Censored Planet Observatory

Release 2.0

Ram Sundara Raman, Elisa Tsai, Apurva Virkud, Armin Huremagi

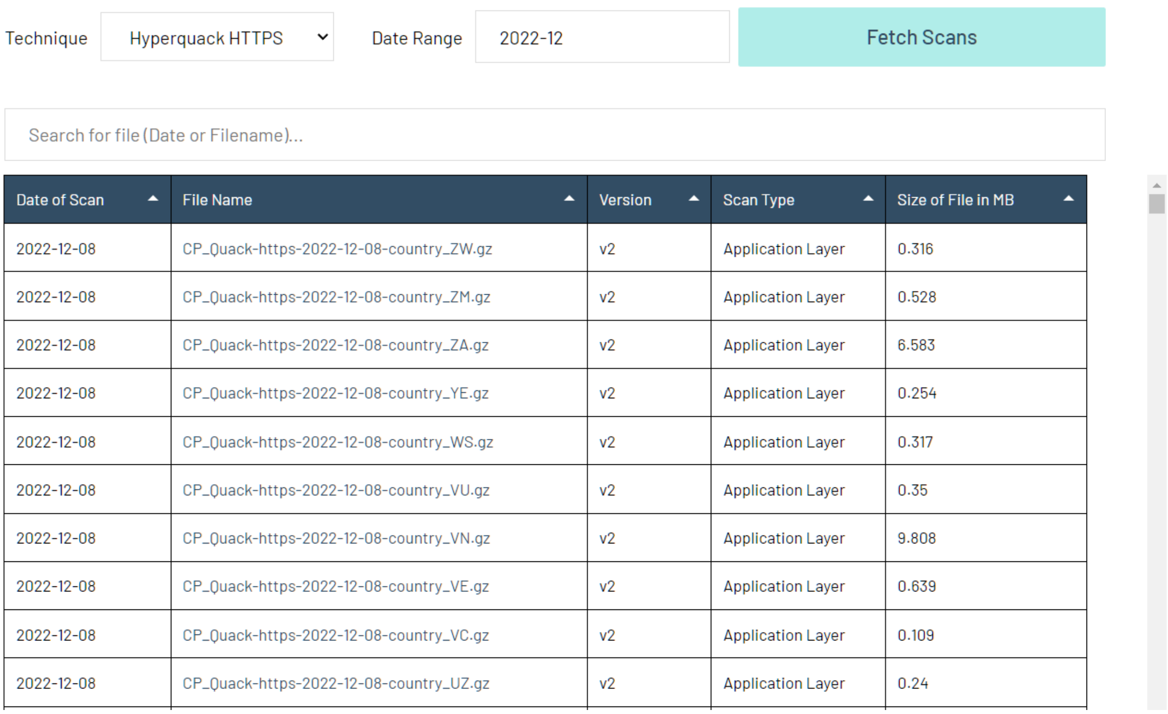
Feb 10, 2023

CONTENTS

Censored Planet is a longitudinal censorship measurement platform that collects remote measurement measurements in more than 200 countries. Censored Planet was launched in August 2018, and has since then collected more than 60 billion measurement data points. Censored Planet measures network interference on the TCP/IP, DNS, and HTTP(S) protocols, using remote measurement techniques Augur, Satellite, and Hyperquack respectively. Every week, Censored Planet collects reachability data about 2000 popular and sensitive websites from more than 95,000 vantage points around the world. An academic paper about Censored Planet can be found here.

Censored Planet's measurement data has been crucial in identifying and monitoring several important censorship and network interference events. In 2019, Censored Planet data was used to study the large-scale HTTPS interception that occurred in Kazakhstan, and was instrumental in driving changes in major web browsers that blocked the interception attack. Censored Planet data has been used to study Russia's decentralized censorship mechanism, and the throttling attack they performed on Twitter. Censored Planet has also been used to identify the deployment of network censorship devices, and track the blocking of COVID-19 related websites around the world.

Censored Planet data is available to the public through the Censored Planet website: data.censoredplanet.org/raw. The Censored Planet raw data website contains archived compressed data files. Each filename contains corresponding measurement technique name and the ISO Alpha-2 country code.



Date of Scan	File Name	Version	Scan Type	Size of File in MB
2022-12-08	CP_Quack-https-2022-12-08-country_ZW.gz	v2	Application Layer	0.316
2022-12-08	CP_Quack-https-2022-12-08-country_ZM.gz	v2	Application Layer	0.528
2022-12-08	CP_Quack-https-2022-12-08-country_ZA.gz	v2	Application Layer	6.583
2022-12-08	CP_Quack-https-2022-12-08-country_YE.gz	v2	Application Layer	0.254
2022-12-08	CP_Quack-https-2022-12-08-country_WS.gz	v2	Application Layer	0.317
2022-12-08	CP_Quack-https-2022-12-08-country_VU.gz	v2	Application Layer	0.35
2022-12-08	CP_Quack-https-2022-12-08-country_VN.gz	v2	Application Layer	9.808
2022-12-08	CP_Quack-https-2022-12-08-country_VE.gz	v2	Application Layer	0.639
2022-12-08	CP_Quack-https-2022-12-08-country_VC.gz	v2	Application Layer	0.109
2022-12-08	CP_Quack-https-2022-12-08-country_UZ.gz	v2	Application Layer	0.24

Figure - Raw data files on the Censored Planet website

Each scan contains a test of all 2,000 websites tested by Censored Planet with a set of vantage points across different countries. The files can be downloaded, extracted and analyzed. The data formats change based on the version of the measurement technique. The data formats and tips for analyzing the data for each of the published data files and versions are available below. For more information about using the data, please refer to the Censored Planet Analysis Github page, or email Censored Planet at censoredplanet@umich.edu.

CHAPTER

ONE

CONTENTS

1.1 Censored Planet Version Timeline

The timelines of the data generated by different versions of Censored Planet's remote measurement techniques are shown below. Refer to documentation about each of the techniques to get information about the data format for each version.

Satellite		Hyperquack	
v1	>= 2018-08-03 && <= 2021-02-28	v1	>= 2018-07-27 && <= 2021-04-22
v2.0	>= 2021-03-15 && <= 2021-04-14	v2.0	>= 2021-04-25 && <= 2021-05-30
v2.1	>= 2021-04-14 && <= 2021-06-07	v2.1	>= 2021-06-01 && <= 2021-07-21
v2.2	>= 2021-06-07	v2.2	>= 2021-07-21

1.2 DNS Data - Satellite

Satellite/Iris is Censored Planet's remote measurement technique that detects DNS interference using Open DNS resolvers. Below, we provide an overview of Satellite and its data format. Refer to our academic papers for in-depth details about Satellite.

1.2.1 Satellite-v2.2-raw

To provide raw data for easy data analysis, we made the following changes:

1. Split data based on the country of resolvers so that it is easier to select and download data according to users' country of interest.
2. Separated the data collection phase and data analysis phase. Right now the Satellite data from our raw measurement data website is truthful to the data collected without further analysis. We deprecated the "anomaly" field since there are misunderstandings that anomaly represents censorship.
3. Added new data containing further metadata fields and flattened nested data for easy analysis. Modified field names for disambiguation purposes.
    - `domain` [String] The test domain being queried.
    - `domain_is_control` [Boolean] Equals true if the queried domain is the root server for liveness test.
    - `test_url` [String] The URL of the queried domain.
    - `date` [String] The date of the measurement.
    - `start_time` [String] The start time of the measurement.

- **end_time** [String] The end time of the measurement.
- **resolver_ip** [String] The IP address of the vantage point (a DNS resolver).
- **resolver_name** [String] The hostname of the vantage point.
- **resolver_is_trusted** [Boolean] Equals true if the resolver is a control resolver.
- **resolver_netblock** [String] The netblock the vantage point belongs to.
- **resolver_asn** [String] The AS number of the AS the vantage point resides in.
- **resolver_as_name** [String] The name of the AS the vantage point resides in.
- **resolver_as_full_name** [String] The full name of the AS the vantage point resides in.
- **resolver_as_class** [String] The class of the AS the vantage point resides in.
- **resolver_country** [String] The country the vantage point resides in.
- **resolver_organization** [String] The IP organization the vantage point resides in.
- **received_error** [String] Flatten error messages from the received responses.
- **received_rcode** [Integer] Flatten rcode from the received responses. Response code mapping to success (0) or errors (-1 for connection error, > 0 for errors specified in RFC 2929).
- **source** [String] Tar file name of the measurement.
- **answers** [JSON object] The resolver's returned answers for queried domain.
  - **ip: String** Returned IP.
  - **asn: String** The AS number of the AS the returned IP resides in.
  - **as_name: String** The AS name of the AS the returned IP resides in.
  - **censys_http_body_hash: String** The hash of the HTTP body from Censys.
  - **censys_ip_cert: String** The hash of the TLS certificate from Censys.
  - **http_error: String** Parsed HTTP page error message from `fetch` module.
  - **http_response_status: String** Parsed HTTP page status code from `fetch` module.
  - **http_response_headers: String** Parsed HTTP page headers from `fetch` module.
  - **http_response_body: String** Parsed HTTP page body from `fetch` module.
  - **https_error: String** Parsed HTTPS page error message from `fetch` module.
  - **https_response_status: String** Parsed HTTPS page status code from `fetch` module.
  - **https_response_headers: String** Parsed HTTPS page headers from `fetch` module.
  - **https_response_body: String** Parsed HTTPS page body from `fetch` module.
  - **https_tls_version: String** Parsed TLS version from `fetch` module.
  - **https_tls_cipher_suite: String** Parsed TLS cipher suite from `fetch` module.
  - **https_tls_cert: String** Parsed TLS certificate from `fetch` module.
  - **https_tls_cert_common_name: String** Parsed common name field from TLS certificate.
  - **https_tls_cert_alternative_names: String** Parsed alternative name field from TLS certificate.
  - **https_tls_cert_issuer: String** Parsed issuer field from TLS certificate.
  - **https_tls_cert_start_date: String** Parsed start date of the TLS certificate.

– **https_tls_cert_end_date: String** Parsed end date of the TLS certificate.

1.2.2 Satellite-v1 (deprecated)

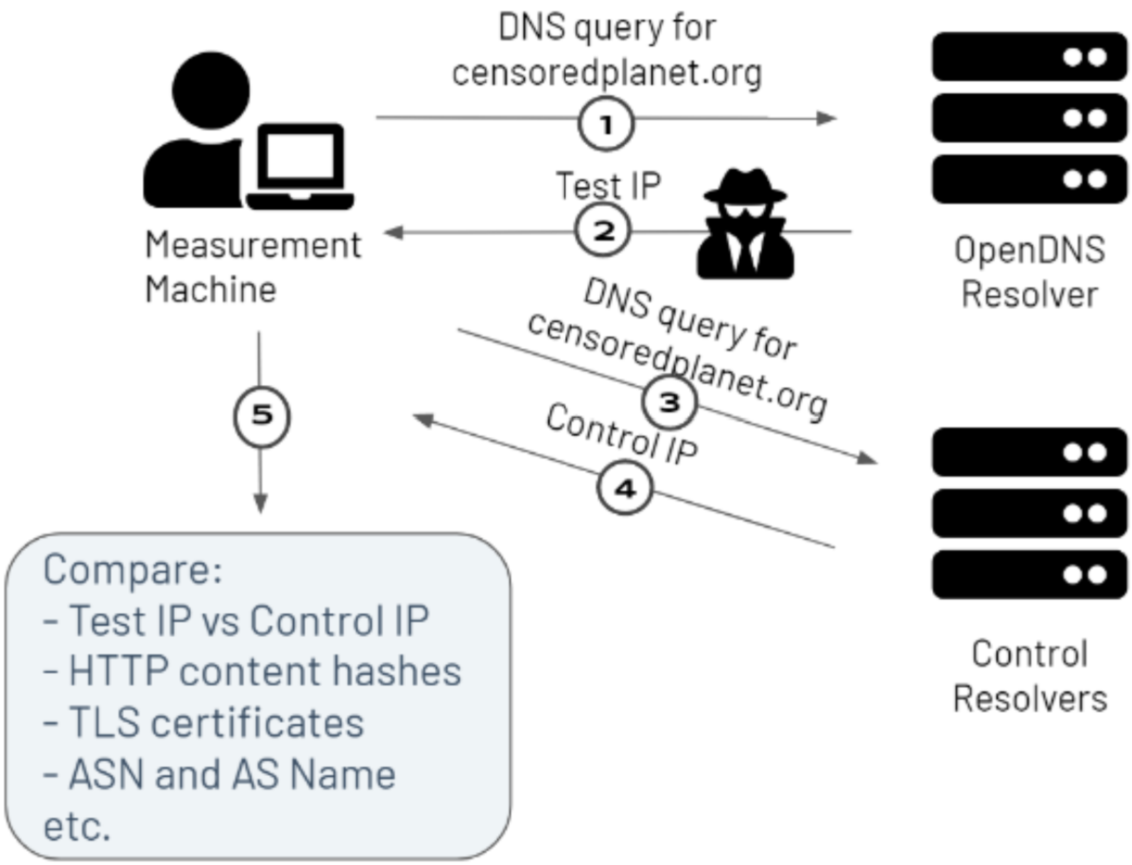


Figure - Overview of Satellite-v1

Satellite-v1 is the first version of Satellite that we operated from August 2018 - February 2021. The primary function of Satellite is to detect incorrect DNS resolutions from open DNS resolvers in many countries.

- From a measurement machine at the University of Michigan, we send a DNS query for a website whose reachability we're interested in, to an open DNS resolver in a country of interest (1). The response from the DNS resolver is our Test IP (2).
- We also send a DNS query for the same website to trusted control resolvers (3), and record their response as the control IP (4).
- We then compare the test and control responses using several heuristics, including a direct IP address comparison, and comparison of the AS number, AS names, HTTP content hashes, and TLS certificates associated with the test and control IP addresses (5). Satellite-v1 only labels a measurement as an anomaly when all of the heuristics mismatch.

Our various publications and reports have used Satellite-v1 to detect many cases of DNS manipulation. For instance, in our recent investigation into the filtering of COVID-19 websites , Satellite-v1 found many networks using website filtering products to manipulate DNS responses of COVID-related websites.

1.2.2.1 Limitations

Although Satellite-v1 was extremely useful in detecting DNS interference at large scale, it suffered from several limitations, which form the improvements in Satellite-v2.x.

- Satellite-v1 could not detect DNS censorship where A records were not available i.e. Satellite-v1 primarily focused on detecting incorrect DNS resolutions through the resolved IP address, and did not contain heuristics to measure DNS manipulation which manifested through timeouts, NXDOMAIN responses, SERVFAIL responses, etc.
- Satellite-v1 required post-processing to remove false positives and confirm the presence of anomalies, such as through using post-measurement heuristics and blockpage regexes. Satellite-v2 has the inbuilt capability to perform most post-processing measurements.

1.2.3 Satellite-v2 (deprecated)

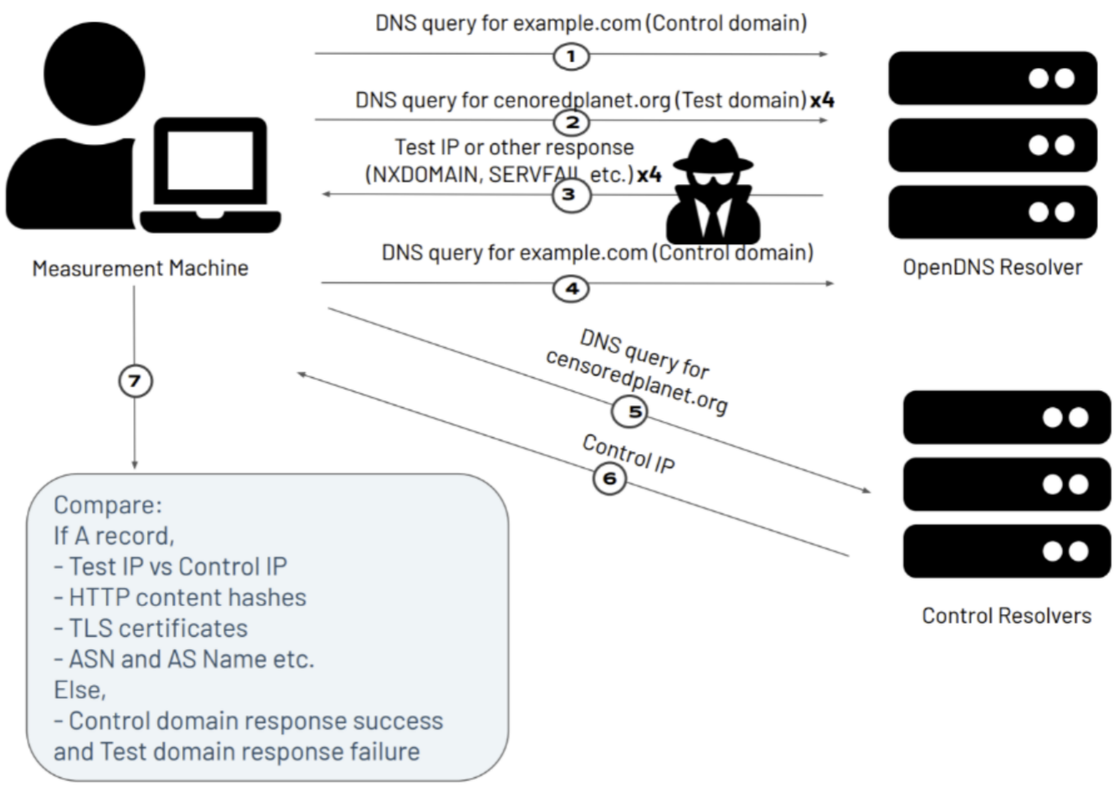


Figure - Overview of Satellite-v2

Satellite-v2 is our brand new version of Satellite, where we've made several modifications to the measurement technique and data format for facilitating accurate and efficient remote DNS interference measurements. Below, we detail the major changes we've made in Satellite-v2.

- **Fetching HTML pages hosted at resolved IPs marked as an anomaly** - Satellite-v2 has an in-built fetch feature that performs HTTP and HTTPS GET requests to resolved IPs that fail our heuristics. This step was being performed as a post-processing step in Satellite-v1. This addition helps in quickly identifying blockpages such as the example shown in the figure below. Moreover, we are in the process of developing a technique to use TLS certificates to detect DNS manipulation. Reach out to *censoredplanet@umich.edu* for more information.

- **Measuring DNS interference without A records** - In Satellite-v2, we have added a sandwiched retry mechanism to our Satellite measurements in order to detect DNS interference that results in a non-zero R code response. A description of the method is shown in the figure below. We first make a control query to the open DNS resolver, providing a domain name that we do not expect to be blocked (eg. www.example.com). After the control query, we make up to 4 retries of the test DNS query, providing the test domain name. In case an A record is detected, we stop the test measurement. At the end, we perform another control query similar to the first measurement. The control queries ensure that the resolver is behaving correctly for an innocuous domain, and the multiple retry mechanism accounts for temporary errors in the network. With the help of the sandwiched retry mechanism, Satellite-v2 is able to detect DNS interference that manifests as timeouts, NXDOMAIN, SERVFAIL etc. From our preliminary analysis of Satellite-v2 data, we've already found several cases of DNS interference that can be identified using this method. For example, from the Satellite-v2 scan performed on 2021-03-17, we are able to identify 174,795 responses that have non-zero R codes from China, which makes up 15.6% out of the responses marked as interference. This kind of DNS interference was previously omitted by satellite v1. Shown below is an example measurement that passed the sandwich control tests, but received server failure R code. This could be an indicator of censorship or geoblocking.
- **Adding scan-level heuristics to exclude false positives** - Another step part of the post-processing pipeline of Satellite-v1 that is inbuilt in Satellite-v2. We exclude potentially false positive anomalies by using scan-level heuristics, such as the number of domains resolving to the anomalous IP address, or the anomalous IP address being part of a big CDN. Note that this step may lead to Satellite-v2 missing certain censorship.
- **Other changes** - We updated the heuristics to determine whether a DNS response is interfered - Satellite-v2 now includes a new "confidence" field, which addresses the certainty of interference according to the state of comparison between responses from the test resolvers and the control resolvers. We also make sure that IPs with no metadata information from Censys are not marked as interference.

1.2.4 Satellite-v2.1

Satellite-v2.1 incorporates minor changes from Satellite-v2.0, starting after April 14, 2021. Most of these changes are related to change in data formats.

1.2.5 Satellite-v2.2

Satellite-v2.2 incorporates major changes in code and data structure from Satellite-v2.1, but no major changes in the functionality of Satellite. The changes are made after June 7, 2021 and they include,

- Store information generated from the query, tag, detect, and verify module in memory, producing only one file (results.json) as output, instead of generating outputs for every module. Renamed query-tag-detect-verify as "test" module, and probe-filter as "discovery".
- Updated test module so that it first conducts queries for control resolvers, and then query, tag and detect test resolvers in batches.

Satellite v2 is divided into three parts:

1. `discovery`: consist of `probe` and `filter` modules.
2. `test`: consist of `query`, `tag` and `detect` modules.
3. verification and blockpage fetching: consist of `fetch` and `verify`.

1. Generate a DNS A query packet for a controlled domain (`dns.pkt`).
2. Perform a ZMap (Internet-wide) scan with the probe packet for open DNS resolvers.

    `resolvers_raw.json` contains the ZMap output:

    - **`saddr`** [String] IP address of a DNS resolver.

- **data** [String] Raw response to probe domain.

1. Perform PTR queries on the IPs of resolvers found by ZMap and filter out the ones without PTR records.
2. Perform Liveness test on the infrastructural resolvers and filter out the ones that fail.
3. Add predefined "control" and "special" resolvers to form the final set of vantage points.
4. Tag each resolver with the location from Maxmind.

   `resolvers.json` contains the infrastructure, "control", and "special" resolvers.

   - **vp** [String] The IP address of the vantage point (a DNS resolver).
   - **name** [String] Result from PTR query (if infrastructure), "control", or "special".
   - **location: JSON object**
     - **country_name** [String] The full name of the country where the resolver is located.
     - **country_code** [String] The two-letter ISO 3166 code of the country where the resolver is located.

1. Make DNS queries for each test domain to each resolver. The query for the test domain is attempted up to four times in case of connection error. To check the status of the resolver, a control measurement is conducted before the queries for the test domain. If the first control measurement fails, no further measurements will be conducted for the same `(resolver, domain)` pair. If all 4 trials for the test domain fail, another control measurement will be conducted.
2. Parse and separate responses from control resolvers and non-control resolvers.

1. **Tag each answer IP with information from Censys. Note:**

   - Fields may have empty strings if the information was not available on Censys.

1. Compare query responses between non-control resolvers and control resolvers to identify interference. When running satellite v2 as a whole module, `detect` does not output any files. However, when run separately, `detect` outputs `results.json` with the `excluded` field set to `false` and the `excluded_reason` field set to `null` by default. (See the output structure in `verify` section)

   Note:

   - For each response, the answer IPs and their tags are compared to the set of answer IPs and tags from all the control resolvers for the same query domain. A response is classified as an anomaly if there is no overlap between the two.

1. Perform HTTP(S) GET requests to the IPs identified as anomalies.

   `blockpages.json` contains the responses:

   - **ip** [String] The IP address from an anomalous DNS response.
   - **keyword** [String] The domain queried for the anomalous DNS response.
   - **http** [Object] HTTP response.
   - **https** [Object] HTTPS response.
   - **fetched** [Boolean] Equals true if a page is successfully fetched.
   - **start_time** [String] The start time of the measurement.
   - **end_time** [String] The end time of the measurement.

1. **New heuristics to exclude possible cases of erroneous answers from resolvers. Currently, `verify` excludes answer IPs that**

   `results.json` contains all the information when running `full` mode.

   - **vp** [String] The IP address of the vantage point (a DNS resolver).

- **test_url** [String] The test domain being queried.
- **location: JSON object**
  - **country_name** [String] The full name of the country where the resolver is located.
  - **country_code** [String] The two-letter ISO 3166 code of the country where the resolver is located.
- **passed_liveness** [Boolean] Equals `false` if both control queries were unsuccessful.
- **in_control_group** [Boolean] Equals true if at least one control resolver had a valid response for this test domain.
- **connect_error** [Boolean] Equals true if all test domain query attempts returned errors. This field is also set to be `true` if the first control measurement fails, and no further measurements for the test domain are conducted. Use this field in conjunction with the `passed_liveness` field to find anomalies.
- **anomaly** [Boolean] Equals true if an anomaly is detected. In case there are no tags for the answers or control, then this field is conservatively marked as false.
- **start_time** [String] The start time of the measurement.
- **end_time** [String] The end time of the measurement.
- `response` : JSON object

  The resolver's returned answers for the queried domain are the keys.

  - **url: String** The domain being queried in this trial, either the control domain for liveness test or `test_url`. The liveness test DNS responses are only recorded if they do not contain a type-A RR.
  - **has_type_a: Boolean** Equals `true` if the query returned a valid A resource record.
  - **error: String** Contains error information.
  - **rcode: Integer** Response code mapping to success (0) or errors (-1 for connection error, > 0 for errors specified in RFC 2929).
  - **response: JSON Object** Consist of a map between IPs the resolver responded for the queried domain and tags from Maxmind:
    * **http** [String] The hash of the HTTP body.
    * **cert** [String] The hash of the TLS certificate.
    * **asname** [String] The autonomous system (AS) name.
    * **asnum** [Integer] The autonomous system (AS) number.
    * **matched** [Array] An array of its tags that matched the control tags - if the IP is in the control set, "ip" is appended and if the IP has no tags, "no_tags" is appended.

1.2.6 Notes

While Satellite includes multiple control resolvers intended to avoid false inferences there is still a possibility that certain measurements are marked as anomalies incorrectly. To confirm censorship, it is critical that the raw DNS responses are compared to known blockpage fingerprints. The blockpage fingerprints currently recorded by Censored Planet are available here. Moreover, aggregations can be used to avoid anomalous vantage points and domains. Please use our analysis pipeline to process the data before using it.

Censored Planet detects network interference of websites using remote measurements to infrastructural vantage points within networks (eg. institutions). Note that this raw data cannot determine the entity responsible for the blocking or the intent behind it. Please exercise caution when using the data, and reach out to us at *censoredplanet@umich.edu* if you have any questions.

1.3 HTTP(S) Data - Hyperquack

Hyperquack (and Quack) is Censored Planet's measurement techniques that measure application-layer interference using the Echo, Discard, HTTP, and HTTPS protocols. Below, we provide a detailed overview of Hyperquack, and the data formats of Hyperquack data published on the Censored Planet website. Refer to our academic papers for more information about Quack and Hyperquack.

1.3.1 Hyperquack-v2-raw

To provide raw data for easy data analysis, we made the following changes:

1. Split data based on the country of vantage points so that it is easier to select and download data according to users' country of interest.
2. Separated the data collection phase and data analysis phase. Right now the Hyperquack data from our raw measurement data website is truthful to the data collected without further analysis. We deprecated the "anomaly" field since there are misunderstandings that anomaly represents censorship.
3. Added new data containing further metadata fields and flattened nested data for easy analysis. Modified field names for disambiguation purposes.
    - **`domain`** [String] The test domain being queried.
    - **`domain_is_control`** [Boolean] Equals true if the queried domain is for the liveness test.
    - **`date`** [String] The date of the measurement.
    - **`start_time`** [String] The start time of the measurement.
    - **`end_time`** [String] The end time of the measurement.
    - **`server_ip`** [String] The IP address of the vantage point.
    - **`server_netblock`** [String] The netblock the vantage point belongs to.
    - **`server_asn`** [String] The AS number of the AS the vantage point resides in.
    - **`server__as_name`** [String] The name of the AS the vantage point resides in.
    - **`server__as_full_name`** [String] The full name of the AS the vantage point resides in.
    - **`server__as_class`** [String] The class of the AS the vantage point resides in.
    - **`server_country`** [String] The country the vantage point resides in.
    - **`server_organization`** [String] The IP organization the vantage point resides in.

- **source** [String] Tar file name of the measurement.
- **received_error** [String] Flatten error messages from the received responses.
- **received_status: String** Flatten status code from the received responses.
- **received_headers: String** Parsed HTTPS page headers.
- **received_body: String** Parsed HTTPS page body.
- **received_tls_version: String** Parsed TLS version.
- **received_tls_cipher_suite: String** Parsed TLS cipher suite.
- **received_tls_cert: String** Parsed TLS certificate.
- **received_tls_cert_common_name: String** Parsed common name field from TLS certificate.
- **received_tls_cert_alternative_names: String** Parsed alternative name field from TLS certificate.
- **received_tls_cert_issuer: String** Parsed issuer field from TLS certificate.
- **matches_template: Boolean** Equals true if the response given by the vantage point matches the known template.
- **no_response_in_measurement_matches_template: Boolean** Equals true if the responses from all the trials failed to match the known template.
- **controls_failed: Boolean** Set to `true` when all control probes sent to the vantage point fail to match the known template. This implies that the mismatching responses are due to an error in the vantage point or the network, not censorship. Rows with `controls_failed` set to `true` should not be considered for analysis.
- **stateful_block** [Boolean] Equals true if another control probe sent immediately after our sensitive probes is blocked, but the second control measurement sent after 2 minutes was not.

1.3.2 Hyperquack-v1 (deprecated)

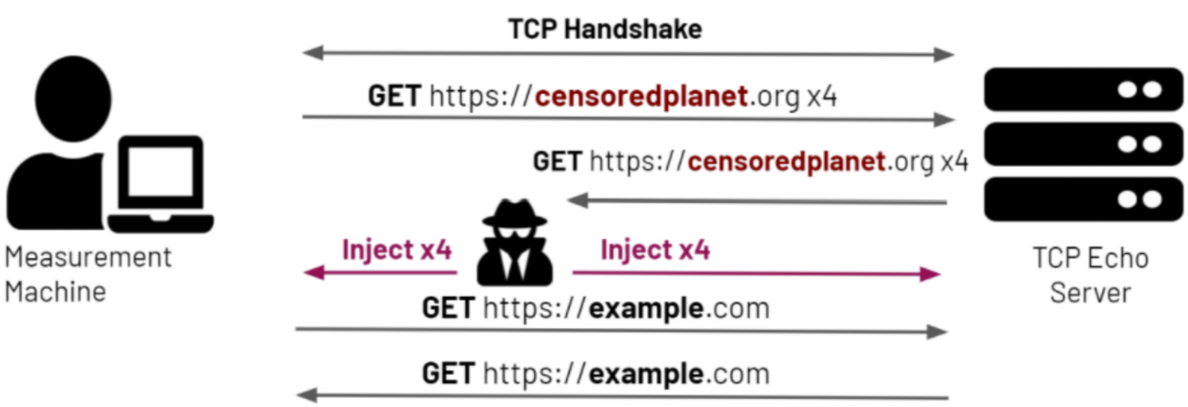


Figure - Overview of Hyperquack-

Quack-v1 and Hyperquack-v1 were operated from August 2018 to April 2021. Quack-v1 detects application-layer interference using the Echo and Discard protocols. Quack-v1's workflow is pictured in Figure 1.

- From a remote measurement machine, we send an HTTP get look-alike request containing a non-sensitive control URL to a vantage point's Echo or Discard port. Vantage points are selected from infrastructural servers such as ISP routers to minimize risk to their owners. We observe the result, and if the port is responding incorrectly according to its protocol, we abort the test, mark the vantage point as broken, and remove the vantage point from our test list.

- If the control test succeeds, we then send an HTTP get look-alike request containing a potentially sensitive URL to the vantage point. If the vantage point responds correctly, we record that there is not an anomaly. If the vantage point responds incorrectly, we repeat the request up to four more times. If any such request results in a correct response, we again record that there is not an anomaly.
- If all five requests result in incorrect responses, we then send another request containing a control keyword. If this request results in a correct response, we record the possibility of interference.
- If this control request results in an incorrect response, we wait some time then resend the request, to account for stateful interference. If the second request fails, we mark the vantage point as broken and remove the vantage point from our test list. If the request results in a correct response, we mark both potential interference and stateful interference.

Hyperquack-v1 is built up from the Quack-v1 protocol to include support for the HTTP and HTTPS protocols. Before performing any tests, we send multiple HTTP get requests containing non-sensitive control URLs to each of the vantage points we are testing. If the responses to all of the requests are consistent, the responses are stripped of dynamic content such as cookies and turned into a template for the vantage point. Then when performing the tests with the sensitive keywords, we compare the vantage point's response to its template.

Our various publications and reports have used Quack-v1 and Hyperquack-v1 to detect many cases of application-layer interference. For instance, in our recent investigation into the filtering of COVID-19 websites , Quack-v1 was used to detect censorship in unexpected places like Canada.

1.3.3 Hyperquack-v2 (deprecated)

Hyperquack-v2 is our new version of both the Quack and Hyperquack measurement techniques. We've restructured the system to work as a request-based measurement server rather than a single-use measurement program. A user will run the program on a machine that will act as a server, and then users can interact with the program using a JSON API. The implications of this restructure are as follows.

- **Flexibility in Scheduling** - Unlike in Quack-v1 and Hyperquack-v1, when a scan is performed using Hyperquack-v2, a list of vantage points are added to Hyperquack-v2, then test keywords are added as work for the server to complete. When adding work, the user can specify which vantage points that work applies to, such as specifying all the vantage points in a given country, all the vantage points in a given subnet, or simply a list of specific vantage points. This allows users to more easily schedule targeted scans. To make differentiating between these concurrent scans easier, we also added a tagging system that allows for the output of Hyperquack-v1 to be redirected to custom files
- **On-the-fly Changes to Scans** - As a scan is running, the user can call endpoints to add work, add more vantage points, or remove vantage points. This further increases the flexibility of Hyperquack-v2, as scans can be updated in the middle of running as opposed to being re-run with updated parameters in Quack-v1 and Hyperquack-v1.
- **Stronger Vantage Point Evaluation** - In Quack-v1 and Hyperquack-v1, if a vantage point responded incorrectly to control probes, it would be completely removed from the scan. Since Hyperquack-v2 is continuously running, we have made it so a vantage point that fails one of the intermittent 'health checks' that Hyperquack-v2 performs has the potential to come back after a user-defined period of time. This will allow for greater coverage in cases where a vantage point experiences momentary failure.
- **Ability for More Complex Scheduling** - This paradigm allows for far more complex scheduling of work than the previous system. In future, our goal is to produce a system where users that want a scan performed can submit the scan parameters to a scheduler server, which will then send that work to any number of worker servers, each running an instance of Hyperquack-v2. This paradigm will allow for multiple workloads to be scheduled simultaneously alongside any rapid response scans that crop up.

Below is a list of the other major changes we've made to Hyperquack-v2.

- **Combining Quack and Hyperquack** - Hyperquack-v2 combines the Quack and Hyperquack measurement methods by creating a standard interface for how internet protocols can be used for internet censorship measurement. With this interface, new protocols can be easily added to Hyperquack-v2.
- **Changes to Output Format** - In addition to the output from censorship trial, Hyperquack-v2 outputs the results of the previously mentioned 'health checks' from vantage points. This output is very similar to the trial output, with the change that if the 'health check' is passed, a template will be included. All responses from the vantage point will be compared to the template to detect interference. At the moment, the templates for the Echo and Discard protocols are pre-defined by the protocol, so only the HTTP and HTTPS protocols will have these dynamically-computed templates included.

1.3.4 Notes

While Hyperquack-v2 includes multiple trials intended to avoid random network errors, there is still a possibility that certain measurements are marked as anomalies incorrectly. To confirm censorship, it is critical that the raw responses are compared to known blockpage fingerprints. The blockpage fingerprints currently recorded by Censored Planet are available here. Moreover, network errors (such as TCP handshake and Setup errors) must be filtered out to avoid false inferences. Please use our analysis pipeline to process the data before using it.

Censored Planet detects network interference of websites using remote measurements to infrastructural vantage points within networks (eg. institutions). Note that this raw data cannot determine the entity responsible for the blocking or the intent behind it. Please exercise caution when using the data, and reach out to us at *censoredplanet@umich.edu* if you have any questions.

1.4 Censored Planet Data Analysis

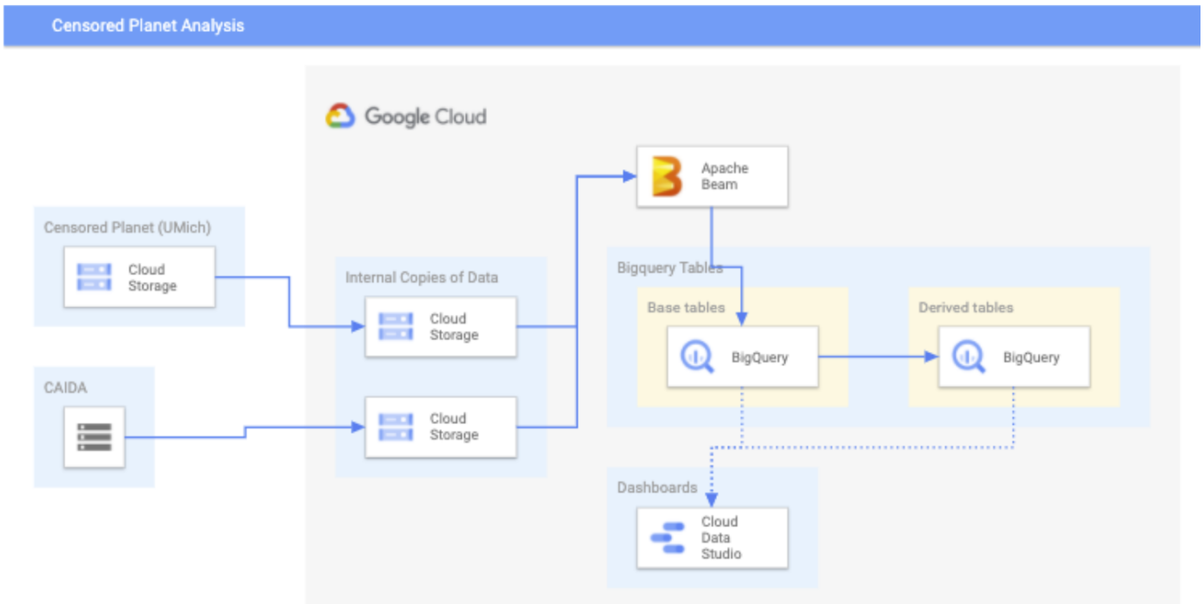
The Censored Planet processed data files can be found on our website. The website provides raw data files separated by each country for every measurement scan provided by Censored Planet. If you need individual measurement data for a specific country, the best avenue to explore Censored Planet data is to download individual files from the website and explore them.

For more large-scale data analysis, we have developed the Censored Planet data analysis pipeline and accompanying public Censored Planet Dashboard.

1.4.1 Censored Planet Data Analysis Pipeline

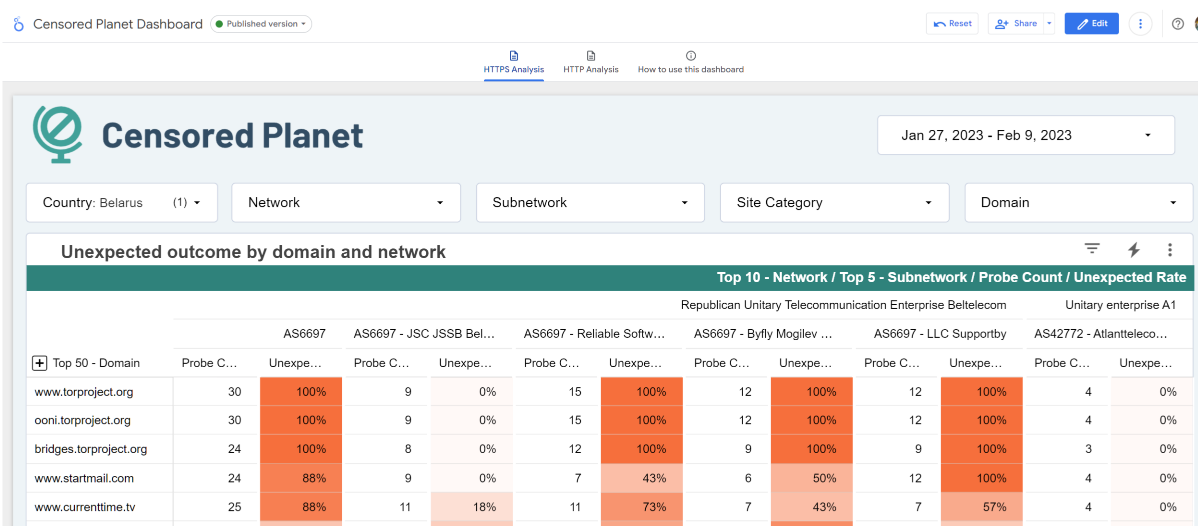
The Censored Planet data analysis pipeline, developed in collaboration with Google Jigsaw. takes raw data from the Censored Planet Observatory and runs it through a data analysis pipeline to create database tables for easier data analysis.

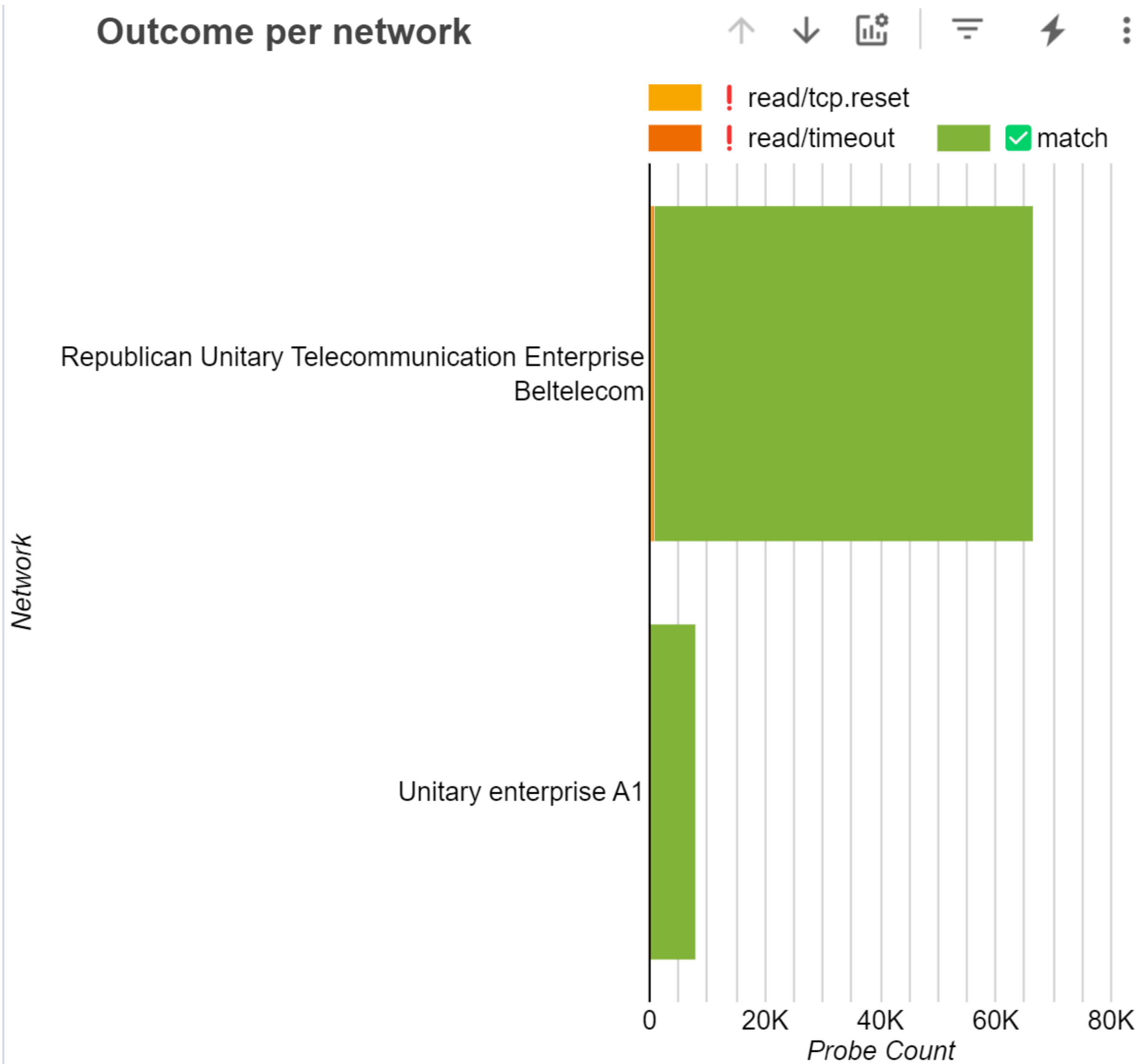
Because of the size of the data involved (many TB) this project requires a devoted Google Cloud project to run in. It is not recommended for end users to run the full pipeline because of cost considerations, but the code is made available for anyone who wants to understand how the data pipeline works. Moreover, users can apply the pipeline on a small portion of the data which is of interest, which will be more cost-effective. Please refer to the development documentation at our Github page and reach out to us at *censoredplanet-analysis@umich.edu* with any questions.

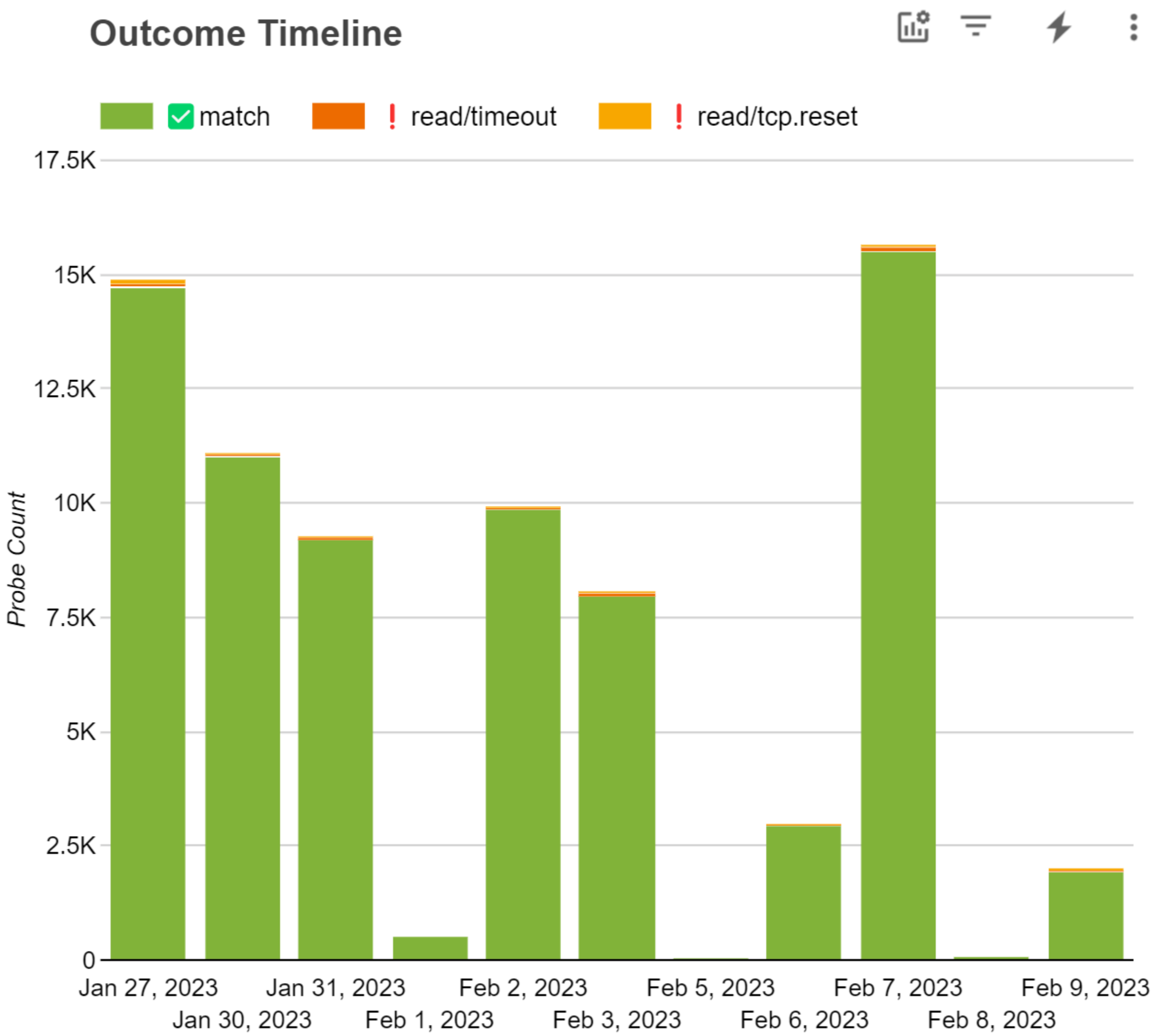


1.4.2 Censored Planet Dashboard

The Censored Planet Dashboard, built in collaboration with with Google Jigsaw is an exploratory public data dashboard that uses data analyzed using the Censored Planet data analysis pipeline, and contains visualizations that allow easy exploration of Censored Planet measurements.




Outcome per network
read/tcp.reset
read/timeout
match
Republican Unitary Telecommunication Enterprise Beltelecom
Unitary enterprise A1
Network
0
20K
40K
60K
80K
Probe Count



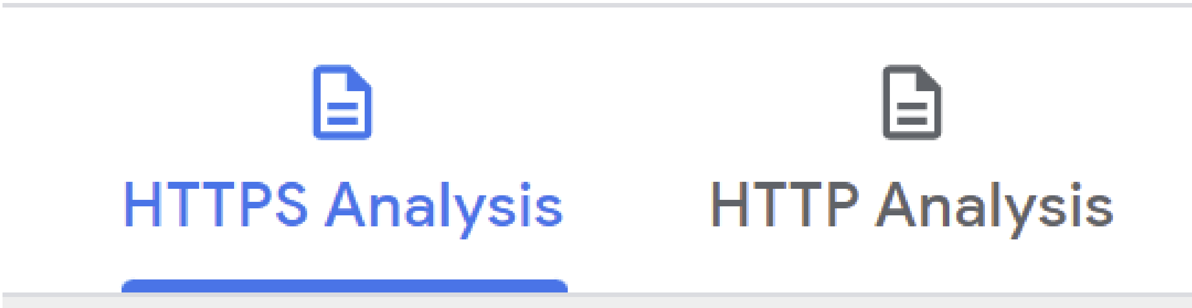
1.4.2.1 FAQ

Q. How do I access the dashboard?

A. The dashboard is public to all users. If you are having any access issues, reach out to *censoredplanet-analysis@umich.edu*.

Q. What data is available on the dashboard?

A. Currently, the dasboard contains data from Censored Planet's Hyperquack measurement technique, which measures network interference on the HTTP, HTTPS, Echo and Discard protocols. The measurements on the Echo and Discard protocols also aim to detect HTTP censorship. The two tabs on the top of the dashboard (see figure below) can be used to view the data related to HTTPS and HTTP tests. The DNS data from Satellite will be added to the dashboard soon.



Q. What domains does Censored Planet test and how often are the tests run?

A. Every week, Censored Planet runs two identical measurements to each domains from the Citizen Lab Global Test List and a selection of popular domains from the Alexa Top Domains list on each protocol. The dashboard also contains some data from special measurements to other domains (including those in the Citizen Lab regional lists). Only domains are tested during the longitudinal scans, but full URLs can be tested in special scans.

Q. How are site categories calculated?

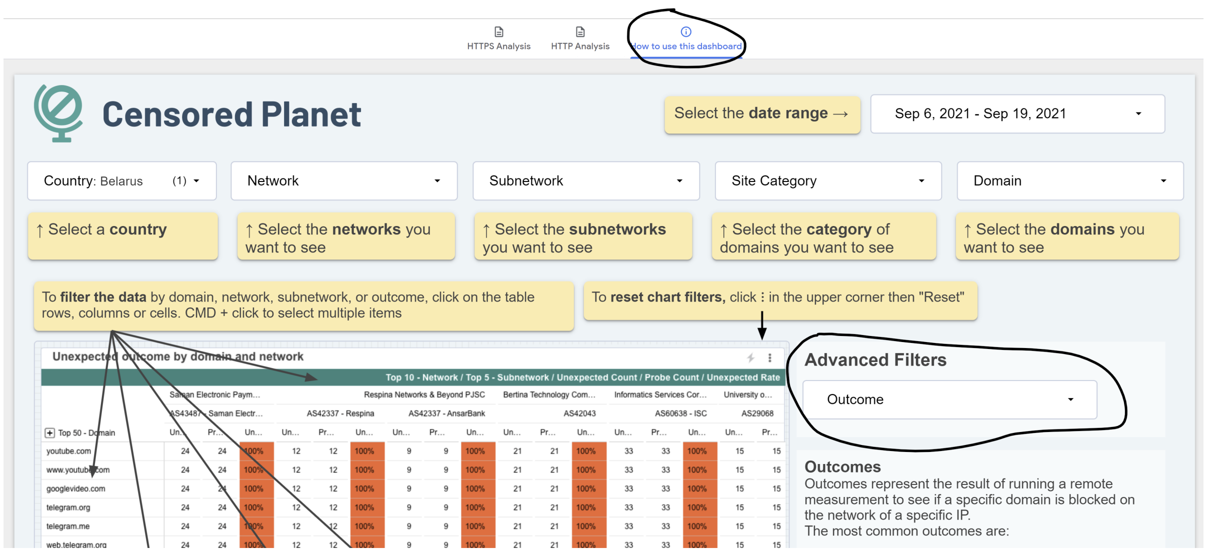
A. Site categories are derived from Citizen Lab, and some websites are categorized manually. The site categorization may contain errors. Please submit a corrected category at this form if you notice any errors in site categorization.

Q. Can I filter by outcome?

A. Yes, please use the outcome filter on the ‘How to use this dashboard’ pane (shown below) to filter certain outcomes. A brief description of outcomes is also provided.

Q. What do the emojis on the outcome mean (for example, an ‘’ before the ‘read/tcp.reset’)?

A. We use emojis to better communicate our confidence of a particular outcome being indicative of a censorship event. For all “expected” outcomes where we do not observe any censorship, we use the ‘’ emoji. For unexpected outcomes that are more indicative of censorship, we use the ‘’ emoji, while outcomes that may be both caused due to censorship as well as other reasons such as CDN localization or network errors are categorized using the ‘’ emoji. Failed tests are marked in gray.



1.4.2.2 Dashboard Walkthrough

We describe a demonstrative walkthough of how to use the Censored Planet dashboard to characterize censorship in a country.

1. Select country and time range
   - Open the dashboard and select the country of interest.
   - The dashboard will show the data for the last 14 days. Select a different time range if you would like to analyze a specific event. Keep in mind that longer ranges make the dashboard take longer to load.
2. Clean up the networks
   - Identify the networks of interest. This requires knowledge of the local context. You should remove all networks that do not correspond to local ISPs so they do not interfere with the row order.
     - The Customers per AS table from APNIC may be useful if you are not familiar with the local ISPs.
     - Keep in mind that not all ISPs may be present because of a limitation in the collection methodology.
   - Alternatively, you may select one Network at a time using the filter, but it helps to get a view across Networks first to see if there's consistency. Consistent blocking across Networks is evidence of a national policy.
   - Remove CDNs and private companies such as banks from the list of IP organizations. Banks tend to have a lot more censorship than ISPs and may use different methods.
3. Identify how each ISP blocks sites (see list of outcomes)
   - In the outcomes per network chart, you can click on the "Optional metrics" icon and select "Unexpected count". That will show what types of unexpected outcomes you get for each network.
   - Keep in mind that not all mechanisms are measurable from outside the network, so the site may look unblocked when in fact it is.
   - Near 100% of the unexpected probe count should be for a single outcome. If you see more than one unexpected outcome for a network, you may need to dig deeper. You can click on the down arrow to see the unexpected outcomes per subneworks. If each subnetwork only has one unexpected outcome, you have characterized the censorship mechanisms for them.
   - If you still see different outcomes in a subnetwork, it may be the case that different domains are blocked by different mechanisms. Reset the "Optional Metrics" to "Probe Count" and click on some of the domains and see if you get consistent results. If you get consistent results, you can drill back up to networks to see if they stay consistent within the network. That will give you a simpler view.
   - If you still see inconsistent results, you should check the Outline Timeline. It may be the case that censorship for a domain changed during the selected time period.
   - If the results are still inconsistent, you may need further investigation beyond the dashboard, and look at the raw data.
4. Identify the blocked websites and categories
   - Click on the domains to confirm how and where they are blocked. For this it's better to restore the "Optional metrics" to "Probe Count". Take note of the site category they are in.
   - You can click on the "+" button over the domains column to see their categories.
   - As you identify blocked categories, you may exclude them from the Site Category filter to clean up the list. Pornogrpahy and gambling wesbites are often blocked and will monopolize the list.
   - You can look for a specific domain using the Domain filter

5. Analyze both HTTP and HTTPS blocking. DNS and IP-based blocking are not available yet but will be added soon
6. It's always helpful to try to confirm your observations with independent corroborating evidence from OONI, or by running your own probes (having access to the IPs and commands would help here). Make sure the other data sources report the same outcome you've identified in the Censored Planet data.

CHAPTER TWO

INDICES AND TABLES

- genindex
- modindex
- search