SOCCRATES

SOC & CSIRT Response to Attacks & Threats based on attack defense graphs Evaluation Systems

D4.3

Threat Identification and Threat Trend Prediction – Final Prototype

Deliverable type:	DEM
Contributing work packages:	WP4 AI based Detection & Prediction
Due date of deliverable:	28/02/2022
Submission date:	28/02/2022
Dissemination level:	PU

Responsible organisation:	SHS
Editor:	Piotr Kijewski
Revision:	1.0

Abstract	This deliverable provides an overview of the prototyping work carried out as part of Task T4.2 in WP4: Threat Identification and Threat Trend Prediction. It provides an overview of the platform developed for DNS/DGA threat identification and trend prediction. It goes in depth into DGA research with a focus on machine learning algorithms and their effectiveness at identifying DGA based threats.
Keywords:	DNS, DGA, malware, botnet, machine learning

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 833481
Call H2020-SU-ICT-2018 • Innovation Action • Start date: September 1st, 2019

Management summary

This deliverable focuses on the “Threat Identification and Threat Trend Prediction” component prototype that has been developed at TRL9 and is now operational at Shadowserver. The work is focused on the malware sandbox and malware data repository operated in production by Shadowserver that handles hundreds of thousands of unique malware ingestions per day. More specifically, it is focused on DNS traffic seen in Shadowserver’s sandboxes and on Domain Generation Algorithm (DGA) domain queries generated by certain malware families. DNS domains and DGAs are used by these malware families for ensuring stable and resilient Command and Control (C2) communications, and enable threat actors to maintain control of their malware and avoid disruptions. Detecting, understanding and predicting future DGA domains is the first step toward combating them.

The deliverable is intended as a step by step guide towards explaining various elements of the software prototype - a collection of tools developed - which is the primary output of the work in Task 4.2. We explain the dataset we have at our disposal and hence any limitations in the interpretation of what prediction is and what is the goal we want to achieve at TRL9.

We explain different machine learning feature-based and featureless approaches towards detecting and classifying DGA domains, along with example results and their initial evaluations. We then introduce DGA Detective, an open source DGA machine learning classifier developed as part of the project and in use at Shadowserver (and also independently at Mnemonic and F-Secure). We explain how the DGA Detective is used daily in the production environment at Shadowserver and the various tools (including a new dashboard) built at Shadowserver to enable hunting for DNS/DGA based threats, as well as for tracking and observing new trends in attacks. We also introduce Middleman, a tool that can interact with the Shadowserver API to download information relevant to detecting threats on local networks. Finally, we provide an overview of actual real world results obtained with the above prototype, including both research and operational results that have led to DNS sinkholing of certain groups of threats and sharing of infection information with operational communities worldwide.

Contributor(s)	Piotr Kijewski (SHS) Irina Chiscop (TNO) Federico Falconieri (TNO) Francesca Soro (AIT)

Reviewer(s)	Paul Smith (AIT) David Watson (SHS)

Security Assessment	**See deliverables table for security assessment requirements**
Approval Date	25/02/2022
Remarks	None

TABLE OF CONTENTS

1 INTRODUCTION – RATIONALE OF THIS DOCUMENT 6

1.1 The SOCCRATES project 6
1.2 This deliverable 7

2 THREAT PREDICTION 9

3 UPDATED VIEW OF THE DGA THREAT LANDSCAPE 10

3.1 Flubot DGA 10
3.2 Solarwinds/SUNBURST malware 11
3.3 DoH and DoT 12
3.4 Cryptocurrrency Domains and other alternative DNS systems 12

4 THE SHADOWSERVER SANDBOX DATASET REVISTED 13

5 DGA CLASSIFICATION METHODS AND PIPELINE 15

5.1 Goals 15
5.2 DGA classification methods 15
5.2.1 Feature-based approach 15
5.2.2 Featureless approach 17

6 DGA CLASSIFIER IMPLEMENTATION AND DEPLOYMENT 22

6.1 DGA Detective 22
6.2 Implementation 23
6.3 Deployment in the production environment 24

7 EVALUATION OF DGA DOMAIN CLASSIFICATION RESULTS 25

7.1 Feature-based approach 25
7.1.1 Binary classification 25
7.1.2 Family classification 26
7.2 Featureless approach 27
7.2.1 Binary classification 27
7.2.2 Family classification 28
7.3 Comparison of approaches 29
7.3.1 Binary classification on unknown domains 29
7.3.2 Comparison of the two methods on the family classification task 31
7.4 Lessons learned 32

8 THREAT HUNTING : TOOLS FOR DETECTING ANOMALIES AND TRENDS 34

8.1 TLD monitoring 34
8.2 DoH/DoT monitoring 36
8.3 Registrar monitoring 37
8.4 Daily unregistered domain e-mail reports 38

9 THE SHADOWSERVER API 40

9.1 Methods implemented 40

9.2 Example data returned for DGA Detective findings 41
10 THE MIDDLEMAN 42
11 FINDINGS AND CONTRIBUTIONS TO THE COMMUNITY 43
11.1 Phorpiex example sinkholing results 44
11.2 Neksminer example sinkoling results 45
11.3 M0yv/Expiro example sinkholing results 46
12 FUTURE DIRECTIONS 48
13 CONCLUSION 49
14 REFERENCES 50
15 ABBREVIATIONS 51

1 Introduction – Rationale of this document

This section introduces the SOCCRATES project and defines the goals of this deliverable.

1.1 The SOCCRATES project

SOCCRATES (SOC & CSIRT Response to Attacks & Threats based on attack defence graphs Evaluation Systems) is an EU funded project under the Horizon2020 programme that has the following main challenge:

How can SOC and CSIRT operations effectively improve their capability in detecting and managing response to complex cyber-attacks and emerging threats, in complex and continuously evolving ICT infrastructures while there is a shortage of qualified cybersecurity talent?

The main objective of SOCCRATES is to develop and implement a security automation and decision support platform (‘**the SOCCRATES platform’)** that will significantly improve an organisation’s capability (usually implemented by a SOC and/or CSIRT) to quickly and effectively detect and respond to new cyber threats and ongoing attacks.

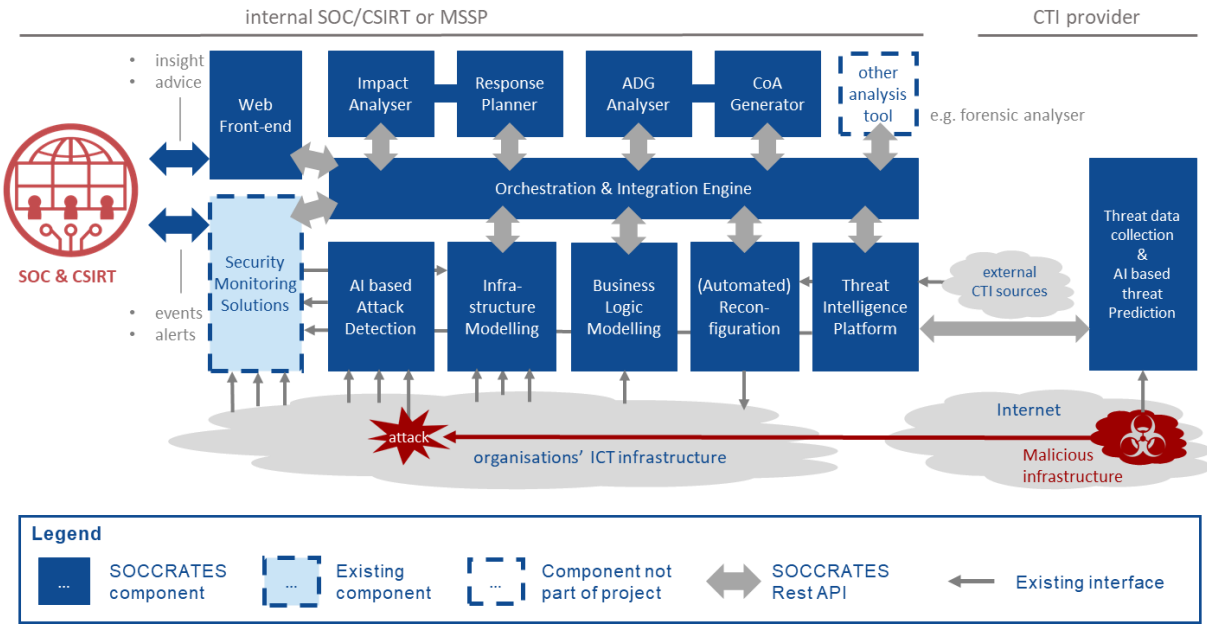


Figure 1-1 – The SOCCRATES platform

The SOCCRATES platform (see **Figure 1-1**) consists of an orchestrating function and a set of innovative components for automated infrastructure modelling, attack detection, cyber threat intelligence utilization, threat trend prediction, and automated analysis using attack defence graphs and business impact modelling to aid human analysis and decision making on response actions, and enable the execution of defensive actions at machine-speed.

SOCCRATES has the following concrete project objectives:

1. Deliver the SOCCRATES platform consisting of an orchestration function and a unique integration of innovative background solutions that seamlessly work together.
2. Show that the SOCCRATES platform can improve SOC operations by evaluating the SOCCRATES platform in two diverse real-life pilot environments.
3. Examine and illustrate the benefits of automation for selected SOC activities to help manage the cyber security skills gap in organizations.
4. Prepare for successful exploitation by the SOCCRATES partners of the individual innovated components and the integrated SOCCRATES platform in commercial products that are offered to the market and are available for the European (business) community.

Please visit www.soccrates.eu for more information on the SOCCRATES project.

1.2 This deliverable

This deliverable is intended to provide an overview of the SOCCRATES developed "Threat Identification and Threat Trend Prediction" prototype that is operational at The Shadowserver Foundation (SHS), and is a summary of work carried out in Task T4.2 of WP4 by TNO, AIT and SHS. The prototype focuses on analysing domain name traffic at scale in Shadowserver's sandbox. It consists of a number of elements, including:

1) A DGA Detective component that can classify domains as DGAs as well as suggest potential malware family classification.
2) A component that tracks anomalies in DNS queries in the sandbox from a TLD, DNS/DoH/DoT and Registrar perspective.
3) A dashboard component that visualizes trends in domain traffic patterns and enables drill down search capabilities, based on 1) and 2).
4) An API that enables external access to a set of methods specified in D2.3.
5) A Middleman component developed in T4.1 that can pull results from the API (such as DGA Detective family classifications) and apply them toward detection of potentially new malicious traffic.

We would like to stress that work in this Task 4.2 is intended to produce results at a TRL9 level "actual system proven in operational environment (competitive manufacturing in the case of key enabling technologies; or in space)"[1] hence the focus of our work is on implementing innovation in production environments, rather than pure research.

1.3 Structure of this deliverable

The structure of this deliverable is as follows:

- Section 1: Introduction to the SOCCRATES Project and this Deliverable.
- Section 2: A discussion on the prediction of threats, and how we chose to approach this issue with the datasets at our disposal and TRL 9 goal.

[1] https://ec.europa.eu/research/participants/data/ref/h2020/other/wp/2016_2017/annexes/h2020-wp1617-annex-g-trl_en.pdf

- Section 3: Updated view of the malware DGA landscape as seen by Shadowserver, including recent events and prominent DGA-related threats.
- Section 4: Shadowserver Sandbox Dataset Revisited. This section covers a brief update on the Sandbox and DGA threat data shared within SOCCRATES.
- Section 5: DGA Classification methods and pipeline. This section deals with different machine learning approaches tested in the context of DGAs and developed as part of the SOCCRATES project.
- Section 6: DGA Classifier implementation and deployment. This section describes the DGA Detective component responsible for classifying DNS domains and how it was deployed in practice.
- Section 7: Evaluation of DGA results. This section discusses DGA classification results obtained through various approaches.
- Section 8: Threat Hunting: Tools for Detecting Anomalies and Trends. This section introduces a dashboard for monitoring DNS queries in the Sandbox, trend analysis algorithms and how these are used in combination with DGA Detective to hunt for threats and trends to achieve real world results.
- Section 9: The Shadowserver SOCCRATES API. A short recap of functionality offered by the Shadowserver API (at TRL9) to project partners and outside users, along with an example query of DGA Detective results. The Full API specification was described in D2.3.
- Section 10: Overview of Middleman, a component developed to pull Shadowserver data from the API and apply it to network traffic (including other pilot sites).
- Section 11: Findings and Contributions to the Community. Insight into how the work presented in the deliverable has contributed in practice to both the research and operational communities (SOCs/CSIRTs).
- Section 12: Future Directions. Future directions for possible research and implementation.
- Section 13: Conclusion

2 Threat Prediction

In this section we explain how we approached the issue of "predicting threats" with the datasets at our disposal, and our stated goal of achieving a solution at TRL 9 for maximum impact.

As mentioned in the project proposal, the investigations in Task 4.2 are primarily aimed at providing insight into the data collected in the Shadowserver malware repository, focusing on large scale sandbox observations, with hundreds of thousands of unique by hash malware samples ingested daily. Especially interesting are the DNS queries generated when malware is executed in our sandboxes, with live Internet access, since they represent resources often used for C2 purposes that can potentially be taken down by Shadowserver, as part of its mission to make the Internet a more secure environment.

As such, all "predictions" are limited by the specific features of malicious activity available in the sandbox dataset. For this reason, the obvious area of focus are DGAs, algorithms used to generate many potential domain names. Detecting and understanding DGAs allows us to predict which DNS domains will be likely used for malicious intent in the future and allow us to block them by registering them before an attacker or otherwise taking them down in collaboration with law enforcement or industry.

We have also addressed the prediction issue by building an environment where anomalies in DNS queries can be more easily observed and subsequently investigated. This supports an analyst in their ability to understand trends and thus on a certain level "predict" what can happen in the future. This is described in Section 8 Threat Hunting: Tools for Detecting Anomalies and Trends, where we explain how we observe trends in domain TLD usage, crypto domain usage and selection of domain registrars.

The above interpretation allows us to build a platform that operates under TRL9, which was our project goal.

3 Updated view of the DGA Threat Landscape

Domain Generation Algorithms (DGAs) were introduced and explained in the preliminary D4.2 deliverable. We will not go into the details of that explanation and will refer the reader to that deliverable instead, but for easiness of reading of this deliverable, we will repeat the basics:

Domain Generation Algorithms (DGAs) are used by malware authors to programmatically generate many potential domain names, changing the entire dataset every pre-set time period. Knowledge of a DGA, the time period and input seed allows a criminal to determine what unique domain name(s) will be required to be registered on a particular day in order to maintain command and control (C2) communications with malware infected computers. For defenders lacking such knowledge, it is very difficult to know what C2 domains will be used by individual malware samples, until live traffic can be observed (such as after malware detection and collection, in a sandbox). By which point, the criminal will likely already have registered the C2 domain for that day and ensured they have gained or maintained control of their botnet.
Since the time of submission of D4.2, we have had two very notable DGA threats appear that have had significant Internet-wide impact, showing how relevant DGAs remain to cyber security today. The number of malware families that continue to utilize DGA techniques has also grown.

3.1 Flubot DGA

One of the new notable DGA threats is Flubot[2], an Android banking trojan that started spreading in Europe at the start of 2021. It uses a DGA algorithm that generates thousands of domains per month for every seed. As of late January 2022, nine such different seeds are known.

domain
eoywqlhvvyaspov.com
mvdeebtofbvqnyh.su
gllkhhkxkhbpysy.su
odypyybdhstookn.ru
gduncmixshrsmod.ru
cistqphxkavqlbb.su
eosxwsgdcgnehej.su
qgeaiqnfnxlxdji.cn
bdaebuwihfhsxbl.com
rrgfphctvofguxe.ru

Figure 3-1 Sample Flubot domains for January 2022, for seed flubot_dga_0x0_15_5000_1_0_ru-su-cn

[2] https://malpedia.caad.fkie.fraunhofer.de/details/apk.flubot

Shadowserver sinkholes sample monthly Flubot domains every month as part of the EU HaDEA CEF VARIoT project[3] that is focused on IoT related threats, as well as sharing data from partner sinkholes with the Internet defender community.

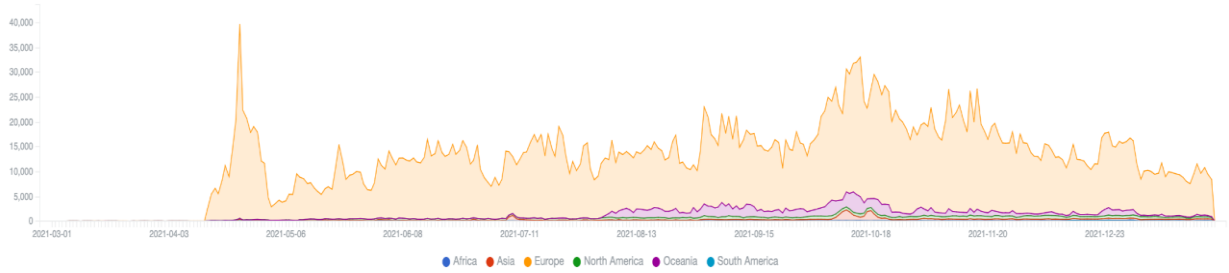


Figure 3-2 Flubot infections worldwide since Q1 2021 by unique daily IP addresses seen in sinkholes.

3.2 Solarwinds/SUNBURST malware

The second DGA related global threat is not a classic DGA use-case. In the 2020 Solarwinds intrusion, the Sunburst actor[4] used DGA domains to encode information about infected victims in the first stage of compromise. This is a different use case than the typical generation of pseudo random domains to achieve better botnet resilience to takedowns. Initial C2 activity was orchestrated through DNS to achieve stealth by using similar text patterns to those often used for CDNs/cloud services[5].

```
03l3eootsf8ifn9dhu4tslosf1ja0l2b.appsync-api.us-west-2.avsvmcloud.com
03lqu964p7ac9ohco0ohonuhrsoeuv0c.appsync-api.us-west-2.avsvmcloud.com
03m200v6ah1cg2ub13c60tvri.appsync-api.us-west-2.avsvmcloud.com
03ml6njbl2sg9lfcje0tjovte2fvip0c.appsync-api.us-west-2.avsvmcloud.com
03nshqnsvbm1qi8bd2oi60cvri.appsync-api.us-west-2.avsvmcloud.com
03nshqnsvbm1qi8bd2oi60ivri.appsync-api.us-west-2.avsvmcloud.com
03oc8p7h42p97e5ce5oh30ee5o2st0tv.appsync-api.us-west-2.avsvmcloud.com
03p6uofmo4601t5covirsvu10bhehet0.appsync-api.us-west-2.avsvmcloud.com
03p6uofmo4601t7covirsvu10thehet0.appsync-api.us-west-2.avsvmcloud.com
03pjdttjm1lpmgccufs0bvuio2vu10oe.appsync-api.us-west-2.avsvmcloud.com
03pjdttjm1lpmgqcufs0gvuio2vu102e.appsync-api.us-west-2.avsvmcloud.com
03pr9ut8d5shvnnbeovcr110oe2h.appsync-api.us-west-2.avsvmcloud.com
```

Figure 3-3 Example DGA-like encodings used in Sunburst.

Shadowserver worked with Law Enforcement to sinkhole all the SUNBURST domains involved in the original incident. It is worth noting that if monitoring of traffic for such suspect DGA-domains was common in the industry, the SUNBURST actors would have been detected much earlier than they were, and thus have been much less successful in their efforts.

[3] https://variot.eu

[4] https://www.mandiant.com/resources/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor

[5] https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/solarwinds-unique-dga

3.3 DoH and DoT

Recent years have seen the introductions of new protocols for DNS services - DNS over HTTP (DoH)[6] and DNS over TLS (DoT)[7]. These protocols were introduced to reduce the potential for eavesdropping and manipulation of DNS data through man-in-the-middle attacks. Unfortunately for Internet defenders, they also have the side-effect of not allowing security administrators to be able to easily observe and block DNS queries to potentially malicious domain names, which is a common security practice.

From our sandbox perspective, this means that we cannot see what domains are being queried by malware being executed either. We can however identify cases where DoH and DoT queries are being made (at least with a high certainty) and make an assessment of the scale of the problem, as well as observe general trends in usage (see Section 8 for more details). If the malware that is making such queries is reverse engineered, it is still possible to identify the domains in question and take them down. If a malicious DoH or DoT server is set up for the purpose of hiding domain queries, it is also possible to take down such a service, if such activity can be proven.

3.4 Cryptocurrrency Domains and other alternative DNS systems

Blockchain-based DNS[8] is an alternative to the traditional DNS system. In such a setup, the "DNS system" runs on the blockchain infrastructure of a cryptocurrency. An example is the .bit TLD which runs on Namecoin[9] infrastructure. To query a domain in the .bit namespace, a user has to have an application that can natively use the Namecoin blockchain or use a proxy that can translate traditional DNS queries to the blockchain.

The most popular cryptocurrency DNS systems aside from .bit include .bazar, .coin, .emc and .lib, which are all EmerCoin[10] domains.

Cryptocurrency domains are attractive from the point of view of attackers, as there is no central DNS record that can be taken down, allowing threat actors to operate seemingly with impunity[11]. In practice however, malware has often used the blockchain via DNS proxies, making the DNS proxies the "weakest-link" that can then be taken down. Shadowserver has worked with Law Enforcement to do this in a number of cases. Blockchain DNS, when used in malware, is often used as a backup method of maintaining control of the botnet - with the primary C2 channel being DNS-based, but crypto-DNS providing a secondary channel in the event that the malware's primary DNS channel becomes unavailable.

[6] https://datatracker.ietf.org/doc/html/rfc8484
[7] https://datatracker.ietf.org/doc/html/rfc7858
[8] https://blog.nameshield.com/blog/2020/04/08/dns-on-blockchain-the-next-evolution-of-domain-names/
[9] https://www.namecoin.org/
[10] https://emercoin.com/en/
[11] The Trickbot malware is one such recent example, https://malpedia.caad.fkie.fraunhofer.de/details/win.trickbot

4 The Shadowserver Sandbox Dataset Revisted

As was explained in D4.2, the malware used as a basis for the generated data is collected by Shadowserver primarily through reciprocal bulk data sharing agreements with AV/TI vendors and researchers/research groups as well as the Shadowserver sandbox itself (second-stage malware dropped and other artefacts retrieved during initial malware sample execution), honeypots and honeyclients, spampots and submissions from investigations.

At the time of writing this deliverable in late January 2022, Shadowserver's malware repository consists of nearly 1.7 billion unique samples. The daily average of unique samples ingested over the last three months is around 410 000 (out of which around 100k are executed in the Sandbox for a given day), for a total storage of nearly 6.2 petabytes.

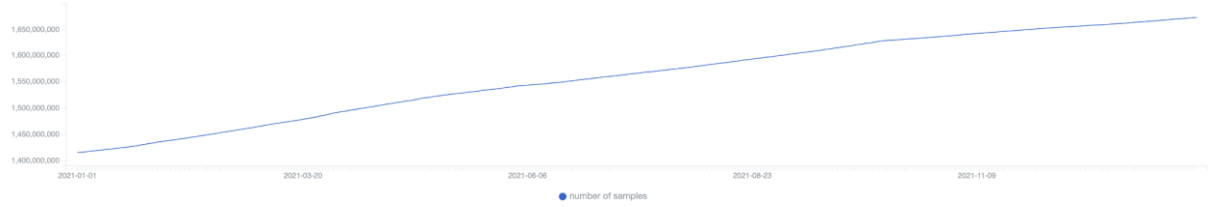


Figure 4-1 Number of Shadowserver malware repository samples over time.

Aside from storing the results of malware sandbox runs, which is the basis of work in T4.2, Shadowserver also indexes all predicted malware DGA domains programmatically generated for each day by 112 different malware families (as of the time of writing of this deliverable).

As can be seen, the number of malware families with DGA capabilities has grown considerably since D4.2 where there were 91, demonstrating the continuous need of tackling the DGA problem. This includes new findings made possible by SOCCRATES project activities (which are discussed later in section 11).

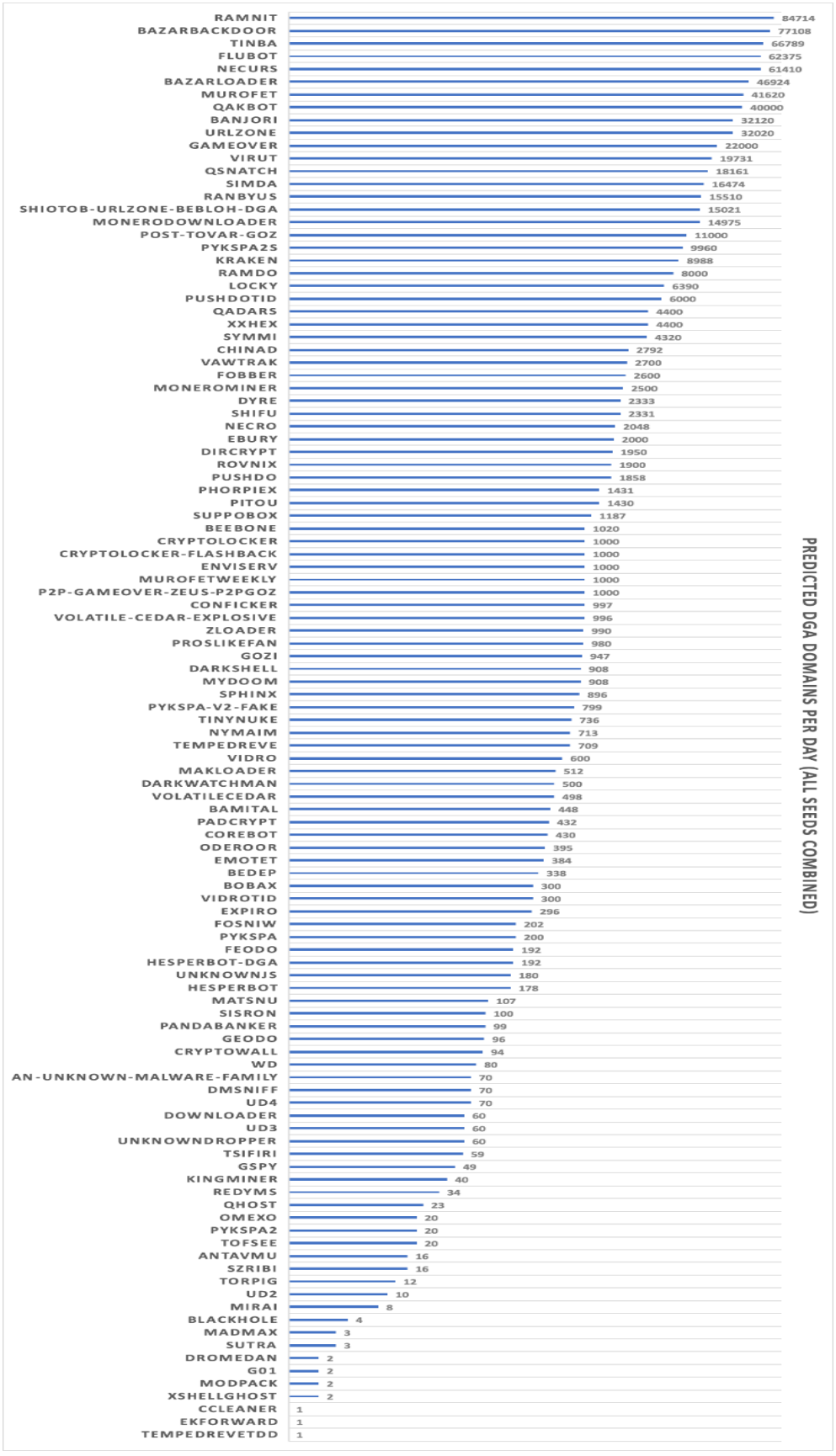


Figure 4-2 DGA malware families.

5 DGA Classification Methods and Pipeline

This section describes our different approaches to DGA classification, building on experience gained in the project.

5.1 Goals

Within the SOCCRATES project, we regard DGA detection as a way of collecting threat intelligence, and predicting new malware trends. Our goal was to develop a prototype DGA detector that can classify domains at `industrial' scale (the target is a few million unique domains daily) and label these according to the DGA or malware family which created them. The existing literature on DGA detection focuses particularly on machine learning-methods [1][2][3] that can either classify a given domain as being benign or malicious, or determine the malware family to which the domain belongs to. However, little attention has been paid to the deployment of these models in live environments, to enable detection at large scale. To address this major challenge, we adopted an integration-driven approach, in which we sought to first optimise the deployment of DGA detectors, and then improved the neural networks/classifiers that are running in the background.

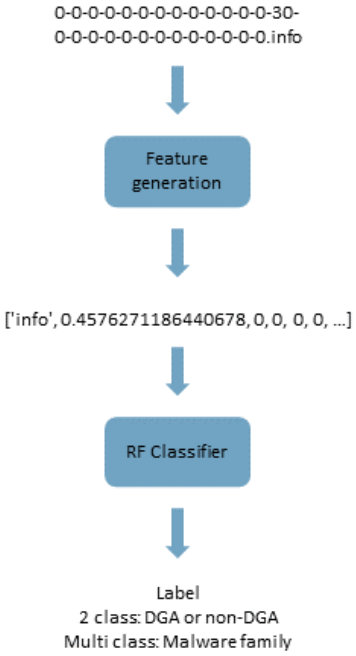
5.2 DGA classification methods

According to the most recent literature survey on this topic by Sivaguru et. al [4], DGA detection approaches can be split into two categories, depending on what type of input data is being used: feature-based or featureless approaches. In feature-based approaches, lexical or lexicographic information is extracted from the domain names to create new features. These features can include for instance n-grams, ratios of consecutive vowels or consonants, or the length of the top-level domain (TLD). On the other hand, in feature-less approaches, a character encoding is applied to the domain name, and the resulting numerical sequence is used as input for the respective algorithm. Both approaches have their own advantages and weaknesses. Feature-less approaches require less data preprocessing, and their results are difficult to explain, whilst feature-based approaches are explainable by design, but may benefit less from deep learning capabilities. We decided to pursue both options to investigate their potential and suitability for performing DGA classification at scale. Throughout the remainder of this chapter, both approaches are explained in detail, and some initial results are provided. We note that the training and evaluation data differs among the two proposed algorithms. That is mainly due to the different project timelines at TNO and AIT. To ensure a fair comparison between the feature-based and featureless approach, we provide a separate numerical experiment in Section 7.3 in which both methods were trained and evaluated on a commonly defined dataset.

5.2.1 Feature-based approach

The Feature-Based approach targets DGA detection and DGA family classification by exploiting a dataset of relevant features which describe the main characteristics of each domain name. These features are fed to two different Random Forest classifiers, trained i) to discriminate between benign and DGA domains and ii) to perform a finer-grained classification recognizing which malware family generated each domain. The full pipeline is depicted in Figure 5.1.

The chosen approach is based on the assumption that benign and DGA domains show different *linguistic features*, as suggested in [5]; Overall, we extract 11 features:



- **Domain Length:** often domains of the same length are generated within the same DGA family;
- **Top Level Domain:** a single family may use a single tld (OneHotEncoder needed);
- **Numerical Characters Ratio:** portion of numerical characters in the domain, it is usually higher in DGA domains than in benign ones;
- **Special Characters Ratio:** portion of special characters in the domain (e.g., &,+, *,_, -, etc.), it is usually higher in DGA domains than in benign ones;
- **Meaningful n-grams**: we extract 3, 4, 5 and 6-grams from the original domain, and calculate the number of meaningful (i.e., existing in the english vocabulary) ones. Benign domains generally show a higher number of pronounceable n-grams, being less often randomly generated;
- **String distance (Levenshtein) statistics:** Mean, Standard Deviation, Maximum and Median of sample distance with respect to the rest of the dataset (samples belonging to families with domains from the same root are closer to each other than to the rest of the dataset).

Figure 5-1 Feature-based pipeline.

The feature generation phase is significantly slowed down by the extraction of the pairwise Levenshtein distance among domains. The classic Levenshtein distance has a complexity of $O(n^2)$, given strings of length n, which causes a drastic increase in the feature computation time as the dataset enlarges. Given the limited computational power of our testbed, we limit both datasets to a number of samples in the order of 10^4, we train the Random Forest models on the 70% of the available samples, and leave the remaining 30% for testing.

5.2.1.1 Binary classification

The aim of this task is to train a binary RF classifier to distinguish between "Benign" and "DGA" labels. We test the methodology on a balanced dataset (i.e., the same amount of benign and DGA domains is fed to the model) of 20,000 samples. The feature-based approach achieves an accuracy of **0.93** on the test set.

	Precision	Recall	F1-Score
DGA	0.92	0.92	0.92
Benign	0.93	0.93	0.93

Table 5-1 Binary classification report for feature-based DGA detection.

Table 5.1 reports the set of classification metrics. For this task, using a balanced dataset is fundamental to achieve a correct DGA detection. Indeed, we observe a foreseeable progressive degradation of the detection performance as the dataset imbalance grows. In the following we show that similar considerations do not hold in the family classification case.

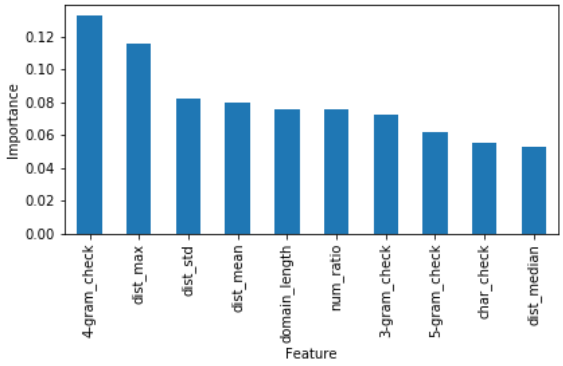


Figure 5-2 Feature importance for RF binary classification.

Figure 5.2 shows the top-10 features relevant for the DGA detection. The most useful features for this task appear to be those addressing the domain pronounceability (N-gram_check), the domain length and the ratio of special and numerical characters, as well as the distance statistics. The TLD does not appear as a discriminative feature for this application.

5.2.1.2 Family classification

Different considerations hold for the Family Classification task. In this case, the RF model is trained to discriminate among 78 classes: the "benign" one and other 77 malware families. We cap the dataset to 1,000 samples per class, or take the entire class in case it has less than 1,000 samples in total. Given the harder task, we record a general decrease in the classification performance, leading to an accuracy of **0.63**. Table 5.2 shows the remaining metrics.

	Precision	**Recall**	**F1-Score**
Micro AVG	0.63	0.63	0.63
Weighted AVG	0.61	0.62	0.61

Table 5-2 DGA Family classification report for feature-based case.

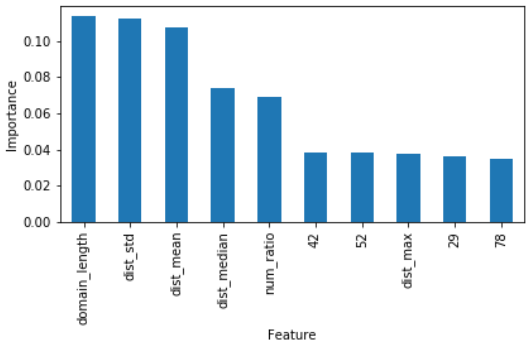


Figure 5-3 Feature importance for RF family classification.

5.2.2 Featureless approach

The featureless approach taken in this project relies on Temporal Convolutional Networks (TCN), which receives as input character encodings of the domain names. TCNs are a category of

convolutional networks particularly suitable for modelling long-term dependencies in sequential data [6][7]. They have been shown to outperform recurrent architectures on many different sequence classification tasks such as the adding problem and image classification on sequential MNIST and P-MNIST [6]. This network is an improvement on the LSTM-based approach which was first introduced in Deliverable 4.2: Threat Identification and Threat Trend Prediction – Initial Prototype.

The classification process is depicted in Figure 5.4. At first, a few pre-processing steps are applied to the domain name. The Top Level Domain (TLD) is removed, the subdomains are extracted and then each subdomain receives a numerical encoding. In other words, each character is mapped to a positive integer. The list of characters which are considered includes all digits, letters in the English alphabet (no distinction is made between upper and lower case) and symbols ".", "-" and "_". Finally, this numerical encoding is padded with zeros until a maximum specified domain length is reached. This last step ensures that all input data samples have the same size. This encoding is then passed to the TCN which can classify a domain as generated by a DGA or not (for binary classification), or classify it according to the malware that created it (for multiple label classification).

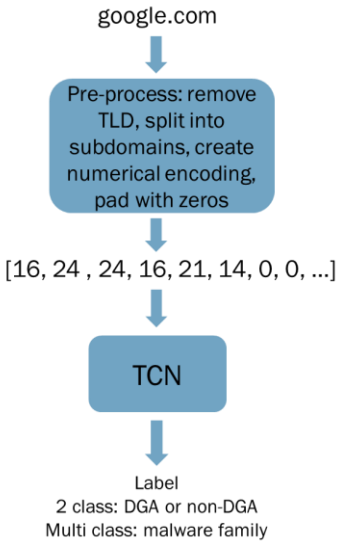


Figure 5-4 TCN-based classifier structure.

The core element of a TCN is the residual block, which implements dilated (causal) convolutions. Figure 5.5 shows the generic architecture of a TCN (left) as a stack of *n* residual blocks. Such a residual block contains two dilated causal convolution layers (marked in Figure 5.4 by the blue 'basic layers') each having a specific number of filters *num_filters* and the same dilation factor *d*. The Rectified Linear Unit (ReLU) is used as an activation function on these layers. Weight normalisation and a dropout layer can be optionally added, when the problem at hand benefits from this. The filter size is kept at the same value of *k* across all residual blocks. A skip connection is also enabled, which adds the output from the previous layer to the next layer. This is marked by the element-wise addition $\oplus$. Due to zero-padding, this operation may receive inputs that differ in shape. To circumvent this, a 1x1 convolution is added. A dilated (causal) convolutional layer is shown in the right side of Figure 5.5. Simply put, the filter size *k* specifies that the value of a neuron on the output layer (top layer) is influenced by *k* neurons in the input layer (bottom layer). The dilation factor *d,* indicates that the convolution filter (represented by the black arrows) skips *d* neurons when sliding over the input sequence. Moreover, this filter is only slided over previously observed input values, hence the name of "causal" convolution.

Different TCN models are created for the two classification tasks, since they significantly vary in complexity. In the following sections we elaborate on the neural network architectures and the results obtained.

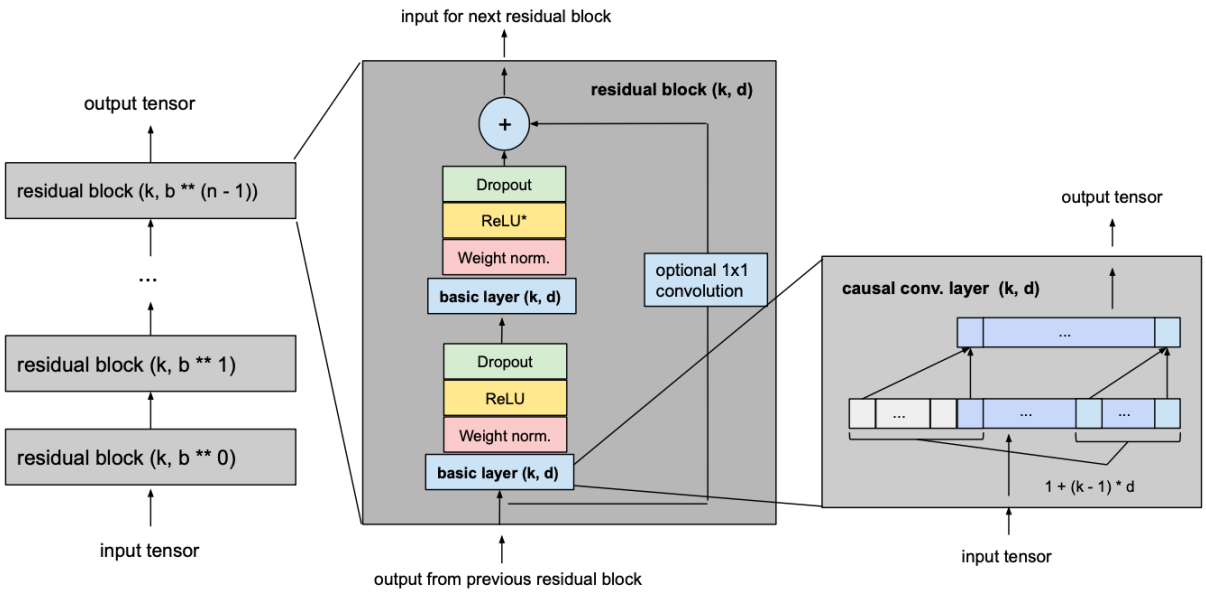


Figure 5-5 Structure of a TCN, from left to right: stacked residual blocks in a TCN, layers composing the residual block and representation of a causal convolutional layer (image source: https://unit8.com/resources/temporal-convolutional-networks-and-forecasting/)

5.2.2.1 Binary classification

The binary TCN classifier consists of:

- one embedding layer: it projects the sequences into a higher dimension space to allow for more degrees of freedom;
- one TCN residual block with 8 filters, a filter size *k=4*, and 6 dilated convolutions *d=[1,2,4,8,16,32]*;

output layer: produces the result for the given input (the predicted class based on sigmoid function). The output for the binary classification is either 0 (non-DGA) or 1 (DGA).

This neural network was trained on a labelled dataset provided by the Shadowserver Foundation which included approximately 450.000 domain names (50% benign and 50% malicious, generated by DGAs corresponding to 80 different malware families). We performed a 5-fold cross validation. For each data fold, the models were trained until no improvement was achieved in the validation loss for the last 20 epochs. We selected the best resulting TCN model and compared the results of the TCN with the initial LSTM classifier presented in Deliverable 4.2 using a new test set consisting of 2.9 million domain names. The results are summarised in Table 5.3. We note here that the labelling was done on the full domain name (not on subdomain level) and that the TLD was removed in both training and testing phase.

	Accuracy	Precision	Recall	F1-Score
TCN	0.969	0.960	0.957	0.958
LSTM	0.962	0.951	0.946	0.948

Table 5-3 Binary classification report for featureless DGA detection.

Table 5.3 shows that the TCN model outperforms the previous LSTM classifier on all metrics considered. Whilst the improvement factor may seem minor, in the range of 1%, the practical impact is much higher. For instance, in this experiment, the TCN reduces the number of false positives by 10.000 domain names compared to the LSTM. This also shows that one residual block in the TCN is sufficient to model and distinguish traits of benign and DGA-generated domains. A more detailed evaluation of this classifier (including generalisation capability) is discussed in Section 7.

5.2.2.2 Family classification

The family classification (multi-label) task is more difficult and requires a more complex neural network in order to be able to distinguish between 80 different malware families. To this end, we created a classifier with the following structure:

- one embedding layer: it projects the sequences into a higher dimension space to allow for more degrees of freedom. This is the same as in the binary classification scenario.
- one TCN residual blocks, having 64 filters, a filter size *k=16*, and 4 dilated convolutions *d=[1,2,4,8]*. Additionally, after each dilated convolution we added a dropout layer, which discards 10% of the input units.
- output layer: produces the result for the given input (the predicted class based on softmax function). The output label for the family classification is either “benign” or the name of the malware family.

Similar to the feature-based approach, we tried to create a somewhat balanced training dataset i.e. for each of the 80 malware families available, we selected at random 1000 domain names. Certain families had less than 1000 samples, so we applied a 95% train/5% test split. There were in total 52 families which had at least 1000 samples. After this step, all remaining domain names were used to create a test set. The model was trained for a fixed number of 100 epochs. In the initial experiments, we included benign domains from Alexa/Cisco/Quantcast repositories as well, however this resulted in poor overall performance (weighted F1=0.3). Therefore, we decided to remove this class from the model. In practice, benign domain names can be filtered out using the binary classifiers. Furthermore, based on DGA detection literature [8], we have added 1000 benign NX domains (collected from the corporate network at TNO) and kept the TLD since this practice was expected to improve the classification performance.

Training set	Averaging method	Precision	Recall	F1-Score
all 80 malware families	weighted AVG.	0.916	0.864	0.883
	micro AVG.	0.582	0.837	0.675
52 well represented families	weighted AVG.	0.917	0.875	0.889
	micro AVG.	0.698	0.834	0.721

Table 5-4 Family classification report for featureless DGA detection.

When training on the full dataset, the TCN classifier achieved an overall accuracy of **0.864**, and when the training set is restricted to the well-represented families only, this increased to **0.875**. Note that for both situations the test set is of approximately 1.2 million domains. Table 5.4 shows the rest of the performance metrics. These results are comparable to the state-of-the-art approaches in DGA detection such as ResNet [8], but we see that the classification performance drastically varies across different malware families. We elaborate on this aspect in Section 7.

6 DGA Classifier Implementation and Deployment

This section describes how our DGA Classifier - the DGA Detective - was implemented and deployed in production at Shadowserver.

6.1 DGA Detective

DGA Detective is a software deliverable developed within SOCCRATES WP4 to help defenders identify and classify DGA domains. It has been open sourced and it is currently being used by three organisations involved in the project: Shadowserver, Mnemonic and F-Secure. It is available at https://github.com/cossas/dgad under Apache 2.0 licence.

The core of DGA Detective is dgad[12], a python package that implements functionality to perform training and classification of DGA domains. It comes with a default model trained with Shadowserver data, allowing a user that is lacking a training set to quickly be able to classify domains. Dgad allows to classify domains from the command line (one at a time) or in batch from a csv file. Furthermore, dgad also includes code to easily run a server with an API to classify domains over a network.

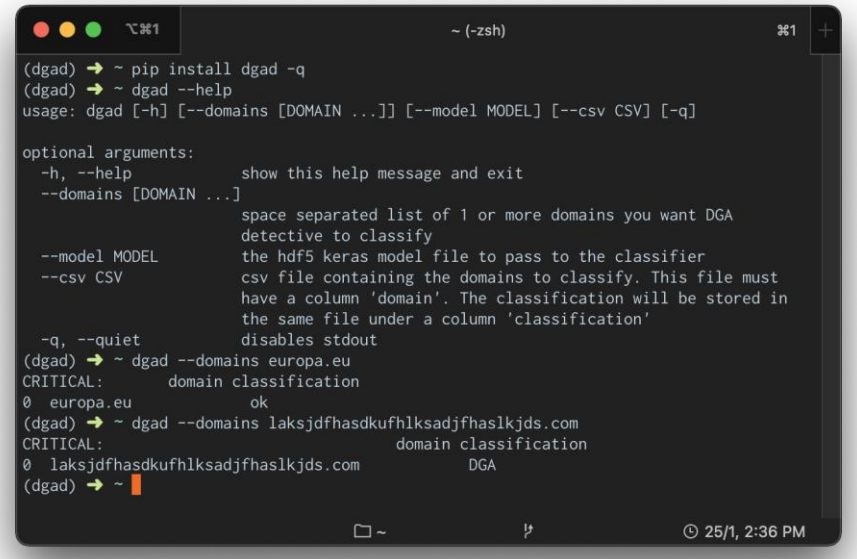


Figure 6-1 DGA Detective - example execution.

[12] https://pypi.org/project/dgad/

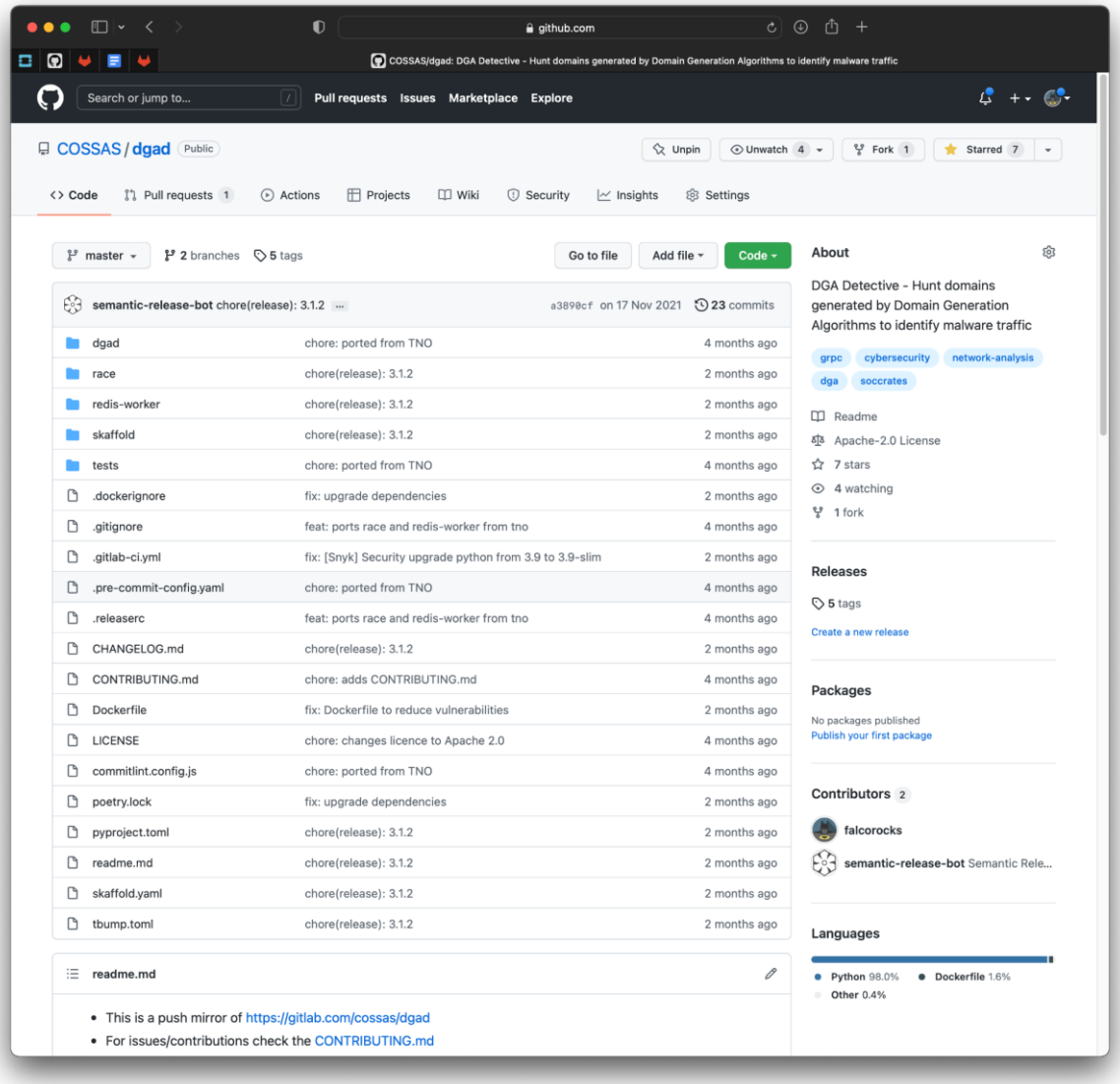


Figure 6-2 DGA Detective (dgad) opensourced on github.

6.2 Implementation

The dgad python package relies on the python libraries tensorflow[13] and keras-tcn[14] to perform the training and classification of DGA domains. It provides custom classes to model DGA domains and custom methods to preprocess raw data. The package requires python > 3.6.1.

Performance is a key feature of DGA Detective. Shadowserver needs a tool capable of classifying millions of domains daily. At the same time we have kept the implementation as simple as possible to allow users to easily plug the tool into their toolchain. We achieve this through an included API. This means that users can use dgad either as a python package directly in their python scripts or deploy a dgad server to classify domains through its API. This latter functionality is implemented with gRPC[15] a high performance API framework that allows to automatically generate client and server code in a

[13] https://pypi.org/project/tensorflow/
[14] https://pypi.org/project/keras-tcn/
[15] https://grpc.io

number of languages. We provide code to deploy a python server and an example of a python client, but the user can easily generate code for other client languages of choice (Javascript, Golang…) since the gRPC protocol definition is provided. The gRPC server exposes a single API to classify a domain. The user sends a domain to the endpoint and classification information is returned.

DGA Detective is distributed as a container image to achieve scalability in an operational setting. This enables production grade deployments with container orchestration solutions like Kubernetes. Depending on their needs, the user can deploy one or more containers running the DGA Detective gRPC server behind a load balancer. The user should then send domain classification requests to the load balancer, which will spread the load among all the containers active. This setup can easily scale to classify any amount of domains because the user can increase the amount of containers at any time.

6.3 Deployment in the production environment

Early versions of DGA Detective (and under earlier names) were installed in the Shadowserver production environment starting at the end of January 2021. These versions were at different times debugged and had to be updated or rewritten to function under the load of a production environment, having to handle around 300K domain queries per day on average.

For these purposes, DGA Detective was configured to run on a dedicated VM with 8 virtual CPUs, 16GB of memory and a 170 GB disk. Since mid-June, DGA Detective has been able to run in a stable manner, tagging potential DGA domains as “tno-dga-tagged”.

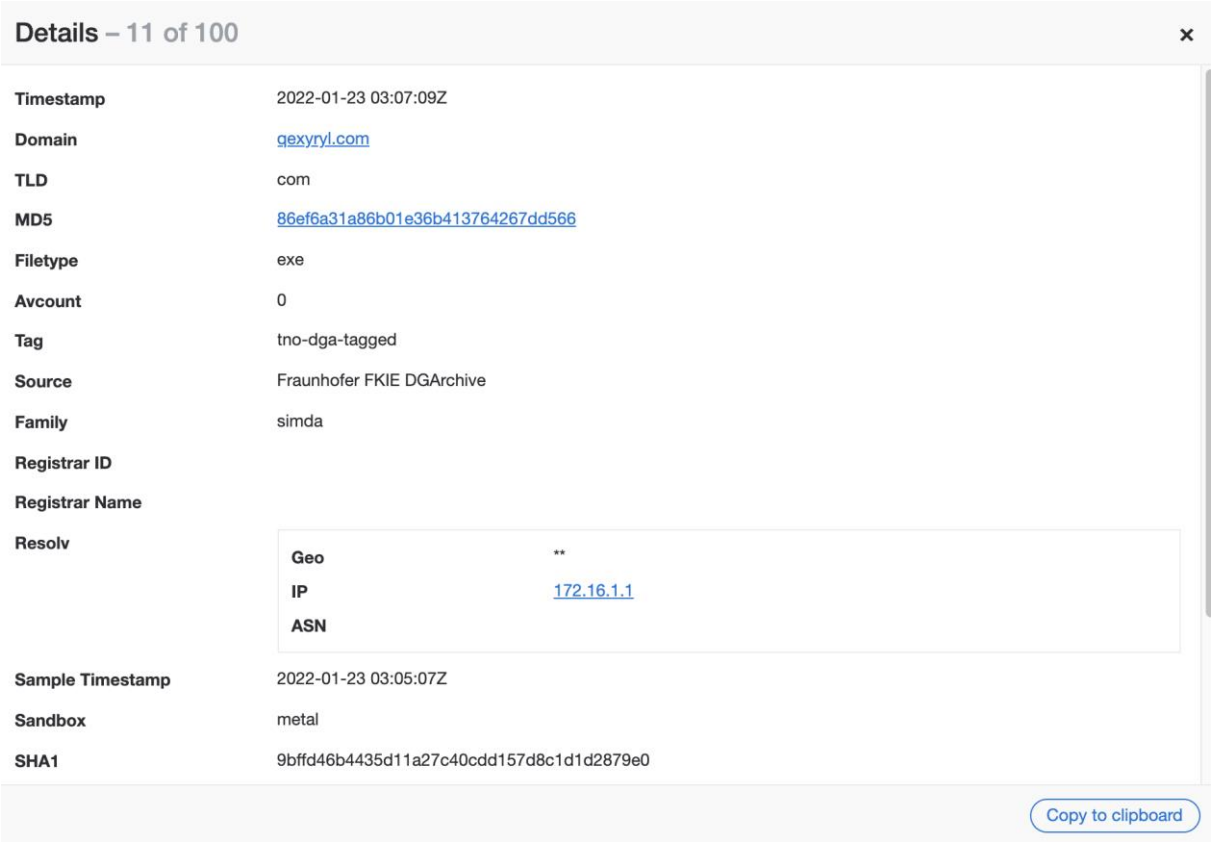


Figure 6-3 Example of a DGA domain found in Shadowserver production sandbox(automatically tagged/labelled as “tno-dga-tagged”).

7 Evaluation of DGA domain classification results

This section provides a deeper evaluation of DGA domain classification results obtained through the various feature-based and featureless machine learning methods that were described in Section 5.

7.1 Feature-based approach

This subsection provides a more detailed overview of our results and discusses the advantages and limitations of the feature-based approach introduced in Section 5.2.1.

7.1.1 Binary classification

Recall from previous sections that we evaluated the performance of a Random Forest classifier on a balanced dataset of 20,000 samples, of which 70% were used for the training, 30% for the testing (a total of 6,600 samples). Figure 7.1 summarises the algorithm output in the form of a confusion matrix: the majority of tested samples (93% of both classes) are correctly classified. A manual exploration of the misclassified domains allows us to identify the categories of domains that are more likely to fool the algorithm.

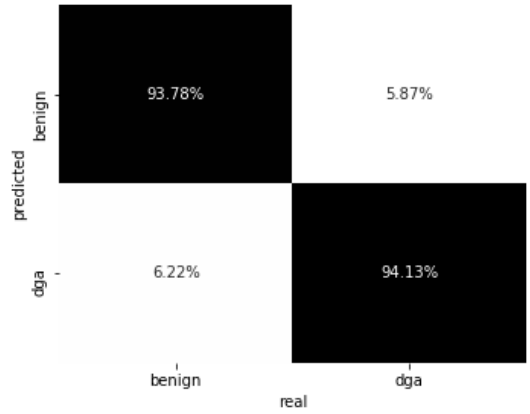


Figure 7-1 Confusion matrix for the binary classification case.

We observed that the majority of misclassified DGA domains contain proper words, e.g., `till-dont.ru`, `facewhere.ru`, `wheelgone.net`. Indeed, misclassified DGAs show on average a higher meaningful n-grams count (1.62 meaningful 3-grams, 0.78 meaningful 4-grams, 0.21 meaningful 5-grams vs 0.67, 0.1 and 0.02 respectively, in the correctly classified case).

domain
ugbgqltphwpdqnwqmmm.tw
2mzqqgq8.micpn.com
ie2sdspp.cdnppb.net
kyxnya.com
mdxxuif.me

We also observe the opposite: Figure 7.2 reports an example of misclassified benign domains. Many of these samples show several characteristics which are typical of DGAs, such as long sequences of random characters (highlighted in blue), the presence of numerical characters (in green) or short domains with no meaningful n-grams (in red).

Figure 7-2 Misclassified benign domains

7.1.2 Family classification

Figure 7.3 reports the confusion matrix obtained for the family classification task. More than 47% of the families obtained an F1-Score higher than 0.7. 17 classes (~3.4% of the samples in the test set) got misclassified in full. Recall from Section 5.2.1.2 that errors in classification are more common in underrepresented families or in domains lacking a fixed structure.

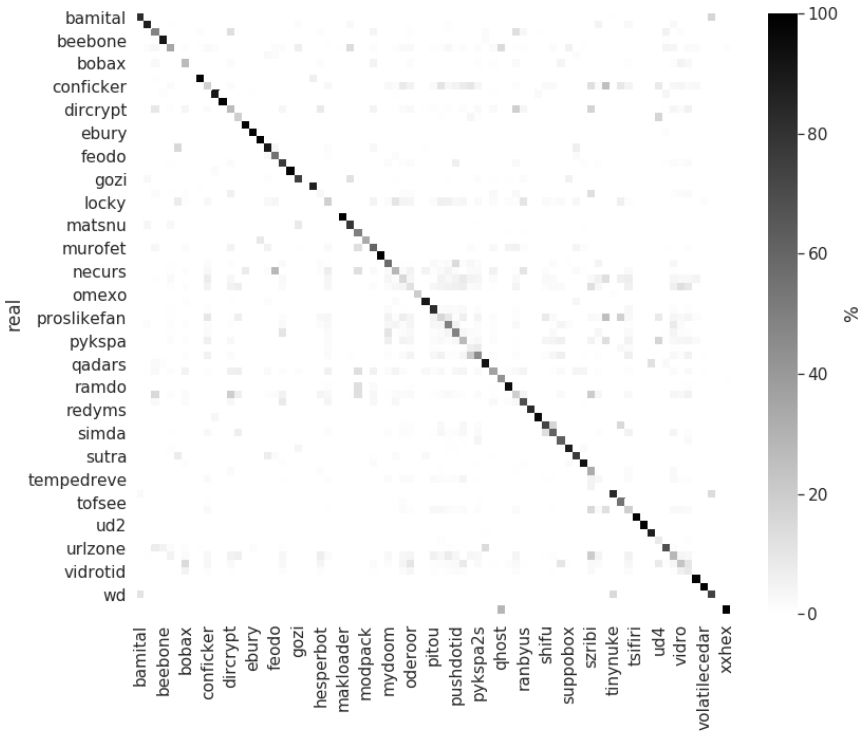


Figure 7-3 Confusion matrix for the family classification case.

Figure 7.4 confirms the first assumption. We observe that the classes obtaining the lowest F1-Score values in the testing are those having a smaller support in the training phase (30 samples per class in median). The underrepresented families included in the best categories (F1-Score higher than 0.8), instead, are those showing a very clear structure.

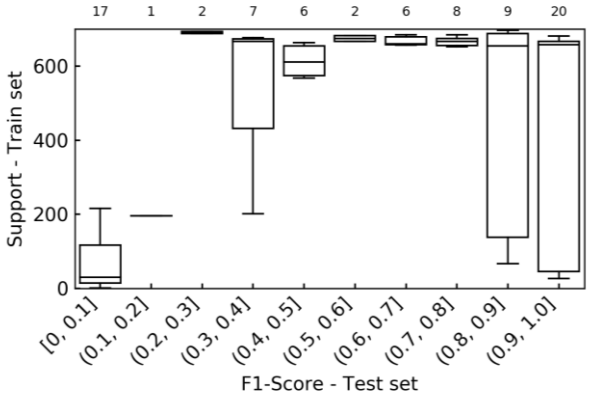


Figure 7-4 Distribution of training support set size per F1-Score bin.

Example of correctly classified small families are *tsifiri*, which shows domains almost exclusively composed by numbers and special characters, e.g., `0-0-0-0-0-0-0-0-0-0-0-0-0-30-0-0-0-0-0-0-0-0-0-0-0-0-0.info`, and downloader, for which all the domains in the test set share the same subdomain.

Among the wrongly classified families we mention *blackhole*, a small family whose domains ending with .ru are mapped as *sutra*, a larger family using the same tld. Families like *vidro*, *vidrotid* and *bobax*, often use the same subdomain and tld, getting frequently confused with each other.

7.2 Featureless approach

In this subsection we discuss in more detail the performance of the models proposed in Section 5.2.2, in order to better understand the results and later on assess their generalisation capability (to correctly classify domain names unseen during the training process).

7.2.1 Binary classification

Recall that the binary TCN classifier presented in Section 5.2.2.1 was trained on approximately 450.000 domain names, with a 50%-50% split between benign and DGA domains, and tested on a very large dataset of 2.9 million samples. Figure 7.5 depicts the resulting confusion matrix, showing that the model is able to classify the majority of benign and malicious samples correctly, with an accuracy greater than 95% and 97%, respectively.

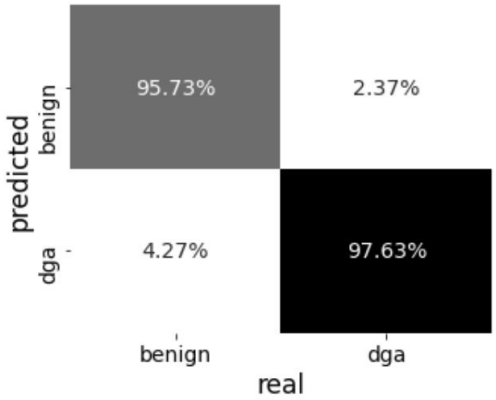


Figure 7-5 Confusion matrix for the binary classification case.

When inspecting the misclassified samples, we observed several interesting patterns. First of all, many of the misclassified benign samples are domain names in different languages such as `cuoi-hoihoangquan`, `kysuketcau` (both in Vietnamese). This is to be expected, since most benign domains in the training datasets are in English. Secondly, the training set includes mistyped domains. Some of these domains are also misclassified, for instance `fceajspp` (generated from `faceapp`). This can be tackled in the future through an additional preprocessing step, which would filter all mistyped domains and report them separately to the analyst. Other benign domains which are wrongly labeled as DGA either include digits in an unexpected sequence (`v3trckngnow2`) or contain 20

characters or more (`dishwasherrepairnearyou`). The DGA-generated domain names which are misclassified, contain, just as in the case of the feature-based approach, one or more proper words (e.g. `etonlyqu`) **.** Finally, we noticed that very short DGA domain names (of 5 or 6 characters and including at least 2 vowels) are also often wrongly labelled as benign.

7.2.2 Family classification

In this subsection we briefly discuss some of the family classification methods introduced in Section 5.2.2. We focus in particular on the experiment performed on the full-dataset (with 80 malware classes and one class corresponding to benign NX-domains). Figure 7.6 shows the full confusion matrix obtained for this family classification task. For more than 52% of the classes the model

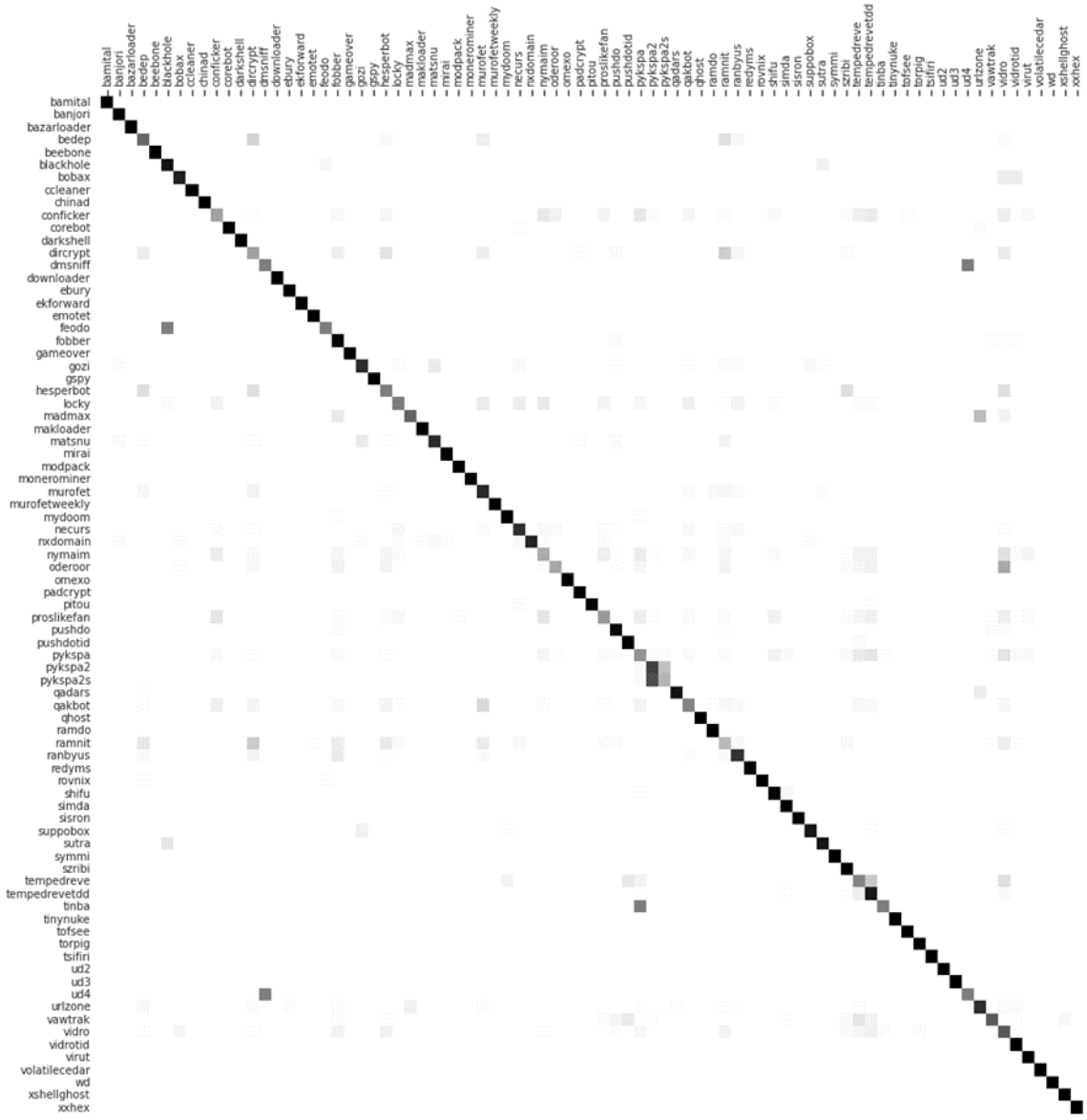


Figure 7-6 Confusion matrix for the family classification case.

achieves an F1-score greater than 0.8. There are 5 malware families ('ccleaner', 'downloader', 'makloader', 'omexo' and 'tsifiri' for which a perfect F1-score is obtained. As it can be seen in Figure 7.7, under-represented malware families in the training set are most often misclassified. However, there exist under-represented classes which show a very clear domain name structure which are easy to identify by the multi-class TCN.

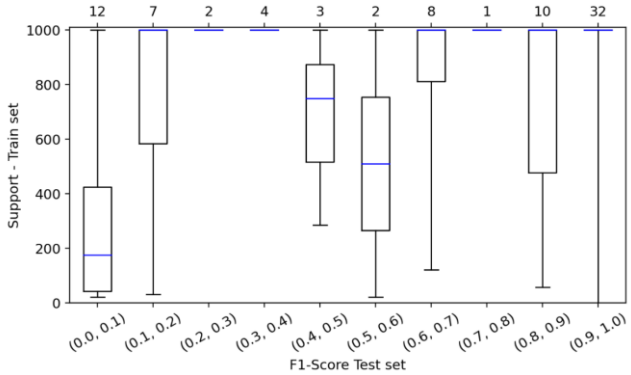


Figure 7-7 Distribution of training support set size per F1-Score bin.

Looking at the confusion matrix in Figure 7.6, the most striking misclassifications that occur are as follows:

- Domain names from the Pykspa2 class are confused with Pyksa2s, and vice-versa. The domains generated by these two malware families are very similar, but are clearly distinguished from the other classes because of some specific traits: varying domain length, usage of various TLDs such as *.biz, *.cc. *.net and *.info.
- Samples generated by 'dmsniff' are classified as 'ud4' and vice-versa. These classes generate very similar domain names, in which the first two characters are 'sn' or 'al', and the domain length is exactly 8 characters. Furthermore, these two malware families also employ the same TLDs: *.in, *.net, *.ru, .*org and *.com. We suspect these two classes might employ the same DGA algorithm, perhaps with different parameters (e.g. `albdfhln, alcgkown, aldjpvqt, alemuoun , alfpmrnq, algspvqt`, where the first, third and fifth domain in the sequence are from the 'dmsniff' class, and the rest from 'ud4').
- Similar to the results obtained in the feature-based approach, malware families which use the same TLDs are also confused with each other. An example thereof are domains from the 'blackhole' class, which are misclassified as 'feodo' since both use the TLD *.ru.

7.3 Comparison of approaches

In this section we compare our proposed feature-based and featureless approaches in two ways. First, we assess the performance of the best binary classifier of each kind on an unseen dataset. In this context, "best" is defined as the model which resulted in the highest accuracy during testing on all the data available at that point in time. Since TNO and AIT have had slightly different timelines, the train and test dataset is not identical for the two binary classifiers. Second, we compare the performance of the two approaches on the family classification task, on a jointly selected dataset, to identify advantages, disadvantages and complementarities.

7.3.1 Binary classification on unknown domains

Both binary classification methodologies introduced in Section 5.2 were evaluated on two recent DGAs: Flubot (Section 3.1) and Sunburst (Section 3.2). Domains from these malware families were

not included in the training dataset so are completely unknown to the model. For the Flubot DGA we used an evaluation set of 10000 samples, whilst for the Sunburst DGA we had at the time 5000 domain names available. The classification results can be seen in Table 7.1 and Table 7.2 for the feature-based and the feature-less approach, respectively.

The Flubot DGA is detected by both methodologies with high accuracy, even though it is completely unknown to the model. For the featureless approach, we noticed that the few domain names that were misclassified, contain more vowels (median = 6.0, mean = 6.05, std = 1.26) compared to the rest (median = 4.0, mean = 3.73, std = 1.60) and contain one or more proper words (e.g. sitchromsvkghal.cn, almanarnebttaic.cn, freasomedpmfitn.com).

Evaluation set	True Positives	False negatives	Accuracy
Flubot	9863	173	0.986
Sunburst (full domain)	0	5000	0
Sunburst (subdomains)	4156	844	0.831

Table 7-1 Binary classification report for Flubot and Sunburst DGAs - Feature-based case.

Evaluation set	True Positives	False negatives	Accuracy
Flubot	9937	63	0.993
Sunburst (full domain)	1	4999	0.0002
Sunburst (subdomains)	5000	0	1

Table 7-2 Binary classification report for Flubot and Sunburst DGAs - Featureless case.

The Sunburst case was far more challenging. When using the full domain name for labelling, we obtained very poor classification results. The only domain classified correctly in this process was unusually long:
174utqcr31cn293c6d6n0o6j0oeu.appsync-api.us-dmfd67ucs3n4mc4r6d6n0o6j0oeu.appsync-api.us-east-1.avsvmcloud.come8h2vel9711jjcpte3so6iore1ovoe0t.appsync-api.us-east-1.avsvmcloud.com-1.avsvmcloud.com

At a closer look, this poor classification result can be explained by the fact that in Sunburst-generated domains only the first subdomain is obviously suspicious, whereas the following three/four subdomains look rather benign (e.g. *03l3eootsf8ifn9dhu4tslosf1ja0l2b.appsync-api.us-west-2.avsvmcloud.com*). The first subdomain is not randomly generated but instead encodes information about the infected computer. This domain name structure is difficult to classify (the Sunburst malware was detected 8 months after the initial intrusion).

For this particular DGA, we came to the conclusion the we should apply a different labelling procedure: first split the original domain into subdomain, label each subdomain individually, if at least one of the subdomains received the "DGA-generated" label, then the full domain name is labelled as such. With this procedure, we were able to significantly improve the performance of both methodologies, achieving 100% accuracy on the Sunburst evaluation set in the featureless case. This finding, while preliminary, suggests that applying classifiers to both the full domain name, and the extracted subdomains may drastically increase the detection rate.

7.3.2 Comparison of the two methods on the family classification task

To allow a fair comparison between the two methods, they were both trained on an identical dataset consisting of 32880 training samples and 16195 testing samples from 77 malware families. Figure 7.8 shows the F1-scores of malware families on which the featureless approach achieves at least 0.1 improvement over the feature-based approach. This is the case for 39 classes, and in some cases such as '`gspy`', '`omexo`', '`torpig`', '`ud2`', the difference can be as large as 0.9. We do note however, that the support in the train set for these samples is also small. There are a few classes for which the feature-based approach achieves better classification, as it can be seen in Figure 7.9. For these specific malware families, the Random Forest classifier is able to slightly outperform the multi-class TCN, the largest difference being a 12% improvement in the F1-score for the '`ekforward`' family. This particular malware family generates domains which contain many digits, an aspect which is easier to model with an explicit feature (such as the number of digits in the domain name). The overall better performance of the featureless approach can be explained by the fact that the deep learning algorithm can extract complex patterns that are more representative for the structure of the domain names, in comparison to the features employed by the Random Forest classifier. Moreover, this experiment suggests that a combination of these two approaches may lead to an overall better classification. Enriching the list of features in the Random Forest classifier will likely lead to higher classification accuracy.

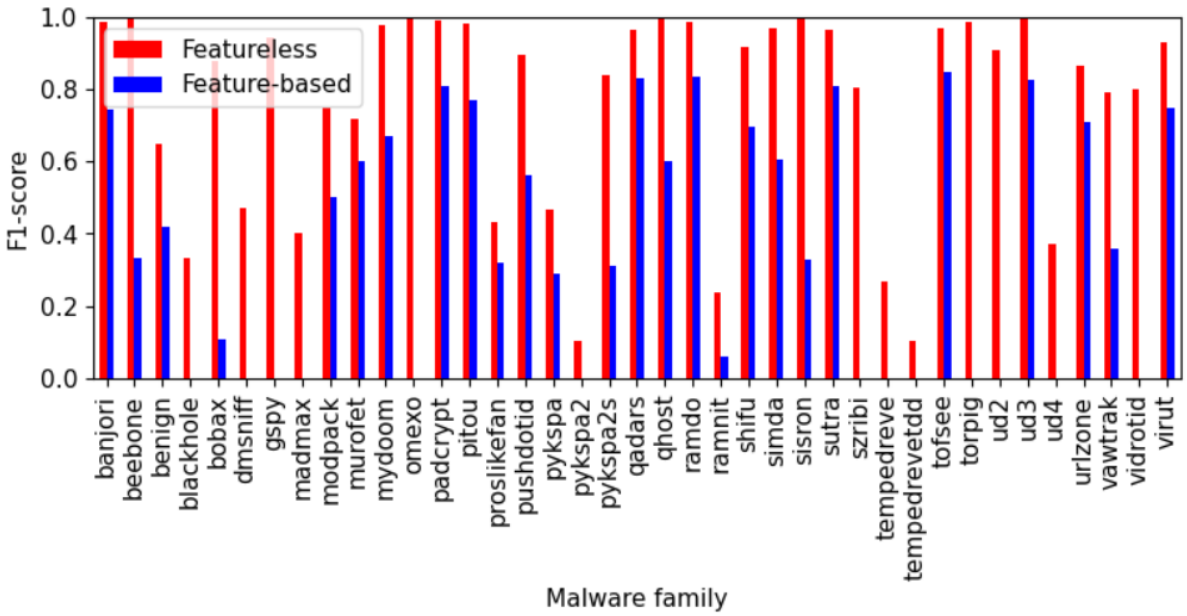


Figure 7-8 F1-score for malware families on which the featureless approach performs better.

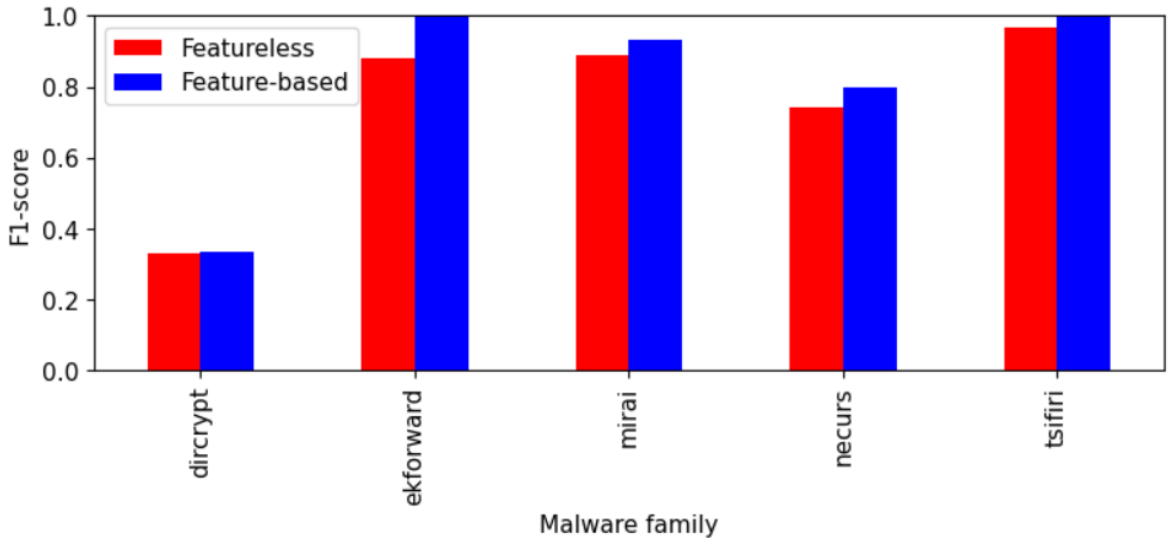


Figure 7-9 F1-score for malware families on which the feature-based approach performs better.

7.4 Lessons learned

All in all, we observed that both methods show good DGA detection capabilities, with some limitations due to the specific structure of some of the analysed domains (e.g. those belonging to the `Sunburst` family). This problem can be overcome by applying the subdomain classification strategy described in Section 7.3.1: we showed that the subdomain classification significantly improves the detection capabilities in presence of complex domain name structures, especially in the featureless case.

Family classification proved to be a harder task. As already mentioned in Section 7.1.2 and 7.2.2, the performance of both models are heavily dependent on the number of samples available for each family. The worst scores are yielded in case of underrepresented families, as well as families showing a very variable structure. In many cases, samples coming from underrepresented families are assigned to larger classes having a similar structure (e.g., using the same TLD or having a comparable domain length).

The featureless approach entails training a deep learning model and requires appropriate computational resources. All the results for the TCN binary and family classifiers were obtained using a spot virtual machine (VM) in Microsoft Azure. Specifically, we employed one NCv3 VM with 112 GiB of memory and one GPU (V100 card)[16]. The training process was quite rapid, with the family classifier learning a sufficient representation of the domains in less than 10 minutes. However, this spot instance is quite unstable, since it can be automatically deallocated when costs exceed a certain threshold. Because of this, it was not possible to perform a complete hyperparameter tuning of the neural networks, which might have an impact on the current performance.

In the feature-based case, while the model training and parameter tuning were relatively fast tasks, requiring less than 10 minutes as well, the most computationally intensive operation was the feature extraction, and the pairwise string distance in particular. The entire process runs on a local machine equipped with one Intel® Core® i7 processor providing 4 cores and 32 GB of RAM. This setup strongly

[16] NCv3-series - Azure Virtual Machines | Microsoft Docs

impacted the required computation time, that was growing up to days, forcing us to limit the dataset size. Distributing the distance matrix computation, or relying on other more scalable techniques may be a viable way to address this issue in the future.

In Section 7.3.2 we showed that although the featureless approach achieves better classification results on the majority of families, several classes exist that benefit in particular from the feature-based approach. Moreover, when inspecting misclassified samples from families such as '`ud4`' and '`dmsniff`', we discover new distinguishing patterns such as the first two characters, or the inclusion of a specific letter ('ud4' domains do not contain the letter 'b'), which can be included in the feature set. These findings indicate that combining the two approaches, either in the form of an ensemble classifier, or a multi-input neural network, has great potential to achieve accurate family classification.

8 Threat Hunting : Tools for Detecting Anomalies and Trends

Shadowserver built an internal dashboard and a number of support mechanisms to better explore anomalies and trends in our sandbox data, with a focus on malicious domains and DGAs. These include the results supplied by DGA Detective running in an operational environment. The goal is to have a set of tools supported by machine learning to enable effective threat alerting and hunting, which can also be used for the upcoming SOCCRATES pilot implementations as well as future post-project exploitation of results.

The internal dashboard signals anomalies and trends in the following DNS related sandbox activity, and facilitates their exploration:

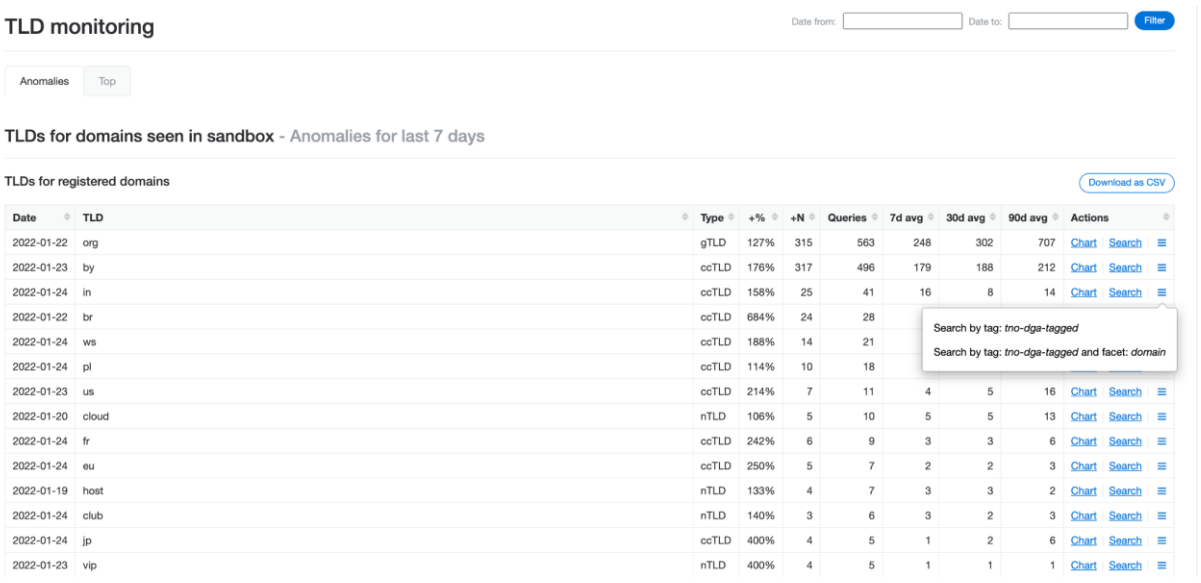
- TLD monitoring
- DoH/DoT usage monitoring
- Registrar usage monitoring

The dashboard is supported by daily e-mail reports of unregistered domains (including blockchain based domains) to analysts.

The following sections elaborate on the types of monitoring and detections. We plan to refine the hunting process and dashboards as the SOCCRATES pilots progress.

8.1 TLD monitoring

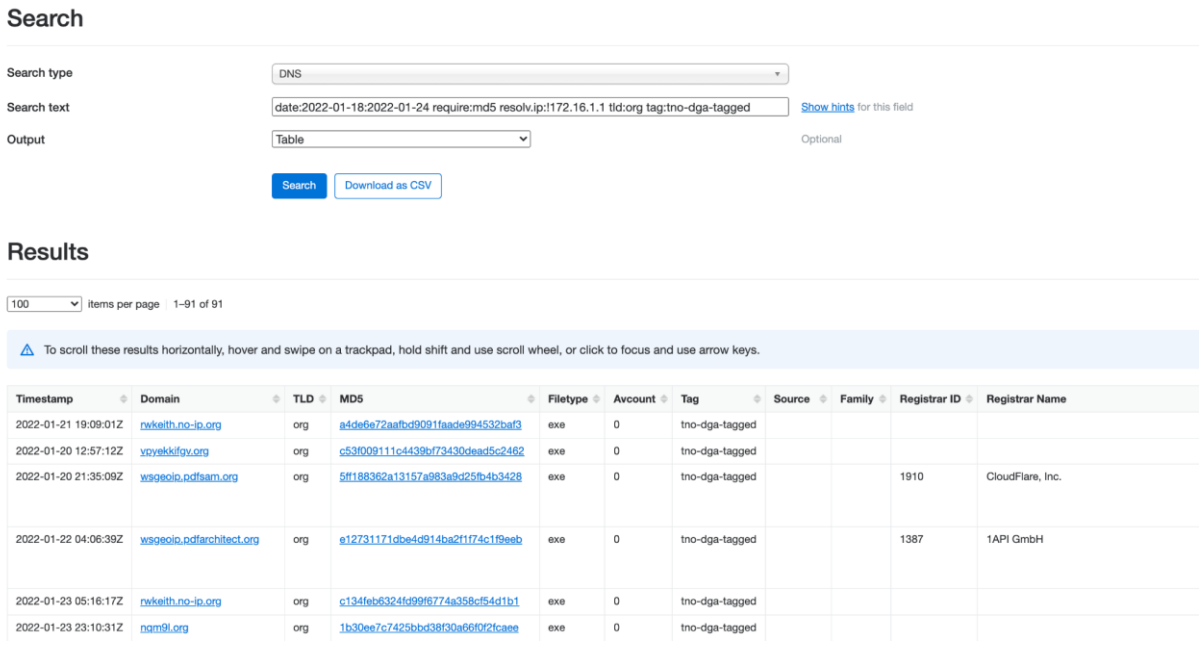
A 7 day moving average is calculated per TLD for the number of DNS queries attempted by malware for domains within that TLD. If a threshold exceeds the weekly average at least twice, an anomaly is signalled and displayed on the dashboard for investigation by an analyst.



Date	TLD	Type	+%	+N	Queries	7d avg	30d avg	90d avg	Actions
2022-01-22	org	gTLD	127%	315	563	248	302	707	Chart Search
2022-01-23	by	ccTLD	176%	317	496	179	188	212	Chart Search
2022-01-24	in	ccTLD	158%	25	41	16	8	14	Chart Search
2022-01-22	br	ccTLD	684%	24	28				
2022-01-24	ws	ccTLD	188%	14	21				
2022-01-24	pl	ccTLD	114%	10	18				
2022-01-23	us	ccTLD	214%	7	11	4	5	16	Chart Search
2022-01-20	cloud	nTLD	106%	5	10	5	5	13	Chart Search
2022-01-24	fr	ccTLD	242%	6	9	3	3	6	Chart Search
2022-01-24	eu	ccTLD	250%	5	7	2	2	3	Chart Search
2022-01-19	host	nTLD	133%	4	7	3	3	2	Chart Search
2022-01-24	club	nTLD	140%	3	6	3	2	3	Chart Search
2022-01-24	jp	ccTLD	400%	4	5	1	2	6	Chart Search
2022-01-23	vip	nTLD	400%	4	5	1	1	1	Chart Search

Figure 8-1 TLD Monitoring dashboard with anomalies and drill down options.

This monitoring is done in parallel for both registered and unregistered (NXDOMAIN) type DNS responses.



Search

Search type: DNS

Search text: date:2022-01-18:2022-01-24 require:md5 resolv.ip:!172.16.1.1 tld:org tag:tno-dga-tagged — Show hints for this field

Output: Table — Optional

Search | Download as CSV

Results

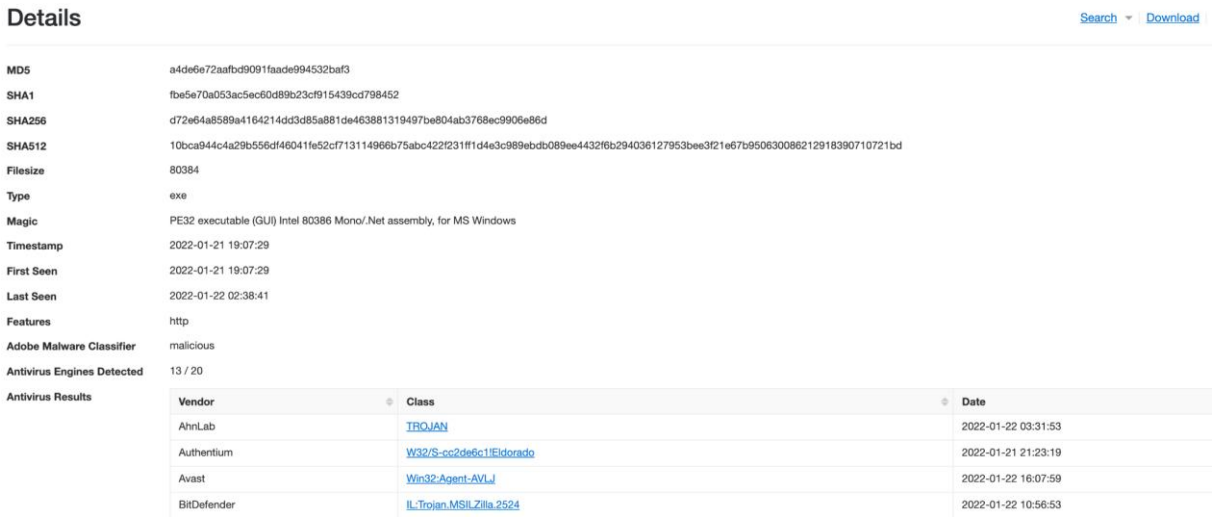
100 items per page 1–91 of 91

To scroll these results horizontally, hover and swipe on a trackpad, hold shift and use scroll wheel, or click to focus and use arrow keys.

Timestamp	Domain	TLD	MD5	Filetype	Avcount	Tag	Source	Family	Registrar ID	Registrar Name
2022-01-21 19:09:01Z	rwkeith.no-ip.org	org	a4de6e72aafbd9091faade994532baf3	exe	0	tno-dga-tagged				
2022-01-20 12:57:12Z	vpyekkifgv.org	org	c53f009111c4439bf73430dead5c2462	exe	0	tno-dga-tagged				
2022-01-20 21:35:09Z	wsgeoip.pdfsam.org	org	5ff188362a13157a983a9d25fb4b3428	exe	0	tno-dga-tagged			1910	CloudFlare, Inc.
2022-01-22 04:06:39Z	wsgeoip.pdfarchitect.org	org	e12731171dbe4d914ba2f1f74c1f9eeb	exe	0	tno-dga-tagged			1387	1API GmbH
2022-01-23 05:16:17Z	rwkeith.no-ip.org	org	c134feb6324fd99f6774a358cf54d1b1	exe	0	tno-dga-tagged				
2022-01-23 23:10:31Z	nqm9l.org	org	1b30ee7c7425bbd38f30a66f0f2fcaee	exe	0	tno-dga-tagged				

Figure 8-2 Drilling down into malware querying .org domains (search by tno-dga-tagged).

An analyst can perform various types of actions on the anomalies found. This includes drilling down into potential domains tagged as DGAs by the DGA Detective. For example, in the above case by clicking the “Search” function and selecting “Search by tno-dga-tagged” they get redirected to the actual DNS queries seen in the sandbox tagged by the DGA Detective, which provides additional information on the malware performing the DNS query, classification of the malware family, the registrar used for the domain registration etc.



Details — Search | Download

MD5	a4de6e72aafbd9091faade994532baf3
SHA1	fbe5e70a053ac5ec60d89b23cf915439cd798452
SHA256	d72e64a8589a4164214dd3d85a881de463881319497be804ab3768ec9906e86d
SHA512	10bca944c4a29b556df46041fe52cf713114966b75abc422f231ff1d4e3c989ebdb089ee4432f6b294036127953bee3f21e67b95063008621291839071072 1bd
Filesize	80384
Type	exe
Magic	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
Timestamp	2022-01-21 19:07:29
First Seen	2022-01-21 19:07:29
Last Seen	2022-01-22 02:38:41
Features	http
Adobe Malware Classifier	malicious
Antivirus Engines Detected	13 / 20

Antivirus Results

Vendor	Class	Date
AhnLab	TROJAN	2022-01-22 03:31:53
Authentium	W32/S-cc2de6c1!Eldorado	2022-01-21 21:23:19
Avast	Win32:Agent-AVLJ	2022-01-22 16:07:59
BitDefender	IL:Trojan.MSILZilla.2524	2022-01-22 10:56:53

Figure 8-3 Exploring details of the malware behind the DGA queries.

The malware itself can also be explored in more detail, including AV classifications of various vendors and additional sandbox results.

Trends in TLD usage can also be visualized, as in the case of .org domains above (clicking on chart in the corresponding dashboard entry above):

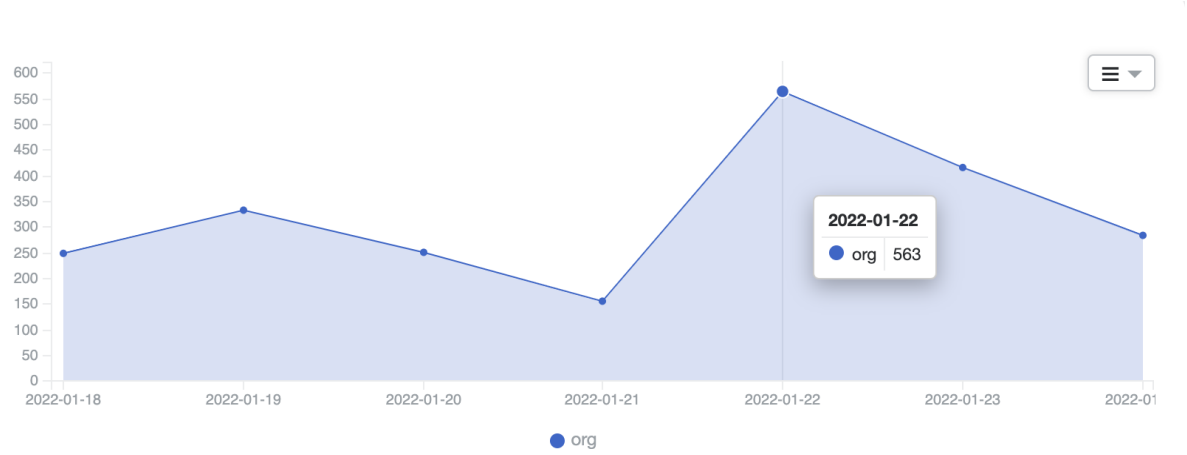


Figure 8-4 Chart showing queries to .org domains in the past 7 days.

As mentioned, similar features are also available for unregistered (NXDOMAIN) domains:

TLDs for non-existent domains

Download as CSV

Date	TLD	Type	+%	+N	Queries	7d avg	30d avg	90d avg	Actions
2022-01-24	org	gTLD	227%	96	139	43	66	133	Chart Search
2022-01-24	in	ccTLD	723%	92	105	13	60	62	Chart Search
2022-01-24	info	gTLD	104%	42	83	41	53	57	Chart Search
2022-01-24	biz	gTLD	415%	54	67	13	809	293	Chart Search
2022-01-24	ru	ccTLD	128%	22	40	18	20	124	Chart Search
2022-01-21	tw	ccTLD	168%	24	38	14	13	13	Chart Search
2022-01-24	us	ccTLD	1,185%	33	36	3	16	15	Chart Search
2022-01-22	br	ccTLD	900%	31	35	4	2	4	Chart Search
2022-01-22	ro	ccTLD	1,833%	27	29	2	1	1	Chart Search
2022-01-22	tr	ccTLD	766%	23	26	3	3	3	Chart Search
2022-01-23	su	ccTLD	199%	15	23	8	9	7	Chart Search
2022-01-23	id	ccTLD	148%	5	9	4	3	2	Chart Search
2022-01-18	pl	ccTLD	425%	6	7	1	1	4	Chart Search
2022-01-19	at	ccTLD	350%	5	6	1	3	1	Chart Search
2022-01-22	cz	ccTLD	400%	4	5	1	1	5	Chart Search
2022-01-23	pw	ccTLD	100%	2	4	2	3	29	Chart Search
2022-01-23	eu	ccTLD	100%	2	4	2	2	255	Chart Search
2022-01-24	nu	ccTLD	200%	2	3	1	2	1	Chart Search
2022-01-22	mx	ccTLD	200%	2	3	1	1	1	Chart Search
2022-01-20	fr	ccTLD	100%	1	3	2	2	4	Chart Search

Figure 8-5 TLD Monitoring dashboard with anomalies and drill down options (unregistered domains).

These can be explored in a similar manner as the registered domains.

8.2 DoH/DoT monitoring

A 7 day moving average for queries is calculated for each known DoH or DoT server. If a threshold exceeds the weekly average at least twice, an anomaly is signalled and displayed on the dashboard for investigation by the analyst.

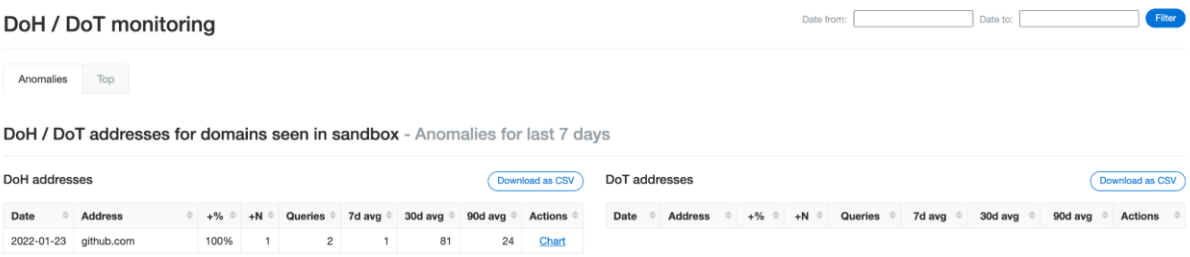


Figure 8-6 Dashboard showing DoH/DoT query anomalies for the last 7 days.

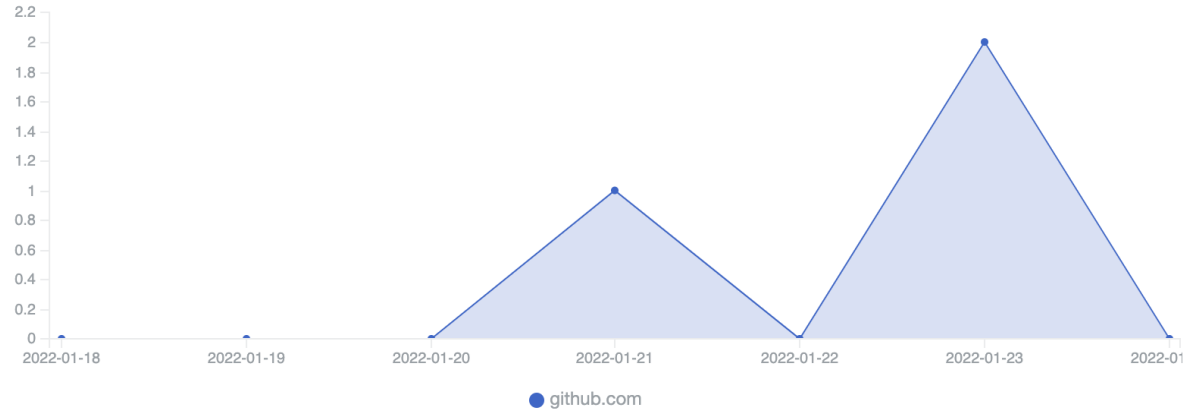


Figure 8-7 Chart showing an anomaly in queries to a DoH server at github.com.

We have also built a list of known DoH/DoT servers, which can be updated by an analyst.

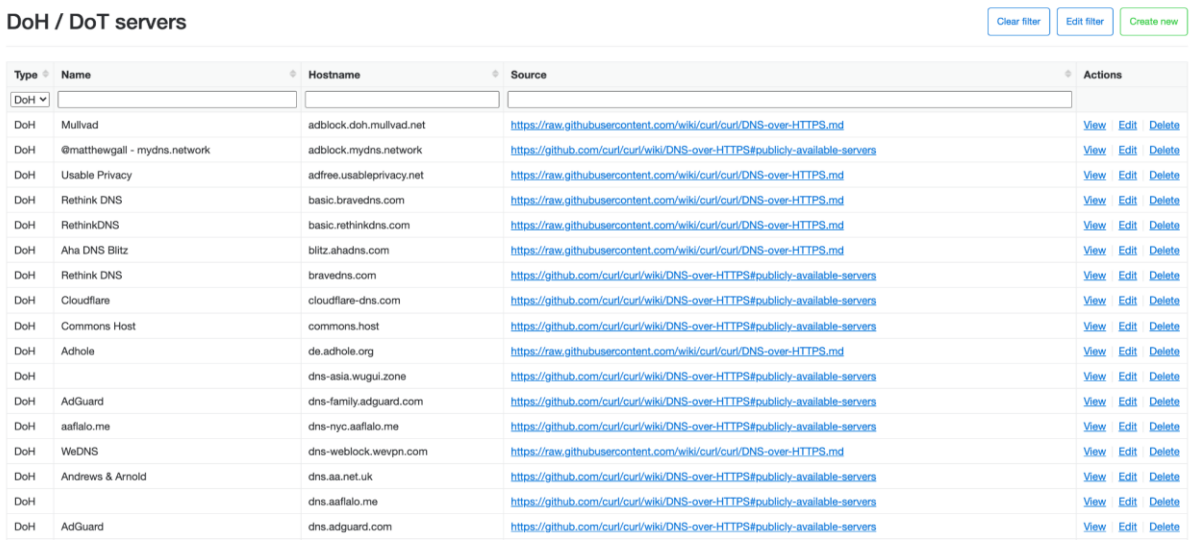


Figure 8-8 List of DoH/DoT servers.

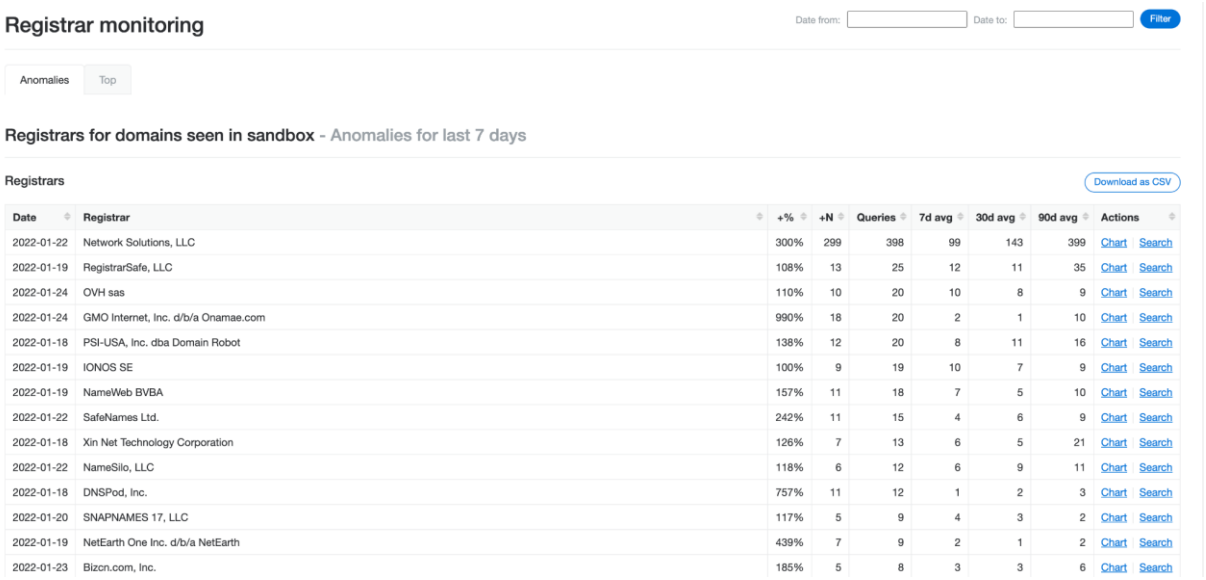
8.3 Registrar monitoring

One of the enhancements to sandbox data collected has been the enrichment of domain data with registrar information. This is done via Registration Data Access Protocol (RDAP) queries[17]. However,

[17] https://www.icann.org/rdap

as not all registries support RDAP, the lookups will be incomplete. Nevertheless thanks to the enrichment we are able to gain insight into what registrars are being used to register malicious domains and track the trends.

A 7 day moving average is calculated per each registrar over queries seen to domains registered through that registrar. If a threshold exceeds the weekly average at least twice, an anomaly is signalled and displayed on the dashboard for investigation by the analyst



Date	Registrar	+%	+N	Queries	7d avg	30d avg	90d avg	Actions
2022-01-22	Network Solutions, LLC	300%	299	398	99	143	399	Chart Search
2022-01-19	RegistrarSafe, LLC	108%	13	25	12	11	35	Chart Search
2022-01-24	OVH sas	110%	10	20	10	8	9	Chart Search
2022-01-24	GMO Internet, Inc. d/b/a Onamae.com	990%	18	20	2	1	10	Chart Search
2022-01-18	PSI-USA, Inc. dba Domain Robot	138%	12	20	8	11	16	Chart Search
2022-01-19	IONOS SE	100%	9	19	10	7	9	Chart Search
2022-01-19	NameWeb BVBA	157%	11	18	7	5	10	Chart Search
2022-01-22	SafeNames Ltd.	242%	11	15	4	6	9	Chart Search
2022-01-18	Xin Net Technology Corporation	126%	7	13	6	5	21	Chart Search
2022-01-22	NameSilo, LLC	118%	6	12	6	9	11	Chart Search
2022-01-18	DNSPod, Inc.	757%	11	12	1	2	3	Chart Search
2022-01-20	SNAPNAMES 17, LLC	117%	5	9	4	3	2	Chart Search
2022-01-19	NetEarth One Inc. d/b/a NetEarth	439%	7	9	2	1	2	Chart Search
2022-01-23	Bizcn.com, Inc.	185%	5	8	3	3	6	Chart Search

Figure 8-9 Anomalies associated with Registrars (last 7 days).

8.4 Daily unregistered domain e-mail reports

`md5` - hash of malware generating a query
`domain` - domain in question
`avcount` - number of antivirus engines that identified a signature for the malware
`age` - when the domain was first seen in the sandbox (in days)
`samples` - number of unique samples querying a domain
`dga` - DGA if any
`dga_to` - period DGA is valid to, if any
`dga_from` - period DGA is valid from, if any
`dga_rule` - DGA rule with seed, of any
`family` - Shadowserver internal AV classification of domain (typically YARA rule based)
`tags` - additional meta-information, for example, DGA Detective 'tno-dga-tagged'
`av_class` - AV classification results from various AV engines

2022-01-24-unregistered-domain-nodga

md5	domain	avcount	avtotal	age	samples	dga	dga_from	dga_to	dga_rule	family	tag	av_class
7782ef1bc4312b9f2957bedb9143e1f6	gweqgsczjfnwx.museum	0	20	0	1						tno-dga-tagged	
7782ef1bc4312b9f2957bedb9143e1f6	geetiwosqhe.museum	0	20	0	1						tno-dga-tagged	
7782ef1bc4312b9f2957bedb9143e1f6	ssnlsqq.st	0	20	0	1						tno-dga-tagged	
1131ac66c51cc2946dc0ed983892c543	server.ip-teleport.com	1	20	995	1					datper;whiskeybravo		
c9a310249d87198ec517df0e7a482a17	qgwkkeuyuq.mp	2	20	0	1						tno-dga-tagged	
b68e6bbddfc740f5ebdc25694ff3ef79	install.oinstaller5.com	2	20	2767	18							
abe03a49b237a2630ee9b092697fc613	host123.zz.am	2	20	2264	33							
647764e2d0edde99e1585d58bf04385b	bigzz.by	3	20	2694	1					pushdo;qsnatch;virut		
5c769c136e235fc703d9c1f174729910	newculturemediablog.com	3	20	2149	15					alphacrypt;teslacrypt		
647764e2d0edde99e1585d58bf04385b	amba-tc.si	3	20	2694	1					pushdo;qsnatch;virut		
647764e2d0edde99e1585d58bf04385b	polprime.com	3	20	2685	1					pushdo;qsnatch;virut		
647764e2d0edde99e1585d58bf04385b	ktenergo.ru	3	20	3323	1					pushdo;qsnatch;virut		
647764e2d0edde99e1585d58bf04385b	www.wkhk.net	3	20	3125	1					pushdo;qsnatch;virut		
647764e2d0edde99e1585d58bf04385b	eos-i.com	3	20	2694	1					pushdo;qsnatch;virut		
127988987c8eec71c80980355f9abf2f	picturebrought.net	4	20	2912	2					suppobox	tno-dga-tagged	
59f35cd31d49591846b5e476f90bafa2	hotcasinogames.org	4	20	2124	7					alphacrypt;teslacrypt		
7089ce13707f699b53adc304eb4b3a7d	dptejojbrbss.com	4	20	2386	3					tinba	tno-dga-tagged	
127988987c8eec71c80980355f9abf2f	familybrought.net	4	20	2912	2					suppobox		
127988987c8eec71c80980355f9abf2f	becauseoclock.net	4	20	2912	2					suppobox	tno-dga-tagged	
127988987c8eec71c80980355f9abf2f	cigarettewrite.net	4	20	2524	1					suppobox	tno-dga-tagged	
7089ce13707f699b53adc304eb4b3a7d	iyukfuxqgdps.com	4	20	2386	3					tinba	tno-dga-tagged	
7089ce13707f699b53adc304eb4b3a7d	dssldonncslo.com	4	20	2379	3					tinba	tno-dga-tagged	
127988987c8eec71c80980355f9abf2f	whetherwrite.net	4	20	2912	2					suppobox	tno-dga-tagged	
127988987c8eec71c80980355f9abf2f	familymethod.net	4	20	2912	2					suppobox	tno-dga-tagged	
7089ce13707f699b53adc304eb4b3a7d	urelgfeqjcls.com	4	20	2386	3					tinba	tno-dga-tagged	
7089ce13707f699b53adc304eb4b3a7d	ysrqibuwxpxx.com	4	20	2455	3					tinba	tno-dga-tagged	
127988987c8eec71c80980355f9abf2f	personniece.net	4	20	2912	2					suppobox		
7089ce13707f699b53adc304eb4b3a7d	trbdefedkjyn.com	4	20	2386	3					tinba	tno-dga-tagged	
7089ce13707f699b53adc304eb4b3a7d	lsdplonfiess.com	4	20	2386	3					tinba		

Figure 8-10 Unregistered domain list for with potential new DGA candidates (2022-01-24).

An analyst can choose to investigate such a summary daily and pick out potentially interesting domains, especially DGA candidates.

A similar summary is made for cryptocurrency based domains.

2022-01-22-cryptocurrency-domains

md5	domain	avcount	avtotal	age	samples	dga	dga_from	dga_to	dga_rule	family	tag	av_class
fe2bc7cbfa393ea974f30e6ec7329bb6	carder.bit	14	19		13					metamorfo		
b86bdb7973e220a14db74bc9546e583d	ransomware.bit	16	19		24					metamorfo		mcafee|packed-fgq!b86bdb7973e2

Figure 8-11 Unregistered domain list for cryptocurrency domains (2022-01-22).

9 The Shadowserver API

The Shadowserver API is described in detail in D2.3 System Architecture and Interface Specification – Final version. It enables the functionality listed below:

- Query for a domain name and provide a classification of whether or not the domain is likely to be used in a DGA, and optionally the likely associated malware family.
- Query for a malware hash and provide any associated DGA domains discovered through the AI based methods developed.
- Query for the latest DGA domains tagged by AI based methods developed (DGA Detective)
- Query for the latest DGA training set data for any anomaly detection tools developed
- Query for the latest TOP domain lists, which can also be used as a training set with (potentially) benign data for any anomaly detection
- Query for latest trends seen in DNS traffic, for example, by TLD, crypto domains or registrar.

9.1 Methods implemented

The above functionality is intended for the ACT TIP (T4.4) and Middleman tool, as well as external partners accessing the API for their own use-cases. It is implemented in the Shadowserver API[18] through the following new or extended methods:

1. **research/malware-domain**

   Returns a JSON response containing static details about matching samples as well as antivirus vendor and signature details.

2. **research/malware-info**

   Returns a JSON response containing static details about the requested sample as well as DNS and antivirus vendor and signature details. Domains that are detected as potential DGAs by the TNO classifier will have an additional tag present.

3. **research/tno-export**

   List all domains classified as DGAs by the TNO classifier (DGA Detective) from sandbox events in the past day. Optionally export as snort rules.

4. **research/rpz-export**

   Returns the list of active RPZ (Response Policy Zone) DGA domains valid for the next five days.

5. **research/site-ranking-export**

   Provides domain name ranking details compiled from a number of public sources.

6. **research/sandbox-dns-anomaly**
   Provides sandbox DNS alert details for one or more query types: crypto (DoH, DoT usage), registrar and TLD trend analysis.

[18] https://www.shadowserver.org/what-we-do/network-reporting/api-documentation/

9.2 Example data returned for DGA Detective findings

A user can query the research/tno-export method to discover DGA Detective classified domain queries in the Shadowserver Sandbox in the past day, as in this example using the `call-api.py` tool (published by Shadowserver)[19]:

```
./call-api.py research/tno-export '{"skip":["rpz-dga","noise","family"]}
```

where we filter out known DGAs, families and any noise.

Example output (truncated)[20]:

```
{"domain":"gaueudbuwdbuguuh.ws","tld":"ws"}
{"tld":"ws","domain":"eafueudzefverrgh.ws"}
{"domain":"neverwinwlaq.xyz","tld":"xyz"}
{"tld":"com","domain":"i.kpzip.com"}
{"domain":"oeeoeuueueuuepn.su","tld":"su"}
{"tld":"com","domain":"ngb3bw2iggx1egw.mxp4088.com"}
{"domain":"okqhu702okorsq92v.f28z3uv9t4.com","tld":"com"}
{"tld":"com","domain":"king.junyuewl.com"}
{"tld":"com","domain":"tj.kpzip.com"}
{"tld":"com","domain":"ryuduq.mxp4102.com"}
{"domain":"seuufhehfueugheh.ws","tld":"ws"}
{"tld":"ws","domain":"okdoekeoehghaoeh.ws"}
{"domain":"feauhueudughuurt.to","tld":"to"}
{"tld":"to","domain":"aauaaaeieiieeao.to"}
{"domain":"www.issuejeju.com","tld":"com"}
{"domain":"eaffuebudbeudbbh.ws","tld":"ws"}
{"domain":"uokwa.agesonest.com","tld":"com"}
{"tld":"com","domain":"lokias111234.blog.163.com"}
{"domain":"n.jntbxduhz.ru","tld":"ru"}
{"tld":"org","domain":"5ipygnlfou.nnnn.eu.org"}
{"tld":"com","domain":"txcdn.djduoduo.com"}
{"domain":"wme9ihgpd9.tkp55h1dgp.com","tld":"com"}
```

More examples of methods used can be found in the D2.3 System Architecture and Interface Specification – Final version API specification.

[19] https://www.shadowserver.org/wp-content/uploads/2021/03/call-api.py_.zip

[20] We are also working on an FP list maintenance process to eliminate obvious domains that should not have appeared

10 The Middleman

Section 6 describes DGA Detective, a novel software developed for SOCCRATES to detect DGA domains at scale. DGA Detective runs at Shadowserver, where it analyses domain names from its large data infrastructure. Shadowserver then provides the result of this analysis through the API it has developed for SOCCRATES (see section 9). The AAD (see D4.1 AI-based Detection and Prediction and D2.3 System Architecture and Interrface Specification – Final version) leverages these results to improve its anomaly detection capabilities through a novel component, the Middleman.

The Middleman has been designed to fulfil the specific task of interacting with the Shadowserver API to improve detection of DGA domains within the AAD. The Middleman regularly queries the endpoint `research/tno-exports`, which returns the list of domains labelled as DGA by DGA Detective at Shadowserver in the past 24 hours. It ingests domain names and it checks if any of them is in the list retrieved from Shadowserver. In soccrates the middleman ingests the domain names observed queried by hosts in the test infrastructure. If there is a match, it sends an alert as a message to the Middleman Kafka topic, which is then later consumed by REASENS (described in D4.1 AI-based Attack Detection to Detect Advanced Threats).

The Middleman combines the benefits of running DGA Detective at Shadowserver to the already existing DGA detection provided by DNS Ninja. DNS Ninja's statistical detection, which is unsupervised (it is based on statistical distribution of characters in the domain string) works well on average, however it lacks the possibility of fine tuning detection based on recent examples from real malicious traffic. This is provided by DGA Detective.

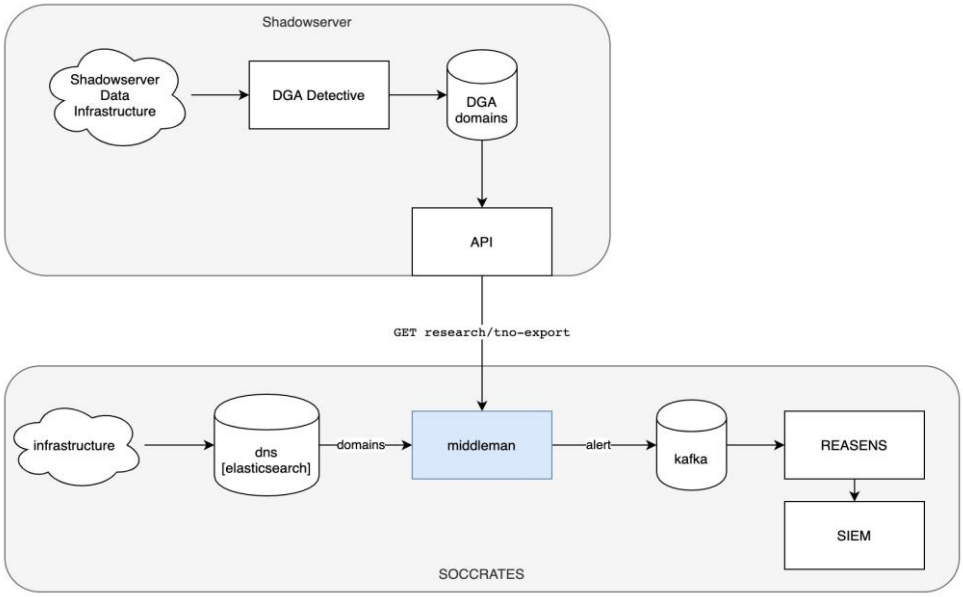


Figure 10-1 The role of the new "Middleman" within SOCCRATES.

11 Findings and Contributions to the Community

Since the start of 2021, DGA Detective has been tested operationally in Shadowserver's production environment, with additional Dashboard functionality added to support improved Threat Hunting mid-2021.

This has meant that the pilot at Shadowserver has been operational before the formal 2nd SOCCRATES Pilot phase, which officially starts in February 2022.

This lengthy period of production (TLR9) level evaluation and testing has resulted in a number of contributions to the research and operational communities worldwide:

- Contributions to the DGA Archive[21] operated by Fraunhofer FKIE. This is the leading source of information on DGA domains generated by malware families. It contains past and future DGA domains generated per family. It is the premier source of DGA-related data for both researchers and operational teams such as CSIRTs, aiding in detection, analysis and incident response. There have already been a number of contributions from SOCCRATES to the DGA archive. This includes the identification of new DGAs for *phorpiex* and *mydoom* threats, and also *m0yv/expiro* (which was also discovered and reported independently by other parties). Based on the information provided and malware attribution, FKIE was able to reverse engineer the DGA algorithms and add them to the DGA Archive. Other DGA candidates will also be reported and we expect more additions in the future. Thanks to these actions, many organizations and their users worldwide receive extra protection from DGA-based threats.
- Sinkholing of threats, such as *Phorpiex*, *M0yv* and *Neksminer* and sharing data on infected machines with the Internet Defender community. By analysing the domain data in their sandboxes with the support of tools developed during the SOCCRATES project, Shadowserver was able to sample sinkhole a number of threats. This was achieved by submitting them to the Registrar of Last Resort (RoLR)[22], which is a special purpose DNS registrar created and supported by Shadowserver that enables free or discounted malicious domain registrations in order to block them from use by criminals. Thanks to this, data collected about infected hosts connecting to these DGA malware C2 domains is shared for free with 131 National CSIRTs and 6000+ other organisations worldwide via Shadowserver's daily threat feeds.
- Sharing of DGA DNS-RPZ[23] data. Shadowserver currently has a DGA DNS-RPZ project in a pilot phase where it shares all of it's DGA data with selected partners, for the purpose of blocking DGA domains at the ISP level. By having new DGAs reverse engineered and future potential DGA domains forward predicted, such as those detected through the SOCCRATES project, Internet networks can receive protection - i.e. infected computers on an ISP's network will not be able to connect to malicious C2 services. SOCCRATES therefore improves the quality of our Shadowserver's free public benefit services.

[21] https://dgarchive.caad.fkie.fraunhofer.de/welcome/
[22] Registrar of Last Resort (ROLR) https://www.rolr.eu/
[23] Response Policy Zones, https://dnsrpz.info/

In the following subsections we provide a short overview of some example sinkholing results obtained through sinkholing some of the threats detected by our SOCCRATES project work. We expect to provide further examples and elaboration further in future pilot and exploitation deliverables.

11.1 Phorpiex example sinkholing results

Phorpiex[24] is a botnet that has been used for the distribution of ransomware, malware downloaders and cryptominers. It has been active for many years and undergoes constant evolution.

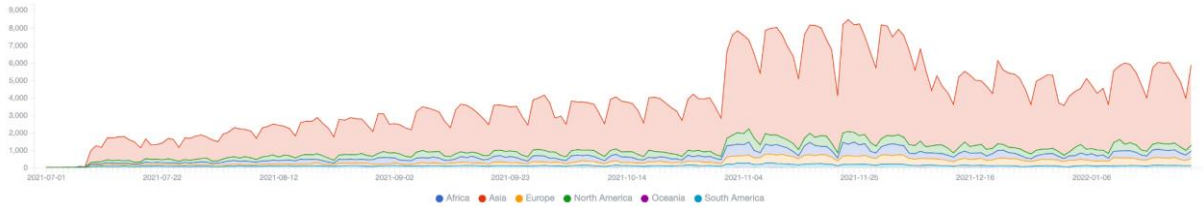


Figure 11-1 Sinkhole infections over time on Phorpiex DGA domains.

As can be seen in the chart above, the peak infection rate (i.e. when the most infected unique IP addresses were seen connecting to the sinkhole in a single day) was 8,454 unique IP addresses worldwide on 2021-11-23.

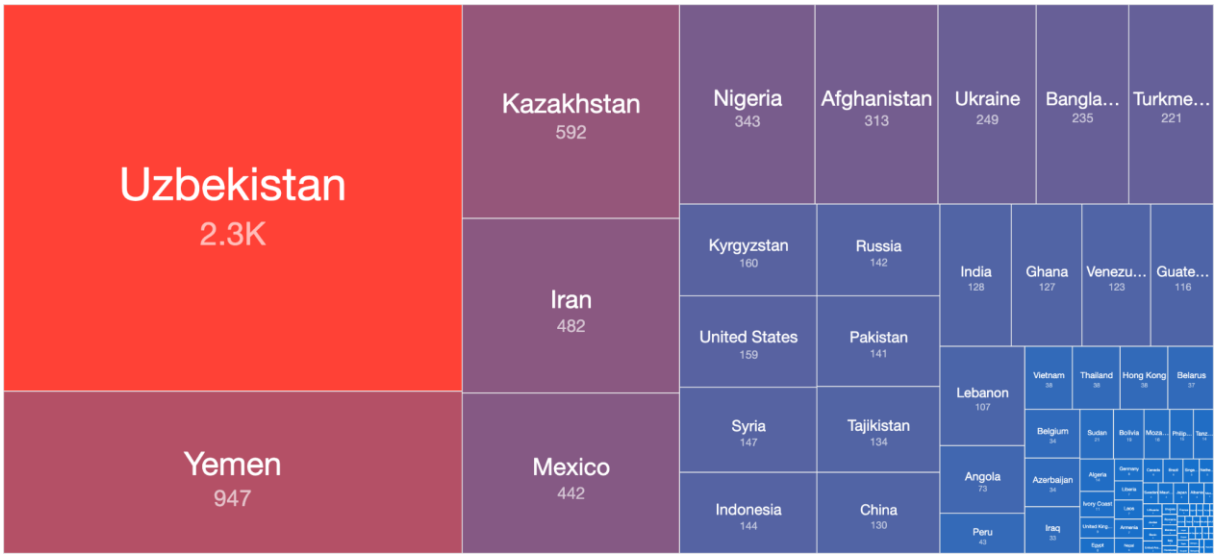


Figure 11-2 Geographical distribution of Phorpiex infections on 2021-11-23.

Most of the IP addresses affected by the malware are in Asia, particularly in Uzbekistan.

The data is obtained from the following sinkholed DGA DNS domains controlled by Shadowserver:

```
wduufbaueeubffgt.to
eafueudzefverrgt.to
okdoekeoehghaoet.to
```

[24] https://malpedia.caad.fkie.fraunhofer.de/details/win.phorpiex

```
fheuhdwdzwgzdggt.to
seuufhehfueughet.to
aeufuaehfiuehfuhfr.su
iugouehoeohfh.com
daedagheauehfuuhfl.co
gaghpaheiafhjefijs.top
gaoehuoaoefhuhfugu.cc
aegohaohuoruitiier.su
```

11.2 Neksminer example sinkoling results

Neksminer[25] is a cryptocurrency mining threat that is constantly evolving.

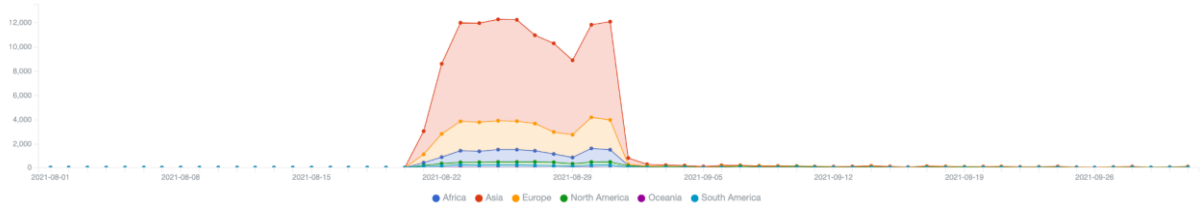


Figure 11-3 Sinkhole connections over time for sinkholed Neksminer domains.

As can be seen in the above chart, the peak infection rate (i.e. when the most infected unique IP addresses were seen connecting to the sinkhole in a single day) was 12,239 unique IP addresses worldwide on 2021-08-25. The sinkhole abruptly stopped receiving connections shortly after this date, suggesting that the attackers managed to reconfigure the malware to connect to other C2 domains.

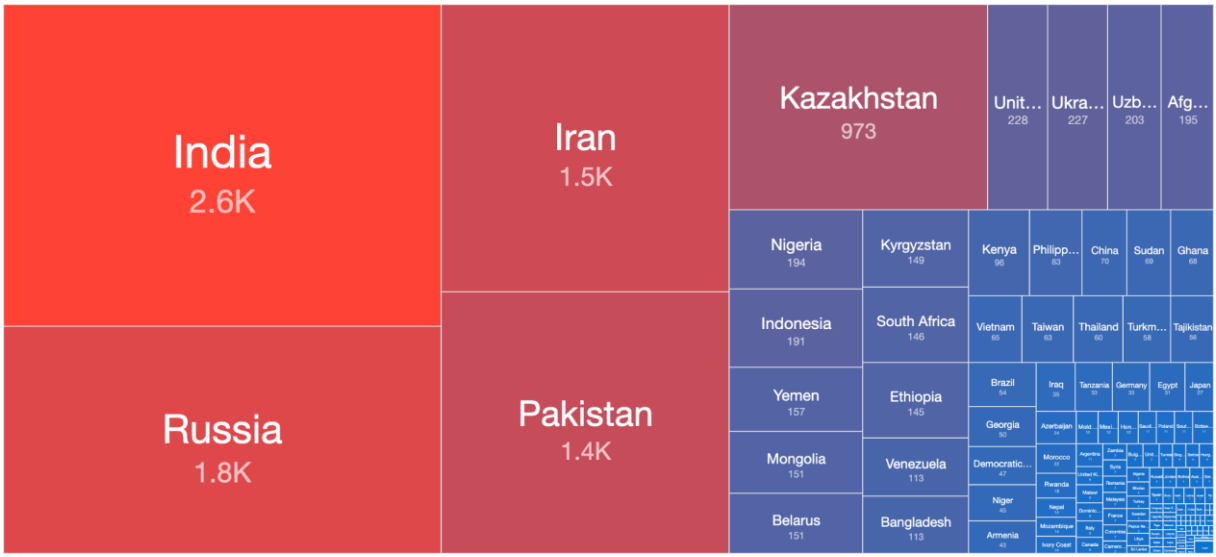


Figure 11-4 Geographical distribution of Neksminer infections on 2021-08-25.

As we can see in the chart above, most of the infected unique IP addresses are again located in Asia, but also in Russia and Africa too.

[25] https://www.trendmicro.com/vinfo/us/threat-encyclopedia/malware/Worm.Win32.MALXMR.B/

The data is obtained from the following sinkholed DGA DNS domains controlled by Shadowserver:

```
48a8fa5183.pw
ae2be1ffad.pw
13fffae465.pw
41d1fc7766.pw
cbbf4ddbb5.pw
923944c68c.pw
```

11.3 M0yv/Expiro example sinkholing results

M0yv/Expiro[26] is a polymorphic malware family of file infectors.

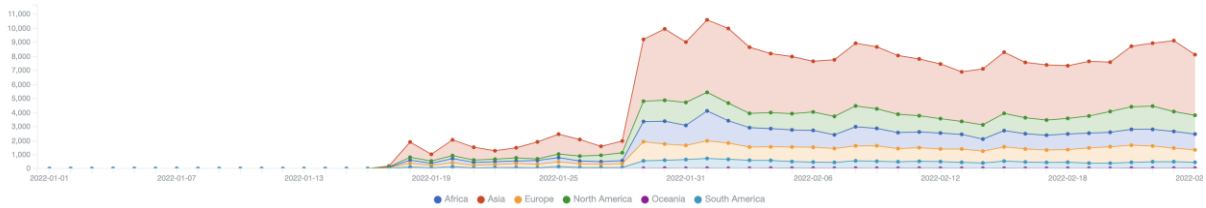


Figure 11-5 Sinkhole connections over time on M0yv/Expiro domains.

As can be seen in the above chart, the peak infection rate (i.e. when the most infected unique IP addresses were seen connecting to the sinkhole in a single day) was 10,589 unique IP addresses worldwide on 2022-02-01.

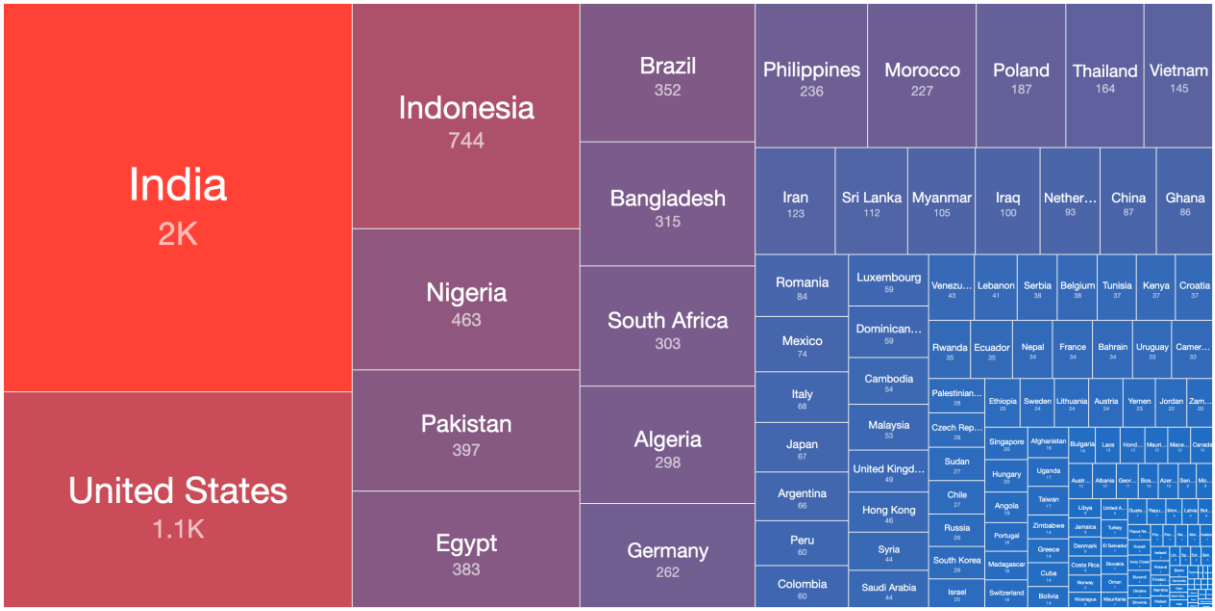


Figure 11-6 Sinkhole connections over time on M0yv/Expiro domains.

The worldwide distribution of M0yv/Expiro infections is more balanced, but Asia and Africa are dominant.

[26] https://www.sophos.com/en-us/threat-center/threat-analyses/viruses-and-spyware/W32~Expiro-H/detailed-analysis#:~:text=Expiro%20is%20a%20family%20of,well%20as%20steal%20login%20credentials.

The data is obtained from the following sinkholed DGA DNS domains controlled by Shadowserver:

```
zlenh.biz
uhxqin.biz
knjghuig.biz
anpmnmxo.biz
ifsaia.biz
lpuegx.biz
pywolwnvd.biz
```

12 Future Directions

For future directions, we envision exploration of the following approaches to detecting DGAs:

- Feature-based approach for family classification - scalability: the computational expense of the string distance matrix calculation significantly impacts the scalability of the methodology. A further step would be the evaluation of other ways to calculate it (e.g. https://github.com/Bergvca/string_grouper based on TF-IDF) to lower the memory usage and the computation time.
- Feature-based approach for family classification - underrepresented families: the results obtained to date are strongly impacted by the lack of training samples for some malware families. Training a Few Shots Learning classifier, a technique which is less dependent on large training sets, could be useful to overcome this problem.
- Deep learning approaches for domain name classification: the classification performed by the proposed TCNs is currently based solely on an encoding of the domain names, which makes the result difficult to interpret. To make this method more explainable, a parallel neural network that processes pre-computed features could be introduced. This could also lead to a more accurate domain classification.
- Generalisation capabilities of featureless approach to family classification: the TCN-based method can, in principle, distinguish domain names among the different families present in the training set. However, in practice, the approach should also be able to identify new, previously unseen, DGAs. To this end, we could define a confidence threshold for each malware family. If a domain name does not achieve the threshold for any of the classes, then it should be labelled as a new, unseen, DGA sample.
- DGA Detective will likely become less precise as new DGAs are developed, unless it is re-trained frequently. Currently this process is manual. In future, DGA Detective could be updated to automatically retrain itself with the latest data (such as training sets from Shadowserver's production sandboxing of malware at scale).

We also expect refinements to Shadowserver's internal dashboard to improve threat hunting for DGAs and other suspect domains, to try to automate the discovery process as much as possible.

13 Conclusion

The work performed in T4.2 of SOCCRATES WP4 has resulted in a prototype system that can be used to detect and hunt for DGA threats in production (at TRL9). This system couples practical machine learning methods and tools developed as part of the project with an anomaly/trend detection component to identify and classify new threats. While the scale of data that is handled means that the system cannot be fully automated (false positives are possible), the system is effective as an aid to threat analysts in their work. The knowledge derived can be used to predict which DNS domains will be used for malicious activity, once the DGA algorithms are reversed and understood - which is done through collaboration with external partner entities such as Fraunhofer FKIE, the maintainer of the DGA Archive.

The system developed has already been validated in the Shadowserver production environment and has been proven as an aid to threat analysts in their work on understanding DNS flows and sandboxed malware activity. As explained in this deliverable, the successful outputs have delivered real world operational results at scale and improved situational awareness of global cyber security threats.

As part of the upcoming pilot activities, we will continue to refine the system, build improved processes for threat hunting and handling false positives, support SOCCRATES platform demonstrations and use cases, as well as aiming to achieve more real world results that can be included as examples in the final project deliverables.

14 References

[1] Cucchiarelli, Alessandro, Christian Morbidoni, Luca Spalazzi and Marco Baldi. "Algorithmically generated malicious domain names detection based on n-grams features." *Expert Syst. Appl.* 170 (2021): 114551.

[2] Li, Yi, Kaiqi Xiong, Tommy Chin and Chengbin Hu. "A Machine Learning Framework for Domain Generation Algorithm-Based Malware Detection." *IEEE Access* 7 (2019): 32765-32782.

[3] Schüppen, Samuel, Dominik Teubert, Patrick Herrmann and Ulrike Meyer. "FANCI : Feature-based Automated NXDomain Classification and Intelligence." *USENIX Security Symposium* (2018).

[4] Sivaguru, Raaghavi, Chhaya Choudhary, Bin Yu, Vadym Tymchenko, Anderson Clayton Alves Nascimento and Martine De Cock. "An Evaluation of DGA Classifiers." 2018 IEEE International Conference on Big Data (Big Data) (2018): 5058-5067.

[5] Schiavoni, S., Maggi, F., Cavallaro, L., & Zanero, S. (2014, July). Phoenix: DGA-based botnet tracking and intelligence. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 192-211). Springer, Cham.

[6] Bai, Shaojie, J. Zico Kolter and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling." *ArXiv* abs/1803.01271 (2018).

[7] Oord, Aäron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior and Koray Kavukcuoglu. "WaveNet: A Generative Model for Raw Audio." *ArXiv* abs/1609.03499 (2016).

[8] Drichel, Arthur, Ulrike Meyer, Samuel Schüppen and Dominik Teubert. "Analyzing the real-world applicability of DGA classifiers." *Proceedings of the 15th International Conference on Availability, Reliability and Security* (2020).

15 Abbreviations

This glossary serves as inventory of abbreviations used in the document.
This is a standard glossary, used for all SOCCRATES report deliverables; it will be expanded when necessary.

Acronym	Description
AAD	AI-based Attack Detection
ACT	semi-Automated Cyber Threat intelligence
ADG	Attack Defence Graph
AEF	Argus Event Format
AI	Artificial Intelligence
AIT	AIT Austrian Institute of technology
API	Application Programming Interface
APT	Advance Persistent Threat
ATOS	ATOS Spain
AV	AntiVirus
BPMN	Business Process Model and Notation
CC	Command and Control
CERT	Computer Emergency Response Team
CMDB	Configuration Management Database
CSIRT	Computer Security Incident Response Team
CoA	Course of Action
CTI	Cyber Threat Intelligence
DC	DataCentre
DGA	Domain Generated Algorithm
DNS	Domain Name System
EDR	Endpoint Detection and Response
ELK	Elasticsearch/Logstash/Kibana
FRS	Foreseeti
FSC	F-secure
ICT	Information and Communication Technology
IDS	Intrusion Detection System
IMC	Infrastructure Modelling Component
IMT	Institut Mines Télécom – Télécom SudParis
INTF	Interface
IoC	Indicators of Compromise
IP	Internet Protocol
IPS	Intrusion Prevention System
IRM	Incident Response and Management
ITIL	Information Technology Infrastructure Library
KTH	Kungliga Tekniska högskolan - Royal Institute of Technology
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
M_n	Infrastructure Model (at time *n*)
MNM	Mnemonic

MSSP	Managed Security Service Provider
MTTD	Mean Time To Detection
NOC	Network Operations Centre
OT	Operational Technology
OS	Operating System
RORI	Return on Response Investment
RPZ	Response Policy Zones
SDN	Software Defined Network
SHS	Shadowserver
SIEM	Security information and event management
SOAR	Security Orchestration, Automation and Response
SOC	Security Operation Centre
SOCCRATES	SOC & CSIRT Response to Attacks & Threats based on attack defence graph Evaluation Systems
SSL	Secure Sockets Layer
TAP	Test Access Point
TI	Thrreat Intelligence
TIP	Threat Intelligence platform
TLS	Transport Layer Security
TNO	Nederlandse Organisatie voor toegepast natuurwetenschappelijk onderzoek
TTC	Time To Compromise
UC	Use Case
VLAN	Virtual LAN
VM	Virtual Machine
VTF	Vattenfall