

Open Data: Receive it Yourself

0xA, Marenz, Tassilo (dump@dvb.solutions)

Datenspuren 2022, Dresden



Radio & VDV 420

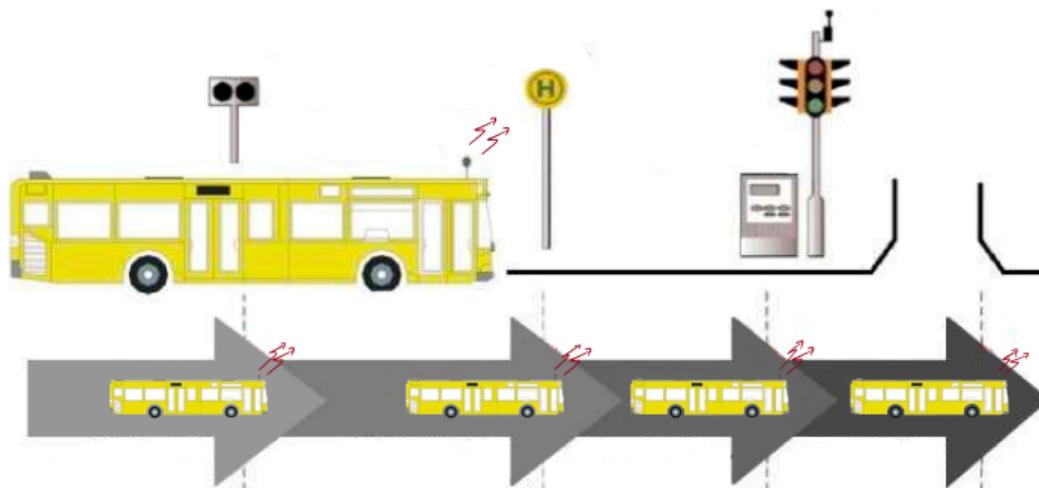


Figure 1: Traffic light controlled by radio link of busses and trams. Modified graphic from urbic



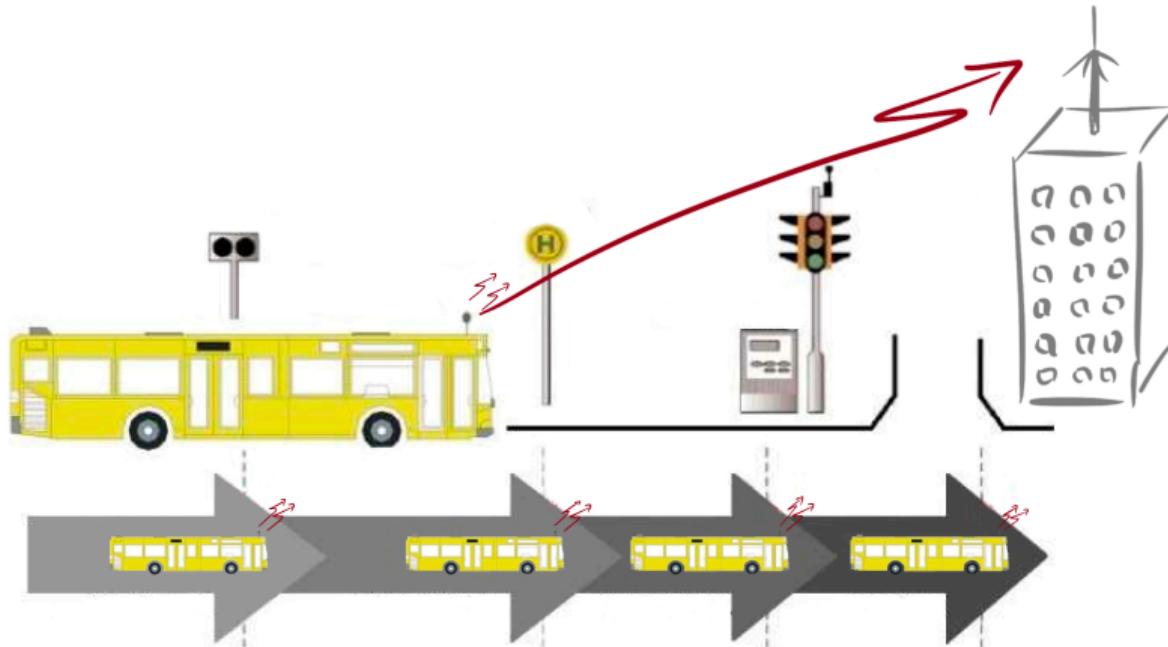


Figure 2: Radio signals can be received by our antennas

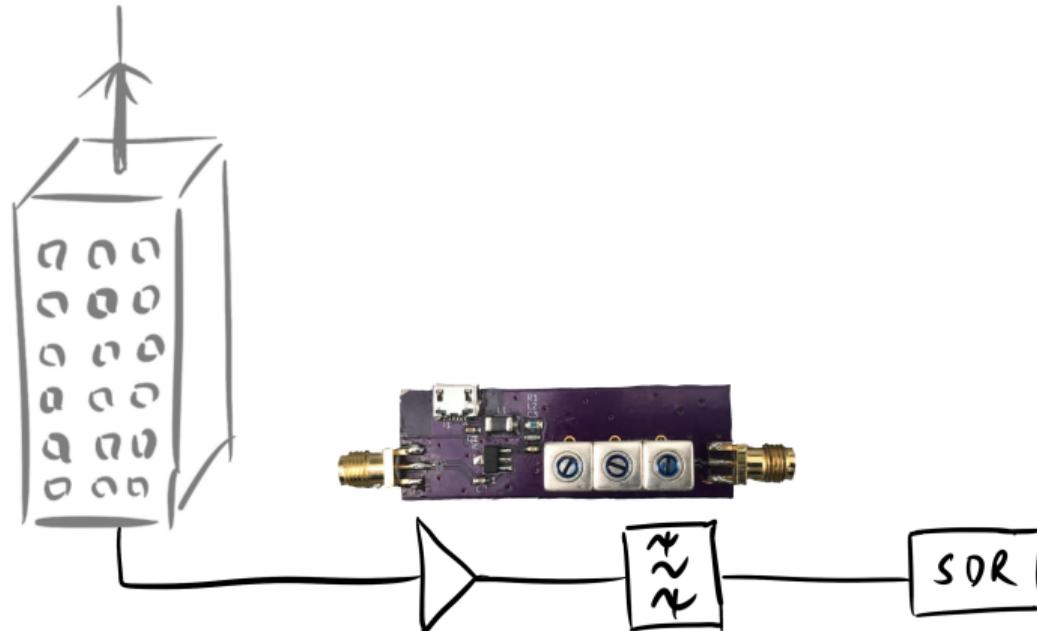


Figure 3: Schematic overview of the receiver hardware



- ▶ R09.1x telegrams of trams and busses standardized in VDV 420 ↗
- ▶ R09.18 is specified not in the standard, but there is a wireshark dissector ↗ for it

Bez.	Bedeutung	Infobyte 1	Infobyte 2	Infobyte 3	Zusatzbyte 1 - 6
R 09.10	Reduzierte Meldung	1001 0001	ZV ZW ZW ZW 1 2 3	0000 MP MP MP MP 1 2 3 4	MP MP MP MP 5 6 7 8
R 09.11	Standard Meldung	Info-byte 1	ZV ZW ZW ZW 1 2 3	0001 Info-byte 3 MP MP MP MP 9 10 11 12	Zusatzbyte 1 MP MP MP MP 13 14 15 16
R.09.12	Standard - Meldung mit Priorität	Info-byte 1	ZV ZW ZW ZW 1 2 3	0010 Info-byte 3 Zusatzbyte 1 PR PR HA HA 1 2 1 2	Zusatzbyte 2 R R R R
R.09.13	Standard - Meldung mit Liniennummer	Info-byte 1	ZV ZW ZW ZW 1 2 3	0011 Info-byte 3 Zusatzbyte 1 PR PR HA HA 1 2 1 2	Zusatzbyte 3 LN LN LN LN 11 12 13 14
R.09.14	Standard - Meldung mit Linie/Kurs Nr.	Info-byte 1	ZV ZW ZW ZW 1 2 3	0100 Info-byte 3 Zusatzbyte 1 Zusatzbyte 2 KN KN KN KN 11 12 13 14	Zusatzbyte 4 KN KN KN KN 21 22 23 24
R.09.15	Nicht verwendet			0101	Zusatzbyte 5
R.09.16	Maximal Meldung	Info-byte 1	ZV ZW ZW ZW 1 2 3	0110 Info-byte 3 Zusatzbyte 1 Zusatzbyte 2 Zusatzbyte 3 ZN ZN ZN ZN 11 12 13 14	Zusatzbyte 6 R ZL ZL ZL 31 32 33 34

Figure 4: VDV 420 ↗ specifies R09.1x types containing different amount of data



What data do we receive?

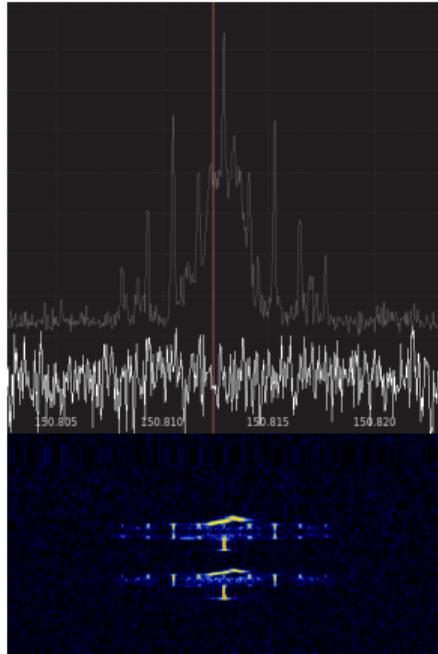
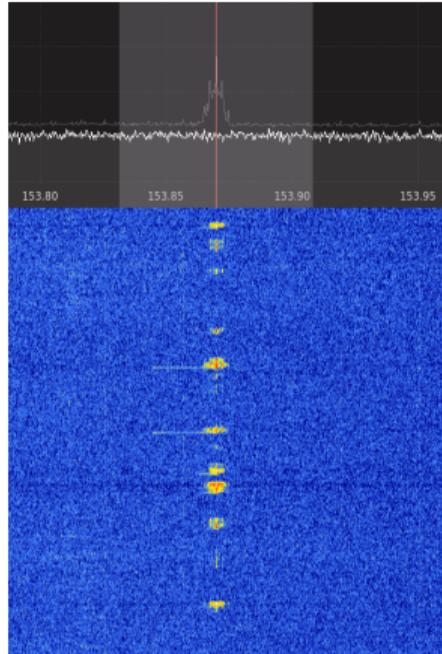
- ▶ Tram identification: line number (3 decimal digits), run number (2 decimal digits), destination number (3 digits)
- ▶ Location: Reporting point (16 bit) which might contain 2 bit registration_type (Pre-registration, Registration, De-registration, Doors closed), traffic light id and direction
- ▶ Other data: Delay (± 7), train length (until now always zero) etc.



- ▶ Representation of data might be different from the original standard
- ▶ Dresden: Reporting Point = ($<TRAFFIC\ LIGHT\ ID> * 10 + <DIRECTION>$) $\ll 2$ | $<REGISTRATION>$
- ▶ Tirol: Reporting Point = ($<DIRECTION> * 1000 + <TRAFFIC\ LIGHT\ ID>$) $\ll 2$ | $<REGISTRATION>$ ↗

<16>	10010001	Z V	Z W	0110	MP	M A	P R	H A	LN	KI	KO	FW	0	ZL
<16-4>	10010001	Z V	Z W	0110	MP	M A	P R	H A	LN	FW		ZN	0	ZL
<16-6>	10010001	Z V	Z W	0110	MP	M A	P R	H A	LN	KN		ZN	0	ZL
<16-5>	10010001	Z V	Z W	0110	MP	P R	H A		LN	KN		ZN	0	ZL
VDV R09.16	10010001	Z V	Z W	0110	MP	P R	H A		LN	KN		ZN	0	ZL

Figure 5: VDV 426 ↗ specifies different formats for the data fields



- ▶ Bursty transmission
- ▶ Minimum Shift Keying with 2400 Baud

Figure 6: Screenshots of the bursty transmission pattern



Figure 7: Telegram receiver on the right

What are the frequency ranges to look for the signal?

- ▶ Technical documentation of different receivers, i.e. RBL-380 ↗ or WZ LSA 2-3/G ↗ , provide these details
- ▶ Frequency 70cm Band 450 MHz – 470 MHz
- ▶ Frequency 2m Band 146 MHz – 174 MHz
- ▶ Frequency 4m Band 68 MHz – 87.5 MHz
- ▶ We have a table of known frequencies ↗
- ▶ OSINT: Search for “R09 frequency <city>” or “Ampelbeeinflussung <city>”

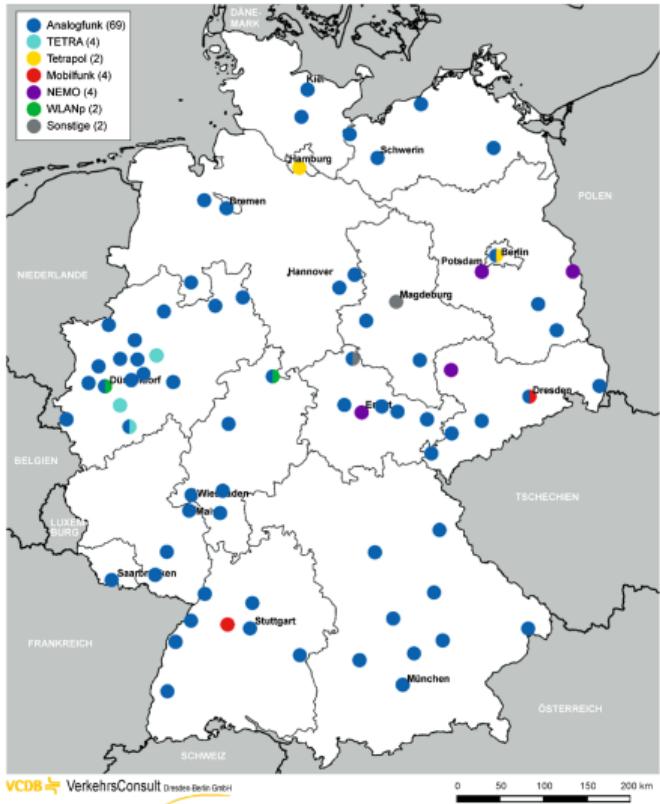


Figure 8: Map from bast ↗ displaying selected cities with traffic lights controlled by public transport. We implemented the standard blue points use.

- ▶ Different physical layer implementations exist too, i.e. Berlin with Tetrapol
- ▶ VDV 426 ↗ has more information on this topic
- ▶ Encoded data doesn't seem to be different



Mapping



- ▶ R09Telegram doesn't provide map position data
- ▶ location is identified by arbitrary integer reporting_point

```
{  
    ...  
    "reporting_point":8366,  
    "junction":209,  
    "direction":1,  
    "request_status":2,  
    ...  
}
```



- ▶ R09Telegram doesn't provide map position data
- ▶ location is identified by arbitrary integer reporting_point
- ▶ To the Google we go!

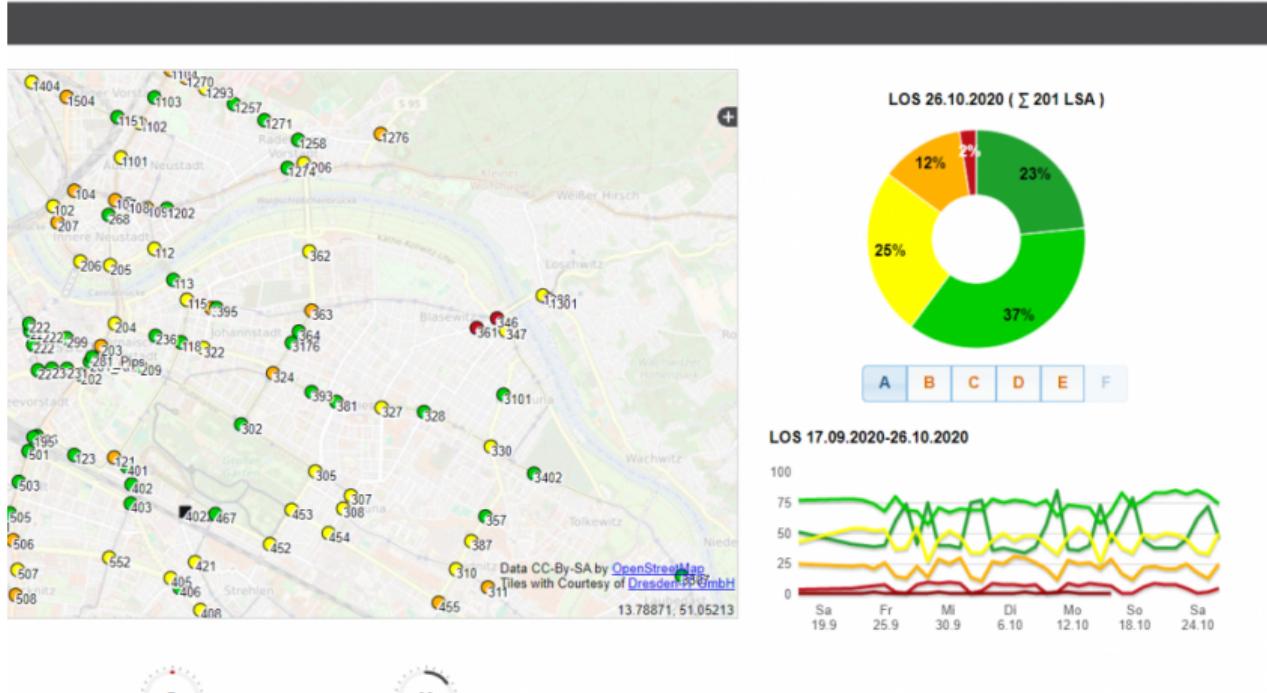
```
{  
    ...  
    "reporting_point":8366,  
    "junction":209,  
    "direction":1,  
    "request_status":2,  
    ...  
}
```



Dump-DVB Institute

OSINT Data

Or Google Your Own LSA ID's



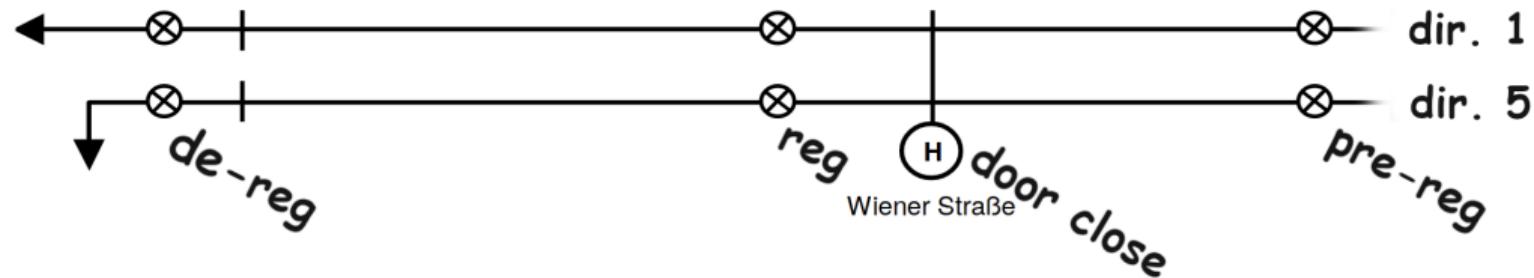


A map of Dresden, Germany, showing numerous green and yellow circular markers scattered across the city area, each containing a unique ID number. The map includes labels for various neighborhoods like Neustadt, Johannstadt, and Albertplatz.

```
"102": {  
    "name": "LSA Anton-/Leipziger Stra\u00e4se",  
    "lat": 51.063033,  
    "lon": 13.736958  
},  
"103": {  
    "name": "Anton-/Zur Eisenbahnstra\u00e4se",  
    "lat": 51.064067,  
    "lon": 13.736935  
},  
"104": {  
    "name": "Schlesischer Platz",  
    "lat": 51.064783,  
    "lon": 13.740362  
},  
"107": {  
    "name": "Albertplatz",  
    "lat": 51.063252,  
    "lon": 13.746367  
},  
"108": {  
    "name": "Albertplatz West",  
    "lat": 51.064241,  
    "lon": 13.744951  
},  
"109": {  
    "name": "Bautzner/Rothenburger",  
    "lat": 51.062925,  
    "lon": 13.751871  
},  
"119": {  
    "name": "Dresden",  
    "lat": 51.052133,  
    "lon": 13.78871  
}
```



- ▶ “by hand” doesn’t scale too well
- ▶ osint is unreliable source of information
- ▶ junction number corresponds to several reporting points
- ▶ need a way to correlate telegrams to map location!





- ▶ Vehicle ID: Line Number & Run Number
- ▶ Dresden Trams: Run number present almost always
- ▶ Dresden Busses: Hit-or-miss, can be seen on the driver monitor
- ▶ Chemnitz:
REEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE



Telegram Recording:

- ▶ SDR with a computer in a tupperware
- ▶ Running off a powerbank
- ▶ Wartrammer-40k: filters telegrams

Location Recording: Phone

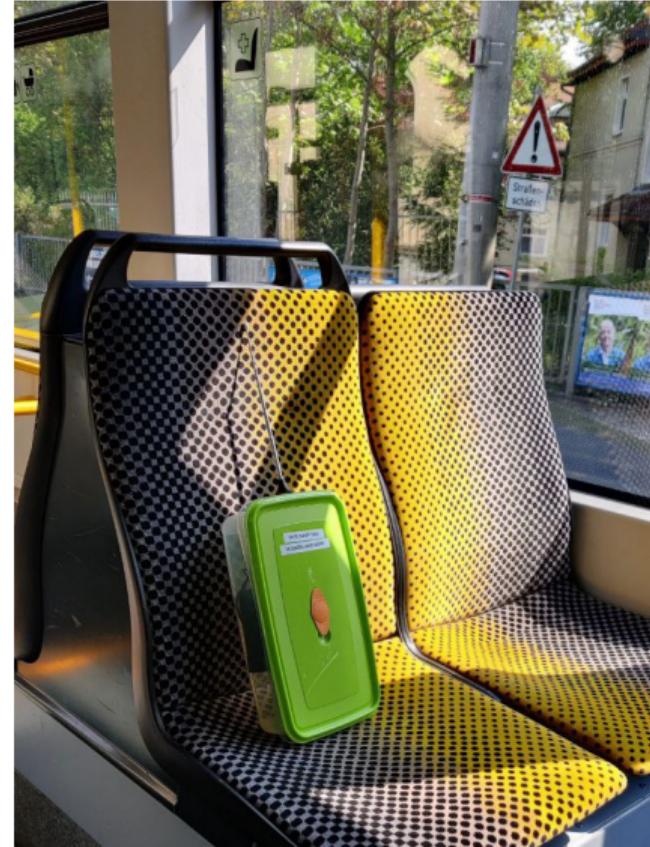
Correlation: lofi





Bootstrapping:

- ▶ Go around the city with a warferry
- ▶ Track your position and line/run number
- ▶ Correlate the data



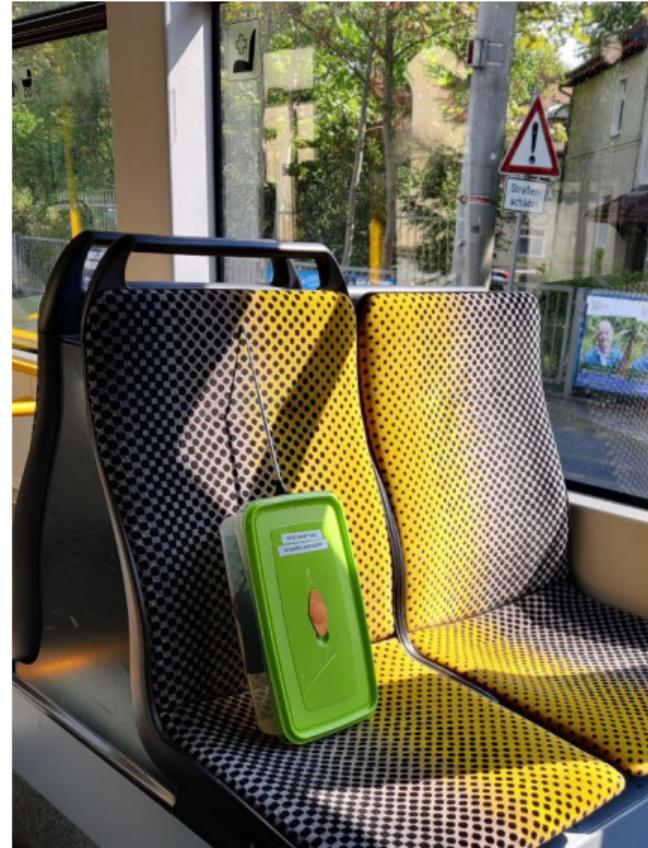


Bootstrapping:

- ▶ Go around the city with a warferry
- ▶ Track your position and line/run number
- ▶ Correlate the data

Decent city coverage by radio stations:

- ▶ Go around the city
- ▶ Track your position and line/run number
- ▶ Correlate the positions to telegrams from the station





- ▶ In some cities there is no easy way to get run numbers ⇒ inference from reporting point
- ▶ “Go around the city” part sucks
- ▶ Nice gps track submission web thing (together with line number tracking)



Architecture & Infra



Dump-DVB Institute

Receivers in Operation

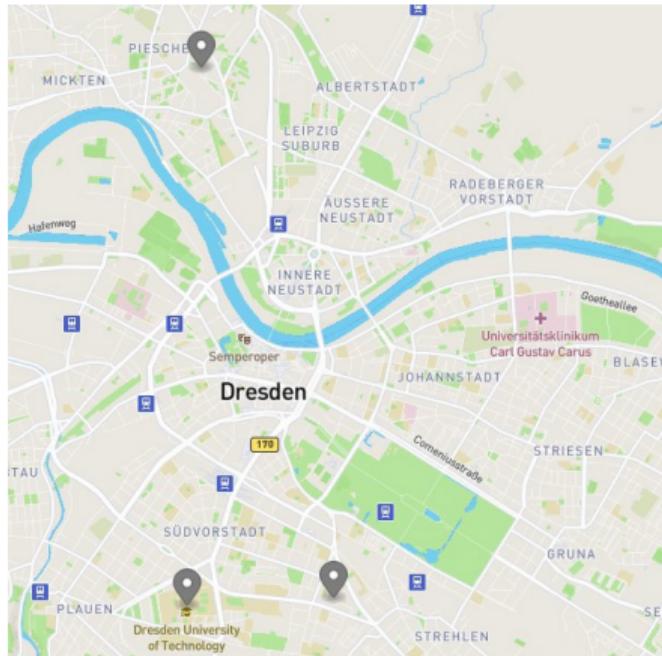


Figure 9: Receivers in Dresden

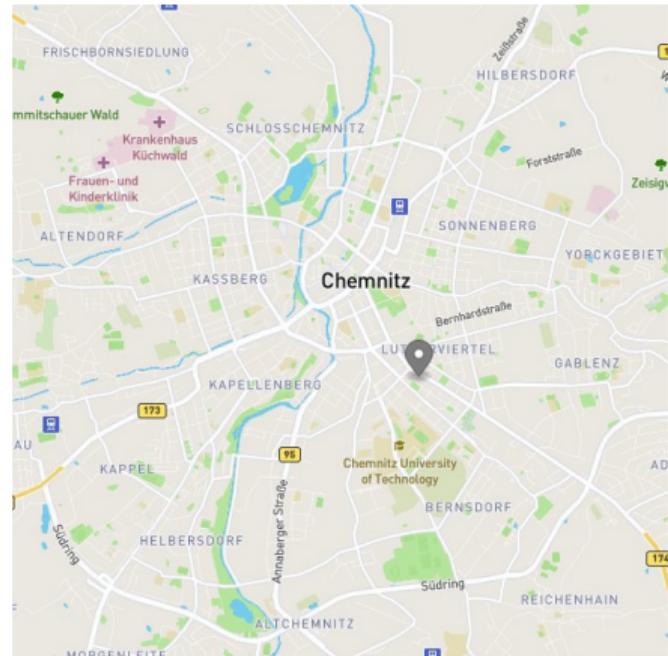
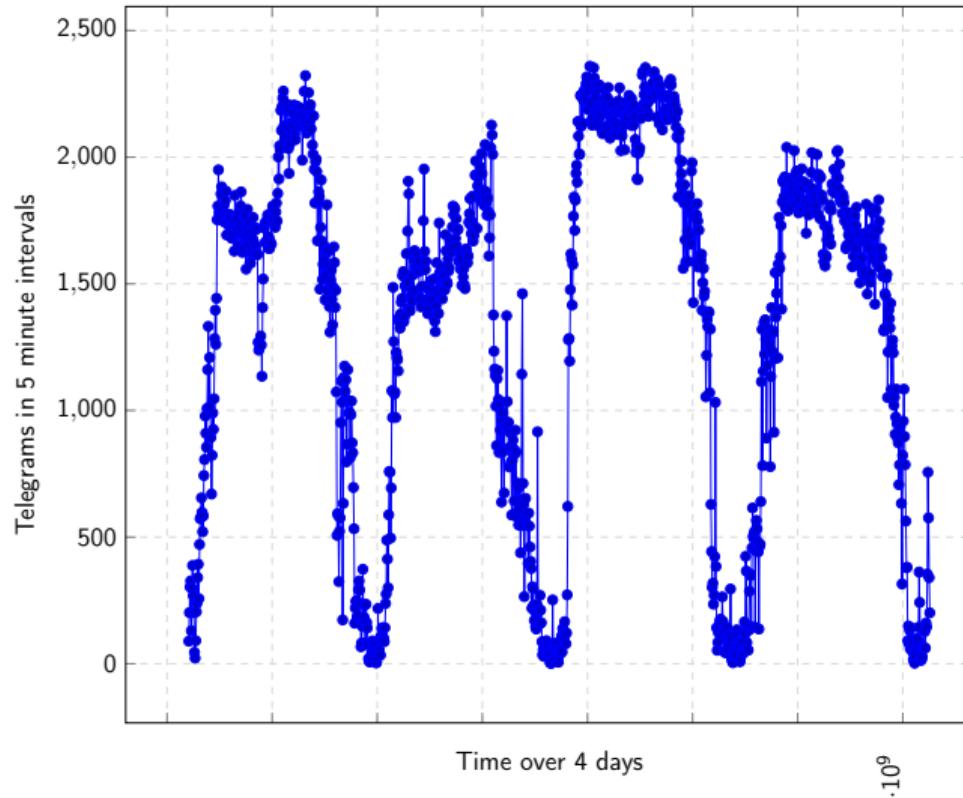


Figure 10: Receivers in Chemnitz



Received Data



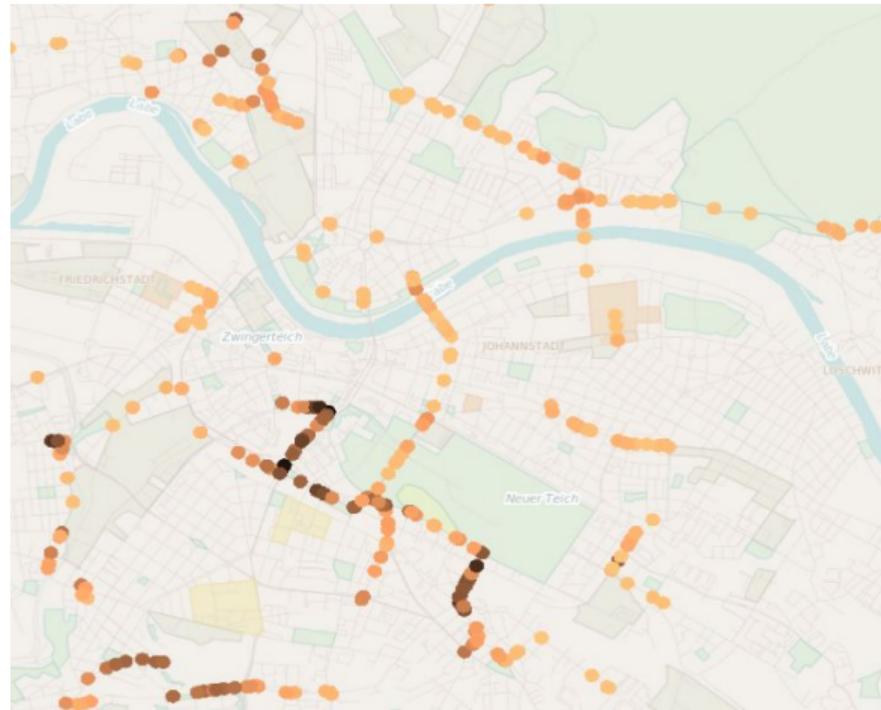


Figure 11: visible reporting points in the city



LSA-Statistik					
LSA-Typen	Anzahl	mit Infrarot-Datenfunk (IDF)		Verlustzeit	
		absolut	relativ	in s	QSV
Knoten-LSA	243	242	100%	20	C
Fußgänger-LSA	151	145	97%	5	A
Bahnübergänge (BOStrab)	33	33	100%	0	A
Bahnübergänge (SOSTrab)	9	9	100%	0	A
Haltlicht-LSA	14	14	100%	0	A
Summe	450	443	Ø 98%	Ø 13	Ø B

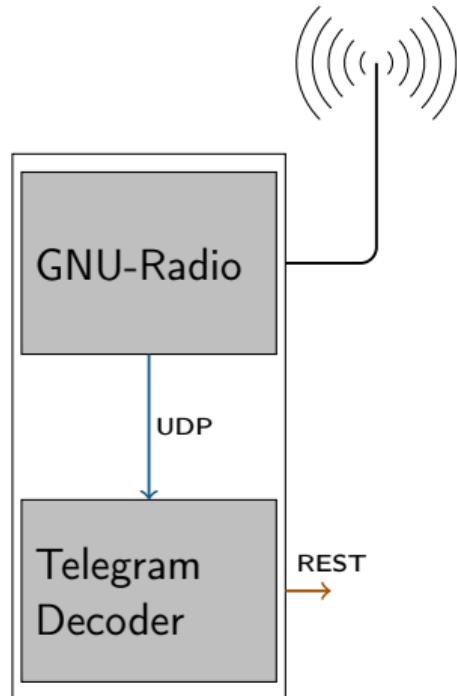
- ▶ Unique junction id with more than 300 received telegram is 325
- ▶ 325 junctions received in a period of 3 months.
- ▶ We can see 70–80% of all reporting points of the city.

Figure 12: Statistic of traffic lights in dresden
from urbic 01/2014 ↗

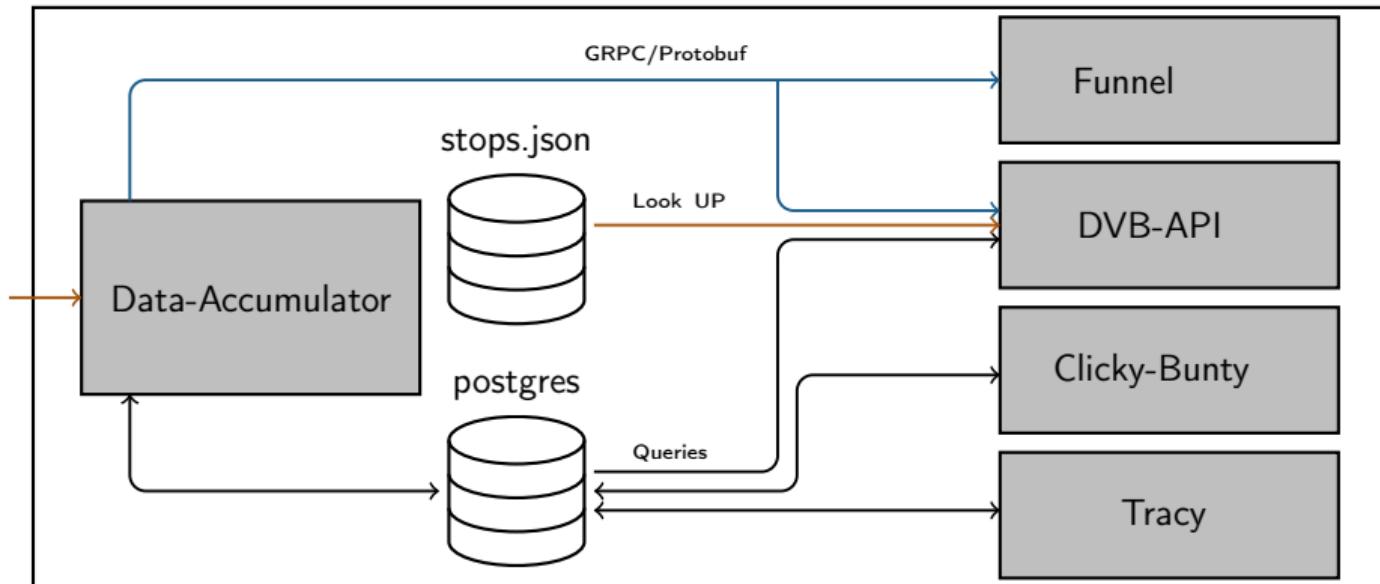


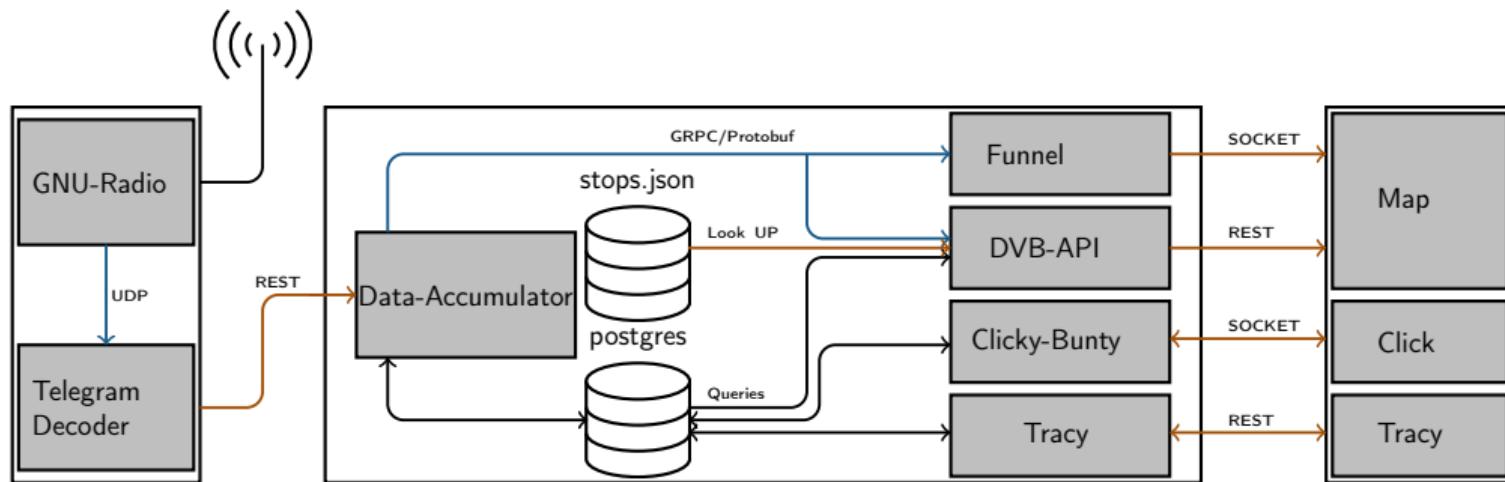
Figure 13: Station Barkhausenbau

- ▶ GDR powersuply casing (10€)
 - ▶ Dell Wyse 3040 (70€)
 - ▶ Rad1o Badge (e.g RTL SDR 30€)
 - ▶ Hardware Filter (20€)
 - ▶ Antenna (15€)
 - ▶ Miscellaneous items (15€)
 - ▶ Healthy amounts of kapton
- ⇒ 160€ per Station



- ▶ Region specific encoding depending on quirks from the city
- ▶ Currently only parses R09.16 and everything else is recorded as raw-telegram
- ▶ Authenticates with station UUID and token







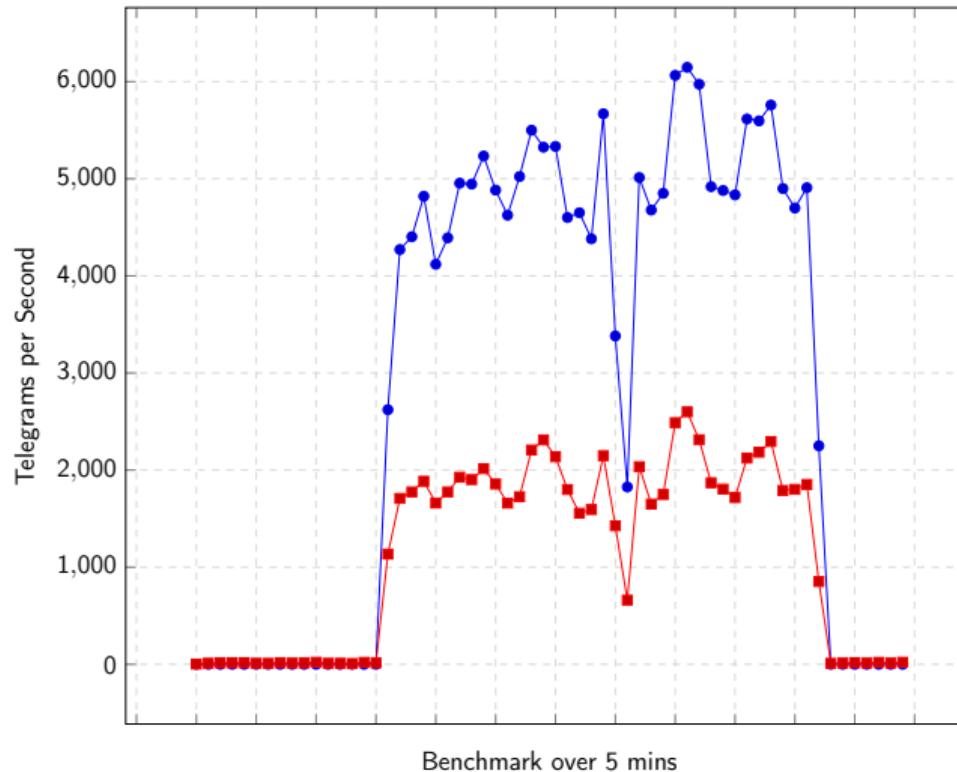
```
dump-dvb.gnuradio = {  
    enable = true;  
    frequency = 170790000;  
    offset = 20000;  
    device = "";  
    RF = 14;  
    IF = 32;  
    BB = 42;  
};  
  
dump-dvb.telegramDecoder = {  
    enable = true;  
    server = [  
        "https://dump.dvb.solutions"  
        "http://dump.staging.dvb.solutions"  
    ];  
};
```

- ▶ Building complete Images for x86 and aarch64 on hydra
- ▶ custom nixos options for all the services
- ▶ wireguard for the receivers that we maintain for punching NAT

Figure 14: users receiver configuration



Throughput Benchmark





- ▶ Influx
 - ▶ Database dumps costs 14GB of RAM
 - ▶ Server died hourly
- ▶ Single point of truth
 - ▶ Implement Protocols in Libraries which is then used by both parties.
- ▶ Hardware homogeneity



- ▶ 7.7 million telegrams are already recorded
- ▶ hourly and daily dumps can be fetched from <https://files.dvb.solutions>



```
{  
  "time":1662932144,  
  "station":"97d028ec-43e2-4473-...",  
  "region":0,  
  "telegram_type":16,  
  "reporting_point":8366,  
  "junction":209,  
  "direction":1,  
  "request_status":2,  
  "delay":0,  
  "priority":0,  
  "direction_request":0,  
  "line":4,  
  "run_number":9,  
  "destination_number":31,  
  "train_length":0  
}
```

- ▶ <https://socket.dvb.solutions>
- ▶ Configurable Filters (region, line, junction)
- ▶ Deduplicated
- ▶ Very raw



- ▶ <https://map.dvb.solutions/stop/<regionid>.json>
- ▶ <https://map.dvb.solutions/stop/all.json>
- ▶ <https://map.dvb.solutions/graph/<regionid>.json>
- ▶ <https://map.dvb.solutions/graph/all.json>



- ▶ <https://api.dvb.solutions>
- ▶ GET /vehicles/0/all
- ▶ POST /vehicles/0/query
- ▶ POST /network/0/estimated_travel_time
- ▶ POST /static/0/coordinates



Call for Action



- ▶ Frontend
 - ▶ **Tofu**: landing page (technology open)
 - ▶ **Tracy**: Mapping Assistant (technology open)
 - ▶ **Click**: User + Station Management (elm-lang)
 - ▶ **Windshield** Live Map
- ▶ setup stations
- ▶ map your city
- ▶ find frequencies
- ▶ test images on new hardware
- ▶ give us roof access
- ▶ All our code is on GitHub:
<https://github.com/dump-dvb/> ↗

Questions