# DomainRadar v1.0 - documentation

A machine-learning-based application for malicious domain name detection

Authors: Radek Hranický, Jan Polišenský, Adam Horák
NES@FIT research group, Faculty of Information Technology, Brno University of Technology, 2022

## Description of the software

DomainRadar is an experimental prototype of a threat detection & incident response application. The tool utilizes machine learning to uncover communication with malicious domains. For a given network, the application shows the list of domain names that devices from this network communicate with. For each domain, DomainRadar shows the **Badness** ratio. The badness ratio describes how potentially malicious the domain is. The application also provides a detailed description of outputs from different classifiers.

Tha classification is based on machine learning (SVM, NN) with features from the following sources:

- **Domain name syntax** – a lexical analysis of the domain name itself;
- **Geolocation data** – analysis of the origin and GPS coordinates of related IP addresses;
- **Whois data** – data available from WHOIS for the given domain;
- **TLS/SLL certificate information** – information from the certificate related to the domain;
- **DNS data** – records from DNS servers available for the given domain.

The DomainRadar app is designed to cooperate with Suricata IDS and QRadar SIEM. It loads the data from the QRadar SIEM, namely:

- **Offenses** – security incidents detected by QRadar and associated Offense sources – IP addresses of the attackers, related events and flows.
- **DNS traffic data** – syslog messages about DNS requests and responses, created by Suricata IDS, and forwareded to QRadar SIEM using the Syslog service.

The architecture of the DomainRadar application backend and related subsystems is shown in Figure 1. Yellow-colored modules are part of DomainRadar, while blue boxes represent external application that DomainRadar communicated with. The etire ecosystem consists of the following:

- **QRadar SIEM** – An installation of IBM QRadar SIEM;
- **QRadar API** – An application interface for remote access to the QRadar SIEM;
- **/offense_sources** endpoint – An endpoint for loading offense source-related information;
- **/ariel/*** - Endpoints for accessing the QRadar Ariel database. The database contains stored syslog messages. DomainRadar loads information about DNS traffic from here;
- Flask App – The core of DomainRadar. Provides endpoints for front-end communication, connection to the SQL (currently SQLite) database, and management of Loader, Resolver, and Analyzer daemons. The App also provides the DomainRadar API with a series of endpoints. Those serve for interaction from the DomainRadar frontend and/or external applications that utilize the DomainRadar API. The following endpoints are implemented:
    - **/loading** - An endpoint for control of loading the data from IBM QRadar.
    - **/resolving** - An endpoint for control of data resolution from external sources.

    - **/analysis** - An endpoint for control of the loading process via the DomainRadar API.
    - **/domain** - An endpoint for obtaining domain name information and classification results via the DomainRadar API.
- **Loader** – A daemon for loading data from the IBM QRadar. The loading process is launched by the Flask application. The action is currently triggered by the user. In future versions, automated scheduling of the loading process is assumed.
- **Resolver** – A daemon for resolving requests for external data. It contains a series of individual modules (resolvers): resolver of Geolocation data, WHOIS data, DNS data, and TLS/SSL certificate information. The loading process is launched by the Flask application. The action is currently triggered by the user. In future versions, automated scheduling of the loading process is assumed.
- **Analyzer** – A daemon for the analysis and classification of the detected domain names. It utilizes a series of individual modules where each module implements a classifier of the given type. The analysis process is launched by the Flask application. The action is currently triggered by the user. In future versions, automated scheduling of the loading process is assumed. (NOTE: the concrete names of classifier modules are displayed for illustration and may differ from the implementation in the current version.)

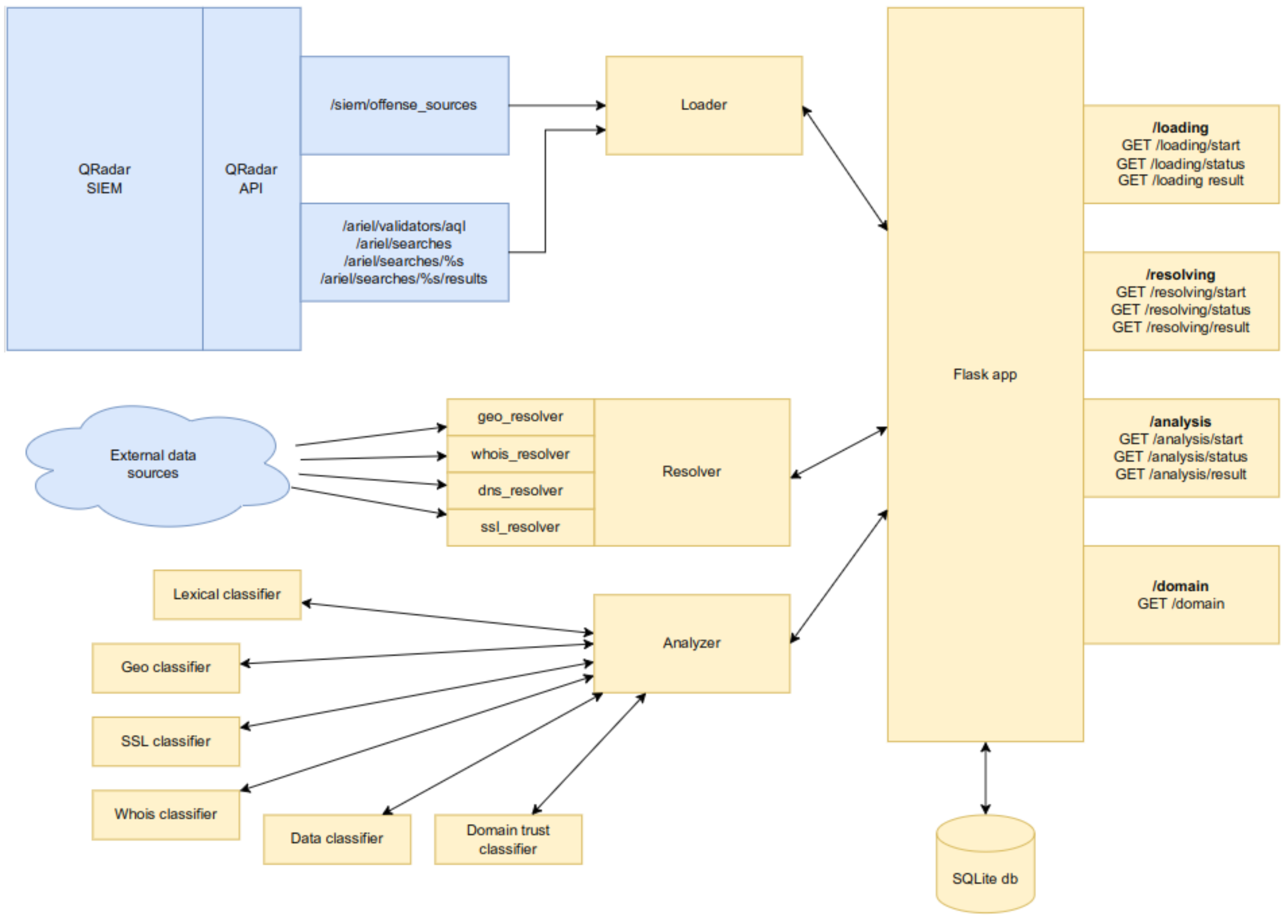


**Figure 1**: The architecture of the DomainRadar back-end and related systems

## Installation and startup

For deploying the application, you need to have a running instance of IBM QRadar SIEM or IBM QRadar CE. Additionally, an instance of Suricata IDS is required with the detection of DNS traffic switched on and syslog message forwarding to the QRadar. Also, ensure TCP ports 5000 and 3000 are free to use. Otherwise, the application port settings need to be reconfigured manually.

To run the **backend** – descend into the "domainradar" directory and proceed as follows:

- Install Python 3 and pip3, if not installed already.
- Install the required Python 3 packages:
  ```
  pip3 install -r requirements.txt
  ```
- In **config.ini**, specify:
  - **API_URL** - the URL of your QRadar API
  - **SEC_TOKEN** - authentication token for accessing the QRadar API
- For testing purposes, you may deploy an example database:
  ```
  cp EXAMPLE.db domainradar.db
  ```
  To start with an empty database, you may skip the step or delete the "domainradar.db" later.
- Type:
  ```
  python3 app.py
  ```
- The backend and related services should be running on port 5000.
- NOTE: For production environment, it is recommended to run the app.py over WSGI and Apache/NGINX web server instead.

To run the **frontend** – descend into the "client" directory and proceed as follows:

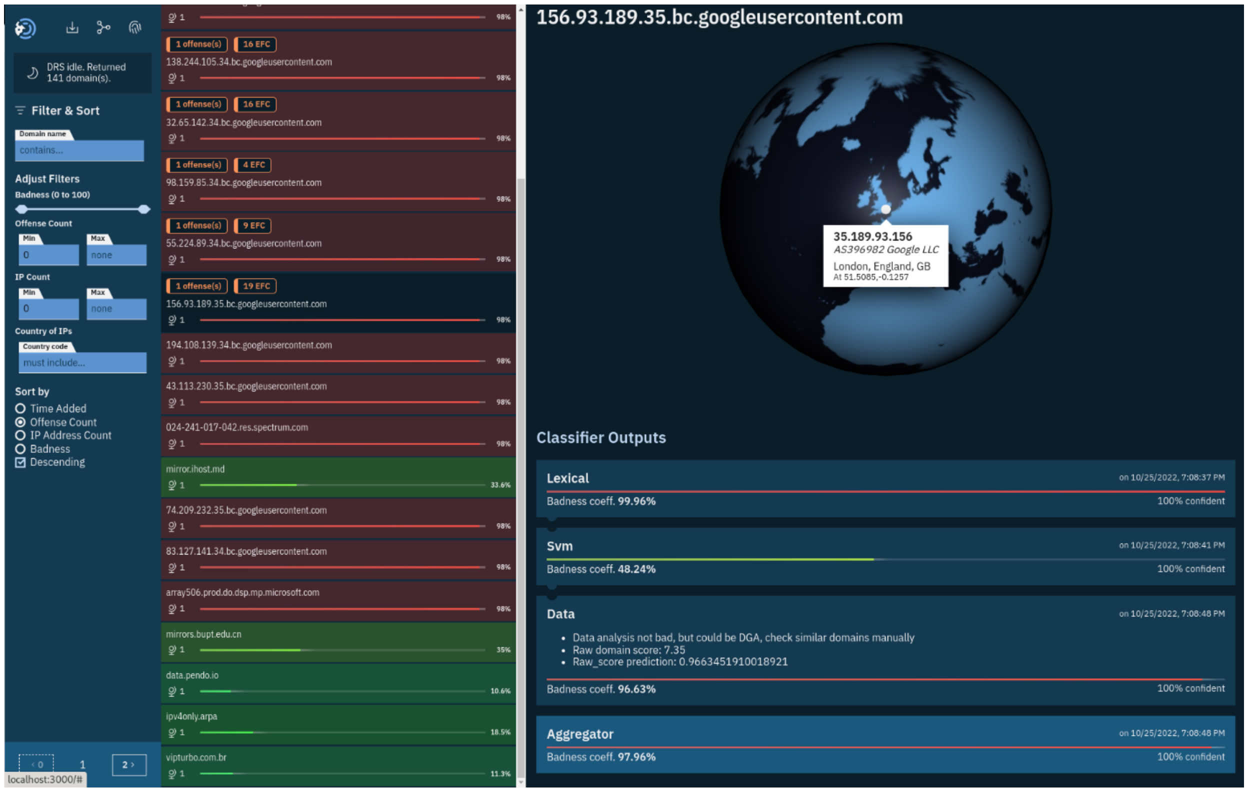
- Install Node JS and Node Package Manager (npm), if not installed already.
- Install requirements by running:
  ```
  npm i
  ```
- Run the application in the development/debug mode:
  ```
  npm run dev
  ```
- Visit http://localhost:3000
- NOTE: For production environment, you may run
  ```
  npm run build
  ```
  which creates a static HTML+JS website under the /dist directory. The contents may be loaded using the Apahce/NGINX web server.

## Usage

Once you access the DomainRadar front-end user interface, you should see controls similar to those in Figure 2. On the left side, there is the control panel with operation buttons on the top, status window, and various filtering and sorting options.

In the middle side, there is a list of domains with the Badness ratio displayed. The **greener** the domain, the more trusted DomainRadar thinks it is. The **redder** it is, the more malicious it could potentially be. For each domain, you also see the number of associated IP addresses, number of related QRadar Offenses (if any), and the total count of offense-releated Events and Flows – the "Event-flow count" (EFC). The domains are displayed to meet the criteria from the left panel. If there are too many domains, the interface divides the listing into pages. Page controls are located on the very bottom.

Aftery clicking on a concrete domain name from the list, DomainRadar shows you detailed in the right-side window. On the top, you see the domain name with a visualisation of the Earth. If there is any IP address detected for the given domain name, the geographic location is shown on the planet. In case there are multiple IP addresses per domain, a drop-down box appears in the top-right corner of the user interface. Using the box you may navigate between the associated IP addresses. In the bottom part, you see the outputs of the individual classifiers.


DRS idle. Returned 141 domain(s).

Filter & Sort

Domain name: contains...

Adjust Filters

Badness (0 to 100)

Offense Count: Min 0, Max none

IP Count: Min 0, Max none

Country of IPs: Country code: must include...

Sort by
- ( ) Time Added
- (●) Offense Count
- ( ) IP Address Count
- ( ) Badness
- ☑ Descending

| Domain | Offenses | EFC | IPs | Badness |
|---|---|---|---|---|
|  |  |  | 1 | 98% |
| 138.244.105.34.bc.googleusercontent.com | 1 offense(s) | 16 EFC | 1 | 98% |
| 32.65.142.34.bc.googleusercontent.com | 1 offense(s) | 16 EFC | 1 | 98% |
| 98.159.85.34.bc.googleusercontent.com | 1 offense(s) | 4 EFC | 1 | 98% |
| 55.224.89.34.bc.googleusercontent.com | 1 offense(s) | 9 EFC | 1 | 98% |
| 156.93.189.35.bc.googleusercontent.com | 1 offense(s) | 19 EFC | 1 | 98% |
| 194.108.139.34.bc.googleusercontent.com |  |  | 1 | 98% |
| 43.113.230.35.bc.googleusercontent.com |  |  | 1 | 98% |
| 024-241-017-042.res.spectrum.com |  |  | 1 | 98% |
| mirror.ihost.md |  |  | 1 | 33.6% |
| 74.209.232.35.bc.googleusercontent.com |  |  | 1 | 98% |
| 83.127.141.34.bc.googleusercontent.com |  |  | 1 | 98% |
| array506.prod.do.dsp.mp.microsoft.com |  |  | 1 | 98% |
| mirrors.bupt.edu.cn |  |  | 1 | 35% |
| data.pendo.io |  |  | 1 | 10.6% |
| ipv4only.arpa |  |  | 1 | 18.5% |
| vipturbo.com.br |  |  | 1 | 11.3% |

‹ 0   1   2 ›

localhost:3000/#

156.93.189.35.bc.googleusercontent.com

35.189.93.156
*AS396982 Google LLC*
London, England, GB
At 51.5085,-0.1257

Classifier Outputs

**Lexical** on 10/25/2022, 7:08:37 PM
Badness coeff. **99.96%** 100% confident

**Svm** on 10/25/2022, 7:08:41 PM
Badness coeff. **48.24%** 100% confident

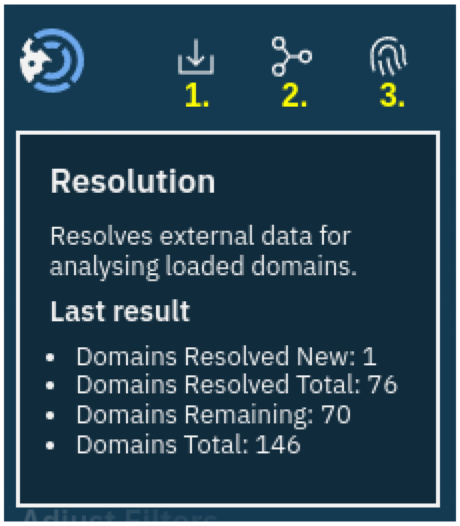
**Data** on 10/25/2022, 7:08:48 PM
- Data analysis not bad, but could be DGA, check similar domains manually
- Raw domain score: 7.35
- Raw_score prediction: 0.9663451910018921

Badness coeff. **96.63%** 100% confident

**Aggregator** on 10/25/2022, 7:08:48 PM
Badness coeff. **97.96%** 100% confident

**Figure 2:** Front-end with the user interface of DomainRadar

Operating the actual computation and domain classification process is done via controls in the very top-left corner of the interface, as shown in Figure 3. See the three buttons with the yellow numbers 1 to 3 under. The buttons operate as follows. Button 1 starts the Loading process that extracts the necessary data (offenses, DNS traffic data) from the QRadar SIEM. Next, Button 2 triggers the Resolving process that download all required domain-releated information from external data sources (WHOIS, DNS, …). Finaly, Button 3 launches the Analysis process that eventually processes all unclassified domains using the available classifiers and data. By hovering the mouse over each button, you can see progress of the individual operation. Once any operation is done, results appear in the box under the buttons, as shown in Figure 3.



Resolution

Resolves external data for analysing loaded domains.

Last result

- Domains Resolved New: 1
- Domains Resolved Total: 76
- Domains Remaining: 70
- Domains Total: 146

**Figure 3:** Controls of DomainRadar operations

The data loading and domain evaluation processes are incremental. This means DomainRadar only adds new domains, domain-IP mappings and offense sources when it contants QRadar. If a domain name or offense source already exists, it is only updated. All previous results remain available. The classification is only performed for domains that are not classified already. All information is stored inside the DomainRadar database.

## Acknowledgements

This  software prototype was developed under the AI methods to cyber security and control systems project, FIT-S-20-6293, funded by Brno University of Technology.

We also thank other researchers, namely Ondřej Ryšavý, Petr Matoušek, Ondrej Lichtner, Kamil Jeřábek and Jan Pluskal who provided us with assistance, useful tips and feedback during the creation of the DomainRadar prototype.