



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



Rolf Scheuner

Representation of Internet Path Transparency

Semester Thesis SA-2018-25
June 2018 to September 2018

Tutor: Mirja Kühlewind
Co-Tutor: Brian Trammell
Supervisor: Laurent Vanbever

Acknowledgments

I thank my supervisors Mirja Kühlewind and Brian Trammell for their great support during my semester project. Whenever I stuck with a problem they helped me out with the right idea. In each meeting they shared their passion and inspiration with me.

Abstract

During this semester thesis a web front end for the Path Transparency Observatory (PTO) was designed and implemented. It visualizes data about Internet path transparency held in the PTO. It was designed and implemented during the last few years by the Networked Systems Group (NSG). Data from several measurement campaigns is already available during this project. Nonetheless, the design of the web front end is generic enough to allow adding visualizations of new measurement campaigns without changing JavaScript, CSS or HTML sources.

Contents

1	Introduction	11
1.1	Situation	11
1.1.1	Path transparency	11
1.1.2	ECN	11
1.2	Motivation	11
1.3	The Task	12
2	Background	13
2.1	PTO	13
2.1.1	Raw Data	13
2.1.2	Observation Data	13
2.1.3	Queries	14
2.2	PATHspider	14
2.3	Tracebox	14
2.4	C3js / D3	14
3	Design	15
3.1	Users	15
3.1.1	Simple User	15
3.1.2	Advanced User with API Key	15
3.1.3	PTO Administrator	15
3.2	Requirements	15
3.2.1	Data Overview	15
3.2.2	Data Visualization	15
3.2.3	Drill down	16
3.2.4	Resource Monitoring	16
3.2.5	Cached Queries	16
3.2.6	Query Composer	16
3.2.7	Data work flow	16
3.3	Charts Comparison	17
3.3.1	Simple Stacked Bars Chart	17
3.3.2	Two Y-Axes Chart	17
3.3.3	Two Sub Charts, monolithic Volume	18
3.3.4	Two Sub Charts, grey scaled Volume	18
4	Implementation	19
4.1	Architecture	19
4.2	File Structure	19
4.2.1	navbar.html, footer.html	19
4.2.2	app.js	19
4.2.3	pto.css	19
4.3	API Key Page	20
4.4	Available Measurements	20
4.5	Chart Pages	20
4.6	Data Charts	21
4.6.1	Shares Sub-Chart	21
4.6.2	Volumes Sub-Chart	21

4.6.3	Observation List	22
4.7	Resource Overview	23
5	Results	25
5.1	Available Data	25
5.2	Data Comparability	25
5.3	Adjustments to the PTO	25
5.3.1	CORS Handling	25
5.3.2	Query Retrieve	25
5.3.3	Cache Invalidation	26
6	Conclusion	27
6.1	Summary	27
6.2	Outlook	27
A	ECN Charts	29
B	Admin Documentation	35
B.1	Configuration	35
B.2	Page Configuration	35
B.3	Chart Configuration	36
C	Current Configuration	37
C.1	Global Configuration	37
C.2	Page Configuration	37
C.2.1	ECN	37

List of Figures

3.1	Simple stacked bars chart	17
3.2	Two Y-axes chart	17
3.3	Two sub charts, monolithic volume chart	18
3.4	Two sub charts, grey scaled volume chart	18
4.1	The matrix chart giving an overview of the available observation data	20
4.2	The data chart displaying all measurements regarding ecn.multipoint.negotiation.	21
4.3	The observation list of ecn.connectivity.offline of August 20th 2018.	22
4.4	The meta data of the observation set 118.	23
4.5	A list of all queries.	23
5.1	Data chart of ecn.connectivity with working and broken condition hidden	26
A.1	Chart for ecn.connectivity	29
A.2	Chart for ecn.stable.connectivity	30
A.3	Chart for ecn.multipoint.connectivity	30
A.4	Chart for ecn.negotiation	31
A.5	Chart for ecn.stable.negotiation	31
A.6	Chart for ecn.multipoint.negotiation	32
A.7	Chart for ecn.ipmark.ce	32
A.8	Chart for ecn.ipmark.ect0	33
A.9	Chart for ecn.ipmark.ect1	33

Chapter 1

Introduction

This chapter describes the current situation and a definition of path transparency is provided. The motivation for this project and the actual task will follow.

1.1 Situation

According to the Internet's original design, packets are sent from a source to a target without any changes. But middleboxes have been installed for performance and security reasons. They might filter or alter packets for these purposes which contradicts the original idea. The problem with middleboxes is that they are designed to work with a certain protocol stack. However, new protocols and protocol extensions have been developed to increase performance of specific applications. Middleboxes might hinder the effectiveness of such protocols by not supporting or even blocking traffic with that feature enabled.

1.1.1 Path transparency

A path is called transparent to a certain feature if packets with the feature enabled are treated the same way as packets without that feature. Middleboxes can be an impairment of path transparency.

1.1.2 ECN

Explicit Congestion Notification is an extension to the TCP protocol which allows to handle traffic congestion without dropping packets in a cooperative way. Most data available during this project is about ECN, therefore all visualizations shown here are based on ECN measurements.

1.2 Motivation

The EU founded MAMI project [3] addresses the issues with middleboxes by developing a Middlebox Cooperation Protocol. In the scope of the MAMI project, a Path Transparency Observatory (PTO) was developed. The PTO holds data about path transparency and allows to organize, normalize, analyze and query that data.

A former version of the PTO had a web front end which provided a visual query composer and basic visualizations for the query results. It also showed a list of precomputed (cached) queries. The new version of the PTO needs a front end to make its data more accessible and therefore more valuable.

1.3 The Task

The goal of this semester thesis is to design a web front end for the PTO based on an analysis of the front end of the former version of the PTO. This includes the definition of users and their needs as well as an experimental implementation of the front end to test and refine the design.

[6]

Chapter 2

Background

This chapter describes the tools and libraries which are important to understand the context of this semester thesis.

2.1 PTO

Several measurement campaigns have already been run for the MAMI project. The Path Transparency Observatory is the central tool where all data from these campaigns comes together. It was developed and is maintained by the NSG in the scope of the MAMI project.

As described in the deliverable D1.2 of the MAMI project:

"The architecture of the PTO has three main goals in collecting large-scale data and making them available for research, protocol engineering, and operation: maximizing comparability of data across diverse measurement campaigns, formats and tools; supporting repeatability of experimentation; and providing protection of raw measurement data." [1]

The current version of the PTO is designed as a RESTful API and implemented in Go. It has the following types of resources: raw data, observation data and queries. These resource types are described in the following subsections. There is a permission system in the PTO based on API Keys. The basic idea of the permission system is "decreasing access with increasing detail". API Keys can be obtained from the PTO developers.

Further information about the PTO and its resource types, please refer to [1] and [12].

2.1.1 Raw Data

Raw data is organized in campaigns. A campaign corresponds to one series of comparable measurements. The results from measurements are uploaded as raw data. Raw data can have various formats because it might have been produced by completely different tools. It is stored in archive files and only accessible by its owner and users authorized by the owner.

2.1.2 Observation Data

Normalizers are used to generate comparable observation data from raw data. It is organized in observation sets and stored in observation files with a well defined format. Observations are also loaded into a relational database to make them queryable. New observation data can also be obtained by running analyzers on existing observation data. The PTO also provides the resolution of the provenance of observations, i.e., each observation can be tracked down to the raw data it originates from.

Observations are defined by a start time, an end time, a condition and a path (including the target) the packet has traveled. The condition has a well defined structure, e.g., `ecn.connectivity.works`: `ecn` is the feature that has been tested, `ecn.connectivity` is the aspect of the feature being tested and `ecn.connectivity.works` is the full condition observed in the test.

2.1.3 Queries

Queries provide a high level view of the observation data. They are not directly run on the database but processed by the PTO which takes care of caching them and running them on the database. Caching is needed because queries might take over an hour to return a result. The queries are submitted to the PTO in a URL encoded form.

2.2 PATHspider

PATHspider [4] is a tool for A/B testing of path transparency in the Internet. Actual measurements are taken by writing a plugin for the features to be tested. Various plugins have been written during the MAMI project, e.g., for ECN, MSS or TFO. In the MAMI project, the A/B testing is mostly done against Amazon's Alexa top 1 million list. Data generated by PATHspider is uploaded to the PTO together with the available meta data for further analysis. It builds the core information held in the PTO.

2.3 Tracebox

Tracebox [11] is a tool to reconstruct the path a certain package has traveled and to detect which hops might have changed the package. Therefore, it can be used to identify the source of an impairment on path transparency. Data generated by Tracebox is combined with the data from PATHspider and gives additional information about where impairments occurred.

2.4 C3js / D3

C3js [10] is a library for reusable charts based on the JavaScript library D3 (Data Driven Documents) [2]. It provides powerful configuration of advanced data visualization charts. It is used in this project to visualize the data retrieved by queries from the PTO.

Chapter 3

Design

This chapter explains how the front end was designed. First the users are defined and then requirements are extracted from the users needs.

3.1 Users

3.1.1 Simple User

As a simple user we consider someone who is interested in information about Internet protocol evolution. We do not specify or restrict the motivation for his interest. He is familiar with commonly used Internet technologies and protocols, but does not necessarily know about the MAMI project or the PTO back end. This might be an operator of Internet infrastructure.

3.1.2 Advanced User with API Key

An advanced user with an API Key is someone who is already familiar with the PTO. He knows the documentation [12] and has had contact with the PTO developers to get an API Key. He has run his own measurement campaigns and wants to upload his data to the PTO. This user also wants more detailed information about failed measurements (e.g. to which targets a connection could not be established with ECN enabled).

3.1.3 PTO Administrator

This is an administrator or developer of the PTO who wants to have a quick overview of the state of the PTO. He wants to know what campaigns are available and who is their owner. He also needs to know what queries are currently cached to identify failed ones.

3.2 Requirements

3.2.1 Data Overview

All defined users need an overview of the data in the PTO. The overview provides a temporal resolution of one month and an informational resolution of a feature, i.e., it shows for which features in which months data is available. It was considered giving a temporal resolution of one year, expandable to its months. However, we came to the conclusion that a direct view at the months is more intuitive and faster for the user.

3.2.2 Data Visualization

Simple and advanced users want to have visualizations of the data in the PTO. They want to see how the support of Internet protocols has evolved. Advanced users are also interested in more

detailed information about the results, e.g., to which targets no connection could be established with a certain feature used.

Data visualizations (charts) are organized in pages. A page holds charts which belong together "somehow". The design does not specify requirements on the "somehow" to give the administrator maximal flexibility in grouping charts into pages. At the time of writing there is one chart page configured per feature. Pages are defined in the file *json/config.json* and configured in a separate page configuration file per page.

The page configuration file also contains the chart definitions and configurations. The idea of a data chart is to visualize the evolution of a certain aspect of a feature over time. A chart has its own query which provides the data to be visualized. It is therefore not restricted to only show data about one aspect of one feature.

We had a lot of discussion about how to visualize the data. We decided to display shares and volumes of a query result in two sub charts to provide comparability between measurements with different amount of data, while still having the information about volumes. The different options we considered are described in Section 3.3.

3.2.3 Drill down

To provide more detailed information about the observations which led to a certain result in the chart, the front end offers drill down on the bars of a chart. This will submit a new query to the back end and show all observations which led to the clicked bar on the chart. These observations are paged by the back end by 1000 observations per page. For sorting observations, the front end would have to load all pages first. This is not feasible, since there may be over a million observations behind a bar.

3.2.4 Resource Monitoring

Advanced users and administrators (users with an API Key) are provided an overview of the meta data about the available resources per resource type. That means they have a list of meta data about raw data (organized in campaigns), observation data (organized in observation sets) and cached queries. An administrator can use this list to identify failed queries and inspect them.

3.2.5 Cached Queries

The PTO already offers query caching because queries often run for up to an hour and longer. The delay was also the reason why we decided to only use cached queries for data charts. However, if a query result is not cached and not being computed, the front end asks the user if it should submit the query to the back end. It takes also care of informing the user if the query is still pending or has failed.

3.2.6 Query Composer

The query composer for the former version of the PTO was designed to give the user full flexibility. However, it required a substantial knowledge about the PTO and what data is in the PTO. Because of that, we did not integrate it into the new front end. Advanced users still have the possibility to submit their own queries by directly working with the PTO API.

3.2.7 Data work flow

The whole data work flow with raw data upload, data normalization and analysis has not been integrated into the front end since the PTO API already addresses these tasks.

3.3 Charts Comparison

3.3.1 Simple Stacked Bars Chart

This stacked bars chart shows the number of observations of conditions (see Fig. 3.1). It was discarded because it does not provide poor comparability of measurements with different volume.

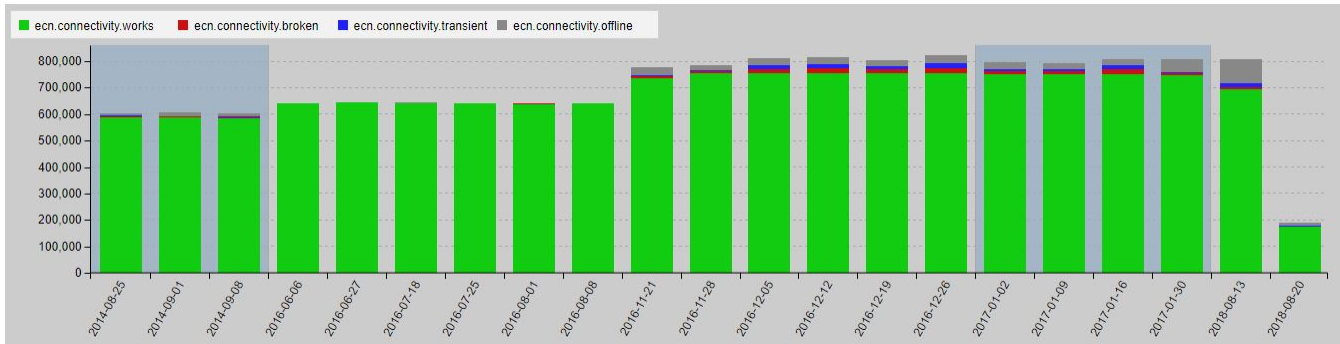


Figure 3.1: Simple stacked bars chart

3.3.2 Two Y-Axes Chart

This stacked bars chart shows the shares of the conditions observed normalized to one hundred percent. The volume was drawn as an overlaid line and scaled on a second y-axis (see Fig. 3.2). This chart was discarded because it might become too overloaded.

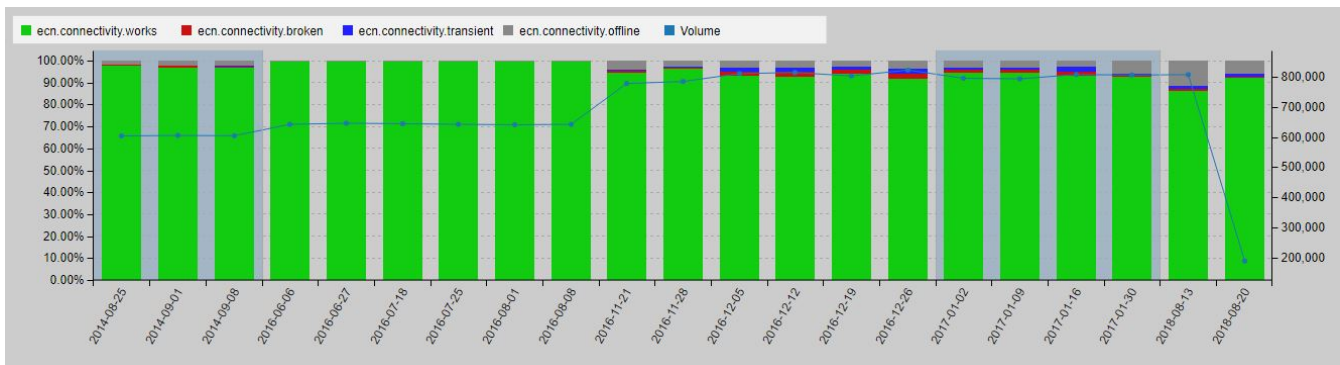


Figure 3.2: Two Y-axes chart

3.3.3 Two Sub Charts, monolithic Volume

This chart shows shares and volumes in separate sub charts. The shares sub chart is a stacked bars chart of the conditions observed normalized to one hundred percent. The volume is drawn in the second sub chart as bar chart with only one data series (the volume as a whole) (see Fig. 3.3). This chart was discarded because different conditions could not be distinguished in the volume chart.

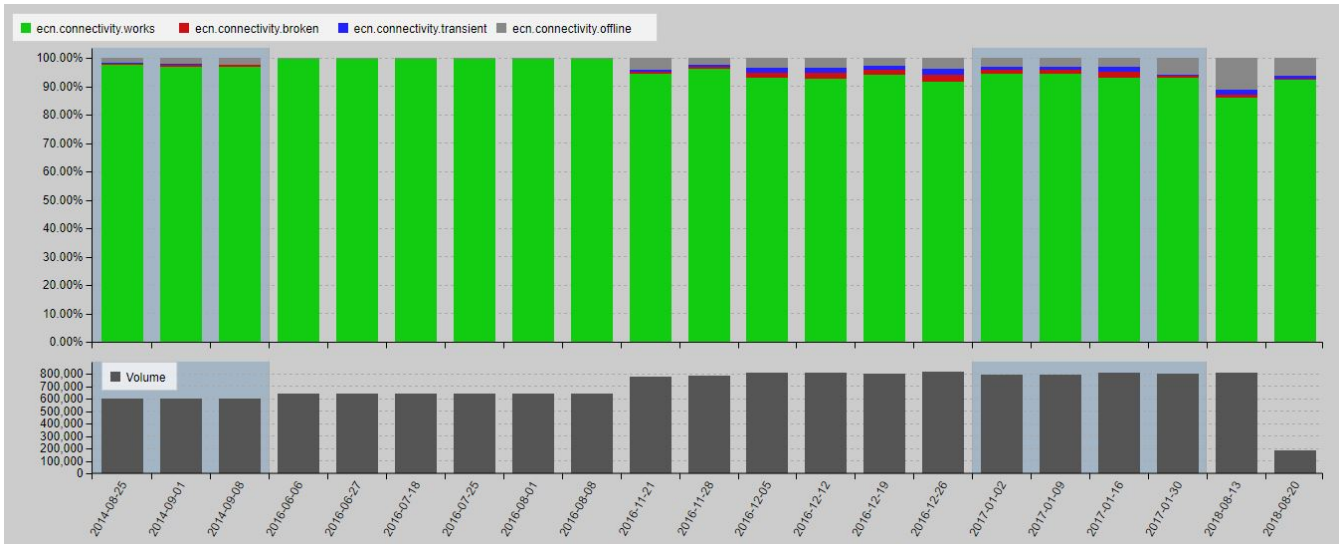


Figure 3.3: Two sub charts, monolithic volume chart

3.3.4 Two Sub Charts, grey scaled Volume

This is basically the same chart as the one described above with the difference that the volume chart is a stacked bars chart with grey scaled volumes (see Fig. 3.4). This one was chosen because it makes shares comparable independently from the amount of measurements while still giving detailed volume information.

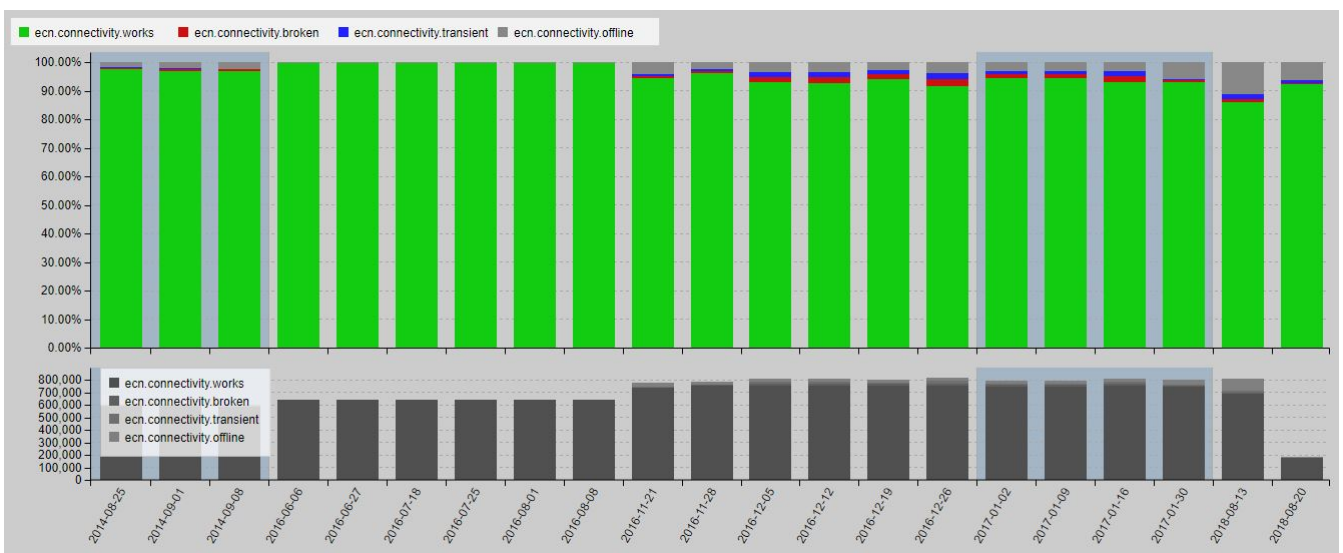


Figure 3.4: Two sub charts, grey scaled volume chart

Chapter 4

Implementation

This chapter illustrates implementation details and shows how the design from Chapter 3 was implemented.

4.1 Architecture

The PTO web front end is implemented using the standard web technologies HTML, CSS and JavaScript. In order to minimize external dependencies, only Vanilla JavaScript is used. The only libraries used are C3js and D3. The design of the website, including style sheets from [5], is reused from the former version of the website. The implementation described in this chapter was used to refine the design step-by-step and to achieve a better understanding of the data in the PTO.

To meet the generic design, the implementation does not have any code dependencies on data in the PTO. The only dependencies are the queries configured in JSON files. The user will be informed if a requested query is not available. He is asked if he wants to submit the query to the back end. This requires an API Key with the corresponding permissions.

4.2 File Structure

The source and configuration files are organized in the following way: HTML files in the root directory, style sheets in /css, JavaScript files in /js, JSON files in /json and graphics in /img. This section describes the roles of the source files with general purpose while the rest of the source files are mentioned in the following sections.

4.2.1 navbar.html, footer.html

These two files hold the HTML code for the navigation bar and the footer. The code is separated from the other HTML files to be reusable on multiple pages.

4.2.2 app.js

This file holds JavaScript logic used on multiple pages. It takes care of loading the navigation bar and footer as well as accessing the stored API Key. It also provides utilities for query submission and query result retrieving. It is loaded on all pages of the front end.

4.2.3 pto.css

All styles for the PTO web front end (excluding third party styles) are defined in this file.

4.3 API Key Page

The API Key page enables the user to store an API Key and therefore to get access to advanced features. It displays the current API Key (if there is one) and the queries used by the front end. The API Key is stored via the *localStorage* interface in JavaScript. The files *apekey.html* and *js/apikey.js* take care of that.

4.4 Available Measurements

The overview of the available measurements is shown on the landing page of the website (*index.html*). It is implemented as table with the features as columns and the time periods (months) as rows (see Fig. 4.1). Adding new features or new time periods to the database will not require any code changes, because it is completely generic. The data for the overview is provided by a single query counting all observations grouped by feature and month. The query is defined in the configuration file *json/config.json*. The file *js/directory.js* takes care of loading and displaying the overview.

Month \ Feature	ecn	mss	pathspider
2018-09	✓	✓	✓
2018-08	✓	✓	✓
2017-02	✓	✗	✗
2017-01	✓	✗	✗
2016-12	✓	✗	✗
2016-11	✓	✗	✗
2016-08	✓	✗	✗
2016-07	✓	✗	✗
2016-06	✓	✗	✗
2014-09	✓	✗	✗
2014-08	✓	✗	✗

Figure 4.1: The matrix chart giving an overview of the available observation data

4.5 Chart Pages

A chart page is defined by an entry in the file *json/config.json*, the configuration of the page and the charts is in a separate file, specified in the page entry in *json/config.json*. Each chart page has its own link in the navigation bar at the top of the website. The page configuration file holds a page title, a page description which should inform the user about what he is looking at on this page and a list of charts to be displayed on this page. All chart pages are built on the same HTML page. The page key (matching the property name in *json/config.json*) of the page to be loaded is passed by the URL parameter *page* (e.g. the URL of the ECN page looks like this: <https://baseUrl/charts.html?page=ecn>).

The empty template for chart pages is in *charts.html* and *js/charts.js* takes care of populating the page with the data charts according to the page configuration.

4.6 Data Charts

The data charts show the shares and the volumes of the conditions of the result of the query assigned to the chart. There are two sub-charts: one for the shares of conditions and one for the volumes (see Fig. 4.2).

The charts use a non-linear, non-continuous time scale on which the data is grouped into weeks indicated by date of each Monday. The blue and white background regions make different years visible more clearly. Clicking on the legend items will toggle the visibility of the according data series. This allows the user to see proportions of small shares by hiding big ones.

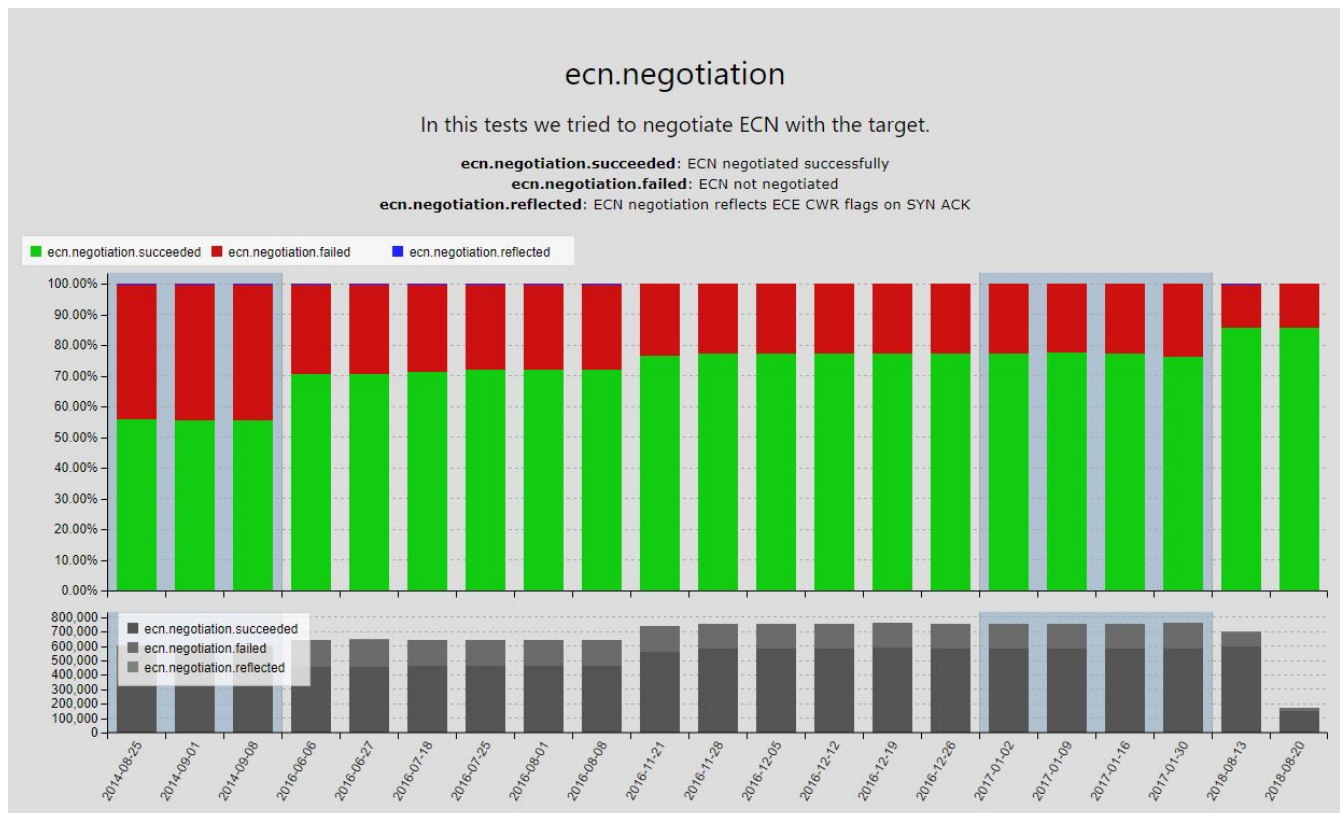


Figure 4.2: The data chart displaying all measurements regarding ecn.multipoint.negotiation.

The chart configuration in the page configuration file holds a chart title, a chart description, a list of conditions to be shown in the chart, the colors of the conditions and the descriptions of the conditions.

4.6.1 Shares Sub-Chart

The shares sub-chart shows its data normalized to one hundred percent. The front end's JavaScript code takes care of the normalization of the data because C3js does not offer that feature for stacked bar charts.

4.6.2 Volumes Sub-Chart

The volumes sub-chart shows the total number of targets that have been observed under a certain condition. The color of the conditions is automatically grey scaled to give the user an intuition of the volume as a whole while still having different conditions visible.

4.6.3 Observation List

A click on a bar of the data chart opens the observation list (see Fig. 4.3) in a modal dialog. If there are more than one thousand observations, the back end provides pagination of the data. The front end shows buttons to the first, previous, next and last page. The observations are retrieved by a query. If the query is not already cached, the front end asks whether it should submit a query for these results. If the user has no API Key, he is informed that he needs an API Key to submit a query. The front end is able to build that query based on the bar that has been clicked. The logic for this drill down function is contained in *js/obslist.js*.

Row	Obs-Set ID	Time Start	Source	Path	Target	Value
1	118	2018-08-20T00:00:09Z	178.128.255.68 AS14061	*	206.188.192.201	0
2	118	2018-08-20T00:00:18Z	178.128.255.68 AS14061	*	207.104.144.142	0
3	118	2018-08-20T00:00:38Z	2a03:b0c0:2:f0::35:4001 AS14061	*	240c:f0:ffe3::7	0
4	118	2018-08-20T00:00:28Z	178.128.255.68 AS14061	*	66.180.167.17	0
5	118	2018-08-20T00:00:48Z	178.128.255.68 AS14061	*	144.208.64.229	0
6	118	2018-08-20T00:00:58Z	178.128.255.68 AS14061	*	207.210.91.243	0
7	118	2018-08-20T00:00:58Z	178.128.255.68 AS14061	*	23.235.46.80	0
8	118	2018-08-20T00:01:24Z	178.128.255.68 AS14061	*	216.224.132.100	0
9	118	2018-08-20T00:01:34Z	178.128.255.68 AS14061	*	35.161.157.228	0
10	118	2018-08-20T00:01:37Z	2a03:b0c0:2:f0::35:4001 AS14061	*	2404:8280:a222:bbbb:bba1:69:ffff:ffff	0
11	118	2018-08-20T00:01:37Z	178.128.255.68 AS14061	*	34.225.211.45	0
12	118	2018-08-20T00:01:37Z	178.128.255.68 AS14061	*	45.32.233.22	0
13	118	2018-08-20T00:01:37Z	178.128.255.68 AS14061	*	141.162.101.251	0
14	118	2018-08-20T00:01:47Z	178.128.255.68 AS14061	*	81.138.19.226	0
15	118	2018-08-20T00:01:47Z	178.128.255.68 AS14061	*	74.54.232.135	0
16	118	2018-08-20T00:01:57Z	178.128.255.68 AS14061	*	199.191.101.20	0
17	118	2018-08-20T00:02:07Z	178.128.255.68 AS14061	*	50.236.160.103	0
18	118	2018-08-20T00:01:57Z	178.128.255.68 AS14061	*	70.172.196.220	0
19	118	2018-08-20T00:02:22Z	178.128.255.68 AS14061	*	184.161.33.93	0
20	118	2018-08-20T00:02:07Z	178.128.255.68 AS14061	*	133.153.71.32	0
21	118	2018-08-20T00:02:43Z	178.128.255.68 AS14061	*	192.48.241.202	0
22	118	2018-08-20T00:02:33Z	178.128.255.68 AS14061	*	34.206.141.91	0
23	118	2018-08-20T00:02:43Z	178.128.255.68 AS14061	*	67.23.226.179	0
24	118	2018-08-20T00:02:43Z	178.128.255.68 AS14061	*	35.153.45.113	0
25	118	2018-08-20T00:02:43Z	178.128.255.68 AS14061	*	66.228.58.221	0
26	118	2018-08-20T00:02:43Z	178.128.255.68 AS14061	*	174.77.1.198	0
27	118	2018-08-20T00:02:53Z	178.128.255.68 AS14061	*	64.147.8.35	0
28	118	2018-08-20T00:02:53Z	178.128.255.68 AS14061	*	123.1.154.121	0
29	118	2018-08-20T00:03:03Z	178.128.255.68 AS14061	*	173.236.194.1	0
30	118	2018-08-20T00:03:18Z	178.128.255.68 AS14061	*	82.196.8.24	0
31	118	2018-08-20T00:03:32Z	178.128.255.68 AS14061	*	38.118.71.181	0
32	118	2018-08-20T00:03:46Z	178.128.255.68 AS14061	*	149.129.220.242	0
33	118	2018-08-20T00:03:59Z	178.128.255.68 AS14061	*	71.14.254.210	0
34	118	2018-08-20T00:04:26Z	178.128.255.68 AS14061	*	209.36.33.42	0

Figure 4.3: The observation list of ecn.connectivity.offline of August 20th 2018.

The source and the target of an observation have a link to its page on [8], where more information is available. The link opens a new browser tab to not interrupt the user's work on the PTO website. Clicking on the Obs-Set-ID in an observation list will open another modal dialog displaying the meta data of that observation set (see Fig. 4.4).

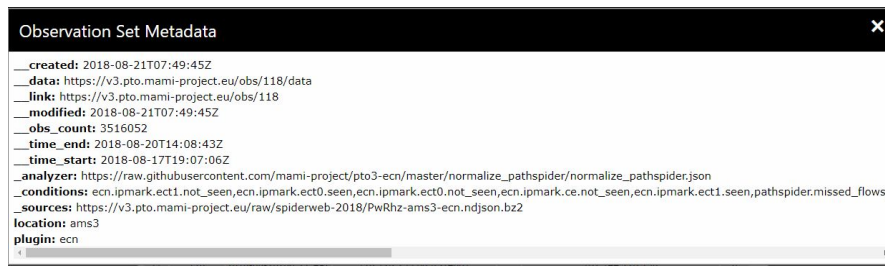


Figure 4.4: The meta data of the observation set 118.

4.7 Resource Overview

Each of the resource types raw data, observation data and queries has its own overview page which all work in the same way. The PTO back end does not offer listing all meta data of all resources of a type. For each resource type, it provides a list of links to the available resource meta data. The front end then loads the meta data retrieved from each link. These meta data objects are then displayed in a completely generic table. This table contains all properties of the meta data objects as columns and each resource as a row. Figure 4.5 illustrates how these tables look like.

__completed	__created	__encoded
2018-09-17T16:10:18+02:00	2018-09-17T16:07:04+02:00	time_start=2017-01-30T00%3A00%3A00Z&time_end=2017-02-05T23%3A59%3A59Z&condition=ecm.multipoint
2018-09-07T16:47:58+02:00	2018-09-07T16:41:11+02:00	time_start=2016-12-12T00%3A00%3A00Z&time_end=2016-12-18T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-07T16:43:05+02:00	2018-09-07T16:38:33+02:00	time_start=2017-01-30T00%3A00%3A00Z&time_end=2017-02-05T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-07T16:39:53+02:00	2018-09-07T16:36:32+02:00	time_start=2018-08-13T00%3A00%3A00Z&time_end=2018-08-19T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-07T13:33:23+02:00	2018-09-07T13:23:40+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.ipmark.ect1
2018-09-07T14:00:47+02:00	2018-09-07T13:23:33+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.ipmark.ect0
2018-09-07T13:32:42+02:00	2018-09-07T13:23:23+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.ipmark.ce&g
2018-09-07T13:37:23+02:00	2018-09-07T13:23:08+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.stable.negot
2018-09-07T13:37:59+02:00	2018-09-07T13:22:59+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.stable.conne
2018-09-07T13:30:45+02:00	2018-09-07T13:22:50+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.multipoint.cc
2018-09-07T13:30:41+02:00	2018-09-07T13:22:38+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.multipoint.n
2018-09-07T14:05:52+02:00	2018-09-07T13:21:57+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.negotiation&
2018-09-07T13:21:39+02:00	2018-09-07T13:21:39+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.connectivity.
2018-09-07T13:17:17+02:00	2018-09-07T12:26:18+02:00	time_start=2014-01-01T00%3A00%3A00Z&time_end=2020-12-31T23%3A59%3A59Z&aspect=ecm.connectivity
2018-09-06T18:33:43+02:00	2018-09-06T18:30:37+02:00	time_start=2017-01-30T00%3A00%3A00Z&time_end=2017-02-05T23%3A59%3A59Z&condition=ecm.multipoint
2018-09-06T17:06:29+02:00	2018-09-06T16:57:29+02:00	time_start=2016-08-01T00%3A00%3A00Z&time_end=2016-08-07T23%3A59%3A59Z&condition=ecm.ipmark.ec
2018-09-05T14:28:45+02:00	2018-09-05T14:24:45+02:00	time_start=2018-08-20T00%3A00%3A00Z&time_end=2018-08-26T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-05T12:43:13+02:00	2018-09-05T12:41:09+02:00	time_start=2018-08-13T00%3A00%3A00Z&time_end=2018-08-19T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-05T12:39:39+02:00	2018-09-05T12:37:38+02:00	time_start=2018-08-20T00%3A00%3A00Z&time_end=2018-08-26T23%3A59%3A59Z&condition=ecm.connectivi
2018-09-04T16:00:10+02:00	2018-09-04T15:56:37+02:00	time_start=2018-08-20T00%3A00%3A00Z&time_end=2018-08-26T23%3A59%3A59Z&condition=ecm.connectivi
2018-08-29T13:01:28+02:00	2018-08-29T12:57:00+02:00	time_start=2017-01-30T00%3A00%3A00Z&time_end=2017-02-05T23%3A59%3A59Z&condition=ecm.multipoint
2018-08-29T13:00:17+02:00	2018-08-29T12:56:01+02:00	time_start=2017-01-30T00%3A00%3A00Z&time_end=2017-01-30T00%3A00%3A00Z&condition=ecm.multipoint

Figure 4.5: A list of all queries.

The overviews of the three resources are provided by the files *cachedqueries.html* and *js/-cachedqueries.js*, *obssets.html* and *js/obssets.js* respectively *raw.html* and *js/raw.js*

Chapter 5

Results

The main result of this project is the design and implementation of the PTO web front end. However, the project has also given more insights into the current state of data in the PTO and changes to the back end have been initiated.

5.1 Available Data

As visible in Figure 4.1 the available data is mainly about ECN. This indicates that more effort needs to be put into running measurement campaigns for other features.

5.2 Data Comparability

The data about one single aspect is in general from different measurement campaigns with different setups. This needs to be considered when looking at the data charts, because it influences the comparability of results from different dates. As visible in Figure 5.1, the share of broken and transient results from ecn.connectivity measurements increases over time. That is contra-intuitive and indicates that a different measurement setup was used for the measurements in mid 2016.

5.3 Adjustments to the PTO

The adjustments in the back end are not part of this project, but they were required for the experimental implementation of the front end. Beside some minor bug fixes, three major changes to the PTO have been initiated. They are described in the following subsections.

5.3.1 CORS Handling

Since the new front end is not on the same origin as the PTO, it needs to offer CORS handling (Cross-Origin Resource Sharing). This includes adding certain headers to responses and handling preflight requests. See [7] for more information about CORS.

5.3.2 Query Retrieve

This change was made to retrieve query meta data (if it is available) for an encoded query. Prior to this change, finding query meta data from an encoded query required looping through all query meta data to find the right one. Since a separate HTTP request is needed for each query, this would generate a big overhead.

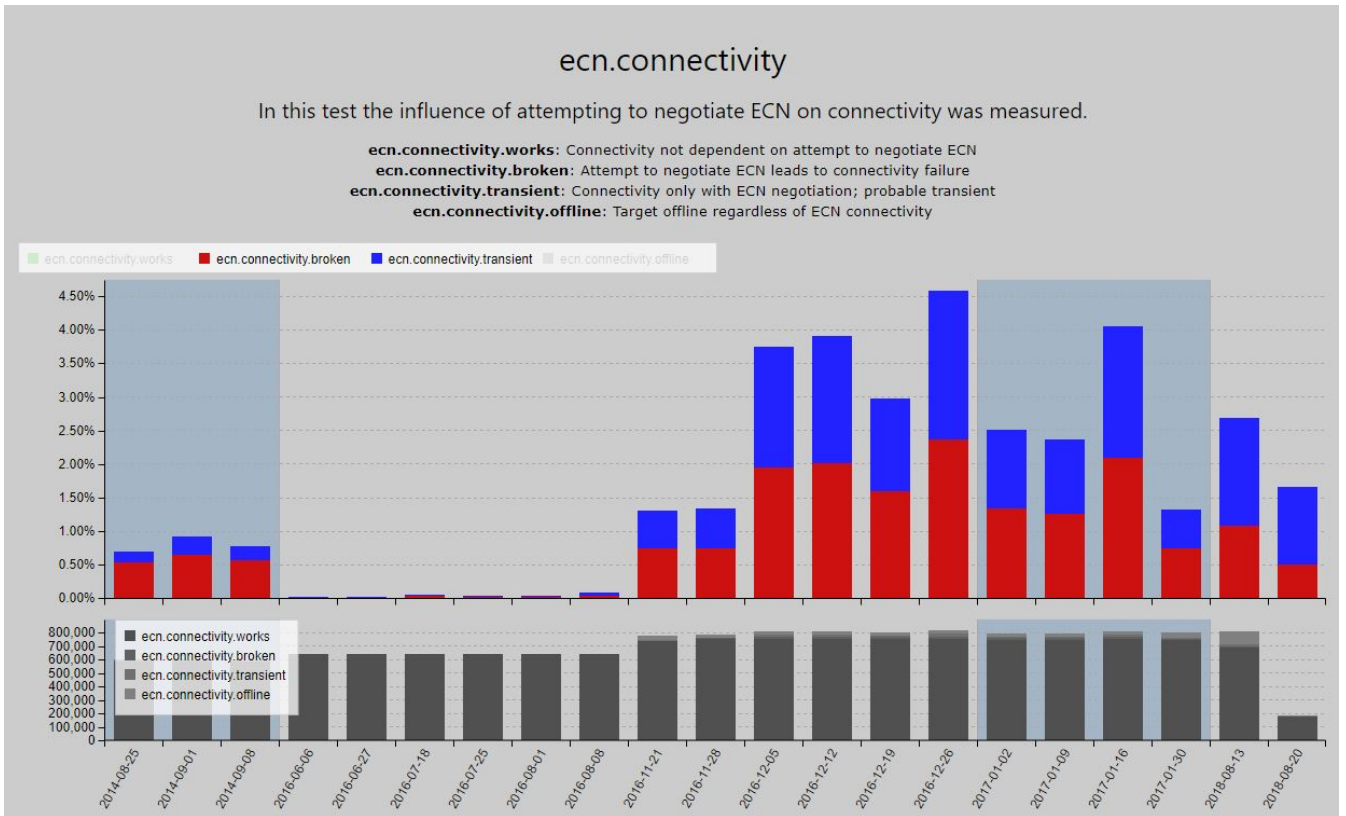


Figure 5.1: Data chart of ecn.connectivity with working and broken condition hidden

5.3.3 Cache Invalidation

When data in the database changes some cached query results turn invalid. This change aims to identify the affected queries and mark their cached results as dirty. At the time of writing this request is not completed, meaning the queries need to be identified and updated/rerun manually.

Chapter 6

Conclusion

6.1 Summary

The goal of this project was to design and implement a web front end for the PTO. This was achieved by the design described in Chapter 3 and the implementation explained in Chapter 4. Further insights into the current state of the PTO have been achieved and the PTO back end API was extended to fit the needs of the front end.

6.2 Outlook

In order to make the PTO an even more valuable source of information about path transparency in the Internet, more measurement campaigns must be run. These campaigns should also accomplish the need of data comparability, i.e., it needs to be clear what is measured and how it is measured. As more data will be available in the PTO, new charts can be configured. The PTO and its front end are ready for handling and visualizing that data.

Appendix A

ECN Charts

This chapter shows all charts that have been drawn for ECN.

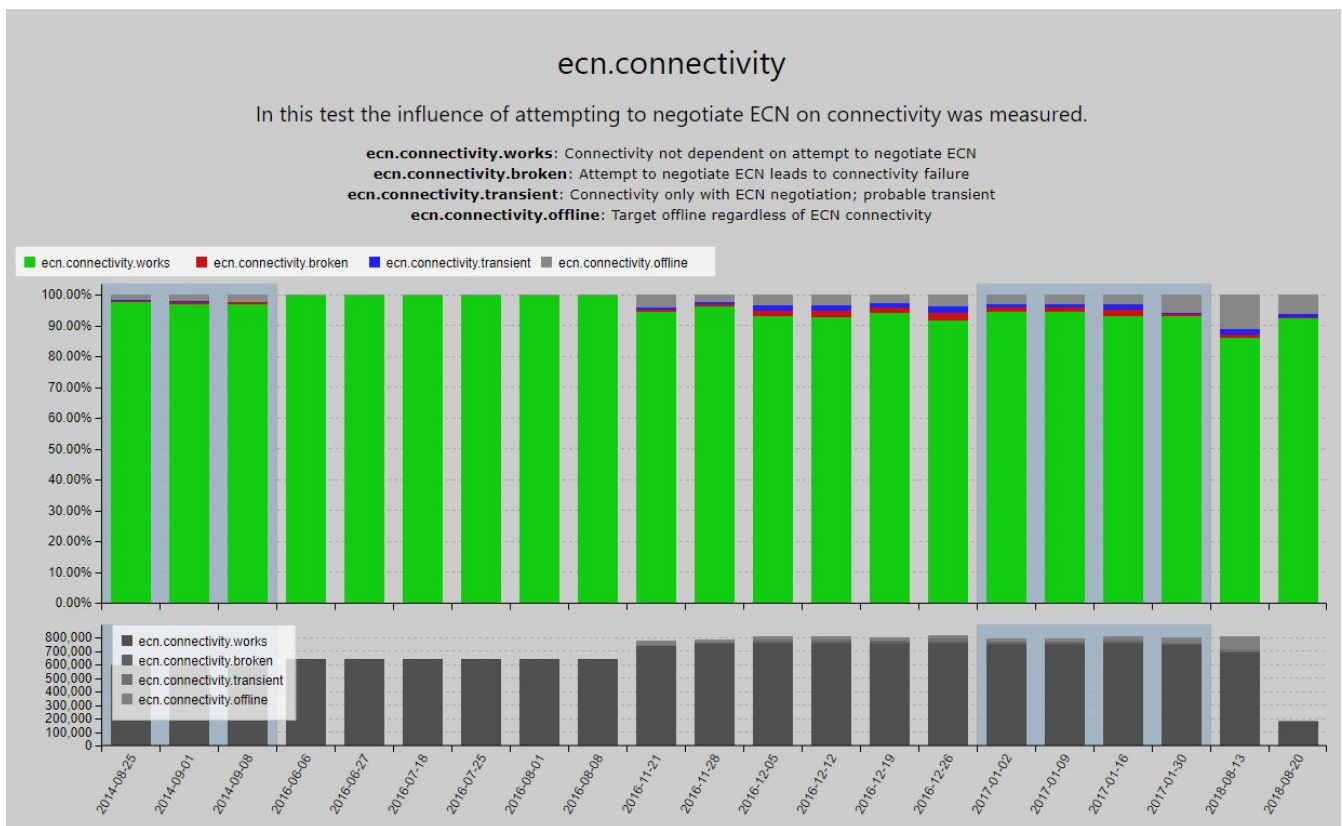


Figure A.1: Chart for ecn.connectivity

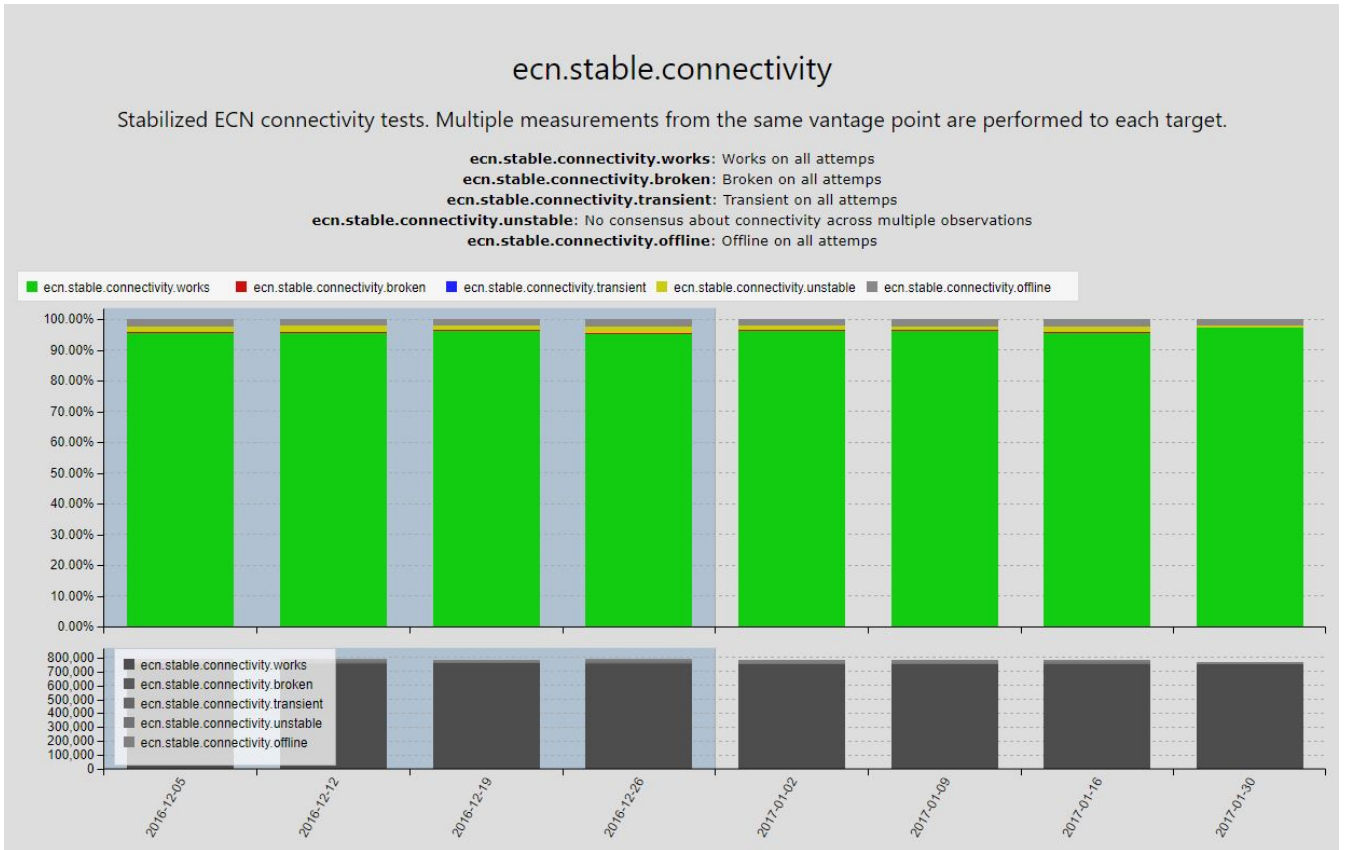


Figure A.2: Chart for ecn.stable.connectivity

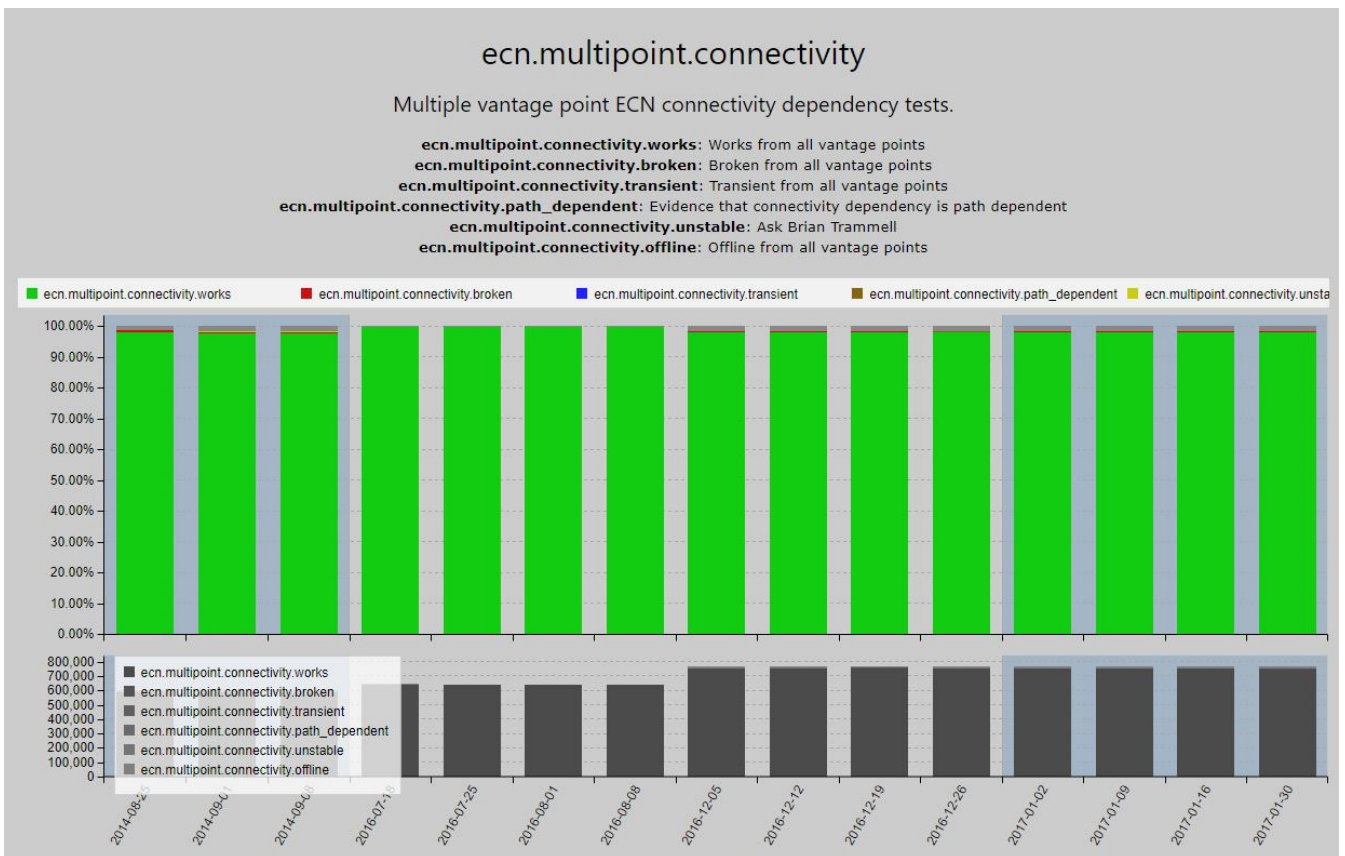


Figure A.3: Chart for ecn.multipoint.connectivity

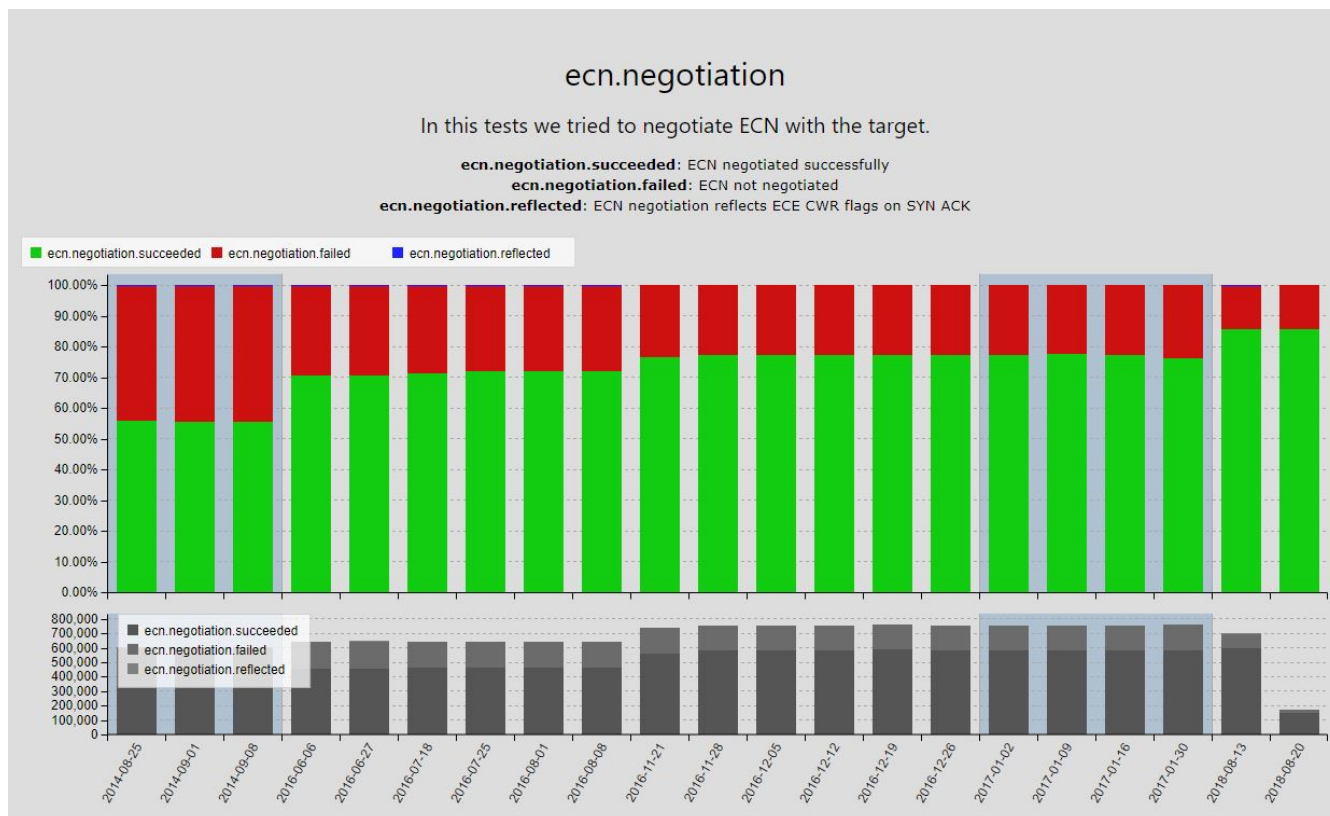


Figure A.4: Chart for ecn.negotiation

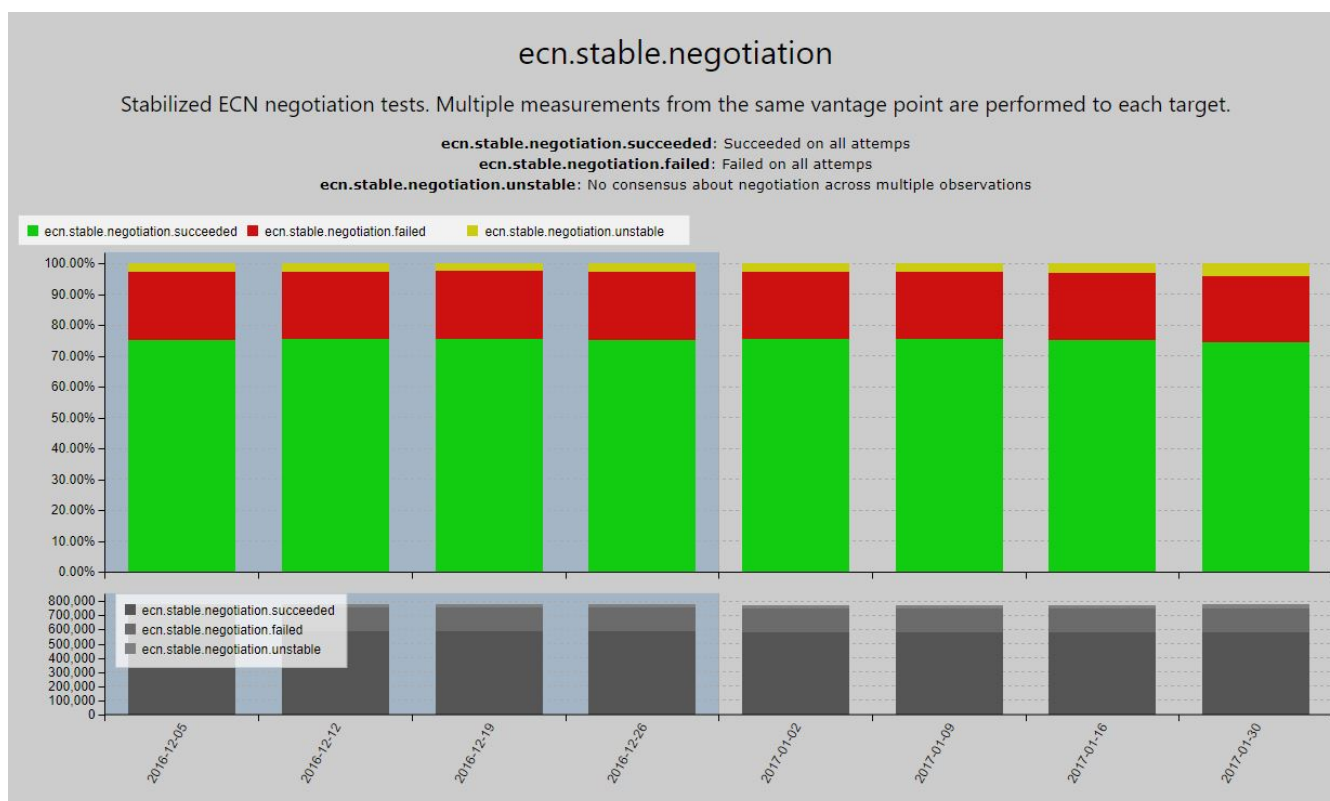


Figure A.5: Chart for ecn.stable.negotiation

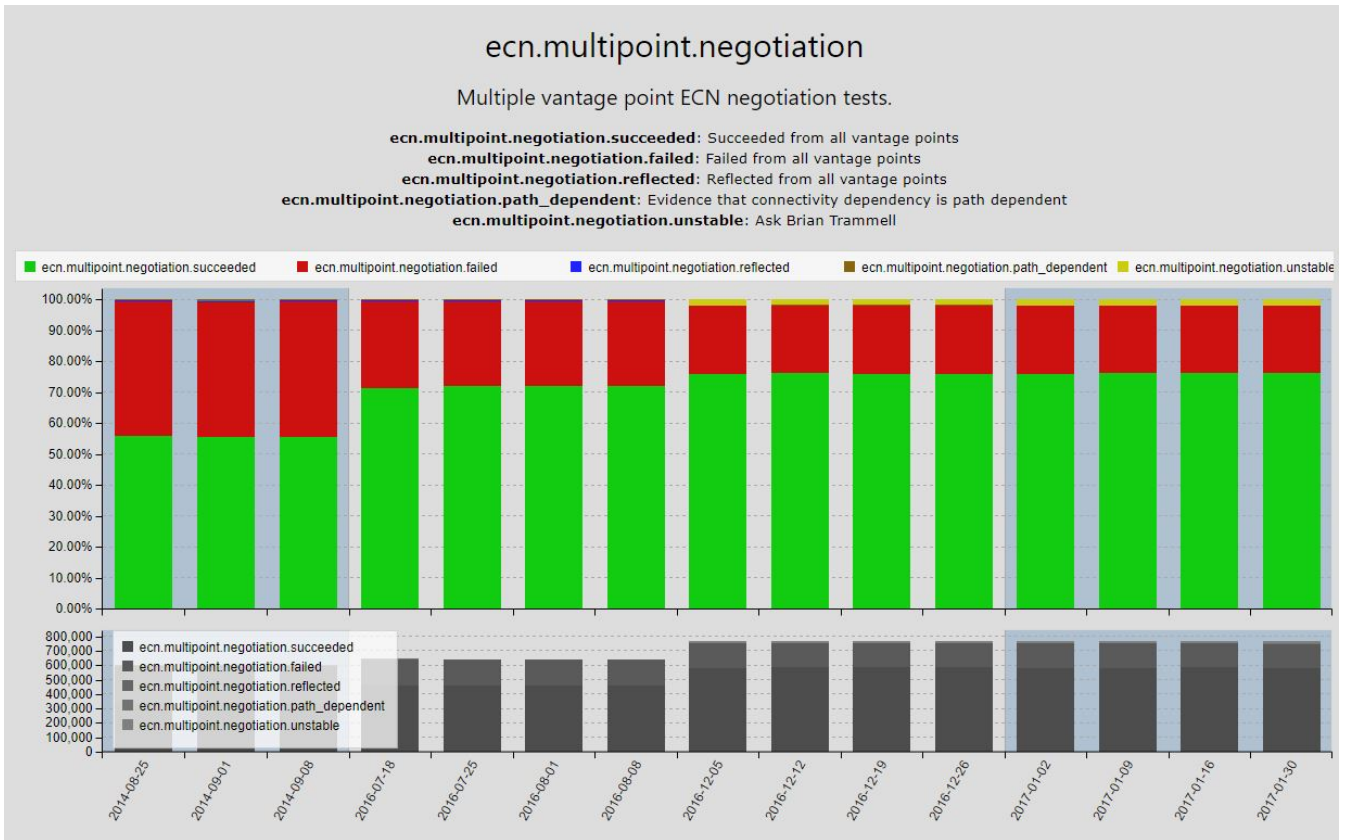


Figure A.6: Chart for ecn.multipoint.negotiation

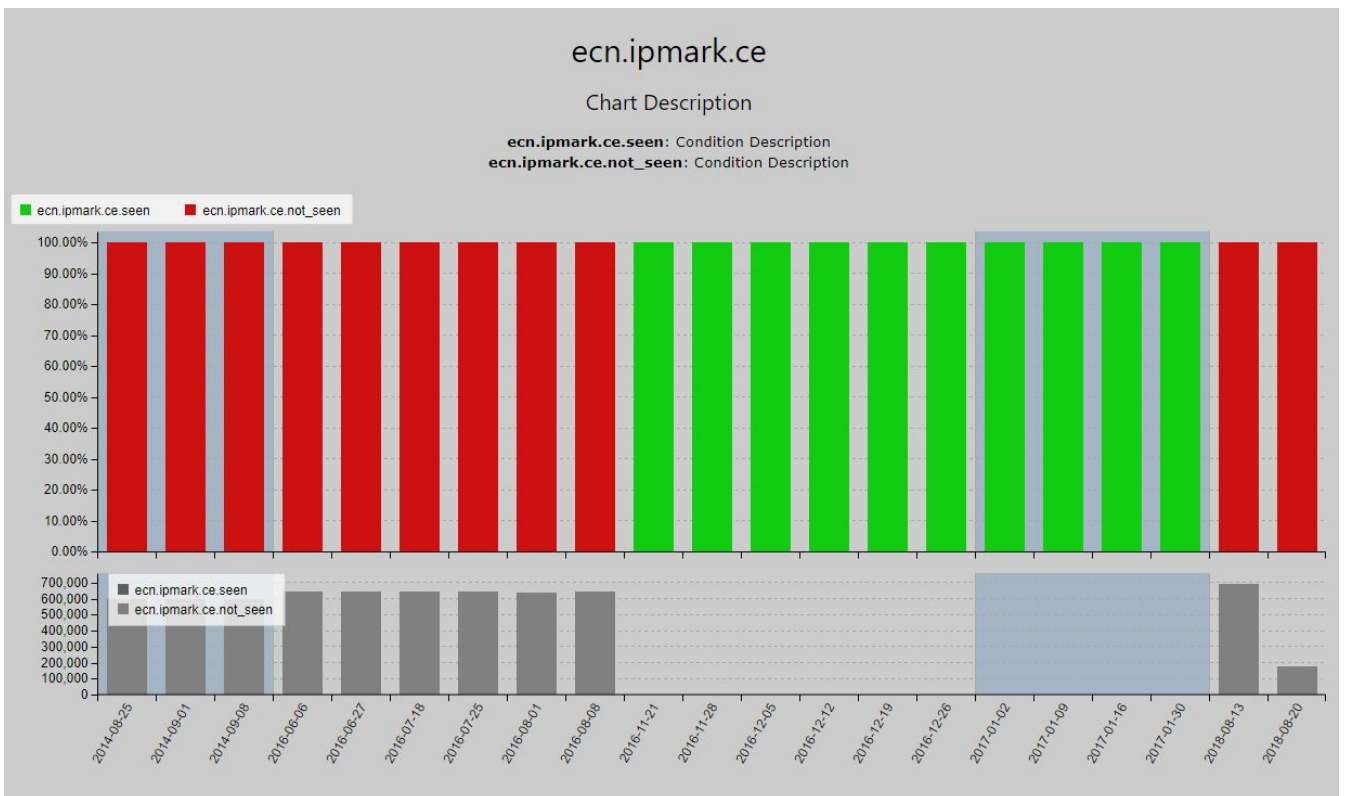


Figure A.7: Chart for ecn.ipmark.ce

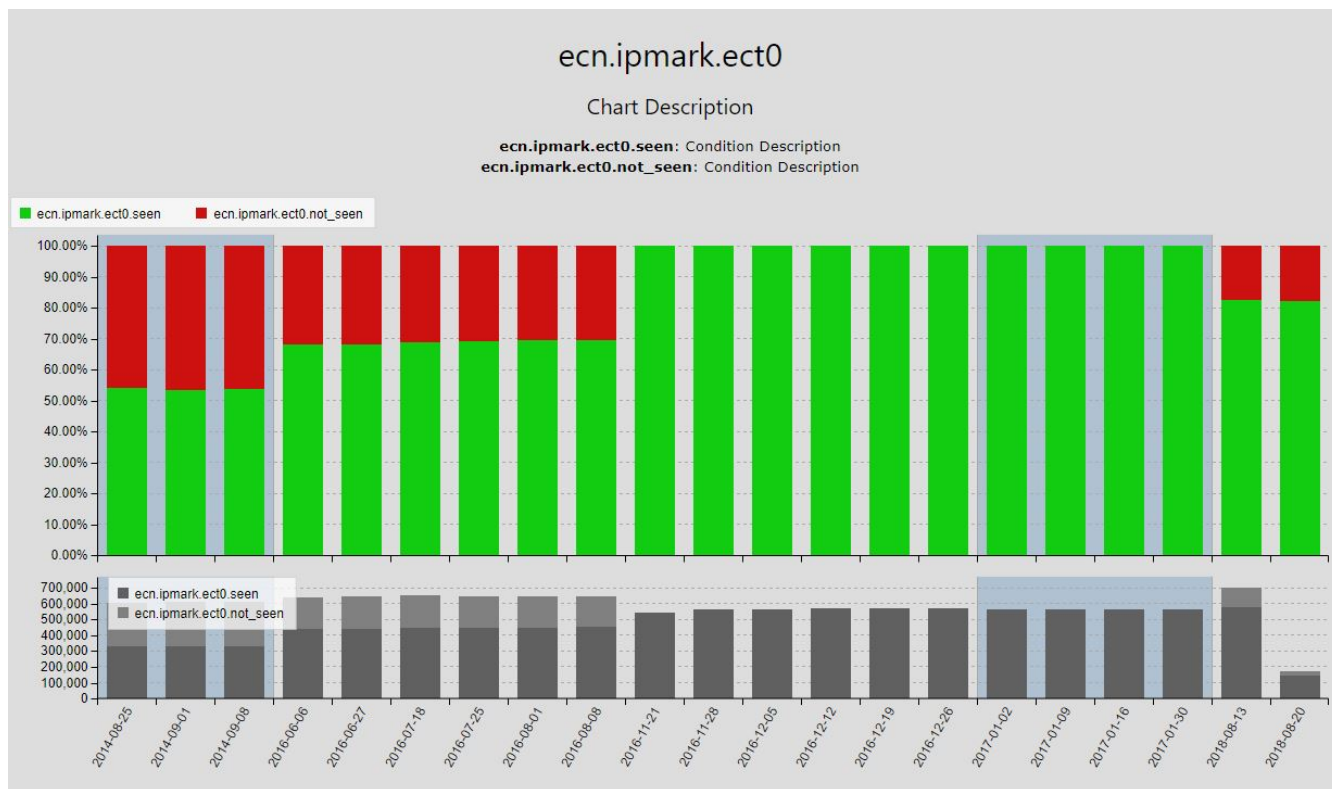


Figure A.8: Chart for ecn.ipmark.ect0

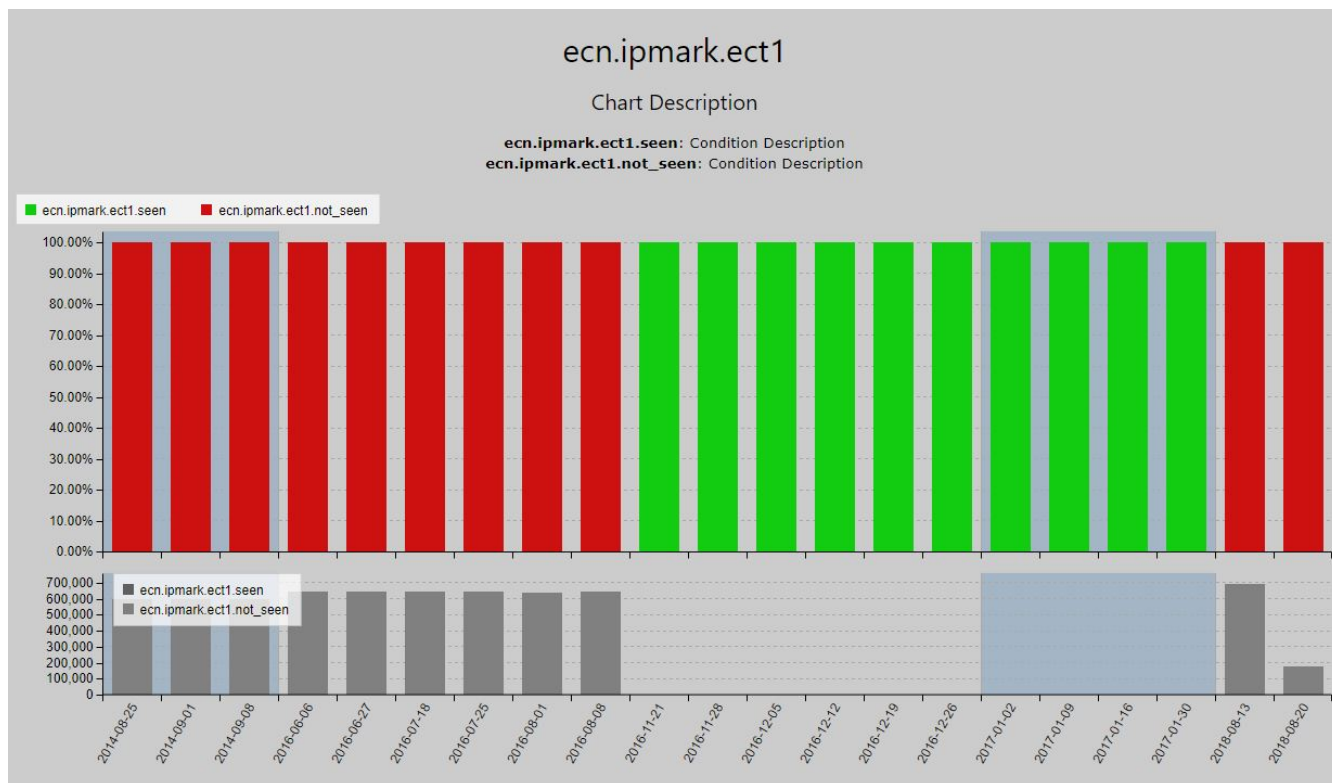


Figure A.9: Chart for ecn.ipmark.ect1

Appendix B

Admin Documentation

This chapter describes the configuration of the front end. It is directly taken from the Github project of the front end. [9]

B.1 Configuration

The file config.json is the main configuration file for the pto3-web.

directoryQuery

The URL of the query to be used for the directory matrix.

pages

Holds all chart pages that will be available on the website. Each property defines a page.

Key

If the page corresponds to a feature, it's key should be the id of the feature.

linktitle

The text of the link to be displayed in the navigation bar.

pageConfig

The file name of the page configuration file

B.2 Page Configuration

Each page defined in config.json is configured in a separate JSON file.

title

The title to be shown in the header of the feature page.

description

The description displayed underneath the title on the feature page.

charts

An array of chart configurations.

B.3 Chart Configuration**query**

The encoded query of which provides the data for the chart.

title

The title of the chart to be shown in the above the chart.

description

The description of the chart to be shown in the above the chart.

conditions

The conditions which will be shown in this chart. Usually these are all conditions of one aspect in the PTO.

colors

The color for each condition. The key must match a name of a condition.

descriptions

The descriptions for each condition. The key must match a name of a condition.

Appendix C

Current Configuration

At the time of writing, following configuration was applied to the PTO web front end.

C.1 Global Configuration

The global configurations can be found in *json/config.json*:

```
{
  "directoryQuery": "time_start=2014-01-01T00%3A00%3A00Z&time_end
    =2021-12-31T00%3A00%3A00Z&group=feature&group=month",
  "pages": {
    "ecn": {
      "linktitle": "ECN",
      "pageConfig": "ecn.json"
    },
    "mss": {
      "linktitle": "MSS",
      "pageConfig": "mss.json"
    },
    "tfo": {
      "linktitle": "TFO",
      "pageConfig": "tfo.json"
    }
  }
}
```

C.2 Page Configuration

The defined pages are configured in a separate file per page.

C.2.1 ECN

```
{
  "title": "Explicit Congestion Notification",
  "description": "Explicit Congestion Notification (ECN) is an
    extension to the Internet Protocol and to the Transmission
    Control Protocol. ECN allows end-to-end notification of network
    congestion without dropping packets. It is an optional feature
    that may be used between two ECN-enabled endpoints when the
    underlying network infrastructure also supports it.",
  "charts": [
    {
```

```

"query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
        =2020-12-31T23%3A59%3A59Z&aspect=ecn.connectivity&group=
        condition&group=week&option=count_targets",
"title": "ecn.connectivity",
"description": "In this test the influence of attempting to
        negotiate ECN on connectivity was measured.",
"conditions": [
    "ecn.connectivity.works",
    "ecn.connectivity.broken",
    "ecn.connectivity.transient",
    "ecn.connectivity.offline"
],
"colors": {
    "ecn.connectivity.works": "#11CC11",
    "ecn.connectivity.broken": "#CC1111",
    "ecn.connectivity.transient": "#2222FF",
    "ecn.connectivity.offline": "#888888"
},
"descriptions": {
    "ecn.connectivity.works": "Connectivity not dependent on
        attempt to negotiate ECN",
    "ecn.connectivity.broken": "Attempt to negotiate ECN leads to
        connectivity failure",
    "ecn.connectivity.transient": "Connectivity only with ECN
        negotiation; probable transient",
    "ecn.connectivity.offline": "Target offline regardless of ECN
        connectivity"
}
},
{
"query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
        =2020-12-31T23%3A59%3A59Z&aspect=ecn.stable.connectivity&
        group=condition&group=week&option=count_targets",
"title": "ecn.stable.connectivity",
"description": "Stabilized ECN connectivity tests. Multiple
        measurements from the same vantage point are performed to
        each target.",
"conditions": [
    "ecn.stable.connectivity.works",
    "ecn.stable.connectivity.broken",
    "ecn.stable.connectivity.transient",
    "ecn.stable.connectivity.unstable",
    "ecn.stable.connectivity.offline"
],
"colors": {
    "ecn.stable.connectivity.works": "#11CC11",
    "ecn.stable.connectivity.broken": "#CC1111",
    "ecn.stable.connectivity.transient": "#2222FF",
    "ecn.stable.connectivity.unstable": "#CCCC11",
    "ecn.stable.connectivity.offline": "#888888"
},
"descriptions": {
    "ecn.stable.connectivity.works": "Works on all attempts",
    "ecn.stable.connectivity.broken": "Broken on all attempts",
    "ecn.stable.connectivity.transient": "Transient on all
        attempts",
    "ecn.stable.connectivity.unstable": "No consensus about
        connectivity across multiple observations",

```

```

    "ecn.stable.connectivity.offline": "Offline on all attempts"
  },
  {
    "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
      =2020-12-31T23%3A59%3A59Z&aspect=ecn.multipoint.connectivity
      &group=condition&group=week&option=count_targets",
    "title": "ecn.multipoint.connectivity",
    "description": "Multiple vantage point ECN connectivity
      dependency tests.",
    "conditions": [
      "ecn.multipoint.connectivity.works",
      "ecn.multipoint.connectivity.broken",
      "ecn.multipoint.connectivity.transient",
      "ecn.multipoint.connectivity.path_dependent",
      "ecn.multipoint.connectivity.unstable",
      "ecn.multipoint.connectivity.offline"
    ],
    "colors": {
      "ecn.multipoint.connectivity.works": "#11CC11",
      "ecn.multipoint.connectivity.broken": "#CC1111",
      "ecn.multipoint.connectivity.transient": "#2222FF",
      "ecn.multipoint.connectivity.path_dependent": "#886611",
      "ecn.multipoint.connectivity.unstable": "#CCCC11",
      "ecn.multipoint.connectivity.offline": "#888888"
    },
    "descriptions": {
      "ecn.multipoint.connectivity.works": "Works from all vantage
        points",
      "ecn.multipoint.connectivity.broken": "Broken from all
        vantage points",
      "ecn.multipoint.connectivity.transient": "Transient from all
        vantage points",
      "ecn.multipoint.connectivity.path_dependent": "Evidence that
        connectivity dependency is path dependent",
      "ecn.multipoint.connectivity.unstable": "Ask Brian Trammell",
      "ecn.multipoint.connectivity.offline": "Offline from all
        vantage points"
    }
  },
  {
    "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
      =2020-12-31T23%3A59%3A59Z&aspect=ecn.negotiation&group=
      condition&group=week&option=count_targets",
    "title": "ecn.negotiation",
    "description": "In this tests we tried to negotiate ECN with
      the target.",
    "conditions": [
      "ecn.negotiation.succeeded",
      "ecn.negotiation.failed",
      "ecn.negotiation.reflected"
    ],
    "colors": {
      "ecn.negotiation.succeeded": "#11CC11",
      "ecn.negotiation.failed": "#CC1111",
      "ecn.negotiation.reflected": "#2222FF"
    },
    "descriptions": {

```

```

    "ecn.negotiation.succeeded": "ECN negotiated successfully",
    "ecn.negotiation.failed": "ECN not negotiated",
    "ecn.negotiation.reflected": "ECN negotiation reflects ECE
        CWR flags on SYN ACK"
    }
},
{
    "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
        =2020-12-31T23%3A59%3A59Z&aspect=ecn.stable.negotiation&
        group=condition&group=week&option=count_targets",
    "title": "ecn.stable.negotiation",
    "description": "Stabilized ECN negotiation tests. Multiple
        measurements from the same vantage point are performed to
        each target.",
    "conditions": [
        "ecn.stable.negotiation.succeeded",
        "ecn.stable.negotiation.failed",
        "ecn.stable.negotiation.unstable"
    ],
    "colors": {
        "ecn.stable.negotiation.succeeded": "#11CC11",
        "ecn.stable.negotiation.failed": "#CC1111",
        "ecn.stable.negotiation.unstable": "#CCCC11"
    },
    "descriptions": {
        "ecn.stable.negotiation.succeeded": "Succeeded on all attempts",
        "ecn.stable.negotiation.failed": "Failed on all attempts",
        "ecn.stable.negotiation.unstable": "No consensus about
            negotiation across multiple observations"
    }
},
{
    "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
        =2020-12-31T23%3A59%3A59Z&aspect=ecn.multipoint.negotiation&
        group=condition&group=week&option=count_targets",
    "title": "ecn.multipoint.negotiation",
    "description": "Multiple vantage point ECN negotiation tests.",
    "conditions": [
        "ecn.multipoint.negotiation.succeeded",
        "ecn.multipoint.negotiation.failed",
        "ecn.multipoint.negotiation.reflected",
        "ecn.multipoint.negotiation.path_dependent",
        "ecn.multipoint.negotiation.unstable"
    ],
    "colors": {
        "ecn.multipoint.negotiation.succeeded": "#11CC11",
        "ecn.multipoint.negotiation.failed": "#CC1111",
        "ecn.multipoint.negotiation.reflected": "#2222FF",
        "ecn.multipoint.negotiation.path_dependent": "#886611",
        "ecn.multipoint.negotiation.unstable": "#CCCC11"
    },
    "descriptions": {
        "ecn.multipoint.negotiation.succeeded": "Succeeded from all
            vantage points",
        "ecn.multipoint.negotiation.failed": "Failed from all vantage
            points",
    }
}

```



```

    "ecn.multipoint.negotiation.reflected": "Reflected from all
      vantage points",
    "ecn.multipoint.negotiation.path_dependent": "Evidence that
      connectivity dependency is path dependent",
    "ecn.multipoint.negotiation.unstable": "Ask Brian Trammell"
  }
},
{
  "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
    =2020-12-31T23%3A59%3A59Z&aspect=ecn.ipmark.ce&group=
    condition&group=week&option=count_targets",
  "title": "ecn.ipmark.ce",
  "description": "Chart Description",
  "conditions": [
    "ecn.ipmark.ce.seen",
    "ecn.ipmark.ce.not_seen"
  ],
  "colors": {
    "ecn.ipmark.ce.seen": "#11CC11",
    "ecn.ipmark.ce.not_seen": "#CC1111"
  },
  "descriptions": {
    "ecn.ipmark.ce.seen": "Condition Description",
    "ecn.ipmark.ce.not_seen": "Condition Description"
  }
},
{
  "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
    =2020-12-31T23%3A59%3A59Z&aspect=ecn.ipmark.ect0&group=
    condition&group=week&option=count_targets",
  "title": "ecn.ipmark.ect0",
  "description": "Chart Description",
  "conditions": [
    "ecn.ipmark.ect0.seen",
    "ecn.ipmark.ect0.not_seen"
  ],
  "colors": {
    "ecn.ipmark.ect0.seen": "#11CC11",
    "ecn.ipmark.ect0.not_seen": "#CC1111"
  },
  "descriptions": {
    "ecn.ipmark.ect0.seen": "Condition Description",
    "ecn.ipmark.ect0.not_seen": "Condition Description"
  }
},
{
  "query": "time_start=2014-01-01T00%3A00%3A00Z&time_end
    =2020-12-31T23%3A59%3A59Z&aspect=ecn.ipmark.ect1&group=
    condition&group=week&option=count_targets",
  "title": "ecn.ipmark.ect1",
  "description": "Chart Description",
  "conditions": [
    "ecn.ipmark.ect1.seen",
    "ecn.ipmark.ect1.not_seen"
  ],
  "colors": {
    "ecn.ipmark.ect1.seen": "#11CC11",
    "ecn.ipmark.ect1.not_seen": "#CC1111"
  }
}

```

```
    },  
    "descriptions": {  
      "ecn.ipmark.ect1.seen": "Condition Description",  
      "ecn.ipmark.ect1.not_seen": "Condition Description"  
    }  
  }  
] }  
}
```

Bibliography

- [1] S. Neuhaus B. Trammell, I. Learmonth. Measurement and architecture for a middleboxed internet. Deliverable, ETH, UNIABDN, ZHAW, June 2018.
- [2] M. Bostock. Data-driven documents. <https://d3js.org/>. Accessed: 2018-09-24.
- [3] MAMI Consortium. Measurement and architecture for a middleboxed internet. <https://mami-project.eu/>. Accessed: 2018-09-18.
- [4] MAMI Consortium. Pathspider. <https://pathspider.net/>. Accessed: 2018-09-24.
- [5] Refsnes Data. W3schools. <https://c3js.org/>. Accessed: 2018-09-24.
- [6] B. Trammell M. Kühlewind. Representation of internet path transparency. <https://www.tik.ee.ethz.ch/file/f5420cac7bb1c8051ae5f16425cf2fea/PTO.pdf>. Accessed: 2018-09-24.
- [7] Mozilla. Cross-origin resource sharing. <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>. Accessed: 2018-09-20.
- [8] RIPE NCC. Ripestat. <https://c3js.org/>. Accessed: 2018-09-24.
- [9] R. Scheuner. pto3-web. <https://github.com/mami-project/pto3-web/tree/connectPTOv3>. Accessed: 2018-09-24.
- [10] M. Tanaka. C3js. <https://c3js.org/>. Accessed: 2018-09-24.
- [11] O. Tilmans. Tracebox. <http://www.tracebox.org/>. Accessed: 2018-09-24.
- [12] B. Trammell. Path transparency observatory api specification. <https://github.com/mami-project/pto3-go/blob/master/doc/API.md>. Accessed: 2018-09-18.