Ltd., located in South Korea.

Ookla - Provider of a widely used commercial bandwidth and latency test provided by Ookla. We are including this popular test to compare the Ookla results with the M-Lab results.

Open Balena - An open-source version of the Balena software.

Open source - Denoting software for which the original source code is made freely available and may be redistributed and modified.

Passive testing or monitoring - A software testing technique that observes the system without interaction.

Power Supply - A hardware component that supplies power to an electrical device.

Server - A server is a computer that provides data to other computers. Many types of servers exist, including web, mail, and file servers.

Software as a service (SaaS) - A method of software delivery and licensing in which software is accessed online via a subscription, rather than bought and installed on individual computers.

Wide Area Network (WAN) - A telecommunications network that extends over a large geographical area for the primary purpose of computer networking.

WiFi Measurement Computer - An Odroid computer that has been configured to take measurements of a WiFi access point.

Wired Measurement Computer - An Odroid computer that has been configured to take measurements on the wired network.

Appendices

Appendix A: The MLBN Team

There are a number of people contributing their time and expertise to the MLBN Project, including a Board of Advisors; Simmons University; Measurement Labs (M-Lab); Internet 2; and other partners. To get in touch with the team, send an email to: **mlbn@measurementlab.net**.

Board of Advisors

Larra Clark Deputy Director Office of Information Technology Policy (OITP), American Library Association (ALA)
James Deaton Executive Director, Great Plains Network (GPN)
Amina Fazlullah Mozilla Policy Fellow, Policy Advisor - National Digital Inclusion Alliance
Dr. Jon Gant Dean, Library and Information Sciences, North Carolina Central University
Susan Hildreth Strategic Advisor, Califa
Dylan Baker Broadband Consultant, Idaho Commission for Libraries
Jarrid Keller Assistant Director for Infrastructure, Sacramento Public Library
Paul Kissman Library Information Systems Specialist, Massachusetts Board of Library Commissioners
Christopher Mitchell Director, Community Broadband Networks Initiative, Institute for Local Self-Reliance
Sarah Morris Director, New America's Open Technology Institute
Angela Siefer Director, National Digital Inclusion Alliance
Sharon Strover Philip G. Warner Regents Professor of Communication & Director of Technology and Information Policy Institute, University of Texas in Austin

Simmons University

Simmons University (www.simmons.edu) is a nationally recognized private university located in the heart of Boston. Founded in 1899, Simmons is the only undergraduate women's college in Boston, and maintains a history of visionary thinking and a focus on social responsibility. The college offers world-class coeducational graduate programs in nursing and health sciences, including physical therapy and nutrition; education; behavior analysis; library and information science; management; and social work. The School of Library and Information Science started the same year as the College and may be the only American library school so related to the origin of its parent institution.

Colin Rhinesmith - *Ph.D., Principal Investigator, Assistant Professor, School of Library and Information Science, Simmons University.*

Dr. Rhinesmith's research has focused on the role of public libraries in promoting digital inclusion and broadband adoption in rural, suburban, and urban communities across the U.S. His recent work with researchers at the University of Texas at Austin

and Oklahoma State University funded by the IMLS has examined how rural libraries address the challenges of Internet connectivity with hotspot lending programs. Rhinesmith's research has been published in several top scholarly journals, including Government Information Quarterly; Information, Communication & Society; Telecommunications Policy and Public Library Quarterly.

Susan Kennedy - *Research Assistant, School of Library and Information Science, Simmons University.*
Susan received her BA from Bennington College, and is currently working toward her MLIS at Simmons. Her focus is on public library leadership, advocacy, and policy, and she is particularly passionate about the role public libraries play in promoting digital equity and information literacy.

Jo Dutilloy - *Research Assistant, School of Library and Information Science, Simmons University.*
Jo received a BA in Comparative Literature from Bryn Mawr College and is working towards a dual Master's degree in Archives and History at Simmons. They have been passionate about digital stewardship and public access to information since their summer as a Digital Humanities Intern at the Library Company of Philadelphia. Jo continues to pursue the development of open and equitable access to information in this and other projects.

Code for Science and Society / Measurement Lab

Code for Science and Society (https://codeforscience.org) is a US-based 501(c)(3) nonprofit supporting open collaboration in public interest technology through fiscal sponsorship and other programs supporting sustainable open source. Measurement Lab (M-Lab) is an open source project with contributors from civil society organizations, educational institutions, and private sector companies, and is a fiscally sponsored project of Code for Science & Society. M-Lab provides a global infrastructure enabling academically vetted Internet measurement tests to be run by people and for the resulting data to be published in the public domain. The general public running M-Lab tests gain information about the speed and quality of service of their Internet connections, and a rich, longitudinal, and freely available dataset of all tests conducted via M-Lab is made available for research.

Georgia Bullen - *Executive Director, Simply Secure.*
Georgia Bullen has been an advocate in the internet health movement through her work and passion around issues such as net neutrality, security, privacy, and equitable access to technology. Her work has focused on the intersection of human-centered design, urban space, and technology – specifically, how technology intersects with

human rights, e.g access to information and the right to communicate. She has a background in human-centered design, data visualization, urban planning and software development, and is an advocate for diversity in technology.

Chris Ritzo - *Program Management & Community Lead at Measurement Lab, Code for Science & Society.*
Chris Ritzo supports researchers, policy makers, advocacy groups, and individuals interested in M-Lab's open Internet measurement data. Chris' research examines technology's impact on and use within communities, particularly within public institutions such as schools and libraries. His work is grounded in experiences within the maker movement, community organizations, grassroots media, cooperatives, community access television, and indymedia. Chris holds Master's degrees in information science and communication from the University of Illinois.

Internet2

Internet2® (https://www.internet2.edu/) is a non-profit, member-driven advanced technology community founded by the nation's leading higher education institutions in 1996. Internet2 serves 328 U.S. universities, 60 government agencies, 43 regional and state education networks and through them supports more than 100,000 community anchor institutions, over 900 InCommon participants, and 69 leading corporations working with our community, and 61 national research and education network partners that represent more than 100 countries.

Internet2 delivers a diverse portfolio of technology solutions that leverages, integrates, and amplifies the strengths of its members and helps support their educational, research and community service missions. Internet2's core infrastructure components include the nation's largest and fastest research and education network that was built to deliver advanced, customized services that are accessed and secured by the community-developed trust and identity framework.

Stephanie Stenberg - *Director, Community Anchor Program (CAP), Internet2.*
Stephanie Stenberg works with regional networking partners, community anchor institutions, and Internet2 membership organizations to support the mutually beneficial goals of bringing networking, trust and identity services, and advanced applications to community anchor institutions nationwide. CAP works with state and regional research and education networks across the country to connect the full range of community anchor institutions to advanced broadband capabilities.

Carson Block - *President, Carson Block Consulting Inc*. http://www.carsonblock.com/
Carson previously worked with Internet2 on the Toward Gigabit Libraries project (http://www.internet2.edu/tgl/) and joined the MLBN team to assist with continuity between the projects and write a training manual for MLBN participants.

Carson Block Consulting Inc. is a full-service library technology consulting firm established as Carson Block LLC in 2010, and incorporated as Carson Block Consulting Inc. in 2012. Primary areas of consulting including library technology planning; facilities and services master planning; IT Department audits, assessments, and improvement plans; and teaching, training, facilitation and public speaking on library technology and general library advocacy topics.

Carson Block has been a library technologist for more than 25 years – as a library worker, IT Director and now a Library Technology Consultant. He has been called "a geek who speaks English" and enjoys acting as a bridge between the worlds of librarians and hard-core technologists. He has a passion to de-mystify technology for the uninitiated, and to help IT professionals understand and support the goals of libraries.

Institute of Museum and Library Services

The Institute of Museum and Library Services (https://www.imls.gov/) is the primary source of federal support for the nation's approximately 120,000 libraries and 35,000 museums. Their mission is to inspire libraries and museums to advance innovation, lifelong learning, and cultural and civic engagement. Their grant making, policy development, and research help libraries and museums deliver valuable services that make it possible for communities and individuals to thrive.

Pilot Libraries - Year One

The MLBN team would like to thank the year one pilot libraries for their participation in the program. The efforts of these libraries allowed for testing and refinement of technology and processes for the MLBN Project.

Hollis Public Library - Hollis, AK

Ventura County Library - Ventura County, CA

Pasco County Public Library Cooperative - Pasco County, FL

Twin Falls Public Library - Twin Falls, ID

Truro Public Library - North Truro, MA

Saint Paul Public Library - Saint Paul, MN

Thomas J. Harrison Pryor Public Library - Pryor, OK

Multnomah County Library - Portland, OR

Bennington Free Library - Bennington, VT

Westchester County Library - Westchester, NY

Participating Libraries - Year Two

The MLBN team would like to thank the libraries who participated in year two.

Saline County Library - Benton, AR

Grand County Library - Sheridan, AR

Arkansas River Valley Regional Library - Dardanelle, AR

Safford City-Graham County Library - Safford, AZ

Elfrida Library - Elfrida, AZ

Denver Public Library - Denver, CO

The Ferguson Library - Stamford, CT

Hall County Library System - Gainesville, GA

Live Oak Public Libraries - Savannah, GA

Cherokee Public Library - Cherokee, IA

Estherville Public Library - Estherville, IA

Waltham Public Library - Waltham, MA

Memorial Hall Library - Andover, MA

Clarkston Independence District Library - Clarkston, MI

Monroe County Library System - Temperance, MI

Traverse Area District Library - Traverse City, MI

Caruthersville Public Library - Caruthersville, MO

Gunn Memorial Public Library - Yanceyville, NC

Avery Mitchell Yancy (AMY) Regional Library - Newland, NC

The Public LIbrary for Union County - Lewisburg, PA

Appendix B: Example of data generated by the Measurement Computers

Do you want to peek under the hood to see what the measurement devices are collecting and sending? Here's an example!

NDT (version 5)

Description:

NDT is a single stream performance measurement of a connection's capacity for "bulk transport" (as defined in IETF's RFC 3148. NDT measures "single stream performance" or "bulk transport capacity". Version 5 of the NDT test uses the CUBIC TCP congestion control algorithm, and runs over non-standard TCP ports. More information is available at: https://www.measurementlab.net/tests/ndt/ndt5/

Sample data:


```
{
        "ServerName": "ndt-iupui-mlab2-iad05.measurement-lab.org",
        "ServerIP": "38.90.140.139",
        "ClientIP": "104.145.221.196",
        "MurakamiLocation": "<MURAKAMI 'LOCATION' ENV VAR>",
        "MurakamiNetworkType": "<MURAKAMI 'NETWORK_TYPE' ENV VAR>",
        "MurakamiConnectionType": "<MURAKAMI 'CONNECTION_TYPE' ENV VAR>",
        "TestUUID": "ndt-zv8fb_1580286045_0000000000000F33",
        "TestProtocol": "ndt5",
        "DownloadTestStartTime": "2020-02-18 20:32:52.639720903 +0000 UTC m=+5825.539350534",
        "DownloadTestEndTime": "2020-02-18 20:33:02.640686711 +0000 UTC m=+5835.540316330",
        "DownloadValue": 18.06056116735436,
        "DownloadUnit": "Mbit/s",
        "UploadValue": 4.773627705349652,
        "UploadUnit": "Mbit/s",
        "DownloadRetransValue": 0.16013824826395937,
        "DownloadRetransUnit": "pct",
        "MinRTTValue": 22.826,
        "MinRTTUnit": "ms"
}
```

NDT (version 7)

Description:

NDT is a single stream performance measurement of a connection's capacity for "bulk transport" (as defined in IETF's RFC 3148. NDT measures "single stream performance" or "bulk transport capacity". Version 7 of the NDT test uses the BBR TCP congestion control algorithm where possible, falling back to CUBIC, and runs over standard TCP ports 443 or 80. Additionally, ndt7 uses standards-based methods to estimate the full capacity of the end-to-end link between client and server. More information is available at: https://www.measurementlab.net/tests/ndt/ndt7/

Sample data:


```
{
        "ServerName": "ndt-iupui-mlab2-iad05.measurement-lab.org",
        "ServerIP": "38.90.140.139",
        "ClientIP": "104.145.221.196",
        "MurakamiLocation": "<MURAKAMI 'LOCATION' ENV VAR>",
        "MurakamiNetworkType": "<MURAKAMI 'NETWORK_TYPE' ENV VAR>",
        "MurakamiConnectionType": "<MURAKAMI 'CONNECTION_TYPE' ENV VAR>",
        "TestUUID": "ndt-zv8fb_1580286045_0000000000000F33",
        "TestProtocol": "ndt7",
        "DownloadTestStartTime": "2020-02-18 20:32:52.639720903 +0000 UTC m=+5825.539350534",
        "DownloadTestEndTime": "2020-02-18 20:33:02.640686711 +0000 UTC m=+5835.540316330",
        "DownloadValue": 18.06056116735436,
        "DownloadUnit": "Mbit/s",
        "UploadValue": 4.773627705349652,
        "UploadUnit": "Mbit/s",
        "DownloadRetransValue": 0.16013824826395937,
        "DownloadRetransUnit": "pct",
        "MinRTTValue": 22.826,
        "MinRTTUnit": "ms"
}
```

Speedtest-cli (single and multi-stream)

Description:

Speedtest-cli is a community-developed command line interface for testing internet bandwidth using the speedtest.net/Ookla platform. More information is available at: https://github.com/sivel/speedtest-cli

Sample data:


```
{
        "MurakamiLocation": "<MURAKAMI 'LOCATION' ENV VAR>",
        "MurakamiNetworkType": "<MURAKAMI 'NETWORK_TYPE' ENV VAR>",
        "MurakamiConnectionType": "<MURAKAMI 'CONNECTION_TYPE' ENV VAR>",
        "TestProtocol": "speedtest-cli-multi-stream",
        "download": 20255092.280781172,
        "downloadUnits": "Bits/s",
        "upload": 4461811.659646774,
        "downloadUnits": "Bits/s",
        "ping": 20.901,
        "pingUnits": "ms",
        "serverUrl": "http://ookla1.portnetworks.net:8080/speedtest/upload.php",
        "serverLat": "39.2833",
        "serverLon": "-76.6167",
        "serverName": "Baltimore, MD",
        "serverCountry": "United States",
        "serverCountryCode": "US",
        "serverSponsor": "Port Networks",
        "serverId": "8987",
        "serverUrl2": "http://ookla2.portnetworks.net/speedtest/upload.php",
        "serverHost": "ookla1.portnetworks.net:8080",
        "serverD": 2.1472729314798276,
        "serverLatency": 20.901,
        "serverLatencyUnits": "ms",
        "timestamp": "2020-01-28T14:52:36.952200Z",
        "bytes_sent": 5750784,
        "bytes_received": 25442834,
        "share": null,
        "clientIp": "104.145.220.198",
        "clientLat": "39.2645",
        "clientLon": "-76.6224",
        "isp": "Port Networks",
        "isprating": "3.7",
```

```
        "rating": "0",
        "ispdlavg": "0",
        "ispulavg": "0",
        "loggedin": "0",
        "country": "US"
}
```

Appendix C: Pre-Installation Questions

Before you receive your Measurement Computers, you will receive a questionnaire from the MLBN team (called the *Information Sheet and Questionnaire - Measuring Library Broadband Networks - Year 2 Libraries*) asking some **questions about your technology environment**, and making **suggestions for configuring your library data network** to allow the Measurement Computers to work properly.

The questions are provided in this appendix for possible future reference.

MLBN Measurement Device Installation

1. For measuring your library's public WiFi network, our measurement device can be connected using WiFi like a patron might, or connected to the WiFi access point or WiFi controller using ethernet. Which method would you like to use at your library?
    a. #1 - Connect over WiFi like a patron - measuring what an individual patron would experience
    b. #2 - Connect to the WiFi access point or controller - measuring the overall bandwidth that an access point would provide to multiple patrons connecting to it
2. If you selected #1 above: What is the name (SSID) of the WiFi network that your patrons use? Please provide the exact name, including any capitalization, spaces, or special characters.
3. If you selected #1 above: If a password for WiFi is required, what is the password?
4. If a password for WiFi is required, what type of security is used?
5. If you selected #1 above: Does this WiFi network require that patrons view a splash page and accept terms of use?
6. If the answer to the previous question was YES, can you allow our WiFi measurement device to connect to the Internet without requiring a splash page and/or acceptance of terms of use?
7. Do you wish us to configure static IP addresses on any of our measurement devices?
8. If you answered YES to needing static IP addresses, please provide the two IP addresses our devices should use, the subnet size, and the IP address of the gateway for each device (if they are different).
9. Do you use DNS servers other than Google DNS? If so, please provide the IP addresses of the DNS servers you require to be used in your network.
10. Our measurement devices and management software require a range of Internet ports to be open in order to function properly within your network. Are you able to configure your firewall to allow the following ports? Please check to indicate yes, or leave unchecked to indicate no.
    a. 3001-3010, 32768-65535 TCP
    b. 22 TCP
    c. 443 TCP
    d. 123 UDP
    e. 53 UDP

11. If your library network employs URL blocking or filtering, can your staff allow the following domains for our measurement devices? Please check to indicate yes, or leave unchecked to indicate no.
    a. *.measurement-lab.org
    b. *.balena-cloud.com
    c. *.docker.com
    d. *.docker.io
12. Do you wish to be provided the MAC address for each measurement device?

Network Management Practices

1. Who manages the networked services and connectivity for your library or libraries? *
    a. IT staff employed by the library
    b. Library staff with technology experience
    c. ISP and/or other vendors
    d. IT staff employed by the municipality, library system, or other agency/provider
    e. Other:
2. If you answered "other" above, please provide more details.
3. What network management practices or analysis tools does your organization employ to understand the usage and capacity of your network? For example, is there a per-device bandwidth limit imposed? Please explain.
4. Tell us about your firewall and security practices at your library or libraries. For example, what ports/protocols are blocked or filtered? Please explain.
5. Does your organization use a vendor provided WiFi system to provide wireless internet access at your library or libraries? Please answer yes or no, and provide the name of the vendor.
6. Does your organization use or subscribe to any other vendor systems for managing or monitoring broadband at your library or libraries? Please explain.

When completed, please email your responses to the MLBN team: mlbn@measurementlab.net

Appendix D: Using MLBN Measurement Devices After the Program Ends

The MLBN team planned to allow participating libraries to continue using the small measurement computers that were used in the program for their own purposes, to continue collecting measurement data, or choose to return the devices to the MLBN team. Near the conclusion of the program, the MLBN team contacted all participating libraries to identify their preferences and coordinated with them to support their choice. This section documents each option.

Re-purpose the measurement devices for your own use

If your library has a use for the small computers, such as maker-focused activities or otherwise, no action is needed other than letting us know your preference. Our team is happy to talk with you about possible uses if you would like.

Continue using the measurement devices to measure Internet service

If you wish to continue using the measurement computers as we have during this research program, you have a couple of options. The measurement computers used in the MLBN research program were managed using Balena.io, an Internet of Things device management platform. You can choose to manage your device using the Balena.io service as well, or you can run the measurement software on the device without Balena.

Managing your Devices with Balena.io

If you wish to use Balena.io to manage your device(s), the MLBN team can transfer those devices to your Balena.io account and application. You will need to sign up for a Balena.io free account, and set up an application. Then let the MLBN team know these details and we'll transfer the device to you.

You can also set up each device from scratch using the steps in this guide.

Running the Measurement Software on the Device without Balena.io

It is possible to install and run the Murakami measurement software on the devices MLBN provided without using Balena.io. Complete information about how to do this can be found in the *Murakami Deployment Scenarios* section in the Murkami documentation. Follow the guide for a *Standalone* deployment.