IVRE

Sep 20, 2023

Contents

IVRE (French: *Instrument de veille sur les réseaux extérieurs*) or DRUNK (Dynamic Recon of UNKnown networks) is an open-source framework for network recon, written in Python. It relies on powerful open-source tools to gather intelligence from the network, actively or passively.

It aims at leveraging network captures and scans to let you understand how a network works. It is useful for pentests & red-teaming, incident response, monitoring, etc.

- Web site: https://ivre.rocks/
- Twitter: @IvreRocks
- Mastodon: @ivre@infosec.exchange
- Github: ivre/ivre

CHAPTER 1

Features

IVRE can aggregate scan results as well as intelligence from network captures. It accepts results from several tools:

- Active recon (network scanners):
  - Nmap
  - Masscan
  - Dismap
  - Tools from the ZMap project:
    - Zgrab2
    - ZDNS
  - Tools from the Project Discovery:
    - Nuclei
    - Httpx
    - Dnsx
- Passive recon (from network traffic and/or captures):
  - Zeek (formerly known as Bro)
  - p0f
  - airodump-ng
  - Argus
  - Nfdump

CHAPTER 2

Use-cases

IVRE can prove useful in several different scenarios (you may want to have a look at the *Screenshots gallery*). Here are some examples:

- Create your own Shodan-like service, using Nmap and/or Masscan and/or Zmap / Zgrab / Zgrab2, against the whole Internet or your own networks, (private or not).
- Store each X509 certificate seen in SSL/TLS connections, SSH public keys and algorithms, DNS answers, HTTP headers (`Server`, `Host`, `User-Agent`, etc.), and more… This can be useful to:
  - Validate X509 certificates independently from the clients.
  - Monitor phishing domains (based on DNS answers, HTTP `Host` headers, X509 certificates) hit from your corporate network.
  - Run your own, private (or not) passive DNS service.

CHAPTER 3

Getting started

If you want to learn more about the different purposes of IVRE, you should start reading the *Principles*.

After that, you can start the *Installation* process.

Once you are ready, dive into the "Usage" section!

CHAPTER 4

Contributing

Code contributions (pull-requests) are of course welcome!

The project needs scan results and capture files that can be provided as examples. If you can contribute some samples, or if you want to contribute some samples and would need some help to do so, or if you can provide a server to run scans, please contact the author.

CHAPTER 5

Contact

For both support and contribution, the repository on Github should be used: feel free to create a new issue or a pull request!

You can also join the Gitter conversation (that is the preferred way to get in touch for questions), or use the e-mail `dev` on the domain `ivre.rocks`.

On Twitter, you can follow and/or mention @IvreRocks.

On Mastodon, you can follow and/or mention @ivre@infosec.exchange.

CHAPTER 6

Content

6.1 Overview

6.1.1 Principles

IVRE is a network cartography (or network recon) framework.

Purposes

IVRE has five **purposes** (we use this word to refer to the different types of data IVRE handles), which can be stored by one or more **backend** databases:

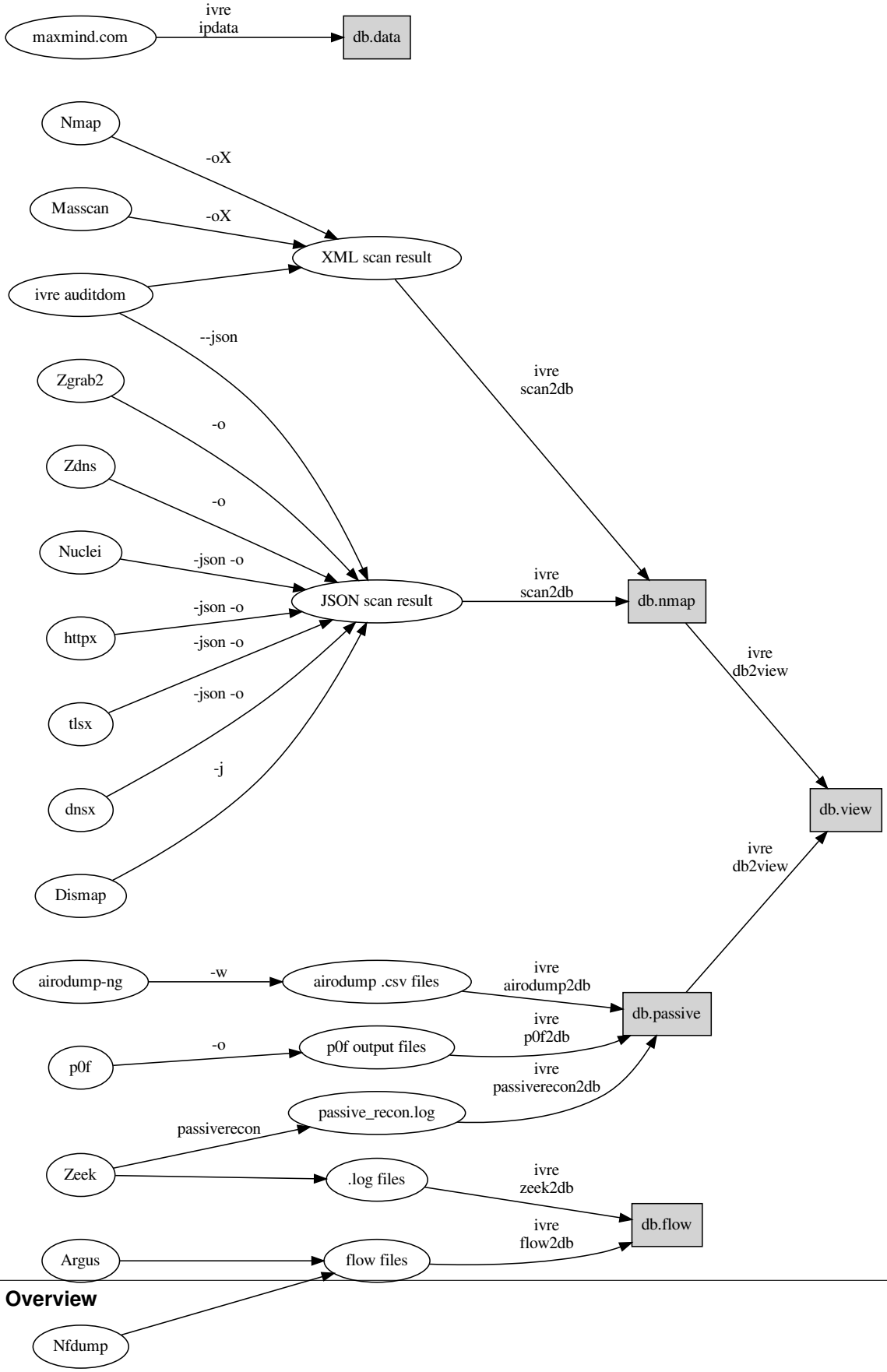
- `data`: associates IP ranges to Autonomous Systems (AS numbers and names), and geographical information (country, region, city), based on data from Maxmind GeoIP. It can be queried using:
  - Python API: the `db.data` object from the `ivre.db` module.
  - Command line: the `ivre ipdata` tool.
  - Web (JSON) API: the `/cgi/ipdata/<address>` URL.
- `nmap` (sometimes also referred to as `scans`): contains Nmap, Masscan, Dismap, Zgrab2, ZDNS, Nuclei, httpx, tlsx and dnsx scan results, as well as `ivre auditdom` results. Each record represents one host seen during one network scan. It can be queried using:
  - Python API: the `db.nmap` object from the `ivre.db` module.
  - Command line: the `ivre scancli` tool.
  - Web (JSON) API: the `/cgi/scans` and `/cgi/scans/*` URLs.
- `passive`: contains host intelligence captured from the network using a Zeek dedicated module called `passiverecon`, p0f and airodump-ng logs. Each record represents one piece of information (*e.g.*, the HTTP `Server:` header value `Apache` has been seen 10 times on port 80 of host 1.2.3.4). It can be queried using:
  - Python API: the `db.passive` object from the `ivre.db` module.

  - Command line: the `ivre ipinfo` and `ivre iphost` tools. The latter is dedicated to passive DNS queries.
  - Web (JSON) APIs: the `/cgi/passive` and `/cgi/passivedns` URLs. The latter is dedicated to passive DNS and is compatible with the Common Output Format implemented for example in CIRCL's PyPDNS.

- `view`: contains a consolidated view of hosts based on data from `nmap` and `passive`. The structure of the records is similar to `nmap`, but each record represents a host, seen during one or more network scans and/or seen from network captures. It can be queried using:
  - Python API: the `db.view` object from the `ivre.db` module.
  - Command line: the `ivre view` tool.
  - Web (JSON) API: the `/cgi/view` and `/cgi/view/*` URLs.
  - Web UI: the `/` or `/index.html` Web page.
- `flow`: contains aggregated network flows, as seen by Zeek, Argus or Netflows (using Nfdump). It can be queried using:
  - Python API: the `db.flow` object from the `ivre.db` module.
  - Command line: the `ivre flowcli` tool.
  - Web (JSON) API: the `/flows` URL.
  - Web UI: the `/flow.html` Web page.

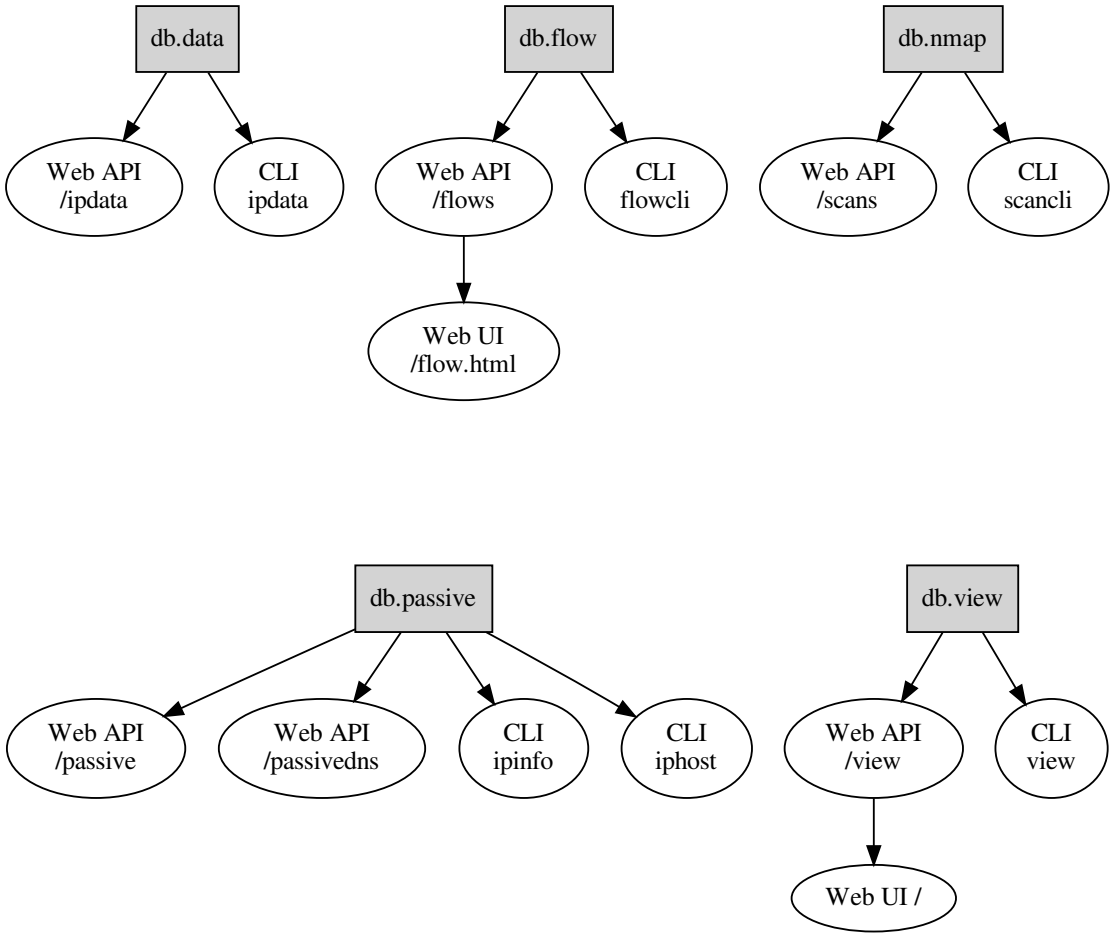
The following (non-exhaustive) figure shows how the data gets from your favorite open-source tools to IVRE's databases.

Storing data



Accessing data

The following (also non-exhaustive) figures show how the data gets from IVRE's databases back into your hands.



6.1.2 Screenshots gallery

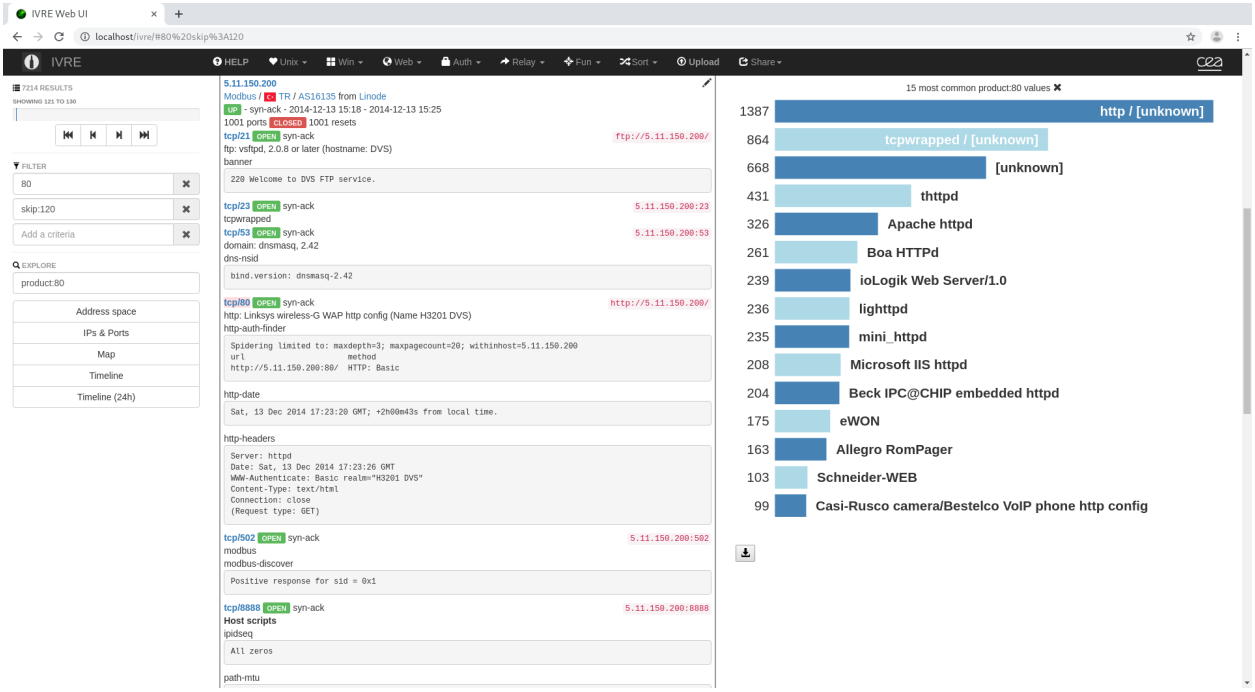
Nmap results

See *Active recon*.

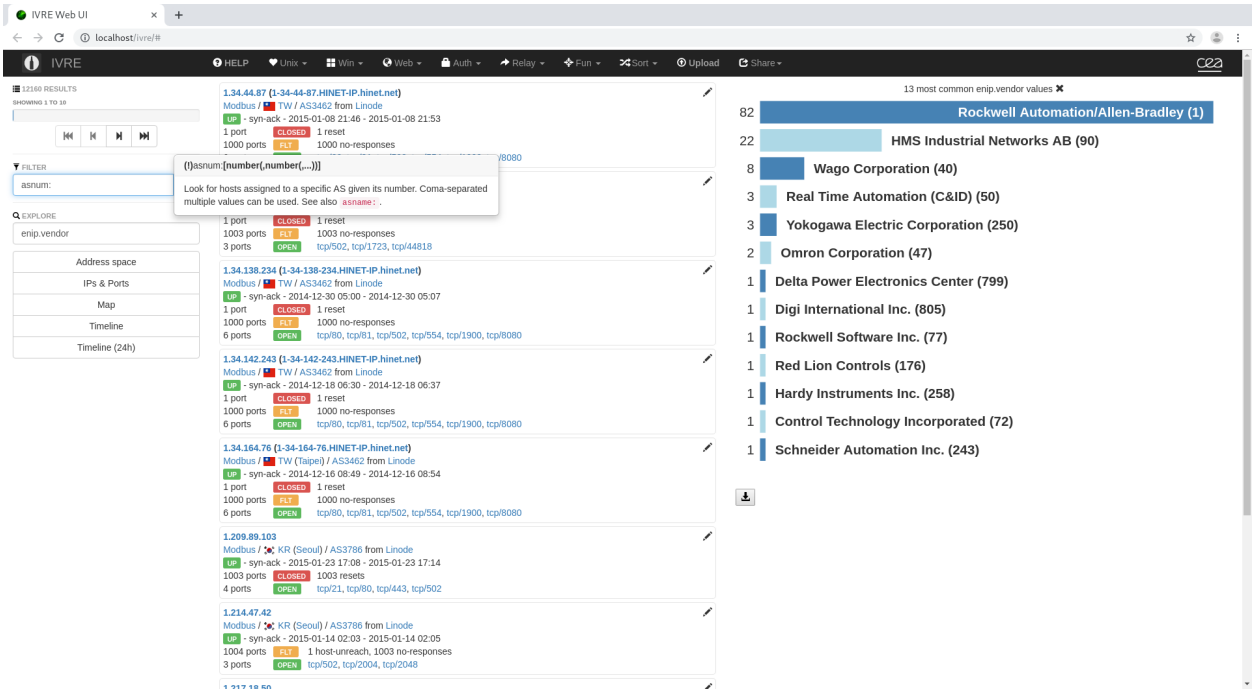
Home page with “heatmap” IP addresses.

Scan result details, using the “heatmap” IP addresses to “zoom” in the address space

Screenshots containing the word “solar” and map



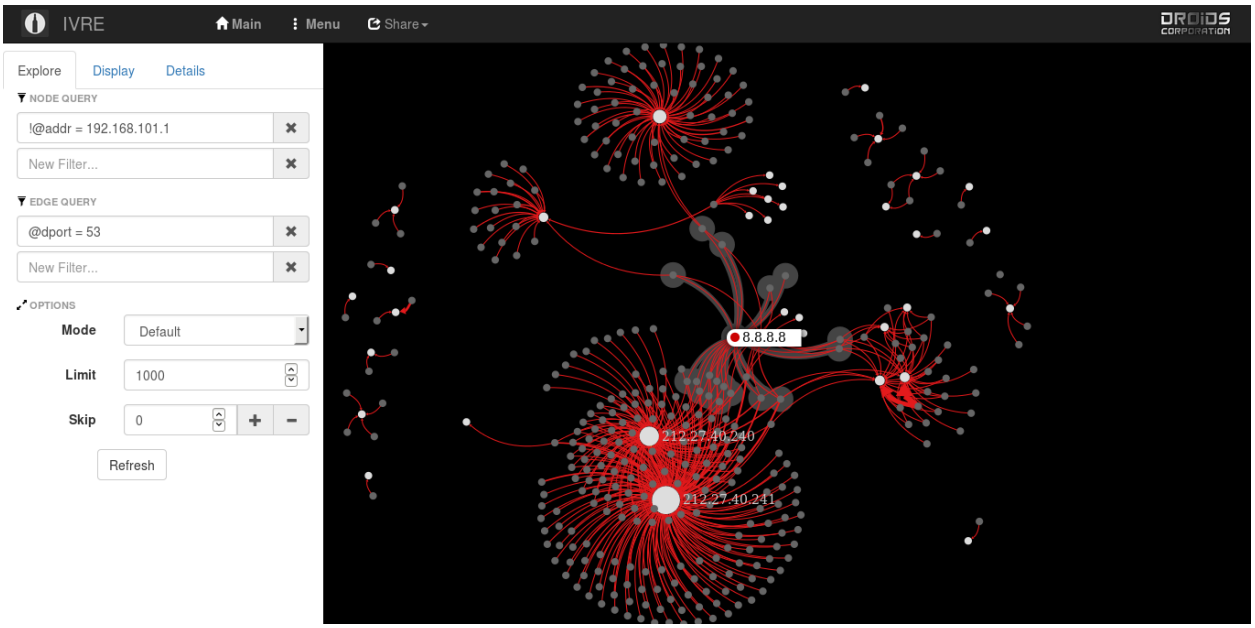
Most common products seen on port 80



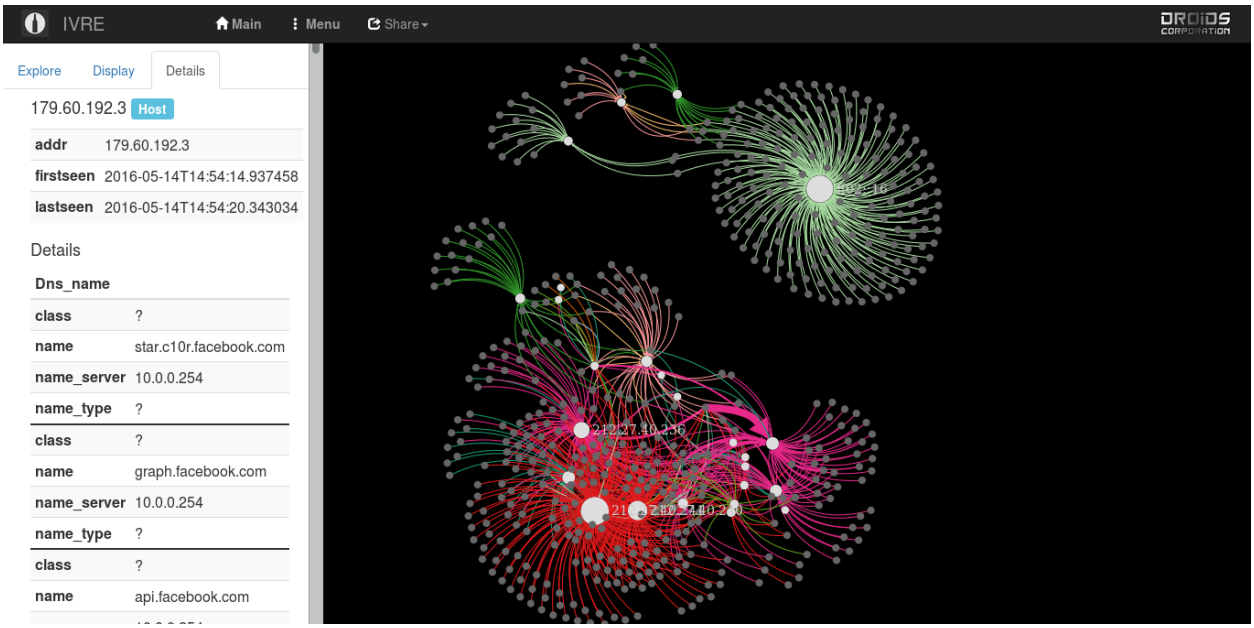
Help tooltip and most common ENIP vendors

Flow analysis

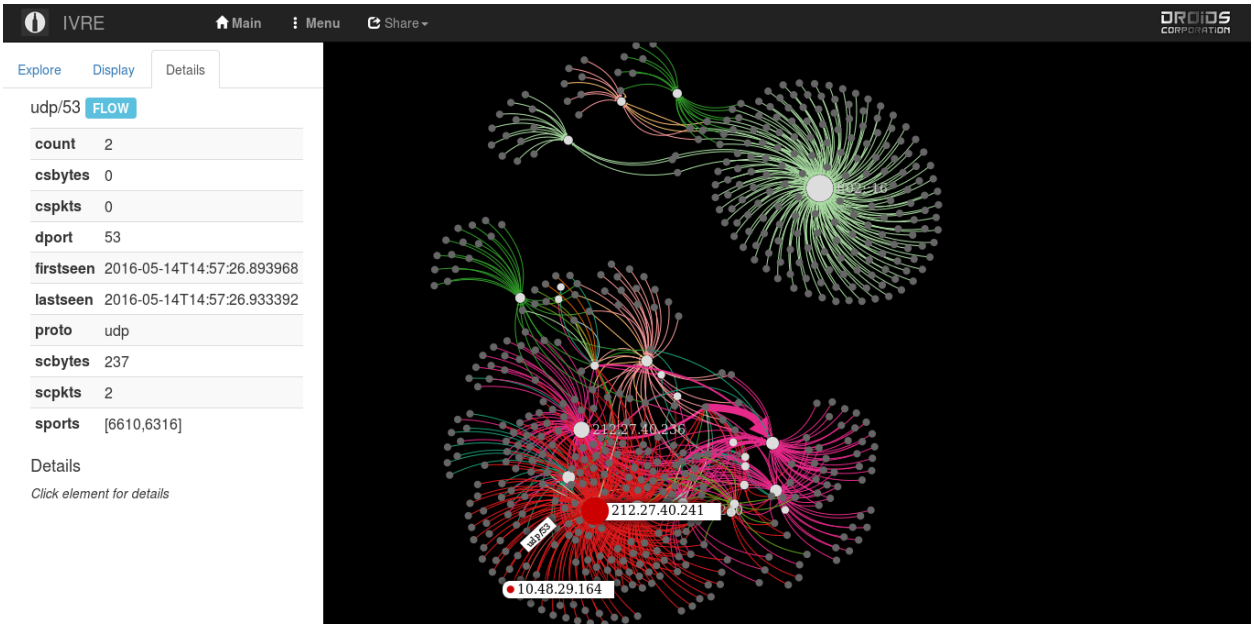
See *Flow*.



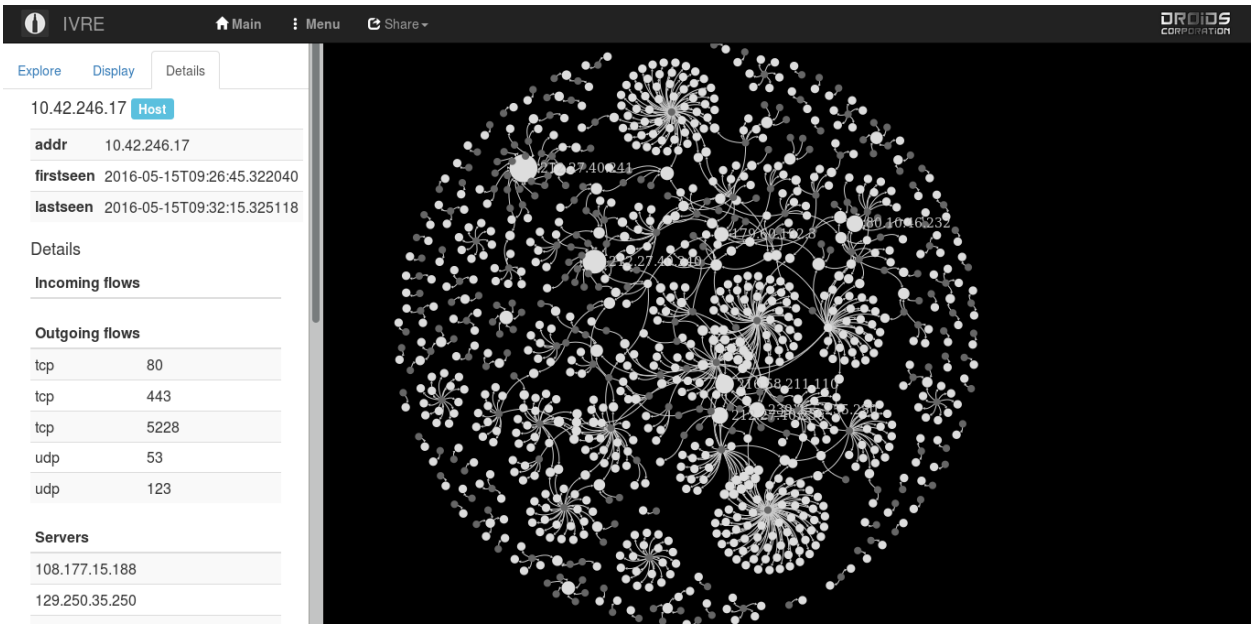
DNS flows with halo to show connected nodes



Flows with details for a specific host



Flows with details for a specific flow



Flow map

Passive network analysis

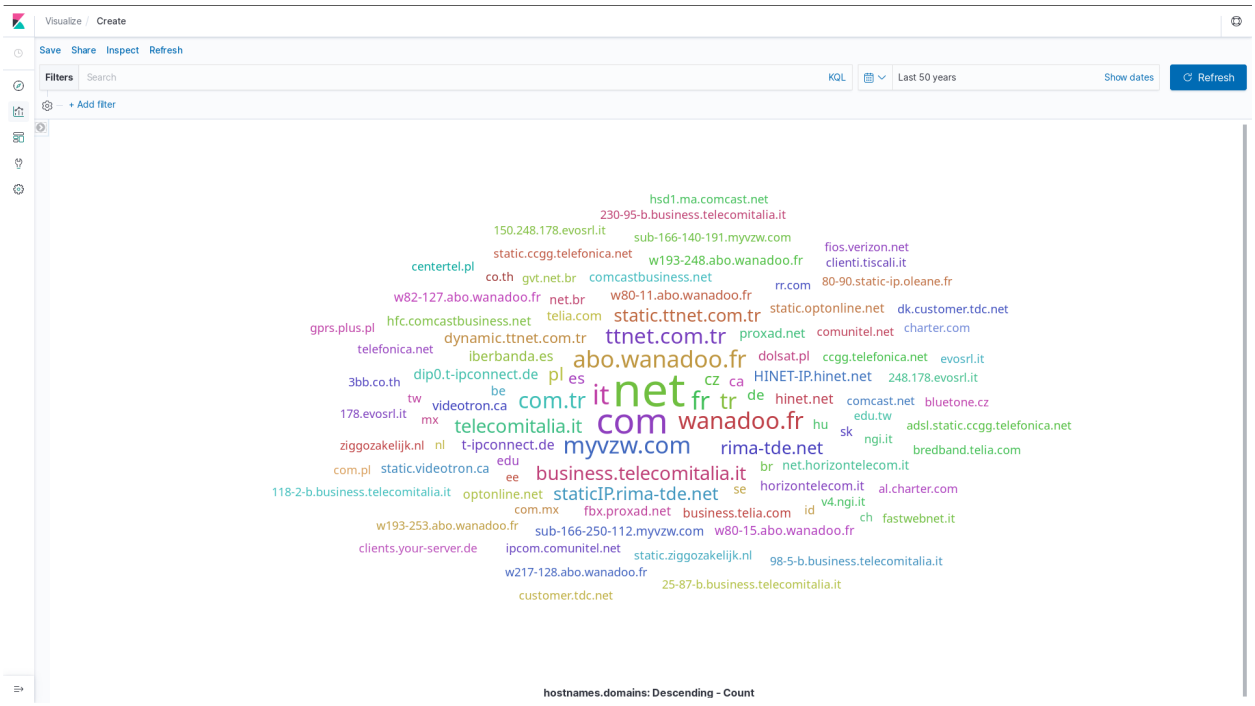
See *Passive*.

A simple passive analysis demonstration

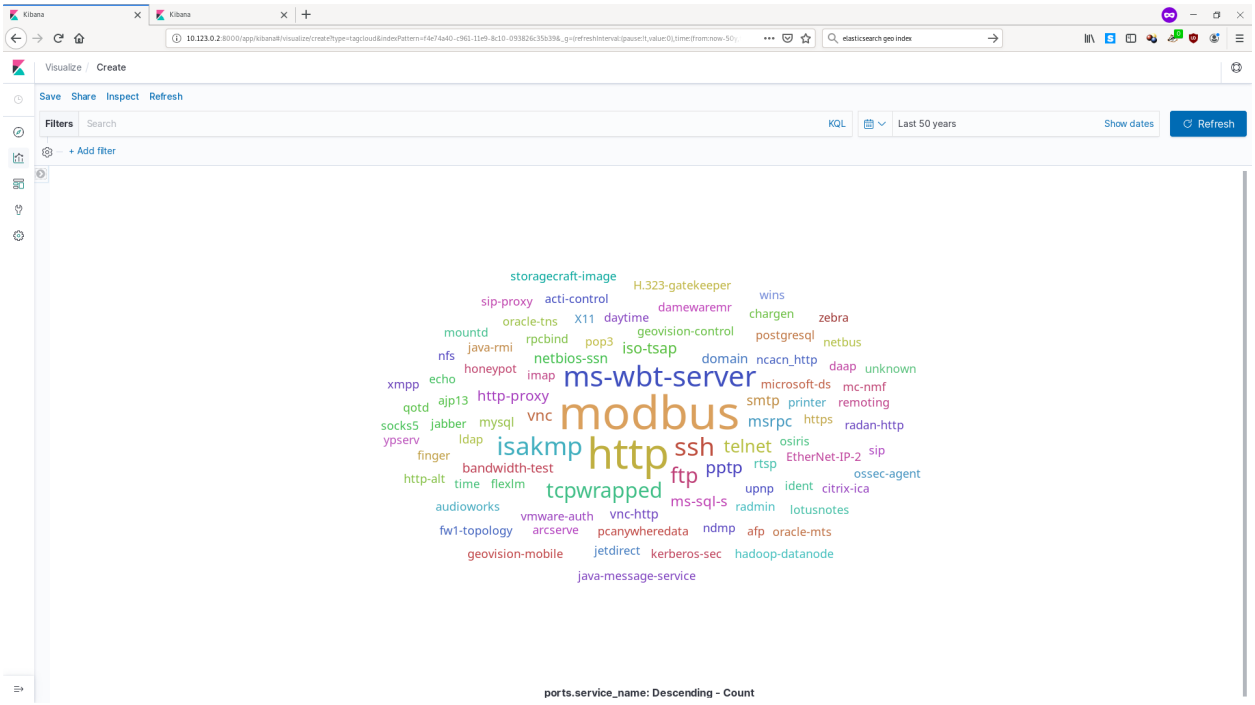
The data from the previous scene used to create an Nmap-like result

Kibana exploration

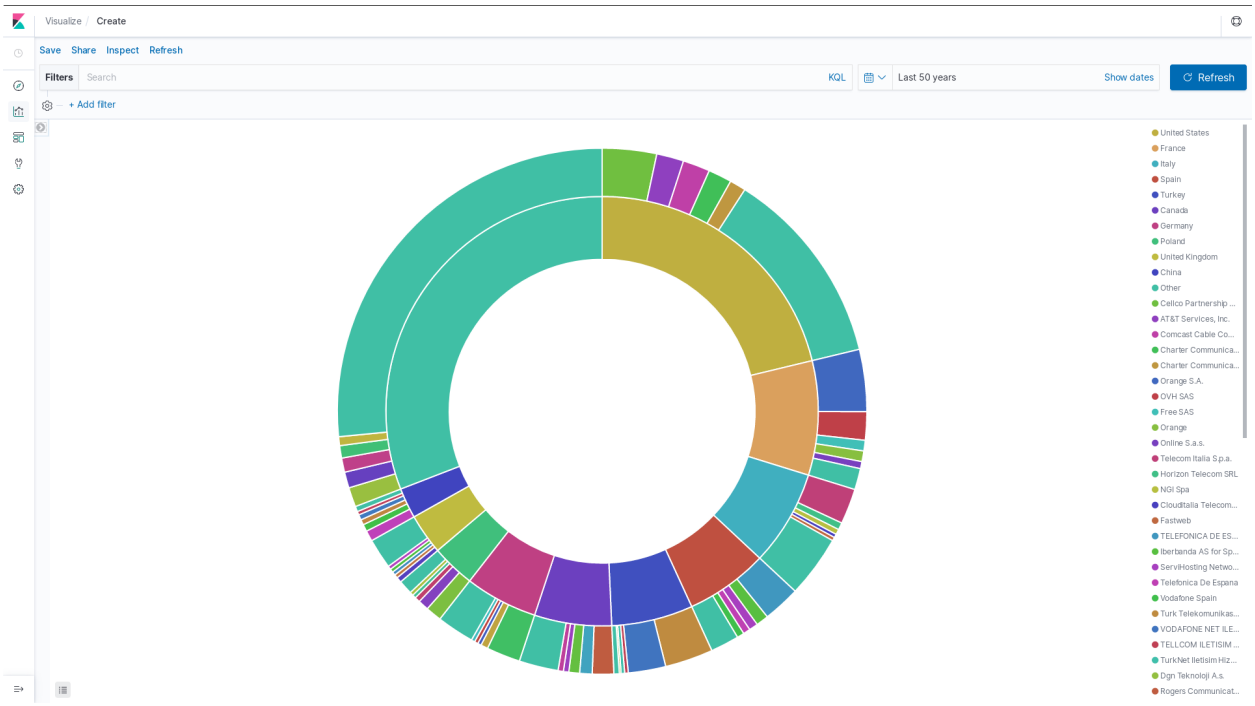
See *IVRE with Kibana*.



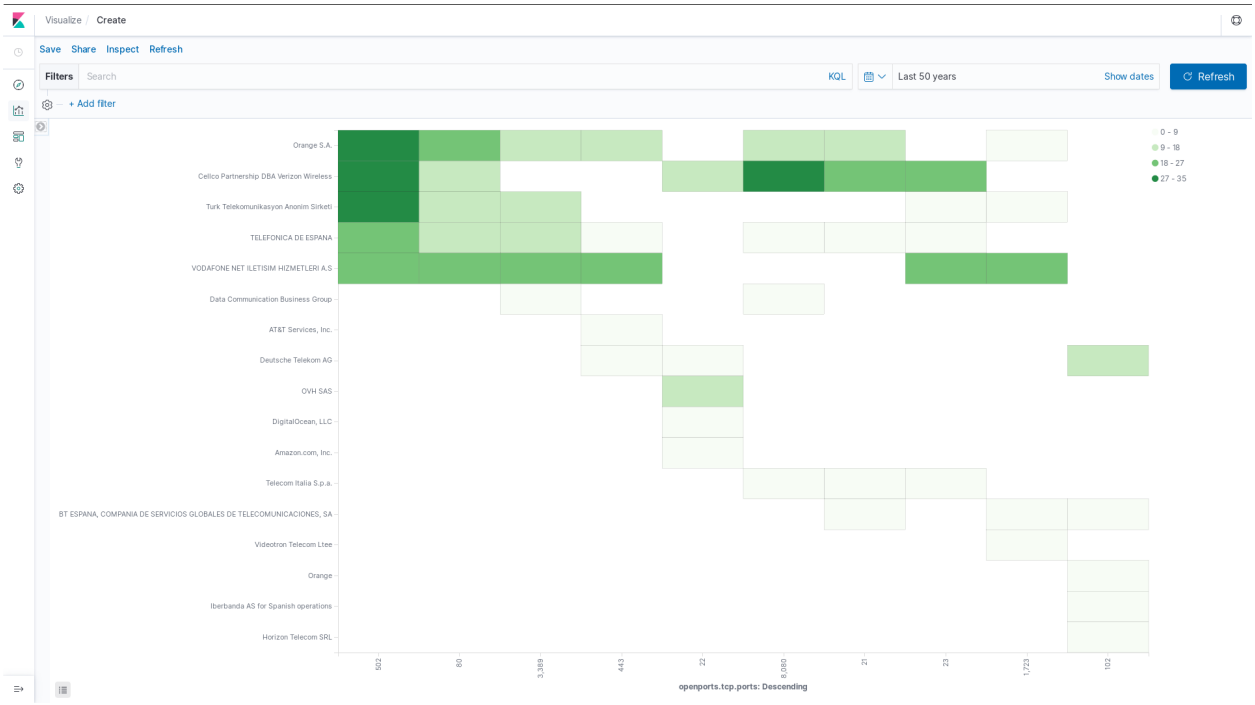
Domain names tag cloud



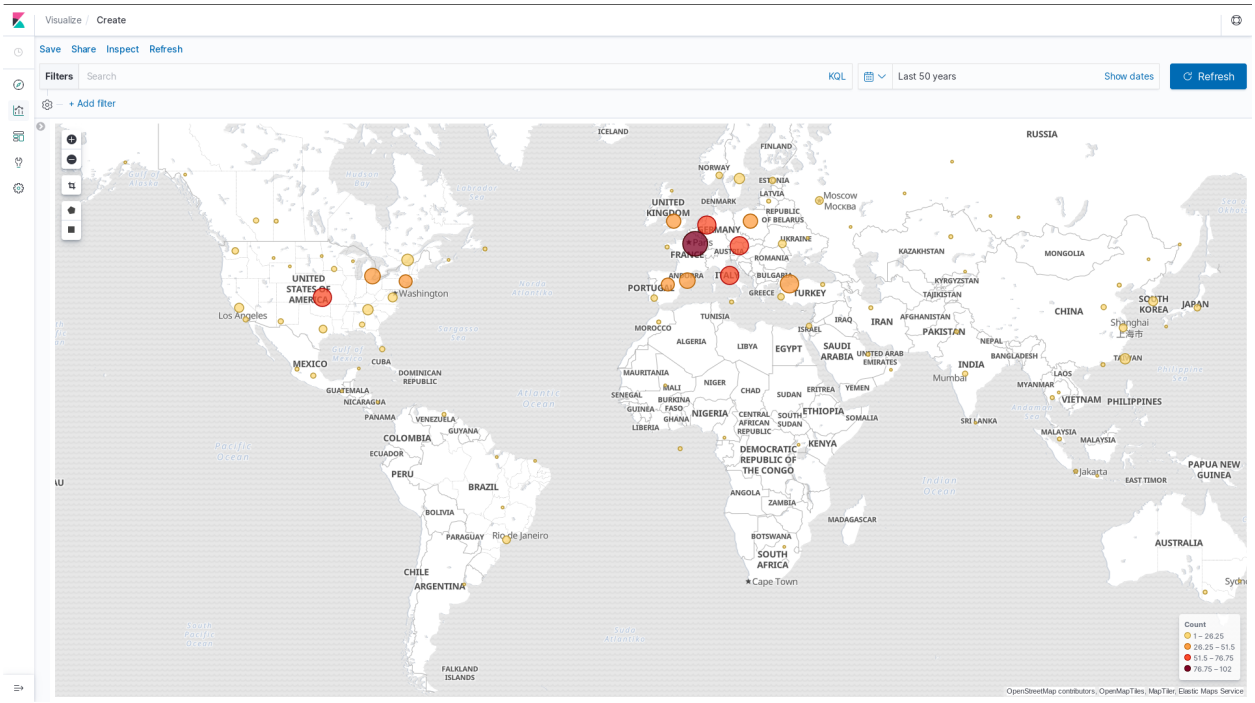
Service names tag cloud



Countries / AS numbers pie



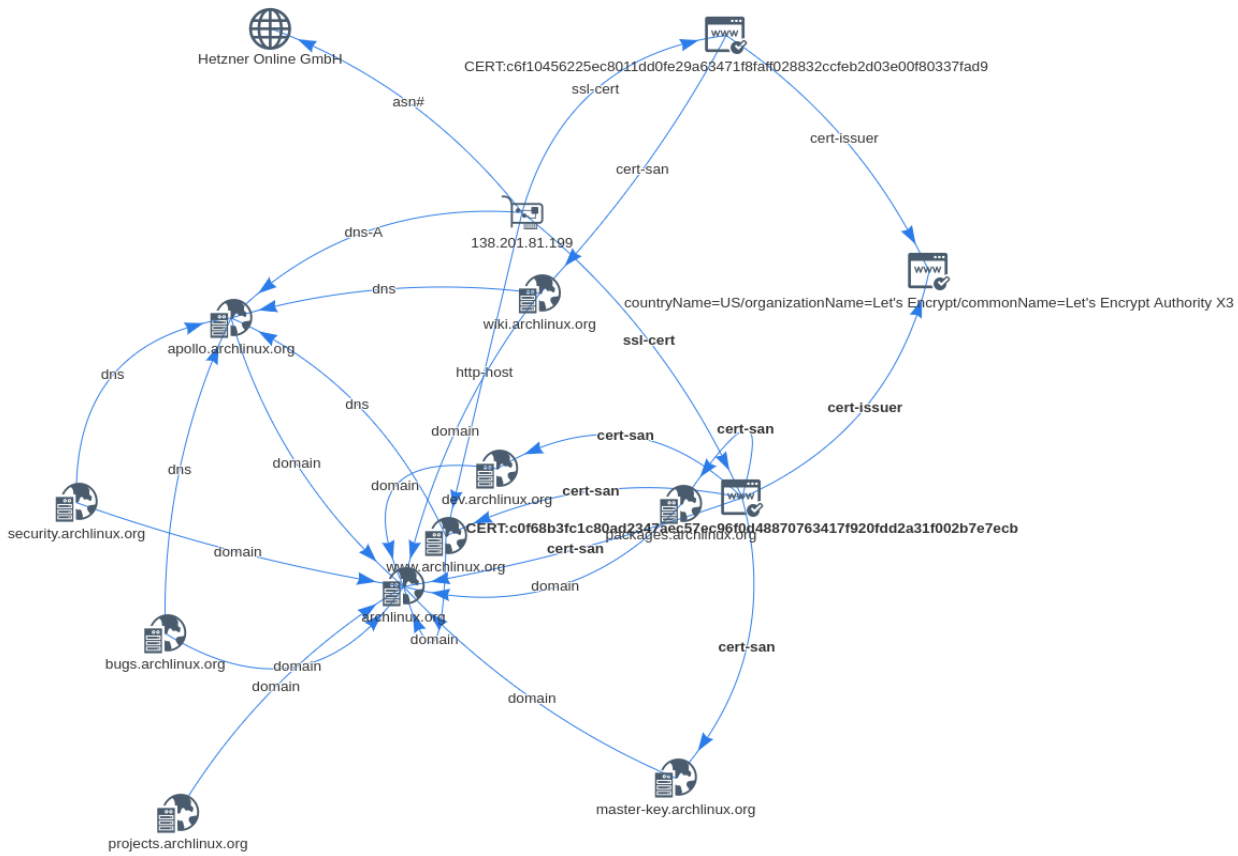
Heatmap showing correlations between AS and open ports



World map

IVRE as a plugin

See *YETI plugin*, *Cortex analyzer* and *OpenCTI connector* use cases.



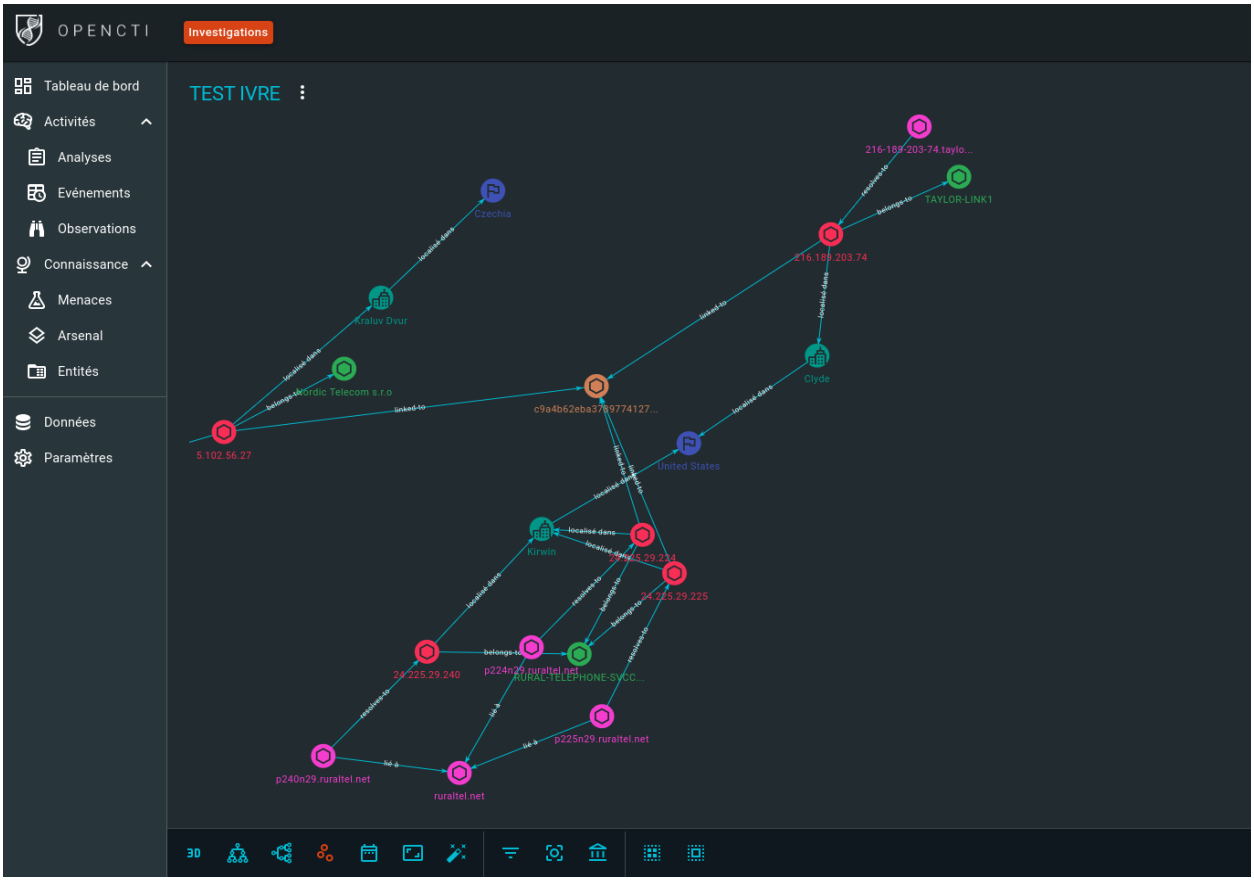
Fictitious investigation in Yeti about an IP address used by the archlinux.org domain, based on data from IVRE.

IVRE / scans for **80[.]38[.]223[.]227**

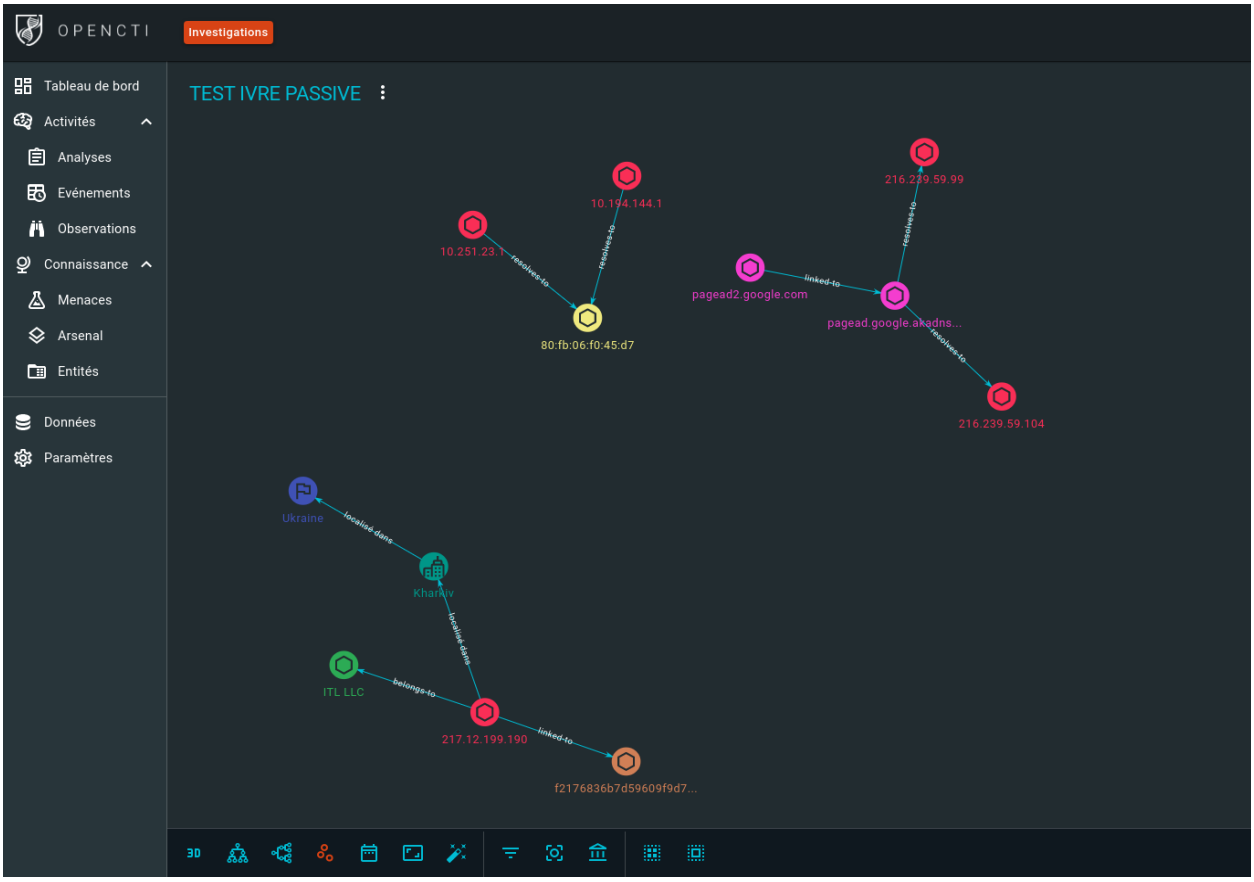
Results from 2021-01-12 14:27:54 to 2021-01-12 14:27:54
Sources: MySource
Categories: ROUTABLE / ROUTABLE-CAMPAIGN-001 / banner / clock-skew / fcrdns / http-auth-finder / http-comments-displayer / http-date / http-grep / http-headers / http-methods / http-robots.txt / http-security-headers / http-title / http-useragent-tester / ipidseq / path-mtu / qscan
Hostnames: 227.red-80-38-223.staticip.rima-tde.net / mk-hsava-balance3011
Ports, services & products

Open ports	tcp/53 / tcp/443 / tcp/1723 / tcp/2000 / tcp/8291 / tcp/8888
Services	bandwidth-test / domain / http / pptp
Products	MikroTik / MikroTik bandwidth-test server / MikroTik router config httpd

Cortex report about an IP address using data from IVRE.



Fictitious investigation in OpenCTI based on scans data from IVRE.



Fictitious investigation in OpenCTI based on passive data from IVRE.

6.1.3 FAQ

If you cannot find the answer to your question, either here or in this documentation, feel free to open an issue and use the label "question".

Web interface

Notebook shows "Forbidden"

I cannot access the notepad (the Dokuwiki content), and get a "Forbidden" message.

You need to configure your web server to allow access from other hosts on the network to the Dokuwiki content. It is often restricted, by default, to local users only. If you are using Apache, you can look for an ACL like `Allow from localhost 127.0.0.1 ::1` and adapt it to your network.

The Web interface shows no result

I have inserted scan results, yet when I open the Web interface, it remains empty.

Two problems can explain this situation:

- The results are stored in the scan collection, but no view has been created (the Web interface displays results from the view).

- The Web interface does not access the database for some reason.

First, from the command line, check that a view has been created by running `ivre view --count`. If it displays `0`, it means that while you have inserted results in the `scan` database, you have not updated the `view` (see *Purposes*). You can create a view by using the `ivre db2view` CLI tool.

If `ivre view --count` does not display `0` but a (positive!) number, it means that, for some reason, the CGI cannot access the database. It could be because you are using a user-specific configuration (in `~/.ivre.conf`) and the CGI application runs with a different user. To investigate the problem, you have to check the Web server error logs.

How can I restrict access to IVRE's Web interface

I want to prevent unauthorized access to IVRE's results.

First, you have to configure your web server to authenticate remote users. The most important, of course, is to protect access to CGI files (the static files are publicly available and do not contain any result).

In an AD or Kerberos environment for example, Apache can be configured to provide SSO authentication.

Then, if you want to restrict access to the results based on the user login or domain, you can add the following lines to `/etc/ivre.conf`:

```
WEB_DEFAULT_INIT_QUERY = 'noaccess'
WEB_INIT_QUERIES = {
    'admin@SUBNETWORK.NETWORK.AD': 'category:SubNetwork',
    '@ADMIN.NETWORK.AD': 'full',
}
```

By default, users won't have access to any result. The user `admin@SUBNETWORK.NETWORK.AD` will have access to the results in the category `SubNetwork`. The users in the `ADMIN.NETWORK.AD` realm will have access to all the results.

Scanning the Internet is slow!

This is based on issue GH#822.

When running `ivre runscans --routable --limit 40`, one can notice the scan really takes a long time to terminate.

First of all, IVRE is not guilty here. IVRE runs Nmap, feeds it with targets, and wait for its output. You would get the same results using the same Nmap options as IVRE.

That being said, we have several ways to speed up a scan.

Use Masscan rather that Nmap

This is pretty radical, and have an important drawback: Masscan results gather less intelligence than Nmap (a lot less in some situations).

However, it is often the only option to get comprehensive scans of the IPv4 routable address space.

A trade-off could be, for some protocols, to use Zmap / Zgrab2. Compare the possibilities of Masscan (`--banner`) versus Zgrab2 for the protocol(s) you want to scan.

IVRE will happily combine results from Nmap, Masscan and Zgrab / Zgrab2: you can build your own, perfectly suited, scanning solution and use IVRE to merge and browse the results.

Parallelize Nmap scans

Another option is to run several Nmap processes instead of one. Theoretically it should not work, since Nmap is supposed to handle efficiently the resources, but it has proven useful in several situations, particularly when scanning heavily filtered hosts or random hosts across the Internet.

For that, one can either use an agent (see *Agents*) or `ivre runscans --output XMLFork --processes <n>` where `<n>` is the number of simultaneous Nmap processes to use.

Can IVRE be used to look for XXX?

IVRE is not a scanner or a network traffic analyzer. It relies on tools like Nmap, Masscan, ZGrab2, and Zeek, parses their results and stores them in a database.

So when you are asking, for example, "can IVRE scan a network for hosts with the Heartbleed vulnerability?", in reality you are asking two different questions:

- "Can Nmap or Masscan or Zgrab2 detect when a scanned hosts is vulnerable to the Heartbleed vulnerability?"
- "How can IVRE list the hosts that have been found vulnerable to Heartbleed by Nmap or Masscan?"

The first question is not related to IVRE (and should probably be asked to Nmap, Masscan or Zgrab2 developers), but the second question is (and may be asked as a "question" labeled issue).

For that particular Heartbleed example, Nmap, Masscan and Zgrab2 can (reliably) report hosts with the Heartbleed vulnerability, and IVRE can be used to find such hosts.

How can I configure iptables to get logs used by flow2db tool

When you don't have access to low level network data, an easy way to discover a part of network traffic is to use netfilter logs collected via syslog.

To be efficient, all the systems must have iptables activated and configured to send logs.

For example

```
-A INPUT   -j LOG --log-prefix "IPTABLES/INPUT: "
-A OUTPUT  -j LOG --log-prefix "IPTABLES/OUTPUT: "
-A FORWARD -j LOG --log-prefix "IPTABLES/FORWARD: "
```

To log all traffic, the rules can be set at the top of all rules. Be careful with the OUTPUT rule if the logs are sent over the network!

On the syslog server or on each host, just run grep to collect the data needed for the iptables flow2db parser:

```
$ grep -l 'IPTABLES/' /var/log/syslog /var/log/kernel.log ... \
    > syslog-iptables.log
```

Then import data to ivredb using flow2db tool:

```
$ ivre flow2db -t iptables syslog-iptables.log
```

6.2 Installation

6.2.1 Installation guidelines

Database

Depending on the backends you wan to use, install a database server. Please keep in mind that currently, MongoDB is currently the only supported backend for all the purposes. To learn more about the different purposes, read the *Principles*.

The database servers installation and setup is not covered here, and depends on your platform and needs. Please refer to the server documentation on how to install it. For MongoDB you can read the installation section of their documentation.

Dependencies

External programs

If you plan to run scans from a machine, install Nmap, Masscan, and/or Zmap / Zgrab / Zgrab2. If you want to integrate screenshots, install Tesseract, ImageMagick, FFmpeg and PhantomJS.

If you plan to analyze PCAP file on a machine, install, depending on your needs:

- Zeek (previously known as Bro, version 3 minimum).
- Argus.
- Nfdump.

Python

To install IVRE, you'll need Python 3.7 minimum, with the following modules:

- bottle.
- cryptography.
- pymongo version 3.7 minimum.
- tinydb, to use the **experimental** TinyDB backend (this does not require a database server).
- sqlalchemy and psycopg2 to use the **experimental** PostgreSQL backend.
- elasticsearch and elasticsearch-dsl to use the **experimental** Elasticsearch backend.
- PIL optional, to trim screenshots.
- pyOpenSSL version 16.1.0 minimum, optional, to parse X509 certificates (a fallback exists that calls `Popen()` the `openssl` binary and parses its output, but it is much slower and less reliable).

Databases

IVRE's reference backend service is MongoDB, version 3.6 minimum. It is highly suggested that you use the latest stable release (the performances tend to improve a lot).

The `passive`, `nmap` and `view` purposes have an **experimental** PostgreSQL backend that can be used in lieu of MongoDB.

The `view` purpose has an **experimental** Elasticsearch backend. It can be used to create views accessible to other Elasticsearch tools, such as Kibana (see *IVRE with Kibana*).

Please refer to the database servers (or your distribution) documentation on how to install and configure them.

Web

For production services, it is recommended to install either Apache with the WSGI module, or Nginx with uWSGI.

IVRE can use Dokuwiki as its notepad, it is also recommended to install it.

Please refer to the servers (or your distribution) documentation on how to install and configure them.

Configuration file samples are provided in IVRE's source repository, under `pkg/apache` and `pkg/nginx`. Also, the *Docker* creation files in `docker/web*` can provide useful examples.

If you do not want (or cannot) to install a Web server, you can try IVRE's integrated server, suited for tests or tiny installations. Just run `ivre httpd`!

IVRE

The installation of IVRE itself can be done:

- On Kali, just install the package by running `apt update && apt install ivre`. You can also install `ivre-doc` if needed.
- On Fedora, you can use the Copr package; follow the instructions.
- On other RPM-based Linux distributions, you can easily build RPM packages (using the provided `pkg/buildrpm` script, or use the `setup.py` script with your own options).
- On Arch Linux, there are AUR packages that can be installed using yay for example. The packages are:
  - `ivre`: the main package, which depends on `python-ivre`.
  - `python-ivre` the Python library.
  - `ivre-web`: the Web application.
  - `ivre-docs`: the documentation.

  These packages are based on the latest stable version; they all have a `-git` version, based on the current development code from the Github repository. You can install for example `ivre-git` and `ivre-web-git` if you want to test the latest developments.

  All the packages are based on the same bases: ivre and ivre-git.
- On BlackArch Linux (an Arch Linux-based penetration testing distribution) IVRE is packaged (and installed in the Live ISO).
- Using pip: run `pip install ivre` (this will download and install for you the IVRE package and its Python dependencies from PyPI, the Python Package Index).
- From the source code, using the `setup.py` (classical `./setup.py build; sudo ./setup.py install`) script.
- Using *Docker* (in this case you do not need to follow the instructions in *Configuration*, as the Docker containers are already configured).

Configuration

You can set configuration values in several files:

- system-wide: `ivre.conf` in the following directories: `/etc/`, `/etc/ivre`, `/usr/local/etc`, `/usr/local/etc/ivre`.
- user-specific: `~/.ivre.conf` (read after the system-wide configuration files, so highest priority).

- execution-specific: another configuration file can be specified using the `$IVRE_CONF` environment variable.

The configuration files are Python files. They may set, for example, the variable `DB` to use a different database than the default one.

See *Configuration* to learn more about the different configuration parameters.

Initialization

Once IVRE has been properly configured, it's time to initialize its databases.

For that, the command-line tools (namely `ivre ipinfo`, `ivre scancli`, `ivre view`, `ivre flowcli` and `ivre runscansagentdb`, respectively for information about IP addresses, passive information, active information and running scans through agents) have a `--init` option.

So you can run, with a user or from a host where the configuration has a write access to the database (add `< /dev/null` to skip the confirmation):

```
$ yes | ivre ipinfo --init
$ yes | ivre scancli --init
$ yes | ivre view --init
$ yes | ivre flowcli --init
$ yes | sudo ivre runscansagentdb --init
```

Getting IP data

To fetch the IP address data files (mainly from Maxmind) and parse them (required if you want to scan or list all IP addresses from a country or an AS), just run the following command (it takes a long time, usually more than 40 minutes on a decent server):

```
$ sudo ivre ipdata --download
```

It is advised to run this command on a regular basis (e.g., weekly). If you use IVRE on several machines, you may want to run the command on one machine and create an `ivre-data` package containing the files under the `/usr/share/ivre/geoip` directory (or distribute those files somehow).

The URLs downloaded are stored in the configuration. By default, the following files are downloaded:

```
$ python
>>> from ivre.config import IPDATA_URLS
>>> for fname, url in IPDATA_URLS.items():
...     print("%s: %s" % (fname, url))
...
GeoLite2-City.tar.gz: https://ivre.rocks/data/geolite/GeoLite2-City.tar.gz
GeoLite2-City-CSV.zip: https://ivre.rocks/data/geolite/GeoLite2-City-CSV.zip
GeoLite2-Country.tar.gz: https://ivre.rocks/data/geolite/GeoLite2-Country.tar.gz
GeoLite2-Country-CSV.zip: https://ivre.rocks/data/geolite/GeoLite2-Country-CSV.zip
GeoLite2-ASN.tar.gz: https://ivre.rocks/data/geolite/GeoLite2-ASN.tar.gz
GeoLite2-ASN-CSV.zip: https://ivre.rocks/data/geolite/GeoLite2-ASN-CSV.zip
GeoLite2-dumps.tar.gz: https://ivre.rocks/data/geolite/GeoLite2-dumps.tar.gz
iso3166.csv: https://dev.maxmind.com/static/csv/codes/iso3166.csv
BGP.raw: https://thyme.apnic.net/current/data-raw-table
```

Using Agents

If you do not plan to run active scans with remote agents (where IVRE will not be installed), you can skip this section.

The agent does not require IVRE to be installed. It is a script that needs to be adapted to each situation.

The agent is only needed when you cannot install IVRE on the machine used to scan or when you want to use several machines to run one scan.

It requires a POSIX environment, and the commands `screen`, `rsync` and `nmap` (of course). See the *Agents* documentation for more information about that.

6.2.2 Configuration

IVRE has several configuration variables. The default values are hard-coded in `ivre/config.py`. You should not change this file, unless you are modifying IVRE and you want to change the default configuration. You do not need to do this if you want to install IVRE with a non-default configuration, you just need to distribute a proper configuration file.

IVRE can be configured using different configuration files:

- system-wide: `ivre.conf` in the following directories: `/etc/`, `/etc/ivre`, `/usr/local/etc`, `/usr/local/etc/ivre`.
- user-specific: `~/.ivre.conf` (read after the system-wide configuration files, so higher priority).
- execution-specific: another configuration file can be specified using the `$IVRE_CONF` environment variable (read after the user-specific file, so highest priority).

The configuration files are Python files setting global variables.

Debug

Debug messages are turned off by default, since IVRE has no bugs. `DEBUG_DB` turns on database-specific debug messages, and can be very noisy. Setting `DEBUG` to `True` is mandatory to run IVRE's tests.

Databases

Databases are specified using URLs:

```
db_type://[username[:password]@][host[:port]]/databasename?options
```

`DB` is the generic database URL (will be used for all *Purposes* unless a purpose-specific URL has been specified). The value `"mongodb:///ivre"` is the default and means "use MongoDB on localhost, database `ivre`, default collection names".

Purpose-specific URLs can be specified using `DB_<purpose>`; `DB_DATA` is specific and defaults to `None`, which has the special meaning `"maxmind:///<ivre_share_path>/geoip"`.

Here are some examples:

```
DB_PASSIVE = "sqlite:////tmp/ivre.db"
DB_NMAP = "postgresql://ivre@localhost/ivre"
DB_VIEW = "elastic://192.168.0.1:9200/ivre"
DB_DATA = "maxmind:///share/data/ivre/geoip"
```

Batch insert or upsert operations can be tuned using backend-specific variables:

```
LOCAL_BATCH_SIZE = 10000  # used with --local-bulk
MONGODB_BATCH_SIZE = 100
POSTGRES_BATCH_SIZE = 10000
```

Paths and commands

All variables ending with `_PATH` (except `AGENT_MASTER_PATH` and `NMAP_SHARE_PATH`) default to `None`, a special value which means "try to guess the path based on IVRE installation".

Here are the values with examples on a regular installation:

```
DATA_PATH = None                    # /usr/share/ivre/data
GEOIP_PATH = None                   # /usr/share/ivre/geoip
HONEYD_IVRE_SCRIPTS_PATH = None     # /usr/share/ivre/data/honeyd
WEB_STATIC_PATH = None              # /usr/share/ivre/web/static
WEB_DOKU_PATH = None                # /usr/share/ivre/dokuwiki
```

`AGENT_MASTER_PATH` defaults to `"/var/lib/ivre/master"`.

`NMAP_SHARE_PATH` defaults to `None`, which means IVRE will try `"/usr/local/share/nmap"`, `"/opt/nmap/share/nmap"`, then `"/usr/share/nmap"`.

IVRE may need some executables:

```
TESSERACT_CMD = "tesseract"
OPENSSL_CMD = "openssl"
```

Nmap scan templates

Nmap scan templates are defined in the `NMAP_SCAN_TEMPLATES` variable. Usually, this variable should **not** be overridden, but rather modified.

By default, `NMAP_SCAN_TEMPLATES` contains one template, named `"default"`, which is defined as follows:

```
NMAP_SCAN_TEMPLATES: Dict[str, NmapScanTemplate] = {
    "default": {
        # Commented values are default values and to not need to be
        # specified:
        # "nmap": "nmap",
        # "pings": "SE",
        # "scans": "SV",
        # "osdetect": True,
        # "traceroute": True,
        # "resolve": 1,
        # "verbosity": 2,
        # "ports": None,
        # "top_ports": None,
        "host_timeout": "15m",  # default value: None
        "script_timeout": "2m",  # default value: None
        "scripts_categories": ["default", "discovery", "auth"],  # default value: None
        "scripts_exclude": [
            "broadcast",
            "brute",
            "dos",
            "exploit",
            "external",
            "fuzzer",
            "intrusive",
        ],  # default value: None
        # "scripts_force": None,
        # "extra_options": None,
```

(continues on next page)

(continued from previous page)

```
    }
}
```

To create another template, the easiest is to copy, either using `.copy()` or using the `dict()` constructor, the `"default"` template; the following configuration entry creates an `"aggressive"` template that will run more scripts (including potentially dangerous ones) and have more permissive timeout values:

```
NMAP_SCAN_TEMPLATES["aggressive"] = dict(
    NMAP_SCAN_TEMPLATES["default"],
    host_timeout="30m",
    script_timeout="5m",
    scripts_categories=['default', 'discovery', 'auth', 'brute',
                        'exploit', 'intrusive'],
    scripts_exclude=['broadcast', 'external'],
)
```

It is possible to check the options a template will use by running the following command (the output has been modified, the command line is normally on one single line):

```
$ ivre runscans --output CommandLine
Command line to run a scan with template default
    nmap -A -PS -PE -sS -vv --host-timeout 15m --script-timeout 2m
         --script '(default or discovery or auth) and not (broadcast
         or brute or dos or exploit or external or fuzzer or intrusive)'

$ ivre runscans --output CommandLine --nmap-template aggressive
Command line to run a scan with template aggressive
    nmap -A -PS -PE -sS -vv --host-timeout 30m --script-timeout 5m
         --script '(default or discovery or auth or brute or exploit or
         intrusive) and not (broadcast or external)'
```

Masscan probes

IVRE can use the service fingerprint database from Nmap to find service and product names from Masscan results. For that, IVRE needs to know which probe (or "hello string") has been used. This depends on Masscan source code (compile-time) and options (run-time). You can adjust what IVRE will use per port (from the configuration) or globally (from the command-line option).

The default configuration value is based on the Masscan fork of the IVRE project.

```


# Based on IVRE's fork source code --- you may want to adapt these
# settings if you use another version of Masscan.
MASSCAN_PROBES = {
    "tcp": {
        53: "DNSVersionBindReqTCP",
        88: "Kerberos",
        104: "dicom",
        111: "RPCCheck",
        130: "NotesRPC",
        135: "DNSVersionBindReqTCP",
        256: "LDAPSearchReq",
        257: "LDAPSearchReq",
        389: "LDAPSearchReq",
        390: "LDAPSearchReq",
```

(continues on next page)

(continued from previous page)

```
        406: "SIPOptions",
        427: "NotesRPC",
        548: "afp",
        554: "RTSPRequest",
        1098: "JavaRMI",
        1099: "JavaRMI",
        1352: "NotesRPC",
        1433: "ms-sql-s",
        1702: "LDAPSearchReq",
        1972: "NotesRPC",
        2049: "RPCCheck",
        2345: "dicom",
        2375: "docker",
        2379: "docker",
        2380: "docker",
        2761: "dicom",
        2762: "dicom",
        3268: "LDAPSearchReq",
        3892: "LDAPSearchReq",
        4242: "dicom",
        5060: "SIPOptions",
        6000: "X11Probe",
        6001: "X11Probe",
        6002: "X11Probe",
        6003: "X11Probe",
        6004: "X11Probe",
        6005: "X11Probe",
        6006: "X11Probe",
        6007: "X11Probe",
        6008: "X11Probe",
        6009: "X11Probe",
        6010: "X11Probe",
        6011: "X11Probe",
        6012: "X11Probe",
        6013: "X11Probe",
        6014: "X11Probe",
        6015: "X11Probe",
        6016: "X11Probe",
        6017: "X11Probe",
        6018: "X11Probe",
        6019: "X11Probe",
        6379: "redis-server",
        7171: "NotesRPC",
        8081: "SIPOptions",
        8554: "RTSPRequest",
        8728: "NotesRPC",
        9001: "mongodb",
        11112: "dicom",
        11711: "LDAPSearchReq",
        22001: "NotesRPC",
        27017: "mongodb",
        31337: "SIPOptions",
        49153: "mongodb",
        50000: "DNSVersionBindReqTCP",
        50001: "DNSVersionBindReqTCP",
        50002: "DNSVersionBindReqTCP",
    },
```

(continues on next page)

(continued from previous page)

```
}
```

The `flow` purpose

The `flow` purpose has several specific configuration options, which may have important impacts on performances; here are the options and their default values:

```
# Dictionary that helps determine server ports of communications. Each entry
# is {proto: {port: proba}}. The when two ports are known, the port with the
# highest probability is used.
# When /usr/share/nmap/nmap-services is available, these probas are taken,
# otherwise /etc/services is used with proba=0.5 for each entry.
# KNOWN_PORTS entries have the highest priority.
# Example:
#  KNOWN_PORTS = {
#      "udp": {
#          9999: 1.0,
#          12345: 0.5,
#      },
#      "tcp": {
#          20202: 0.8,
#      },
#  }
KNOWN_PORTS: Dict[str, Dict[int, float]] = {}
# Enable the recording of appearance times for flows. Will slow down a
# bit the insertion rate
FLOW_TIME = True
# Precision (in seconds) to use when recording times when flows appear
FLOW_TIME_PRECISION = 3600
# When recording flow times, record the whole range from start_time to end_time
# This option is experimental and possibly useless in practice
FLOW_TIME_FULL_RANGE = True
# When recording flow times, represents the beginning of the first timeslot
# as a Unix timestamp shifted to local time.
# 0 means that the first timeslot starts at 1970-01-01 00:00 (Local time).
FLOW_TIME_BASE = 0
# Store high level protocols metadata in flows. It may take much more space.
FLOW_STORE_METADATA = True
```

The `data` purpose

The URLs used to get IP address databases are set in the dictionary `IPDATA_URLS`:

```
IPDATA_URLS = {
    # None has a special meaning:
    # https://download.maxmind.com/app/geoip_download?edition_id=XXX&suffix=XXX&
↪license_key=XXX
    #
    # You can use this value for the GeoLite2-* files (and set
    # MAXMIND_LICENSE_KEY below) to download files from MaxMind
    # instead of ivre.rocks directly. Maxmind license keys are free
    # and can be obtained from <https://www.maxmind.com/>
    "GeoLite2-City.tar.gz": "https://ivre.rocks/data/geolite/GeoLite2-City.tar.gz",
```

(continues on next page)

(continued from previous page)

```
    "GeoLite2-City-CSV.zip": "https://ivre.rocks/data/geolite/GeoLite2-City-CSV.zip",
    "GeoLite2-Country.tar.gz": "https://ivre.rocks/data/geolite/GeoLite2-Country.tar.
↪gz",
    "GeoLite2-Country-CSV.zip": "https://ivre.rocks/data/geolite/GeoLite2-Country-CSV.
↪zip",
    "GeoLite2-ASN.tar.gz": "https://ivre.rocks/data/geolite/GeoLite2-ASN.tar.gz",
    "GeoLite2-ASN-CSV.zip": "https://ivre.rocks/data/geolite/GeoLite2-ASN-CSV.zip",
    # For other files, None has a special meaning "do not
    # download". The following file can be computed based the
    # GeoLite2-* files using `ivre ipdata --import-all`. You should do
    # that if you get your files from Maxmind.
    "GeoLite2-dumps.tar.gz": "https://ivre.rocks/data/geolite/GeoLite2-dumps.tar.gz",
    "iso3166.csv": "https://dev.maxmind.com/csv-files/codes/iso3166.csv",
    # This one is not from maxmind -- see https://thyme.apnic.net/
    "BGP.raw": "https://thyme.apnic.net/current/data-raw-table",
}
MAXMIND_LICENSE_KEY = None
```

GeoIP uses a locale to report country, region and city names. The locale to use is set in `GEOIP_LANG` and defaults to `"en"`.

Web server

Paths

Two variables (`WEB_STATIC_PATH` and `WEB_DOKU_PATH`) are used for the Web application; see *Paths and commands*.

Notepad

If Dokuwiki (or another web application for notes) is used, the variable `WEB_NOTES_BASE` should be set to the URL path to access the notes (`#IP#` will be replaced with the IP address). This variable defaults to `/dokuwiki/#IP#`.

If you use Dokuwiki, you also want to set:

```
WEB_GET_NOTEPAD_PAGES = "localdokuwiki"
```

Or:

```
WEB_GET_NOTEPAD_PAGES = ("localdokuwiki", ("/path/to/dokuwiki/data/pages",))
```

The second option is needed if the path to Dokuwiki pages is different from the default `"/var/lib/dokuwiki/data/pages"`.

If you use Mediawiki, you need to set

```
WEB_GET_NOTEPAD_PAGES = ("mediawiki", ("server", "username", "password",
                                       "dbname", "base"))
```

Anti-CSRF

As an anti-CSRF option, IVRE will check the `Referer:` header of the requests to any dynamic URLs (under `/cgi/`). Normally (when `ivre httpd` is used or when the WSGI application is exposed directly, IVRE will figure

out the allowed referrer URLs alone; under certain circumstances however (e.g., when a reverse-proxy is used, or when the IVRE dynamic URLs are used by another Web application), this is not possible. In this case, the variable `WEB_ALLOWED_REFERERS` should be set to a list or URLs that are allowed to trigger Web accesses to the IVRE application; for example:

```
WEB_ALLOWED_REFERERS = [
    'http://reverse-proxy.local/ivre',
    'http://reverse-proxy.local/ivre/',
    'http://reverse-proxy.local/ivre/index.html',
    'http://reverse-proxy.local/ivre/report.html',
    'http://reverse-proxy.local/ivre/upload.html',
    'http://reverse-proxy.local/ivre/compare.html',
    'http://reverse-proxy.local/ivre/flow.html'
]
```

Authentication and ACLs

If you want to use an authentication in IVRE, you have to configure your Web server (e.g., Apache or Nginx) to do so and set the environment variable `REMOTE_USER` to the username.

If you want to do some authorization based on the authentication, you can do so by setting a couple of variables; by default, ACL is disabled, and everyone (that can access the `/cgi/` URLs) can access to all the results:

```
WEB_DEFAULT_INIT_QUERY = None
WEB_INIT_QUERIES = {}
```

In the following, we call and "access filter" either the special value `None` which means "unrestricted", or a string describing a filter to apply before performing any query. The strings can be:

- "full": unrestricted.
- "noaccess": no result will be returned to the user.
- "category:[category name]": the user will only have access to results within `[category name]` category.
- "source:[source name]": the user will only have access to results within `[source name]` source.

`WEB_DEFAULT_INIT_QUERY` should be set to an "access filter" that will apply when the current user does not match any user in `WEB_INIT_QUERIES`.

Here is a simple example, where user `admin` has full access, user `admin-site-a` has access to all results in category `site-a`, and user `admin-scanner-a` has access to all results with source `scanner-a`:

```
WEB_DEFAULT_INIT_QUERY = 'noaccess'
WEB_INIT_QUERIES = {
    'admin': 'full',
    'admin-site-a': 'category:site-a',
    'admin-scanner-a': 'source:scanner-a',
}
```

If you user Kerberos authentication (or if you have `@` in your usernames that provide some kind of "realms", you can use them; in the following example, any user in the `admin.sitea` realm has access to all results in category `site-a`:

```
WEB_DEFAULT_INIT_QUERY = 'noaccess'
WEB_INIT_QUERIES = {
    '@admin.sitea': 'category:site-a',
}
```

Misc

IVRE handles DNS blacklist (as defined in the RFC 5782) answers, for domains listed in the set `DNS_BLACKLIST_DOMAINS`. By default, it is defined as:

```
# Domains used for DNS blacklists (RFC 5782)
DNS_BLACKLIST_DOMAINS = set(
    [
        "blacklist.woody.ch",
        "zen.spamhaus.org",
    ]
)
```

To add a domain, just add in your configuration file:

```
DNS_BLACKLIST_DOMAIN.add("dnsbl.example.com")
```

Or, to add several entries at once:

```
DNS_BLACKLIST_DOMAIN.update([
    "dnsbl1.example.com",
    "dnsbl2.example.com",
])
```

6.2.3 Fast install & first run

This file describes the steps to install IVRE, run the first scans and add the results to the database with all components (scanner, web server, database server) on the same (Debian or Ubuntu) machine.

You might also want to adapt it to your needs, architecture, etc.

For another way to run IVRE easily (probably even more easily), see *Docker*.

Install MongoDB

Follow the instructions from the MongoDB project, for example:

- MongoDB on Debian
- MongoDB on Ubuntu

Install IVRE

```
$ sudo apt -y --no-install-recommends install python3-pymongo \
>   python3-cryptography python3-bottle python3-openssl apache2 \
>   libapache2-mod-wsgi-py3 dokuwiki
$ git clone https://github.com/ivre/ivre
$ cd ivre
$ python3 setup.py build
$ sudo python3 setup.py install
```

Setup

```
$ sudo -s
# cd /var/www/html ## or depending on your version /var/www
# rm index.html
# ln -s /usr/local/share/ivre/web/static/* .
# cd /var/lib/dokuwiki/data/pages
# ln -s /usr/local/share/ivre/dokuwiki/doc
# cd /var/lib/dokuwiki/data/media
# ln -s /usr/local/share/ivre/dokuwiki/media/logo.png
# ln -s /usr/local/share/ivre/dokuwiki/media/doc
# cd /usr/share/dokuwiki
# patch -p0 < /usr/local/share/ivre/patches/dokuwiki/backlinks-20200729.patch
# cd /etc/apache2/mods-enabled
# for m in rewrite.load wsgi.conf wsgi.load ; do
>   [ -L $m ] || ln -s ../mods-available/$m ; done
# cd ../
# echo 'Alias /cgi "/usr/local/share/ivre/web/wsgi/app.wsgi"' > conf-enabled/ivre.conf
# echo '<Location /cgi>' >> conf-enabled/ivre.conf
# echo 'SetHandler wsgi-script' >> conf-enabled/ivre.conf
# echo 'Options +ExecCGI' >> conf-enabled/ivre.conf
# echo 'Require all granted' >> conf-enabled/ivre.conf
# echo '</Location>' >> conf-enabled/ivre.conf
# sed -i 's/^\(\s*\)#Rewrite/\1Rewrite/' /etc/dokuwiki/apache.conf
# echo 'WEB_GET_NOTEPAD_PAGES = "localdokuwiki"' >> /etc/ivre.conf
# service apache2 reload  ## or start
# exit
```

Open a web browser and visit http://localhost/. IVRE Web UI should show up, with no result of course. Click the HELP button to check if everything works.

Database init, data download & importation

```
$ yes | ivre ipinfo --init
$ yes | ivre scancli --init
$ yes | ivre view --init
$ yes | ivre flowcli --init
$ yes | sudo ivre runscansagentdb --init
$ sudo ivre ipdata --download
```

Run a first scan

Against 1k (routable) IP addresses, with a single nmap process:

```
$ sudo ivre runscans --routable --limit 1000
```

Go have some coffees and/or beers (remember that according to the traveler's theorem, for any time of the day, there exists a time zone in which it is OK to drink).

When the command has terminated, import the results and create a view:

```
$ ivre scan2db -c ROUTABLE,ROUTABLE-CAMPAIGN-001 -s MySource -r \
>              scans/ROUTABLE/up
$ ivre db2view nmap
```

The `-c` argument adds categories to the scan results. Categories are arbitrary names used to filter results. In this example, the values are `ROUTABLE`, meaning the results came out while scanning the entire reachable address space (as opposed to while scanning a specific network, AS or country, for example), and `ROUTABLE-CAMPAIGN-001`, which is the name I have chosen to mark this particular scan campaign.

The `-s` argument adds a name for the source of the scan. Here again, it is an arbitrary name you can use to unambiguously specify the network access used to run the scan. This can be used later to highlight result differences depending on where the scans are run from.

Go back to the Web UI and browse your first scan results!

Some remarks

There is no tool (for now) to automatically import scan results to the database. It is your job to do so, according to your settings.

If you run very large scans (particularly against random hosts on the Internet), do NOT use the default `--output=XML` option. Rather, go for the `--output=XMLFork`. This will fork one nmap process per IP to scan, and is (sadly) much more reliable.

Another way to run scans efficiently is to use an agent and the `ivre runscansagent` command.

6.2.4 Docker

Versions

The images published on Docker hub are built from the current `master` repository branch (tag `latest`, will be used by default) and from the current release (tag `vX.Y.Z`, use `ivre/<imagename>:vX.Y.Z` to use it).

Using docker compose

The easiest way, just run:

```
$ docker compose up
```

The containers should now be running, with the TCP port 80 of your host redirected to the `ivreweb` container.

To get a shell with the CLI tools and Python API, attach to the `ivreclient` container:

```
$ docker attach ivreclient
root@fd983ba5e6fd:/#
```

You can detach from the container (without stopping it) by using `C-p C-q` and attach to it again later with the same `docker attach ivreclient` command.

To initialize the database and start playing with IVRE, you need to enter some commands described in the *related section below*.

Using Vagrant

If you already manage your Docker containers using Vagrant, you can use it to run the containers.

With the `Vagrantfile` as it is provided, the TCP port 80 of your host will be used, so you need either to make sure it is not already in use, or to modify the `Vagrantfile` after the `cp` step in the instructions below to use another port.

To use the `Vagrantfile` located in the `docker/` directory of the source tree (or the `[PREFIX]/share/ivre/docker/` directory when IVRE has been installed), run (from the folder where you want to store your data):

```
$ mkdir -m 1777 var_lib_mongodb ivre-share dokuwiki_data
  # For people using SELinux enforced, you need to run
$ sudo chcon -Rt svirt_sandbox_file_t var_lib_mongodb ivre-share dokuwiki_data

$ cp [path to ivre source]/docker/Vagrantfile .
$ vagrant up --no-parallel
```

The `--no-parallel` option prevents Vagrant from starting the `ivreuwsgi` container before the `ivredb` is ready.

To access the `ivreclient` container, see the *Using Docker Compose* since it is similar.

Build the images

By default, the images will be downloaded from the Docker Hub. But you also can build the images from the provided `Dockerfile`s. For that, from the `docker/` directory, run:

```
$ docker pull debian:12
$ for img in base client agent web web-doku web-uwsgi ; do
> docker build -t "ivre/$img" "$img"
> done
```

This might take a long time.

Alternative builds for the base image

Local archive

It is also possible to build the `ivre/base` image without fetching the *tarball* from GitHub, by creating it locally and using the `base-local` directory instead of `base`. From the repository root, run:

```
$ git archive --format=tar --prefix=ivre/ HEAD -o docker/base-local/ivre.tar
$ tmp=`mktemp | sed 's#^/##'`; python setup.py --version | tr -d '\n' > "/$tmp"
$ tar rf docker/base-local/ivre.tar --transform="s#$tmp#ivre/ivre/VERSION#" /$tmp
$ rm "/$tmp"
$ docker pull debian:12
$ docker build -t ivre/base docker/base-local
```

Using pip

Another way to create the `ivre/base` image is to use pip. From the `docker/` directory, run:

```
$ docker pull debian:12
$ docker build -t ivre/base base-pip
```

Initialization

Attach to the `ivreclient` container and run the initialization commands:

```
user@host:~$ docker attach ivreclient
root@ivreclient:/# yes | ivre ipinfo --init
root@ivreclient:/# yes | ivre scancli --init
root@ivreclient:/# yes | ivre view --init
root@ivreclient:/# yes | ivre flowcli --init
root@ivreclient:/# yes | ivre runscansagentdb --init
root@ivreclient:/# ivre ipdata --download
```

Then we can integrate the Nmap results to the database `nmap` database and create a `view` from it:

```
root@ivreclient:/# ivre scan2db -r -s MySource -c MyCategory /ivre-share
root@ivreclient:/# ivre db2view nmap
```

You can then detach from the container (`C-p C-q`).

```
root@ivreclient:/# exit
```

You can start the container again later by issuing:

```
$ docker start -i ivreclient
root@ivreclient:/#
```

If you do not want to exit the shell but only detach from it, use `C-p C-q`. You can attach to it again later by issuing `docker attach ivreclient`.

6.2.5 Agents

IVRE agent may be run in an environment not totally controlled (e.g., during a pentest, on a machine you have just owned and want to use to do some network recon without installing IVRE), since it has a reduced number of dependencies.

IVRE agent only requires `nmap` (of course), `screen` and `rsync` (plus `/bin/sh` and basic shell utils, including `grep`).

Set-up

On the "master", install IVRE following the *Installation guidelines*. Install also `screen`, `tmux` or `nohup` if you want to be able to "detach" from the `agent` script (which is not a daemon).

On the "worker(s)", the `agent` script must be deployed, together with `nmap`, and `rsync`.

Run the worker(s)

The computer running IVRE (the "master") needs to be able to access via `rsync` the data directory of the agents (to add targets and to retrieve results): this is not an issue if you are running the agent and IVRE itself on the same machine. If you are running IVRE and the agent on two different hosts (and, except for simple or testing configurations, you should do that), you have to run `sshd` or `rsyncd` on the agent host, or share the agent files (using NFS, SMB or whatever the IVRE side can mount).

First, `mkdir` & `cd` to the directory you want to use as your agent data directory.

Make sure the needed binaries are in the `PATH` environment variable (including `nmap`). Generate the `agent` script, on a computer with IVRE installed, by running `ivre runscans --output Agent > agent; chmod +x agent` , adapt if needed the variables at the beginning of the script, particularly `THREADS`.

By default, the `default` template is used. You can generate agents using other scan templates using `--nmap-template [template name]`.

Then just run the `agent` script.

When the scan is over, to stop the agent, type `C-c` or kill the parent `agent` process.

Run the master

You need to make sure the user running `ivre runscansagent` or `ivre runscansagentdb` on the "master" can access (without password) to the agents data directories.

When the agents are all ready, you have two options, using `ivre runscansagent` or `ivre runscansagentdb`. In both cases, scan options are the same than with `ivre runscans`.

The first one (`ivre runscansagent`) is the "old-school" version: it will not allow to dynamically add or remove agents, and will fetch the results under `./agentsdata/output` directory, you have to import the results by yourself.

On the other hand, the second one (`ivre runscansagentdb`) will use the DB to manage the agents, but is still experimental.

runscansagent, the "old-school" one

You have to specify the agent(s) data directory. For example, run:

```
$ ivre runscansagent --routable --limit 1000 \
>     agenthost1:/path/to/agent/dir      \
>     agenthost2:/path/to/agent/dir      \
```

You can now import the results as if you had run the "regular" `ivre runscans` program to scan locally. The results are stored under `agentsdata/output/`

runscansagentdb, the "modern" (but probably broken) one

Please note that it is important to run all the `ivre runscansagentdb` from the same host (the "master", which does not need to be the same host than the database server), since it relies on local directories.

First, let's create a master and add the agent(s):

```
$ ivre runscansagentdb --add-local-master
$ ivre runscansagentdb --source MySource --add-agent \
>     agenthost1:/path/to/agent/dir \
>     agenthost2:/path/to/agent/dir
```

Let's check it's OK:

```
$ ivre runscansagentdb --list-agents
agent:
  - id: 543bfc8a312f915728f1709b
  - source name: MySource
  - remote host: agenthost1
  - remote path: /path/to/agent/dir/
  - local path: /var/lib/ivre/master/sbOist
  - rsync command: rsync
```

(continues on next page)

(continued from previous page)

```
  - current scan: None
  - currently synced: True
  - max waiting targets: 60
  - waiting targets: 0
  - can receive: 60
agent:
  - id: 543bfc8a312f915728f1709c
  - source name: MySource
  - remote host: agenthost2
  - remote path: /path/to/agent/dir/
  - local path: /var/lib/ivre/master/m2584z
  - rsync command: rsync
  - current scan: None
  - currently synced: True
  - max waiting targets: 60
  - waiting targets: 0
  - can receive: 60
```

Now we can add a scan, and assign the (available) agents to that scan:

```
$ ivre runscansagentdb --assign-free-agents --routable --limit 1000
```

And see if it works:

```
$ ivre runscansagentdb --list-scans
scan:
  - id: 543bfcbf312f9158d6caeadf
  - categories:
    - ROUTABLE
  - targets added: 0
  - results fetched: 0
  - total targets to add: 1000
  - available targets: 2712693508
  - internal state: (2174385484, 551641673, 387527645, 0)
  - agents:
    - 543bfc8a312f915728f1709b
    - 543bfc8a312f915728f1709c
```

For now, nothing has been sent to the agents. To really start the process, run:

```
$ ivre runscansagentdb --daemon
```

After some time, the first results get imported in the database (`READING [...]`, `HOST STORED: [...]`, `SCAN STORED: [...]`). You can stop the daemon at any time by `(p)kill`-ing it (using `CTRL+c` will do).

When all the targets have been sent to an agent, the agents get disassociated from the scan so that another scan can use them. You can check the scan evolution by issuing `ivre runscansagentdb --list-scans`.

6.3 Usage

6.3.1 Some use cases

As a *framework*, IVRE has several possible use cases. Of course, you probably want to use only parts of what IVRE can do.

Your own Shodan / ZoomEye / Censys / Binaryedgeio / whatever

You can use IVRE as a private (or even public, if you want) alternative to Shodan (or any other similar service).

The main difference with public services is that you will have the control of your data. You can scan whatever you want (your private networks, public networks, a specific country or Autonomous System, the whole Internet, etc.), for any port or protocol. You can run any query on your data; no-one has to know what you are really looking for.

Of course, this require more work than just using an existing public service, but the benefits are huge!

IVRE does not come with a scanner, and takes advantage of Nmap, Masscan and Zgrab / Zgrab2. Depending on your use case, you can choose one or use both (IVRE will happily merge the results for you). Remember to use the `-oX` option (which works with both Nmap and Masscan) or `-o` for Zgrab2, as IVRE needs the XML output file for Nmap and Masscan, and JSON for Zgrab2.

You can use `ivre runscans`, `ivre runscansagent` or `ivre runscansagentdb` to run Nmap scans against wide targets (more) easily.

You will then store the results from the XML or JSON output files into IVRE database using `ivre scan2db`.

Finally, use `ivre db2view nmap` to create a `view` (see *Purposes*) that you can explore with the *Web User Interface*.

See *IVRE with Kibana* if you want to use Kibana to explore your scan results.

Your own Passive DNS service

Passive DNS services log DNS answers into a database and let you run queries against them.

IVRE uses its Zeek script `passiverecon` to, among others, log DNS answers. They are stored in the `passive` purpose (see *Purposes*) via `ivre passiverecon2db` CLI tool as `DNS_ANSWER` records.

They can be queried using `ivre iphost` CLI tool, as in the following example (the results come from a PCAP file used in IVRE's *Tests*):

```
$ ivre iphost ipv4.icanhazip.com
ipv4.icanhazip.com A 216.69.252.101 (109.0.66.10:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)
ipv4.icanhazip.com A 216.69.252.100 (109.0.66.10:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)
ipv4.icanhazip.com A 216.69.252.100 (109.0.66.20:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)
ipv4.icanhazip.com A 216.69.252.101 (109.0.66.20:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)

$ ivre iphost 216.69.252.101
ipv4.icanhazip.com A 216.69.252.101 (109.0.66.10:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)
ipv4.icanhazip.com A 216.69.252.101 (109.0.66.20:53, 1 time, 2014-01-02 09:37:57.
↪197000 - 2014-01-02 09:37:57.197000)
```

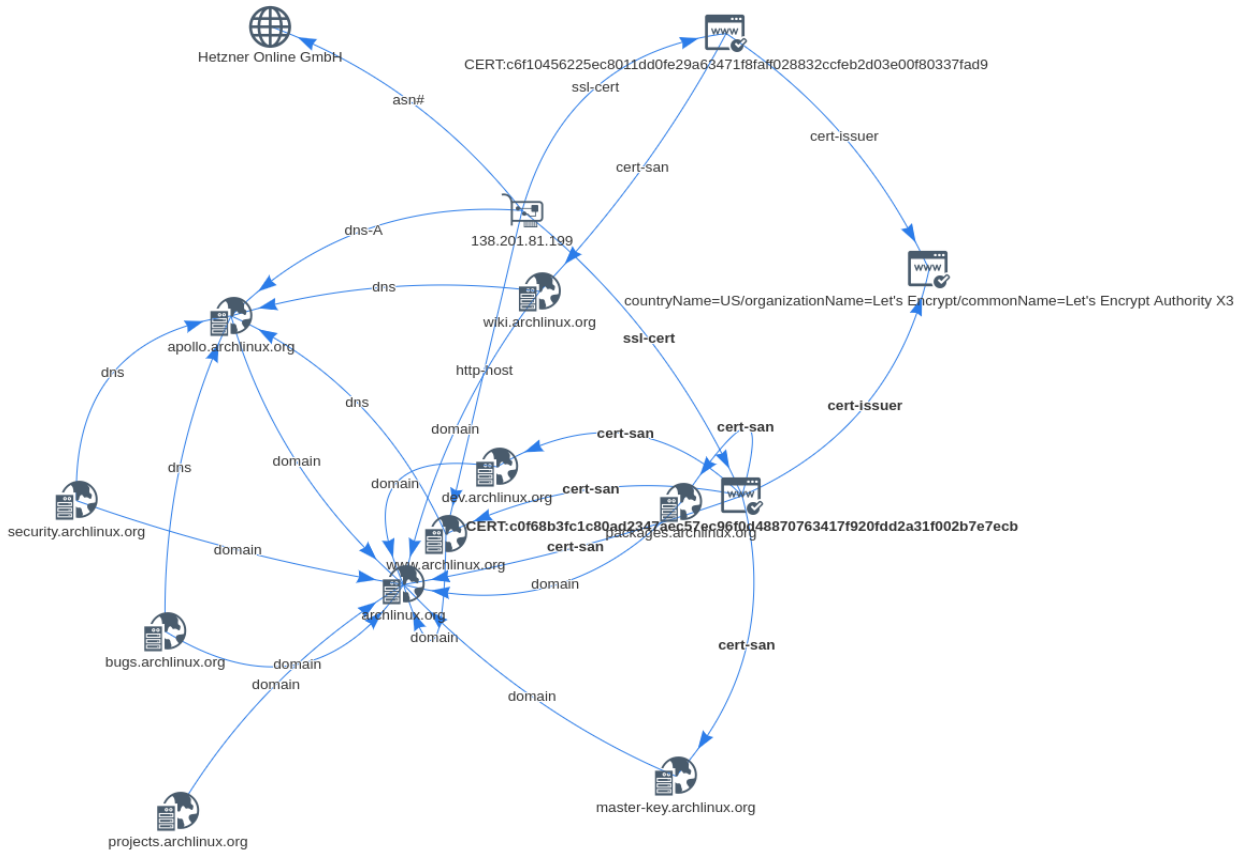
To see an interactive session of IVRE using passive data (including DNS answers), have a look at *Passive network analysis*.

YETI plugin

Yeti is a platform meant to organize observables, indicators of compromise, TTPs, and knowledge on threats in a single, unified repository.

It comes with an “analytics” plugin that uses IVRE’s data to create links between IP addresses, hostnames, certificates, etc.

To learn more about this plugin, have a look at its documentation.

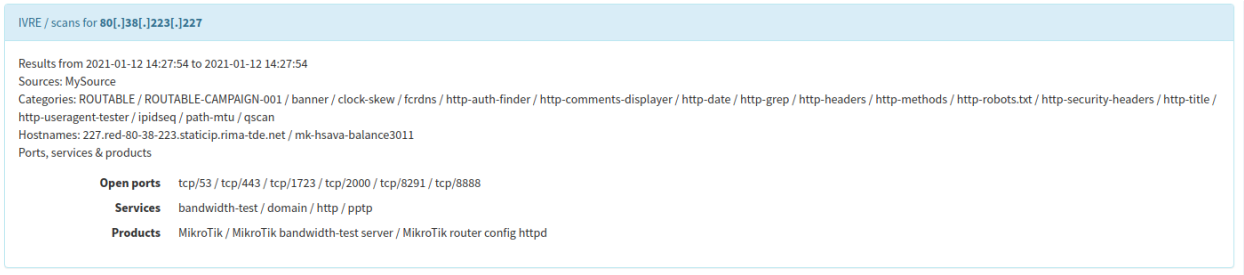


Cortex analyzer

Cortex is a tool to analyze observables for SOCs, CSIRTs and security researchers; it integrates well with TheHive.

It comes with an “Analyzer” that uses IVRE’s data to report intelligence about Autonomous Systems, certificates, domain and host names, IP addresses, networks, open ports, etc.

To learn more about this analyzer, have a look at its documentation.


IVRE / scans for **80[.]38[.]223[.]227**

Results from 2021-01-12 14:27:54 to 2021-01-12 14:27:54
Sources: MySource
Categories: ROUTABLE / ROUTABLE-CAMPAIGN-001 / banner / clock-skew / fcrdns / http-auth-finder / http-comments-displayer / http-date / http-grep / http-headers / http-methods / http-robots.txt / http-security-headers / http-title / http-useragent-tester / ipidseq / path-mtu / qscan
Hostnames: 227.red-80-38-223.staticip.rima-tde.net / mk-hsava-balance3011
Ports, services & products

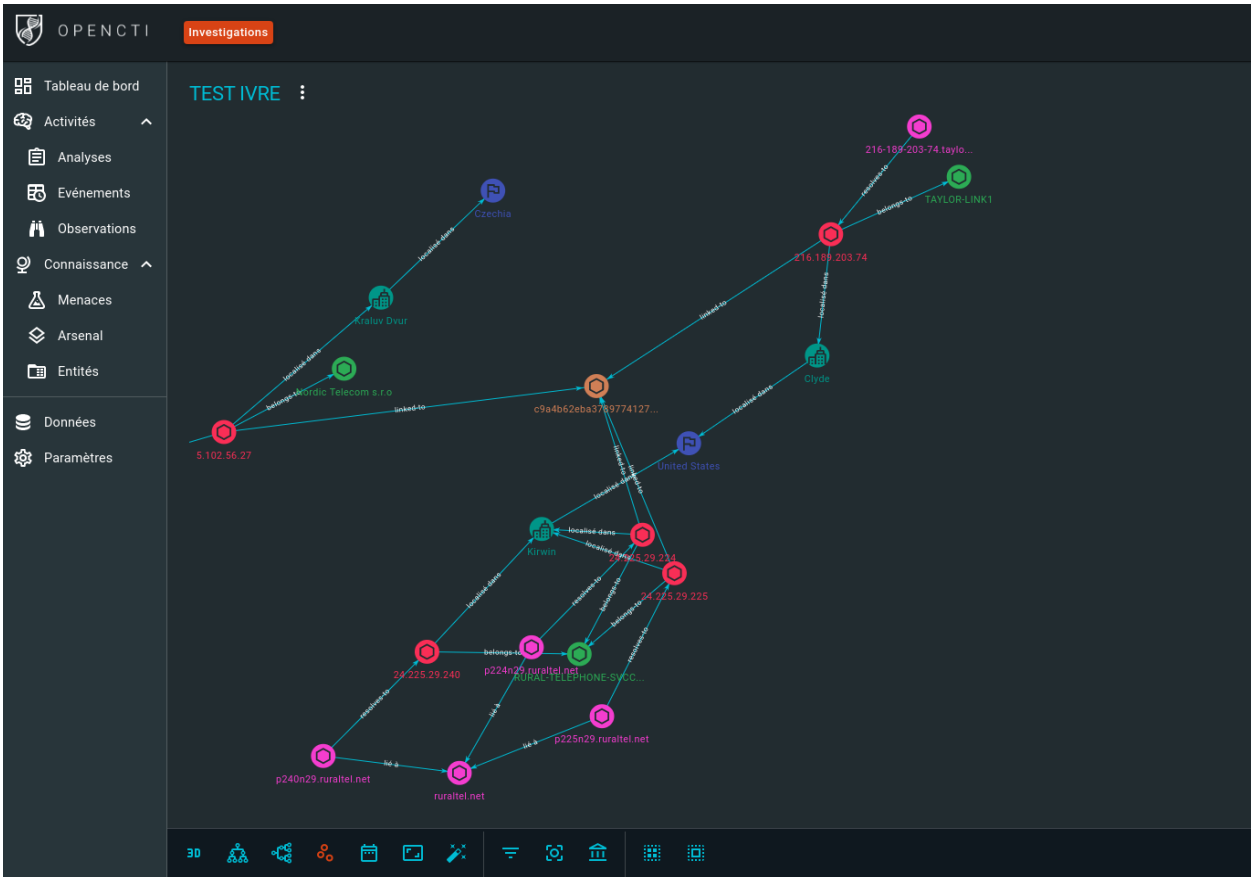
Open ports	tcp/53 / tcp/443 / tcp/1723 / tcp/2000 / tcp/8291 / tcp/8888
Services	bandwidth-test / domain / http / pptp
Products	MikroTik / MikroTik bandwidth-test server / MikroTik router config httpd

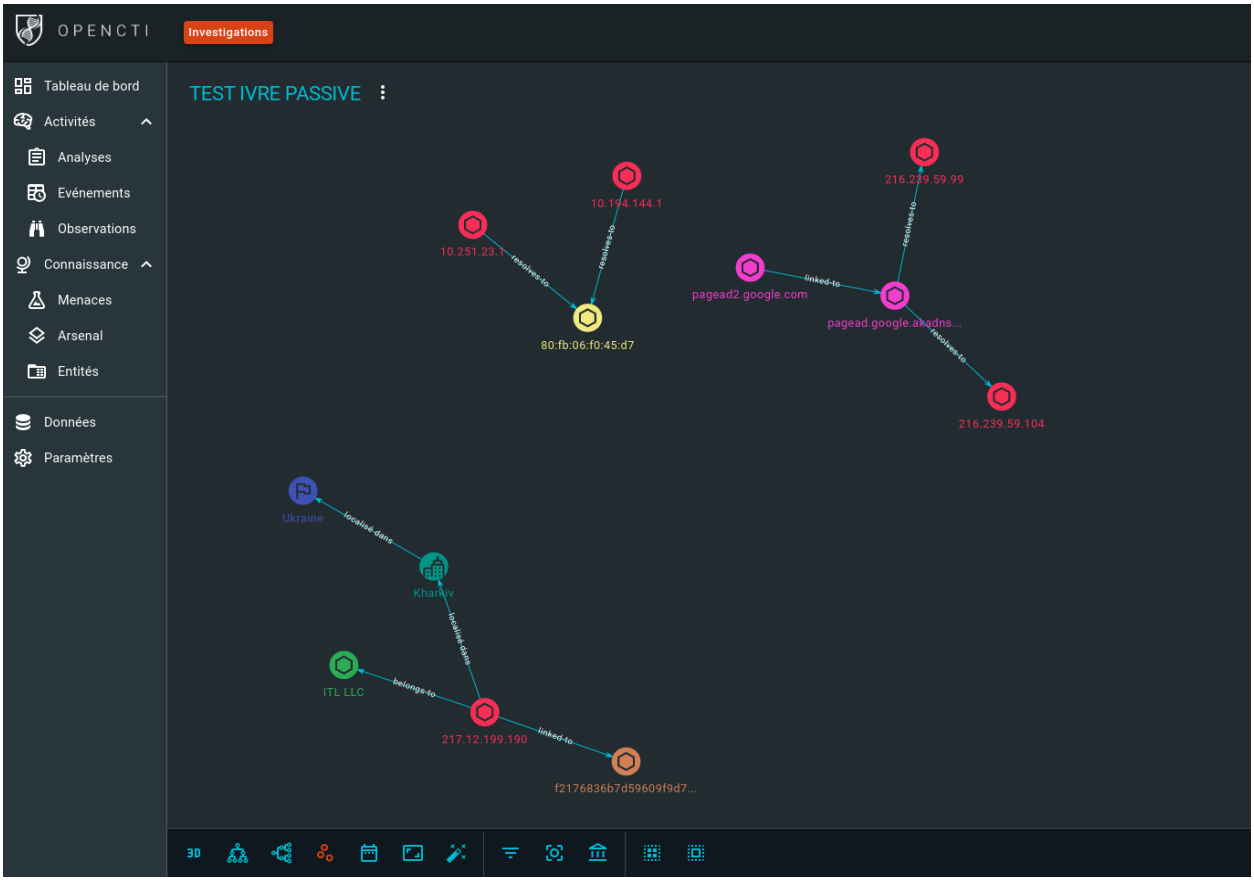
OpenCTI connector

OpenCTI is an open-source cyber threat intelligence (CTI) platform.

It comes with an “internal enrichment connector” that uses IVRE’s data to create links between IP addresses, MAC addresses, hostnames, certificates, AS numbers and locations.

To learn more about this connector, have a look at its documentation.



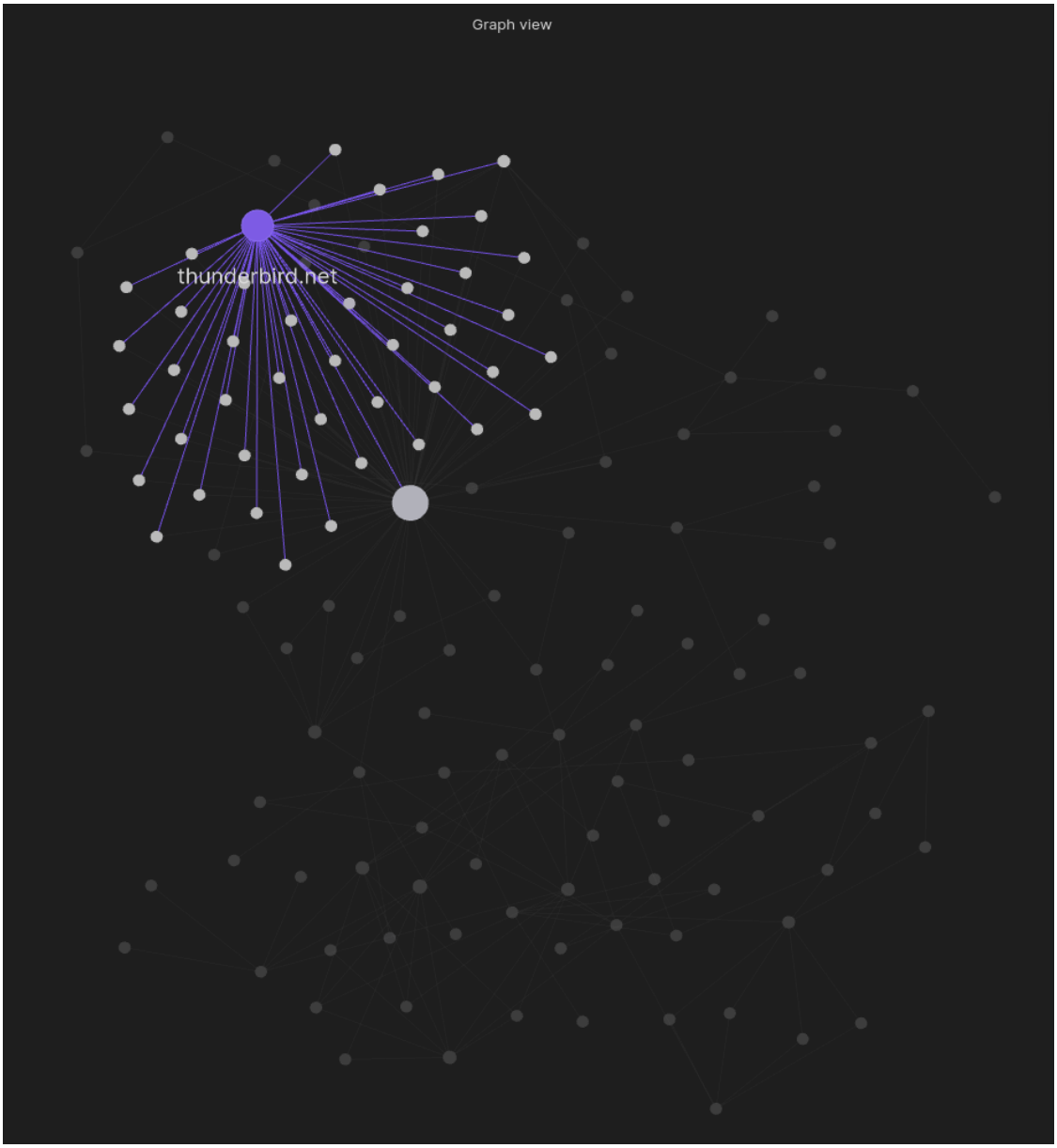


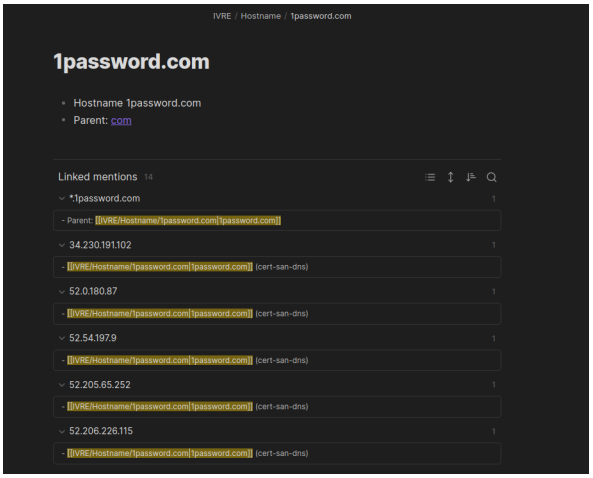
Obsidian plugin

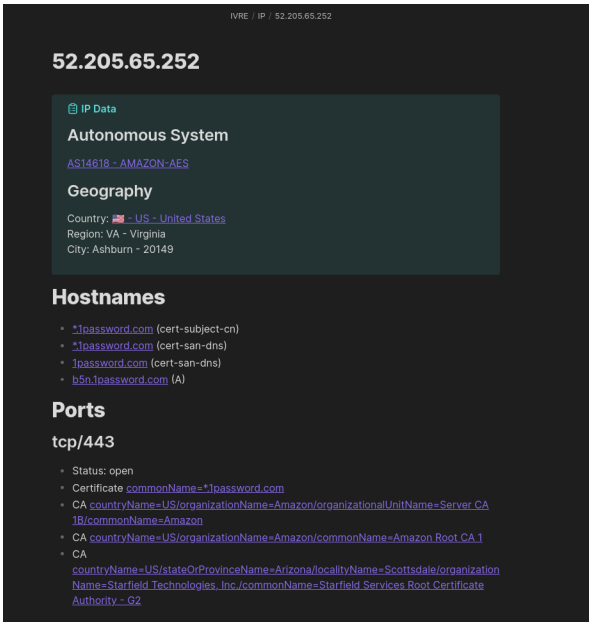
Obsidian is a knowledge base and note-taking application that relies on Markdown files.

A community plugin exists that uses IVRE's data to create notes based on IVRE's data that provides context to your notes related to pentest or red team engagements, bug bounty hunting, cyber threat intelligence, etc.

See the plugin's README.


Graph view
thunderbird.net


IVRE / Hostname / 1password.com
1password.com
Hostname 1password.com
Parent: com
Linked mentions 14
*.1password.com 1
- Parent: [[IVRE/Hostname/*1password.com|1password.com]]
34.230.191.102 1
- [[IVRE/Hostname/*1password.com|1password.com]] (cert-san-dns)
52.0.180.87 1
- [[IVRE/Hostname/*1password.com|1password.com]] (cert-san-dns)
52.54.197.9 1
- [[IVRE/Hostname/*1password.com|1password.com]] (cert-san-dns)
52.205.65.252 1
- [[IVRE/Hostname/*1password.com|1password.com]] (cert-san-dns)
52.206.226.115 1
- [[IVRE/Hostname/*1password.com|1password.com]] (cert-san-dns)



Blog posts and other resources

The author's blog has some IVRE-related blog posts that might be useful.

Here is a list of other blog posts about or around IVRE:

- External attack surface monitoring:
  - Building an Automated Perimeter Scanning System with Open Source Tools - NMAP, IVRE and Netbox
  - Re-discover your company network with Ivre
- Scan the hosts that hit your honeypots, and exploit the results!
  - Who's Attacking Me?
  - Three Honeypots and a Month After
- Scanning SAP Services:
  - gelim/nmap-erpscan on Github
  - SAP Services detection via nmap probes
  - SAP Dispatcher Security
- IVRE tests & reviews:
  - IVRE
  - IVRE! Drunk Frenchman Port Scanner Framework!
  - Visualizing Scans Part 1: IVRE
- Spanish:
  - Reconocimiento de redes con IVRE

You have found (or written) a document that might help other use IVRE or decide if they need it? Please let us know: open an issue or *Contact* us so that we can add a link here!

6.3.2 Active recon

Scanning

With Nmap, Masscan or Zgrab2

You can use network scanners directly:

- Nmap
- Masscan
- Zgrab2 (`http` and `jarm` commands)
- ZDNS
- Nuclei
- httpx
- tlsx
- dnsx
- Dismap

IVRE can insert XML output files for Nmap and Masscan, and JSON output files for the other tools, using the command line tool `ivre scan2db`.

You can insert scan results from different tools, then use `ivre db2view nmap` to merge results from different scans and create a view you can explore with the the *Web User Interface*, the `ivre view` command line tool or the Python API (`ivre.db.db.view.*`).

With IVRE

Masscan does not provide results as complete as Nmap, when using the "interesting" options (for example, `-vv -A`) or scripts. That being said, Nmap (with such "interesting" options) cannot run efficiently against huge networks.

The `ivre runscans` tool can run one Nmap process per target (option `--output=XMLFork`). This should be less efficient in theory, because Nmap supposedly knows better how to handle the host and network resources, but in practice it is much more efficient. You can adjust how many Nmap processes you want to run in parallel using the `--processes N` option.

Another advantage of using `ivre runscans --output=XMLFork` over using Nmap directly is that `ivre runscans` produces output files as soon as each host has been scanned (in the `scans/*/up` directory).

Here is a simple example:

```
$ sudo ivre runscans --routable --limit 1000 --output=XMLFork
```

This will run a standard scan against 1000 random hosts on the Internet by running 30 nmap processes in parallel. See the output of `ivre help runscans` if you want to do something else.

When it's over, to import the results in the database and create a view from them, run (`ROUTABLE-001` is the category name, and `MySource` is the source name, usually referencing the machine used to run the scan):

```
$ ivre scan2db -c ROUTABLE-001 -s MySource -r scans/ROUTABLE/up
$ ivre db2view nmap
```

Enjoying the results

You have several options, depending on what you want to do:

- Command line interfaces: the `ivre scancli` tool.
- Python API: use the `db.nmap` object of the `ivre.db` module.
- Web API: `/cgi/scans`.

If you want to combine several tools, for example Masscan and Nuclei results, you need to use a view: run `ivre db2view nmap` to create or update a view from the scan data, that can then be accessed by the `view` purpose (see *Purposes*), which includes the *Web User Interface*.

CLI

To get all the hosts with the port 22 open:

```
$ ivre scancli --port 22
```

See the output of `ivre help scancli`.

Python module

To use the Python module, run for example:

```
$ python
>>> from ivre.db import db
>>> db.nmap.get(db.nmap.flt_empty)[0]
```

For more, run `help(db.nmap)` from the Python shell.

6.3.3 Passive

With Zeek

You need to run Zeek (formerly known as Bro), version 3.0 minimum (tested with 3.0 and 3.1) with the option `-b` and the location of the `passiverecon/bare.zeek` file. If you want to run it on the `eth0` interface, for example, run (replace `/usr/share/ivre` by the appropriate location; use `python -c 'import ivre.config; print(ivre.config.guess_prefix())'` if you cannot find it):

```
$ mkdir logs
$ sudo LOG_PATH=logs/passiverecon \
>   zeek -b /usr/share/ivre/zeek/ivre/passiverecon/bare.zeek -C -i eth0
```

If you want to run it on the `capture` file (`capture` needs to a PCAP file), run:

```
$ mkdir logs
$ LOG_PATH=logs/passiverecon \
>   zeek -b /usr/share/ivre/zeek/ivre/passiverecon/bare.zeek -r capture
```

This will produce log files in the `logs` directory. You need to run a `ivre passivereconworker` to process these files. You can try:

```
$ ivre passivereconworker --directory=logs
```

This program will not stop by itself. You can `kill` it, it will stop gently (as soon as it has finished to process the current file).

You can also send the data from `zeek` to the database without using intermediate files:

```
$ zeek -b /usr/share/ivre/zeek/ivre/passiverecon/bare.zeek [option] \
>   | ivre passiverecon2db
```

With p0f

You need to install p0f v3, and use it with the option `-o` to produce an output file. Then, provide that output file to `ivre p0f2db`.

For now, only `syn` and `syn+ack` modes are supported.

Enjoying the results

You have several options, depending on what you want to do:

- Command line interfaces (see also *Passive network analysis* in the screenshots gallery):
  - `ivre ipinfo` tool, for any passive data.
  - `ivre iphost` tool, for Passive DNS data (see *Your own Passive DNS service*).
- Python API: use the `db.passive` object of the `ivre.db` module.
- Web interface:
  - Using `ivre db2view`, you can create or update a view with passive data, that can then be accessed by the `view` purpose (see *Purposes*), which includes the *Web User Interface*.

CLI

To show everything stored about an IP address or a network:

```
$ ivre ipinfo 1.2.3.4
$ ivre ipinfo 1.2.3.0/24
```

See the output of `ivre help ipinfo` and `ivre help iphost`.

Python module

To use the Python module, run for example:

```
$ python
>>> from ivre.db import db
>>> db.passive.get(db.passive.flt_empty)[0]
```

For more, run `help(db.passive)` from the Python shell.

6.3.4 Flow

IVRE flow is a beta feature meant to analyze network flows between hosts. It can be seen as:

- a recon tool for the case of an unknown network (hence its apparition in IVRE/DRUNK)
- a cartography tool to get a better understanding of a supposedly known network (but there is no such thing as a "known network")
- a monitoring tool to spot unwanted flows in your network

Usage

Data insertion

There are two tools for data insertion, the first is based on Zeek (previously known as Bro):

```
$ zeek -r capture_file.pcap
$ ivre zeek2db ./*.log
$ ivre flowcli
```

The second can take either argus logs or netflow logs:

```
$ argus -m -r capture_file.pcap -w flows.argus
$ ivre flow2db flows.argus
```

Or:

```
$ ivre flow2db flows.nfdump
```

Or:

```
$ ivre flow2db -t iptables iptables-from-syslog.log
```

Any of these tools can be called with '–init' to reinitialize the DB.

Data exploration

The main exploration tool are the CLI (`ivre flowcli`) and the Web UI (`<ivre-web-root>/flow.html`).

CLI

You can access the CLI through `ivre flowcli`. Features include:

- Searching for flows and nodes with filters (see the Flow Filters section of this document)
- Producing top values for given criteria
- Plotting flows amounts over hours of days, on average.

See `ivre flowcli -h` for usage details.

Web UI

Overview

The central view is a graph representing the network:

- nodes represent hosts; white ones represent hosts that have incoming network flows, grey ones those who do not have any
- edges represent network flows; same [proto, dport] couple will have the same color

Flows are aggregated by destination port (or code, for icmp), two different connection from the same source to the same destination on the same destination port (so called `dport`) but with different source ports will be aggregated on the same edge.

On the bottom of the graph, there is a timeline, representing the amount of different flows during some time ranges. This timeline can be played by going to the **Display** pane.

On the left, there is a control pane with 3 tabs:

- **Explore:** Allows to explore and reduce the dataset to display with node-based or edge-based queries. See the next section for more details. It also allows to navigate through the data (limit/skip) and change the query mode. At the top of this pane, there is a count of the flows, servers and clients matching the current query. Note that servers can also be counted as clients if they have outgoing flows.
- **Display:** Allows to change the way data is displayed (size of nodes and edges, timeline precision).
- **Details:** Details on the currently selected item.

Interaction

Hover nodes and edges to display their basic properties in the **Details** tab. Click on an edge or a node to query the database for more information, including any associated metadata (for example DNS queries happening on a network flow).

There are two ways of filtering the data:

- Right click on a node or edge and `Filter by`/`Filter out` by attribute
- Write filters yourself

See the Flow Filter section of this document for more information on the filter syntax.

The **Display** pane allows to change the size of nodes and edges based on some criteria:

- On nodes, available keywords are `$in` and `$out`, to make the size proportional to the number of incoming or outgoing flows of a node.
- On edges, a property can be specified (for example `scbytes`, the number of bytes from the server to the client).

Do not forget to increase the `Size scale` to make the result more visible.

The **Display** pane also allows to change the amount of time slots to represent on the timeline (capped by the actual time precision set in `ivre.conf`). The timeline can also be played on the graph by clicking the 'Play timeline' button.

Flow Filters

To write filters, the syntax is as follows:

```
[!][ANY|ALL|ONE|LEN ][src.|dst.][meta.]<attribute> [<operator> <value>]
[OR <other filter>]
```

The `[src.|dst.]` part is only available for node filters.

The special keywords `ANY`, `ALL`, `ONE` and `LEN` are for working with array attributes:

- ALL: matches if all the elements of the array fulfil the predicate
- ANY: the same if any of the elements match
- ONE: the same if exactly one of the elements match
- LEN: the predicate will use the len of the array

Some examples:

- Node filter `dst.addr = 192.168.1.1` will match all the flows whose destination is a host with address `192.168.1.1`.
- Node filter `addr =~ 192\.168\.1\..*` will match all the flows that come from or go to a host whose address matches the `192\.168\.1\..*` regex (sorry, CIDR masks are on their way to be implemented).
- Edge filter `dport > 10000` will match all the flows with a `dport` (destination port) above 10000. `!dport <= 10000` will match the same flows plus the ones that do not have any destination port.
- Edge filter `meta.query =~ .*google.*` will match all the flows that have an associated metadata which have a `query` attribute that match the `.*google.*` regex.
- Edge filter `ANY sports < 1024` will match flows with at least one source port < 1024.
- Edge filter `LEN sports = 1` will match flows with only one known source port.
- Filter `ANY meta.answers =~ .*example.com` will match any metadata that contain an array attribute `answers` where at least one entry matches `'.*example.com'`.

Available operators are:

- = or `:` (equality)
- `!=`
- <, <=, >, >=
- =~

6.3.5 Web User Interface

This web interface presents results of the `view` purpose (see *Purposes*) that can be filtered with keywords (for some of them, shortcuts are available in the menus).

Keep in mind that the information available in this interface highly depends on the options used to run Nmap.

The interface

The top navigation bar

It contains several elements; from left to right:

- A shortcut to the start page, that cleans every keyword.
- A button to display this help page.

- Some menus with shortcuts to add filtering, sort or display commands.
- Some links to "share" (export) the current page.

The left side bar

The first part allows to navigate within the results. Be careful with the last button that goes to the last result page, as it can be very slow when a lot of results are available.

The progress bar shows where the currently displayed results are within the whole results set.

The second part allows to add, modify or remove filter, sort or display commands.

The third part allows to explore the results by generating graphs displayed in the rightmost part of the screen.

- The first field displays a graph with the 15 most common values of a variable in the filtered results. This can be slow when the number of results to scan is important. Here is a list of (sometimes) interesting values to try here:
  - `category`, `source`
  - `country`, `city`, `as`
  - `net`, `net:[mask]`
  - `domains`, `domains:[level]`, `domains:[domain]`, `domains:[domain]:[level]`
  - `hop`, `hop:[number]`
  - `port`, `port:[open/closed/filtered]`, `port:[service]` `portlist:[open/closed/filtered]`, `countports:[open/closed/filtered]`
  - `service`, `service:[port]`, `product`, `product:[port]`, `version`, `version:[port]`
  - `cpe`, `cpe.[type/vendor/product/version]`, `cpe:[cpe spec]`, `cpe.[type/vendor/product/version]:[cpe spec]` (examples: `cpe.product:a:microsoft` will show top product names in CPEs from vendor `microsoft`, `cpe.vendor:o:/^m/` will show top vendor names in CPEs that start with an `m`)
  - `devicetype`, `devicetype:[port]`
  - `script`
  - `script:[scriptname]`
  - `file` (or `file.filename`), `file.time`, `file.size`, `file.uid`, `file.gid`, `file.permission`
  - `smb.os`, `smb.lanmanager`, `smb.domain`, `smb.dnsdomain`, `smb.forest`, `smb.workgroup`
  - `cert.issuer`, `cert.subject`, `cert.md5`, `cert.sha1`, `cert.sha256`
  - `cacert.issuer`, `cacert.subject`, `cacert.md5`, `cacert.sha1`, `cacert.sha256`
  - `sshkey.type`, `sshkey.bits`, `sshkey.fingerprint`
  - `ike.notification`, `ike.transforms`, `ike.transforms.Authentication`, `ike.transforms.Encryption`, `ike.transforms.GroupDesc`, `ike.transforms.Hash`, `ike.transforms.LifeDuration`, `ike.transforms.LifeType`, `ike.vendor_ids`, `ike.vendor_ids.name`, `ike.vendor_ids.value`
  - `modbus.deviceid`, `enip.vendor`, `enip.product`, `enip.serial`, `enip.devtype`, `enip.prodcode`, `enip.rev`, `enip.ip`
  - `httphdr`, `httphdr.name`, `httphdr.value`, `httphdr:[header]`

  - `httpapp`, `httpapp:[application]`

- The *Address space* button displays a graphical representation of the filtered addresses. The abscissa axis represents the two high bytes (or the three when the results belong to the same /16 network), and the ordinate axis represents the two low bytes (or the low byte).
- The *Map* button displays the locations of the results on a world map.
- The *Timeline* and *Timeline 24h* buttons display time-lines where the abscissa axis represents the time and the ordinate axis represents the IP addresses.

Scan results

Ten results (maximum) are displayed per page by default.

Each result has its own frame. In the default display mode, it displays a summary for the host. Long-clicking a result frame toggles between the summary display and the full display for the result.

The pencil icon in the upper-right corner opens the notepad page for the current host (see below) in the rightmost part of the screen.

Each blue element in the results can be clicked to add a filter.

Available commands

Command specification

The commands might require a parameter, provided after the colon sign `:`. Some commands can be used negatively, by prefixing them with `!` or `-`.

The commands can be entered in the input boxes in the second part of the left side bar or added by clicking on a shortcut in the top bar menus.

In the following list, a `[!]` before the command shows it can be used negatively, and a `:` after the command indicates it requires a parameter.

When a parameter is required the full value must be specified, or when appropriate, a regular expression can be used, with the `/[expression]/[flags]` syntax (e.g.: `script:smb-enum-shares:/WRITE/`).

If your command includes spaces, you need to protect it by using single or double quotes.

Command list

Filters

- `[!]host:[IP address]` filter a specific IP address. Using the IP address directly (without `host:`) is equivalent.
- `[!]net:[IP address/netmask]` filter a specific network (CIDR notation). Using the CIDR notation directly (without `net:`) is equivalent.
- `[!]range:[IP address]-[IP address]` filter a specific IP address range
- `[!]hostname:[FQDN]` look for results with a matching hostname.
- `[!]domain:[FQDN]` look for results with a hostname within a matching domain name.
- `[!]category:` filter a category.

- `[!]tag[:value[:info]]` filter a tag.
- `[!]country:[two letters code]` filter a country.
- `[!]city:` filter a city (use with `country:`).
- `[!]asnum:` filter by AS number (lists allowed).
- `[!]asname:` filter by AS name (regular expressions allowed).
- `[!]source:` filter a source (specify the source name).
- `[!]timerange:[timestamp]-[timestamp]` filter results within a specific time range.
- `[!]timeago:` filter recent enough results; the value can be specified in seconds or with the appropriate suffix in minutes (`m`), hours (`h`), days (`d`) or years (`y`).
- `service:[expression]`, `service:[expression]:[port number]` look for an expression in the name of a service.
- `product:[service]:[product]`, `product:[service]:[product]:[port number]` look for a product.
- `version:[service]:[product]:[version]`,`product:[service]:[product]:[version]:[port number]` look for a specific version of a product.
- `script:[scriptid]`, `script:[scriptid]:[output]` look for a specific script.
- `anonftp` filter results with anonymous FTP allowed.
- `anonldap` look for LDAP servers with anonymous bind working.
- `authbypassvnc` look for VNC servers with authentication that can be bypassed.
- `authhttp` look for HTTP servers with authentication and a default (e.g., `admin`/`admin`) login/password working. The Nmap script seems to get a lot a false positives.
- `banner:` look for a specific banner of a service.
- `cookie:` look for HTTP servers setting a specific cookie.
- `file`, `file:[pattern]`, `file:[scriptid]:[pattern]`, `file:[scriptid],[scriptid],...:[pattern]` look for a pattern in the shared files (FTP, SMB, ...).
- `geovision` look for GeoVision web-cams.
- `httptitle:` look for a specific HTML title value of the homepage of a web site.
- `nfs` look for NFS servers.
- `nis`, `yp` look for NIS servers.
- `mssqlemptypwd` look for MS-SQL servers with an empty password for the `sa` account.
- `mysqlemptypwd` look for MySQL servers with an empty password for the `root` account.
- `httphdr`, `httphdr:[header]`, `httphdr:[header]:[value]` look for HTTP headers.
- `httpapp`, `httpapp:[application]`, `httpapp:[application]:[version]` look for HTTP applications.
- `owa` look for OWA (Outlook Web App) servers.
- `phpmyadmin` look for phpMyAdmin servers.
- `smb.dnsdomain:[FQDN]` search results with SMB service in a specific DNS domain.
- `smb.domain:[NetBIOS]` search results with SMB service in a specific NetBIOS domain.

- `smb.fqdn:[NetBIOS]` search results with SMB service in a specific host name (FQDN).
- `smb.forest:[FQDN]` search results with SMB service in a specific forest (DNS name).
- `smb.lanmanager:[LAN Manager]` search results with SMB service with a specific LAN Manager.
- `smb.os:[OS]` search results with SMB service with a specific OS.
- `smb.server:[NetBIOS]` search results with SMB service in a specific host name (NetBIOS).
- `smb.workgroup:[NetBIOS]` search results with SMB service in a specific workgroup (NetBIOS).
- `smbshare`, `smbshare:[access mode]` search results with SMB shares with anonymous access. Access can be 'r', 'w' or 'rw' (default is read or write).
- `sshkey:` look for a particular SSH key.
- `cert.md5:`, `cert.sha1:`, `cert.sha256:` look for a particular certificate.
- `cacert.md5:`, `cacert.sha1:`, `cacert.sha256:` look for a particular CA certificate.
- `torcert` look for Tor certificates.
- `webfiles` look for "typical" web files in the shared folders.
- `webmin` look for Webmin servers.
- `x11open` look for open X11 servers.
- `x11srv` look for X11 servers.
- `xp445` look for Windows XP machines with TCP/445 port open.
- `[!]ssl-ja3-client[:JA3]` look for hosts with a JA3 client or with the given JA3 client.
- `[!]ssl-ja3-server[:[JA3S][:JA3C]]` look for hosts with a JA3 server, with the given JA3 server (optionally corresponding to the given JA3 client).
- `[!]ssl-jarm[:JARM]` look for hosts with a (specific, when specified) JARM fingerprint.
- `hassh[:HASSH]` look for hosts with a (specific, when specified) HASSH fingerprint.
- `[!]useragent[:USERAGENT]` look for hosts with a User-Agent.
- `os:` look for a specific value in the OS discovery results.
- `devtype:`, `devicetype:` look for a type of devices.
- `netdev`, `networkdevice` look for network devices (firewalls, routers, ...).
- `phonedev` look for telephony devices.
- `cpe(:[type](:[vendor](:[product](:[version]))))` look for a given cpe. Each field can be a /regex/.
- `[!]hop:[IP]`, `[!]hop:[IP]:[TTL]` look for a particular IP address in the traceroute results.
- `[!]hopname:` look for a matching hostname in the traceroute results.
- `[!]hopdomain:` look for a hostname within a matching domain name in the traceroute results.
- `[!]tcp/[port number]`, `[!]udp/[port number]`, look for an open TCP or UDP port (using `[!][port number]` directly is equivalent to `[!]tcp/[port number]`).
- `[!]openport` look for hosts with at least one open port.
- `otheropenport:[port number]`, `otheropenport:[port number],[port number],...` look for hosts with at least one open port other than those specified.
- `notes` search results with an associated note.

Sort

- `skip:[count]` skip `count` first results.
- `limit:[count]` only display `count` results.
- `[!]sortby:[field name]` sort according to a field value. Be careful with this setting as consequences on the performances can be terrible.

Display

- `display:host` set the default display mode.
- `display:cpe` only display CPEs.
- `display:script:`, `display:script:[script id]` or `display:script:[script id],[script id],...` only display (a particular) script outputs.
- `display:screenshot` only display screenshots.
- `display:vulnerability` only display vulnerabilities.

6.3.6 IVRE with Kibana

IVRE has an *experimental* backend for Elasticsearch for the `view` purpose (see *Purposes*). Only Elasticsearch 7 supported and tested for now.

While this backend lacks a lot of features, it is enough to create a view into an Elasticsearch cluster. Other tools using Elasticsearch can then use IVRE's data.

Installation

As stated in the installation page (see the *Python* section), you will need to install the elasticsearch and elasticsearch-dsl Python packages.

View creation

About views

Views are created from Nmap, Masscan or Zgrab2 scan results (stored in the `nmap` purpose) and passive host intelligence collected by Zeek (stored in the `passive` purpose). That is a prerequisite of view creation so if you have not read it yet, you should go read *Active recon* and *Passive* first.

You can check you have data in the `nmap` and/or `passive` purposes using the command line: `ivre scancli --count` and `ivre ipinfo --count`.

Configuration

We need to configure IVRE to use the Elasticsearch database for the `view` purpose. Since we want to do that only to create the view, we are going to create a dedicated IVRE configuration file, for example in `~/.ivre-elastic.conf`; for example, to use an Elasticsearch server running on the local machine:

```
echo 'DB_VIEW = "elastic://127.0.0.1:9200/ivre"' > ~/.ivre-elastic.conf
```

Then, to use this dedicated configuration file, we just have to set the `IVRE_CONF` environment variable:

```
IVRE_CONF=~/.ivre-elastic.conf ivre view --count
```

Index creation & Data insertion

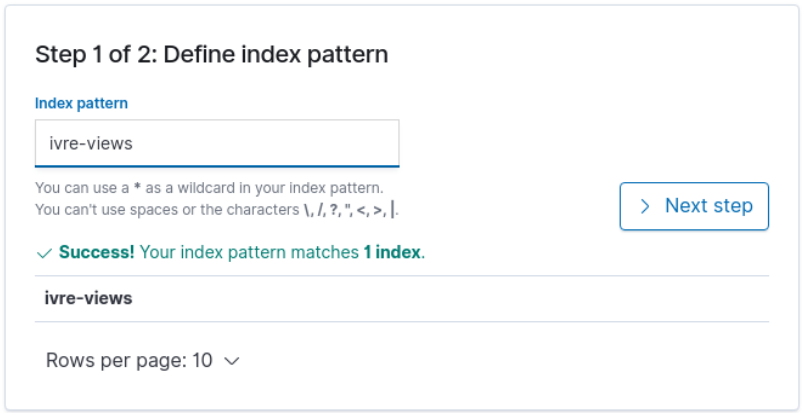
So now, we can create a view as we would do with any other backend. For example, if we want to create a view using all the records from the `nmap` and `passive` purposes:

```
IVRE_CONF=~/.ivre-elastic.conf ivre view --init < /dev/null
IVRE_CONF=~/.ivre-elastic.conf ivre db2view
```

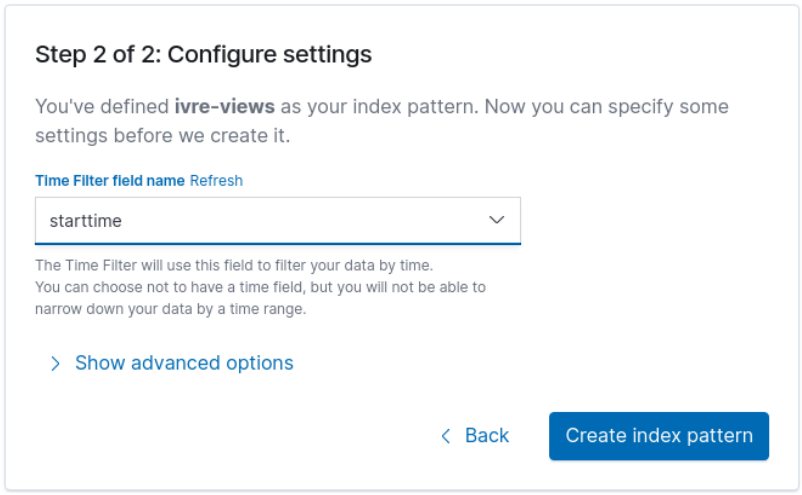
The first command will drop any existing data, and create the index and mapping, and the second will create the view itself.

Using Kibana

From Kibana, you will have to create an index pattern (this can only be done after the view creation). The default index name from view is `ivre-views`; you can use this value as index pattern (and remove the final `*` since we use only one index).



The field `starttime` can be used as the "Time Filter field name".



You are all set! Now, explore this data set as you would explore any other one.

For a couple of examples of how Kibana can be used to explore IVRE's data see the *Kibana exploration* part of the screenshot gallery for examples of useful visualizations.

If you have any troubles with Kibana, please refer to its documentation.

6.4 Development

6.4.1 Web API

GET /config
Returns JavaScript code to set client-side configuration values

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **config** (*object*) – the configuration values

GET /(subdb:re:scans|view)/(action:re:onlyips|ipsports|timeline|coordinates|countopenports
Get special values from Nmap & View databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")
- **action** (*str*) – specific value to get (must be one of "onlyips", "ipsports", "timeline", "coordinates", "countopenports" or "diffcats")

Query Parameters

- **q** (*str*) – query (including limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results (forces "json" format)
- **ipsasnumbers** (*bool*) – to get IP addresses as numbers rather than as strings
- **datesasstrings** (*bool*) – to get dates as strings rather than as timestamps
- **format** (*str*) – "json" (the default), "ndjson" or "txt"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Array of Objects

- **object** – results

GET /(subdb:re:scans|view)/count
Get special values from Nmap & View databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")

Query Parameters

- **q** (*str*) – query (including limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **int** – count

GET /(subdb:re:scans|view|passive)/top/(field: *path*)
Get top values from Nmap, View & Passive databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")
- **field** (*str*) – (pseudo-)field to get top values (e.g., "service")

Query Parameters

- **q** (*str*) – query (including limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results
- **ipsasnumbers** (*bool*) – to get IP addresses as numbers rather than as strings
- **datesasstrings** (*bool*) – to get dates as strings rather than as timestamps
- **format** (*str*) – "json" (the default) or "ndjson"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Array of Objects

- **label** (*str*) – field value
- **value** (*int*) – count for this value

GET /(subdb:re:scans|view|passive)/distinct/(field: *path*)
Get distinct values from Nmap, View & Passive databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")
- **field** (*str*) – (pseudo-)field to get distinct values (e.g., "service")

Query Parameters

- **q** (*str*) – query (including limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results
- **ipsasnumbers** (*bool*) – to get IP addresses as numbers rather than as strings
- **datesasstrings** (*bool*) – to get dates as strings rather than as timestamps

- **format** (*str*) – "json" (the default) or "ndjson"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Array of Objects

- **label** (*str*) – field value
- **value** (*int*) – count for this value

GET /(subdb:re:scans|view)
Get records from Nmap & View databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")

Query Parameters

- **q** (*str*) – query (including limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results
- **ipsasnumbers** (*bool*) – to get IP addresses as numbers rather than as strings
- **datesasstrings** (*bool*) – to get dates as strings rather than as timestamps
- **format** (*str*) – "json" (the default) or "ndjson"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Array of Objects

- **object** – results

POST /(subdb:re:scans|view)
Add records to Nmap & View databases

Parameters

- **subdb** (*str*) – database to query (must be "scans" or "view")

Form Parameters

- **categories** – a coma-separated list of categories
- **source** – the source of the scan results (mandatory)
- **result** – scan results (as XML or JSON files)

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer, source or username missing

Response JSON Object

- **count** (*int*) – number of inserted results

GET /flows
Get special values from Nmap & View databases

Query Parameters

- **q** (*str*) – query (including limit/skip, orderby, etc.)
- **callback** (*str*) – callback to use for JSONP results
- **action** (*str*) – can be set to "details"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **object** – results

GET /ipdata/ (*addr*)
Returns (estimated) geographical and AS data for a given IP address.

Parameters

- **addr** (*str*) – IP address to query

Query Parameters

- **callback** (*str*) – callback to use for JSONP results

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **object** – the result values

GET /passivedns/(**query:** *path*)
Query passive DNS data. This API is compatible with the Common Output Format and implemented in CIRCL's PyPDNS.

It accepts two extra parameters, not supported (yet?) in PyPDNS:

- *subdomains*: if this parameter exists and a domain name is queried, records for any subdomains will also be returned.
- *reverse*: if this parameter exists and a domain name is queried, records pointing to the queried domain (CNAME, NS, MX) will be returned.

It also returns additional information:

- "sensor": the "sensor" field of the record; this is useful to know where this answer has been seen.
- "source": the IP address of the DNS server sending the answer.

Parameters

- **query** (*str*) – IP address or domains name to query

Query Parameters

- **subdomains** (*bool*) – query subdomains (domain name only)
- **reverse** (*bool*) – use a reverse query (domain name only)

- **type** (*str*) – specify the DNS query type

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **object** – the result values (JSONL format: one JSON result per line)

GET /passive
Get records from Passive database

Query Parameters

- **q** (*str*) – query (only used for limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results
- **ipsasnumbers** (*bool*) – to get IP addresses as numbers rather than as strings
- **datesasstrings** (*bool*) – to get dates as strings rather than as timestamps
- **format** (*str*) – "json" (the default) or "ndjson"

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Array of Objects

- **object** – results

GET /passive/count
Get special values from Nmap & View databases

Query Parameters

- **q** (*str*) – query (only used for limit/skip and sort)
- **f** (*str*) – filter
- **callback** (*str*) – callback to use for JSONP results

Status Codes

- 200 OK – no error
- 400 Bad Request – invalid referer

Response JSON Object

- **int** – count

6.4.2 Tests

The `tests` directory is mainly intended for people who want to contribute to IVRE and want to make sure their changes do not break IVRE.

Dependencies

To run IVRE tests you will need coverage.py.

Test case

The first thing is to find samples. You need both (recent) Nmap XML scan result files (or Nmap JSON files, as generated by `ivre scancli --json`) and PCAP files (dump while you browse, and browse a lot, or sniff a busy open Wi-Fi network, if that's legal in your country).

A good test case should have a lot of various data (sniff from different places, scan different hosts with different Nmap options).

It is mandatory to have at least, for the Nmap test:

- Two scanned (and up) hosts with different IP addresses
- One host scanned with the script `http-robots.txt` reporting `/cgi-bin` in its output.
- One host scanned with an anonymous FTP server.
- One scan result with traceroute and at least one hop with a hostname.
- One host scanned with a hostname ending with ".com".

For the passive test:

- Two records with different IP addresses.
- One SSL certificate.

First run

From the `tests` directory, create the `samples` subdirectory and place your samples there (the PCAP files must have the extension `.pcap`, the Nmap XML result files must have the extension `.xml`, and the Nmap JSON results must have the extension `.json`).

Then, run `python ./tests.py` (optionally replace `python` by the alternative interpreter you want to use, e.g., `python3.11`; note that `coverage.py` must be installed for this interpreter). The first run will create a `samples/results` file with the expected values for some results. The next runs will use those values to check if something has been broken.

For this reason, it is important to:

- Run the tests for the first time with a "known-working" version.
- Remove the file `samples/results` whenever a sample file is added, modified or removed.

Improving the test case

If you want to make sure to have enough samples, you can:

- Check the `samples/results` file for `*_count` entries with low values (particularly 0, of course) and find or create new samples that will improve those values.
- Check the report generated by coverage.py under the `htmlcov` directory, and check whether your current test case covers at least the code you want to change.

Failures

Tests failure are not always an issue. Apart from a new bug, of course, here are some reasons that can explain test failures:

- You have added new samples and have not removed the `samples/results` file.
- Your samples do not match the minimum requirements detailed above.
- A new feature has been added to IVRE and the new results are actually better than the stored ones.

GitHub actions

Tests are run with several MongoDB and PostgreSQL versions, as well as TinyDB, SQLite and Elasticsearch for each pull requests. The tests run with Python 3.7 to 3.11.

The configurations are in the *.github/workflows/*.yml* YAML files.

6.4.3 Code linting

IVRE uses code linters to prevent some easy-to-spot (for a computer) mistakes and to enforce a consistent code style (or at least, attempt to do so).

So far, only the Python code uses such linters (Flake8, Pylint, Mypy, Bandit and Black). Adding similar code linting capabilities to the Zeek scripts (*zeek/*), LUA capabilities to the Zeek scripts (*zeek/*), LUA (*patches/nmap/scripts/*) and JavaScript / HTML (*web/static/*) could be a good PR idea!

For all the code and the documentation, we also use Codespell to prevent typos.

Running the linters

To install the Python code linters and Codespell you can simply use the `requirements-linting.txt` file with Pip, or use any method to install the latest versions of the `black`, `codespell`, `flake8` and `pylint` Python modules.

The script `pkg/runchecks` will run all the tests for you with the expected options and exceptions.

GitHub actions

Code linting and spell checking is performed in a dedicated GitHub action (see *GitHub actions*), togethter with the Maxmind tests. Pylint and Codespell only run with Python 3.11, while Flake8 runs with all Python versions.

6.5 IVRE: GPL v3

IVRE is released under the GNU GPL version 3.

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

6.5.1 Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

6.5.2 TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or

running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- **a)** The work must carry prominent notices stating that you modified it, and giving a relevant date.
- **b)** The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- **c)** You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- **d)** If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- **a)** Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- **b)** Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- **c)** Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- **d)** Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- **e)** Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product

in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- **a)** Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- **b)** Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- **c)** Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- **d)** Limiting the use for publicity purposes of names of licensors or authors of the material; or
- **e)** Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- **f)** Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringe-

ment). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and

conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

6.6 Licenses for external files

The following files are distributed with IVRE but are **not** part of the project and therefore have their own copyright and licenses.

6.6.1 AngularJS

Files under the `web/static/an` directory come from the AngularJS project.

They are licensed under MIT license.

6.6.2 Twitter Bootstrap

Files under the `web/static/bs` directory come from the Twitter Bootstrap project.

They are licensed under Apache License v2.0.

The files under the `web/static/bs/img` come from Glyphicons and are made available by the Twitter Bootstrap project.

6.6.3 jQuery

Files under the `web/static/jq` directory come from the jQuery project.

They are licensed under MIT license.

6.6.4 D3.js

Files under the `web/static/d3` directory come from the D3.js project.

They are licensed under BSD license.

6.6.5 Linkurious

Files under the `web/static/lk` directory come from the Linkurious.

They are licensed under GPL v3 license.

6.6.6 flag-icon-css

Files under the `web/static/fi` directory come from the flag-icon-css project.

It includes the European flag from pull request #71.

These files are licensed under MIT license.

6.6.7 ike-scan Vendor ID database

The file `data/ike-vendor-ids` comes from the ike-scan project.

It includes new fingerprints discovered during Internet-wide ISAKMP scans. Those fingerprints have of course been contributed to ike-scan.

This file is licensed under GPL v3 license.

See also Use of ike-vendor-ids in other (open-source) programs.

6.6.8 manuf Vendor database

The file `data/manuf` is built from the tool `pkg/buildmanuf` from the wireshark project.

Both files are licensed under GPL v2 license or later.

6.6.9 Natural Earth

The file `web/static/world-110m.json` has been generated with the tools from Geospatial Data Abstraction Library and World Atlas from data published by Natural Eath in the public domain.

6.6.10 Logo

The file `web/static/droids.png` is the logo of the Droids Corporation. It is not covered by IVRE license.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

HTTP Routing Table

/(subdb:re:scans|view)

/(subdb:re:scans|view|passive)

/config

/flows

/ipdata

/passive

/passivedns