

# ourProbes Manual

A network Capabilities monitoring System

For V2.3

[LupinCorp.com](http://LupinCorp.com)

David Somerville

11 March 2020



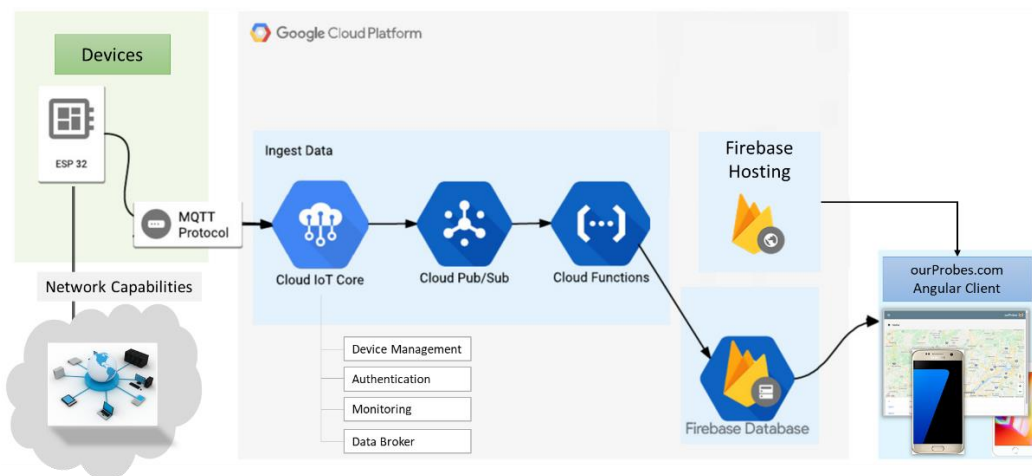
## Contents

Overview .....	2
Workflow.....	2
B: ourProbes Activities.....	4
C: Device List .....	4
E/G: Create/Update Device information .....	5
J: Probe List .....	8
L,N Create/Update/Delete Probes .....	8
Q: Data Analysis .....	10
S: Overview Tab .....	10
T: Trends Tab.....	12
U: Data Extract Tab .....	13
P: My Profile.....	14
V: User Management .....	15
Signing In .....	17
Technical .....	19
Data (Firestore Document Model).....	19
Security .....	20
Data Extract Examples .....	22
Provisioning a New Device .....	23
Creating RSA keys.....	23
Create the Device Definition in ourProbes .....	23
Download and run the Provisioning files and Job.....	23

## Overview

The ourProbes system is used to manage a distributed network monitoring devices, and the collection and analysis of the measurements provided by those devices. The distributed devices include the use of low cost esp32 microcontrollers communicating and managed via a Google IOT cloud service. A responsive, Angular based, web/mobile client is provided to manage the IOT service and the data (measurements) collected from the devices.

- Overall system management using an Angular based web client.
- Low cost microcontroller-based devices. Streamlined and consistent device provisioning process.
- Leverage Google cloud services (IOT Core, Firebase) for simple, reliable and distributed middleware services.
- Full development documentation and all code is opensourced. All code in GitHub and free to use.

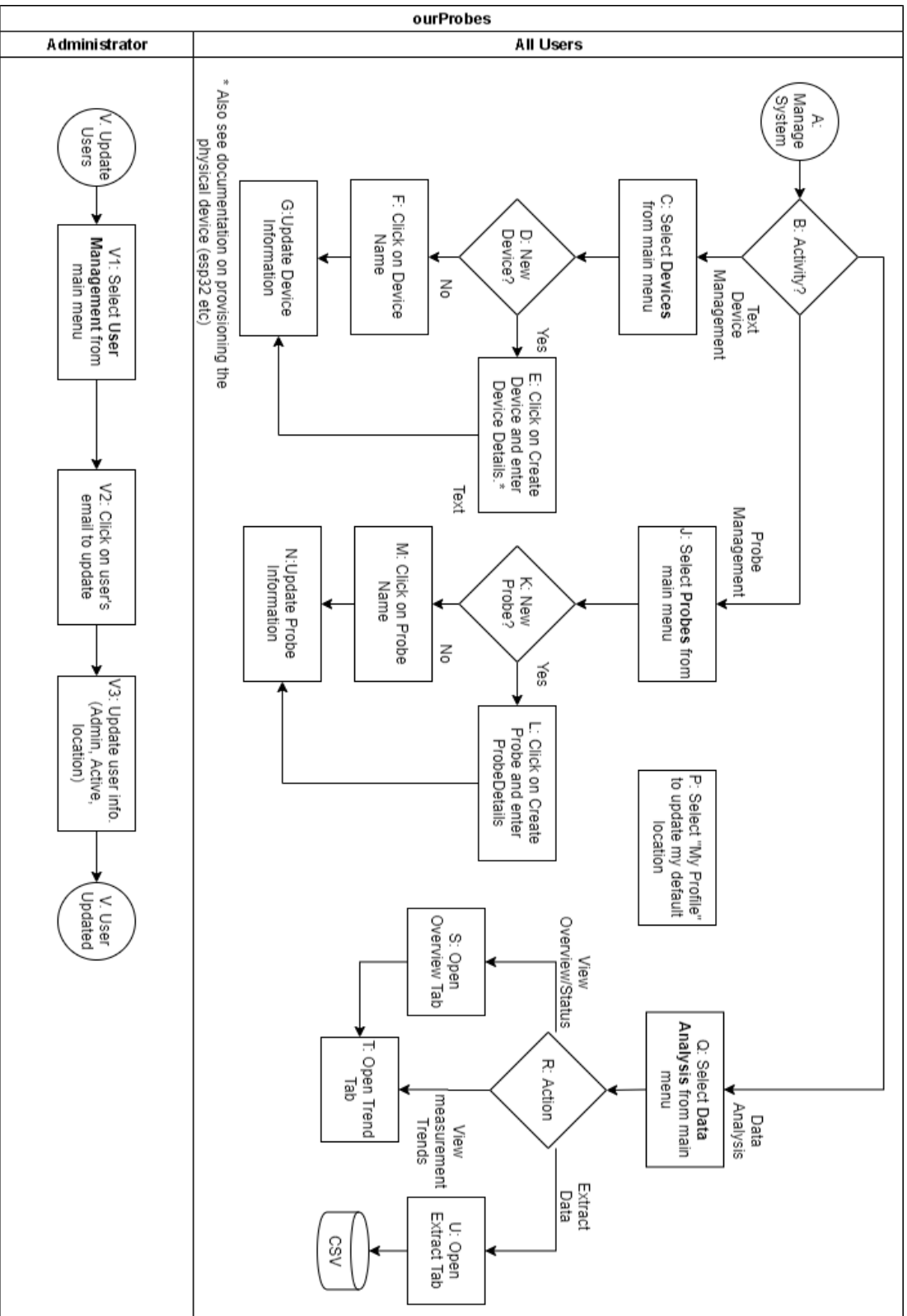


## Workflow

The typical user operations are

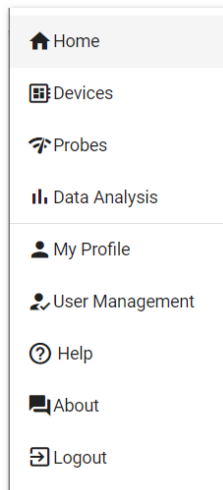
- Create and Manage devices (Including provisioning new devices). Assignment of probes to devices.
- Create and manage probes
- Data analysis
- Strong/flexible user authentication and authorization processes. Administrators can manage other user properties: Active Status, Administrator Rights and default user location.

See below for overall workflow



## B: ourProbes Activities

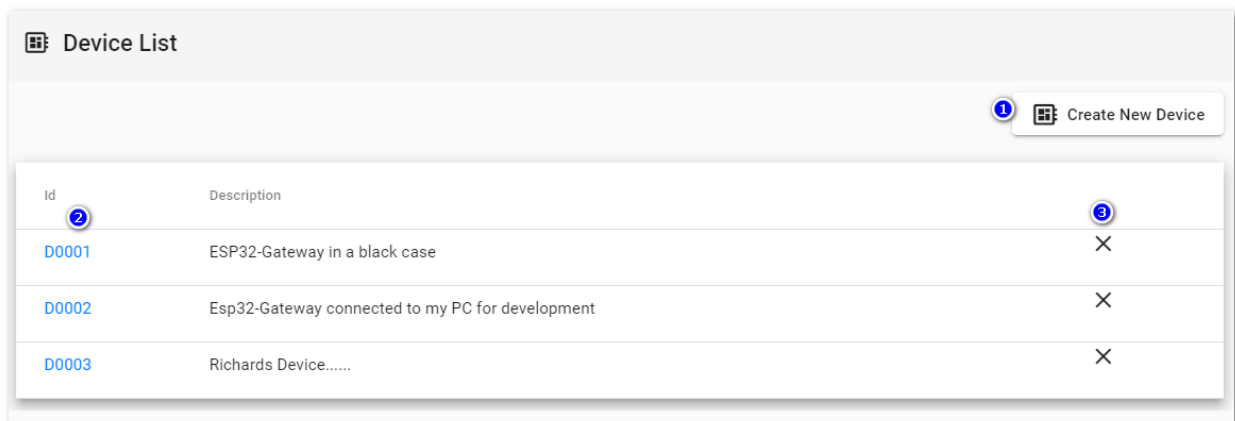
The ourProbes client allows management of system functions from the main menu. The main menu is always available from the dropdown located in the top left of all the pages.



The main user activities are C: Manage Devices, J: Manage Probes and Q: Data Analysis. As well users can manage their default location info using P: My Profile and administrative users can manage other user information and rights using the V: User Management option.

## C: Device List

Clicking on the Devices menu item will take you to the “Device List” page where you can; 1-Create a New Device, 2-Edit a device or 3-Delete a device. All these activities will percolate through the system to update information in the Google Cloud IOT configuration (See technical)



On this form you have the choice to:

1. Create a new Device definition
2. Edit properties of an existing device
3. Delete a device.

## E/G: Create/Update Device information

Depending on the action selected on the Device List form the Device form will be presented. The form will be in Create/Update or Delete mode. The full form in Create mode is shown below.

The screenshot shows the 'Create New Device' form in the 'ourProbes' application. The form is titled 'Create New Device' and includes the following fields and sections:

- Device Id \*** (1): A text input field containing 'D0000'.
- Device Description \*** (2): A text input field.
- Location details** (3): A text input field.
- Type** (4): A dropdown menu showing 'esp32GatewayOlimex'.
- Governance Seconds** (5): A dropdown menu showing '300'.
- The type of device.**: A label for the device type.
- IOT Communication Allowed?** (6): A checkbox that is checked.
- Run Probes?** (7): A checkbox that is unchecked.
- Keys** (8): A section containing three text input fields:
  - Public Key \*** (9): A text input field.
  - Private Key \*** (10): A text input field.
  - Private Key Tuple \*** (11): A text input field.
- Location** (12): A dropdown menu.
- + Create** (13): A button to create the device.

The information captured in this form is

1. Device ID: Required and is expected to start with a letter and followed by 4 numeric characters.
2. Device Description: Required, will be shown on device lists and device maps

3. Location Details: Optional – more details about where the device can be found (i.e. “Bottom of the 3<sup>rd</sup> server rack on the left”)
4. Type: Required, select the hardware that matches the physical device.
5. Governance Seconds: Set between 300 seconds (5 minutes) and 3600 seconds. This is the time between each new loop of probes are run on the device.
6. IOT communications Allowed. This is a direction to the cloud IOT service that determines if any communications is allowed between Cloud IOT and the selected device. This is useful if a device becomes compromised and you want to make sure no data is sent or received from the device.
7. Run Probes: Determines if the device will run the probes assigned to the device. Useful if you want to temporarily stop the probes running on the device.
8. Keys: An expansion panel used to enter the public/private and tuple formatted version of the private key for the device. (See Provisioning a New Device for more information on how to create these)
9. Public Key
10. Private Key
11. Private Key (Tuple format)
12. Location: Expand to specify the Latitude and Longitude for the device location. This is used when displaying the devices on a map.
13. Action Button (Create/Delete) – This is dependent on the mode of the form. This becomes active when the form fields have valid values. Note: An update button is not shown, all updates are real time and occur as field content is changed.
14. Return to device List.
15. Main menu.




☒ Run Probes?

Keys **1**

Location **2**

Latitude  
40.174819021900326

Longitude  
-75.30212153308668



Generate & Download config.py for this device **3**

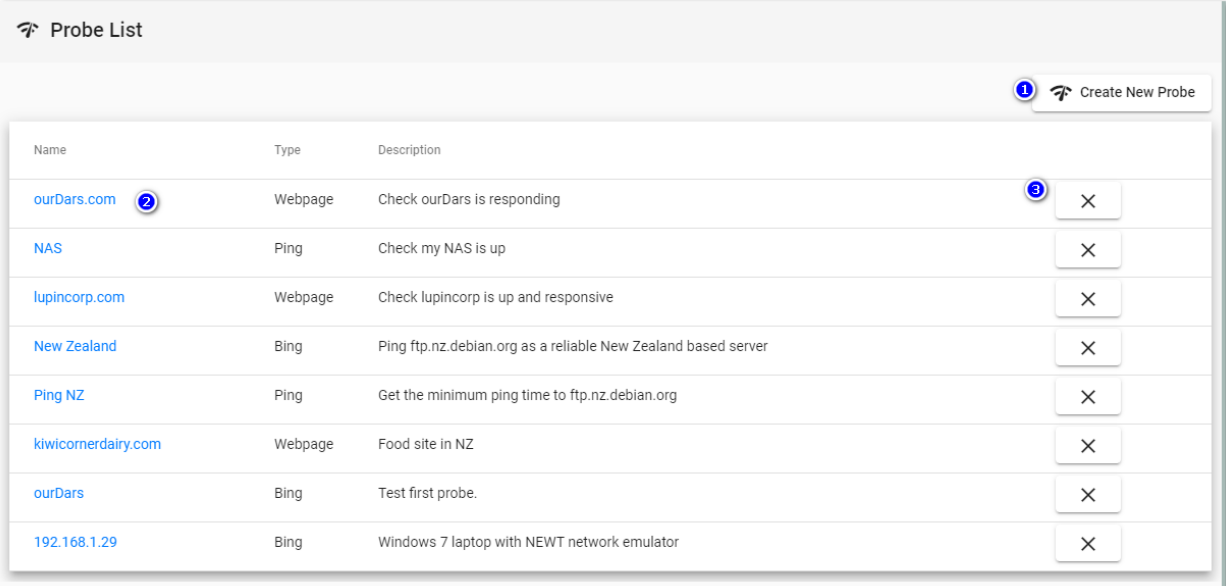
Name	Type	Target	<b>4</b>
ourDars.com	ExYPkVHVW1Es9CT1eSAV	https://ourDars.com	<input checked="" type="checkbox"/>

The button of the device form is shown above and is how it looks in update mode. Note: the following differences from the create mode.

1. The keys panel is collapsed – usually the key information only gets set, once, when the device is first created.
2. The location panel is expanded. The latitude and longitude can be entered manually but it is much easier to use the map view to click on the location for the device and the lat/Ing .will be updated.
3. A download option is provided to generate and download the config.py file that needs to be placed on the device when being provisioned.
4. A list of probes. Selecting the checkbox item next to the probe will assign the probe to be run (or not run) on the device. Note: there is a 2 second delay for each change to a probe assignment for a device. This delay is imposed by the Cloud IOT service to restrict the rate of change to device information.

## J: Probe List

Clicking on the Probes menu item will take you to the “Probe List” page where you can; 1-Create a New Probe, 2-Edit a probe or 3-Delete a probe. A Probe is a definition of a network capability test.



Name	Type	Description	
<a href="#">ourDars.com</a> ②	Webpage	Check ourDars is responding	③ X
<a href="#">NAS</a>	Ping	Check my NAS is up	X
<a href="#">lupincorp.com</a>	Webpage	Check lupincorp is up and responsive	X
<a href="#">New Zealand</a>	Bing	Ping ftp.nz.debian.org as a reliable New Zealand based server	X
<a href="#">Ping NZ</a>	Ping	Get the minimum ping time to ftp.nz.debian.org	X
<a href="#">kiwicornerdairy.com</a>	Webpage	Food site in NZ	X
<a href="#">ourDars</a>	Bing	Test first probe.	X
<a href="#">192.168.1.29</a>	Bing	Windows 7 laptop with NEWT network emulator	X

From the probe list you can:

1. Create a new probe
2. Edit a probe
3. Delete a probe

## L,N Create/Update/Delete Probes

Depending on the action selected on the Probe List form the Probe form will be presented. The form will be in Create/Update or Delete mode. The full form in Create mode is shown below.

Probe information in this form is

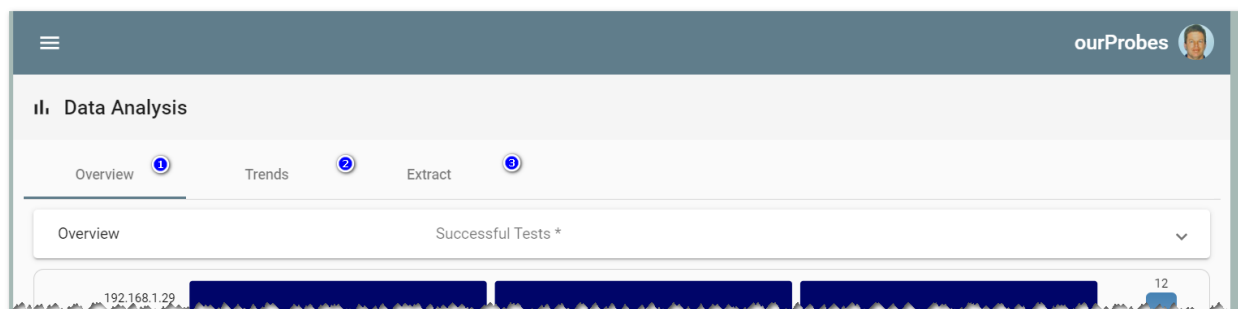
1. Main Menu
2. Return to Probe List
3. Name: Required, this is a short description of a probe and function. The approach I have used is to use the web URL for webpage probes (i.e. "google.com") and domain name for ping probes (i.e. google), for ping probes I have used the domain name with a suffix of ping (i.e. "google ping").
4. Description: Required A full description of the probe, possibly including why it was created and unique aspects of the probe
5. Network target: Required – the resolvable network address for ping and ping probes ( i.e. "192.168.1.1" or "jsmith25.noip.me"), and a valid web URL for a webpage probe (i.e. <https://lupincorp.com>). The web URL must resolve immediately to the required web page and not trigger a redirect.
6. Type : Select the type of probe – these are
  - a. Bing: Performs series of ICMP pings of different lengths against the target address to determine the round-trip latency (rtl) and available bandwidth (bps – bits per second). As well, a "success" measurement is recorded when a successful bing is performed (or a "fail" if not). ("bps","rtl","success"/"fail" measurements)
  - b. Ping: A small icmp echo request is performed against the target address. A "success" measurement is recorded when a successful ping is performed (or a "fail" if not). The round-trip latency (rtl) is recorded as a measurement on a successful ping test. ("rtl","success"/"fail" measurements)
  - c. webpage: A html(s) get is performed against the target address (URL). If successful a time to first byte in milliseconds (ttfb) measurement is recorded. If a match string (7) is defined and the first 1000 bytes of the web page response contains the string then a success measurement is also created, if no match string is defined and a status of 200 is returned then a success measurement will be created. If the http request fails (i.e. not a HTTP 200 status response) then a fail measurement is created with a value of the http status code. ("ttfb","success"/"fail" measurements)

7. Match String: Optional – if entered, then the http response will be tested to see if it contains the match string (Note: only the first 1000 bytes of the response will be scanned). It is recommended that the page (as defined by the target address UTL (5) be opened in a browser first and to do a “View source” to check that the string will be in the first 1000 bytes of the response. If no “Match String” is entered, then as long as the web page is successfully returned it will be marked as a successful test.
8. Action Button (Create/Delete) – is shown when the requested action can be performed. Note: The delete is a “logical” delete, the probe will be removed from any assigned devices when deleted. Alerts will be shown when probe updates are not possible (i.e. Measurements have been created using the probe specification, in some cases a new probe may need to be created rather than changing an existing probe). See example of the message below

Probe's target and type properties can not be updated because measurements have already been created for this probe. Options: Delete and create another probe.

#### Q: Data Analysis

Clicking on the Data Analysis menu item will take you to the “Data Analysis” page where you can select from one of the three data analysis tabs . The following section walks through the three main data analysis functions. This manual will walk through the available options, however it is recommended that users watch the videos available under the help menu to get a better idea of how the data analysis is used. See overall tab layout on the Data Analysis page below:



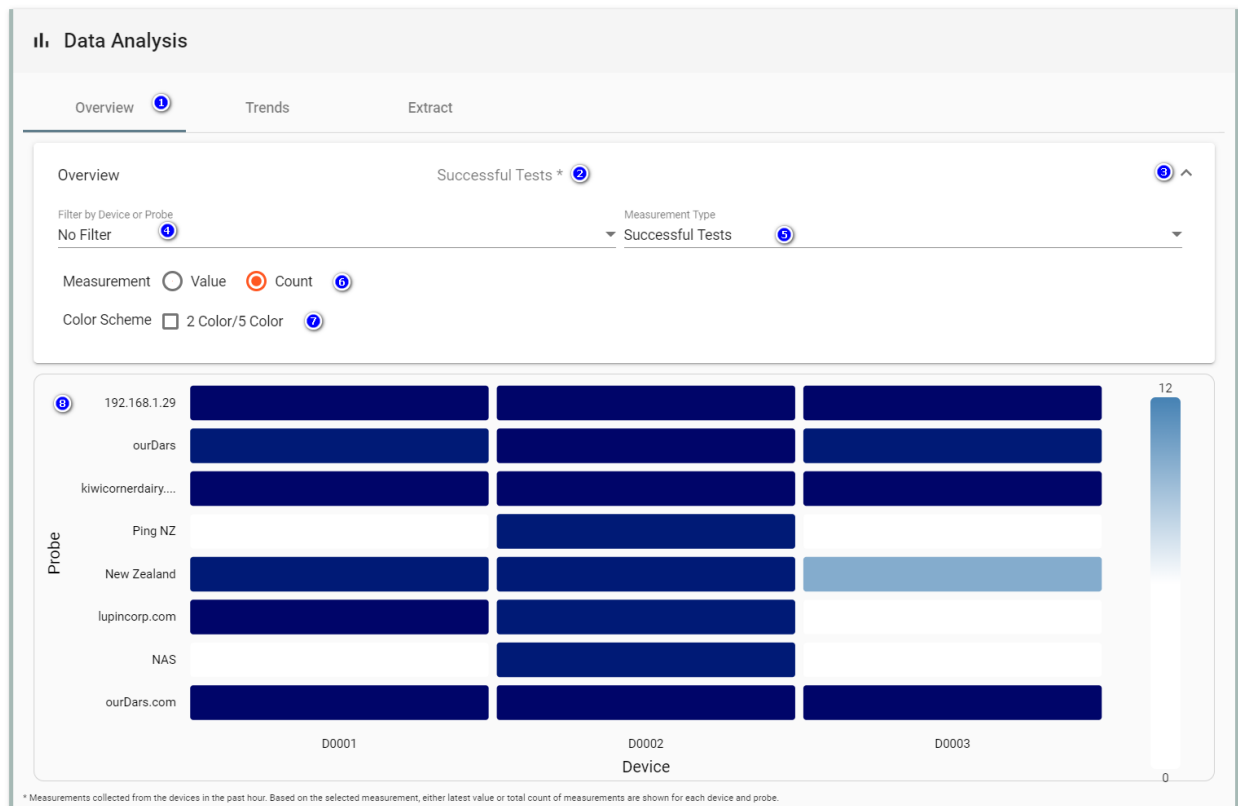
Tabs are

1. Overview – shows heat charts based on the latest information coming from devices and probes.
2. Trends – Show trends based on summarized data coming from the devices and probes.
3. Extract – pull down data to analyze in external tools such as Excel, Power BI etc.

#### S: Overview Tab

The overview tab is a heat map visualization based on the last hour's measurements received from the devices and probes running on those devices. By default it shows the number of successful runs of each

probe on each device to quickly see if there has been problems with a particular device or probe running on a device. See an example of this default view below:



The information shown on the heat map is determined by the parameters selected for the overview. These settings are:

1. The overview tab selector
2. The overview Title (displays the currently selected measurement type)
3. Panel expander: click to open and close the settings panel.
4. Filter: The heat map can be filtered to show a particular device or probe. The probe filter is useful for comparing the network measurements from various devices/locations. (i.e. Available bandwidth is frequently different from different locations)
5. Measurement Type: The measurement to be viewed. This could be "Bits per Second", "Round Trip Latency", "Time to first Byte", "Successful Tests", "Failed Tests"
6. Value/Count selector. If **value** is selected, then the most recent value seen for the measurement is shown. If **count** is selected, then the number of measurements seen in the last hour is shown.
7. Color Scheme: By default, a 2-color scheme is shown, with higher values having a darker color than lower values. The 5-color scheme shows more colors and is useful for spotting differences if there is a wide range of values being shown on the heat map.

Note: Clicking on a heat map Device/Probe cell will take the user to the Trends tab with the appropriate Device/Probe and Measurement Type settings.

## T: Trends Tab

The Trends tab shows summary measurements over time. The summary measurements get calculated at the start of every hour, or UMT day, by ourProbes background functions and are available as hourly and daily summaries.



The Trends tab has the following properties:

1. Trends Tab: On the Data Analysis Page
2. Trends chart Title/Information: Shows the selected device, Selected Probe and Time range being shown.
3. Expand/Close Settings Panel
4. Select Time Range: Choose the time range to be displayed, Time ranges more than 7 days will show daily summary information, otherwise hourly summary information is shown.
5. Select Measurement Type: Select the measurement type to show . This could be “Bits per Second”, “Round Trip Latency”, “Time to first Byte”, “Successful Tests”, “Failed Tests”
6. Select Device: Select the device measurement summaries to be shown
7. Select Probe: Select the probe measurement summaries to show.
8. Available Series: Select what statistics are to be shown on the trend chart. The series starting with “p” are percentiles. Note: “Count” is most useful for “Success”/“Fail” measurement types and the aggregate statistics like p50, mean is most useful for all others with values.
9. Auto Y Axis: Click to set the y axis to the lowest y value, otherwise sets the Y axis to zero.
10. Trend Chart: The resulting chart. Rolling over the chart will show the specific values at the point clicked. Clicking a series legend will highlight the series on the chart.

## U: Data Extract Tab

The Extract tab facilitates getting a data extract (A CSV file) of selected measurement data or measurement summary data. See the extract tab below:

The screenshot shows the 'Data Analysis' interface with the 'Extract' tab selected. The interface includes several input fields and a button, each marked with a numbered callout:

- 1:** The 'Extract' tab selector at the top.
- 2:** The 'Choose a starting date (Local Time Zone)' field, currently showing '3/14/2020'.
- 3:** The 'Select Time Range' dropdown menu, currently set to '4 hours'.
- 4:** The 'Extract Type' dropdown menu, currently set to 'Hourly Summary'.
- 5:** The 'Maximum Rows' dropdown menu, currently set to '10'.
- 6:** The 'Measurement Type' dropdown menu, currently set to 'All'.
- 7:** The 'Extract Data' button.
- 8:** The 'Download Extracted Data (Rows:10)' link.

Properties of this tab are:

1. Extract Tab Selector
2. Starting Date: Select when to start the data extract from. This represents the start based on the start of the day (12:00 AM of the selected day) using the users local time zone (So may start at different time depending on the location/time zone of the user). The extracted data provides both local and UMT time zone timestamps.
3. Select Time Range: How many hours or days of data to extract starting from the start date.
4. Extract Type: Three collections of measurement data exist that can be extracted. These are
  - a. Hourly Summary: Statistics that are calculated based on measurements seen over the previous hour. (Relative to the UMT timestamp on each extracted data row).
  - b. Daily Summary: Statistics that are calculated based on measurements seen over the previous day (Relative to the UMT timestamp on each extracted data row).
  - c. Raw Measurements: The measurement data produced by the probes running on the devices. No aggregation or summarization has been performed on this data (So there may be a lot more of it)
5. Measurement Rows: A limiter to restrict the number of rows extracted. It is useful for
  - a. While determining the data to be extracted, to quickly download a small sample of extracted data (i.e. 10 rows) to check it is really what is required.
  - b. Google cloud charges based on number of rows read. The upper limit of 5000 rows will reduce the pain of chargebacks.
6. Measurement Type: Select "All" or refine by type – i.e. "Bits per Second", "Round Trip Latency", "Time to first Byte", "Successful Tests", "Failed Tests" or "startup". Note: Startup measurement type is a special measurement not shown in the other Data Analysis functions, it is produced by the devices on each startup and is useful for monitoring the health pf the devices (i.e. how often they restart, IP assigned and micropython memory footprint)
7. Extract Data: Triggers the extract of the data – when complete the download link (8) is displayed.



- Download Link: Click to download the extracted data to your local machine. The download file is given a unique name of ourProbes-<ISO date-time>.csv e.g. "ourProbes-2020-03-08T23\_24\_03.021Z.csv"

## P: My Profile

Clicking on the "My Profile" menu item will take you to the "My Profile" page where you can view your user profile and update your default location. Access right changes (i.e. Active, Administrator) must be requested from a user with administrator rights. The My Profiles page is shown below

**User Profile**

**Display Name**

David Somerville

**User Id**

Vgh4kbbLJUa9dh9QebKky8ji0U2**eMail****Photo URL****Is Administrator?****Is Activated?****Photo****Location**

Edit Location Details

**Latitude**

40.17482783993688

**Longitude**

-75.30209777760138

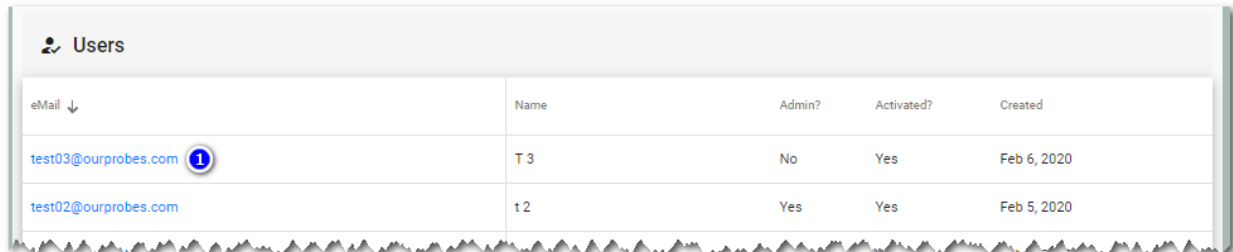
A user can only update their default location on there own profile.

- Expand/Close the location panel
- Click on the map to update your default location.



## V: User Management

Administrators have access to the user management function. Clicking on the “User Management” menu item will take you to the “Users” page where you can view and manage users who are registered in the system.

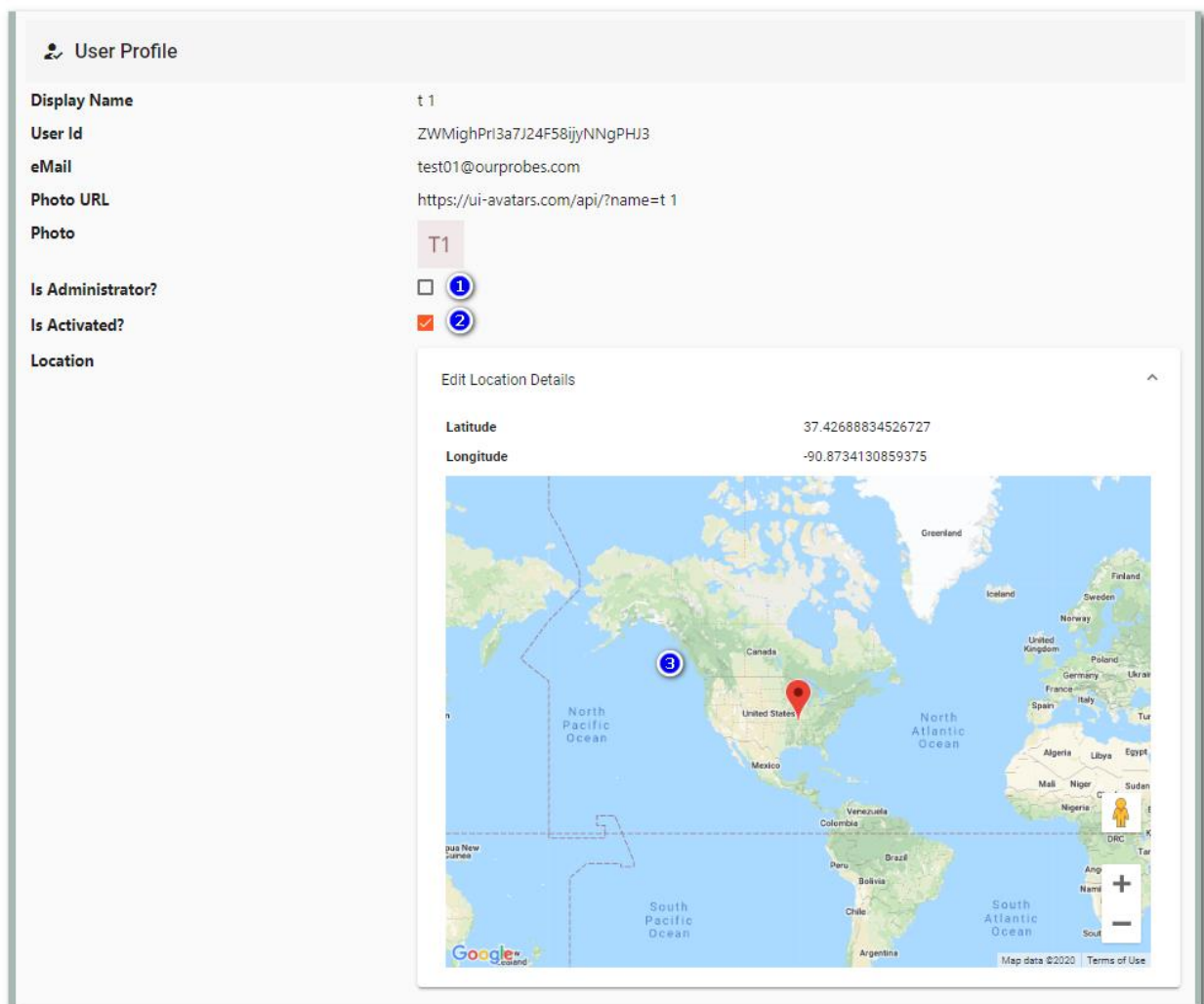


The screenshot shows a web interface titled "Users" with a user icon. Below the title is a table with the following columns: eMail (with a dropdown arrow), Name, Admin?, Activated?, and Created. There are two rows of user data. The first row has a blue circle with the number "1" next to the email address. The second row has a blue circle with the number "2" next to the email address.

eMail ↓	Name	Admin?	Activated?	Created
<a href="#">test03@ourprobes.com</a> 1	T 3	No	Yes	Feb 6, 2020
<a href="#">test02@ourprobes.com</a> 2	t 2	Yes	Yes	Feb 5, 2020

1. Click on the user’s email address to be taken to the profile page for that user.

On the user page an administrator can update the Admin, Active flags and location for the user. See the User Profile page in edit mode below.



The screenshot shows a web interface titled "User Profile" with a user icon. The page displays user information and edit options. The "Is Administrator?" checkbox is unchecked, and the "Is Activated?" checkbox is checked. Below these is a "Location" section with a map. The map shows the United States with a red pin. A blue circle with the number "3" is next to the map. The "Edit Location Details" section shows the Latitude and Longitude coordinates.

**User Profile**

Display Name: t 1

User Id: ZWMighPri3a7J24F58jyNNGPHJ3

eMail: test01@ourprobes.com

Photo URL: https://ui-avatars.com/api/?name=t 1

Photo: T1

Is Administrator? ☐ 1

Is Activated? ☒ 2

Location

Edit Location Details

Latitude: 37.42688834526727

Longitude: -90.8734130859375

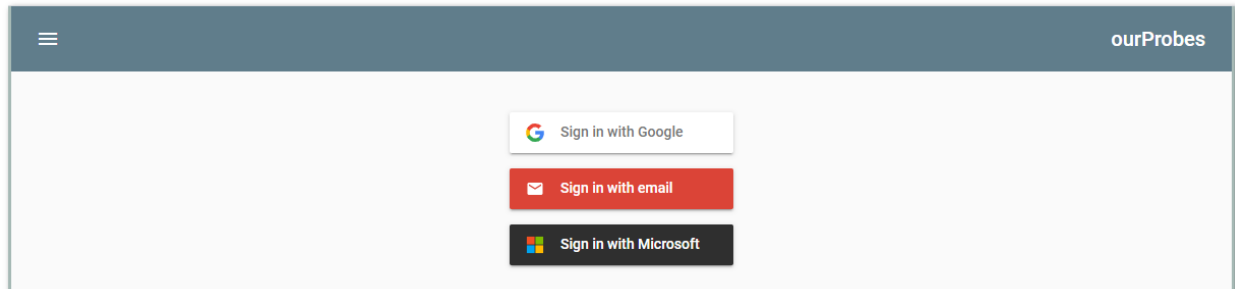
Map data ©2020 Terms of Use

The administrator can change the following properties for other users.

1. Is Administrator?: This set a user to have administrator rights (or not)
2. Is Active?: When users first sign into the system they get registered but have a status of “Inactive”. Inactive users can not use the functionality of the system. After the first login the user contacts an administrator to make them active. The administrator does this by checking this box.
3. Default Location Map. By default, new users are given a location of Lat=0 and Lgn=0. The user can update this themselves or an administrator can do this by clicking on the location map.

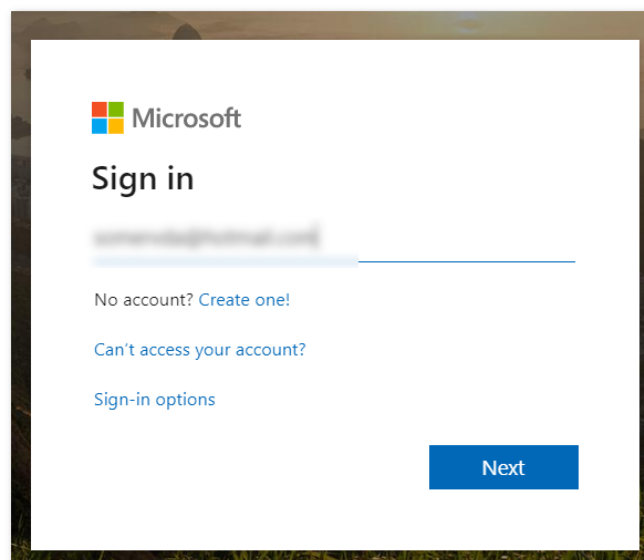
## Signing In

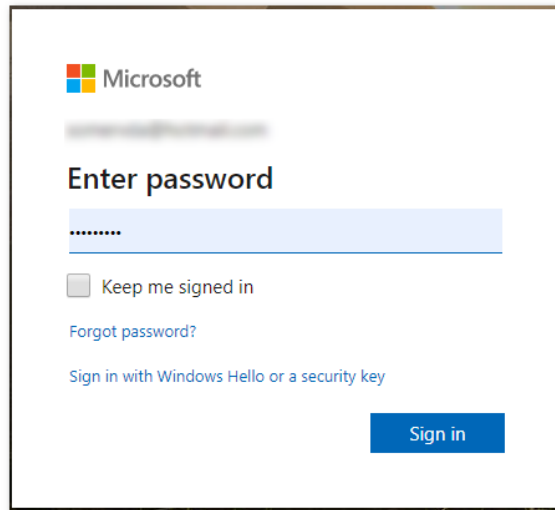
All users must sign in to use ourProbes functionality, and the sign-in page is the first page presented to users.



OurProbes includes, out of the box, implementations of three login providers enabled by Firebase authentication services (Google, Email and Microsoft). The log-in form is the starting point, then the appropriate provider is selected. In enterprises it may be more typical for the unused providers to be removed so only the supported sign in processes is available (Usually only one is used inside an enterprise). See more about available authentication providers at <https://firebase.google.com/docs/auth>.

Typically, a user will choose the log-in method, they will be taken to the authentication providers authentication service (ourProbes does not see users' passwords etc.) and enter their user code and password. See example below for a Microsoft sign in.





Microsoft

Enter password

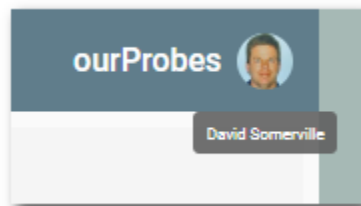
.....

☐ Keep me signed in

[Forgot password?](#)

[Sign in with Windows Hello or a security key](#)

Sign in



The first time a user logs into ourProbes they need to be activated by a system administrator (See user Management), to have access to the ourProbes system.

## Technical

### Data (Firestore Document Model)

Device
id: string; description: string; location?: string; communication: boolean; type: DeviceType; longitude: number; latitude: number; dateCreated?: Date; governorSeconds: number; runProbes: boolean; publicKey: string; privateKey: string; privateKeyTuple: string; probeList: ProbeListItem[];

measurementSummaries
deviceId: string; probeld: string; name: string; type: string; umt: firebase.firestore.Timestamp; period: measurementSummaryPeriod; mean: number; p00: number; p25: number; p50: number; p75: number; p100: number; stdDev: number; count: number;

Probes
id?: string; name: string; description: string; type: ProbeType; target: string; dateCreated?: Date; status: ProbeStatus; match?: string;

users
uid: string; email: string; photoURL?: string; displayName?: string; dateCreated?: Date; isAdmin?: Boolean; isActive?: Boolean; longitude?: number; latitude?: number;

measurements
deviceId: string; probeld: string; name: string; type: string; UMT: firebase.firestore.Timestamp; value: number;

middlewareEventItem *
title: string, logTime: firebase.firestore.Timestamp, id: string, details?: string

\* middlewareEventItem only produced if a middleware operation fails. Should not have any

#### Types and Enums

```
enum DeviceType {  
  esp32HiLetGo = 1,  
  esp32GatewayOlimex = 2,  
  raspberryPi4 = 3  
}
```

```
interface ProbeListItem {  
  id: string;  
  name: string;  
  target: string;  
  type: ProbeType; match?: string;  
}
```

```
enum measurementSummaryPeriod {  
  hour = 1,  
  day = 2  
}
```

```
enum ProbeType {  
  bing = 1,  
  ping = 2,  
  webPage = 3  
}
```

```
export enum ProbeStatus {  
  active = 1,  
  deleted = 9  
}
```

measurement types: bps, rtl, ttfb,  
success, fail, startup

## Security

Firestore security rules can be found in the projects Firestore.rules file. Below is the high level (Document level) rules in that file that controls access to the Firestore instance and documents.

```
// ***** Reusable functions *****
function isAuthenticated() {
    return request.auth.uid !=null;
}

function isAdmin() {
    return isAuthenticated() &&
        get(
/databases/$(database)/documents/users/$(request.auth.uid)).data.isAdmin == true;
}

function isActivated() {
    return isAuthenticated() &&
        get(
/databases/$(database)/documents/users/$(request.auth.uid)).data.isActivated == true;
}

function updatingField(field) {
    // Check to see if named field is being updated
    return (field in request.resource.data) && resource.data[field] !=
request.resource.data[field];
}

// ***** Document Access functionsnp

// user document rules (Delete not allowed)

match /users/{user} {
    allow read : if isAdmin()
        ||
        (isAuthenticated() &&
            user==request.auth.uid) ;
    allow create: if isAuthenticated() &&
        user==request.auth.uid ;
    allow update: if (isAdmin() &&
        user!=request.auth.uid )
        ||
        (isAuthenticated() &&
            user==request.auth.uid &&
            !updatingField("isAdmin") &&
            !updatingField("isActivated")
        );
}

match /{path=**}/devices/{did} {
    allow read: if isActivated();
    allow write: if isActivated();
}

match /{path=**}/probes/{did} {
    allow read: if isActivated();
    allow write: if isActivated();
}
```

```
// metrics are read only by anyone

match /{path=**}/measurements/{did} {
  allow read:  if (true);
}

match /{path=**}/measurementSummaries/{did} {
  allow read:  if (true);
}

// Let anyone read the youtubevideos documents (Used to populate the help
component)
match /YouTubeVideos/{youtubevideo} {
  allow read:  if true ;
}
}
```

## Data Extract Examples

There are 2 csv formats that are dependent on if measurements or measurement summaries are extracted, examples are shown below

### Measurements Summary Extract

id	count	deviceId	mean	name	p00	p100	p25	p50	p75	period	probeld	stdDev	type	umt	localeDate	umtDate	umtHour
0EpDS5Qn	12	D0002	3.702667	NAS	3.569	3.995	3.588	3.617	3.669	1	ISmgbvFH	0.154787	rtl	Timestamp	#####	#####	4
1lqO06uol	12	D0001	2975.529	kiwicorne	2948.451	3007.008	2961.078	2974.024	2981.57	1	Yiuo8Gouj	18.13138	ttfb	Timestamp	#####	#####	4
1qgMYlhD	12	D0001	0	192.168.1.	0	0	0	0	0	1	zHc8Rqlk4	0	success	Timestamp	#####	#####	4
2FXpAUJ3	12	D0002	2991.253	kiwicorne	2904.664	3323.161	2925.581	2973.329	2987.335	1	Yiuo8Gouj	105.0496	ttfb	Timestamp	#####	#####	4
2To9PXx1	12	D0001	1211.441	lupincorp.	1106.035	1363.197	1147.791	1159.671	1282.447	1	lUKgohd2	80.28848	ttfb	Timestamp	#####	#####	4
3TZGLWBt	12	D0002	42032579	New Zealand	12510310	88926320	24804102	37306840	55419136	1	NWYYzm1	21457310	bps	Timestamp	#####	#####	4
4WkAxT0v	3	D0003	1096778	New Zealand	924230	1407348	924230	958756	958756	1	NWYYzm1	220058	bps	Timestamp	#####	#####	4
51gjlC2Te	12	D0002	233.0621	Ping NZ	232.21	233.807	232.286	233.12	233.528	1	VVbNWoi	0.556626	rtl	Timestamp	#####	#####	4
7thVxGa5	9	D0003	-17.1111	New Zealand	-18	-17	-17	-17	-17	1	NWYYzm1	0.31427	fail	Timestamp	#####	#####	4
862Vsvb6	12	D0002	496063.1	192.168.1.	493281	498446	494600	495764	497452	1	zHc8Rqlk4	1691.156	bps	Timestamp	#####	#####	4

### Measurement Extract

id	UMT	deviceId	name	probeld	type	value	localeDate	umtDate	umtHour	umtMinut
N6RS7UKF	Timestamp	D0003	ourDars	mF4wmQ	bps	24885034	3/14/2020 0:00	3/14/2020 4:00	4	0
tB9g9MbA	Timestamp	D0003	ourDars	mF4wmQ	rtl	7.612	3/14/2020 0:00	3/14/2020 4:00	4	0
RVMpedp	Timestamp	D0003	ourDars	mF4wmQ	success	0	3/14/2020 0:00	3/14/2020 4:00	4	0
pqlL5zdkSr	Timestamp	D0003	New Zealand	NWYYzm1	fail	-17	3/14/2020 0:00	3/14/2020 4:00	4	0
6eTyjdUTF	Timestamp	D0003	kiwicorne	Yiuo8Gouj	success	0	3/14/2020 0:00	3/14/2020 4:00	4	0
jwPDXSJM	Timestamp	D0003	kiwicorne	Yiuo8Gouj	ttfb	4237.273	3/14/2020 0:00	3/14/2020 4:00	4	0
dhfDnSX2	Timestamp	D0003	Web OurC	ExYPkVHV	ttfb	1157.416	3/14/2020 0:00	3/14/2020 4:00	4	0
M4AFQiM	Timestamp	D0003	Web OurC	ExYPkVHV	success	0	3/14/2020 0:00	3/14/2020 4:00	4	0
Jse9sEQDz	Timestamp	D0003	192.168.1.	zHc8Rqlk4	success	0	3/14/2020 0:00	3/14/2020 4:00	4	0
S54LxH4ds	Timestamp	D0003	192.168.1.	zHc8Rqlk4	bps	68486	3/14/2020 0:00	3/14/2020 4:00	4	0



## Provisioning a New Device

Normally esp32 devices come with no preinstalled software. To provision a new device tree main steps are needed:

1. Create a RSA public Key, Private Key, and Private key for the device.
2. Create the device definition in the ourProbes application (RSA keys are needed for this)
3. Connect the Device by serial connection, download and run the provisioning job.

### *Creating RSA keys*

This is based on blog at <https://medium.com/google-cloud/connecting-micropython-devices-to-google-cloud-iot-core-3680e632681e> also see development blog video at [https://youtu.be/7hMDnVk\\_rqA](https://youtu.be/7hMDnVk_rqA) (minute 13:57) for a walkthoug of the RSA creation process. Use the following openssl commands

```
openssl genrsa -out rsa_private.pem 2048  
openssl rsa -in rsa_private.pem -pubout -out rsa_public.pem
```

Then convert the rsa\_private.pem key to a tuple using the python utility see <https://medium.com/google-cloud/connecting-micropython-devices-to-google-cloud-iot-core-3680e632681e>

```
python utils/decode_rsa.py
```

### *Create the Device Definition in ourProbes*

See section “E/G: Create/Update Device information” of this manual. Download and save the config.py file available with this new definition.

### *Download and run the Provisioning files and Job*

See video on the [ourProbes help](#) page ( [https://youtu.be/6il\\_UG8eSqA](https://youtu.be/6il_UG8eSqA) ).

1. Download and open the [deviceSetUp.zip](#) zip file .
2. From the zip file, copy the ourProbesDeviceMaker folder to a temporary location on your machine.
3. In a command window “cd” to the ourProbesDeviceMaker folder.
4. In the probe subfolder update the config.py file (Usually by using the config.py creator in the device page, update any ssid/password info if needed.).
5. Connect the new esp32 to your pc device via a serial connection (usb cable).
6. Update the deviceDeploy.bat file to use the correct serial port. Note: you can check what serial port the device is on using the "mode" command in windows.

7. Start the deviceDeploy.bat job.